KV4x Reference Manual

Supports MKV46F256VLx16, MKV46F128VLx16, MKV44F256VLx16, MKV44F128VLx16, MKV44F64VLx16, MKV42F256VLx16, MKV42F128VLx16

Document Number: KV4XP100M168RM
Rev. 5, 08/2019

Contents

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

Chapter 1
About This Document

Chapter 2
Introduction

Chapter 3
Core overview

Chapter 4
Memories and Memory Interfaces

Chapter 5
Memory Map

Chapter 6
Clock Distribution

Chapter 7
Power Management

Chapter 8
Security

Chapter 9
Debug

Chapter 10
Reset and Boot

Chapter 11
Signal Multiplexing

Chapter 12
Port control and interrupts (PORT)

Chapter 13
System Integration Module (SIM)

Chapter 14
Kinetis Flashloader

Chapter 15
Reset Control Module (RCM)

Chapter 16

System Mode Controller (SMC)

Chapter 17

Miscellaneous Control Module (MCM)

Chapter 18
Power Management Controller (PMC)

Chapter 19
Low-Leakage Wakeup Unit (LLWU)

Chapter 20
Crossbar Switch Lite (AXBS-Lite)

Chapter 21
Peripheral Bridge (AIPS-Lite)

Chapter 22
Direct memory access multiplexer (DMAMUX)

Chapter 23
Direct Memory Access Controller (eDMA)

Chapter 24
External Watchdog Monitor (EWM)

Chapter 25
Watchdog Timer (WDOG)

Chapter 26
Inter-Peripheral Crossbar Switch A (XBARA)

Chapter 27
Inter-Peripheral Crossbar Switch B (XBARB)

Chapter 28
Crossbar AND/OR/INVERT (AOI) Module

Chapter 29
Oscillator (OSC)

Chapter 30
Multipurpose Clock Generator (MCG)

Chapter 31
Flash Memory Controller (FMC)

Chapter 32
Flash Memory Module (FTFA)

Chapter 33
Cyclic redundancy check (CRC)

Chapter 34
12-bit Cyclic Analog-to-Digital Converter (ADC)

Chapter 35
Comparator (CMP)

Chapter 36
12-bit Digital-to-Analog Converter (DAC)

Chapter 37
Pulse Width Modulator A (PWMA/eFlexPWM)

Chapter 38
Programmable Delay Block (PDB)

Chapter 39
FlexTimer Module (FTM)

Chapter 40
Periodic Interrupt Timer (PIT)

Chapter 41
Quadrature Encoder/Decoder (ENC)

Chapter 42
Low-Power Timer (LPTMR)

Chapter 43
Flex Controller Area Network (FlexCAN)

Chapter 44
Serial Peripheral Interface (SPI)

Chapter 45
Inter-Integrated Circuit (I2C)

Chapter 46
Universal Asynchronous Receiver/Transmitter (UART) / FlexSCI

Chapter 47
General-Purpose Input/Output (GPIO)

Chapter 48
JTAG Controller (JTAGC)

Chapter 1
About This Document

1.1 Overview

1.1.1 Purpose

This document describes the features, architecture, and programming model of the KV4x family of microcontrollers.

1.1.2 Audience

This document is primarily for system architects and software application developers who are using or considering using the microcontroller in a system.

1.2 Conventions

1.2.1 Numbering systems

The following suffixes identify different numbering systems:

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is written 101b. In some cases, binary numbers are shown with the prefix *0b*.
d	Decimal number. Decimal numbers are followed by this suffix only when the possibility of confusion exists. In general, decimal numbers are shown without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is written 3Ch. In some cases, hexadecimal numbers are shown with the prefix *0x*.

1.2.2 Typographic notation

The following typographic notation is used throughout this document:

Example	Description
placeholder, x	Items in italics are placeholders for information that you provide. Italicized text is also used for the titles of publications and for emphasis. Plain lowercase letters are also used as placeholders for single letters and numbers.
`code`	Fixed-width type indicates text that must be typed exactly as shown. It is used for instruction mnemonics, directives, symbols, subcommands, parameters, and operators. Fixed-width type is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.
SR[SCM]	A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets and separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus.

1.2.3 Special terms

The following terms have special meanings:

Term	Meaning
asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
reserved	Refers to a memory space, register, or field that is either reserved for future use or for which, when written to, the module or chip behavior is unpredictable.

Chapter 2
Introduction

2.1 Overview

This chapter provides an overview of the KV4x product family of Arm® Cortex®-M4 MCUs. It also presents high-level descriptions of the modules available on the devices covered by this document.

2.2 Module Functional Categories

The modules on this device are grouped into functional categories. The following sections describe the modules assigned to each category in more detail.

Table 2-1. Module functional categories

Module category	Description
Arm® Cortex®-M4 core	• 32-bit MCU core from Arm's Cortex-M class adding DSP instructions and single-precision floating point unit, based on Armv7-M architecture, up to 168 MHz operating frequency
System	• System integration module (SIM) • Power management and mode controllers (PMC) • Multiple power modes available based on run, wait, stop, and power-down modes • Low-leakage wakeup unit (LLWU) • Miscellaneous control module (MCM) • Crossbar switch • Peripheral bridge (AIPS) • Direct memory access (DMA) controller with multiplexer to increase available DMA requests. • External watchdog monitor (EWM) • Watchdog (WDOG)

Memories	• Internal memories include:
	  • Up to 256 KB flash memory
	  • Up to 32 KB SRAM

Clocks	• Multiple clock generation options available from internally- and externally-generated clocks • System oscillator to provide clock source for the MCU

Table continues on the next page...

Table 2-1. Module functional categories (continued)

Module category	Description
Security	• Cyclic Redundancy Check module for error detection
Analog	• Two 12-bit analog-to-digital converters (ADC) with 240 ns conversion time • Four high speed comparators (CMP) • Digital-to-analog converter (DAC)
Timers	• Two Programmable Delay Blocks (PDB) • Three FlexTimers (FTM) • Pulse Width Modulator (eFlexPWM) • Periodic interrupt timer (PIT) • Quadrature Encoder/Decoder (ENC) • Low-Power Timer (LPTMR)
Communications	• Two Flex Controller Area Network (FlexCAN) • Serial Peripheral Interface (SPI) • Inter-integrated circuit (I^2C) • Two Universal Asynchronous Receiver/Transmitters (UART)
Human-Machine Interfaces (HMI)	• General purpose input/output controller

2.2.1 Arm® Cortex®-M4 Core Modules

The following core modules are available on this device.

Table 2-2. Core modules

Module	Description
Arm Cortex-M4	The Arm® Cortex®-M4 is the newest member of the Cortex M Series of processors targeting microcontroller cores focused on very cost sensitive, deterministic, interrupt driven environments. The Cortex M4 processor is based on the Armv7 Architecture and Thumb®-2 ISA and is upward compatible with the Cortex M3, Cortex M1, and Cortex M0 architectures. Cortex M4 improvements include an Armv7 Thumb-2 DSP (ported from the Armv7-A/R profile architectures) providing 32-bit instructions with SIMD (single instruction multiple data) DSP style multiply-accumulates and saturating arithmetic.
Floating Point Unit (FPU)	A single-precision floating point unit (FPU) that is compliant to the *IEEE Standard for Floating-Point Arithmetic* (IEEE 754).
NVIC	The Armv7-M exception model and nested-vectored interrupt controller (NVIC) implement a relocatable vector table supporting many external interrupts, a single non-maskable interrupt (NMI), and priority levels. The NVIC replaces shadow registers with equivalent system and simplified programmability. The NVIC contains the address of the function to execute for a particular handler. The address is fetched via the instruction port allowing parallel register stacking and look-up. The first sixteen entries are allocated to Arm internal sources with the others mapping to MCU-defined interrupts.
AWIC	The primary function of the Asynchronous Wake-up Interrupt Controller (AWIC) is to detect asynchronous wake-up events in stop modes and signal to clock control logic to resume system clocking. After clock restart, the NVIC observes the pending interrupt and performs the normal interrupt or event processing.

Table continues on the next page...

Table 2-2. Core modules (continued)

Module	Description
Debug interfaces	Most of this device's debug is based on the Arm CoreSight™ architecture. Four debug interface is supported: • JTAG Controller (JTAG) • IEEE 1149.7 JTAG (cJTAG) • Serial Wire Debug (SWD) • Arm Real-Time Trace Interface

2.2.2 System Modules

The following system modules are available on this device.

Table 2-3. System modules

Module	Description
System integration module (SIM)	The SIM includes integration logic and several module configuration settings.
System mode controller (SMC)	The SMC provides control and protection on entry and exit to each power mode, control for the Power management controller (PMC), and reset entry and exit for the complete MCU.
Power management controller (PMC)	The PMC provides the user with multiple power options. Ten different modes are supported that allow the user to optimize power consumption for the level of functionality needed. Includes power-on-reset (POR) and integrated low voltage detect (LVD) with reset (brownout) capability and selectable LVD trip points.
Low-leakage wakeup unit (LLWU)	The LLWU module allows the device to wake from low leakage power modes (VLLSx) through various internal peripheral and external pin sources.
Miscellaneous control module (MCM)	The MCM includes integration logic and embedded trace buffer details.
Crossbar switch (AXBS)	The XBS connects bus masters and bus slaves, allowing all bus masters to access different bus slaves simultaneously and providing arbitration among the bus masters when they access the same slave.
Peripheral bridges	The peripheral bridge converts the crossbar switch interface to an interface to access a majority of peripherals on the device.
DMA multiplexer (DMAMUX)	The DMA multiplexer selects from many DMA requests down to a smaller number for the DMA controller.
Direct memory access (DMA) controller	The DMA controller provides programmable channels with transfer control descriptors for data movement via dual-address transfers for 8-, 16-, 32-, 128-, and 256-bit data values.
External watchdog monitor (EWM)	The EWM is a redundant mechanism to the software watchdog module that monitors both internal and external system operation for fail conditions.
Software watchdog (WDOG)	The WDOG monitors internal system operation and forces a reset in case of failure. It can run from an independent 1 KHz low power oscillator with a programmable refresh window to detect deviations in program flow or system frequency.
Inter-Peripheral Crossbar (XBARA/ XBARB) and Crossbar AND/OR/INVERT (AOI) Module	Provides generalized connections between and among on-chip peripherals: Cyclic ADC, 12-bit DAC, Comparators, Timers, eFlexPWMs, PDBs, EWM, Quadrature Decoder, and select I/O pins. AND-OR-INVERT function that provides a universal Boolean function generator using a four-term sum-of-products expression, with

Table 2-3. System modules

Module	Description
	each product term containing true or complement values of the four selected inputs (A, B, C, D).

2.2.3 Memories and Memory Interfaces

The following memories and memory interfaces are available on this device.

Table 2-4. Memories and memory interfaces

Module	Description
Flash memory	Program flash memory — non-volatile flash memory that can execute program code.
Flash memory controller	Manages the interface between the device and the on-chip flash memory.
SRAM	Internal system RAM.

2.2.4 Clocks

The following clock modules are available on this device.

Table 2-5. Clock modules

Module	Description
Multi-clock generator (MCG)	The MCG provides several clock sources for the MCU that include: • Phase-locked loop (PLL) — Voltage-controlled oscillator (VCO) • Frequency-locked loop (FLL) — Digitally-controlled oscillator (DCO) • Internal reference clocks — Can be used as a clock source for other on-chip peripherals
System oscillator (OSC)	The system oscillator, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

2.2.5 Security and Integrity modules

The following security and integrity module is available on this device:

Table 2-6. Security and integrity modules

Module	Description
Cyclic Redundancy Check (CRC)	Hardware CRC generator circuit using 16/32-bit shift register. Error detection for all single, double, odd, and most multi-bit errors, programmable initial seed value, and optional feature to transpose input data and CRC result via transpose register.
Watchdog (WDOG)	The WDOG monitors internal system operation and forces a reset in case of failure. It can run from an independent 1 kHz low-power oscillator with a programmable refresh window to detect deviations in program flow or system frequency.

2.2.6 Analog modules

The following analog modules are available on this device:

Table 2-7. Analog modules

Module	Description
12-bit Analog-to-Digital Converter (ADC)	12-bit Cyclic ADC (ADCA and ADCB)
High speed comparator	Four high speed comparators each having its own 6-bit DAC sub-block
12-bit digital-to-analog converters (DAC)	Low-power general-purpose DAC, whose output can be placed on an external pin or set as one of the inputs to the comparator or ADC.

2.2.7 Timer modules

The following timer modules are available on this device:

Table 2-8. Timer modules

Module	Description
Programmable delay block (PDB)	• 16-bit resolution • 3-bit prescaler • Positive transition of trigger event signal initiates the counter • Supports one triggered delay output signal with four pre-trigger,each pre-trigger with an independent controlled delay from the trigger event. • Outputs can be used to schedule two conversions from one input trigger event and can schedule precise edge placement for a pulsed output. This feature is used to generate the control signal for the CMP windowing feature and output to a package pin if needed for applications, such as critical conductive mode power factor correction. • Continuous-pulse output or single-shot mode supported, each output is independently enabled, with possible trigger events • Supports bypass mode • Supports DMA
Flexible timer modules (FTM)	• Selectable FTM source clock, programmable prescaler

Table continues on the next page...

Table 2-8. Timer modules (continued)

Module	Description
	• 16-bit counter supporting free-running or initial/final value, and counting is up or up-down • Input capture, output compare, and edge-aligned and center-aligned eFlexPWM modes • Operation of FTM channels as pairs with equal outputs, pairs with complementary outputs, or independent channels with independent outputs • Deadtime insertion is available for each complementary pair • Generation of hardware triggers • Software control of PWM outputs • Up to 4 fault inputs for global fault control • Configurable channel polarity • Programmable interrupt on input capture, reference compare, overflowed counter, or detected fault condition • Quadrature decoder with input filters, relative position counting, and interrupt on position count or capture of position count on external event • DMA support for FTM events • Modulation feature to modulate one FTM PWM signal with another FTM channel • FTM1 can support XOR capture of three different pins on one channel, for simplified Hall Sensor speed detect for 3-phase motor control
Low-power timer (LPTimer)	• Selectable clock for prescaler/glitch filter of 1 kHz (internal LPO), 32.768 kHz (external crystal), or internal reference clock • Configurable Glitch Filter or Prescaler with 16-bit counter • 16-bit time or pulse counter with compare • Interrupt generated on Timer Compare • Hardware trigger generated on Timer Compare

Pulse Width Modulator A (PWMA)	• eFlexPWM module contains four identical submodules, with up to three outputs per submodule
	• 21-bit resolution for center, edge aligned, and asymmetrical PWM
	• PWMA with NanoEdge high resolution
	  • Fractional delay for enhanced resolution of the PWM period and edge placement
	  • Arbitrary PWM edge placement
	  • NanoEdge implementation: 312 ps PWM frequency and duty-cycle resolution
	• PWM outputs can operate as complementary pairs or independent channels
	• Interrupt generated on Timer Compare
	• Independent control of both edges of each PWM output
	• Enhanced input capture and output compare functionality on each input
	  • Channels not used for PWM generation can be used for buffered output compare functions
	  • Channels not used for PWM generation can be used for input capture functions
	  • Enhanced dual edge capture functionality
	• Up to four fault inputs can be assigned to control multiple PWM outputs
	• All outputs can be programmed to change simultaneously via a FORCE_OUT event
	• Option to supply the source for each complementary PWM signal pair from any of the following:
	  • Crossbar module outputs
	  • External ADC input, taking into account values set in ADC high and low limit registers

Quadrature Encoder/Decoder (ENC)	• Includes logic to decode quadrature signals

Table continues on the next page...

Table 2-8. Timer modules (continued)

Module	Description
	• Configurable digital filter for inputs to remove glitches and ensure only true transitions are recorded • 32-bit position counter register • 16-bit position difference register • Maximum count frequency equals the IPBus clock rate • Position counter can be initialized by software or external events • Position counter and resolution counter can be captured by external trigger signal (new feature) • Preloadable 16-bit revolution counter • Inputs can be connected to a general purpose timer, aiding low speed velocity measurements • Watchdog timer to detect a non-rotating shaft condition • Optional use as a single phase pulse accumulator
Periodic interrupt timers (PIT)	• Four general purpose interrupt timers • Interrupt timers for triggering ADC conversions • 32-bit counter resolution • DMA support

2.2.8 Communication interfaces

The following communication interfaces are available on this device:

Table 2-9. Communication modules

Module	Description
Flex Controller Area Network (FlexCAN)	Supports CAN protocol according to the CAN 2.0 B protocol specification.
Serial peripheral interface (SPI)	Synchronous serial bus for communication to an external device
Inter-integrated circuit (I2C)	Allows communication between a number of devices. Also supports the System Management Bus (SMBus) Specification, version 2.
Universal asynchronous receiver/ transmitters (UART)	Asynchronous serial bus communication interface with programmable 8- or 9-bit data format.

2.2.9 Human-machine interfaces

The following human-machine interfaces (HMI) are available on this device:

Table 2-10. HMI modules

Module	Description
General purpose input/output (GPIO)	All general purpose input or output (GPIO) pins are capable of interrupt and DMA request generation. All GPIO pins have 3.3 V tolerance. Max number of I/Os: • 74 in 100-pin package • 48 in 64-pin package

2.3 Orderable part numbers and features

Table 2-11. Orderable part numbers summary

Part number	CPU frequency (MHz)	Pin count	Total flash memory (KB)	SRAM (KB)	ADC		eFlexPWM		PWM Nano-Edge	Flex Timers			DAC	FlexCAN	
					ADCA	ADCB	PWMA PWMB	PWMX		FTM0	FTM3	FTM1		CAN0	CAN1
MKV46F256VLL16	168	100	256	32	18ch	20ch	1x8ch	1x4ch	Yes	1x8ch	1x8ch	1x2ch	1	1	1
MKV46F256VLH16	168	64	256	32	13ch	16ch	1x8ch	—	Yes	1x8ch	1x8ch	1x2ch	1	1	1
MKV46F128VLL16	168	100	128	24	18ch	20ch	1x8ch	1x4ch	Yes	1x8ch	1x8ch	1x2ch	1	1	1
MKV46F128VLH16	168	64	128	24	13ch	16ch	1x8ch	—	Yes	1x8ch	1x8ch	1x2ch	1	1	1
MKV44F256VLL16	168	100	256	32	18ch	20ch	1x8ch	1x4ch	Yes	—	—	—	1	1	1
MKV44F256VLH16	168	64	256	32	13ch	16ch	1x8ch	—	Yes	—	—	—	1	1	1
MKV44F128VLL16	168	100	128	24	18ch	20ch	1x8ch	1x4ch	Yes	—	—	—	1	1	1
MKV44F128VLH16	168	64	128	24	13ch	16ch	1x8ch	—	Yes	—	—	—	1	1	1
MKV44F128VLF16	168	48	128	24	11ch	10ch	1x8ch	—	Yes	—	—	—	1	1	—
MKV44F64VLH16	168	64	64	16	13ch	16ch	1x8ch	—	Yes	—	—	—	1	1	1
MKV44F64VLF16	168	48	64	16	11ch	10ch	1x8ch	—	Yes	—	—	—	1	1	—
MKV42F256VLL16	168	100	256	32	18ch	20ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	1
MKV42F256VLH16	168	64	256	32	13ch	16ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	1
MKV42F128VLL16	168	100	128	24	18ch	20ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	1
MKV42F128VLH16	168	64	128	24	13ch	16ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	1
MKV42F128VLF16	168	48	128	24	11ch	10ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	—
MKV42F64VLH16	168	64	64	16	13ch	16ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	1
MKV42F64VLF16	168	48	64	16	11ch	10ch	—	—	—	1x8ch	1x8ch	1x2ch	—	1	—

Chapter 3
Core overview

3.1 Arm Cortex-M4 Core Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by Arm and can be found at **arm.com**.

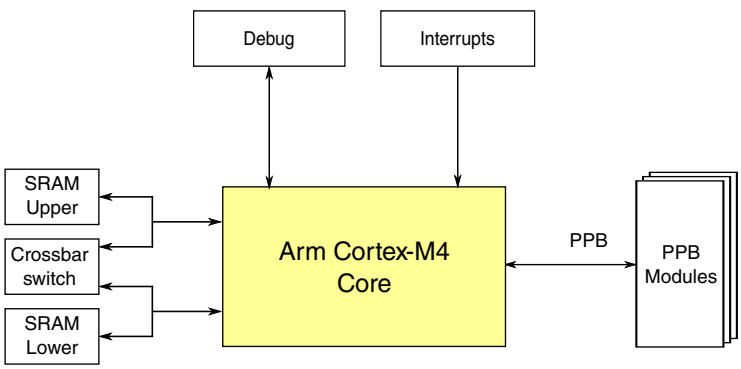


Figure 3-1. Core configuration

Table 3-1. Reference links to related information

Topic	Related module	Reference
Full description	Arm Cortex-M4 core	Arm Cortex-M4 Technical Reference Manual
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Debug	IEEE 1149.7 JTAG (cJTAG) Serial Wire Debug (SWD) Arm Real-Time Trace Interface	Debug
Interrupts	Nested Vectored Interrupt Controller (NVIC)	NVIC

Table continues on the next page...

Table 3-1. Reference links to related information (continued)

Topic	Related module	Reference
Private Peripheral Bus (PPB) module	Miscellaneous Control Module (MCM)	MCM
Private Peripheral Bus (PPB) module	Single-precision floating point unit (FPU)	FPU

3.1.1 Buses, interconnects, and interfaces

The Arm Cortex-M4 core has four buses as described in the following table.

Table 3-2. Buses of Arm Cortex-M4 core and their description

Bus name	Description
Instruction code (ICODE) bus	The ICODE and DCODE buses are muxed. This muxed bus is called the CODE bus and is connected to the crossbar switch via a single master port. In addition, the CODE bus is also tightly coupled to the lower half of the system RAM (SRAM_L).
Data code (DCODE) bus	
System bus	The system bus is connected to a separate master port on the crossbar. In addition, the system bus is tightly coupled to the upper half system RAM (SRAM_U).
Private peripheral (PPB) bus	The PPB provides access to these modules: • Arm modules such as the NVIC, ITM, DWT, FBP, and ROM table • Miscellaneous Control Module (MCM)

3.1.2 System Tick Timer

The System Tick Timer's clock source is always the core clock, FCLK. This results in the following:

- The CLKSOURCE bit in SysTick Control and Status register is always set to select the core clock.
- Because the timing reference (FCLK) is a variable frequency, the TENMS bit in the SysTick Calibration Value Register is always zero.
- The NOREF bit in SysTick Calibration Value Register is always set, implying that FCLK is the only available source of reference timing.

3.1.3 Debug facilities

This device has extensive debug capabilities including run control and tracing capabilities. The standard Arm debug port that supports JTAG and SWD interfaces. Also the cJTAG interface is supported on this device.

3.1.4 Core privilege levels

The Arm documentation uses different terms than this document to distinguish between privilege levels.

If you see this term...	it also means this term...
Privileged	Supervisor
Unprivileged or user	User

3.2 Nested Vectored Interrupt Controller (NVIC) Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by Arm and can be found at **arm.com**.

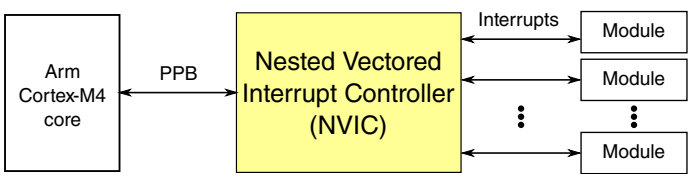


Figure 3-2. NVIC configuration

3.2.1 Interrupt priority levels

This device supports 16 priority levels for interrupts. Therefore, in the NVIC each source in the IPR registers contains 4 bits. For example, IPR0 is shown below:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IRQ3				0	0	0	0	IRQ2				0	0	0	0	IRQ1				0	0	0	0	IRQ0				0	0	0	0
W																																

3.2.2 Non-maskable interrupt

The non-maskable interrupt request to the NVIC is controlled by the external $\overline{\text{NMI}}$ signal. The pin the $\overline{\text{NMI}}$ signal is multiplexed on, must be configured for the $\overline{\text{NMI}}$ function to generate the non-maskable interrupt request.

3.2.3 Interrupt vector assignments

NOTE

Don't put the instruction to clear interrupt in the last line of ISR or the ISR will be re-entered. See Read-after-write sequence and required serialization of memory operations.

Table 3-4. Interrupt vector assignments

Address	Vector	IRQ	Source description	Source module
Arm Core System Handler Vectors				
0x0000_0000	0	—	Initial Stack Pointer	Arm core
0x0000_0004	1	—	Initial Program Counter	Arm core
0x0000_0008	2	—	Non-maskable Interrupt (NMI)	Arm core
0x0000_000C	3	—	Hard Fault	Arm core
0x0000_0010	4	—	MemManage Fault	Arm core
0x0000_0014	5	—	Bus Fault	Arm core
0x0000_0018	6	—	Usage Fault	Arm core
0x0000_001C	7	—	—	—
0x0000_0020	8	—	—	—
0x0000_0024	9	—	—	—
0x0000_0028	10	—	—	—
0x0000_002C	11	—	Supervisor call (SVCall)	Arm core
0x0000_0030	12	—	Debug Monitor	Arm core
0x0000_0034	13	—	—	—
0x0000_0038	14	—	Pendable request for system service (PendableSrvReq)	Arm core
0x0000_003C	15	—	System tick timer (SysTick)	Arm core
Non-Core Vectors				
On-platform Vectors				
0x0000_0040	16	0	DMA channel 0, 16 transfer complete	DMA
0x0000_0044	17	1	DMA channel 1, 17 transfer complete	DMA
0x0000_0048	18	2	DMA channel 2, 18 transfer complete	DMA
0x0000_004C	19	3	DMA channel 3, 19 transfer complete	DMA
0x0000_0050	20	4	DMA channel 4, 20 transfer complete	DMA
0x0000_0054	21	5	DMA channel 5, 21 transfer complete	DMA
0x0000_0058	22	6	DMA channel 6, 22 transfer complete	DMA
0x0000_005C	23	7	DMA channel 7, 23 transfer complete	DMA
0x0000_0060	24	8	DMA channel 8, 24 transfer complete	DMA
0x0000_0064	25	9	DMA channel 9, 25 transfer complete	DMA
0x0000_0068	26	10	DMA channel 10, 26 transfer complete	DMA

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ	Source description	Source module
0x0000_006C	27	11	DMA channel 11, 27 transfer complete	DMA
0x0000_0070	28	12	DMA channel 12, 28 transfer complete	DMA
0x0000_0074	29	13	DMA channel 13, 29 transfer complete	DMA
0x0000_0078	30	14	DMA channel 14, 30 transfer complete	DMA
0x0000_007C	31	15	DMA channel 15, 31 transfer complete	DMA
0x0000_0080	32	16	DMA error interrupt channels 0-1531	DMA
0x0000_0084	33	17	MCM interrupt	MCM
Off-platform Vectors				
0x0000_0088	34	18	Command complete	Flash memory
0x0000_008C	35	19	Read collision	Flash memory
0x0000_0090	36	20	Low-voltage detect, low-voltage warning	Mode Controller
0x0000_0094	37	21	Low Leakage Wakeup	LLWU
0x0000_0098	38	22	Both watchdog modules share this interrupt	WDOG and EWM
0x0000_009C	39	23		
0x0000_00A0	40	24	I2C0	I2C0
0x0000_00A4	41	25	—	
0x0000_00A8	42	26	SPI0	SPI0
0x0000_00AC	43	27		
0x0000_00B0	44	28		
0x0000_00B4	45	29		
0x0000_00B8	46	30		
0x0000_00BC	47	31	UART0 status sources	UART0
0x0000_00C0	48	32	UART0 error sources	UART0
0x0000_00C4	49	33	UART1 status sources	UART1
0x0000_00C8	50	34	UART1 error sources	UART1
0x0000_00CC	51	35		
0x0000_00D0	52	36		
0x0000_00D4	53	37		
0x0000_00D8	54	38	ADC_ERR A&B (zero cross, high/low limit)	ADC_ERR
0x0000_00DC	55	39	ADCA Scan complete	ADCA
0x0000_00E0	56	40	CMP0	CMP0
0x0000_00E4	57	41	CMP1	CMP1
0x0000_00E8	58	42	FTM0 8 channels	FTM0
0x0000_00EC	59	43	FTM1 2 channels	FTM1
0x0000_00F0	60	44		
0x0000_00F4	61	45		
0x0000_00F8	62	46		
0x0000_00FC	63	47	—	

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ	Source description	Source module
0x0000_0100	64	48	PIT Channel 0	PIT
0x0000_0104	65	49	PIT Channel 1	PIT
0x0000_0108	66	50	PIT Channel 2	PIT
0x0000_010C	67	51	PIT Channel 3	PIT
0x0000_0110	68	52	PDB0	PDB0
0x0000_0114	69	53	—	
0x0000_0118	70	54	XBARA	XBARA
0x0000_011C	71	55	PDB1	PDB1
0x0000_0120	72	56	DAC0	DAC0
0x0000_0124	73	57	MCG	MCG
0x0000_0128	74	58	LPTMR	Low Power Timer
0x0000_012C	75	59	Pin detect (Port A)	Port A control module
0x0000_0130	76	60	Pin detect (Port B)	Port B control module
0x0000_0134	77	61	Pin detect (Port C)	Port C control module
0x0000_0138	78	62	Pin detect (Port D)	Port D control module
0x0000_013C	79	63	Pin detect (Port E)	Port E control module
0x0000_0140	80	64	Software	Software
0x0000_0144	81	65		
0x0000_0148	82	66	ENC Compare	ENC Compare
0x0000_014C	83	67	ENC Home	ENC Home
0x0000_0150	84	68	ENC Watchdog/Simultaneous A & B change.	ENC Wdog/SAB
0x0000_0154	85	69	ENC Index/Roll over/Roll Under	ENC Index/Roll over/Roll Under
0x0000_0158	86	70	CMP2	CMP2
0x0000_015C	87	71	FTM3 8 channels	FTM3
0x0000_0160	88	72	—	
0x0000_0164	89	73	ADCB Scan complete	ADCB
0x0000_0168	90	74	—	
0x0000_016C	91	75	FLexCAN0 OR'ed Message buffer (0-15)	CAN0
0x0000_0170	92	76	FLexCAN0 Bus Off	CAN0
0x0000_0174	93	77	FLexCAN0 Error	CAN0
0x0000_0178	94	78	FLexCAN0 Transmit Warning	CAN0
0x0000_017C	95	79	FLexCAN0 Receive Warning	CAN0
0x0000_0180	96	80	FLexCAN0 Wake Up	CAN0
0x0000_0184	97	81	eFlexPWM submodule 0 Compare	eFlexPWM subm0 cmp
0x0000_0188	98	82	eFlexPWM submodule 0 Reload	eFlexPWM subm0 reload
0x0000_018C	99	83	eFlexPWM submodule 1 Compare	eFlexPWM subm1 cmp
0x0000_0190	100	84	eFlexPWM submodule 1 Reload	eFlexPWM subm1 reload
0x0000_0194	101	85	eFlexPWM submodule 2 Compare	eFlexPWM subm2 cmp
0x0000_0198	102	86	eFlexPWM submodule 2 Reload	eFlexPWM subm2 reload

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ	Source description	Source module
0x0000_019C	103	87	eFlexPWM submodule 3 Compare	eFlexPWM subm3 cmp
0x0000_01A0	104	88	eFlexPWM submodule 3 Reload	eFlexPWM submodule 3 Reload
0x0000_01A4	105	89	eFlexPWM all input captures	eFlexPWM all input captures
0x0000_01A8	106	90	eFlexPWM reload error	eFlexPWM error
0x0000_01AC	107	91	eFlexPWM Fault	eFlexPWM Fault
0x0000_01B0	108	92	CMP3	CMP3
0x0000_01B4	109	93	—	
0x0000_01B8	110	94	FLexCAN1 OR'ed Message buffer (0-15)	CAN1
0x0000_01BC	111	95	FLexCAN1 Bus Off	CAN1
0x0000_01C0	112	96	FLexCAN1 Error	CAN1
0x0000_01C4	113	97	FLexCAN1 Transmit Warning	CAN1
0x0000_01C8	114	98	FLexCAN1 Receive Warning	CAN1
0x0000_01CC	115	99	FLexCAN1 Wake Up	CAN1

3.3 Asynchronous Wake-up Interrupt Controller (AWIC) Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by Arm and can be found at **arm.com**.

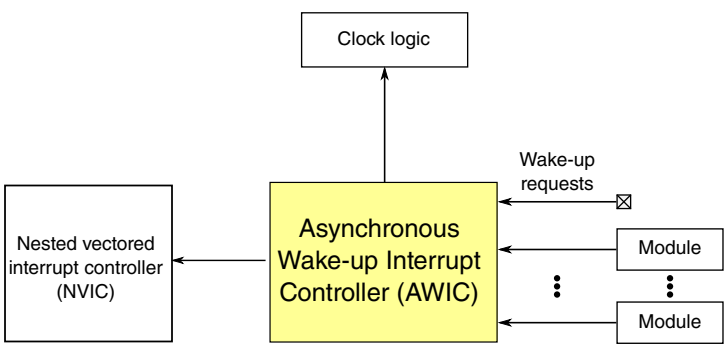


Figure 3-3. Asynchronous Wake-up Interrupt Controller configuration

Table 3-5. Reference links to related information

Topic	Related module	Reference
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management

Table continues on the next page...

Table 3-5. Reference links to related information (continued)

Topic	Related module	Reference
	Nested Vectored Interrupt Controller (NVIC)	NVIC
Wake-up requests		AWIC wake-up sources

3.3.1 Wake-up sources

The device uses the following internal and external inputs to the AWIC module.

Table 3-6. AWIC Stop and VLPS Wake-up Sources

Wake-up source	Description
Available system resets	$\overline{\text{RESET}}$ pin and WDOG when LPO is its clock source, and JTAG
Low-voltage detect	Mode Controller
Low-voltage warning	Mode Controller
Pin interrupts	Port Control Module - Any enabled pin interrupt is capable of waking the system
ADCx	The ADC is functional
CMPx	Since no system clocks are available, functionality is limited
I^2C	Address match wakeup
UART	Active edge on RXD
LPTMR	Functional in Stop/VLPS modes
FlexCAN	Functional in Stop mode
NMI	Non-maskable interrupt

3.4 FPU Configuration

This section summarizes how the module has been configured in the chip.

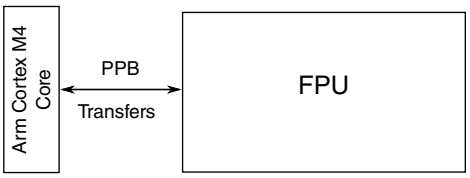


Figure 3-4. FPU configuration

Table 3-7. Reference links to related information

Topic	Related module	Reference
Full description	FPU	Arm Cortex-M4 Technical Reference Manual
System memory map		System memory map
Clocking		Clock Distribution
Power Management		Power Management
Transfers Private Peripheral Bus (PPB)	Arm Cortex M4 core	Arm Cortex-M4 core

3.5 JTAG Controller Configuration

This section summarizes how the module has been configured in the chip.

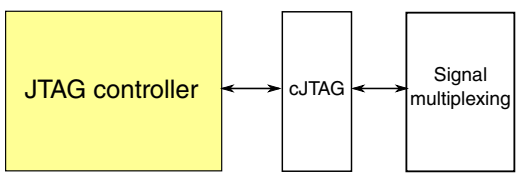


Figure 3-5. JTAGC Controller configuration

Table 3-8. Reference links to related information

Topic	Related module	Reference
Full description	JTAGC	JTAGC
Signal multiplexing	Port control	Signal multiplexing

Chapter 4
Memories and Memory Interfaces

4.1 Flash memory types

This chip contains a non-volatile program flash memory that can execute program code.

4.2 Flash Memory Sizes

The amounts of flash memory for the devices covered in this document are:

Table 4-1. Flash memory size

Part number	Flash (KB)	Block 0 (flash) address range
MKV46F256VLL16	256	0000_0000 - 0003 _FFFF
MKV46F256VLH16		
MKV46F128VLL16	128	0000_0000 - 0001_FFFF
MKV46F128VLH16		
MKV44F256VLL16	256	0000_0000 - 0003 _FFFF
MKV44F256VLH16		
MKV44F128VLL16	128	0000_0000 - 0001_FFFF
MKV44F128VLH16		
MKV44F128VLF16		
MKV44F64VLH16	64	0000_0000 - 0000_FFFF
MKV44F64VLF16		
MKV42F256VLL16	256	0000_0000 - 0003 _FFFF
MKV42F256VLH16		
MKV42F128VLL16	128	0000_0000 - 0001_FFFF
MKV42F128VLH16		
MKV42F128VLF16		
MKV42F64VLH16	64	0000_0000 - 0000_FFFF
MKV42F64VLF16		

4.3 Flash Security

How flash security is implemented on this device is described in Chip Security.

The flash module provides security information to the MCU based on the state held by the FSEC[SEC] bits. The MCU, in turn, confirms the security request and limits access to flash resources. During reset, the flash module initializes the FSEC register using data read from the security byte of the flash configuration field.

NOTE

The security features apply only to external accesses via debug. CPU accesses to the flash are not affected by the status of FSEC.

In the unsecured state all flash commands are available to the programming interfaces (JTAG), as well as user code execution of Flash Controller commands. When the flash is secured (FSEC[SEC] = 00, 01, or 11), and FSEC[MEEN] is not equal to 10, programmer interfaces are only allowed to launch mass erase operations and have no access to memory locations until after a mass erase is completed successfully.

4.4 Flash Modes

The flash memory is always configured in NVM normal. There are no operating conditions in which the flash is configured for NVM special mode.

4.5 Erase All Flash Contents

The flash of the MCU is protected from erasing all of the flash contents by the FTFA_FSEC[MEEN] bits. If the bits are set to 'b10 mass erase is disabled.

An Erase All Flash Blocks operation can be launched by software through a series of peripheral bus writes to flash registers. In addition the entire flash memory may be erased external to the flash memory from the SWJ-DP debug port by setting DAP_CONTROL[0]. DAP_STATUS[0] is set to indicate the mass erase command has been accepted. DAP_STATUS[0] is cleared when the mass erase completes.

4.6 FTFA_FOPT Register

The flash memory's FTFA_FOPT register allows the user to customize the operation of the MCU at boot time. See FOPT boot options for details of its definition.

4.7 SRAM sizes

The amount of SRAM for the devices covered in this document is shown in the following table.

Table 4-2. SRAM size

Part number	SRAM (KB)
MKV46F256VLL16	32
MKV46F256VLH16	32
MKV46F128VLL16	24
MKV46F128VLH16	24
MKV44F256VLL16	32
MKV44F256VLH16	32
MKV44F128VLL16	24
MKV44F128VLH16	24
MKV44F128VLF16	24
MKV44F64VLH16	16
MKV44F64VLF16	16
MKV42F256VLL16	32
MKV42F256VLH16	32
MKV42F128VLL16	24
MKV42F128VLH16	24
MKV42F128VLF16	24
MKV42F64VLH16	16
MKV42F64VLF16	16

4.8 SRAM Arrays

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

The on-chip RAM is implemented such that the SRAM_L and SRAM_U ranges form a contiguous block in the memory map. As such:

- SRAM_L is anchored to 0x1FFF_FFFF and occupies the space before this ending address.
- SRAM_U is anchored to 0x2000_0000 and occupies the space after this beginning address.

This is illustrated in the following figure.

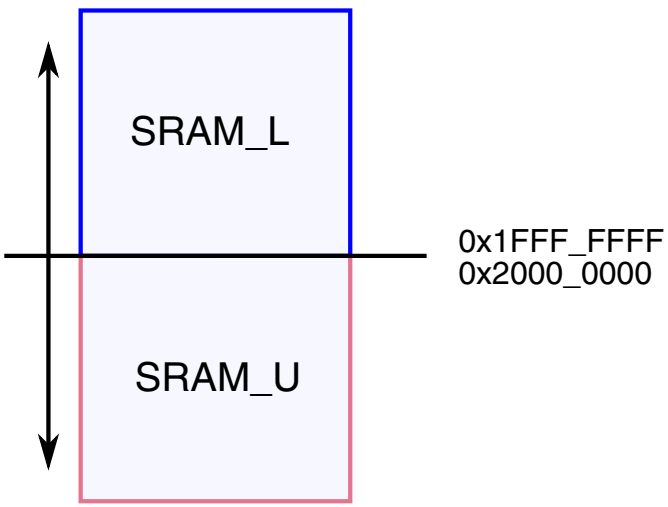


Figure 4-1. SRAM blocks memory map

For this device the SRAM ranges are shown in the following table.

SRAM size (KB)	SRAM_L range	SRAM_U range
32	1FFF_C000 – 1FFF_FFFF (16K)	2000_0000 – 2000_3FFF (16K)
24	1FFF_E000 – 1FFF_FFFF (8K)	2000_0000 – 2000_3FFF (16K)
16	1FFF_E000 – 1FFF_FFFF (8K)	2000_0000 – 2000_1FFF (8K)

4.9 SRAM retention in low power modes

The SRAM is retained down to VLLS3 mode.

In VLLS2 the region of SRAM_U from 0x2000_0000 is powered.

In VLLS1 and VLLS0 no SRAM is retained.

4.10 System Register file

This device includes a 32-byte register file that is powered in all power modes. Also, it retains contents during low-voltage detect (LVD) events and is only reset a power-on reset.

Chapter 5
Memory Map

5.1 Introduction

This chip contains both Flash and RAM memories and memory-mapped peripherals which are located in one contiguous memory space. The Arm M4 core supports both register access of the various peripherals and also bit-band accesses. Following are the memory sizes present.

- 256KB Flash memory
- 32 KB RAM

5.2 System Memory Map

The following table shows the high-level device memory map. This map provides the complete architectural address space definition for the various sections. Based on the physical sizes of the memories and peripherals, the actual address regions used may be smaller.

The system memory map includes multiple alias address spaces that are intended for specific purposes. There are two aliased address spaces that are mapped into the ICode regions (address < 0x2000_0000) for code sections that are normally located in the system region of the memory map. However, two subsets of this space are aliased so they appear in the ICode region. This enables the instructions mapped into this space to be executed with maximum performance.

There is an aliased region that maps a system address space to the Program flash section. The Flash region aliasing is specifically intended for references to read-only data coefficients in the flash while still preserving a full Harvard memory organization in the processor core supporting concurrent instruction fetches (for example, from RAM) and data accesses (from flash via the aliased space).

The bitbanding functionality supported by the processor core uses aliased regions that map to the basic RAM and peripheral address spaces. This functionality maps each 32-bit word of the aliased address space to a unique bit in the underlying RAM or peripheral address space to support single-bit insert and extract operations from the processor.

Table 5-1. System Memory Map

System 32-bit Byte Address Range	Destination Slave	Access
0x0000_0000–0x0003_FFFF	256K Program Flash and read only data	All Masters
0x0004_0000–0x01FF_BFFF	Reserved	-
0x1FFF_C000–0x1FFF_FFFF	TCRAM Lower (16K)	All Masters
0x2000_0000–0x2000_3FFF	TCRAM Upper (16K)	All Masters
0x2000_4000–0x21FF_FFFF	Reserved	–
0x2200_0000–0x23FF_FFFF	Aliased to SRAM_U bitband	Cortex-M4 core only
0x2400_0000–0x2FFF_FFFF	Reserved	–
0x3000_0000–0x33FF_FFFF	Program Flash and Read only data	Cortex M4 core only
0x3400_0000–0x3FFF_FFFF	Reserved	–
0x4000_0000–0x4007_FFFF	Bitband region for peripheral bridge 0 (AIPS-Lite0)	All Masters
0x4008_0000–0x400F_EFFF	Reserved	–
0x400F_F000–0x400F_FFFF	Bitband region for general purpose input/output (GPIO)	All Masters
0x4010_0000–0x41FF_FFFF	Reserved	–
0x4200_0000–0x43FF_FFFF	Aliased to peripheral bridge (AIPS-Lite) and general purpose input/output (GPIO) bitband	Cortex-M4 core only
0x4400_0000–0xDFFF_FFFF	Reserved	-
0xE000_0000–0xE00F_FFFF	Private Peripherals	Cortex-M4 core only
0xE010_0000 - 0xFFFF_FFFF	Reserved	-

5.3 Peripheral Memory Map

The peripheral memory map is accessible via one slave port on the crossbar in the 0x4000_0000–0x4007_FFFF region. The device implements one peripheral bridge that defines a 512 KB address space.

AIPS 0 is located on slave port 2, the address space is 0x4000_0000–0x4007_7FFF. For programming model accesses via the peripheral bridges, there is generally only a small range within the 4 KB slots that is implemented. Accessing an address that is not implemented in the peripheral results in a transfer error termination.

5.3.1 Read-after-write sequence and required serialization of memory operations

In some situations, a write to a peripheral must be completed fully before a subsequent action can occur. Examples of such situations include:

- Exiting an interrupt service routine (ISR)
- Changing a mode
- Configuring a function

In these situations, application software must perform a read-after-write sequence to guarantee the required serialization of the memory operations:

1. Write the peripheral register
2. Read the written peripheral register to verify the write
3. Continue with subsequent operations

5.3.2 Peripheral Bridge 0 (AIPS-Lite 0) Memory Map

- Slots 0-79 are 32-bit data width modules,
- Slots 80-95 are 16-bit data width modules, and
- Slots 96-126 are 8-bit data width modules.

Table 5-2. Peripheral bridge 0 slot assignments

System 32-bit base address	Slot number	Module
On-platform		
0x4000_0000	0	Peripheral bridge 0 (AIPS-Lite 0)
0x4000_1000	1	—
0x4000_2000	2	—
0x4000_3000	3	—
0x4000_4000	4	Reserved for XBAR Lite no registers
0x4000_5000	5	—
0x4000_6000	6	—
0x4000_7000	7	—
0x4000_8000	8	DMA controller
0x4000_9000	9	DMA controller transfer control descriptors
0x4000_A000	10	—
0x4000_B000	11	—
0x4000_C000	12	—
0x4000_D000	13	—
0x4000_E000	14	—
0x4000_F000	15	—

Table continues on the next page...

Table 5-2. Peripheral bridge 0 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4001_0000	16	—
0x4001_1000	17	—
0x4001_2000	18	—
0x4001_3000	19	—
0x4001_4000	20	—
0x4001_5000	21	—
0x4001_6000	22	—
0x4001_7000	23	—
0x4001_8000	24	—
0x4001_9000	25	—
0x4001_A000	26	—
0x4001_B000	27	—
0x4001_C000	28	—
0x4001_D000	29	—
0x4001_E000	30	—
0x4001_F000	31	Flash memory controller (FMC)
Off-platform		
0x4002_0000	32	Flash memory (FTMR)
0x4002_1000	33	DMA_MUX channel multiplexor
0x4002_2000	34	
0x4002_3000	35	
0x4002_4000	36	FlexCAN0
0x4002_5000	37	FlexCAN1
0x4002_6000	38	FTM3 (8 channel FlexTimer)
0x4002_7000	39	-
0x4002_8000	40	
0x4002_9000	41	
0x4002_A000	42	
0x4002_B000	43	
0x4002_C000	44	DSPI0
0x4002_D000	45	
0x4002_E000	46	
0x4002_F000	47	
0x4003_0000	48	-
0x4003_1000	49	PDB1
0x4003_2000	50	CRC
0x4003_3000	51	eFlexPWM
0x4003_4000	52	
0x4003_5000	53	

Table continues on the next page...

Table 5-2. Peripheral bridge 0 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4003_6000	54	PDB0
0x4003_7000	55	PIT (periodic interrupt timer)
0x4003_8000	56	FTM0 (8 channel FlexTimer)
0x4003_9000	57	FTM1 (2 channel FlexTimer)
0x4003_A000	58	
0x4003_B000	59	
0x4003_C000	60	
0x4003_D000	61	-
0x4003_E000	62	
0x4003_F000	63	DAC0
0x4004_0000	64	LPTMR
0x4004_1000	65	System Register File
0x4004_2000	66	-
0x4004_3000	67	
0x4004_4000	68	-
0x4004_5000	69	-
0x4004_6000	70	Analogue Test Annex
0x4004_7000	71	SIM low power logic
0x4004_8000	72	System Integration Module (SIM)
0x4004_9000	73	Port A mux control
0x4004_A000	74	Port B mux control
0x4004_B000	75	Port C mux control
0x4004_C000	76	Port D mux control
0x4004_D000	77	Port E mux control
0x4004_E000	78	-
0x4004_F000	79	-
0x4005_0000	80	-
0x4005_1000	81	-
0x4005_2000	82	Software Watchdog
0x4005_3000	83	
0x4005_4000	84	-
0x4005_5000	85	ENC (encoder/decoder module)
0x4005_6000	86	
0x4005_7000	87	
0x4005_8000	88	-
0x4005_9000	89	XBARA (inter-peripheral crossbar switch A)
0x4005_A000	90	XBARB (inter-peripheral crossbar switch B)
0x4005_B000	91	AOI (AND OR Invert module)
0x4005_C000	92	ADCA and ADCB (cyclic ADCs)

Table continues on the next page...

Table 5-2. Peripheral bridge 0 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4005_D000	93	-
0x4005_E000	94	-
0x4005_F000	95	-
0x4006_0000	96	
0x4006_1000	97	External Watchdog Monitor (EWM)
0x4006_2000	98	-
0x4006_3000	99	-
0x4006_4000	100	MCG
0x4006_5000	101	OSC0
0x4006_6000	102	I2C
0x4006_7000	103	-
0x4006_8000	104	-
0x4006_9000	105	-
0x4006_A000	106	UART0
0x4006_B000	107	UART1
0x4006_C000	108	-
0x4006_D000	109	-
0x4006_E000	110	-
0x4006_F000	111	-
0x4007_0000	112	-
0x4007_1000	113	-
0x4007_2000	114	-
0x4007_3000	115	CMP0, CMP1, CMP2, CMP3
0x4007_4000	116	-
0x4007_5000	117	-
0x4007_6000	118	-
0x4007_7000	119	-
0x4007_8000	120	-
0x4007_9000	121	-
0x4007_A000	122	-
0x4007_B000	123	-
0x4007_C000	124	Low Leakage Wakeup (LLWU)
0x4007_D000	125	Power Management Controller (PMC)
0x4007_E000	126	System Mode Controller (SMC)
0x4007_F000	127	Reset Control Module (RCM)
0x400F_F000	not on peripheral bridge	GPIO controller

Chapter 6
Clock Distribution

6.1 Introduction

The KV4x family is based on the Kinetis Arm M4 based platform and utilizes the MCG (Multiple Clock Generator) module that provides the clocks for the CPU, memories and peripherals. The MCG has input clocks from the OSC module, providing an external feed from a ceramic resonator/crystal/external clock, and internal RC oscillators. The MCG has a PLL that provides a multyiplying function on the input clock source to generate PLL clock frequencies from 110 MHz to 240 MHz and to generate PLL 2x clock frequencies from 220 MHz to 480 MHz.

The MCG works in conjunction with the SIM module which provides optional feeds and prescalers to the CPU, memories, and peripherals. This chip deploys the edge placement module that requires a PLL clock.

The primary clocks for the system are generated from the MCGOUTCLK clock. The clock generation circuitry provides several clock dividers that allow different portions of the device to be clocked at different frequencies. This allows for trade-offs between performance and power dissipation.

Various modules, such as the eFlexPWM, have module-specific clocks that can be generated from the MCGPLLCLK clock. In addition, there are various other module-specific clocks that have other alternate sources. Clock selection for most modules is controlled by the SOPT registers in the SIM module.

6.2 High-level device clocking diagram

The following system oscillator, MCG, and SIM module registers control the multiplexers, dividers, and clock gates shown in the below figure:

	OSC	MCG	SIM
Muliplexers	MCG_C*x*	MCG_C*x*	SIM_SOPT1, SIM_SOPT2
Dividers	—	MCG_C*x*	SIM_CLKDIV*x*
Clock gates	OSC_CR	MCG_C1	SIM_SCGC*x*

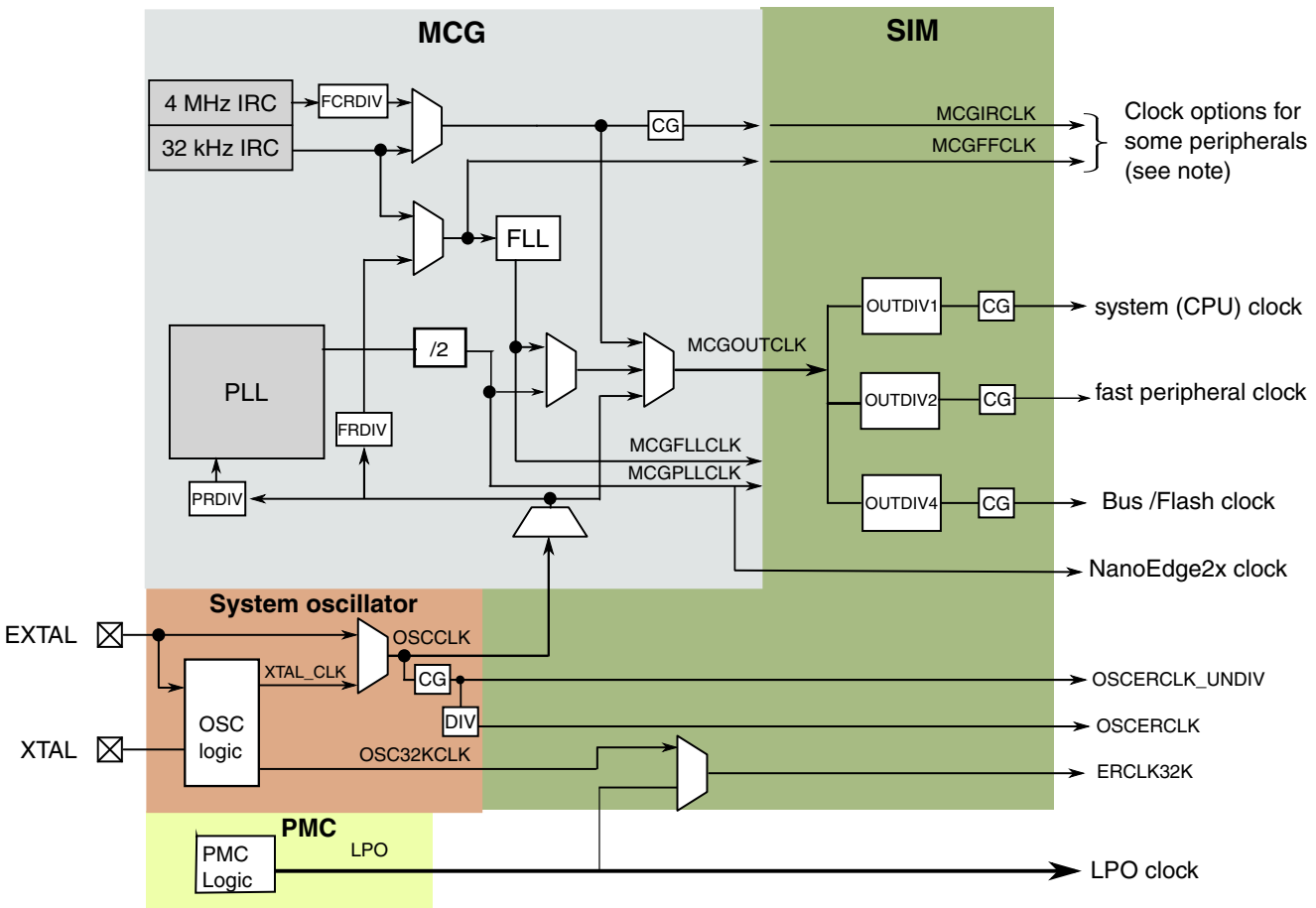


Figure 6-1. Clocking diagram

6.2.1 Clock definitions

The following table describes the clocks in the previous block diagram.

Clock name	Description
CPU clock / System clock	MCGOUTCLK divided by OUTDIV1 clocks the Arm Cortex-M4 core, RAM, DMA, GPIO, FMC module, and crossbar switch bus masters.

Table continues on the next page...

Clock name	Description
Fast Peripheral clock	MCGOUTCLK divided by OUTDIV2 clocks the UARTs, SPI, eFlexPWM, FTMs, PDBs, ENC, FlexCAN, XBARA, and ADC modules.
Bus /Flash clock	MCGOUTCLK divided by OUTDIV4 clocks Flash, I2C, WDOG, EWM, PIT, LPTIMER , OSC, MCG, PMC , XBARB/ AOI, CMP.
NanoEdge clock	The NanoEdge module will be fed with MCGPLLCLK.
MCGIRCLK	MCG output of the slow or fast internal reference clock
MCGFFCLK	MCG output of the slow internal reference clock or a divided MCG external reference clock.
MCGOUTCLK	MCG output of either IRC, MCGPLLCLK, or MCG's external reference clock that sources the core, system, bus and flash clocks. It is also an option for the debug trace clock.
MCGPLLCLK	MCG output of the PLL. MCGFLLCLK or MCGPLLCLK may clock some modules.
OSCCLK	System oscillator output of the internal oscillator or sourced directly from EXTAL
OSCERCLK	System oscillator output sourced from OSCCLKthat may clock some on-chip modules
LPO	PMC 1kHz output

6.3 Internal clocking requirements

The clock dividers are programmed via the SIM module's CLKDIV registers. Each divider is programmable from a divide-by-1 through divide-by-16 setting. The following requirements must be met when configuring the clocks for this device:

1. The System clock frequency that drives the CPU platform must be 168 MHz or slower in HSRUN mode, or 100 MHz or slower in normal RUN mode.
2. The fast peripheral bus clock frequency must be an integer divide or multiple of the core/system clock. ie x2,x3,x4 or divide by 2/4/8. This allows key peripherals can be clocked at high speed, while core/System is running slower to conserve power consumption
3. The (slow peripheral) bus flash clock must not exceed 25MHz, and be an integer divide of the core/System clock and an integer divide of the fast peripheral clock.
4. The NanoEdge module requires a clock input. The fast peripheral bus clock provides the NanoEdge clock, and MCGPLLCLK provides the "2x Fast peripheral bus clock". Each of these clocks must be an integer divide or multiple of the System clock. The NanoEdge module is expected to be programmed to use clock inputs of 84/168 MHz or 100/200 MHz to support sub-nanosecond resolution control of the flexPWM.

NOTE

To enable NanoEdge module for nanosecond resolution, PLL must be enabled to provide high frequencies clock source, that is, MCGPLLCLK. When NanoEdge is enabled, the system clock source (core and system clock, fast bus clock, slow bus clock, and flash clock must be from/ divided from MCGPLLCLK.

The following are a few of the more common clock configurations for this device:

High Speed Run Mode	Frequency
System (CPU) clock	168 MHz max.
Fast Peripheral clock	84 MHz max.
Bus / Flash clock	24 MHz max

RUN mode	Frequency
System (CPU) clock	100 MHz max.
Fast Peripheral clock	100 MHz max.
Bus / Flash clock	25 MHz max

RUN mode	Frequency
System (CPU) clock	50 MHz
Fast Peripheral clock	100 MHz
Bus / Flash clock	25 MHz max

6.3.1 Clock divider values after reset

Each clock divider is programmed via the SIM module's CLKDIV*n* registers. The flash memory's FTFA_FOPT[LPBOOT] bit controls the reset value of the core clock, System (CPU) clock, bus clock, and flash clock dividers as shown below:

FTFA_FOPT [LPBOOT]	System (CPU) clock	Fast Peripheral clock	Bus / Flash clock	Description
0	0x7 (divide by 8)	0x7 (divide by 8)	0xF (divide by 16)	Low power boot
1	0x0 (divide by 1)	0x0 (divide by 1)	0x1 (divide by 2)	Fast clock boot

This gives the user flexibility for a lower frequency, low-power boot option. The flash erased state defaults to fast clocking mode, since where the low power boot (FTFA_FOPT[LPBOOT]) bit resides in flash is logic 1 in the flash erased state.

To enable the low power boot option program FTFA_FOPT[LPBOOT] to zero. During the reset sequence, if LPBOOT is cleared, the system is in a slow clock configuration. Upon any system reset, the clock dividers return to this configurable reset state.

6.3.2 VLPR mode clocking

The clock dividers cannot be changed while in VLPR mode. They must be programmed prior to entering VLPR mode to guarantee:

- the core/system and fast peripheral clocks are less than or equal to 4 MHz, and
- the bus /flash clock is less than or equal to 1 MHz
- the NanoEdge clock is disabled as it cannot support the NanoEdge resolution for eFlexPWM

NOTE

When the MCG is in BLPI and clocking is derived from the Fast IRC, the clock divider controls, MCG_SC[FCRDIV] and SIM_CLKDIV1[OUTDIV4], must be programmed such that the resulting flash clock nominal frequency is 800 kHz or less. In this case, one example of correct configuration is MCG_SC[FCRDIV]=000b and SIM_CLKDIV1[OUTDIV4]=0100b, resulting in a divide by 5 setting.

6.4 Clock Gating

The clock to each module can be individually gated on and off using the SIM module's SCGC*x* registers. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing a module, set the corresponding bit in SCGC*x* register to enable the clock. Before turning off the clock, make sure to disable the module.

Any bus access to a peripheral that has its clock disabled generates an error termination.

6.5 Module clocks

The following table summarizes the clocks associated with each module.

Table 6-1. Module clocks

Module	Bus interface clock	Internal clocks	I/O interface clocks
Core modules			
Arm Cortex-M4 core	Core/System clock	Core clock	—
NVIC	Core/System clock	—	—
DAP	Core/System clock	—	—
ITM	Core/System clock	—	—
cJTAG, JTAGC	—	—	JTAG_CLK
System modules			
DMA	Core/System clock	—	—
DMA Mux	Bus / Flash clock	—	—
Port control	Bus / Flash clock	LPO	—
Crossbar Switch	Core/System clock	—	—
Peripheral bridges	Core/System clock	Bus clock, Flash clock	—
XBARA,XBARB,AOI	Bus / Flash clock		—
LLWU, PMC, SIM, RCM	Bus/ Flash clock	LPO	—
Mode controller	Bus / Flash clock	—	—
MCM	Core/System clock	—	—
EWM	Bus / Flash clock	LPO	—
Watchdog timer	Bus / Flash clock	LPO	—
Clocks			
MCG	Bus /Flash clock	MCGOUTCLK, MCGPLLCLK, MCGFLLCLK, MCGIRCLK, OSCERCLK, OSCERCLK_UNDIV	—
OSC	Bus / Flash clock	OSCERCLK, OSCERCLK_UNDIV	—
Memory and memory interfaces			
Flash Controller	Core/System clock	Flash clock	—
Flash memory	Bus /Flash clock	—	—
NanoEdge module	Fast Peripheral clock + MCGPLLCLK	—	—
Security			
CRC	Core/System clock	—	—
Analog			
ADC	Fast Peripheral clock	MCGIRCLK	—
CMP	Bus / Flash clock	—	—
DAC	Fast Peripheral clock	—	—
Timers			
PDB	Fast Peripheral clock	—	—
FlexTimers	Fast Peripheral clock	MCGFFCLK	FTM_CLKIN*x*
PIT	Bus / Flash clock	—	—

Table continues on the next page...

Table 6-1. Module clocks (continued)

Module	Bus interface clock	Internal clocks	I/O interface clocks
LPTMR	Bus / Flash clock	LPO, OSCERCLK_UNDIV, MCGIRCLK, ERCLK32K	—
eFlexPWM	Fast Peripheral clock	MCGPLLCLK	—
ENC	Fast Peripheral clock		—
Communication interfaces			
FlexCAN	Fast Peripheral clock	OSCERCLK	—
DSPI	Fast Peripheral clock	—	DSPI_SCK
I^2C	Bus / Flash clock	—	I2C_SCL
UART0, UART1	Fast Peripheral clock	—	—
Human-machine interfaces			
GPIO	Core/System clock	—	—

6.5.1 NanoEdge clocking

The clocking options for the eFlexPWMA module with NanoEdge functionality are shown in the following figure.

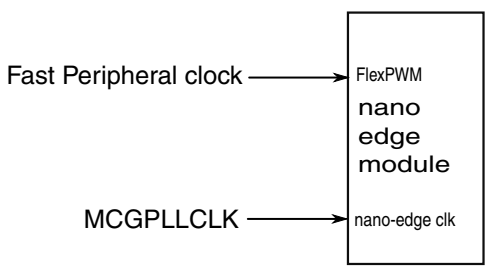


Figure 6-2. NanoEdge module Clock inputs

NOTE

In order to make NanoEdge PWM work, the ratio of Fast Peripheral clock and MCGPLLCLK must be 1:2.

6.5.2 WDOG clocking

The WDOG may be clocked from two clock sources as shown in the following figure.

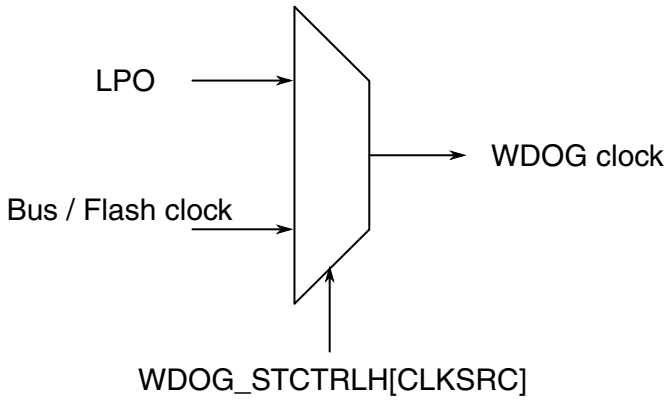


Figure 6-3. WDOG clock generation

6.5.3 Debug trace clock

The debug trace clock source can be clocked as shown in the following figure.

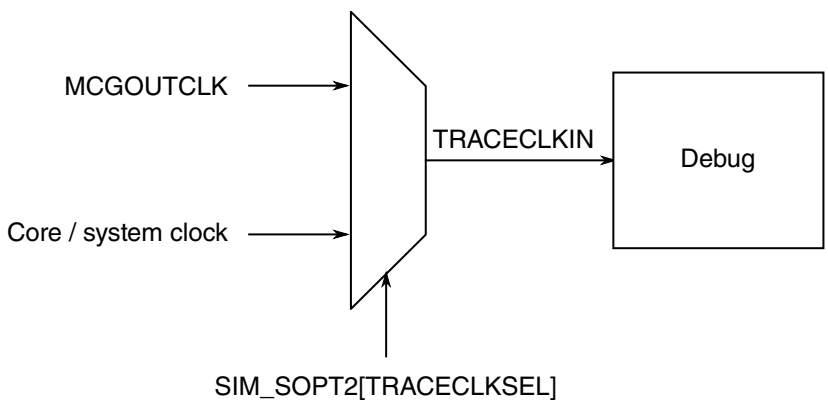


Figure 6-4. Trace clock generation

6.5.4 PMC 1-kHz LPO clock

The Power Management Controller (PMC) generates a 1-kHz clock that is enabled in all modes of operation, including all low power modes. This 1-kHz source is commonly referred to as LPO clock or 1-kHz LPO clock.

6.5.5 PORT digital filter clocking

The digital filters in the PORTD module can be clocked as shown in the following figure.

NOTE

In stop mode, the digital input filters are bypassed unless they are configured to run from the 1 kHz LPO clock source.

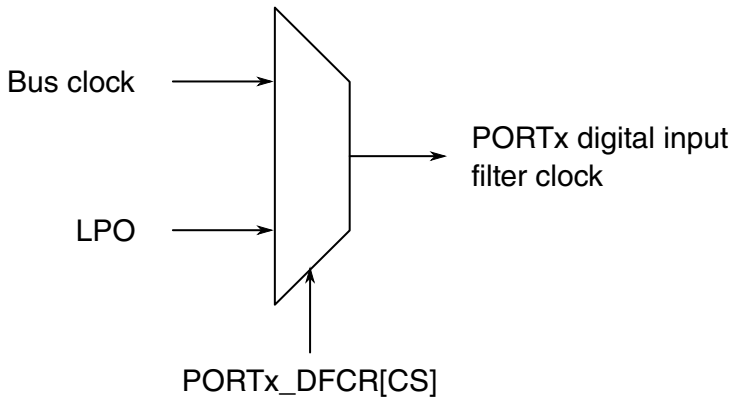


Figure 6-5. PORTx digital input filter clock generation

6.5.6 LPTMR clocking

The prescaler and glitch filters in each of the LPTMR*x* modules can be clocked as shown in the following figure.

NOTE

The chosen clock must remain enabled if the LPTMR*x* is to continue operating in all required low-power modes.

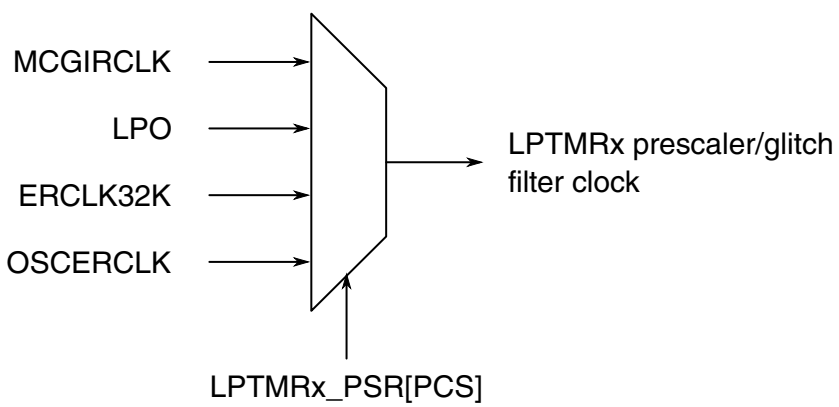


Figure 6-6. LPTMR*x* prescaler/glitch filter clock generation

6.5.7 FlexCAN clocking

The clock for the FlexCAN's protocol engine can be selected as shown in the following figure.

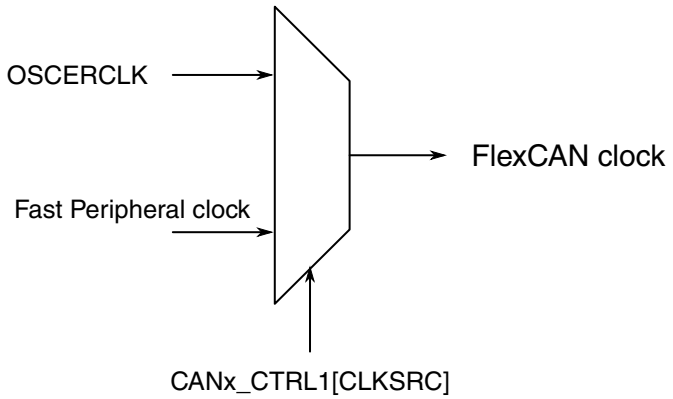


Figure 6-7. FlexCAN clock generation

6.5.8 UART clocking

UART0 and UART1 modules operate from the fast peripheral clock, which provides higher performance level for these modules.

6.6 External clocks

The input clocks to the SoC are described in detail in the MCG chapter.

- FlexTimers
  - The FlexTimers have an external clock input, FTM_XCLK, which must be no faster than 1/4 of the bus_clk frequency
- Arm Trace
  - Arm Trace port outputs are synchronous to this clock

Chapter 7
Power Management

7.1 Introduction

This chapter describes the various chip power modes and functionality of the individual modules in these modes.

7.2 Clocking Modes

This section describes the various clocking modes supported on this device.

7.2.1 Partial Stop

Partial Stop is a clocking option that can be taken instead of entering Stop mode and is configured in the SMC Stop Control Register (SMC_STOPCTRL). The Stop mode is only partially entered, which leaves some additional functionality alive at the expense of higher power consumption. Partial Stop can be entered from either Run mode or VLP Run mode.

When configured for PSTOP2, only the core and system clocks are gated and the bus clock remains active. The bus masters and bus slaves clocked by the system clock enter Stop mode, but the bus slaves clocked by the bus clock remain in Run (or VLP Run) mode. The clock generators in the MCG and the on-chip regulator in the PMC also remain in Run (or VLP Run) mode. Exit from PSTOP2 can be initiated by a reset, an asynchronous interrupt from a bus master or bus slave clocked by the system clock, or a synchronous interrupt from a bus slave clocked by the bus clock. If configured, a DMA request (using the asynchronous DMA wakeup) can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP2.

When configured for PSTOP1, both the system clock and the bus clock are gated. All bus masters and bus slaves enter Stop mode, but the clock generators in the MCG and the on-chip regulator in the PMC remain in Run (or VLP Run) mode. Exit from PSTOP1 can be initiated by a reset or an asynchronous interrupt from a bus master or bus slave. If configured, an asynchronous DMA request can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP1.

PSTOP1 is functionally similar to STOP mode, but offers faster wakeup at the expense of higher power consumption. Another benefit is that it keeps all of the MCG clocks enabled, which can be useful for some of the asynchronous peripherals that can remain functional in Stop modes.

7.2.2 DMA Wakeup

The DMA can be configured to wake up the device on a DMA request whenever it is placed in Stop mode. The wakeup is configured per DMA channel and is supported in Compute Operation, PSTOP, Stop, and VLPS low power modes.

When a DMA wakeup is detected in PSTOP, Stop or VLPS, then the device initiates a normal exit from the low power mode. This can include restoring the on-chip regulator and internal power switches, enabling the clock generators in the MCG, enabling the system and bus clocks (but not the core clock) and negating the Stop mode signal to the bus masters and bus slaves. The only difference is that the CPU remains in the low power mode with the CPU clock disabled.

During Compute Operation, a DMA wakeup initiates a normal exit from Compute Operation. This includes enabling the clocks and negating the Stop mode signal to the bus masters and bus slaves. The core clock always remains enabled during Compute Operation.

Because the DMA wakeup enables the clocks and negates the Stop mode signals to all bus masters and slaves, software needs to ensure that bus masters and slaves that are not involved with the DMA wakeup and transfer remain in a known state. That can be accomplished by disabling the modules before entry into the low power mode or by setting the Doze enable bit in selected modules.

After the DMA request that initiated the wakeup negates and the DMA completes the current transfer, the device transitions back into the original low power mode. This includes requesting all non-CPU bus masters to enter Stop mode and then requesting bus slaves to enter Stop mode. In Stop and VLPS modes, the MCG and PMC then also enter their appropriate modes.

NOTE

If the requested DMA transfer cannot cause the DMA request to negate, then the device remains in a higher power state until the low power mode is fully exited.

If the DMA request asserts during the Stop mode entry sequence (or reentry if the request asserts during a DMA wakeup), then an enabled DMA wakeup can cause an aborted entry into the low power mode, as well as cause the SMC to assert its Stop Abort flag.

An interrupt that occurs during a DMA wakeup causes an immediate exit from the low power mode (this is optional for Compute Operation) without impacting the DMA transfer.

A DMA wakeup can be generated by either a synchronous DMA request or an asynchronous DMA request. Not all peripherals can generate an asynchronous DMA request in Stop modes. In general, though, if a peripheral can generate synchronous DMA requests and also supports asynchronous interrupts in Stop modes, then it can generate an asynchronous DMA request.

7.2.3 Compute Operation

Compute Operation is an execution or compute-only mode of operation that keeps the CPU enabled with full access to the SRAM and Flash read port, but places all other bus masters and bus slaves into their stop mode. Compute Operation can be enabled in either Run mode or VLP Run mode.

NOTE

Do not enter any stop mode without first exiting Compute Operation.

Because Compute Operation reuses the stop mode logic (including the staged entry with bus masters disabled before bus slaves), any bus master or bus slave that can remain functional in stop mode also remains functional in Compute Operation, including generation of asynchronous interrupts and DMA requests. When enabling Compute Operation in Run mode, module functionality for bus masters and slaves is the equivalent of STOP mode. When enabling Compute Operation in VLP Run mode, module functionality for bus masters and slaves is the equivalent of VLPS mode. The MCG, PMC, SRAM and Flash read port are not affected by Compute Operation, although the Flash register interface is disabled.

During Compute Operation, the AIPS peripheral space is disabled and attempted accesses generate bus errors. The private peripheral space remains accessible during Compute Operation, including the MCM, NVIC, IOPORT and SysTick. Although access to the

GPIO registers via the IOPORT is supported, the GPIO port data input registers do not return valid data since clocks are disabled to the Port Control and Interrupt modules. By writing to the GPIO port data output registers, it is possible to control those GPIO ports that are configured as output pins.

Compute Operation is controlled by the CPO register in the MCM, which is only accessible to the CPU. Setting or clearing the CPOREQ bit in the MCM initiates entry or exit into Compute Operation. Compute Operation can also be configured to exit automatically on detection of an interrupt, which is required in order to service most interrupts. Only the core system interrupts (exceptions, including NMI and SysTick) and any edge sensitive interrupts can be serviced without exiting Compute Operation.

When entering Compute Operation, the CPOACK status bit indicates when entry has completed. When exiting Compute Operation in Run mode, the CPOACK status bit negates immediately. When exiting Compute Operation in VLP Run mode, the exit is delayed to allow the PMC to handle the change in power consumption. This delay means the CPOACK bit is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

The DMA wakeup is also supported during Compute Operation and causes the CPOACK status bit to clear and the AIPS peripheral space to be accessible for the duration of the DMA wakeup. At the completion of the DMA wakeup, the device transitions back into Compute Operation.

7.2.4 Peripheral Doze

Several peripherals support a peripheral Doze mode, where a register bit can be used to disable the peripheral for the duration of a low power mode. The Flash can also be placed in a low power state during Peripheral Doze via a register bit in the SIM.

Peripheral Doze is defined to include all of the modes of operation listed below.

- The CPU is in wait mode.
- The CPU is in stop mode, including the entry sequence and for the duration of a DMA wakeup.
- The CPU is in Compute Operation, including the entry sequence and for the duration of a DMA wakeup.

Peripheral Doze can therefore be used to disable selected bus masters or slaves for the duration of WAIT or VLPW mode. It can also be used to disable selected bus slaves immediately on entry into any stop mode (or Compute Operation), instead of waiting for

the bus masters to acknowledge the entry as part of the stop entry sequence. Finally, it can be used to disable selected bus masters or slaves that should remain inactive during a DMA wakeup.

If the Flash is not being accessed during WAIT and PSTOP modes, then the Flash Doze mode can be used to reduce power consumption, at the expense of a slightly longer wakeup when executing code and vectors from Flash. It can also be used to reduce power consumption during Compute Operation when executing code and vectors from SRAM.

7.3 Power modes

The power management controller (PMC) provides multiple power options to allow the user to optimize power consumption for the level of functionality needed.

Depending on the stop requirements of the user application, a variety of stop modes are available that provide state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. The following table compares the various power modes available.

For each run mode there is a corresponding wait and stop mode. Wait modes are similar to Arm sleep modes. Stop modes (VLPS, STOP) are similar to Arm sleep deep mode. The very low power run (VLPR) operating mode can drastically reduce runtime power when the maximum bus frequency is not required to handle the application needs.

The three primary modes of operation are run, wait and stop. The WFI instruction invokes both wait and stop modes for the chip. The primary modes are augmented in a number of ways to provide lower power based on application needs.

Table 7-1. Chip power modes

Chip mode	Description	Core mode	Normal recovery method
Normal run	Default mode out of reset; on-chip voltage regulator is on.	Run	-
High Speed run	Allows maximum performance of the chip. In this state the chip is able to operate at a faster frequency compared to normal mode.	Run	-
Normal Wait - via WFI	Allows peripherals to function while the core is in sleep mode, reducing power. NVIC remains sensitive to interrupts; peripherals continue to be clocked.	Sleep	Interrupt
Normal Stop - via WFI	Places chip in static state. Lowest power mode that retains all registers while maintaining LVD protection. NVIC is disabled; AWIC is used to wake up from interrupt; peripheral clocks are stopped.	Sleep Deep	Interrupt

Table continues on the next page...

Table 7-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
VLPR (Very Low Power Run)	On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. Reduced frequency Flash access mode (1 MHz); LVD off; internal oscillator provides a low power 4 MHz source for the core, the bus and the peripheral clocks.	Run	-
VLPW (Very Low Power Wait) -via WFI	Same as VLPR but with the core in sleep mode to further reduce power; NVIC remains sensitive to interrupts (FCLK = ON). On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency.	Sleep	Interrupt
VLPS (Very Low Power Stop)-via WFI	Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional. Peripheral clocks are stopped, but LPTimer, CMP can be used. NVIC is disabled (FCLK = OFF); AWIC is used to wake up from interrupt. On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. All SRAM is operating (content retained and I/O states held).	Sleep Deep	Interrupt
VLLS3 (Very Low Leakage Stop3)	Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, CMP can be used. NVIC is disabled; LLWU is used to wake up. SRAM_U and SRAM_L remain powered on (content retained and I/O states held).	Sleep Deep	Wakeup Reset[1]
VLLS2 (Very Low Leakage Stop2)	Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, CMP can be used. NVIC is disabled; LLWU is used to wake up. SRAM_L is powered off. A portion of SRAM_U remains powered on (content retained and I/O states held).	Sleep Deep	Wakeup Reset[1]
VLLS1 (Very Low Leakage Stop1)	Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, CMP can be used. NVIC is disabled; LLWU is used to wake up. All of SRAM_U and SRAM_L are powered off. The 32-byte system register file remain powered for customer-critical data.	Sleep Deep	Wakeup Reset[1]
VLLS0 (Very Low Leakage Stop0)	Most peripherals are disabled (with clocks stopped), but LLWU can be used. NVIC is disabled; LLWU is used to wake up	Sleep Deep	Wakeup Reset[1]

1. Follows the reset flow with the LLWU interrupt flag set for the NVIC.

7.4 Module Operation in Low Power Modes

The following table illustrates the functionality of each module while the chip is in each of the low power modes. (Debug modules are discussed separately; see Debug in Low Power Modes.) Number ratings (such as 2 MHz and 1 Mbps) represent the maximum frequencies or maximum data rates per mode. Also, these terms are used:

- FF = Full functionality. In VLPR and VLPW the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- static = Module register states and associated memories are retained.
- powered = Memory is powered to retain contents.
- low power = Memory is powered to retain contents in a lower power state.
- OFF = Modules are powered off; module is in reset state upon wakeup.
- wakeup = Modules can serve as a wakeup source for the chip.

Table 7-2. Module operation in low power modes

Modules	Stop	VLPR	VLPW	VLPS	VLLSx
Core modules					
NVIC	static	FF	FF	static	OFF
System modules					
Mode Controller	FF	FF	FF	FF	FF
LLWU[1]	static	static	static	static	FF
Regulator	ON	low power	low power	low power	low power in VLLS2/3, OFF in VLLS0/1
LVD	ON	disabled	disabled	disabled	disabled
Brown-out Detection	ON	ON	ON	ON	ON in VLLS1/2/3, optionally disabled in VLLS0
DMA	static	FF	FF	static	OFF
Watchdog	FF	FF	FF	FF	OFF
EWM	static	FF	static	static	OFF
Clocks					
1kHz LPO	ON	ON	ON	ON	ON in VLLS1/2/3, OFF in VLLS0
System oscillator (OSC)	OSCERCLK optional	OSCERCLK max of 4MHz crystal	OSCERCLK max of 4MHz crystal	OSCERCLK max of 4MHz crystal	limited to low range/low power in VLLS1/2/3, OFF in VLLS0
MCG	static - MCGIRCLK optional; PLL optionally on but gated	4 MHz IRC	4 MHz IRC	static - MCGIRCLK optional (4 MHz IRC only)	OFF
Core clock	OFF	4 MHz max	OFF	OFF	OFF
System clock	OFF	4 MHz max	4 MHz max	OFF	OFF
Fast peripheral clock	OFF	4 MHz max	4 MHz max	OFF	OFF
Memory and memory interfaces					
Flash/Bus clock	powered	1 MHz max access - no program/erase	low power	low power	OFF
SRAM_U	low power	low power	low power	low power	low power in VLLS3,2; otherwise OFF

Table continues on the next page...

Table 7-2. Module operation in low power modes (continued)

Modules	Stop	VLPR	VLPW	VLPS	VLLSx
All of SRAM_L	low power	low power	low power	low power	low power in VLLS3; otherwise OFF
Register files	powered	powered	powered	powered	powered
Communication interfaces					
UART	static, wakeup on edge	125 kbps	125 kbps	static, wakeup on edge	OFF
SPI	static	1 Mbps	1 Mbps	static	OFF
I^2C	static, address match wakeup	100 kbps	100 kbps	static, address match wakeup	OFF
FlexCAN	wakeup	256 kbps	256 kbps	wakeup	OFF
Security					
CRC	static	FF	FF	static	OFF
Timers					
FTM0	static	FF	FF	static	OFF
FTM1	static	FF	FF	static	OFF
FTM3	static	FF	FF	static	OFF
eFlexPWM	static	FF	FF	static	OFF
PIT	static	FF	FF	static	OFF
PDB0,1	static	FF	FF	static	OFF
XBARA,B,AOI	static	FF	FF	static	OFF
2.7V Vreg	static	static	static	OFF	OFF
LPTMR	FF	FF	FF	FF	FF
Analog					
12-bit cyclic ADCA & B	IRC4M only	FF	FF	IRC4M only	OFF
CMPs 0,1,2,3[2]	HS or LS level compare	FF	FF	HS or LS level compare	LS compare in VLLS1/2/3, OFF in VLLS0
6-bit DAC	static	FF	FF	static	static, OFF in VLLS0
NanoEdge	OFF	OFF	OFF	OFF	OFF
12-bit DAC	static	FF	FF	static	static
Human-machine interfaces					
GPIO	wakeup	FF	FF	wakeup	OFF, pins latched

1. Using the LLWU module, the external pins available for this chip do not require the associated peripheral function to be enabled. It only requires the function controlling the pin (GPIO or peripheral) to be configured as an input to allow a transition to occur to the LLWU.
2. CMP in stop or VLPS supports high speed or low speed external pin to pin or external pin to DAC compares. CMP in VLLSx only supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled & filtered modes of operation are not available while in stop, VLPS, or VLLSx modes.

7.5 Power modes shutdown sequencing

When entering stop or other low-power modes, the clocks are shut off in an orderly sequence to safely place the chip in the targeted low-power state. All low-power entry sequences are initiated by the core executing an WFI instruction. The Arm core's outputs, SLEEPDEEP and SLEEPING, trigger entry to the various low-power modes:

- System level wait and VLPW modes equate to: SLEEPING & $\overline{\text{SLEEPDEEP}}$
- All other low power modes equate to: SLEEPING & SLEEPDEEP

When entering the non-wait modes, the chip performs the following sequence:

- Shuts off Core Clock and System Clock to the Arm Cortex-M4 core immediately.
- Polls stop acknowledge indications from the non-core crossbar masters (DMA), supporting peripherals (SPI, PIT) and the Flash Controller for indications that System Clocks, Bus Clock and/or Flash Clock need to be left enabled to complete a previously initiated operation, effectively stalling entry to the targeted low power mode. When all acknowledges are detected, System Clock, Bus Clock and Flash Clock are turned off at the same time.
- MCG and Mode Controller shut off clock sources and/or the internal supplies driven from the on-chip regulator as defined for the targeted low power mode.

In wait modes, most of the system clocks are not affected by the low power mode entry. The Core Clock to the Arm Cortex-M4 core is shut off. Some modules support stop-in-wait functionality and have their clocks disabled under these configurations.

The debugger modules support a transition from stop, wait, VLPS, and VLPW back to a halted state when the debugger is enabled. This transition is initiated by setting the Debug Request bit in MDM-AP control register. As part of this transition, system clocking is re-established and is equivalent to normal run/VLPR mode clocking configuration.

7.6 Clock Gating

To conserve power, the clocks to most modules can be turned off using the SCGCx registers in the SIM module. These bits are cleared after any reset, which disables the clock to the corresponding module. Prior to initializing a module, set the corresponding bit in the SCGCx register to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution and SIM chapters.

7.7 Flash Program Restrictions

The flash memory on this device should not be programmed or erased while operating in High Speed Run or VLPR power modes.

Chapter 8
Security

8.1 Introduction

This device implements security based on the mode selected from the flash module. The following sections provide an overview of flash security and details the effects of security on non-flash modules.

8.2 Flash Security

The flash module provides security information to the MCU based on the state held by the FSEC[SEC] bits. The MCU, in turn, confirms the security request and limits access to flash resources. During reset, the flash module initializes the FSEC register using data read from the security byte of the flash configuration field.

NOTE

The security features apply only to external accesses via debug. CPU accesses to the flash are not affected by the status of FSEC.

In the unsecured state all flash commands are available to the programming interfaces (JTAG), as well as user code execution of Flash Controller commands. When the flash is secured (FSEC[SEC] = 00, 01, or 11) , programmer interfaces are only allowed to launch mass erase operations and have no access to memory locations, until after a mass erase is completed successfully .

Further information regarding the flash security options and enabling/disabling flash security is available in the Flash Memory Module.

8.3 Security Interactions with other Modules

The flash security settings are used by the SoC to determine what resources are available. The following sections describe the interactions between modules and the flash security settings or the impact that the flash security has on non-flash modules.

8.3.1 Security Interactions with Debug

When flash security is active the JTAG port cannot access the memory resources of the MCU. Boundary scan chain operations work, but debugging capabilities are disabled so that the debug port cannot read flash contents.

Although most debug functions are disabled, the debugger can write to the Flash Mass Erase in Progress bit in the MDM-AP Control register to trigger a mass erase (Erase All Blocks) command. A mass erase via the debugger is allowed even when some memory locations are protected.

When mass erase is disabled, mass erase via the debugger or any other resource is blocked.

Chapter 9
Debug

9.1 Introduction

This device's debug is based on the Arm coresight architecture and is configured in each device to provide the maximum flexibility as allowed by the restrictions of the pinout and other available resources.

Four debug interfaces are supported:

- IEEE 1149.1 JTAG
- IEEE 1149.7 JTAG (cJTAG)
- Serial Wire Debug (SWD)
- Arm Real-Time Trace Interface

The basic Cortex-M4 debug architecture is very flexible. The following diagram shows the topology of the core debug architecture and its components.

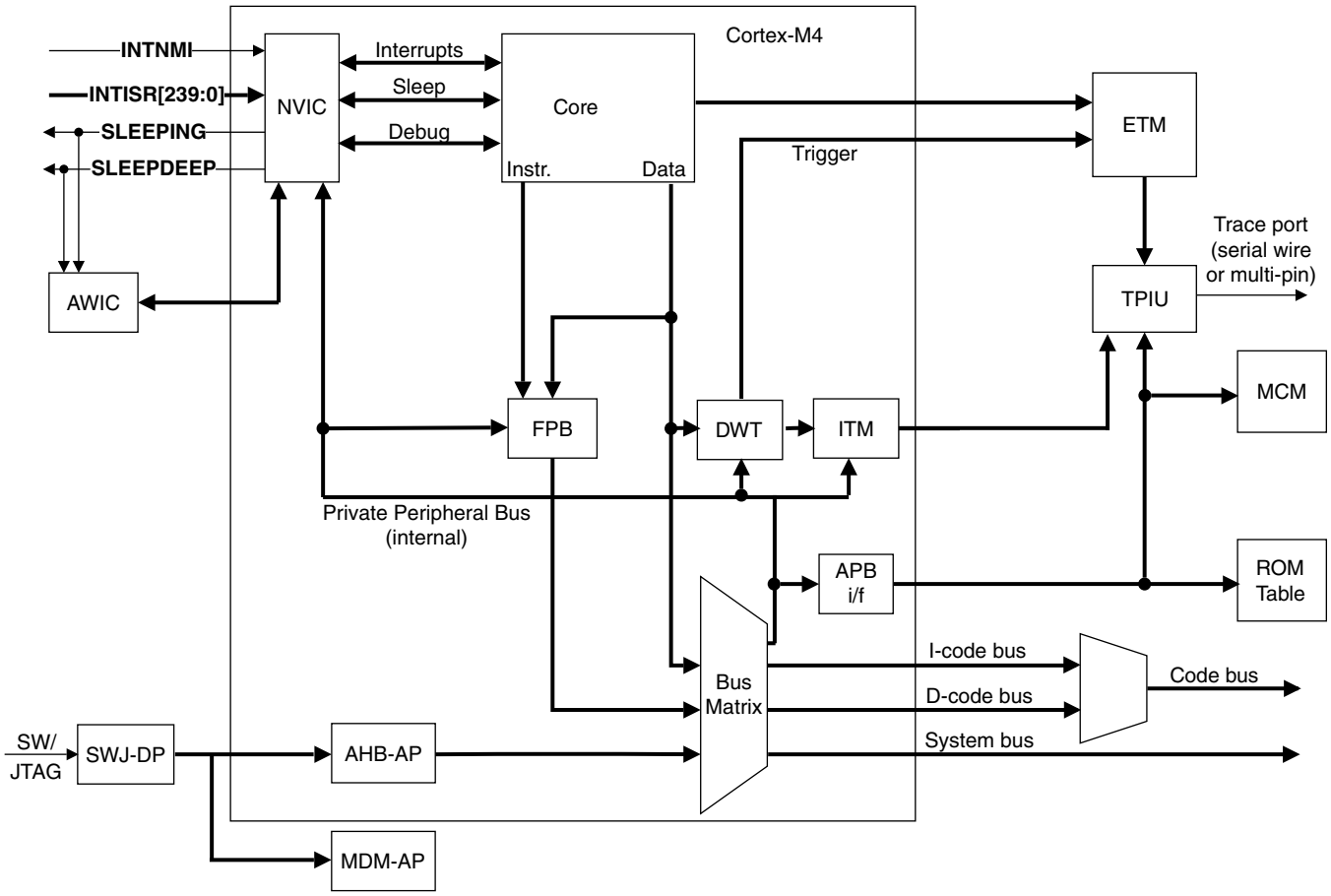


Figure 9-1. Cortex-M4 Debug Topology

The following table presents a brief description of each one of the debug components.

Table 9-1. Debug Components Description

Module	Description
SWJ-DP	Modified Debug Port with support for SWD, JTAG
AHB-AP	AHB Master Interface from JTAG to debug module and SOC system memory maps
JTAG-AP	Bridge to DFT/BIST resources.
ROM Table	Identifies which debug IP is available.
Core Debug	Singlestep, Register Access, Run, Core Status
CoreSight Trace Funnel (not shown in figure)	The CSTF combines multiple trace streams onto a single ATB bus.
CoreSight Trace Replicator (not shown in figure)	The ATB replicator enables two trace sinks to be wired together and operate from the same incoming trace stream.
CoreSight ETB (Embedded Trace Buffer)	Memory mapped buffer used to store trace data.
ITM	S/W Instrumentation Messaging + Simple Data Trace Messaging + Watchpoint Messaging
DWT (Data and Address Watchpoints)	4 data and address watchpoints (configurable for less, but 4 seems to be accepted)

Table continues on the next page...

Table 9-1. Debug Components Description (continued)

Module	Description
FPB (Flash Patch and Breakpoints)	The FPB implements hardware breakpoints and patches code and data from code space to system space. The FPB unit contains two literal comparators for matching against literal loads from Code space, and remapping to a corresponding area in System space. The FBP also contains six instruction comparators for matching against instruction fetches from Code space, and remapping to a corresponding area in System space. Alternatively, the six instruction comparators can individually configure the comparators to return a Breakpoint Instruction (BKPT) to the processor core on a match, so providing hardware breakpoint capability.
TPIU (Trace Port Inteface Unit)	Asynchronous Mode (1-pin) = TRACE_SWO (available on JTAG_TDO)
MCM (Miscellaneous Control Module)	The MCM provides miscellaneous control functions including control of the ETB and trace path switching.

9.1.1 References

For more information on Arm debug components, see these documents:

- Armv7-M Architecture Reference Manual
- Arm Debug Interface v5.1
- Arm CoreSight Architecture Specification

9.2 The Debug Port

The debug port comes out of reset in standard JTAG mode and is switched into either cJTAG or SWD mode by the following sequences. Once the mode has been changed, unused debug pins can be reassigned to any of their alternative muxed functions.

9.2.1 JTAG-to-SWD change sequence

1. Send more than 50 TCK cycles with TMS (SWDIO) =1
2. Send the 16-bit sequence on TMS (SWDIO) = 0111_1001_1110_0111 (MSB transmitted first)
3. Send more than 50 TCK cycles with TMS (SWDIO) =1

NOTE

See the Arm documentation for the CoreSight DAP Lite for restrictions.

9.3 Debug Port Pin Descriptions

The debug port pins default after POR to their JTAG functionality with the exception of JTAG_TRST_b and can be later reassigned to their alternate functionalities. In cJTAG and SWD modes JTAG_TDI and JTAG_TRST_b can be configured to alternate GPIO functions.

Table 9-2. Debug port pins

Pin Name	JTAG Debug Port		cJTAG Debug Port		SWD Debug Port		Internal Pull-up\Down
	Type	Description	Type	Description	Type	Description	
JTAG_TMS	I/O	JTAG Test Mode Selection	I/O	cJTAG Data	I/O	Serial Wire Data	Pull-up
JTAG_TCLK	I	JTAG Test Clock	I	cJTAG Clock	I	Serial Wire Clock	Pull-down
JTAG_TDI	I	JTAG Test Data Input	-	-	-	-	Pull-up
JTAG_TDO/ TRACE_SWO	O	JTAG Test Data Output	O	Trace output over a single pin	O	Trace output over a single pin	N/C
JTAG_TRST_b	I	JTAG Reset	I	cJTAG Reset	-	-	Pull-up

9.4 JTAG status and control registers

Through the Arm Debug Access Port (DAP), the debugger has access to the status and control elements, implemented as registers on the DAP bus as shown in the following figure. These registers provide additional control and status for low power mode recovery and typical run-control scenarios. The status register bits also provide a means for the debugger to get updated status of the core without having to initiate a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

It is important to note that these DAP control and status registers are not memory mapped within the system memory map and are only accessible via the Debug Access Port (DAP) using JTAG or cJTAG. The MDM-AP is accessible as Debug Access Port 1 with the available registers shown in the table below.

Table 9-3. MDM-AP Register Summary

Address	Register	Description
0x0100_0000	Status	See MDM-AP Status Register
0x0100_0004	Control	See MDM-AP Control Register
0x0100_00FC	ID	Read-only identification register that always reads as 0x001C_0000

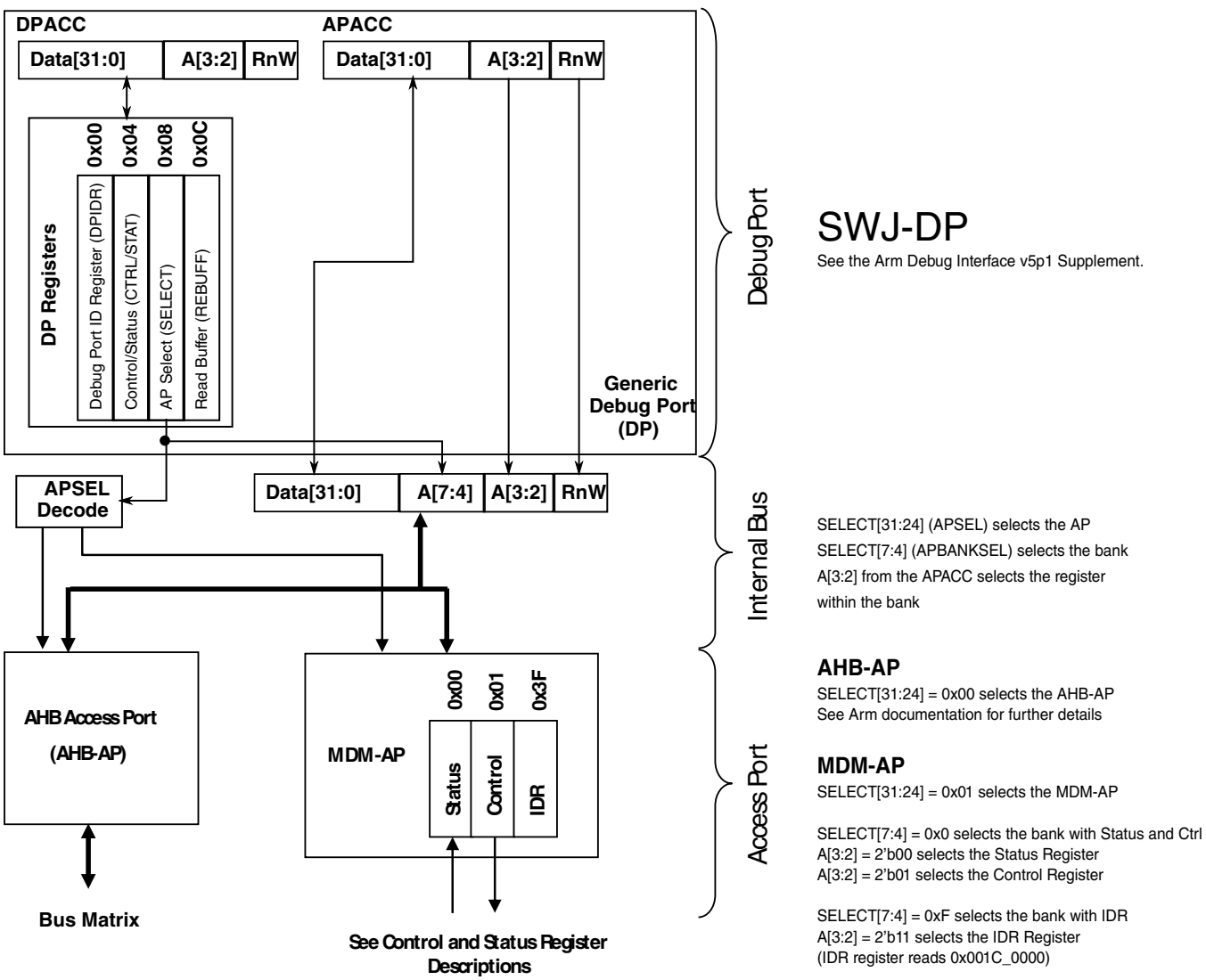


Figure 9-2. MDM AP Addressing

9.4.1 MDM-AP Control Register

Table 9-4. MDM-AP Control register assignments

Bit	Name	Secure[1]	Description
0	Flash Mass Erase in Progress	Y	Set to cause mass erase. Cleared by hardware after mass erase operation completes. When mass erase is disabled (via MEEN settings), the erase request does not occur and the Flash Mass Erase in Progress bit continues to assert until the next system reset.
1	Debug Disable	N	Set to disable debug. Once it is set, the MDM-AP register cannot be written and it can only be cleared by debug reset.
2	Debug Request	N	Set to force the Core to halt. If the Core is in a stop or wait mode, this bit can be used to wakeup the core and transition to a halted state.
3	System Reset Request	N	Set to force a system reset. The system remains held in reset until this bit is cleared.
4	Core Hold Reset	N	Configuration bit to control Core operation at the end of system reset sequencing. 0 Normal operation - release the Core from reset along with the rest of the system at the end of system reset sequencing. 1 Suspend operation - hold the Core in reset at the end of reset sequencing. Once the system enters this suspended state, clearing this control bit immediately releases the Core from reset and CPU operation begins.
5	VLLSx Debug Request (VLLDBGREQ)	N	Set to configure the system to be held in reset after the next recovery from a VLLSx mode. This bit holds the in reset when VLLSx modes are exited to allow the debugger time to re-initialize debug IP before the debug session continues. The Mode Controller captures this bit logic on entry to VLLSx modes. Upon exit from VLLSx modes, the Mode Controller holds the in reset until VLLDBGACK is asserted. The VLLDBGREQ bit clears automatically due to the POR reset generated as part of the VLLSx recovery.
6	VLLSx Debug Acknowledge (VLLDBGACK)	N	Set to release a being held in reset following a VLLSx recovery This bit is used by the debugger to release the system reset when it is being held on VLLSx mode exit. The debugger re-initializes all debug IP and then assert this control bit to allow the Mode Controller to release the from reset and allow CPU operation to begin. The VLLDBGACK bit is cleared by the debugger or can be left set because it clears automatically due to the POR reset generated as part of the next VLLSx recovery.
7	VLLSx Status Acknowledge	N	Set this bit to acknowledge the DAP VLLS Status bit has been read. This acknowledge automatically clears the status bits. This bit is used by the debugger to clear the stickyVLLSx mode entry status bits. This bit is asserted and cleared by the debugger.

1. Command available in secure mode

9.4.2 MDM-AP Status Register

Table 9-5. MDM-AP Status register assignments

Bit	Name	Description
0	Flash Mass Erase Acknowledge	The Flash Mass Erase Acknowledge bit is cleared after POR or debug reset. The bit is also cleared at launch of a mass erase command due to write of Flash Mass Erase in Progress bit in MDM AP Control Register. The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation. When mass erase is disabled (via MEEN settings), an erase request due to seting of Flash Mass Erase in Progress bit is not acknowledged.
1	Flash Ready	Indicate Flash has been initialized and debugger can be configured even if system is continuing to be held in reset via the debugger.
2	System Security	Indicates the security state. When secure, the debugger does not have access to the system bus or any memory mapped peripherals. This bit indicates when the part is locked and no system bus access is possible.
3	System Reset	Indicates the system reset state. 0 System is in reset 1 System is not in reset
4	Reserved	
5	Mass Erase Enable	Indicates if the MCU can be mass erased or not 0 Mass erase is disabled 1 Mass erase is enabled
6	Backdoor Access Key Enable	Indicates if the MCU has the backdoor access key enabled. 0 Disabled 1 Enabled
7	LP Enabled	Decode of LPLLSM control bits to indicate that VLPS, LLS, or VLLSx are the selected power mode the next time the Arm Core enters Deep Sleep. 0 Low Power Stop Mode is not enabled 1 Low Power Stop Mode is enabled Usage intended for debug operation in which Run to VLPS is attempted. Per debug definition, the system actually enters the Stop state. A debugger should interpret deep sleep indication (with SLEEPDEEP and SLEEPING asserted), in conjuntion with this bit asserted as the debugger-VLPS status indication.
8	Very Low Power Mode	Indicates current power mode is VLPx. This bit is not ‘sticky’ and should always represent whether VLPx is enabled or not. This bit is used to throttle JTAG TCK frequency up/down.
9	Reserved	Always read 0.
10	VLLSx Modes Exit	This bit indicates an exit from VLLSx mode has occurred. The debugger loses communication while the system is in VLLSx (including access to this register). Once communication is reestablished, this bit indicates that the system had been in VLLSx. Since the debug modules lose their state during VLLSx modes, they need to be reconfigured.

Table continues on the next page...

Table 9-5. MDM-AP Status register assignments (continued)

Bit	Name	Description
		This bit is set during the VLLSx recovery sequence. The VLLSx Mode Exit bit is held until the debugger has had a chance to recognize that a VLLS mode was exited and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register.
11 – 15	Reserved for future use	Always read 0.
16	Core Halted	Indicates the Core has entered debug halt mode
17	Core SLEEPDEEP	Indicates the Core has entered a low power mode
18	Core SLEEPING	SLEEPING==1 and SLEEPDEEP==0 indicates wait or VLPW mode. SLEEPING==0 and SLEEPDEEP==1 indicates stop or VLPS mode.
19 – 31	Reserved for future use	Always read 0.

9.5 Debug Resets

The debug system receives the following sources of reset:

- JTAG_TRST_b from an external signal. This signal is optional and may not be available in all packages.
- Debug reset (CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register) in the TCLK domain that allows the debugger to reset the debug logic.
- TRST asserted via the cJTAG escape command.
- System POR reset

Conversely the debug system is capable of generating system reset using the following mechanism:

- A system reset in the DAP control register which allows the debugger to hold the system in reset.
- SYSRESETREQ bit in the NVIC application interrupt and reset control register
- A system reset in the DAP control register which allows the debugger to hold the Core in reset.

9.6 AHB-AP

AHB-AP provides the debugger access to all memory and registers in the system, including processor registers through the NVIC. System access is independent of the processor status. AHB-AP does not do back-to-back transactions on the bus, so all transactions are non-sequential. AHB-AP can perform unaligned and bit-band transactions. AHB-AP transactions bypass the FPB, so the FPB cannot remap AHB-AP

transactions. SWJ/SW-DP-initiated transaction aborts drive an AHB-AP-supported sideband signal called HABORT. This signal is driven into the Bus Matrix, which resets the Bus Matrix state, so that AHB-AP can access the Private Peripheral Bus for last ditch debugging such as read/stop/reset the core. AHB-AP transactions are little endian.

For a short period at the start of a system reset event the system security status is being determined and debugger access to all AHB-AP transactions is blocked. The MDM-AP Status register is accessible and can be monitored to determine when this initial period is completed. After this initial period, if system reset is held via assertion of the RESET pin, the debugger has access via the bus matrix to the private peripheral bus to configure the debug IP even while system reset is asserted. While in system reset, access to other memory and register resources, accessed over the Crossbar Switch, is blocked.

9.7 ITM

The ITM is an application-driven trace source that supports printf style debugging to trace Operating System (OS) and application events, and emits diagnostic system information. The ITM emits trace information as packets. There are four sources that can generate packets. If multiple sources generate packets at the same time, the ITM arbitrates the order in which packets are output. The four sources in decreasing order of priority are:

1. Software trace -- Software can write directly to ITM stimulus registers. This emits packets.
2. Hardware trace -- The DWT generates these packets, and the ITM emits them.
3. Time stamping -- Timestamps are emitted relative to packets. The ITM contains a 21-bit counter to generate the timestamp. The Cortex-M4 clock or the bitclock rate of the Serial Wire Viewer (SWV) output clocks the counter.
4. Global system timestamping. Timestamps can optionally be generated using a system-wide 48-bit count value.

9.8 TPIU

The TPIU acts as a bridge between the on-chip trace data from the Instrumentation Trace Macrocell (ITM) and Embedded Trace Macrocell (ETM), with separate IDs, to a data stream, encapsulating IDs where required, that is then captured by a Trace Port Analyzer (TPA). The TPIU is specially designed for low-cost debug.

9.9 DWT

The DWT is a unit that performs the following debug functionality:

- It contains four comparators that you can configure as a hardware watchpoint, a PC sampler event trigger, or a data address sampler event trigger. The first comparator, DWT_COMP0, can also compare against the clock cycle counter, CYCCNT. The second comparator, DWT_COMP1, can also be used as a data comparator.
- The DWT contains counters for:
  - Clock cycles (CYCCNT)
  - Folded instructions
  - Load store unit (LSU) operations
  - Sleep cycles
  - CPI (all instruction cycles except for the first cycle)
  - Interrupt overhead

NOTE

An event is emitted each time a counter overflows.

- The DWT can be configured to emit PC samples at defined intervals, and to emit interrupt event information.

9.10 Debug in Low Power Modes

In low power modes in which the debug modules are kept static or powered off, the debugger cannot gather any debug data for the duration of the low power mode. In the case that the debugger is held static, the debug port returns to full functionality as soon as the low power mode exits and the system returns to a state with active debug. In the case that the debugger logic is powered off, the debugger is reset on recovery and must be reconfigured once the low power mode is exited.

Power mode entry logic monitors Debug Power Up and System Power Up signals from the debug port as indications that a debugger is active. These signals can be changed in RUN, VLPR, WAIT and VLPW. If the debug signal is active and the system attempts to enter stop or VLPS, FCLK continues to run to support core register access. In these modes in which FCLK is left active the debug modules have access to core registers but not to system memory resources accessed via the crossbar.

With debug enabled, transitions from Run directly to VLPS are not allowed and result in the system entering Stop mode instead. Status bits within the MDM-AP Status register can be evaluated to determine this pseudo-VLPS state. Note with the debug enabled, transitions from Run--> VLPR --> VLPS are still possible but also result in the system entering Stop mode instead.

In VLLS mode all debug modules are powered off and reset at wakeup.

Going into a VLLSx mode causes all the debug controls and settings to be reset. To give time to the debugger to sync up with the HW, the MDM-AP Control register can be configured hold the system in reset on recovery so that the debugger can regain control and reconfigure debug logic prior to the system exiting reset and resuming operation.

9.10.1 Debug Module State in Low Power Modes

The following table shows the state of the debug modules in low power modes. These terms are used:

- FF = Full functionality. In VLPR and VLPW the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- static = Module register states and associated memories are retained.
- OFF = Modules are powered off; module is in reset state upon wakeup.

Table 9-6. Debug Module State in Low Power Modes

Module	STOP	VLPR	VLPW	VLPS	LLS	VLLSx
Debug Port	FF	FF	FF	OFF	static	OFF
AHB-AP	FF	FF	FF	OFF	static	OFF
ITM	FF	FF	FF	OFF	static	OFF
TPIU	FF	FF	FF	OFF	static	OFF
DWT	FF	FF	FF	OFF	static	OFF

9.11 Debug & Security

When security is enabled (FSEC[SEC] != 10), the debug port capabilities are limited in order to prevent exploitation of secure data. In the secure state the debugger still has access to the MDM-AP Status Register and can determine the current security state of the device. In the case of a secure device, the debugger also has the capability of performing a mass erase operation via writes to the MDM-AP Control Register. In the case of a secure device that has mass erase disabled (FSEC[MEEN] = 10), attempts to mass erase via the debug interface are blocked.

Chapter 10
Reset and Boot

10.1 Introduction

The following reset sources are supported in this MCU:

Table 10-1. Reset sources

Reset sources	Description
POR reset	• Power-on reset (POR)
System resets	• External pin reset (PIN) • Low-voltage detect (LVD) • Computer operating properly (Gen2008) watchdog reset • Low leakage wakeup (LLWU) reset • Multipurpose clock generator loss of clock (LOC) reset • Multipurpose clock generator loss of Lock (LOL) reset • Software reset (SW) • Lockup reset (LOCKUP) • MDM DAP system reset
Debug reset	• JTAG reset • nTRST reset

Each of the system reset sources has an associated bit in the system reset status (SRS) registers. See the Reset Control Module for register details.

The MCU can exit and reset in functional mode where the CPU is executing code (default) or the CPU is in a debug halted state. There are several boot options that can be configured. See Boot information for more details.

10.2 Reset

This section discusses basic reset mechanisms and sources. Some modules that cause resets can be configured to cause interrupts instead. Consult the individual peripheral chapters for more information.

10.2.1 Power-on reset (POR)

When power is initially applied to the MCU or when the supply voltage drops below the power-on reset re-arm voltage level (V_{POR}), the POR circuit causes a POR reset condition.

As the supply voltage rises, the LVD circuit holds the MCU in reset until the supply has risen above the LVD low threshold (V_{LVDL}). The POR and LVD bits in SRSL register are set following a POR.

10.2.2 System resets

Resetting the MCU provides a way to start processing from a known set of initial conditions. System reset begins with the on-chip regulator in full regulation and system clocking generation from an internal reference. When the processor exits reset, it performs the following:

- Reads the start SP (SP_main) from vector-table offset 0
- Reads the start PC from vector-table offset 4
- LR is set to 0xFFFF_FFFF

The on-chip peripheral modules are disabled and the non-analog I/O pins are initially configured as disabled. The pins with analog functions assigned to them default to their analog function after reset.

During and following a reset, the JTAG pins have their associated input pins configured as:

- TDI in pull-up (PU)
- TCK in pull-down (PD)
- TMS in PU

and associated output pin configured as:

- TDO with no pull-down or pull-up

Note that the nTRST signal is initially configured as disabled, however once configured to its JTAG functionality its associated input pin is configured as:

- nTRST in PU

10.2.2.1 External pin reset (PIN)

On this device, $\overline{\text{RESET}}$ is a dedicated pin. This pin is open drain and has an internal pullup device. Asserting $\overline{\text{RESET}}$ wakes the device from any mode. During a pin reset, the SRSL[PIN] bit is set.

10.2.2.1.1 Reset pin filter

The $\overline{\text{RESET}}$ pin filter supports filtering from both the 1 kHz LPO clock and the bus clock. A separate filter is implemented for each clock source. In stop and VLPS mode operation, this logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected. In low leakage stop modes, a separate LPO filter in the LLWU can continue filtering the $\overline{\text{RESET}}$ pin.

The RPFC[RSTFLTSS], RPFC[RSTFLTSRW], and RPFW[RSTFLTSEL] fields in the reset control (RCM) register set control this functionality; see the RCM chapter. The filters are asynchronously reset by Chip POR. The reset value for each filter assumes the RESET pin is negated.

The two clock options for the $\overline{\text{RESET}}$ pin filter when the chip is not in low leakage modes are the LPO (1 kHz) and bus clock. For low leakage modes VLLS3, VLLS2, VLLS1, VLLS0, the LLWU provides control (in the LLWU_RST register) of an optional fixed digital filter running the LPO. When entering VLLS0, the RESET pin filter is disabled and bypassed.

The LPO filter has a fixed filter value of 3. Due to a synchronizer on the input data, there is also some associated latency (2 cycles). As a result, 5 cycles are required to complete a transition from low to high or high to low.

The bus filter initializes to off (logic 1) when the bus filter is not enabled. The bus clock is used when the filter selects bus clock, and the number of counts is controlled by the RCM's RPFW[RSTFLTSEL] field.

10.2.2.2 Low-voltage detect (LVD) reset

This device includes a system to protect against low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system is comprised of a power-on reset (POR) circuit and a low-voltage detect (LVD) circuit with a user-selectable trip voltage, either high (V_{LVDH}) or low (V_{LVDL}). The trip voltage is selected by the LVDSC1[LVDV] bits. The LVD system is always enabled in normal run, wait, and stop modes. The LVD system is disabled in VLPx, LLSx, and VLLSx modes. Refer to Power Management Controller (PMC) chapter for more details.

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting LVDSC1[LVDRE]. After an LVD reset has occurred, the LVD system holds the MCU in reset until the supply voltage rises above the low voltage detection threshold.

RCM_SRS0[LVD] is set following an LVD reset or POR.

10.2.2.3 Computer operating properly (COP) watchdog reset

The watchdog timer monitors the operation of the system by expecting periodic communication from the software, generally known as servicing (or refreshing) the watchdog. If this periodic refreshing does not occur, the watchdog issues a system reset. The COP reset causes RCM_SRS0[COP] to set.

10.2.2.4 Low leakage wakeup (LLWU) reset

The LLWU allows up to 16 external pins, the $\overline{\text{RESET}}$ pin, and up to seven internal peripherals to wake the MCU from LLS and VLLSx power modes. The LLWU module is only functional in LLS and VLLSx power modes. In both these modes, LLS mode exits via $\overline{\text{RESET}}$ pin and any VLLS mode exits via a wakeup or reset event, RCM_SRS0[WAKEUP] in mode controller module is set indicating the low leakage mode was active prior to the last system reset flow. Using the $\overline{\text{RESET}}$ pin to trigger an exit from LLS or VLLS results in RCM_SRS0[PIN] being set as well. Refer to the mode controller chapter for more details.

After a system reset, the LLWU retains the flags to indicate the source of the last wakeup until the user clears them.

NOTE

Pin wakeup and error condition flags are cleared in the LLWU and module wakeup flags are required to be cleared in the peripheral module. Refer to the individual peripheral specifications for more information.

10.2.2.5 Multipurpose clock generator loss-of-clock (LOC) reset

The MCG includes a clock monitor. The clock monitor resets the device when the following conditions are met:

- The clock monitor is enabled (MCG_C6[CME] = 1)
- The MCG's external reference clock falls outside of the expected frequency range, depending on the MCG_C2[RANGE] bit

RCM_SRS0[LOC] is set to indicate the error.

10.2.2.6 Software reset (SW)

The SYSRESETREQ bit in the NVIC application interrupt and reset control register can be set to force a software reset on the device. (See Arm's NVIC documentation for the full description of the register fields, especially the VECTKEY field requirements.) Setting SYSRESETREQ generates a software reset request. This reset forces a system reset of all major components except for the debug module. A software reset causes RCM_SRS1[SW] to set.

10.2.2.7 Lockup reset (LOCKUP)

The LOCKUP gives immediate indication of seriously errant kernel software. This is the result of the core being locked because of an unrecoverable exception following the activation of the processor’s built in system state protection hardware.

The LOCKUP condition causes a system reset and also causes RCM_SRS1[LOCKUP] to set.

10.2.2.8 MDM-AP system reset request

Set the system reset request bit in the MDM-AP control register to initiate a system reset. This is the primary method for resets via the JTAG interface. The system reset is held until this bit is cleared.

Set the core hold reset bit in the MDM-AP control register to hold the core in reset as the rest of the chip comes out of system reset.

10.2.3 MCU Resets

A variety of resets are generated by the MCU to reset different modules.

10.2.3.1 POR Only

The POR Only reset asserts on the POR reset source only. It resets the PMC.

The POR Only reset also causes all other reset types to occur.

10.2.3.2 Chip POR not VLLS

The Chip POR not VLLS reset asserts on POR and LVD reset sources. It resets parts of the SMC and SIM. It also resets the LPTMR.

The Chip POR not VLLS reset also causes these resets to occur: Chip POR, Chip Reset not VLLS, and Chip Reset (including Early Chip Reset).

10.2.3.3 Chip POR

The Chip POR asserts on POR, LVD, and VLLS Wakeup reset sources. It resets the Reset Pin Filter registers and parts of the SIM and MCG.

The Chip POR also causes the Chip Reset (including Early Chip Reset) to occur.

10.2.3.4 Chip Reset not VLLS

The Chip Reset not VLLS reset asserts on all reset sources except a VLLS Wakeup that does not occur via the RESET_b pin. It resets parts of the SMC, LLWU, and other modules that remain powered during VLLS mode.

The Chip Reset not VLLS reset also causes the Chip Reset (including Early Chip Reset) to occur.

10.2.3.5 Early Chip Reset

The Early Chip Reset asserts on all reset sources. It resets only the flash memory module. It negates before flash memory initialization begins ("earlier" than when the Chip Reset negates).

10.2.3.6 Chip Reset

Chip Reset asserts on all reset sources and only negates after flash initialization has completed and the RESET_b pin has also negated. It resets the remaining modules (the modules not reset by other reset types).

10.2.4 Reset Pin

For all reset sources except a VLLS Wakeup that does not occur via the $\overline{\text{RESET}}$ pin, the $\overline{\text{RESET}}$ pin is driven low by the MCU for at least 128 bus clock cycles and until flash initialization has completed.

After flash initialization has completed, the $\overline{\text{RESET}}$ pin is released, and the internal Chip Reset negates after the $\overline{\text{RESET}}$ pin is pulled high. Keeping the $\overline{\text{RESET}}$ pin asserted externally delays the negation of the internal Chip Reset.

10.2.5 Debug resets

The following sections detail the debug resets available on the device.

10.2.5.1 JTAG reset

The JTAG module generate a system reset when certain IR codes are selected. This functional reset is asserted when EXTEST, HIGHZ and CLAMP instructions are active. The reset source from the JTAG module is released when any other IR code is selected. A JTAG reset causes RCM_SRS1[JTAG] to set.

10.2.5.2 nTRST reset

The nTRST pin causes a reset of the JTAG logic when asserted. Asserting the nTRST pin allows the debugger to gain control of the TAP controller state machine (after exiting LLS or VLLSx) without resetting the state of the debug modules.

The nTRST pin does not cause a system reset.

10.2.5.3 Resetting the Debug subsystem

Use the CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register to reset the debug modules. However, as explained below, using the CDBGRSTREQ bit does not reset all debug-related registers.

CDBGRSTREQ resets the debug-related registers within the following modules:

- SWJ-DP
- AHB-AP
- ATB replicators

- ATB upsizers
- ATB funnels
- TPIU
- MDM-AP (MDM control and status registers)
- MCM

CDBGRSTREQ does not reset the debug-related registers within the following modules:

- CM7 core (core debug registers: DHCSR, DCRSR, DCRDR, DEMCR)
- FPB
- DWT
- ITM
- NVIC
- Crossbar bus switch
- AHB-AP[1]
- Private peripheral bus[1]

10.3 Boot

This section describes the boot sequence, including sources and options.

10.3.1 Boot sources

This device only supports booting from internal flash. Any secondary boot must go through an initialization sequence in flash.

10.3.2 FOPT boot options

The flash option register (FOPT) in the flash memory module allows the user to customize the operation of the MCU at boot time. The register contains read-only bits that are loaded from the NVM's option byte in the flash configuration field. The user can reprogram the option byte in flash to change the FOPT values that are used for subsequent resets. For more details on programming the option byte, refer to the flash memory chapter.

1. CDBGRSTREQ does not affect AHB resources so that debug resources on the private peripheral bus are available during System Reset.

The MCU uses the FOPT register bits to configure the device at reset as shown in the following table.

Table 10-2. Flash Option Register Bit Definitions

Bit Num	Field	Value	Definition
7-6	Reserved	Reserved for future expansion.	
5	FAST_INIT	Select initialization speed on POR, VLLSx, and any system reset.	
		0	Slower initialization. The Flash initialization will be slower with the benefit of reduced average current during this time. The duration of the recovery will be controlled by the clock divider selection determined by the LPBOOT setting.
		1	Fast Initialization.The Flash has faster recoveries at the expense of higher current during these times.
4-3	Reserved	Reserved for future expansion.	
2	NMI_DIS	Enable/disable control for the NMI function.	
		0	NMI interrupts are always blocked. The associated pin continues to default to NMI pin controls with internal pullup enabled.
		1	NMI pin/interrupts reset default to enabled.
1	Reserved	Reserved for future expansion.	
0	LPBOOT	Control the reset value of OUTDIVx values in SIM_CLKDIV1 register. Larger divide value selections produce lower average power consumption during POR, VLLSx recoveries and reset sequencing and after reset exit.	
		0	Low-power boot: OUTDIVx values in SIM_CLKDIV1 register are auto-configured at reset exit for higher divide values that produce lower power consumption at reset exit. • Core and system clock divider (OUTDIV1) and fast peripheral clock divider (OUTDIV2) are 0x7 (divide by 8) • Flash/Bus clock divider (OUTDIV4) are 0xF (divide by 16)
		1	Normal boot: OUTDIVx values in SIM_CLKDIV1 register are auto-configured at reset exit for higher frequency values that produce faster operating frequencies at reset exit. • Core and system clock divider (OUTDIV1) and fast peripheral clock divider (OUTDIV2) are 0x0 (divide by 1 • Flash/Bus clock divider (OUTDIV4) are 0x1 (divide by 2)

10.3.3 Boot sequence

At power up, the on-chip regulator holds the system in a POR state until the input supply is above the POR threshold. The system continues to be held in this static state until the internally regulated supplies have reached a safe operating voltage as determined by the LVD. The Mode Controller reset logic then controls a sequence to exit reset.

1. A system reset is held on internal logic, the $\overline{\text{RESET}}$ pin is driven out low, and the MCG is enabled in its default clocking mode.

2. Required clocks are enabled (Core Clock, System Clock, Flash Clock, and any Bus Clocks that do not have clock gate control).
3. The system reset on internal logic continues to be held, but the Flash Controller is released from reset and begins initialization operation while the Mode Control logic continues to drive the $\overline{\text{RESET}}$ pin out low for a count of ~128 Bus Clock cycles.
4. The $\overline{\text{RESET}}$ pin is released, but the system reset of internal logic continues to be held until the Flash Controller finishes initialization. .
5. When Flash Initialization completes, the $\overline{\text{RESET}}$ pin is observed. If $\overline{\text{RESET}}$ continues to be asserted (an indication of a slow rise time on the $\overline{\text{RESET}}$ pin or external drive in low), the system continues to be held in reset. Once the $\overline{\text{RESET}}$ pin is detected high, the system is released from reset.
6. At release of system reset, clocking is switched to a slow clock if the FOPT[LPBOOT] field in the Flash Memory module is configured for Low Power Boot
7. When the system exits reset, the processor sets up the stack, program counter (PC), and link register (LR). The processor reads the start SP (SP_main) from vector-table offset 0. The core reads the start PC from vector-table offset 4. LR is set to 0xFFFF_FFFF. What happens next depends on the NMI input and the FOPT[NMI_DIS] field in the Flash Memory module:
    - If the NMI input is high or the NMI function is disabled in the NMI_DIS field, the CPU begins execution at the PC location.
    - If the NMI input is low and the NMI function is enabled in the NMI_DIS field, this results in an NMI interrupt. The processor executes an Exception Entry and reads the NMI interrupt handler address from vector-table offset 8. The CPU begins execution at the NMI interrupt handler.

Subsequent system resets follow this reset flow beginning with the step where system clocks are enabled.

Chapter 11
Signal Multiplexing

11.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. This chapter illustrates which of this device's signals are multiplexed on which external pin.

The Port Control block controls which signal is present on the external pin. Reference that chapter to find which register controls the operation of a specific pin.

11.2 Port control and interrupt module features

- 32-pin ports

NOTE

Not all pins are available on the device. See the following section for details.

- Each 32-pin port is assigned one interrupt.

Table 11-1. Ports summary

Feature	Port A	Port B	Port C	Port D	Port E
Pull select control	Yes	Yes	Yes	Yes	Yes
Pull select at reset	PTA1/PTA2/PTA3/ PTA4/PTA5=Pull up, Others=Pull down	Pull down	Pull down	Pull down	Pull down
Pull enable control	Yes	Yes	Yes	Yes	Yes
Pull enable at reset	PTA0/PTA1/PTA2/ PTA3/PTA4/ PTA5=Enabled; Others=Disabled	Disabled	Disabled	Disabled	Disabled

Table continues on the next page...

Table 11-1. Ports summary (continued)

Feature	Port A	Port B	Port C	Port D	Port E
Slew rate enable control	Yes	Yes	Yes	Yes	Yes
Slew rate enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Passive filter enable control	PTA4=Yes; Others=No	No	No	No	No
Passive filter enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Open drain enable control[1]	Yes, pseudo open-drain capability	Yes, pseudo open-drain capability	Yes. PTC6, PTC7 are true open drain, others are pseudo open-drain capability	Yes, pseudo open-drain capability	Yes, pseudo open-drain capability
Open drain enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Drive strength enable control	No	PTB0/PTB1 only	PTC3/PTC4 only	PTD4/PTD5/PTD6/ PTD7 only	No
Drive strength enable at reset	Disabled	Disabled	Disabled	Disabled	Disabled
Pin mux control	Yes	Yes	Yes	Yes	Yes
Pin mux at reset	PTA0/PTA1/PTA2/ PTA3/PTA4=ALT7; Others=ALT0	ALT0	ALT0	ALT0	ALT0
Lock bit	Yes	Yes	Yes	Yes	Yes
Interrupt and DMA request	Yes	Yes	Yes	Yes	Yes
Digital glitch filter	No	No	No	Yes	No

1.
- Pseudo open-drain is implemented to disable PMOS by setting PORTx_PCRn[ODE] so that external power supply can't be higher than the PAD power supply.
- For true open-drain pad, there is only one NMOS to output '0' and can't be controlled by PORTx_PCRn[ODE] bit.

11.3 Clock gating

The clock to the port control module can be gated on and off using the SCGC5[PORTx] bits in the SIM module. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing the corresponding module, set SCGC5[PORTx] in the SIM module to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution chapter.

11.4 Signal multiplexing constraints

1. A given peripheral function must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
2. To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.

11.5 KV4x Signal Multiplexing and Pin Assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT functionality is available on each pin.

100 LQFP	64 LQFP	48 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
1	1	—	PTE0	ADCB_CH6f	ADCB_CH6f	PTE0		UART1_TX	XBAR0_ OUT10	XBAR0_IN11		
2	2	—	PTE1/ LLWU_P0	ADCB_CH7f	ADCB_CH7f	PTE1/ LLWU_P0		UART1_RX	XBAR0_ OUT11	XBAR0_IN7		
3	—	—	PTE2/ LLWU_P1	ADCB_CH6g	ADCB_CH6g	PTE2/ LLWU_P1		UART1_CTS_ b				
4	—	—	PTE3	ADCB_CH7g	ADCB_CH7g	PTE3		UART1_RTS_ b				
5	—	—	PTE4/ LLWU_P2	DISABLED		PTE4/ LLWU_P2						
6	—	—	PTE5	DISABLED		PTE5					FTM3_CH0	
7	—	—	PTE6/ LLWU_P16	DISABLED		PTE6/ LLWU_P16					FTM3_CH1	
8	3	1	VDD	VDD	VDD							
9	4	2	VSS	VSS	VSS							
10	5	3	PTE16	ADCA_CH0	ADCA_CH0	PTE16	SPI0_PCS0	UART1_TX	FTM_CLKIN0		FTM0_FLT3	
11	6	4	PTE17/ LLWU_P19	ADCA_CH1	ADCA_CH1	PTE17/ LLWU_P19	SPI0_SCK	UART1_RX	FTM_CLKIN1		LPTMR0_ ALT3	
12	7	5	PTE18/ LLWU_P20	ADCB_CH0	ADCB_CH0	PTE18/ LLWU_P20	SPI0_SOUT	UART1_CTS_ b	I2C0_SDA			
13	8	6	PTE19	ADCB_CH1	ADCB_CH1	PTE19	SPI0_SIN	UART1_RTS_ b	I2C0_SCL		CMP3_OUT	
14	—	—	ADCA_CH6a	ADCA_CH6a	ADCA_CH6a							
15	—	—	ADCA_CH7a	ADCA_CH7a	ADCA_CH7a							
16	—	7	PTE20	ADCA_CH6b	ADCA_CH6b	PTE20		FTM1_CH0	UART0_TX			
17	—	8	PTE21	ADCA_CH7b	ADCA_CH7b	PTE21		FTM1_CH1	UART0_RX			
18	9	—	ADCA_CH2	ADCA_CH2	ADCA_CH2							
19	10	—	ADCA_CH3	ADCA_CH3	ADCA_CH3							

100 LQFP	64 LQFP	48 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
20	11	—	ADCA_CH6c	ADCA_CH6c	ADCA_CH6c							
21	12	—	ADCA_CH7c	ADCA_CH7c	ADCA_CH7c							
22	13	9	VDDA	VDDA	VDDA							
23	14	10	VREFH	VREFH	VREFH							
24	15	11	VREFL	VREFL	VREFL							
25	16	12	VSSA	VSSA	VSSA							
26	17	13	PTE29	ADCA_CH4/ CMP1_IN5/ CMP0_IN5	ADCA_CH4/ CMP1_IN5/ CMP0_IN5	PTE29		FTM0_CH2		FTM_CLKIN0		
27	18	14	PTE30	DAC0_OUT/ CMP1_IN3/ ADCA_CH5	DAC0_OUT/ CMP1_IN3/ ADCA_CH5	PTE30		FTM0_CH3		FTM_CLKIN1		
28	19	—	ADCA_CH6d/ CMP0_IN4/ CMP2_IN3	ADCA_CH6d/ CMP0_IN4/ CMP2_IN3	ADCA_CH6d/ CMP0_IN4/ CMP2_IN3							
29	—	—	VSS	VSS	VSS							
30	—	—	VDD	VDD	VDD							
31	20	15	PTE24	ADCB_CH4	ADCB_CH4	PTE24	CAN1_TX	FTM0_CH0	XBAR0_IN2	I2C0_SCL	EWM_OUT_b	XBAR0_OUT4
32	21	16	PTE25/ LLWU_P21	ADCB_CH5	ADCB_CH5	PTE25/ LLWU_P21	CAN1_RX	FTM0_CH1	XBAR0_IN3	I2C0_SDA	EWM_IN	XBAR0_OUT5
33	—	—	PTE26	DISABLED		PTE26						
34	22	17	PTA0	JTAG_TCLK/ SWD_CLK		PTA0	UART0_CTS_b/ UART0_COL_b	FTM0_CH5	XBAR0_IN4	EWM_IN		JTAG_TCLK/ SWD_CLK
35	23	18	PTA1	JTAG_TDI		PTA1	UART0_RX	FTM0_CH6	CMP0_OUT		FTM1_CH1	JTAG_TDI
36	24	19	PTA2	JTAG_TDO/ TRACE_SWO		PTA2	UART0_TX	FTM0_CH7	CMP1_OUT		FTM1_CH0	JTAG_TDO/ TRACE_SWO
37	25	20	PTA3	JTAG_TMS/ SWD_DIO		PTA3	UART0_RTS_b	FTM0_CH0	XBAR0_IN9	EWM_OUT_b	FLEXPWMA_A0	JTAG_TMS/ SWD_DIO
38	26	21	PTA4/ LLWU_P3	NMI_b		PTA4/ LLWU_P3		FTM0_CH1	XBAR0_IN10	FTM0_FLT3	FLEXPWMA_B0	NMI_b
39	27	—	PTA5	DISABLED		PTA5		FTM0_CH2		CMP2_OUT		JTAG_TRST_b
40	—	22	VDD	VDD	VDD							
41	—	23	VSS	VSS	VSS							
42	28	—	PTA12	CMP2_IN0	CMP2_IN0	PTA12	CAN0_TX	FTM1_CH0				FTM1_QD_PHA
43	29	—	PTA13/ LLWU_P4	CMP2_IN1	CMP2_IN1	PTA13/ LLWU_P4	CAN0_RX	FTM1_CH1				FTM1_QD_PHB
44	—	—	PTA14	CMP3_IN0	CMP3_IN0	PTA14	SPI0_PCS0	UART0_TX				
45	—	—	PTA15	CMP3_IN1	CMP3_IN1	PTA15	SPI0_SCK	UART0_RX				
46	—	—	PTA16	CMP3_IN2	CMP3_IN2	PTA16	SPI0_SOUT	UART0_CTS_b/ UART0_COL_b				

100 LQFP	64 LQFP	48 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
47	—	—	PTA17	ADCA_CH7e	ADCA_CH7e	PTA17	SPI0_SIN	UART0_RTS_b				
48	30	—	VDD	VDD	VDD							
49	31	—	VSS	VSS	VSS							
50	32	24	PTA18	EXTAL0	EXTAL0	PTA18	XBAR0_IN7	FTM0_FLT2	FTM_CLKIN0	XBAR0_OUT8	FTM3_CH2	
51	33	25	PTA19	XTAL0	XTAL0	PTA19	XBAR0_IN8	FTM1_FLT0	FTM_CLKIN1	XBAR0_OUT9	LPTMR0_ALT1	
52	34	26	RESET_b	RESET_b	RESET_b							
53	35	27	PTB0/ LLWU_P5	ADCB_CH2	ADCB_CH2	PTB0/ LLWU_P5	I2C0_SCL	FTM1_CH0			FTM1_QD_PHA	UART0_RX
54	36	28	PTB1	ADCB_CH3	ADCB_CH3	PTB1	I2C0_SDA	FTM1_CH1	FTM0_FLT2	EWM_IN	FTM1_QD_PHB	UART0_TX
55	37	29	PTB2	ADCA_CH6e/ CMP2_IN2	ADCA_CH6e/ CMP2_IN2	PTB2	I2C0_SCL	UART0_RTS_b	FTM0_FLT1		FTM0_FLT3	
56	38	30	PTB3	ADCB_CH7e/ CMP3_IN5	ADCB_CH7e/ CMP3_IN5	PTB3	I2C0_SDA	UART0_CTS_b/ UART0_COL_b			FTM0_FLT0	
57	—	—	PTB9	DISABLED		PTB9						
58	—	—	PTB10	ADCB_CH6a	ADCB_CH6a	PTB10					FTM0_FLT1	
59	—	—	PTB11	ADCB_CH7a	ADCB_CH7a	PTB11					FTM0_FLT2	
60	—	—	VSS	VSS	VSS							
61	—	—	VDD	VDD	VDD							
62	39	31	PTB16	DISABLED		PTB16		UART0_RX	FTM_CLKIN2	CAN0_TX	EWM_IN	XBAR0_IN5
63	40	32	PTB17	DISABLED		PTB17		UART0_TX	FTM_CLKIN1	CAN0_RX	EWM_OUT_b	
64	41	—	PTB18	DISABLED		PTB18	CAN0_TX		FTM3_CH2			
65	42	—	PTB19	DISABLED		PTB19	CAN0_RX		FTM3_CH3			
66	—	—	PTB20	DISABLED		PTB20				FLEXPWMA_X0	CMP0_OUT	
67	—	—	PTB21	DISABLED		PTB21				FLEXPWMA_X1	CMP1_OUT	
68	—	—	PTB22	DISABLED		PTB22				FLEXPWMA_X2	CMP2_OUT	
69	—	—	PTB23	DISABLED		PTB23		SPI0_PCS5		FLEXPWMA_X3	CMP3_OUT	
70	43	33	PTC0	ADCB_CH6b	ADCB_CH6b	PTC0	SPI0_PCS4	PDB0_EXTRG			FTM0_FLT1	SPI0_PCS0
71	44	34	PTC1/ LLWU_P6	ADCB_CH7b	ADCB_CH7b	PTC1/ LLWU_P6	SPI0_PCS3	UART1_RTS_b	FTM0_CH0	FLEXPWMA_A3	XBAR0_IN11	
72	45	35	PTC2	ADCB_CH6c/ CMP1_IN0	ADCB_CH6c/ CMP1_IN0	PTC2	SPI0_PCS2	UART1_CTS_b	FTM0_CH1	FLEXPWMA_B3	XBAR0_IN6	
73	46	36	PTC3/ LLWU_P7	CMP1_IN1	CMP1_IN1	PTC3/ LLWU_P7	SPI0_PCS1	UART1_RX	FTM0_CH2	CLKOUT	FTM3_FLT0	
74	47	—	VSS	VSS	VSS							
75	48	—	VDD	VDD	VDD							
76	49	37	PTC4/ LLWU_P8	DISABLED		PTC4/ LLWU_P8	SPI0_PCS0	UART1_TX	FTM0_CH3		CMP1_OUT	

100 LQFP	64 LQFP	48 LQFP	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
77	50	38	PTC5/ LLWU_P9	DISABLED		PTC5/ LLWU_P9	SPI0_SCK	LPTMR0_ ALT2	XBAR0_IN2		CMP0_OUT	FTM0_CH2
78	51	39	PTC6/ LLWU_P10	CMP2_IN4/ CMP0_IN0	CMP2_IN4/ CMP0_IN0	PTC6/ LLWU_P10	SPI0_SOUT	PDB0_EXTRG	XBAR0_IN3	UART0_RX	XBAR0_OUT6	I2C0_SCL
79	52	40	PTC7	CMP3_IN4/ CMP0_IN1	CMP3_IN4/ CMP0_IN1	PTC7	SPI0_SIN		XBAR0_IN4	UART0_TX	XBAR0_OUT7	I2C0_SDA
80	53	—	PTC8	ADCB_CH7c/ CMP0_IN2	ADCB_CH7c/ CMP0_IN2	PTC8		FTM3_CH4				
81	54	—	PTC9	ADCB_CH6d/ CMP0_IN3	ADCB_CH6d/ CMP0_IN3	PTC9		FTM3_CH5				
82	55	—	PTC10	ADCB_CH7d	ADCB_CH7d	PTC10		FTM3_CH6				
83	56	—	PTC11/ LLWU_P11	ADCB_CH6e	ADCB_CH6e	PTC11/ LLWU_P11		FTM3_CH7				
84	—	—	PTC12	DISABLED		PTC12			FTM_CLKIN0		FTM3_FLT0	
85	—	—	PTC13	DISABLED		PTC13			FTM_CLKIN1			
86	—	—	PTC14	DISABLED		PTC14		I2C0_SCL				
87	—	—	PTC15	DISABLED		PTC15		I2C0_SDA				
88	—	—	VSS	VSS	VSS							
89	—	—	VDD	VDD	VDD							
90	—	—	PTC16	DISABLED		PTC16	CAN1_RX					
91	—	—	PTC17	DISABLED		PTC17	CAN1_TX					
92	—	—	PTC18	DISABLED		PTC18						
93	57	41	PTD0/ LLWU_P12	DISABLED		PTD0/ LLWU_P12	SPI0_PCS0		FTM3_CH0	FTM0_CH0	FLEXPWMA_ A0	
94	58	42	PTD1	ADCA_CH7f	ADCA_CH7f	PTD1	SPI0_SCK		FTM3_CH1	FTM0_CH1	FLEXPWMA_ B0	
95	59	43	PTD2/ LLWU_P13	DISABLED		PTD2/ LLWU_P13	SPI0_SOUT		FTM3_CH2	FTM0_CH2	FLEXPWMA_ A1	I2C0_SCL
96	60	44	PTD3	DISABLED		PTD3	SPI0_SIN		FTM3_CH3	FTM0_CH3	FLEXPWMA_ B1	I2C0_SDA
97	61	45	PTD4/ LLWU_P14	DISABLED		PTD4/ LLWU_P14	SPI0_PCS1	UART0_RTS_ b	FTM0_CH4	FLEXPWMA_ A2	EWM_IN	SPI0_PCS0
98	62	46	PTD5	ADCA_CH6g	ADCA_CH6g	PTD5	SPI0_PCS2	UART0_CTS_ b/ UART0_COL_ b	FTM0_CH5	FLEXPWMA_ B2	EWM_OUT_b	SPI0_SCK
99	63	47	PTD6/ LLWU_P15	ADCA_CH7g	ADCA_CH7g	PTD6/ LLWU_P15	SPI0_PCS3	UART0_RX	FTM0_CH6	FTM1_CH0	FTM0_FLT0	SPI0_SOUT
100	64	48	PTD7	DISABLED		PTD7		UART0_TX	FTM0_CH7	FTM1_CH1	FTM0_FLT1	SPI0_SIN

11.6 Pinout diagrams

The following diagrams show pinouts for the packages. For each pin, the diagrams show the default function. However, many signals may be multiplexed onto a single pin.

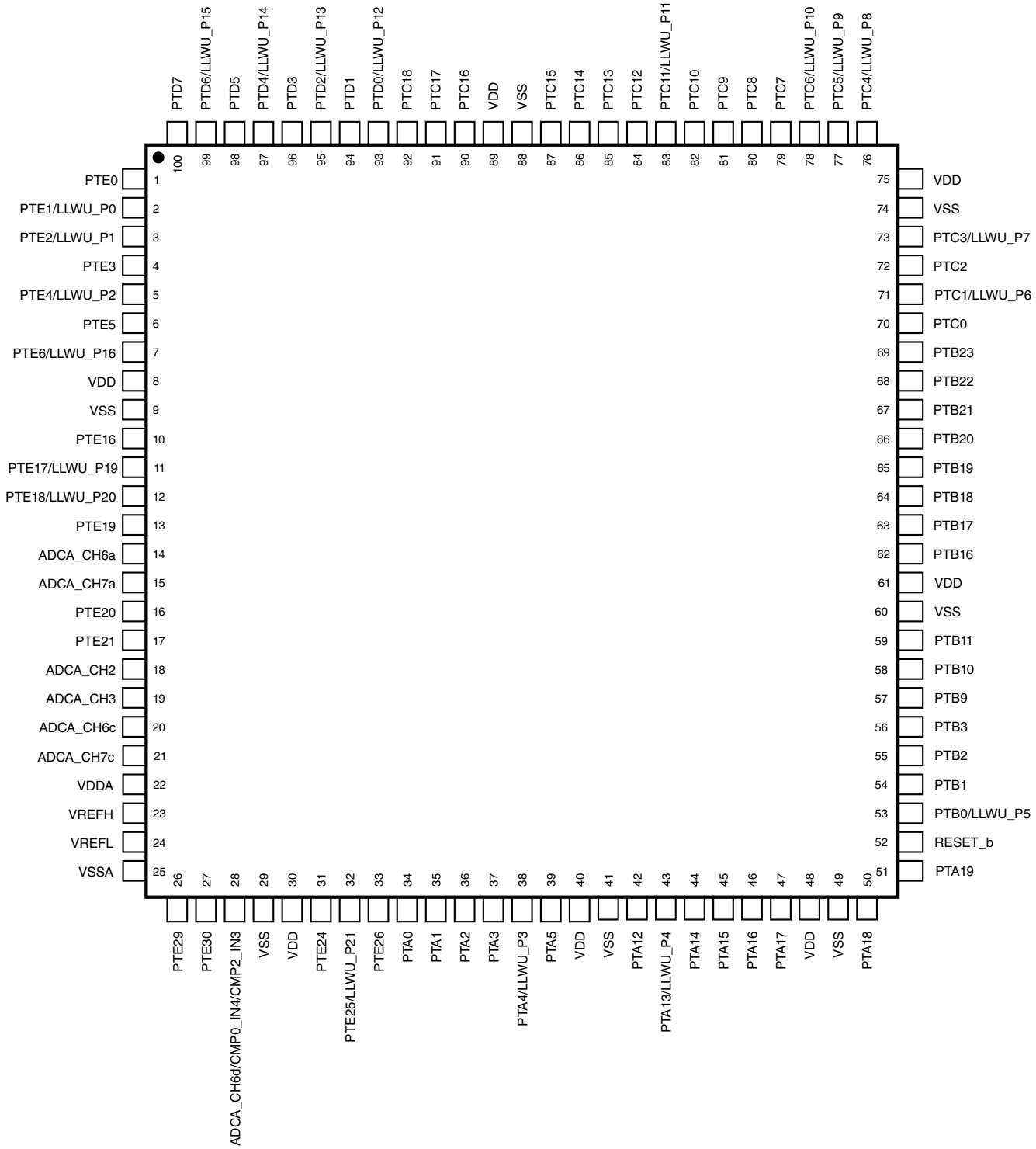


Figure 11-1. 100-pin LQFP

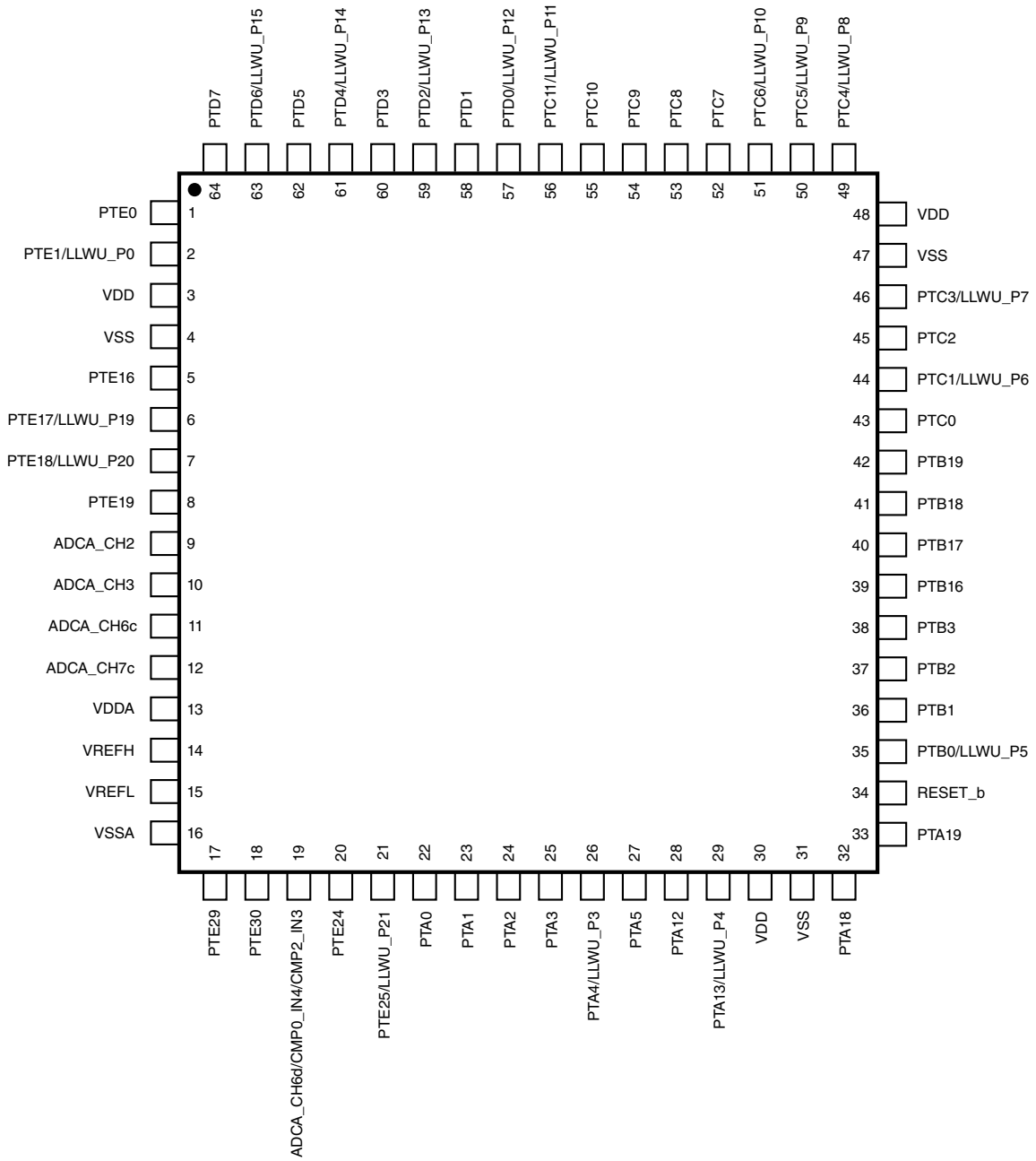


Figure 11-2. 64-pin LQFP

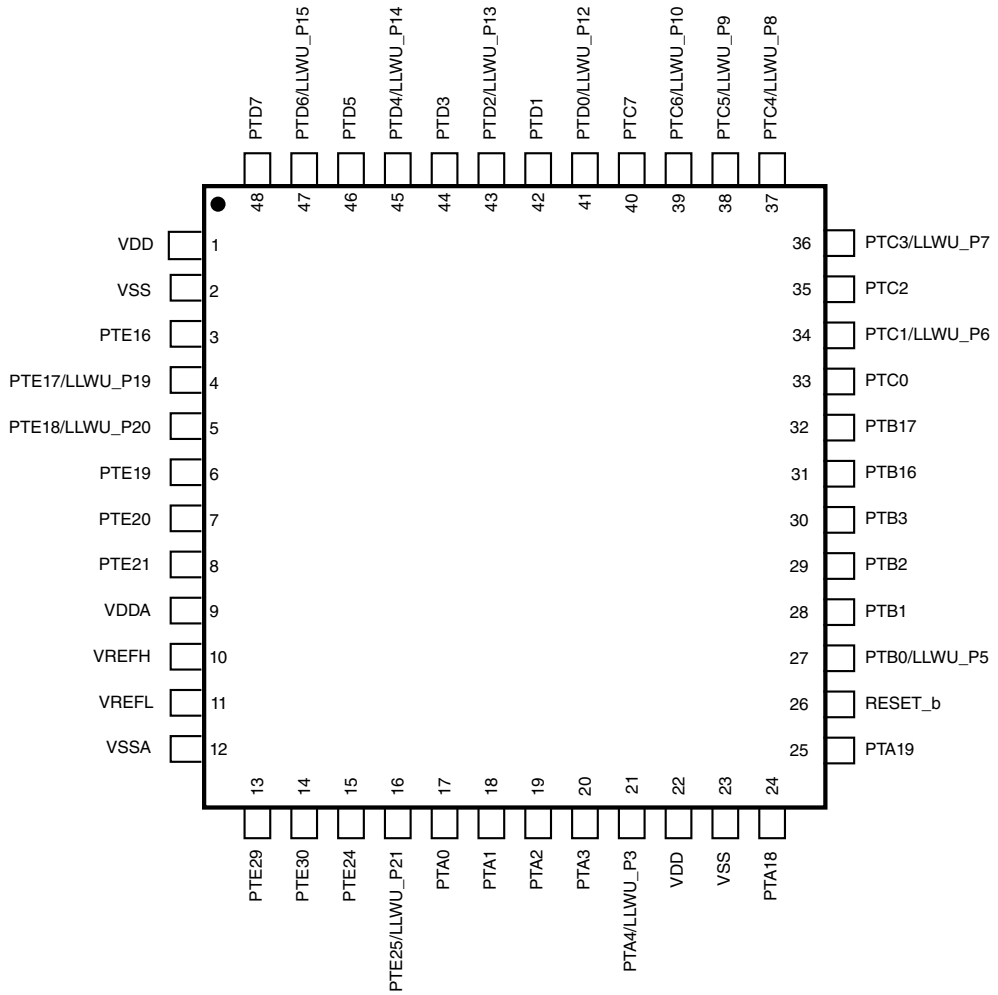


Figure 11-3. 48-pin LQFP

Chapter 12
Port control and interrupts (PORT)

12.1 Introduction

12.2 Overview

The Port Control and Interrupt (PORT) module provides support for port control, digital filtering, and external interrupt functions.

Most functions can be configured independently for each pin in the 32-bit port and affect the pin regardless of its pin muxing state.

There is one instance of the PORT module for each port. Not all pins within each port are implemented on a specific device.

12.2.1 Features

The PORT module has the following features:

- Pin interrupt
  - Interrupt flag and enable registers for each pin
  - Support for edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wake-up in low-power modes
  - Pin interrupt is functional in all digital pin muxing modes
- Digital input filter on selected pins
  - Digital input filter for each pin, usable by any digital peripheral muxed onto the pin
  - Individual enable or bypass control field per pin

    - Selectable clock source for digital input filter with a five bit resolution on filter size
    - Functional in all digital pin multiplexing modes
- Port control
    - Individual pull control fields with pullup, pulldown, and pull-disable support
    - Individual drive strength field supporting high and low drive strength
    - Individual slew rate field supporting fast and slow slew rates
    - Individual input passive filter field supporting enable and disable of the individual input passive filter
    - Individual open drain field supporting enable and disable of the individual open drain output
    - Individual mux control field supporting analog or pin disabled, GPIO, and up to six chip-specific digital functions
    - Pad configuration fields are functional in all digital pin muxing modes.

12.2.2 Modes of operation

12.2.2.1 Run mode

In Run mode, the PORT operates normally.

12.2.2.2 Wait mode

In Wait mode, PORT continues to operate normally and may be configured to exit the Low-Power mode if an enabled interrupt is detected. DMA requests are still generated during the Wait mode, but do not cause an exit from the Low-Power mode.

12.2.2.3 Stop mode

In Stop mode, the PORT can be configured to exit the Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

In Stop mode, the digital input filters are bypassed unless they are configured to run from the LPO clock source.

12.2.2.4 Debug mode

In Debug mode, PORT operates normally.

12.3 External signal description

The table found here describes the PORT external signal.

Table 12-1. Signal properties

Name	Function	I/O	Reset	Pull
PORTx[31:0]	External interrupt	I/O	0	-

NOTE

Not all pins within each port are implemented on each device.

12.4 Detailed signal description

The table found here contains the detailed signal description for the PORT interface.

Table 12-2. PORT interface—detailed signal description

Signal	I/O	Description	
PORTx[31:0]	I/O	External interrupt.	
		State meaning	Asserted—pin is logic 1. Negated—pin is logic 0.
		Timing	Assertion—may occur at any time and can assert asynchronously to the system clock. Negation—may occur at any time and can assert asynchronously to the system clock.

12.5 Memory map and register definition

Any read or write access to the PORT memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states.

PORT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_9000	Pin Control Register n (PORTA_PCR0)	32	R/W	See section	12.5.1/156
4004_9004	Pin Control Register n (PORTA_PCR1)	32	R/W	See section	12.5.1/156
4004_9008	Pin Control Register n (PORTA_PCR2)	32	R/W	See section	12.5.1/156
4004_900C	Pin Control Register n (PORTA_PCR3)	32	R/W	See section	12.5.1/156
4004_9010	Pin Control Register n (PORTA_PCR4)	32	R/W	See section	12.5.1/156
4004_9014	Pin Control Register n (PORTA_PCR5)	32	R/W	See section	12.5.1/156
4004_9018	Pin Control Register n (PORTA_PCR6)	32	R/W	See section	12.5.1/156
4004_901C	Pin Control Register n (PORTA_PCR7)	32	R/W	See section	12.5.1/156
4004_9020	Pin Control Register n (PORTA_PCR8)	32	R/W	See section	12.5.1/156
4004_9024	Pin Control Register n (PORTA_PCR9)	32	R/W	See section	12.5.1/156
4004_9028	Pin Control Register n (PORTA_PCR10)	32	R/W	See section	12.5.1/156
4004_902C	Pin Control Register n (PORTA_PCR11)	32	R/W	See section	12.5.1/156
4004_9030	Pin Control Register n (PORTA_PCR12)	32	R/W	See section	12.5.1/156
4004_9034	Pin Control Register n (PORTA_PCR13)	32	R/W	See section	12.5.1/156
4004_9038	Pin Control Register n (PORTA_PCR14)	32	R/W	See section	12.5.1/156
4004_903C	Pin Control Register n (PORTA_PCR15)	32	R/W	See section	12.5.1/156
4004_9040	Pin Control Register n (PORTA_PCR16)	32	R/W	See section	12.5.1/156
4004_9044	Pin Control Register n (PORTA_PCR17)	32	R/W	See section	12.5.1/156
4004_9048	Pin Control Register n (PORTA_PCR18)	32	R/W	See section	12.5.1/156
4004_904C	Pin Control Register n (PORTA_PCR19)	32	R/W	See section	12.5.1/156
4004_9050	Pin Control Register n (PORTA_PCR20)	32	R/W	See section	12.5.1/156
4004_9054	Pin Control Register n (PORTA_PCR21)	32	R/W	See section	12.5.1/156
4004_9058	Pin Control Register n (PORTA_PCR22)	32	R/W	See section	12.5.1/156
4004_905C	Pin Control Register n (PORTA_PCR23)	32	R/W	See section	12.5.1/156
4004_9060	Pin Control Register n (PORTA_PCR24)	32	R/W	See section	12.5.1/156
4004_9064	Pin Control Register n (PORTA_PCR25)	32	R/W	See section	12.5.1/156
4004_9068	Pin Control Register n (PORTA_PCR26)	32	R/W	See section	12.5.1/156
4004_906C	Pin Control Register n (PORTA_PCR27)	32	R/W	See section	12.5.1/156
4004_9070	Pin Control Register n (PORTA_PCR28)	32	R/W	See section	12.5.1/156
4004_9074	Pin Control Register n (PORTA_PCR29)	32	R/W	See section	12.5.1/156
4004_9078	Pin Control Register n (PORTA_PCR30)	32	R/W	See section	12.5.1/156
4004_907C	Pin Control Register n (PORTA_PCR31)	32	R/W	See section	12.5.1/156
4004_9080	Global Pin Control Low Register (PORTA_GPCLR)	32	W (always reads 0)	0000_0000h	12.5.2/159
4004_9084	Global Pin Control High Register (PORTA_GPCHR)	32	W (always reads 0)	0000_0000h	12.5.3/159
4004_90A0	Interrupt Status Flag Register (PORTA_ISFR)	32	w1c	0000_0000h	12.5.4/160

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_90C0	Digital Filter Enable Register (PORTA_DFER)	32	R/W	0000_0000h	12.5.5/160
4004_90C4	Digital Filter Clock Register (PORTA_DFCR)	32	R/W	0000_0000h	12.5.6/161
4004_90C8	Digital Filter Width Register (PORTA_DFWR)	32	R/W	0000_0000h	12.5.7/161
4004_A000	Pin Control Register n (PORTB_PCR0)	32	R/W	See section	12.5.1/156
4004_A004	Pin Control Register n (PORTB_PCR1)	32	R/W	See section	12.5.1/156
4004_A008	Pin Control Register n (PORTB_PCR2)	32	R/W	See section	12.5.1/156
4004_A00C	Pin Control Register n (PORTB_PCR3)	32	R/W	See section	12.5.1/156
4004_A010	Pin Control Register n (PORTB_PCR4)	32	R/W	See section	12.5.1/156
4004_A014	Pin Control Register n (PORTB_PCR5)	32	R/W	See section	12.5.1/156
4004_A018	Pin Control Register n (PORTB_PCR6)	32	R/W	See section	12.5.1/156
4004_A01C	Pin Control Register n (PORTB_PCR7)	32	R/W	See section	12.5.1/156
4004_A020	Pin Control Register n (PORTB_PCR8)	32	R/W	See section	12.5.1/156
4004_A024	Pin Control Register n (PORTB_PCR9)	32	R/W	See section	12.5.1/156
4004_A028	Pin Control Register n (PORTB_PCR10)	32	R/W	See section	12.5.1/156
4004_A02C	Pin Control Register n (PORTB_PCR11)	32	R/W	See section	12.5.1/156
4004_A030	Pin Control Register n (PORTB_PCR12)	32	R/W	See section	12.5.1/156
4004_A034	Pin Control Register n (PORTB_PCR13)	32	R/W	See section	12.5.1/156
4004_A038	Pin Control Register n (PORTB_PCR14)	32	R/W	See section	12.5.1/156
4004_A03C	Pin Control Register n (PORTB_PCR15)	32	R/W	See section	12.5.1/156
4004_A040	Pin Control Register n (PORTB_PCR16)	32	R/W	See section	12.5.1/156
4004_A044	Pin Control Register n (PORTB_PCR17)	32	R/W	See section	12.5.1/156
4004_A048	Pin Control Register n (PORTB_PCR18)	32	R/W	See section	12.5.1/156
4004_A04C	Pin Control Register n (PORTB_PCR19)	32	R/W	See section	12.5.1/156
4004_A050	Pin Control Register n (PORTB_PCR20)	32	R/W	See section	12.5.1/156
4004_A054	Pin Control Register n (PORTB_PCR21)	32	R/W	See section	12.5.1/156
4004_A058	Pin Control Register n (PORTB_PCR22)	32	R/W	See section	12.5.1/156
4004_A05C	Pin Control Register n (PORTB_PCR23)	32	R/W	See section	12.5.1/156
4004_A060	Pin Control Register n (PORTB_PCR24)	32	R/W	See section	12.5.1/156
4004_A064	Pin Control Register n (PORTB_PCR25)	32	R/W	See section	12.5.1/156
4004_A068	Pin Control Register n (PORTB_PCR26)	32	R/W	See section	12.5.1/156
4004_A06C	Pin Control Register n (PORTB_PCR27)	32	R/W	See section	12.5.1/156
4004_A070	Pin Control Register n (PORTB_PCR28)	32	R/W	See section	12.5.1/156
4004_A074	Pin Control Register n (PORTB_PCR29)	32	R/W	See section	12.5.1/156
4004_A078	Pin Control Register n (PORTB_PCR30)	32	R/W	See section	12.5.1/156
4004_A07C	Pin Control Register n (PORTB_PCR31)	32	R/W	See section	12.5.1/156
4004_A080	Global Pin Control Low Register (PORTB_GPCLR)	32	W (always reads 0)	0000_0000h	12.5.2/159

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A084	Global Pin Control High Register (PORTB_GPCHR)	32	W (always reads 0)	0000_0000h	12.5.3/159
4004_A0A0	Interrupt Status Flag Register (PORTB_ISFR)	32	w1c	0000_0000h	12.5.4/160
4004_A0C0	Digital Filter Enable Register (PORTB_DFER)	32	R/W	0000_0000h	12.5.5/160
4004_A0C4	Digital Filter Clock Register (PORTB_DFCR)	32	R/W	0000_0000h	12.5.6/161
4004_A0C8	Digital Filter Width Register (PORTB_DFWR)	32	R/W	0000_0000h	12.5.7/161
4004_B000	Pin Control Register n (PORTC_PCR0)	32	R/W	See section	12.5.1/156
4004_B004	Pin Control Register n (PORTC_PCR1)	32	R/W	See section	12.5.1/156
4004_B008	Pin Control Register n (PORTC_PCR2)	32	R/W	See section	12.5.1/156
4004_B00C	Pin Control Register n (PORTC_PCR3)	32	R/W	See section	12.5.1/156
4004_B010	Pin Control Register n (PORTC_PCR4)	32	R/W	See section	12.5.1/156
4004_B014	Pin Control Register n (PORTC_PCR5)	32	R/W	See section	12.5.1/156
4004_B018	Pin Control Register n (PORTC_PCR6)	32	R/W	See section	12.5.1/156
4004_B01C	Pin Control Register n (PORTC_PCR7)	32	R/W	See section	12.5.1/156
4004_B020	Pin Control Register n (PORTC_PCR8)	32	R/W	See section	12.5.1/156
4004_B024	Pin Control Register n (PORTC_PCR9)	32	R/W	See section	12.5.1/156
4004_B028	Pin Control Register n (PORTC_PCR10)	32	R/W	See section	12.5.1/156
4004_B02C	Pin Control Register n (PORTC_PCR11)	32	R/W	See section	12.5.1/156
4004_B030	Pin Control Register n (PORTC_PCR12)	32	R/W	See section	12.5.1/156
4004_B034	Pin Control Register n (PORTC_PCR13)	32	R/W	See section	12.5.1/156
4004_B038	Pin Control Register n (PORTC_PCR14)	32	R/W	See section	12.5.1/156
4004_B03C	Pin Control Register n (PORTC_PCR15)	32	R/W	See section	12.5.1/156
4004_B040	Pin Control Register n (PORTC_PCR16)	32	R/W	See section	12.5.1/156
4004_B044	Pin Control Register n (PORTC_PCR17)	32	R/W	See section	12.5.1/156
4004_B048	Pin Control Register n (PORTC_PCR18)	32	R/W	See section	12.5.1/156
4004_B04C	Pin Control Register n (PORTC_PCR19)	32	R/W	See section	12.5.1/156
4004_B050	Pin Control Register n (PORTC_PCR20)	32	R/W	See section	12.5.1/156
4004_B054	Pin Control Register n (PORTC_PCR21)	32	R/W	See section	12.5.1/156
4004_B058	Pin Control Register n (PORTC_PCR22)	32	R/W	See section	12.5.1/156
4004_B05C	Pin Control Register n (PORTC_PCR23)	32	R/W	See section	12.5.1/156
4004_B060	Pin Control Register n (PORTC_PCR24)	32	R/W	See section	12.5.1/156
4004_B064	Pin Control Register n (PORTC_PCR25)	32	R/W	See section	12.5.1/156
4004_B068	Pin Control Register n (PORTC_PCR26)	32	R/W	See section	12.5.1/156
4004_B06C	Pin Control Register n (PORTC_PCR27)	32	R/W	See section	12.5.1/156
4004_B070	Pin Control Register n (PORTC_PCR28)	32	R/W	See section	12.5.1/156
4004_B074	Pin Control Register n (PORTC_PCR29)	32	R/W	See section	12.5.1/156
4004_B078	Pin Control Register n (PORTC_PCR30)	32	R/W	See section	12.5.1/156

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_B07C	Pin Control Register n (PORTC_PCR31)	32	R/W	See section	12.5.1/156
4004_B080	Global Pin Control Low Register (PORTC_GPCLR)	32	W (always reads 0)	0000_0000h	12.5.2/159
4004_B084	Global Pin Control High Register (PORTC_GPCHR)	32	W (always reads 0)	0000_0000h	12.5.3/159
4004_B0A0	Interrupt Status Flag Register (PORTC_ISFR)	32	w1c	0000_0000h	12.5.4/160
4004_B0C0	Digital Filter Enable Register (PORTC_DFER)	32	R/W	0000_0000h	12.5.5/160
4004_B0C4	Digital Filter Clock Register (PORTC_DFCR)	32	R/W	0000_0000h	12.5.6/161
4004_B0C8	Digital Filter Width Register (PORTC_DFWR)	32	R/W	0000_0000h	12.5.7/161
4004_C000	Pin Control Register n (PORTD_PCR0)	32	R/W	See section	12.5.1/156
4004_C004	Pin Control Register n (PORTD_PCR1)	32	R/W	See section	12.5.1/156
4004_C008	Pin Control Register n (PORTD_PCR2)	32	R/W	See section	12.5.1/156
4004_C00C	Pin Control Register n (PORTD_PCR3)	32	R/W	See section	12.5.1/156
4004_C010	Pin Control Register n (PORTD_PCR4)	32	R/W	See section	12.5.1/156
4004_C014	Pin Control Register n (PORTD_PCR5)	32	R/W	See section	12.5.1/156
4004_C018	Pin Control Register n (PORTD_PCR6)	32	R/W	See section	12.5.1/156
4004_C01C	Pin Control Register n (PORTD_PCR7)	32	R/W	See section	12.5.1/156
4004_C020	Pin Control Register n (PORTD_PCR8)	32	R/W	See section	12.5.1/156
4004_C024	Pin Control Register n (PORTD_PCR9)	32	R/W	See section	12.5.1/156
4004_C028	Pin Control Register n (PORTD_PCR10)	32	R/W	See section	12.5.1/156
4004_C02C	Pin Control Register n (PORTD_PCR11)	32	R/W	See section	12.5.1/156
4004_C030	Pin Control Register n (PORTD_PCR12)	32	R/W	See section	12.5.1/156
4004_C034	Pin Control Register n (PORTD_PCR13)	32	R/W	See section	12.5.1/156
4004_C038	Pin Control Register n (PORTD_PCR14)	32	R/W	See section	12.5.1/156
4004_C03C	Pin Control Register n (PORTD_PCR15)	32	R/W	See section	12.5.1/156
4004_C040	Pin Control Register n (PORTD_PCR16)	32	R/W	See section	12.5.1/156
4004_C044	Pin Control Register n (PORTD_PCR17)	32	R/W	See section	12.5.1/156
4004_C048	Pin Control Register n (PORTD_PCR18)	32	R/W	See section	12.5.1/156
4004_C04C	Pin Control Register n (PORTD_PCR19)	32	R/W	See section	12.5.1/156
4004_C050	Pin Control Register n (PORTD_PCR20)	32	R/W	See section	12.5.1/156
4004_C054	Pin Control Register n (PORTD_PCR21)	32	R/W	See section	12.5.1/156
4004_C058	Pin Control Register n (PORTD_PCR22)	32	R/W	See section	12.5.1/156
4004_C05C	Pin Control Register n (PORTD_PCR23)	32	R/W	See section	12.5.1/156
4004_C060	Pin Control Register n (PORTD_PCR24)	32	R/W	See section	12.5.1/156
4004_C064	Pin Control Register n (PORTD_PCR25)	32	R/W	See section	12.5.1/156
4004_C068	Pin Control Register n (PORTD_PCR26)	32	R/W	See section	12.5.1/156
4004_C06C	Pin Control Register n (PORTD_PCR27)	32	R/W	See section	12.5.1/156

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_C070	Pin Control Register n (PORTD_PCR28)	32	R/W	See section	12.5.1/156
4004_C074	Pin Control Register n (PORTD_PCR29)	32	R/W	See section	12.5.1/156
4004_C078	Pin Control Register n (PORTD_PCR30)	32	R/W	See section	12.5.1/156
4004_C07C	Pin Control Register n (PORTD_PCR31)	32	R/W	See section	12.5.1/156
4004_C080	Global Pin Control Low Register (PORTD_GPCLR)	32	W (always reads 0)	0000_0000h	12.5.2/159
4004_C084	Global Pin Control High Register (PORTD_GPCHR)	32	W (always reads 0)	0000_0000h	12.5.3/159
4004_C0A0	Interrupt Status Flag Register (PORTD_ISFR)	32	w1c	0000_0000h	12.5.4/160
4004_C0C0	Digital Filter Enable Register (PORTD_DFER)	32	R/W	0000_0000h	12.5.5/160
4004_C0C4	Digital Filter Clock Register (PORTD_DFCR)	32	R/W	0000_0000h	12.5.6/161
4004_C0C8	Digital Filter Width Register (PORTD_DFWR)	32	R/W	0000_0000h	12.5.7/161
4004_D000	Pin Control Register n (PORTE_PCR0)	32	R/W	See section	12.5.1/156
4004_D004	Pin Control Register n (PORTE_PCR1)	32	R/W	See section	12.5.1/156
4004_D008	Pin Control Register n (PORTE_PCR2)	32	R/W	See section	12.5.1/156
4004_D00C	Pin Control Register n (PORTE_PCR3)	32	R/W	See section	12.5.1/156
4004_D010	Pin Control Register n (PORTE_PCR4)	32	R/W	See section	12.5.1/156
4004_D014	Pin Control Register n (PORTE_PCR5)	32	R/W	See section	12.5.1/156
4004_D018	Pin Control Register n (PORTE_PCR6)	32	R/W	See section	12.5.1/156
4004_D01C	Pin Control Register n (PORTE_PCR7)	32	R/W	See section	12.5.1/156
4004_D020	Pin Control Register n (PORTE_PCR8)	32	R/W	See section	12.5.1/156
4004_D024	Pin Control Register n (PORTE_PCR9)	32	R/W	See section	12.5.1/156
4004_D028	Pin Control Register n (PORTE_PCR10)	32	R/W	See section	12.5.1/156
4004_D02C	Pin Control Register n (PORTE_PCR11)	32	R/W	See section	12.5.1/156
4004_D030	Pin Control Register n (PORTE_PCR12)	32	R/W	See section	12.5.1/156
4004_D034	Pin Control Register n (PORTE_PCR13)	32	R/W	See section	12.5.1/156
4004_D038	Pin Control Register n (PORTE_PCR14)	32	R/W	See section	12.5.1/156
4004_D03C	Pin Control Register n (PORTE_PCR15)	32	R/W	See section	12.5.1/156
4004_D040	Pin Control Register n (PORTE_PCR16)	32	R/W	See section	12.5.1/156
4004_D044	Pin Control Register n (PORTE_PCR17)	32	R/W	See section	12.5.1/156
4004_D048	Pin Control Register n (PORTE_PCR18)	32	R/W	See section	12.5.1/156
4004_D04C	Pin Control Register n (PORTE_PCR19)	32	R/W	See section	12.5.1/156
4004_D050	Pin Control Register n (PORTE_PCR20)	32	R/W	See section	12.5.1/156
4004_D054	Pin Control Register n (PORTE_PCR21)	32	R/W	See section	12.5.1/156
4004_D058	Pin Control Register n (PORTE_PCR22)	32	R/W	See section	12.5.1/156
4004_D05C	Pin Control Register n (PORTE_PCR23)	32	R/W	See section	12.5.1/156
4004_D060	Pin Control Register n (PORTE_PCR24)	32	R/W	See section	12.5.1/156

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_D064	Pin Control Register n (PORTE_PCR25)	32	R/W	See section	12.5.1/156
4004_D068	Pin Control Register n (PORTE_PCR26)	32	R/W	See section	12.5.1/156
4004_D06C	Pin Control Register n (PORTE_PCR27)	32	R/W	See section	12.5.1/156
4004_D070	Pin Control Register n (PORTE_PCR28)	32	R/W	See section	12.5.1/156
4004_D074	Pin Control Register n (PORTE_PCR29)	32	R/W	See section	12.5.1/156
4004_D078	Pin Control Register n (PORTE_PCR30)	32	R/W	See section	12.5.1/156
4004_D07C	Pin Control Register n (PORTE_PCR31)	32	R/W	See section	12.5.1/156
4004_D080	Global Pin Control Low Register (PORTE_GPCLR)	32	W (always reads 0)	0000_0000h	12.5.2/159
4004_D084	Global Pin Control High Register (PORTE_GPCHR)	32	W (always reads 0)	0000_0000h	12.5.3/159
4004_D0A0	Interrupt Status Flag Register (PORTE_ISFR)	32	w1c	0000_0000h	12.5.4/160
4004_D0C0	Digital Filter Enable Register (PORTE_DFER)	32	R/W	0000_0000h	12.5.5/160
4004_D0C4	Digital Filter Clock Register (PORTE_DFCR)	32	R/W	0000_0000h	12.5.6/161
4004_D0C8	Digital Filter Width Register (PORTE_DFWR)	32	R/W	0000_0000h	12.5.7/161

12.5.1 Pin Control Register n (PORTx_PCR*n*)

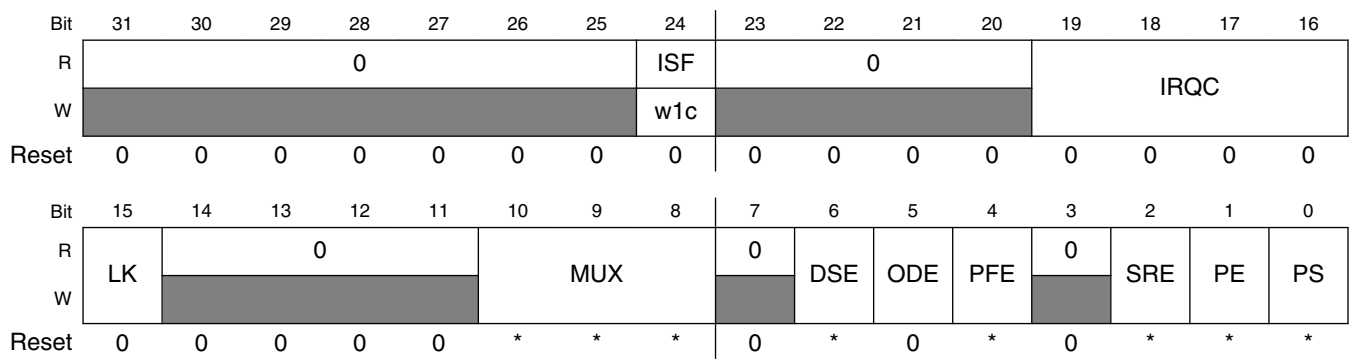
NOTE

See the Signal Multiplexing and Pin Assignment chapter for the reset value of this device.

See the GPIO Configuration section for details on the available functions for each pin.

Do not modify pin configuration registers associated with pins that are not available in a reduced-pin package offering. Unbonded pins not available in a package are disabled by default to prevent them from consuming power.

Address: Base address + 0h offset + (4d × i), where i=0d to 31d



* Notes:
- MUX field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- DSE field: Varies by port. See the Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PFE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- SRE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PS field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.

PORTx_PCR*n* field descriptions

Field	Description
31–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 ISF	Interrupt Status Flag The pin interrupt configuration is valid in all digital pin muxing modes.

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 IRQC	Interrupt Configuration The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows: 0000 Interrupt Status Flag (ISF) is disabled. 0001 ISF flag and DMA request on rising edge. 0010 ISF flag and DMA request on falling edge. 0011 ISF flag and DMA request on either edge. 0100 Reserved. 0101 Reserved. 0110 Reserved. 0111 Reserved. 1000 ISF flag and Interrupt when logic 0. 1001 ISF flag and Interrupt on rising-edge. 1010 ISF flag and Interrupt on falling-edge. 1011 ISF flag and Interrupt on either edge. 1100 ISF flag and Interrupt when logic 1. 1101 Reserved. 1110 Reserved. 1111 Reserved.
15 LK	Lock Register 0 Pin Control Register fields [15:0] are not locked. 1 Pin Control Register fields [15:0] are locked and cannot be updated until the next system reset.
14–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 MUX	Pin Mux Control Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot. The corresponding pin is configured in the following pin muxing slot as follows: 000 Pin disabled (Alternative 0) (analog). 001 Alternative 1 (GPIO). 010 Alternative 2 (chip-specific). 011 Alternative 3 (chip-specific). 100 Alternative 4 (chip-specific). 101 Alternative 5 (chip-specific). 110 Alternative 6 (chip-specific). 111 Alternative 7 (chip-specific).

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DSE	Drive Strength Enable Drive strength configuration is valid in all digital pin muxing modes. 0 Low drive strength is configured on the corresponding pin, if pin is configured as a digital output. 1 High drive strength is configured on the corresponding pin, if pin is configured as a digital output.
5 ODE	Open Drain Enable Open drain configuration is valid in all digital pin muxing modes. 0 Open drain output is disabled on the corresponding pin. 1 Open drain output is enabled on the corresponding pin, if the pin is configured as a digital output.
4 PFE	Passive Filter Enable Passive filter configuration is valid in all digital pin muxing modes. 0 Passive input filter is disabled on the corresponding pin. 1 Passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input. Refer to the device data sheet for filter characteristics.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SRE	Slew Rate Enable Slew rate configuration is valid in all digital pin muxing modes. 0 Fast slew rate is configured on the corresponding pin, if the pin is configured as a digital output. 1 Slow slew rate is configured on the corresponding pin, if the pin is configured as a digital output.
1 PE	Pull Enable Pull configuration is valid in all digital pin muxing modes. 0 Internal pullup or pulldown resistor is not enabled on the corresponding pin. 1 Internal pullup or pulldown resistor is enabled on the corresponding pin, if the pin is configured as a digital input.
0 PS	Pull Select Pull configuration is valid in all digital pin muxing modes. 0 Internal pulldown resistor is enabled on the corresponding pin, if the corresponding PE field is set. 1 Internal pullup resistor is enabled on the corresponding pin, if the corresponding PE field is set.

12.5.2 Global Pin Control Low Register (PORTx_GPCLR)

Only 32-bit writes are supported to this register.

Address: Base address + 80h offset

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

PORTx_GPCLR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (15 through 0) bits [15:0] update with the value in GPWD. If a selected Pin Control Register is locked then the write to that register is ignored. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.
GPWD	Global Pin Write Data Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.

12.5.3 Global Pin Control High Register (PORTx_GPCHR)

Only 32-bit writes are supported to this register.

Address: Base address + 84h offset

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

PORTx_GPCHR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (31 through 16) bits [15:0] update with the value in GPWD. If a selected Pin Control Register is locked then the write to that register is ignored. 0 Corresponding Pin Control Register is not updated with the value in GPWD. 1 Corresponding Pin Control Register is updated with the value in GPWD.
GPWD	Global Pin Write Data Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.

12.5.4 Interrupt Status Flag Register (PORTx_ISFR)

The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Pin Control Register, and each flag can be cleared in either location.

Address: Base address + A0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISF																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTx_ISFR field descriptions

Field	Description
ISF	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt of the same number as the field. 0 Configured interrupt is not detected. 1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

12.5.5 Digital Filter Enable Register (PORTx_DFER)

The corresponding bit is read only for pins that do not support a digital filter. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support digital filter.

The digital filter configuration is valid in all digital pin muxing modes.

Address: Base address + C0h offset

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

PORTx_DFER field descriptions

Field	Description
DFE	Digital Filter Enable

PORTx_DFER field descriptions (continued)

Field	Description
	The digital filter configuration is valid in all digital pin muxing modes. The output of each digital filter is reset to zero at system reset and whenever the digital filter is disabled. Each bit in the field enables the digital filter of the same number as the field. 0 Digital filter is disabled on the corresponding pin and output of the digital filter is reset to zero. 1 Digital filter is enabled on the corresponding pin, if the pin is configured as a digital input.

12.5.6 Digital Filter Clock Register (PORTx_DFCR)

This register is read only for ports that do not support a digital filter.

The digital filter configuration is valid in all digital pin muxing modes.

Address: Base address + C4h offset

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

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

PORTx_DFCR field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 CS	Clock Source The digital filter configuration is valid in all digital pin muxing modes. Configures the clock source for the digital input filters. Changing the filter clock source must be done only when all digital filters are disabled. 0 Digital filters are clocked by the bus clock. 1 Digital filters are clocked by the LPO clock.

12.5.7 Digital Filter Width Register (PORTx_DFWR)

This register is read only for ports that do not support a digital filter.

The digital filter configuration is valid in all digital pin muxing modes.

Address: Base address + C8h offset

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

PORTx_DFWR field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FILT	Filter Length The digital filter configuration is valid in all digital pin muxing modes. Configures the maximum size of the glitches, in clock cycles, that the digital filter absorbs for the enabled digital filters. Glitches that are longer than this register setting will pass through the digital filter, and glitches that are equal to or less than this register setting are filtered. Changing the filter length must be done only after all filters are disabled.

12.6 Functional description

12.6.1 Pin control

Each port pin has a corresponding Pin Control register, PORT_PCRn, associated with it.

The upper half of the Pin Control register configures the pin's capability to either interrupt the CPU or request a DMA transfer, on a rising/falling edge or both edges as well as a logic level occurring on the port pin. It also includes a flag to indicate that an interrupt has occurred.

The lower half of the Pin Control register configures the following functions for each pin within the 32-bit port.

- Pullup or pulldown enable
- Drive strength and slew rate configuration
- Open drain enable
- Passive input filter enable
- Pin Muxing mode

The functions apply across all digital pin muxing modes and individual peripherals do not override the configuration in the Pin Control register. For example, if an I^2C function is enabled on a pin, that does not override the pullup or open drain configuration for that pin.

When the Pin Muxing mode is configured for analog or is disabled, all the digital functions on that pin are disabled. This includes the pullup and pulldown enables, output buffer enable, input buffer enable, and passive filter enable.

The LK bit (bit 15 of Pin Control Register PCR*n*) allows the configuration for each pin to be locked until the next system reset. When locked, writes to the lower half of that pin control register are ignored, although a bus error is not generated on an attempted write to a locked register.

The configuration of each Pin Control register is retained when the PORT module is disabled.

Whenever a pin is configured in any digital pin muxing mode, the input buffer for that pin is enabled allowing the pin state to be read via the corresponding GPIO Port Data Input Register (GPIO_PDIR) or allowing a pin interrupt or DMA request to be generated. If a pin is ever floating when its input buffer is enabled, then this can cause an increase in power consumption and must be avoided. A pin can be floating due to an input pin that is not connected or an output pin that has tri-stated (output buffer is disabled).

Enabling the internal pull resistor (or implementing an external pull resistor) will ensure a pin does not float when its input buffer is enabled; note that the internal pull resistor is automatically disabled whenever the output buffer is enabled allowing the Pull Enable bit to remain set. Configuring the Pin Muxing mode to disabled or analog will disable the pin's input buffer and results in the lowest power consumption.

12.6.2 Global pin control

The two global pin control registers allow a single register write to update the lower half of the pin control register on up to 16 pins, all with the same value. Registers that are locked cannot be written using the global pin control registers.

The global pin control registers are designed to enable software to quickly configure multiple pins within the one port for the same peripheral function. However, the interrupt functions cannot be configured using the global pin control registers.

The global pin control registers are write-only registers, that always read as 0.

12.6.3 External interrupts

The external interrupt capability of the PORT module is available in all digital pin muxing modes provided the PORT module is enabled.

Each pin can be individually configured for any of the following external interrupt modes:

- Interrupt disabled, default out of reset
- Active high level sensitive interrupt
- Active low level sensitive interrupt
- Rising edge sensitive interrupt
- Falling edge sensitive interrupt
- Rising and falling edge sensitive interrupt
- Rising edge sensitive DMA request
- Falling edge sensitive DMA request
- Rising and falling edge sensitive DMA request

The interrupt status flag is set when the configured edge or level is detected on the pin or at the output of the digital input filter, if the digital input digital filter is enabled. When not in Stop mode, the input is first synchronized to the bus clock to detect the configured level or edge transition.

The PORT module generates a single interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that port. The interrupt negates after the interrupt status flags for all enabled interrupts have been cleared by writing a logic 1 to the ISF flag in either the PORT_ISFR or PORT_PCRn registers.

The PORT module generates a single DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that port. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

During Stop mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wake-up signal to exit the Low-Power mode.

12.6.4 Digital filter

The digital filter capabilities of the PORT module are available in all digital Pin Muxing modes if the PORT module is enabled.

The clock used for all digital filters within one port can be configured between the bus clock or the LPO clock. This selection must be changed only when all digital filters for that port are disabled. If the digital filters for a port are configured to use the bus clock, then the digital filters are bypassed for the duration of Stop mode. While the digital filters

are bypassed, the output of each digital filter always equals the input pin, but the internal state of the digital filters remains static and does not update due to any change on the input pin.

The filter width in clock size is the same for all enabled digital filters within one port and must be changed only when all digital filters for that port are disabled.

The output of each digital filter is logic zero after system reset and whenever a digital filter is disabled. After a digital filter is enabled, the input is synchronized to the filter clock, either the bus clock or the LPO clock. If the synchronized input and the output of the digital filter remain different for a number of filter clock cycles equal to the filter width register configuration, then the output of the digital filter updates to equal the synchronized filter input.

The maximum latency through a digital filter equals three filter clock cycles plus the filter width configuration register.

Chapter 13
System Integration Module (SIM)

13.1 Introduction

The System Integration Module (SIM) provides system control and chip configuration registers.

13.1.1 Features

Features of the SIM include:

- System clocking configuration
  - System clock divide values
  - Architectural clock gating control
- Flash and system RAM size configuration
- FlexTimer external clock, hardware trigger, and fault source selection
- UART0 and UART1 receive/transmit source selection/configuration
- Small Regulator configuration
- External ADC MUX configuration

13.2 Memory map and register definition

The SIM module contains many fields for selecting the clock source and dividers for various module clocks. See the Clock Distribution chapter for more information, including block diagrams and clock definitions.

NOTE

The SIM registers can be written only in supervisor mode. In user mode, write accesses are blocked and will result in a bus error.

NOTE

The SIM_SOPT1 register is located at a different base address than the other SIM registers.

SIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_7000	System Options Register 1 (SIM_SOPT1)	32	R/W	See section	13.2.1/169
4004_8004	System Options Register 2 (SIM_SOPT2)	32	R/W	0000_1000h	13.2.2/171
4004_800C	System Options Register 4 (SIM_SOPT4)	32	R/W	0000_0000h	13.2.3/172
4004_8010	System Options Register 5 (SIM_SOPT5)	32	R/W	0000_0000h	13.2.4/175
4004_8018	System Options Register 7 (SIM_SOPT7)	32	R/W	0000_0000h	13.2.5/176
4004_801C	System Options Register 8 (SIM_SOPT8)	32	R/W	0000_0000h	13.2.6/178
4004_8020	System Options Register 9 (SIM_SOPT9)	32	R/W	0000_0000h	13.2.7/181
4004_8024	System Device Identification Register (SIM_SDID)	32	R	See section	13.2.8/182
4004_8034	System Clock Gating Control Register 4 (SIM_SCGC4)	32	R/W	F000_0030h	13.2.9/184
4004_8038	System Clock Gating Control Register 5 (SIM_SCGC5)	32	R/W	0004_0182h	13.2.10/186
4004_803C	System Clock Gating Control Register 6 (SIM_SCGC6)	32	R/W	See section	13.2.11/188
4004_8040	System Clock Gating Control Register 7 (SIM_SCGC7)	32	R/W	See section	13.2.12/190
4004_8044	System Clock Divider Register 1 (SIM_CLKDIV1)	32	R/W	See section	13.2.13/191
4004_8048	System Clock Divider Register 2 (SIM_CLKDIV2)	32	R/W	0000_0000h	13.2.14/193
4004_804C	Flash Configuration Register 1 (SIM_FCFG1)	32	R/W	See section	13.2.14/193
4004_8050	Flash Configuration Register 2 (SIM_FCFG2)	32	R	See section	13.2.15/195
4004_8054	Unique Identification Register High (SIM_UIDH)	32	R/W	See section	13.2.16/195
4004_8058	Unique Identification Register Mid-High (SIM_UIDMH)	32	R/W	See section	13.2.17/196
4004_805C	Unique Identification Register Mid Low (SIM_UIDML)	32	R/W	See section	13.2.18/196
4004_8060	Unique Identification Register Low (SIM_UIDL)	32	R/W	See section	13.2.19/197

Table continues on the next page...

SIM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_8068	System Clock Divider Register 4 (SIM_CLKDIV4)	32	R/W	See section	13.2.20/197
4004_806C	Miscellaneous Control Register 0 (SIM_MISCTRL0)	32	R/W	0000_0000h	13.2.21/198
4004_8070	Miscellaneous Control Register 1 (SIM_MISCTRL1)	32	R/W	0000_0000h	13.2.22/200
4004_8100	WDOG Control Register (SIM_WDOGC)	32	R/W	0000_0000h	13.2.23/202
4004_8104	Power Control Register (SIM_PWRC)	32	R/W	0000_0101h	13.2.24/204
4004_8108	ADC Channel 6/7 Mux Control Register (SIM_ADCOPT)	32	R/W	0000_0000h	13.2.25/206

13.2.1 System Options Register 1 (SIM_SOPT1)

NOTE

The SOPT1 register is only reset on POR or LVD.

Address: 4004_7000h base + 0h offset = 4004_7000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												OSC32KSEL		0	
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.
- x = Undefined at reset.

SIM_SOPT1 field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 OSC32KSEL	32K oscillator clock select Selects the 32 kHz clock source (ERCLK32K) for LPTMR0. This field is reset only on POR/LVD. 00 System oscillator (OSC32KCLK) 01 Reserved

Table continues on the next page...

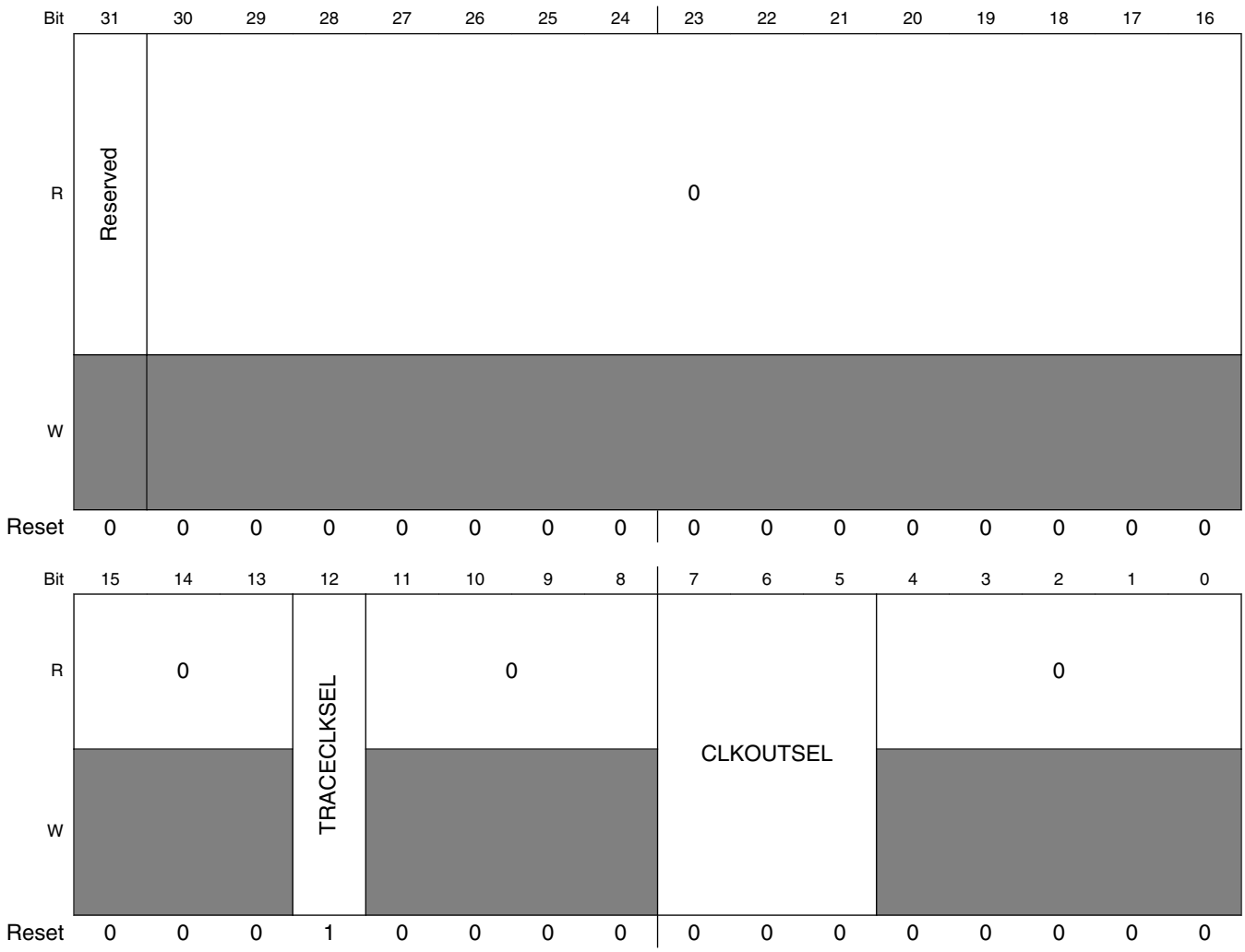
SIM_SOPT1 field descriptions (continued)

Field	Description
	10 Reserved 11 LPO 1 kHz
17–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 RAMSIZE	RAM size This field specifies the amount of system RAM available on the device. 0001 Reserved 0011 16 KB 0100 24 KB 0101 32 KB 0110 Reserved 0111 Reserved 1000 Reserved 1001 Reserved 1011 Reserved
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.2 System Options Register 2 (SIM_SOPT2)

SOPT2 contains the controls for selecting many of the module clock source options on this device. See the Clock Distribution chapter for more information including clocking diagrams and definitions of device clocks.

Address: 4004_7000h base + 1004h offset = 4004_8004h



SIM_SOPT2 field descriptions

Field	Description
31 Reserved	This field is reserved. This field must have a value of 0. If it is set to a value of 1, the NanoEdge PWM module will not work properly.
30–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SIM_SOPT2 field descriptions (continued)

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 TRACECLKSEL	Debug trace clock select Selects the core/system clock or MCG output clock (MCGOUTCLK) as the trace clock source. 0 MCGOUTCLK 1 Core/system clock
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–5 CLKOUTSEL	CLKOUT select Selects the clock to output on the CLKOUT pin. 000 Reserved 001 Reserved 010 Flash clock 011 LPO clock (1 kHz) 100 MCGIRCLK 101 OSCERCLK_UNDIV 110 OSCERCLK 111 Reserved
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.3 System Options Register 4 (SIM_SOPT4)

Address: 4004_7000h base + 100Ch offset = 4004_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	FTM3TRG2SRC	FTM3TRG1SRC	FTM3TRG0SRC	0					FTM1TRG2SRC	0	FTM1TRG0SRC	0	FTM0TRG2SRC	FTM0TRG1SRC	FTM0TRG0SRC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FTM3FLT0	0							FTM1FLT0	FTM0FLT3	FTM0FLT2	FTM0FLT1	FTM0FLT0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_SOPT4 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 FTM3TRG2SRC	FlexTimer 3 Hardware Trigger 2 Source Select Selects the source of FTM3 hardware trigger 2. 0 FTM3_FLT0 pin drives FTM3 hardware trigger 2 1 XBARA output 37 drives FTM3 hardware trigger 2
29 FTM3TRG1SRC	FlexTimer 3 Hardware Trigger 1 Source Select Selects the source of FTM3 hardware trigger 1. 0 PDB1 output trigger drives FTM3 hardware trigger 1 1 FTM1 channel match drives FTM3 hardware trigger 1
28 FTM3TRG0SRC	FlexTimer 3 Hardware Trigger 0 Source Select Selects the source of FTM3 hardware trigger 0. **NOTE:** Also in parallel SIM_OPT8[FTM3_SYNCBIT] will be fed to FTM3 Hardware Trigger 0. 0 CMP0 output drives FTM3 hardware trigger 0 1 FTM1 channel match drives FTM3 hardware trigger 0
27–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 FTM1TRG2SRC	FlexTimer 1 Hardware Trigger 2 Source Select Selects the source of FTM1 hardware trigger 2. 0 FTM1_FLT0 pin drives FTM1 hardware trigger 2 1 XBARA output 35 drives FTM1 hardware trigger 2
21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 FTM1TRG0SRC	FlexTimer 1 Hardware Trigger 0 Source Select Selects the source of FTM1 hardware trigger 0. **NOTE:** Also in parallel SIM_OPT8[FTM1_SYNCBIT] will be fed to FTM1 Hardware Trigger 0. 0 CMP0 output drives FTM1 hardware trigger 0 1 FTM0 channel match drives FTM1 hardware trigger 0
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 FTM0TRG2SRC	FlexTimer 0 Hardware Trigger 2 Source Select Selects the source of FTM0 hardware trigger 2. 0 FTM0_FLT0 pin drives FTM0 hardware trigger 2 1 XBARA output 34 drives FTM0 hardware trigger 2
17 FTM0TRG1SRC	FlexTimer 0 Hardware Trigger 1 Source Select Selects the source of FTM0 hardware trigger 1.

Table continues on the next page...

SIM_SOPT4 field descriptions (continued)

Field	Description
	0 PDB0 output trigger drives FTM0 hardware trigger 1 1 FTM1 channel match drives FTM0 hardware trigger 1
16 FTM0TRG0SRC	FlexTimer 0 Hardware Trigger 0 Source Select Selects the source of FTM0 hardware trigger 0. **NOTE:** Also in parallel SIM_OPT8[FTM0_SYNCBIT] will be fed to FTM0 Hardware Trigger 0. 0 CMP0 output drives FTM0 hardware trigger 0 1 FTM1 channel match drives FTM0 hardware trigger 0
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 FTM3FLT0	FTM3 Fault 0 Select Selects the source of FTM3 fault 0. **NOTE:** The pin source for fault 0 must be configured for the FTM module fault function through the appropriate pin control register in the port control module. 0 FTM3_FLT0 pin 1 CMP0 out
11–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 FTM1FLT0	FTM1 Fault 0 Select Selects the source of FTM1 fault 0. **NOTE:** The pin source for fault 0 must be configured for the FTM module fault function through the appropriate pin control register in the port control module. 0 FTM1_FLT0 pin 1 CMP0 out
3 FTM0FLT3	Selects the source of FTM0 fault 3. **NOTE:** The pin source for fault 3 must be configured for the FTM module fault function through the appropriate pin control register in the port control module. 0 FTM0_FLT3 pin 1 XBARA output 49
2 FTM0FLT2	FTM0 Fault 2 Select Selects the source of FTM0 fault 2. **NOTE:** The pin source for fault 2 must be configured for the FTM module fault function through the appropriate pin control register in the port control module. 0 FTM0_FLT2 pin 1 CMP2 out
1 FTM0FLT1	FTM0 Fault 1 Select Selects the source of FTM0 fault 1.

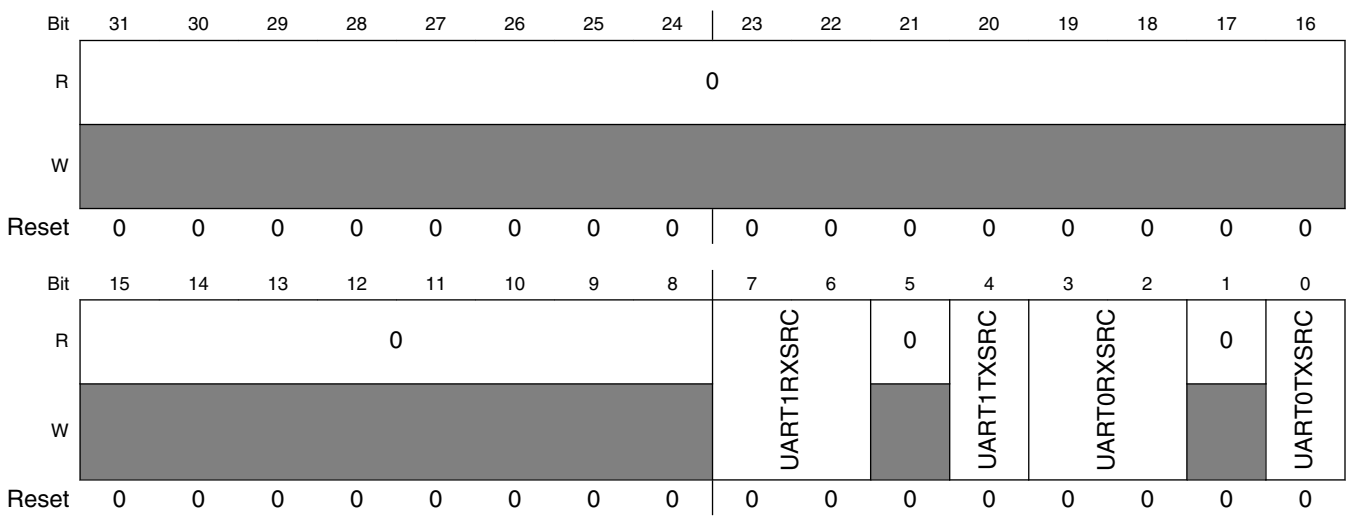
Table continues on the next page...

SIM_SOPT4 field descriptions (continued)

Field	Description
	NOTE: The pin source for fault 1 must be configured for the FTM module fault function through the appropriate pin control register in the port control module. 0 FTM0_FLT1 pin 1 CMP1 out
0 FTM0FLT0	FTM0 Fault 0 Select Selects the source of FTM0 fault 0. **NOTE:** The pin source for fault 0 must be configured for the FTM module fault function through the appropriate pin control register in the port control module. 0 FTM0_FLT0 pin 1 CMP0 out

13.2.4 System Options Register 5 (SIM_SOPT5)

Address: 4004_7000h base + 1010h offset = 4004_8010h



SIM_SOPT5 field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 UART1RXSRC	UART 1 receive data source select Selects the source for the UART 1 receive data. 00 UART1_RX pin 01 CMP0 10 CMP1 11 Reserved

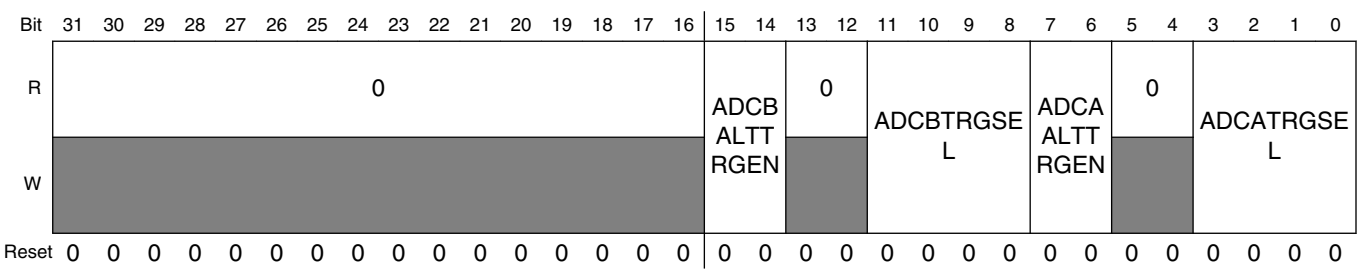
Table continues on the next page...

SIM_SOPT5 field descriptions (continued)

Field	Description
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 UART1TXSRC	UART 1 transmit data source select Selects the source for the UART 1 transmit data. 0 UART1_TX pin 1 UART1_TX pin modulated with FTM1 channel 0 output
3–2 UART0RXSRC	UART 0 receive data source select Selects the source for the UART 0 receive data. 00 UART0_RX pin 01 CMP0 10 CMP1 11 Reserved
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 UART0TXSRC	UART 0 transmit data source select Selects the source for the UART 0 transmit data. 0 UART0_TX pin 1 UART0_TX pin modulated with FTM1 channel 0 output

13.2.5 System Options Register 7 (SIM_SOPT7)

Address: 4004_7000h base + 1018h offset = 4004_8018h



SIM_SOPT7 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–14 ADCBALTTRGEN	ADCB alternate trigger enable Enable alternative conversion triggers for ADCB. 00 XBARA output 13.

Table continues on the next page...

SIM_SOPT7 field descriptions (continued)

Field	Description
	01 PDB1 trigger selected for ADCB 1- Alternate trigger selected for ADCB as defined by ADCBTRGSEL.
13–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 ADCBTRGSEL	ADCB trigger select Selects the ADCB trigger source when alternative triggers are functional in stop and VLPS modes. 0000 Reserved 0001 High speed comparator 0 output 0010 High speed comparator 1 output 0011 High speed comparator 2 output 0100 PIT trigger 0 0101 PIT trigger 1 0110 PIT trigger 2 0111 PIT trigger 3 1000 FTM0 trigger 1001 FTM1 trigger 1010 Reserved 1011 FTM3 trigger 1100 XBARA output 41 1101 Reserved 1110 Low-power timer trigger 1111 Reserved
7–6 ADCAALTTRGEN	ADCA alternate trigger enable Enable alternative conversion triggers for ADCA. 00 XBARA output 12. 01 PDB0 trigger selected for ADCA. 1- Alternate trigger selected for ADCA.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ADCATRGSEL	ADCA trigger select Selects the ADCA trigger source when alternative triggers are functional in stop and VLPS modes. . 0000 PDB external trigger pin input (PDB0_EXTRG) 0001 High speed comparator 0 output 0010 High speed comparator 1 output 0011 High speed comparator 2 output 0100 PIT trigger 0 0101 PIT trigger 1 0110 PIT trigger 2 0111 PIT trigger 3 1000 FTM0 trigger 1001 FTM1 trigger 1010 Reserved

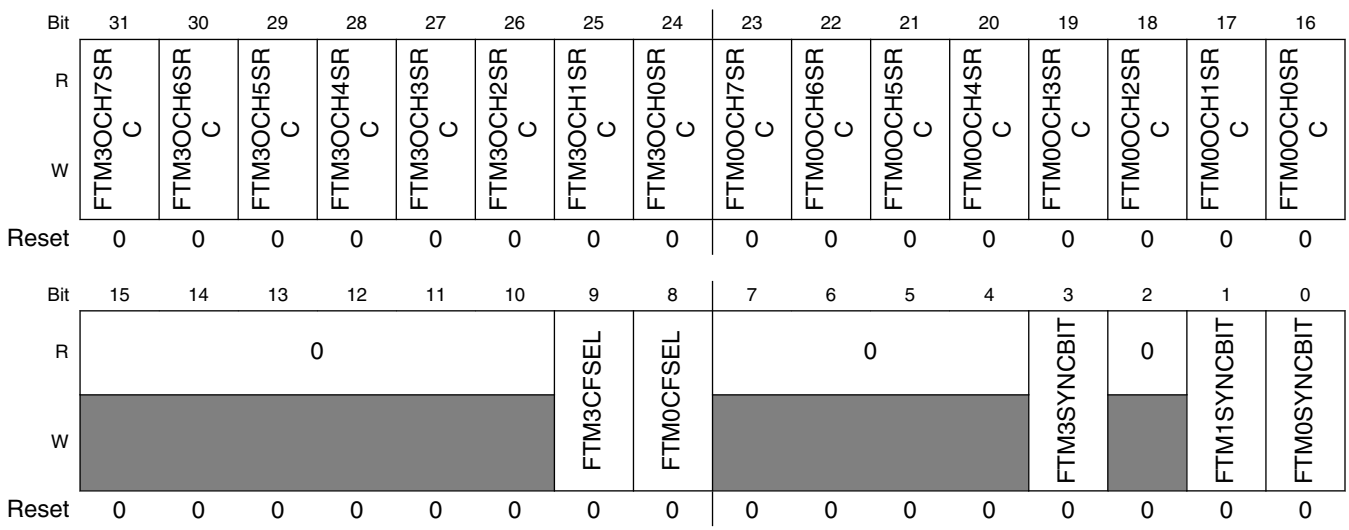
Table continues on the next page...

SIM_SOPT7 field descriptions (continued)

Field	Description
	1011 FTM3 trigger 1100 XBARA output 38 1101 Reserved 1110 Low-power timer trigger 1111 Reserved

13.2.6 System Options Register 8 (SIM_SOPT8)

Address: 4004_7000h base + 101Ch offset = 4004_801Ch



SIM_SOPT8 field descriptions

Field	Description
31 FTM3OCH7SRC	FTM3 channel 7 output source 0 FTM3_CH7 pin is output of FTM3 channel 7 output 1 FTM3_CH7 pin is output of FTM3 channel 7 output modulated by carrier frequency clock, as per FTM3CFSEL.
30 FTM3OCH6SRC	FTM3 channel 6 output source 0 FTM3_CH6 pin is output of FTM3 channel 6 output 1 FTM3_CH6 pin is output of FTM3 channel 6 output modulated by carrier frequency clock, as per FTM3CFSEL.
29 FTM3OCH5SRC	FTM3 channel 5 output source 0 FTM3_CH5 pin is output of FTM3 channel 5 output 1 FTM3_CH5 pin is output of FTM3 channel 5 output modulated by carrier frequency clock, as per FTM3CFSEL.
28 FTM3OCH4SRC	FTM3 channel 4 output source

Table continues on the next page...

SIM_SOPT8 field descriptions (continued)

Field	Description
	0 FTM3_CH4 pin is output of FTM3 channel 4 output 1 FTM3_CH4 pin is output of FTM3 channel 4 output modulated by carrier frequency clock, as per FTM3CFSEL.
27 FTM3OCH3SRC	FTM3 channel 3 output source 0 FTM3_CH3 pin is output of FTM3 channel 3 output 1 FTM3_CH3 pin is output of FTM3 channel 3 output modulated by carrier frequency clock, as per FTM3CFSEL.
26 FTM3OCH2SRC	FTM3 channel 2 output source 0 FTM3_CH2 pin is output of FTM3 channel 2 output 1 FTM3_CH2 pin is output of FTM3 channel 2 output modulated by carrier frequency clock, as per FTM3CFSEL.
25 FTM3OCH1SRC	FTM3 channel 1 output source 0 FTM3_CH1 pin is output of FTM3 channel 1 output 1 FTM3_CH1 pin is output of FTM3 channel 1 output modulated by carrier frequency clock, as per FTM3CFSEL.
24 FTM3OCH0SRC	FTM3 channel 0 output source 0 FTM3_CH0 pin is output of FTM3 channel 0 output 1 FTM3_CH0 pin is output of FTM3 channel 0 output modulated by carrier frequency clock, as per FTM3CFSEL.
23 FTM0OCH7SRC	FTM0 channel 7 output source 0 FTM0_CH7 pin is output of FTM0 channel 7 output 1 FTM0_CH7 pin is output of FTM0 channel 7 output, modulated by carrier frequency clock, as per FTM0CFSEL
22 FTM0OCH6SRC	FTM0 channel 6 output source 0 FTM0_CH6 pin is output of FTM0 channel 6 output 1 FTM0_CH6 pin is output of FTM0 channel 6 output, modulated by carrier frequency clock, as per FTM0CFSEL
21 FTM0OCH5SRC	FTM0 channel 5 output source 0 FTM0_CH5 pin is output of FTM0 channel 5 output 1 FTM0_CH5 pin is output of FTM0 channel 5 output, modulated by carrier frequency clock, as per FTM0CFSEL
20 FTM0OCH4SRC	FTM0 channel 4 output source 0 FTM0_CH4 pin is output of FTM0 channel 4 output 1 FTM0_CH4 pin is output of FTM0 channel 4 output, modulated by carrier frequency clock, as per FTM0CFSEL
19 FTM0OCH3SRC	FTM0 channel 3 output source 0 FTM0_CH3 pin is output of FTM0 channel 3 output 1 FTM0_CH3 pin is output of FTM0 channel 3 output, modulated by carrier frequency clock, as per FTM0CFSEL
18 FTM0OCH2SRC	FTM0 channel 2 output source

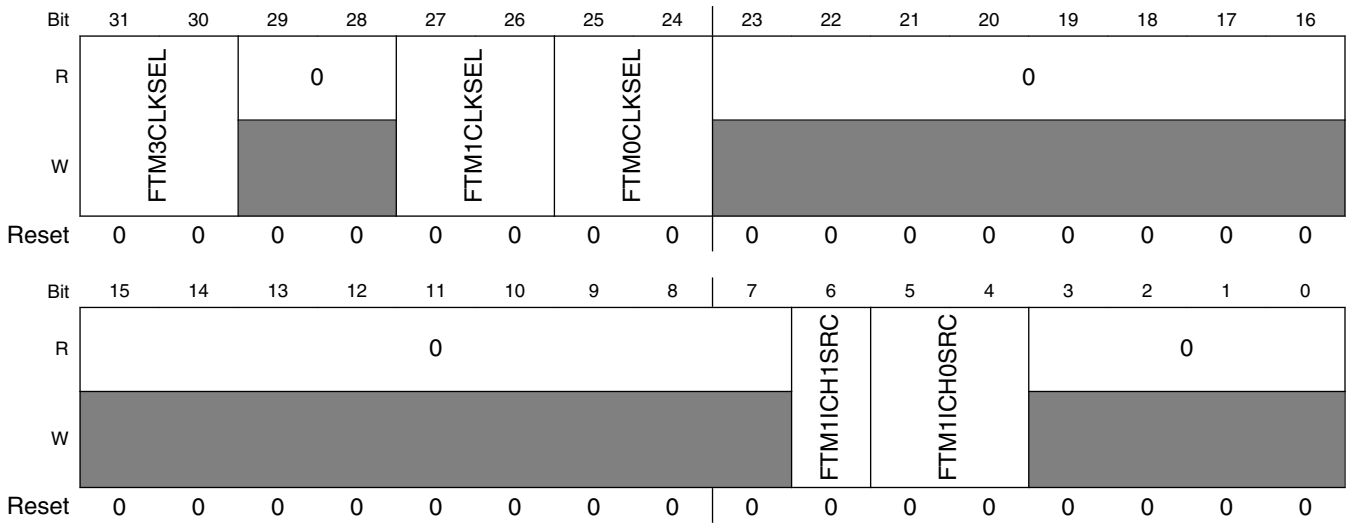
Table continues on the next page...

SIM_SOPT8 field descriptions (continued)

Field	Description
	0 FTM0_CH2 pin is output of FTM0 channel 2 output 1 FTM0_CH2 pin is output of FTM0 channel 2 output, modulated by carrier frequency clock, as per FTM0CFSEL
17 FTM0OCH1SRC	FTM0 channel 1 output source 0 FTM0_CH1 pin is output of FTM0 channel 1 output 1 FTM0_CH1 pin is output of FTM0 channel 1 output, modulated by carrier frequency clock, as per FTM0CFSEL
16 FTM0OCH0SRC	FTM0 channel 0 output source 0 FTM0_CH0 pin is output of FTM0 channel 0 output 1 FTM0_CH0 pin is output of FTM0 channel 0 output, modulated by carrier frequency clock, as per FTM0CFSEL
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 FTM3CFSEL	Carrier frequency selection for FTM3 output channel 0 FTM1 channel 1 output provides the carrier signal for FTM3 Timer Modulation mode. 1 LPTMR0 prescaler output provides the carrier signal for FTM3 Timer Modulation mode.
8 FTM0CFSEL	Carrier frequency selection for FTM0 output channel 0 FTM1 channel 1 output provides the carrier signal for FTM0 Timer Modulation mode. 1 LPTMR0 prescaler output provides the carrier signal for FTM0 Timer Modulation mode.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FTM3SYNCBIT	FTM3 Hardware Trigger 0 Software Synchronization 0 No effect. 1 Write 1 to assert the TRIG0 input to FTM3, software must clear this bit to allow other trigger sources to assert.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FTM1SYNCBIT	FTM1 Hardware Trigger 0 Software Synchronization 0 No effect. 1 Write 1 to assert the TRIG0 input to FTM1, software must clear this bit to allow other trigger sources to assert.
0 FTM0SYNCBIT	FTM0 Hardware Trigger 0 Software Synchronization 0 No effect 1 Write 1 to assert the TRIG0 input to FTM0, software must clear this bit to allow other trigger sources to assert.

13.2.7 System Options Register 9 (SIM_SOPT9)

Address: 4004_7000h base + 1020h offset = 4004_8020h



SIM_SOPT9 field descriptions

Field	Description
31–30 FTM3CLKSEL	FlexTimer 3 External Clock Pin Select Selects the external pin used to drive the clock to the FTM3 module. **NOTE:** The selected pin must also be configured for the FTM3 module external clock function through the appropriate pin control register in the port control module. 00 FTM3 external clock driven by FTM_CLK0 pin 01 FTM3 external clock driven by FTM_CLK1 pin 10 FTM3 external clock driven by FTM_CLK2 pin 11 Reserved
29–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–26 FTM1CLKSEL	FlexTimer 1 External Clock Pin Select Selects the external pin used to drive the clock to the FTM1 module. **NOTE:** The selected pin must also be configured for the FTM1 module external clock function through the appropriate pin control register in the port control module. 00 FTM1 external clock driven by FTM_CLK0 pin 01 FTM1 external clock driven by FTM_CLK1 pin 10 FTM1 external clock driven by FTM_CLK2 pin 11 Reserved
25–24 FTM0CLKSEL	FlexTimer 0 External Clock Pin Select Selects the external pin used to drive the clock to the FTM0 module.

Table continues on the next page...

SIM_SOPT9 field descriptions (continued)

Field	Description
	NOTE: The selected pin must also be configured for the FTM0 module external clock function through the appropriate pin control register in the port control module. 00 FTM0 external clock driven by FTM_CLK0 pin 01 FTM0 external clock driven by FTM_CLK1 pin 10 FTM0 external clock driven by FTM_CLK2 pin 11 Reserved
23–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FTM1ICH1SRC	FTM1 channel 0 input capture source select Selects the source for FTM1 channel 0 input capture. **NOTE:** When the FTM is not in input capture mode, clear this field. 0 FTM1_CH1 signal 1 Exclusive OR of FTM1_CH1, FTM1_CH0 and XBARA output 42
5–4 FTM1ICH0SRC	FTM1 channel 0 input capture source select Selects the source for FTM1 channel 0 input capture. **NOTE:** When the FTM is not in input capture mode, clear this field. 00 FTM1_CH0 signal 01 CMP0 output 10 CMP1 output 11 Reserved
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.8 System Device Identification Register (SIM_SDID)

Address: 4004_7000h base + 1024h offset = 4004_8024h

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.
- x = Undefined at reset.

SIM_SDID field descriptions

Field	Description
31–28 FAMILYID	Kinetis Family ID Specifies the Kinetis family of the device.

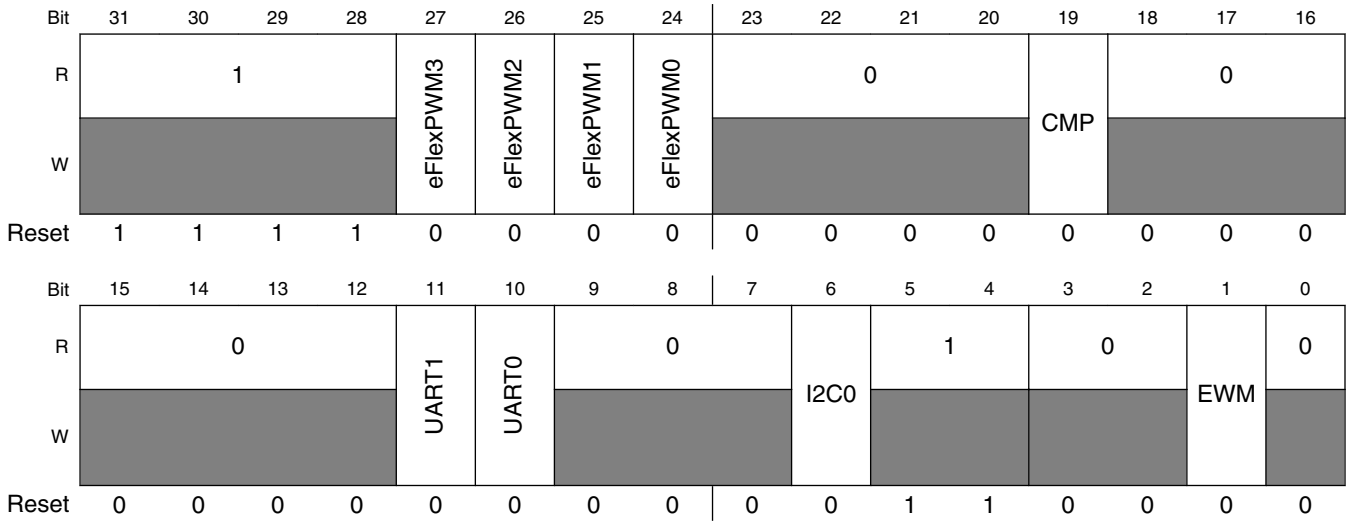
Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
	0100 Kinetis family of this device. This is the Vseries.
27–24 SUBFAMID	Kinetis Sub-Family ID Specifies the Kinetis sub-family of the device. 0001 KVx2 Subfamily (FlexTimer and ADC) 0100 KVx4 Subfamily (eFlexPWM and ADC) 0110 KVx6 Subfamily (eFlexPWM with FlexTimer and ADC)
23–20 SERIESID	Kinetis Series ID Specifies the Kinetis series of the device. 0000 Kinetis K series 0001 Kinetis L series 0101 Kinetis W series 0110 Kinetis V series
19–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 REVID	Device revision number Specifies the silicon implementation number for the device.
11–7 DIEID	Device die number Specifies the silicon implementation number for the device.
6–4 Reserved	This field can reset to 0 or 1. This field is reserved.
PINID	Pincount identification Specifies the pincount of the device. 0000 Reserved 0001 Reserved 0010 Reserved 0011 Reserved 0100 48-pin 0101 64-pin 0110 Reserved 0111 Reserved 1000 100-pin 1001 Reserved 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved

13.2.9 System Clock Gating Control Register 4 (SIM_SCGC4)

Address: 4004_7000h base + 1034h offset = 4004_8034h



SIM_SCGC4 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
27 eFlexPWM3	eFlexPWM submodule 3 Clock Gate Control This bit controls the clock gate to the PWM submodule 3. 0 Clock disabled 1 Clock enabled
26 eFlexPWM2	eFlexPWM submodule 2 Clock Gate Control This bit controls the clock gate to the PWM submodule 2. 0 Clock disabled 1 Clock enabled
25 eFlexPWM1	eFlexPWM submodule 1 Clock Gate Control This bit controls the clock gate to the PWM submodule 1. 0 Clock disabled 1 Clock enabled
24 eFlexPWM0	eFlexPWM submodule 0 Clock Gate Control This bit controls the clock gate to the PWM submodule 0. 0 Clock disabled 1 Clock enabled
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

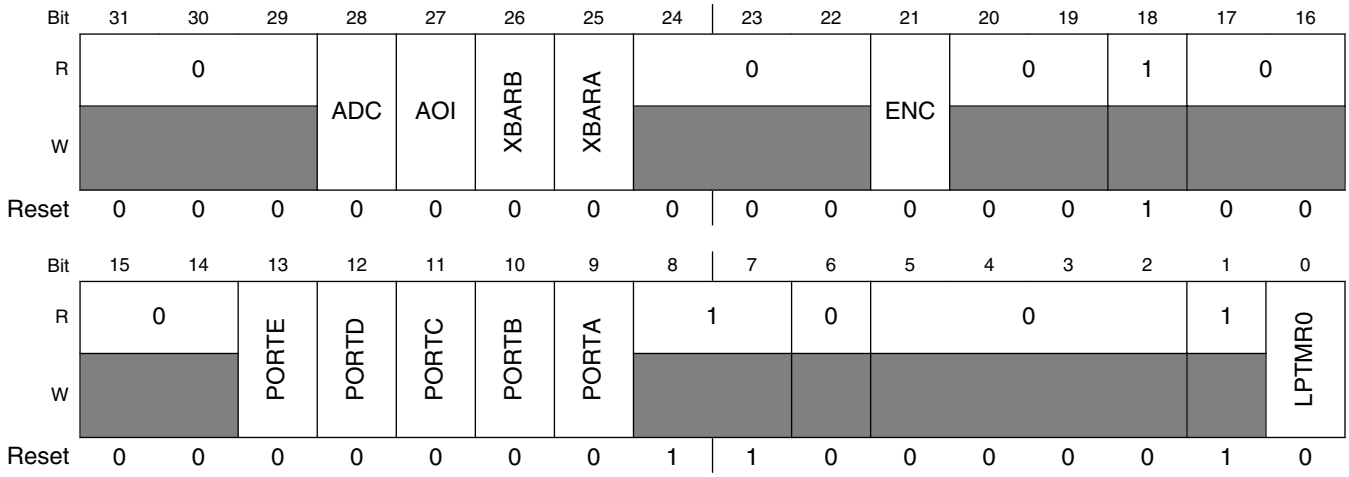
Table continues on the next page...

SIM_SCGC4 field descriptions (continued)

Field	Description
19 CMP	Comparators Clock Gate Control This bit controls the clock gate to the comparator module. 0 Clock disabled 1 Clock enabled
18–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 UART1	UART1 Clock Gate Control This bit controls the clock gate to the UART1 module. 0 Clock disabled 1 Clock enabled
10 UART0	UART0 Clock Gate Control This bit controls the clock gate to the UART0 module. 0 Clock disabled 1 Clock enabled
9–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 I2C0	I2C0 Clock Gate Control This bit controls the clock gate to the I^2C0 module. 0 Clock disabled 1 Clock enabled
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 EWM	EWM Clock Gate Control This bit controls the clock gate to the EWM module. 0 Clock disabled 1 Clock enabled
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.10 System Clock Gating Control Register 5 (SIM_SCGC5)

Address: 4004_7000h base + 1038h offset = 4004_8038h



SIM_SCGC5 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 ADC	ADC Clock Gate Control This bit controls the clock gate to the ADC module. 0 Clock disabled 1 Clock enabled
27 AOI	AOI Clock Gate Control This bit controls the clock gate to the AOI module. 0 Clock disabled 1 Clock enabled
26 XBARB	XBARB Clock Gate Control This bit controls the clock gate to the XBARB module. 0 Clock disabled 1 Clock enabled
25 XBARA	XBARA Clock Gate Control This bit controls the clock gate to the XBARA module. 0 Clock disabled 1 Clock enabled
24–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SIM_SCGC5 field descriptions (continued)

Field	Description
21 ENC	This bit controls the clock gate to the ENC module. 0 Clock disabled 1 Clock enabled
20–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
17–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 PORTE	Port E Clock Gate Control This bit controls the clock gate to the Port E module. 0 Clock disabled 1 Clock enabled
12 PORTD	Port D Clock Gate Control This bit controls the clock gate to the Port D module. 0 Clock disabled 1 Clock enabled
11 PORTC	Port C Clock Gate Control This bit controls the clock gate to the Port C module. 0 Clock disabled 1 Clock enabled
10 PORTB	Port B Clock Gate Control This bit controls the clock gate to the Port B module. 0 Clock disabled 1 Clock enabled
9 PORTA	Port A Clock Gate Control This bit controls the clock gate to the Port A module. 0 Clock disabled 1 Clock enabled
8–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
0 LPTMR0	Low Power Timer Access Control

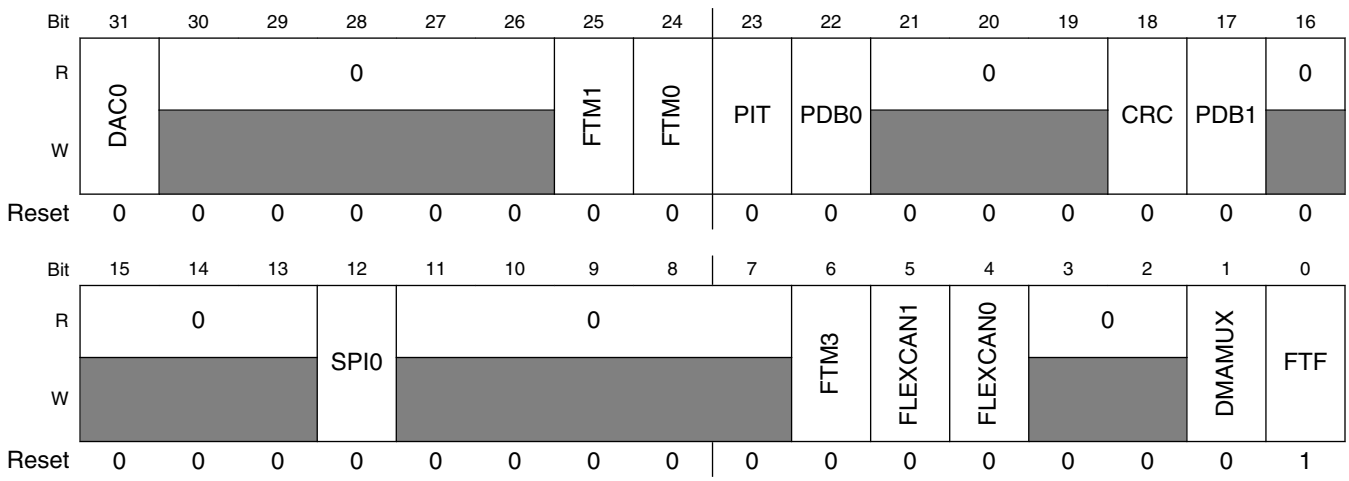
Table continues on the next page...

SIM_SCGC5 field descriptions (continued)

Field	Description
	This bit controls software access to the Low Power Timer module. 0 Access disabled 1 Access enabled

13.2.11 System Clock Gating Control Register 6 (SIM_SCGC6)

Address: 4004_7000h base + 103Ch offset = 4004_803Ch



SIM_SCGC6 field descriptions

Field	Description
31 DAC0	DAC0 Clock Gate Control This bit controls the clock gate to the DAC0 module. 0 Clock disabled 1 Clock enabled
30–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 FTM1	FTM1 Clock Gate Control This bit controls the clock gate to the FTM1 module. 0 Clock disabled 1 Clock enabled
24 FTM0	FTM0 Clock Gate Control This bit controls the clock gate to the FTM0 module. 0 Clock disabled 1 Clock enabled

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

Field	Description
23 PIT	PIT Clock Gate Control This bit controls the clock gate to the PIT module. 0 Clock disabled 1 Clock enabled
22 PDB0	PDB0 Clock Gate Control This bit controls the clock gate to the PDB0 module. 0 Clock disabled 1 Clock enabled
21–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 CRC	CRC Clock Gate Control This bit controls the clock gate to the CRC module. 0 Clock disabled 1 Clock enabled
17 PDB1	PDB1 Clock Gate Control This bit controls the clock gate to the PDB1 module. 0 Clock disabled 1 Clock enabled
16–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 SPI0	SPI0 Clock Gate Control This bit controls the clock gate to the SPI0 module. 0 Clock disabled 1 Clock enabled
11–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FTM3	FTM3 Clock Gate Control This bit controls the clock gate to the FTM3 module. 0 Clock disabled 1 Clock enabled
5 FLEXCAN1	FlexCAN1 Clock Gate Control This bit controls the clock gate to the FlexCAN1 module. 0 Clock disabled 1 Clock enabled
4 FLEXCAN0	FlexCAN0 Clock Gate Control This bit controls the clock gate to the FlexCAN0 module.

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

Field	Description
	0 Clock disabled 1 Clock enabled
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 DMAMUX	DMA Mux Clock Gate Control This bit controls the clock gate to the DMA Mux module. 0 Clock disabled 1 Clock enabled
0 FTF	Flash Memory Clock Gate Control This bit controls the clock gate to the flash memory. Flash reads are still supported while the flash memory is clock gated, but entry into low power modes is blocked. 0 Clock disabled 1 Clock enabled

13.2.12 System Clock Gating Control Register 7 (SIM_SCGC7)

Address: 4004_7000h base + 1040h offset = 4004_8040h

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

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

SIM_SCGC7 field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 DMA	DMA Clock Gate Control This bit controls the clock gate to the DMA module. 0 Clock disabled 1 Clock enabled
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.13 System Clock Divider Register 1 (SIM_CLKDIV1)

When updating CLKDIV1, update all fields using the one write command. Attempting to write an invalid clock ratio to the CLKDIV1 register will cause the write to be ignored. The maximum divide ratio that can be programmed between core/system clock and the other divided clocks is divide by 8. When OUTDIV1 equals 0000 (divide by 1), the other dividers cannot be set higher than 0111 (divide by 8).

NOTE

The CLKDIV1 register cannot be written to when the device is in VLPR mode.

Address: 4004_7000h base + 1044h offset = 4004_8044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OUTDIV1				OUTDIV2				0				OUTDIV4				0															
W																																
Reset	0	x*	x*	x*	0	x*	x*	x*	0*	0*	0*	0*	0	x*	x*	x*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- Reset value loaded during Syetem Reset from FTF_FOPT[LPBOOT].
- x = Undefined at reset.

SIM_CLKDIV1 field descriptions

Field	Description
31–28 OUTDIV1	Clock 1 output divider value This field sets the divide value for the core/system clock from MCGOUTCLK. At the end of reset, it is loaded with either 0000 or 0111 depending on FTF_FOPT[LPBOOT]. 0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.
27–24 OUTDIV2	Clock 2 output divider value

Table continues on the next page...

SIM_CLKDIV1 field descriptions (continued)

Field	Description
	This field sets the divide value for the fast bus clock from MCGOUTCLK. At the end of reset, it is loaded with either 0000 or 0111 depending on FTF_FOPT[LPBOOT]. The bus clock frequency must be an integer divide of the core/system clock frequency. 0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 OUTDIV4	Clock 4 output divider value This field sets the divide value for the bus/flash clock from MCGOUTCLK. At the end of reset, it is loaded with either 0001 or 1111 depending on FTF_FOPT[LPBOOT]. The flash clock frequency must be an integer divide of the system clock frequency. 0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.14 System Clock Divider Register 2 (SIM_CLKDIV2)

Address: 4004_7000h base + 1048h offset = 4004_8048h

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

SIM_CLKDIV2 field descriptions

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

13.2.14 Flash Configuration Register 1 (SIM_FCFG1)

Address: 4004_7000h base + 104Ch offset = 4004_804Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	1				PFSIZE				0				1			
W																
Reset	1	1	1	1	x*	x*	x*	x*	0*	0*	0*	0*	1	1	1	1

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

* Notes:

- Reset value loaded during System Reset from Flash IFR.
- x = Undefined at reset.

SIM_FCFG1 field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
27–24 PFSIZE	Program flash size This field specifies the amount of program flash memory available on the device, as set by IFR bits. These bits are used for device testing only and are read-only. 0101 64 KB of program flash memory 0111 128 KB of program flash memory 1001 256 KB of program flash memory
23–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FLASHDOZE	Flash Doze When set, Flash memory is disabled for the duration of Wait mode. An attempt by the DMA or other bus master to access the Flash when the Flash is disabled will result in a bus error. This bit should be clear during VLP modes. The Flash will be automatically enabled again at the end of Wait mode so interrupt vectors do not need to be relocated out of Flash memory. The wakeup time from Wait mode is extended when this bit is set. 0 Flash remains enabled during Wait mode 1 Flash is disabled for the duration of Wait mode
0 FLASHDIS	Flash Disable Flash accesses are disabled (and generate a bus error) and the Flash memory is placed in a low power state. This bit should not be changed during VLP modes. Relocate the interrupt vectors out of Flash memory before disabling the Flash. 0 Flash is enabled 1 Flash is disabled

13.2.15 Flash Configuration Register 2 (SIM_FCFG2)

Address: 4004_7000h base + 1050h offset = 4004_8050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAXADDR0							1	0						
W																
Reset	0*	x*	x*	x*	x*	x*	x*	x*	1	0*	0*	0*	0*	0*	0*	0*

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

* Notes:
- Reset value loaded during System Reset from Flash IFR.
- x = Undefined at reset.

SIM_FCFG2 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30–24 MAXADDR0	Max address block 0 This field concatenated with 13 trailing zeros indicates the first invalid address of flash block 0 (program flash 0). For example, if MAXADDR0 = 0x20 the first invalid address of flash block 0 is 0x0004_0000. This would be the MAXADDR0 value for a device with 256 KB program flash in flash block 0.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.16 Unique Identification Register High (SIM_UIDH)

Address: 4004_7000h base + 1054h offset = 4004_8054h

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

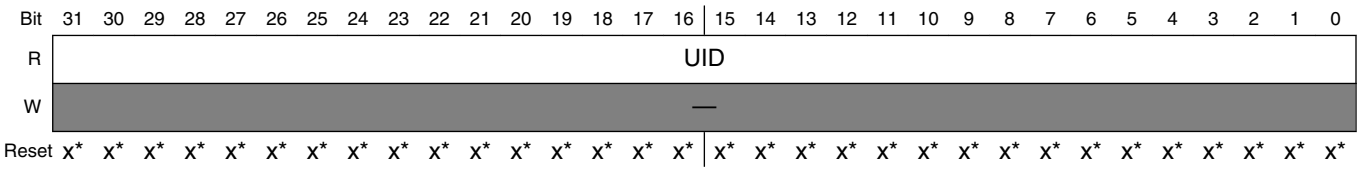
* Notes:
- Reset value loaded during System Reset from Flash IFR.x = Undefined at reset.

SIM_UIDH field descriptions

Field	Description
UID	Unique Identification Unique identification for the device.

13.2.17 Unique Identification Register Mid-High (SIM_UIDMH)

Address: 4004_7000h base + 1058h offset = 4004_8058h



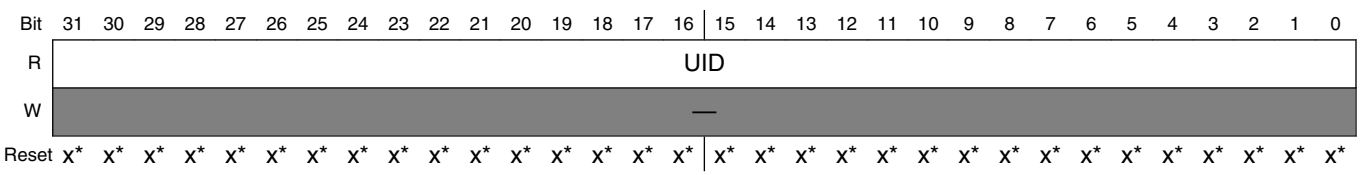
* Notes:
- Reset value loaded during System Reset from Flash IFR.x = Undefined at reset.

SIM_UIDMH field descriptions

Field	Description
UID	Unique Identification Unique identification for the device.

13.2.18 Unique Identification Register Mid Low (SIM_UIDML)

Address: 4004_7000h base + 105Ch offset = 4004_805Ch



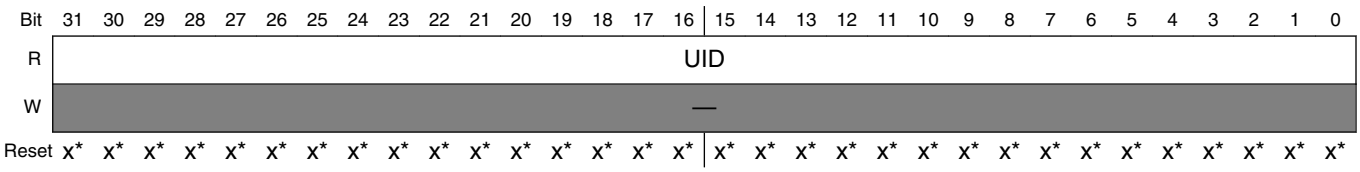
* Notes:
- Reset value loaded during System Reset from Flash IFR.x = Undefined at reset.

SIM_UIDML field descriptions

Field	Description
UID	Unique Identification Unique identification for the device.

13.2.19 Unique Identification Register Low (SIM_UIDL)

Address: 4004_7000h base + 1060h offset = 4004_8060h



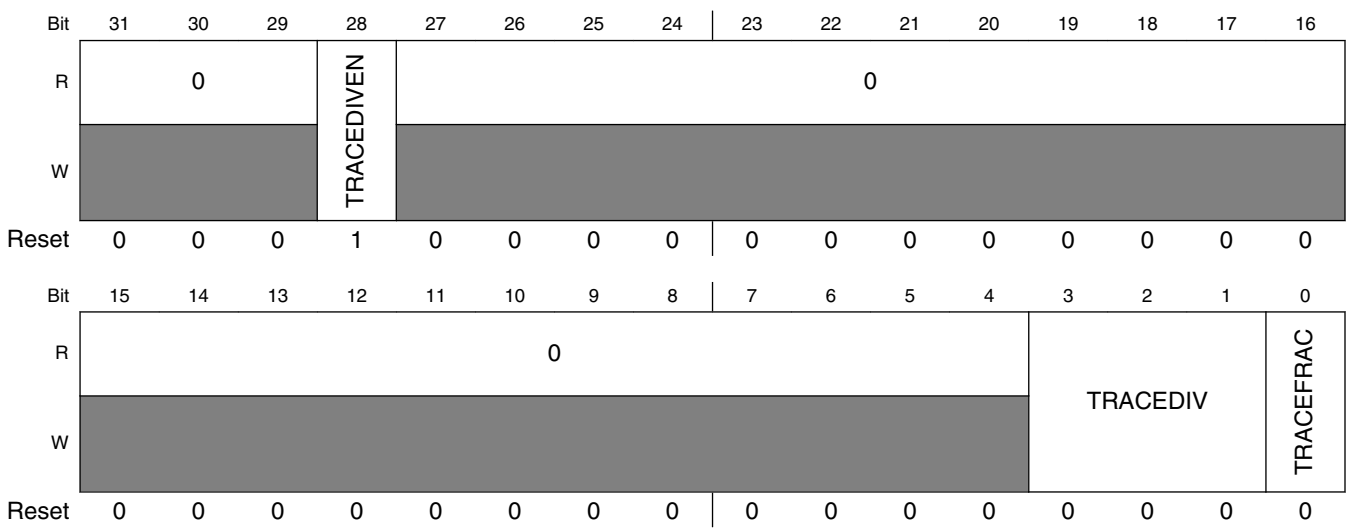
* Notes:
- Reset value loaded during System Reset from Flash IFR.x = Undefined at reset.

SIM_UIDL field descriptions

Field	Description
UID	Unique Identification Unique identification for the device.

13.2.20 System Clock Divider Register 4 (SIM_CLKDIV4)

Address: 4004_7000h base + 1068h offset = 4004_8068h



SIM_CLKDIV4 field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 TRACEDIVEN	Debug Trace Divider Control This bit controls the Debug Trace Divider.

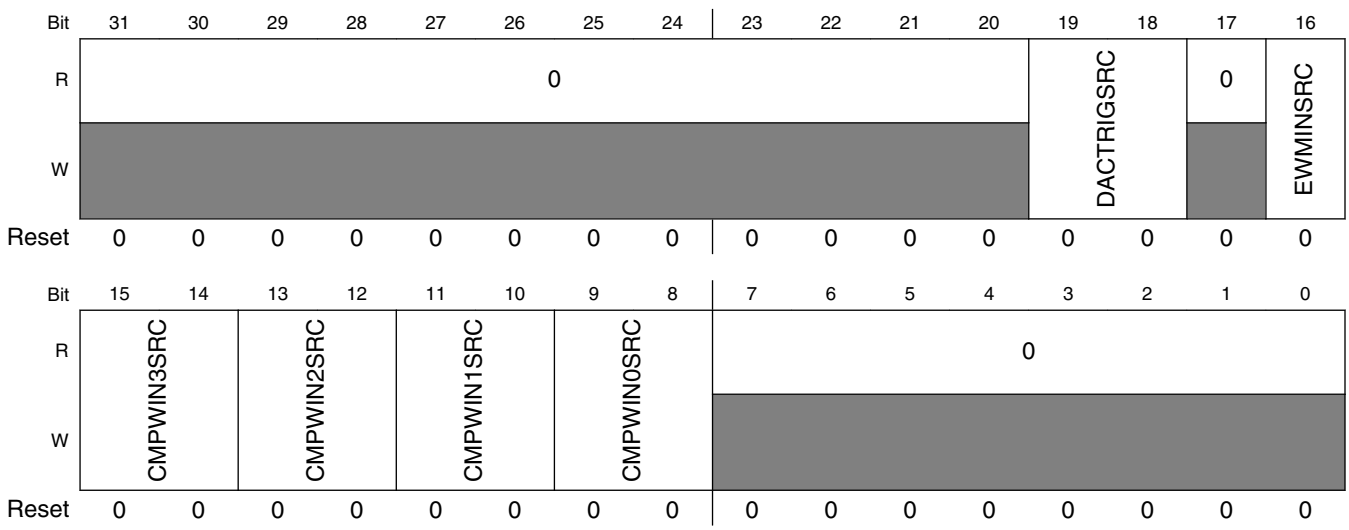
Table continues on the next page...

SIM_CLKDIV4 field descriptions (continued)

Field	Description
	0 Debug trace divider disabled 1 Debug trace divider enabled
27–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–1 TRACEDIV	Trace clock divider divisor This field sets the divide value for the fractional clock divider used as a source for trace clock. The source clock for the fractional clock divider is set by the SOPT2 TRACECLKSEL register bit. Divider output clock = Divider input clock * ((TRACEFRAC+1)/(TRACEDIV+1))
0 TRACEFRAC	Trace clock divider fraction This field sets the divide value for the fractional clock divider used as a source for trace clock. The source clock for the fractional clock divider is set by the SOPT2 TRACECLKSEL register bit. Divider output clock = Divider input clock*((TRACEFRAC+1)/(TRACEDIV+1))

13.2.21 Miscellaneous Control Register 0 (SIM_MISCTRL0)

Address: 4004_7000h base + 106Ch offset = 4004_806Ch



SIM_MISCTRL0 field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 DACTRIGSRC	DAC0 Hardware Trigger Input Source 00 XBARA output 15. 01 DAC0 can be triggered by both PDB0 interval trigger 0 and PDB1 interval trigger 0. 10 PDB0 interval trigger 0 11 PDB1 interval trigger 0

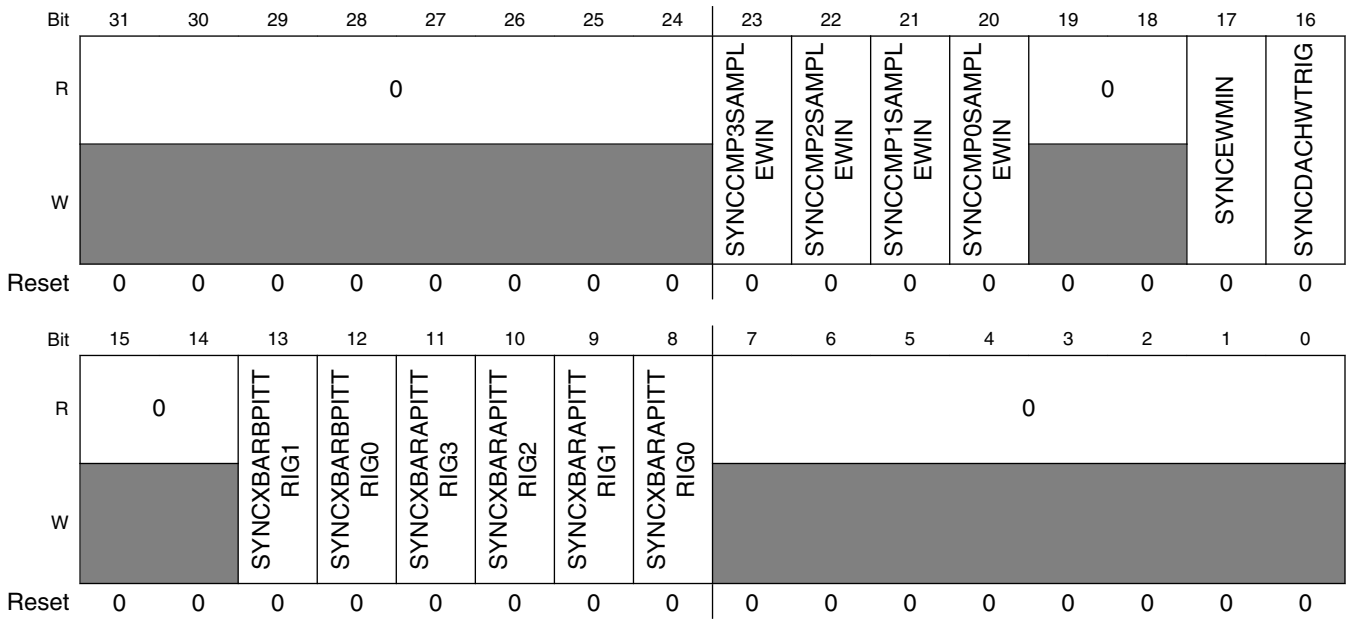
Table continues on the next page...

SIM_MISCTRL0 field descriptions (continued)

Field	Description
17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 EWMINSRC	EWM_IN Source This bit controls the ewm_in source of EWM module. 0 XBARA output 58. 1 EWM_IN pin
15–14 CMPWIN3SRC	CMP Sample/Window Input 3 Source 00 XBARA output 19. 01 CMP3 Sample/Window input is driven by both PDB0 and PDB1 pluse-out channel 3. 10 PDB0 pluse-out channel 3. 11 PDB1 pluse-out channel 3.
13–12 CMPWIN2SRC	CMP Sample/Window Input 2 Source 00 XBARA output 18. 01 CMP2 Sample/Window input is driven by both PDB0 and PDB1 pluse-out channel 2. 10 PDB0 pluse-out channel 2. 11 PDB1 pluse-out channel 2.
11–10 CMPWIN1SRC	CMP Sample/Window Input 1 Source 00 XBARA output 17. 01 CMP1 Sample/Window input is driven by both PDB0 and PDB1 pluse-out channel 1. 10 PDB0 pluse-out channel 1. 11 PDB1 pluse-out channel 1.
9–8 CMPWIN0SRC	CMP Sample/Window Input 0 Source 00 XBARA output 16. 01 CMP0 Sample/Window input is driven by both PDB0 and PDB1 pluse-out channel 0. 10 PDB0 pluse-out channel 0. 11 PDB1 pluse-out channel 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.22 Miscellaneous Control Register 1 (SIM_MISCTRL1)

Address: 4004_7000h base + 1070h offset = 4004_8070h



SIM_MISCTRL1 field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 SYNCCMP3SAMPLEWIN	Synchronize XBARA's output for CMP3's Sample/Window Input with flash/slow clock This field controls the synchronizer between XBARA's output and CMP3's sample/window input. **NOTE:** Set this bit if the CMP3's sample/window input isn't from flash/slow peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
22 SYNCCMP2SAMPLEWIN	Synchronize XBARA's output for CMP2's Sample/Window Input with flash/slow clock This field controls the synchronizer between XBARA's output and CMP2's sample/window input. **NOTE:** Set this bit if the CMP2's sample/window input isn't from flash/slow peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
21 SYNCCMP1SAMPLEWIN	Synchronize XBARA's output for CMP1's Sample/Window Input with flash/slow clock This field controls the synchronizer between XBARA's output and CMP1's sample/window input. **NOTE:** Set this bit if the CMP1's sample/window input isn't from flash/slow peripherials through xbar.

Table continues on the next page...

SIM_MISCTRL1 field descriptions (continued)

Field	Description
	0 Disable, bypass synchronizer. 1 Enable.
20 SYNCCMP0SAMPLEWIN	Synchronize XBARA's output for CMP0's Sample/Window Input with flash/slow clock This field controls the synchronizer between XBARA's output and CMP0's sample/window input. **NOTE:** Set this bit if the CMP0's sample/window input isn't from flash/slow peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
19–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 SYNCEWMIN	Synchronize XBARA's output for EWM's ewm_in with flash/slow clock This field controls the synchronizer between XBARA's output and EWM's ewm_in input. **NOTE:** Set this bit if the EWM's ewm_in isn't from flash/slow peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
16 SYNCDACHWTRIG	Synchronize XBARA's output for DAC Hardware Trigger with flash/slow clock This field controls the synchronizer between XBARA's output and DAC hardware trigger. **NOTE:** Set this bit if the DAC hardware trigger isn't from flash/slow peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 SYNCXBARBPITTRIG1	Synchronize XBARB's Input PIT Trigger 1 with fast clock This field controls the synchronizer between PIT trigger 1 and XBARB's input. **NOTE:** Set this bit if the XBARB's input PIT trigger 1 is fed into fast peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
12 SYNCXBARBPITTRIG0	Synchronize XBARB's Input PIT Trigger 0 with fast clock This field controls the synchronizer between PIT trigger 0 and XBARB's input. **NOTE:** Set this bit if the XBARB's input PIT trigger 0 is fed into fast peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
11 SYNCXBARAPITTRIG3	Synchronize XBARA's Input PIT Trigger 3 with fast clock This field controls the synchronizer between PIT trigger 3 and XBARA's input. **NOTE:** Set this bit if the XBARA's input PIT trigger 3 is fed into fast peripherials through xbar.

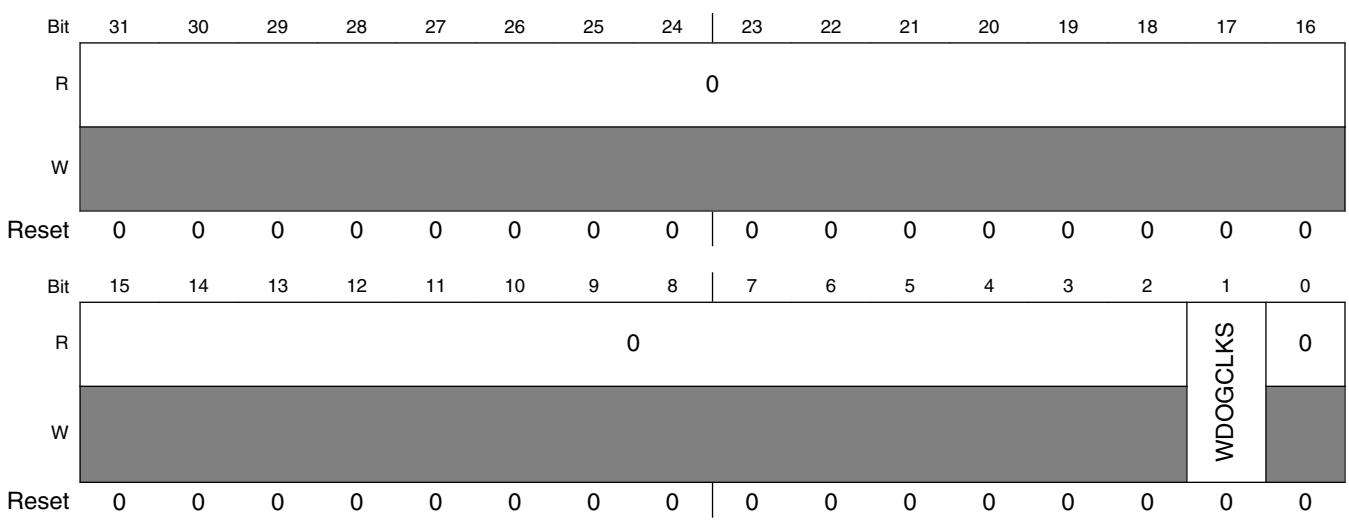
Table continues on the next page...

SIM_MISCTRL1 field descriptions (continued)

Field	Description
	0 Disable, bypass synchronizer. 1 Enable.
10 SYNCXBARAPITTRIG2	Synchronize XBARA's Input PIT Trigger 2 with fast clock This field controls the synchronizer between PIT trigger 2 and XBARA's input. **NOTE:** Set this bit if the XBARA's input PIT trigger 2 is fed into fast peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
9 SYNCXBARAPITTRIG1	Synchronize XBARA's Input PIT Trigger 1 with fast clock This field controls the synchronizer between PIT trigger 1 and XBARA's input. **NOTE:** Set this bit if the XBARA's input PIT trigger 1 is fed into fast peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
8 SYNCXBARAPITTRIG0	Synchronize XBARA's Input PIT Trigger 0 with fast clock This field controls the synchronizer between PIT trigger 0 and XBARA's input. **NOTE:** Set this bit if the XBARA's input PIT trigger 0 is fed into fast peripherials through xbar. 0 Disable, bypass synchronizer. 1 Enable.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.23 WDOG Control Register (SIM_WDOGC)

Address: 4004_7000h base + 1100h offset = 4004_8100h



SIM_WDOGC field descriptions

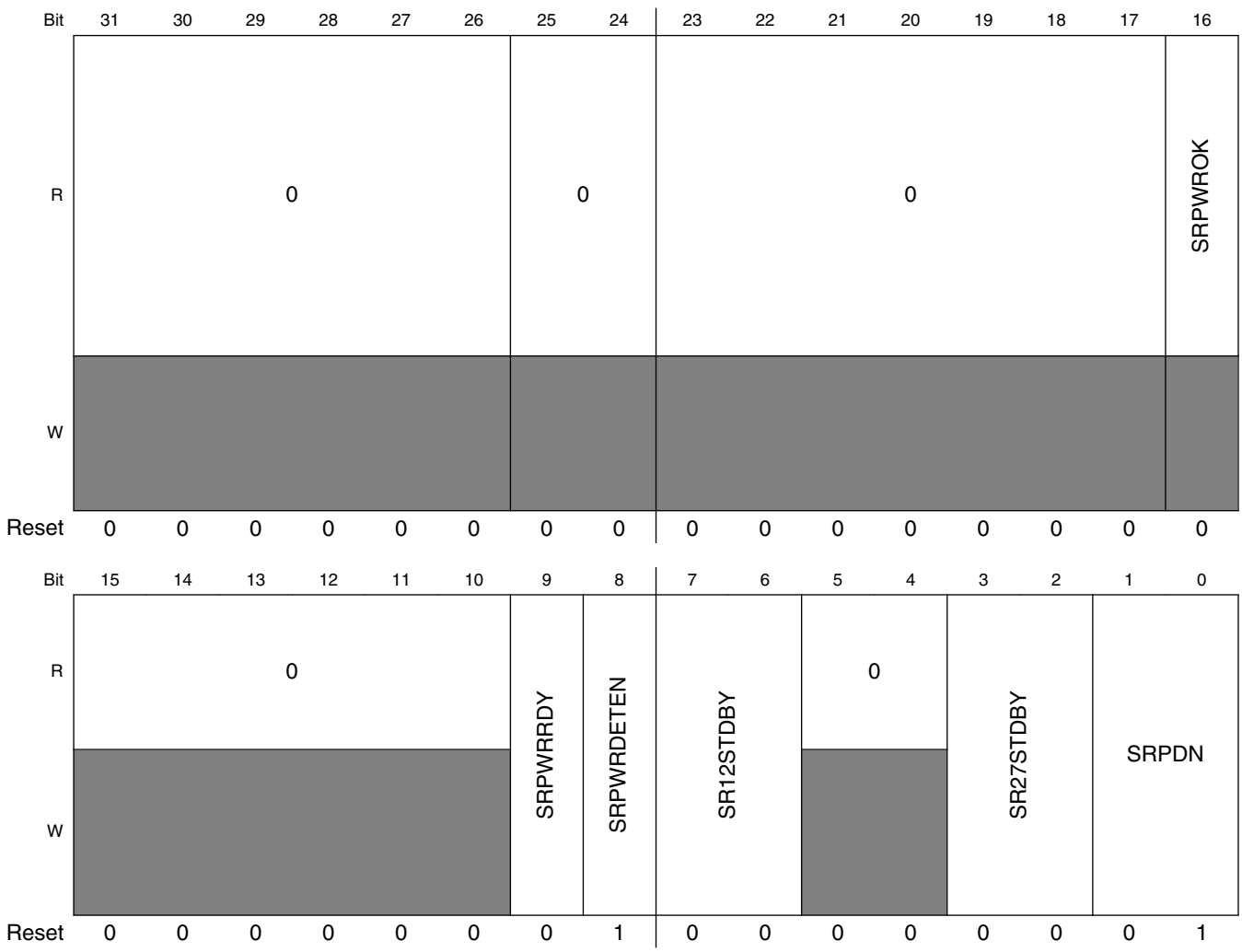
Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 WDOGCLKS	WDOG Clock Select This write-once bit selects the clock source of the WDOG2008 watchdog. **NOTE:** This is the choice of two alternative clock sources that goto the ALTCLK of the WDOG2008. 0 Internal 1 kHz clock is source to WDOG2008 1 MCGIRCLK is source to WDOG2008
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.2.24 Power Control Register (SIM_PWRC)

NOTE

Set PMC_REGSC[BGBE] before the NanoEdge regulator is enabled, because the regulator uses 1V reference of PMC.

Address: 4004_7000h base + 1104h offset = 4004_8104h



SIM_PWRC field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SIM_PWRC field descriptions (continued)

Field	Description
23–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 SRPWROK	NanoEdge PMC Status 0 Power supply for NanoEdge isn't ready. 1 Power supply for NanoEdge is OK.
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 SRPWRRDY	NanoEdge PMC POWER Ready This bit is soft control to indicate NanoEdge PMC is ready when PMC Power dectect is disabled by SRPWRDETEN. 0 Not ready 1 Assert PMC power output ready
8 SRPWRDETEN	NanoEdge PMC POWER Dectect Enable Write 1 to enable NanoEdge PMC power dectect to assert PMC ready signal when PMC is stable. 0 Disable 1 Enable
7–6 SR12STDBY	NanoEdge Regulator 1.2 V Supply Standby Control This field controls the standby mode of the 1.2 V supply from the NanoEdge voltage regulator. Standby mode has restricted drive capacity but substantially reduces power consumption. The field value can be optionally write protected. 00 NanoEdge regulator 1.2 V supply placed in normal mode 01 NanoEdge regulator 1.2 V supply placed in standby mode. 10 NanoEdge regulator 1.2 V supply placed in normal mode and SR12STDBY is write protected until chip reset. 11 NanoEdge regulator 1.2 V supply placed in standby mode and SR12STDBY is write protected until chip reset.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 SR27STDBY	NanoEdge Regulator 2.7 V Supply Standby Control This field controls the standby mode of the 2.7 V supply from the NanoEdge voltage regulator. Standby mode has restricted drive capacity but substantially reduces power consumption. The field value can be optionally write protected. 00 NanoEdge regulator 2.7 V placed in normal mode. 01 NanoEdge regulator 2.7 V placed in standby mode. 10 NanoEdge regulator 2.7 V supply placed in normal mode and SR27STDBY is write protected until chip reset. 11 NanoEdge regulator 2.7 V supply placed in standby mode and SR27STDBY is write protected until chip reset.
SRPDN	NanoEdge Regulator 2.7V and 1.2V Supply Powerdown Control This field controls the powerdown mode of the 2.7V and 1.2V supply from the NanoEdge voltage regulator. Powerdown mode shuts down the 2.7V and 1.2V regulated supply from the NanoEdge regulator

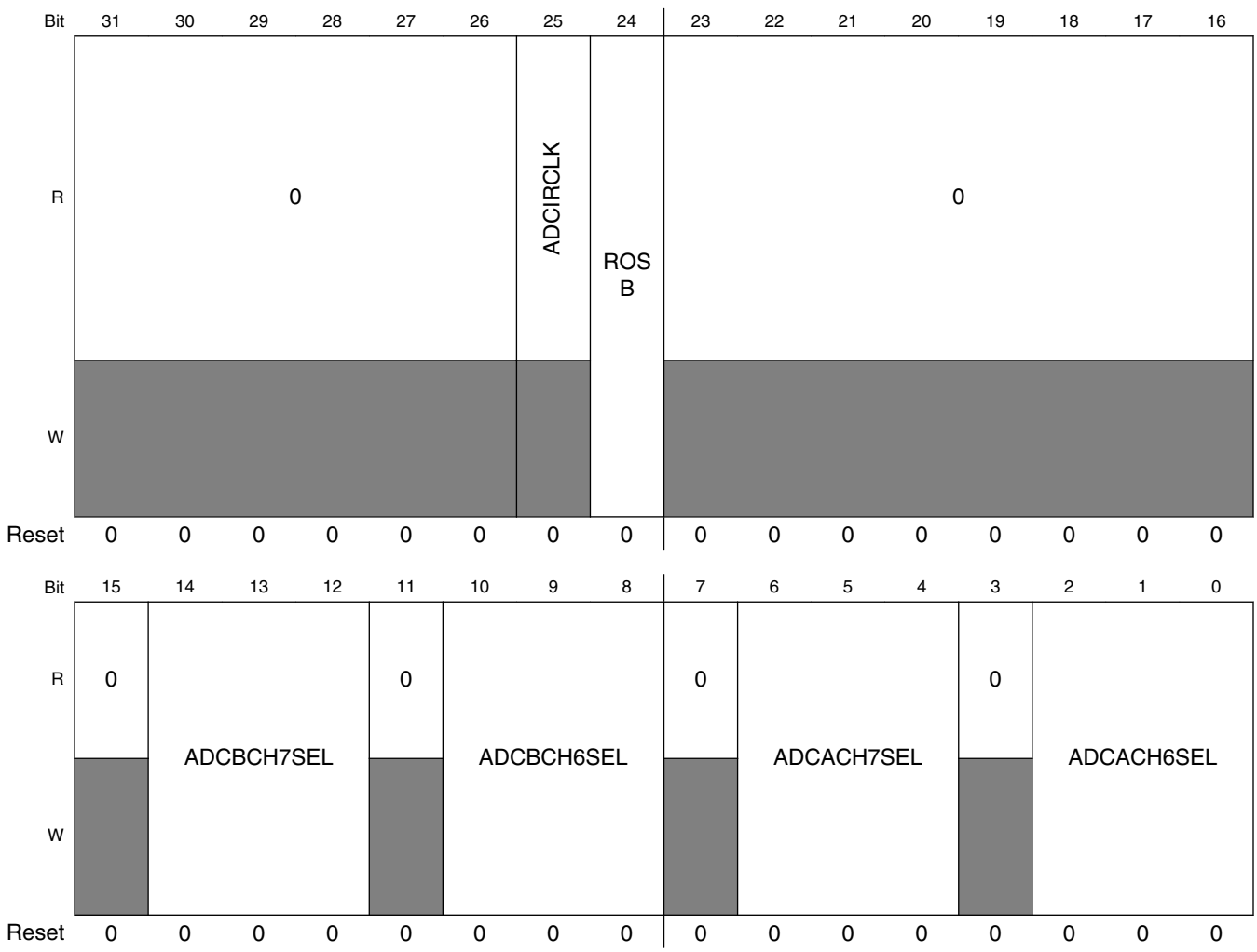
Table continues on the next page...

SIM_PWRC field descriptions (continued)

Field	Description
	and eliminates its power consumption. Analog modules powered by this supply should themselves be powered down before entering this mode. 00 NanoEdge regulator placed in normal mode. 01 NanoEdge regulator placed in powerdown mode. 10 NanoEdge regulator placed in normal mode and SRPDN is write protected until chip reset. 11 NanoEdge regulator placed in powerdown mode and SRPDN is write protected until chip reset.

13.2.25 ADC Channel 6/7 Mux Control Register (SIM_ADCOPT)

Address: 4004_7000h base + 1108h offset = 4004_8108h



SIM_ADCOPT field descriptions

Field	Description
31–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 ADCIRCLK	ADC Clock Status Indicates which clock is fed in ADC. **NOTE:** Can't access ADC's registers when this bit is "1". This bit is used in STOP/VLPS mode to make sure if the ADC clock is switched to the expected clock. 0 ADC clock is fast peripherial clock. 1 ADC clock is MCGIRCLK.
24 ROSB	Enable ADC low current Mode Control ADC low current mode in STOP and VLPS mode. 0 Disable ADC low current mode. 1 Enable ADC low current mode.
23–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–12 ADCBCH7SEL	ADCB MUX1 selection for ADCB channel 7 Selects ADCB MUX1's channel to ADCB channel 7. 000 ADCB MUX1's channel a. 001 ADCB MUX1's channel b. 010 ADCB MUX1's channel c. 011 ADCB MUX1's channel d. 100 ADCB MUX1's channel e. 101 ADCB MUX1's channel f. 110 ADCB MUX1's channel g. 111 Reserved
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 ADCBCH6SEL	ADCB MUX1 selection for ADCB channel 6 Selects ADCB MUX0's channel to ADCB channel 6. 000 ADCB MUX0's channel a. 001 ADCB MUX0's channel b. 010 ADCB MUX0's channel c. 011 ADCB MUX0's channel d. 100 ADCB MUX0's channel e. 101 ADCB MUX0's channel f. 110 ADCB MUX0's channel g. 111 PMC 1V.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 ADCACH7SEL	ADCA MUX1 selection for ADCA channel 7 Selects ADCA MUX1's channel to ADCA channel 7.

Table continues on the next page...

SIM_ADCOPT field descriptions (continued)

Field	Description
	000 ADCA MUX1's channel a. 001 ADCA MUX1's channel b. 010 ADCA MUX1's channel c. 011 Reserved 100 ADCA MUX1's channel e. 101 ADCA MUX1's channel f. 110 ADCA MUX1's channel g. 111 PMC 1V
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ADCACH6SEL	ADCA MUX0 selection for ADCA channel 6 Selects ADCA MUX0's channel to ADCA channel 6. 000 ADCA MUX0's channel a. 001 ADCA MUX0's channel b. 010 ADCA MUX0's channel c. 011 ADCA MUX0's channel d. 100 ADCA MUX0's channel e. 101 Reserved 110 ADCA MUX0's channel g. 111 Reserved

13.3 Functional description

For more information about the functions of SIM, see the Introduction section.

Chapter 14
Kinetis Flashloader

14.1 Chip-specific Kinetis Flashloader information

14.1.1 Kinetis Flashloader peripheral pinmux

This device has various peripherals (UART, I2C, CAN, SPI) supported by the Kinetis Flashloader. The next table shows the pads used by the Kinetis Flashloader.

Table 14-1. Kinetis Flashloader Peripheral Pinmux

Peripheral	Instance	Port	Signal	Alt
UART	0	PTD6	UART0_RX	ALT3
		PTD7	UART0_TX	ALT3
I2C	0	PTB0	I2C0_SCL	ALT2
		PTB1	I2C0_SDA	ALT2
SPI	0	PTE16	SPI0_PCS0	ALT2
		PTE17	SPI0_SCK	ALT2
		PTE18	SPI0_SOUT	ALT2
		PTE19	SPI0_SIN	ALT2
CAN	0	PTB16	CAN0_TX	ALT5
		PTB17	CAN0_RX	ALT5

14.2 Introduction

The Kinetis devices *that do not have an on-chip ROM* are shipped with the pre-programmed Kinetis Flashloader in the on-chip flash memory, for one-time, in-system factory programming. The Kinetis Flashloader's main task is to load a customer firmware image into the flash memory. The image on the flash has 2 programs: flashloader_loader and flashloader. After a device reset, the flashloader_loader program starts its execution

first. The flashloader_loader program copies the contents of flashloader image from the flash to the on-chip RAM; the device then switches execution to the flashloader program to execute from RAM.

For this device, the Kinetis Flashloader can interface with UART, CAN, I2C, and SPI peripherals in slave mode and respond to the commands sent by a master (or host) communicating on one of those ports. The host/master can be a firmware-download application running on a PC or an embedded host communicating with the Kinetis Flashloader. Regardless of the host/master (PC or embedded host), the Kinetis Flashloader always uses a command protocol to communicate with that host/master. Commands are provided to write to memory (flash or RAM), erase flash, and get/set flashloader options and property values. The host application can query the set of available commands.

This chapter describes Kinetis Flashloader features, functionality, command structure and which peripherals are supported.

Features supported by the Kinetis Flashloader :

- Supports UART, CAN, I2C, and SPI peripheral interfaces
- Automatic detection of the active peripheral
- UARTand CAN peripherals with autobaud
- Common packet-based protocol for all peripherals
- Packet error detection and retransmission
- Protection of RAM used by the flashloader while it is running
- Provides command to read properties of the device, such as flash and RAM size

Table 14-2. Commands supported by the Kinetis Flashloader

Command	Description	When flash security is enabled, then this command is
Execute	Run user application code that never returns control to the flashloader	Not supported
FillMemory	Fill a range of bytes in flash with a word pattern	Not supported
FlashEraseAll	Erase the entire flash array	Not supported
FlashEraseRegion	Erase a range of sectors in flash	Not supported
WriteMemory	Write data to memory	Not supported
ReadMemory	Read data from memory	Not supported
GetProperty	Get the current value of a property	Supported
Reset	Reset the chip	Supported
SetProperty	Attempt to modify a writable property	Supported

14.3 Functional Description

The following sub-sections describe the Kinetis Flashloader functionality.

14.3.1 Memory Maps

While executing, the Kinetis Flashloader uses RAM memory.

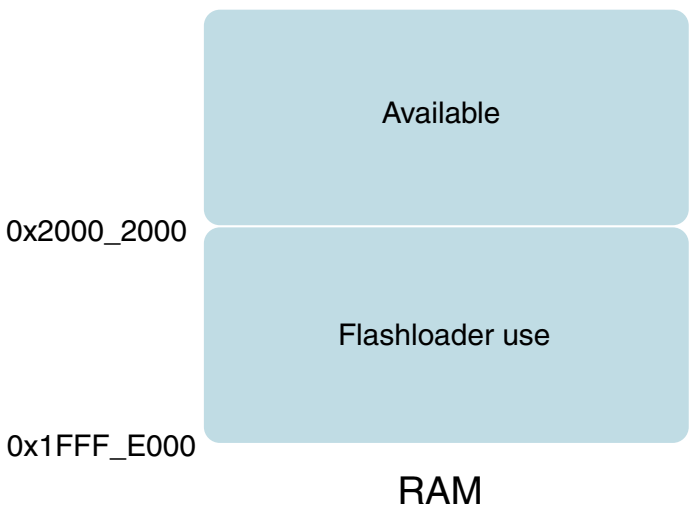


Figure 14-1. Kinetis Flashloader RAM Memory Map

NOTE

The Kinetis Flashloader requires a minimum memory space of 32 KB of RAM. For Kinetis devices with less than this amount of on-chip RAM, the Kinetis Flashloader is not available.

14.3.2 Start-up Process

As the Kinetis Flashloader begins executing, flashloader operations begin:

1. The flashloader's temporary working area in RAM is initialized.
2. All supported peripherals are initialized.
3. The flashloader waits for communication to begin on a peripheral.

- There is no timeout for the active peripheral detection process.
- If communication is detected, then all inactive peripherals are shut down, and the command phase is entered.

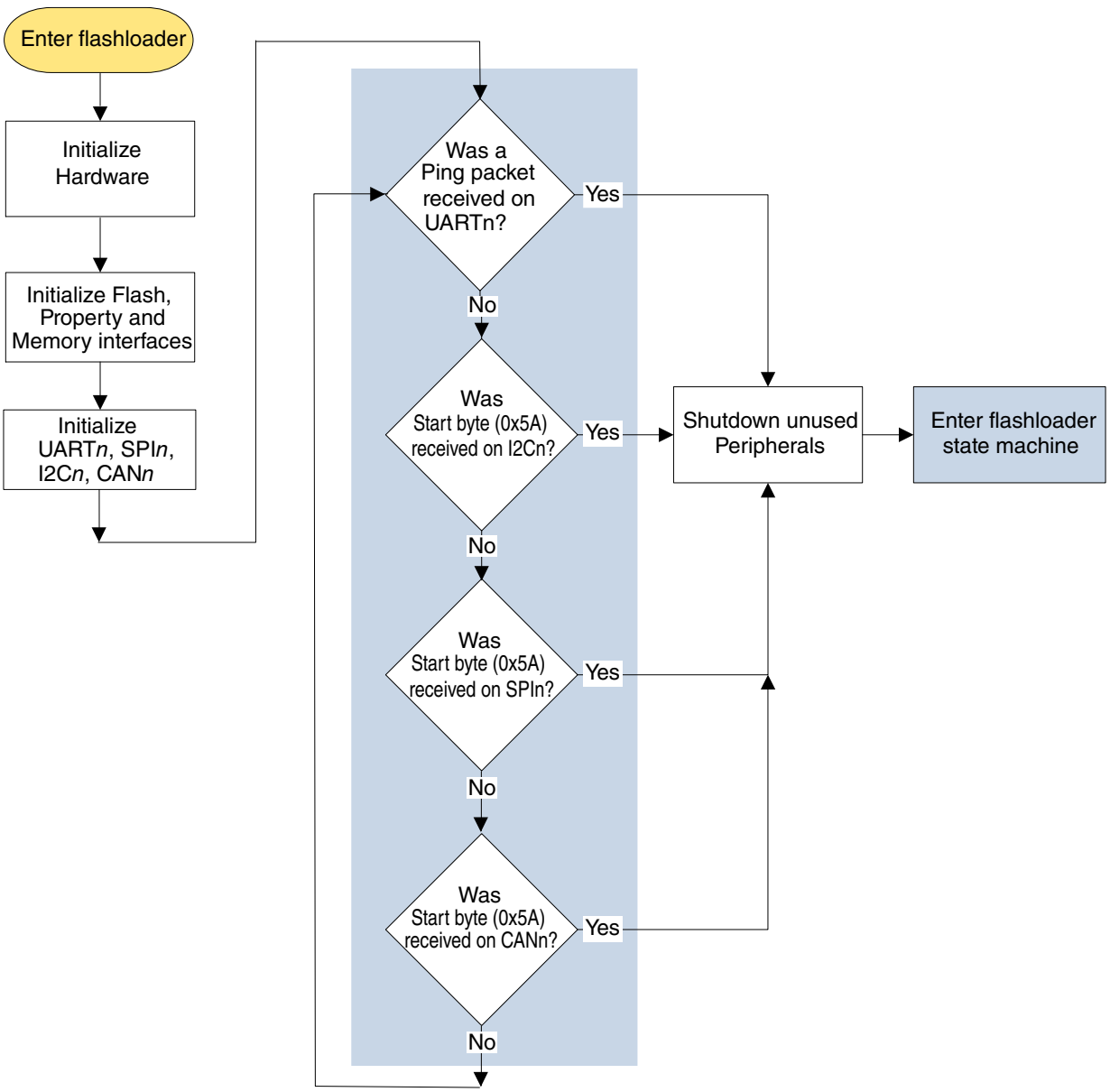


Figure 14-2. Kinetis Flashloader Start-up Flowchart

14.3.3 Clock Configuration

The core runs on the default reset clock (20.9 MHz). The Kinetis Flashloader does not modify any clocks.

14.3.4 Flashloader Protocol

This section explains the general protocol for the packet transfers between the host and the Kinetis Flashloader. The description includes the transfer of packets for different transactions, such as commands with no data phase and commands with incoming or outgoing data phase. The next section describes various packet types used in a transaction.

Each command sent from the host is replied to with a response command.

Commands may include an optional data phase:

- If the data phase is **incoming** (from host to flashloader), then the data phase is part of the **original command**.
- If the data phase is **outgoing** (from flashloader to host), then the data phase is part of the **response command**.

NOTE

In all protocols (described in the next subsections), the Ack sent in response to a Command or Data packet can arrive at any time *before, during, or after* the Command/Data packet has processed.

14.3.4.1 Command with no data phase

The protocol for a command with no data phase contains:

- Command packet (from host)
- Generic response command packet (to host)

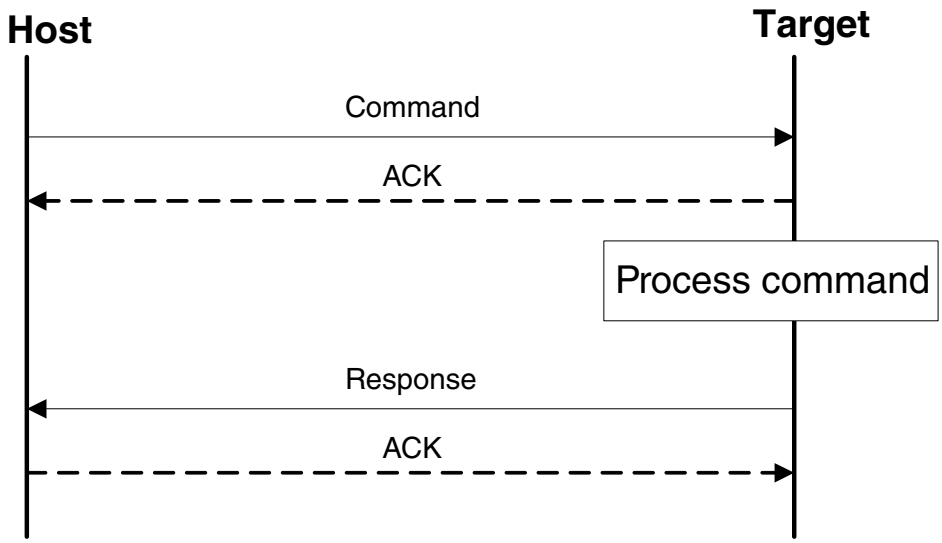


Figure 14-3. Command with No Data Phase

14.3.4.2 Command with incoming data phase

The protocol for a command with an incoming data phase contains:

- Command packet (from host)
- Generic response command packet (to host)
- Incoming data packets (from host)
- Generic response command packet (to host)

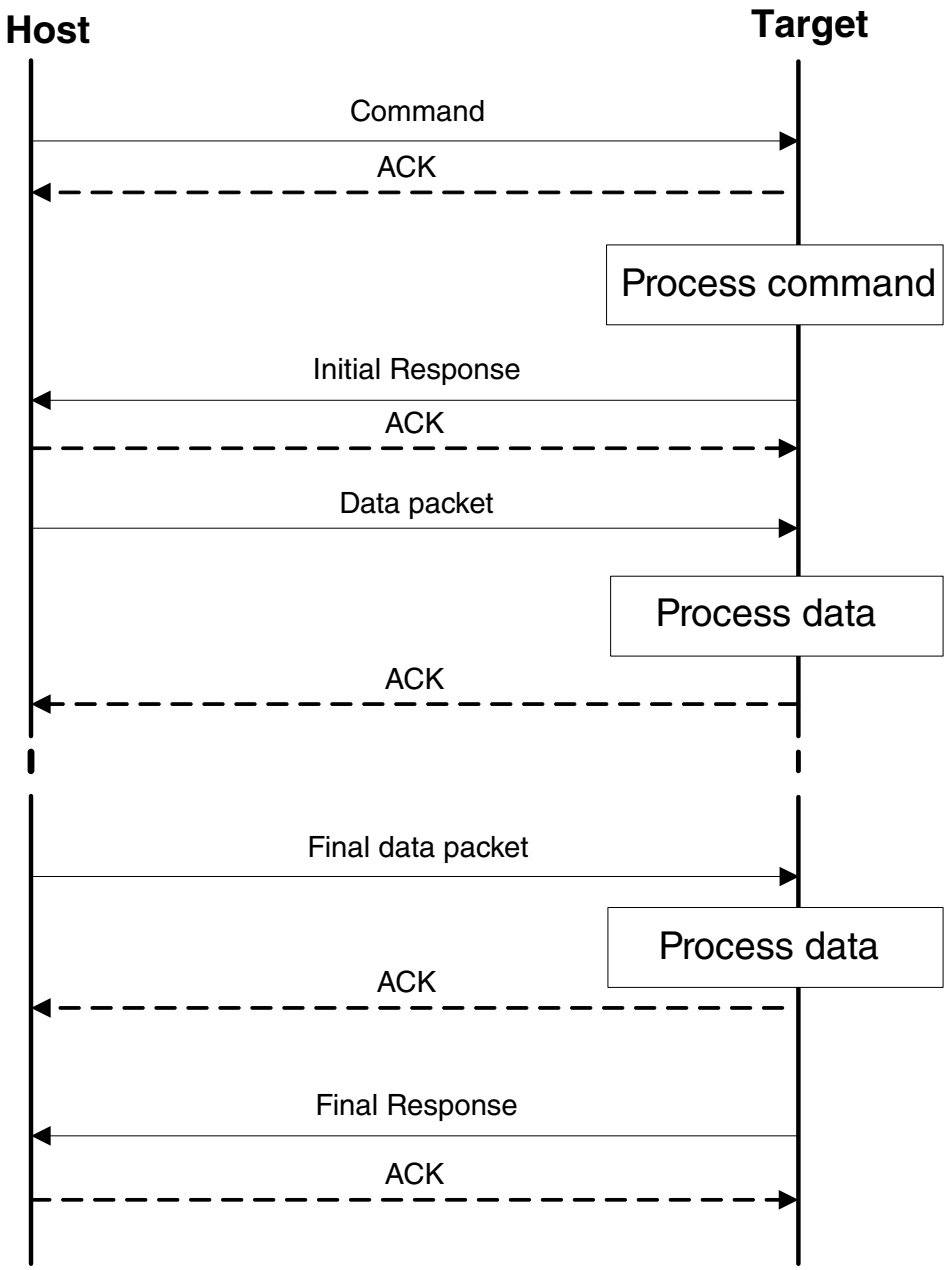


Figure 14-4. Command with incoming data phase

NOTE

- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the Generic Response packet prior to the start of the data phase does not have a status of kStatus_Success, then the data phase is aborted.
- Data phases may be aborted by the receiving side by sending the final Generic Response early with a status of

kStatus_AbortDataPhase. The host may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

14.3.4.3 Command with outgoing data phase

The protocol for a command with an outgoing data phase contains:
- Command packet (from host)
- ReadMemory Response command packet (to host) (kCommandFlag_HasDataPhase set)
- Outgoing data packets (to host)
- Generic response command packet (to host)

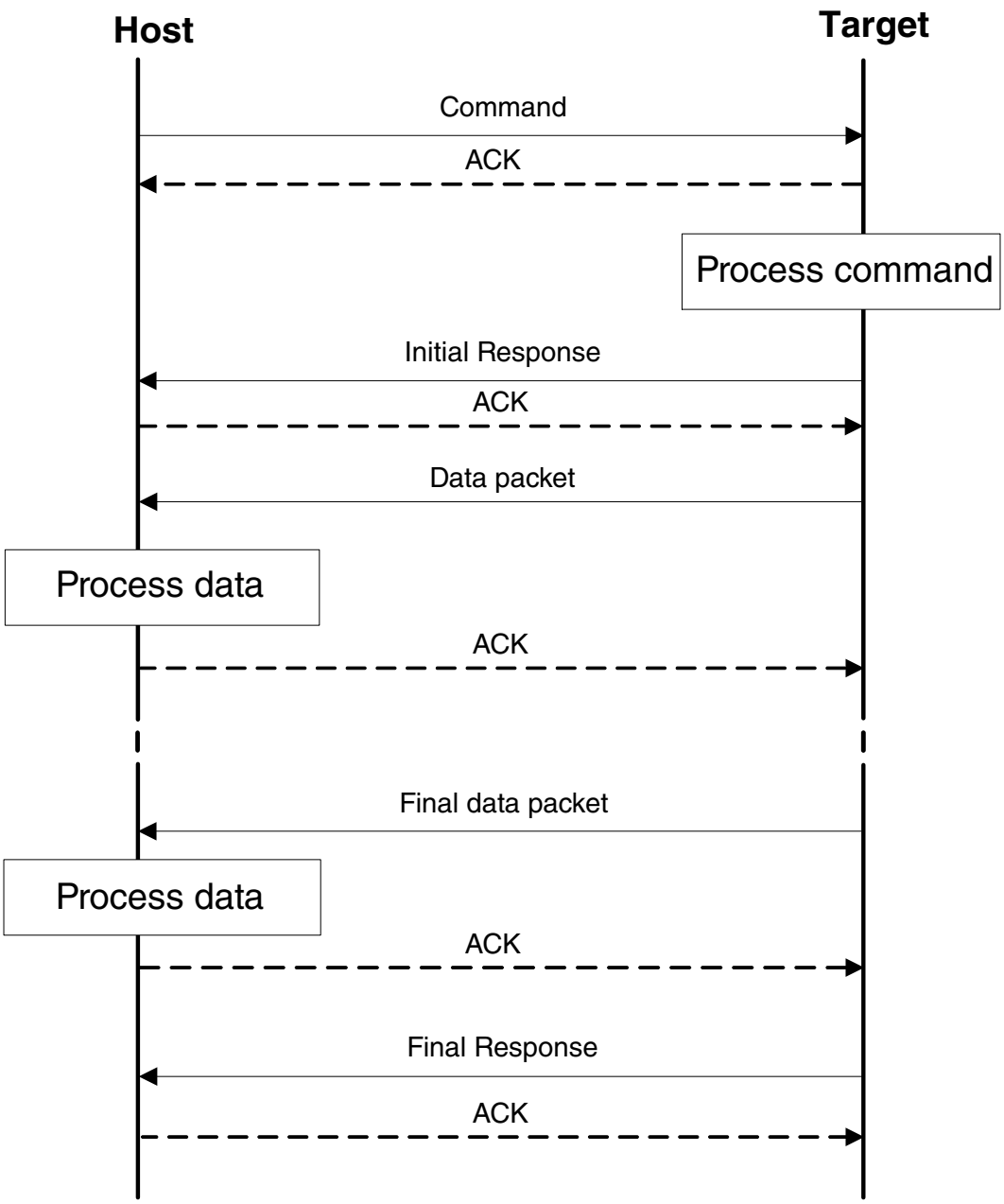


Figure 14-5. Command with outgoing data phase

NOTE

- For the outgoing data phase sequence above, the data phase is really considered part of the response command.
- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the ReadMemory Response command packet prior to the start of the data phase does not contain the kCommandFlag_HasDataPhase flag, then the data phase is aborted.

- Data phases may be aborted by the host sending the final Generic Response early with a status of kStatus_AbortDataPhase. The sending side may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

14.3.5 Flashloader Packet Types

The Kinetis Flashloader device works in slave mode. All data communication is initiated by a host, which is either a PC or an embedded host . The Kinetis Flashloader device is the target, which receives a command or data packet. All data communication between host and target is packetized.

NOTE

The term "target" refers to the "Kinetis Flashloader device."

There are 6 types of packets used in the device:

- Ping packet
- Ping Response packet
- Framing packet
- Command packet
- Data packet
- Response packet

All fields in the packets are in little-endian byte order.

14.3.5.1 Ping packet

The Ping packet is the first packet sent from a host to the target (Kinetis Flashloader), to establish a connection on a selected peripheral. For a UART peripheral, the Ping packet is used to determine the baudrate. A Ping packet must be sent before any other communications. In response to a Ping packet, the target sends a Ping Response packet.

Table 14-3. Ping Packet Format

Byte #	Value	Name
0	0x5A	start byte
1	0xA6	ping

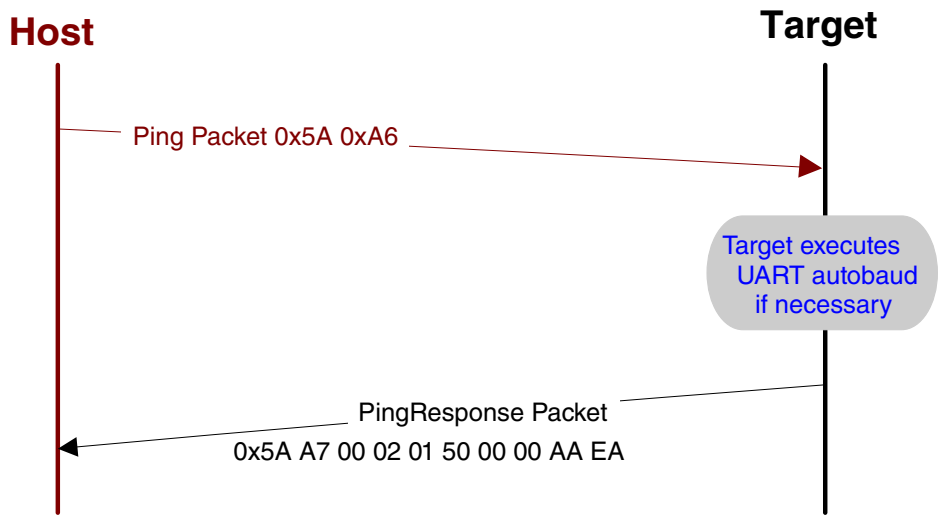


Figure 14-6. Ping Packet Protocol Sequence

14.3.5.2 Ping Response Packet

The target (Kinetis Flashloader) sends a Ping Response packet back to the host after receiving a Ping packet. If communication is over a UART peripheral, the target uses the incoming Ping packet to determine the baud rate before replying with the Ping Response packet. Once the Ping Response packet is received by the host, the connection is established, and the host starts sending commands to the target (Kinetis Flashloader).

Table 14-4. Ping Response Packet Format

Byte #	Value	Parameter
0	0x5A	start byte
1	0xA7	Ping response code
2		Protocol bugfix
3		Protocol minor
4		Protocol major
5		Protocol name = 'P' (0x50)
6		Options low
7		Options high
8		CRC16 low
9		CRC16 high

14.3.5.3 Framing Packet

The framing packet is used for flow control and error detection, and it (the framing packet) wraps command and data packets as well.

The framing packet described in this section is used for serial peripherals including UART, I2C, SPI, and CAN.

Table 14-5. Framing Packet Format

Byte #	Value	Parameter	
0	0x5A	start byte	
1		packetType	
2		length_low	Length is a 16-bit field that specifies the entire command or data packet size in bytes.
3		length_high	
4		crc16_low	This is a 16-bit field. The CRC16 value covers entire framing packet, including the start byte and command or data packets, but does not include the CRC bytes. See the CRC16 algorithm after this table.
5		crc16_high	
6 . . .n		Command or Data packet payload	

A special framing packet that contains only a start byte and a packet type is used for synchronization between the host and target.

Table 14-6. Special Framing Packet Format

Byte #	Value	Parameter
0	0x5A	start byte
1	0xA*n*	packetType

The Packet Type field specifies the type of the packet from one of the defined types (below):

Table 14-7. packetType Field

packetType	Name	Description
0xA1	kFramingPacketType_Ack	The previous packet was received successfully; the sending of more packets is allowed.
0xA2	kFramingPacketType_Nak	The previous packet was corrupted and must be re-sent.
0xA3	kFramingPacketType_AckAbort	Data phase is being aborted.
0xA4	kFramingPacketType_Command	The framing packet contains a command packet payload.
0xA5	kFramingPacketType_Data	The framing packet contains a data packet payload.
0xA6	kFramingPacketType_Ping	Sent to verify the other side is alive. Also used for UART autobaud.

Table continues on the next page...

Table 14-7. packetType Field (continued)

packetType	Name	Description
0xA7	kFramingPacketType_PingResponse	A response to Ping; contains the framing protocol version number and options.

CRC16 algorithm:

```
uint16_t crc16_update(const uint8_t * src, uint32_t lengthInBytes)
{
    uint32_t crc = 0;
    uint32_t j;
    for (j=0; j < lengthInBytes; ++j)
    {
        uint32_t i;
        uint32_t byte = src[j];
        crc ^= byte << 8;
        for (i = 0; i < 8; ++i)
        {
            uint32_t temp = crc << 1;
            if (crc & 0x8000)
            {
                temp ^= 0x1021;
            }
            crc = temp;
        }
    }
    return crc;
}
```

14.3.5.4 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 14-8. Command Packet Format

Command Packet Format (32 bytes)										
Command Header (4 bytes)				28 bytes for Parameters (Max 7 parameters)						
Tag	Flags	Rsvd	Param Count	Param1 (32-bit)	Param2 (32-bit)	Param3 (32-bit)	Param4 (32-bit)	Param5 (32-bit)	Param6 (32-bit)	Param7 (32-bit)
byte 0	byte 1	byte 2	byte 3							

Table 14-9. Command Header Format

Byte #	Command Header Field	
0	Command or Response tag	The command header is 4 bytes long, with these fields.
1	Flags	
2	Reserved. Should be 0x00.	
3	ParameterCount	

The header is followed by 32-bit parameters up to the value of the ParameterCount field specified in the header. Because a command packet is 32 bytes long, only 7 parameters can fit into the command packet.

Command packets are also used by the target to send responses back to the host. As mentioned earlier, command packets and data packets are embedded into framing packets for all of the transfers.

Table 14-10. Commands that are supported

Command	Name
0x01	FlashEraseAll
0x02	FlashEraseRegion
0x03	ReadMemory
0x04	WriteMemory
0x05	FillMemory
0x06	FlashSecurityDisable
0x07	GetProperty
0x08	Reserved
0x09	Execute
0x0A	Reserved
0x0B	Reset
0x0C	SetProperty
0x0D	Reserved

Table continues on the next page...

Table 14-10. Commands that are supported (continued)

Command	Name
0x0E	Reserved
0x0F	Reserved
0x10	Reserved
0x11	Reserved
0x12	Reserved

Table 14-11. Responses that are supported

Response	Name
0xA0	GenericResponse
0xA3	ReadMemoryResponse (used for sending responses to ReadMemory command only)
0xA7	GetPropertyResponse (used for sending responses to GetProperty command only)

Flags: Each command packet contains a Flag byte. Only bit 0 of the flag byte is used. If bit 0 of the flag byte is set to 1, then data packets will follow in the command sequence. The number of bytes that will be transferred in the data phase is determined by a command-specific parameter in the parameters array.

ParameterCount: The number of parameters included in the command packet.

Parameters: The parameters are word-length (32 bits). With the default maximum packet size of 32 bytes, a command packet can contain up to 7 parameters.

14.3.5.5 Data packet

The data packet carries just the data, either host sending data to target, or target sending data to host. The data transfer direction is determined by the last command sent from the host. The data packet is also wrapped within a framing packet, to ensure the correct packet data is received.

The contents of a data packet are simply the data itself. There are no other fields, so that the most data per packet can be transferred. Framing packets are responsible for ensuring that the correct packet data is received.

14.3.5.6 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. Types of responses include:

- GenericResponse
- GetPropertyResponse
- ReadMemoryResponse

GenericResponse: After the Kinetis Flashloader has processed a command, the flashloader will send a generic response with status and command tag information to the host. The generic response is the last packet in the command protocol sequence. The generic response packet contains the framing packet data and the command packet data (with generic response tag = 0xA0) and a list of parameters (defined in the next section). The parameter count field in the header is always set to 2, for status code and command tag parameters.

Table 14-12. GenericResponse Parameters

Byte #	Parameter	Descripton
0 - 3	Status code	The Status codes are errors encountered during the execution of a command by the target (Kinetis Flashloader). If a command succeeds, then a kStatus_Success code is returned. Table 14-38, Kinetis Flashloader Status Error Codes, lists the status codes returned to the host by the Kinetis Flashloader.
4 - 7	Command tag	The Command tag parameter identifies the response to the command sent by the host.

GetPropertyResponse: The GetPropertyResponse packet is sent by the target in response to the host query that uses the GetProperty command. The GetPropertyResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a GetPropertyResponse tag value (0xA7).

The parameter count field in the header is set to greater than 1, to always include the status code and one or many property values.

Table 14-13. GetPropertyResponse Parameters

Byte #	Value	Parameter
0 - 3		Status code
4 - 7		Property value
. . .		. . .
		Can be up to maximum 6 property values, limited to the size of the 32-bit command packet and property type.

ReadMemoryResponse: The ReadMemoryResponse packet is sent by the target in response to the host sending a ReadMemory command. The ReadMemoryResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a ReadMemoryResponse tag value (0xA3), the flags field set to kCommandFlag_HasDataPhase (1).

The parameter count set to 2 for the status code and the data byte count parameters shown below.

Table 14-14. ReadMemoryResponse Parameters

Byte #	Parameter	Descripton
0 - 3	Status code	The status of the associated Read Memory command.
4 - 7	Data byte count	The number of bytes sent in the data phase.

14.3.6 Flashloader Command API

All Kinetis Flashloader command APIs follow the command packet format that is wrapped by the framing packet, as explained in previous sections.

- For a list of commands supported by the Flashloader, see Table 14-2, Commands supported.
- For a list of status codes returned by the Kinetis Flashloader, see Table 14-38, Kinetis Flashloader Status Error Codes.

NOTE

All the examples in this section depict byte traffic on serial peripherals that use framing packets.

14.3.6.1 GetProperty command

The GetProperty command is used to query the flashloader about various properties and settings. Each supported property has a unique 32-bit tag associated with it. The tag occupies the first parameter of the command packet. The target returns a GetPropertyResponse packet with the property values for the property identified with the tag in the GetProperty command.

Properties are the defined units of data that can be accessed with the GetProperty or SetProperty commands. Properties may be read-only or read-write. All read-write properties are 32-bit integers, so they can easily be carried in a command parameter.

For a list of properties and their associated 32-bit property tags supported by the Kinetis Flashloader, see Table 14-34.

The 32-bit property tag is the only parameter required for GetProperty command.

Table 14-15. Parameters for GetProperty Command

Byte #	Command
0 - 3	Property tag

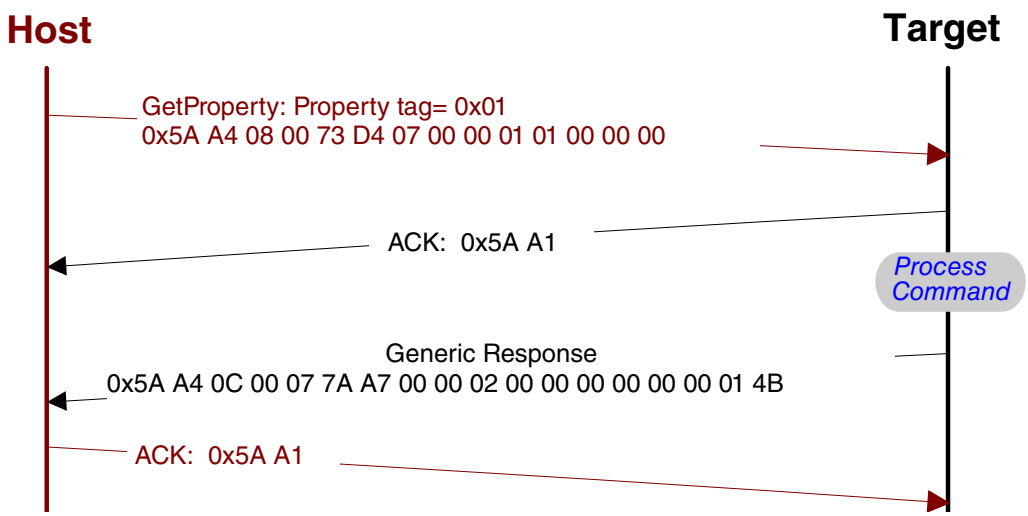


Figure 14-7. Protocol Sequence for GetProperty Command

Table 14-16. GetProperty Command Packet Format (Example)

GetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x08 0x00
	crc16	0x73 0xD4
Command packet	commandTag	0x07 – GetProperty
	flags	0x00
	reserved	0x00
	parameterCount	0x01
	propertyTag	0x00000001 - CurrentVersion

The GetProperty command has no data phase.

Response: In response to a GetProperty command, the target will send a GetPropertyResponse packet with the response tag set to 0xA7. The parameter count indicates the number of parameters sent for the property values, with the first parameter showing status code 0, followed by the property value(s). The next table shows an example of a GetPropertyResponse packet.

Table 14-17. GetProperty Response Packet Format (Example)

GetPropertyResponse	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0c 0x00 (12 bytes)
	crc16	0x07 0x7a
Command packet	responseTag	0xA7
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	status	0x00000000
	propertyValue	0x0000014b - CurrentVersion

14.3.6.2 SetProperty command

The SetProperty command is used to change or alter the values of the properties or options in the Kinetis Flashloader. However, the SetProperty command can only change the value of properties that are writable—see Table 14-34, Properties used by Get/SetProperty Commands. If you try to set a value for a read-only property, then the Kinetis Flashloader will return an error.

The property tag and the new value to set are the 2 parameters required for the SetProperty command.

Table 14-18. Parameters for SetProperty Command

Byte #	Command
0 - 3	Property tag
4 - 7	Property value

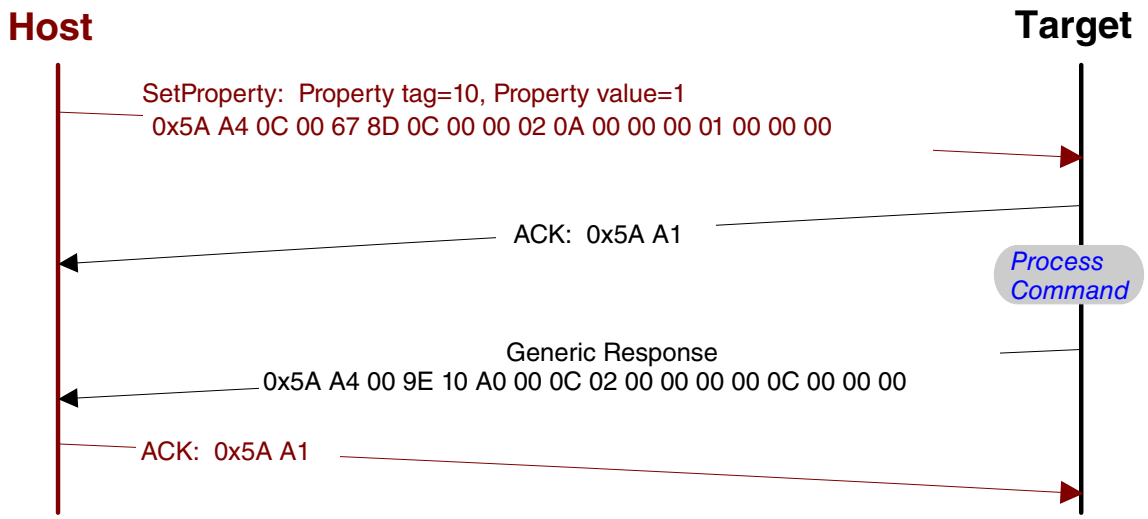


Figure 14-8. Protocol Sequence for SetProperty Command

Table 14-19. SetProperty Command Packet Format (Example)

SetProperty	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x67 0x8D
Command packet	commandTag	0x0C – SetProperty with property tag 10
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	propertyTag	0x0000000A - VerifyWrites
	propertyValue	0x00000001

The SetProperty command has no data phase.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with one of following status codes:

Table 14-20. SetProperty Response Status Codes

Status Code
kStatus_Success
kStatus_ReadOnly
kStatus_UnknownProperty
kStatus_InvalidArgument

14.3.6.3 FlashEraseAll command

The FlashEraseAll command performs an erase of the entire flash memory. If any flash regions are protected, then the FlashEraseAll command will fail and return an error status code. Executing the FlashEraseAll command will release flash security if it (flash security) was enabled, by setting the FTFA_FSEC register. However, the FSEC field of the flash configuration field is erased, so unless it is reprogrammed, the flash security will be re-enabled after the next system reset. The Command tag for FlashEraseAll command is 0x01 set in the commandTag field of the command packet.

The FlashEraseAll command requires no parameters.

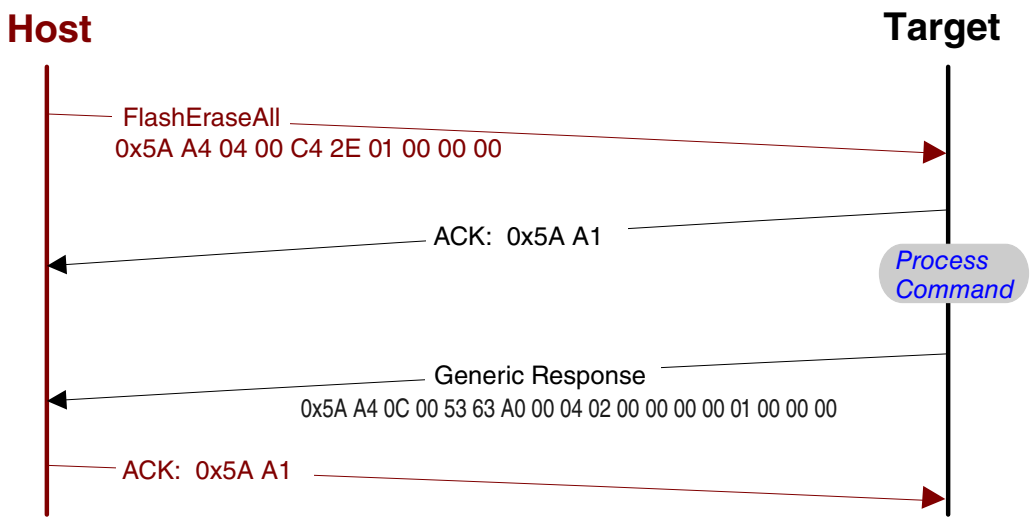


Figure 14-9. Protocol Sequence for FlashEraseAll Command

Table 14-21. FlashEraseAll Command Packet Format (Example)

FlashEraseAll	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0xC4 0x2E
Command packet	commandTag	0x01 - FlashEraseAll
	flags	0x00
	reserved	0x00
	parameterCount	0x00
	MemoryID	• If MemoryID = 0x00h, then internal flash. • If MemoryID = 0x01h, then QSPI0 memory.

The FlashEraseAll command has no data phase.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

14.3.6.4 FlashEraseRegion command

The FlashEraseRegion command performs an erase of one or more sectors of the flash memory or a specified range of flash within the connected SPI flash devices.

The start address and number of bytes are the 2 parameters required for the FlashEraseRegion command. The start and byte count parameters must be 16-byte aligned, or the FlashEraseRegion command will fail and return kStatus_FlashAlignmentError (0x101). If the region specified does not fit in the flash memory space, the FlashEraseRegion command will fail and return kStatus_FlashAddressError (0x102). If any part of the region specified is protected, the FlashEraseRegion command will fail and return kStatus_MemoryRangeInvalid (0x10200).

Table 14-22. Parameters for FlashEraseRegion Command

Byte #	Parameter
0 - 3	Start address
4 - 7	Byte count

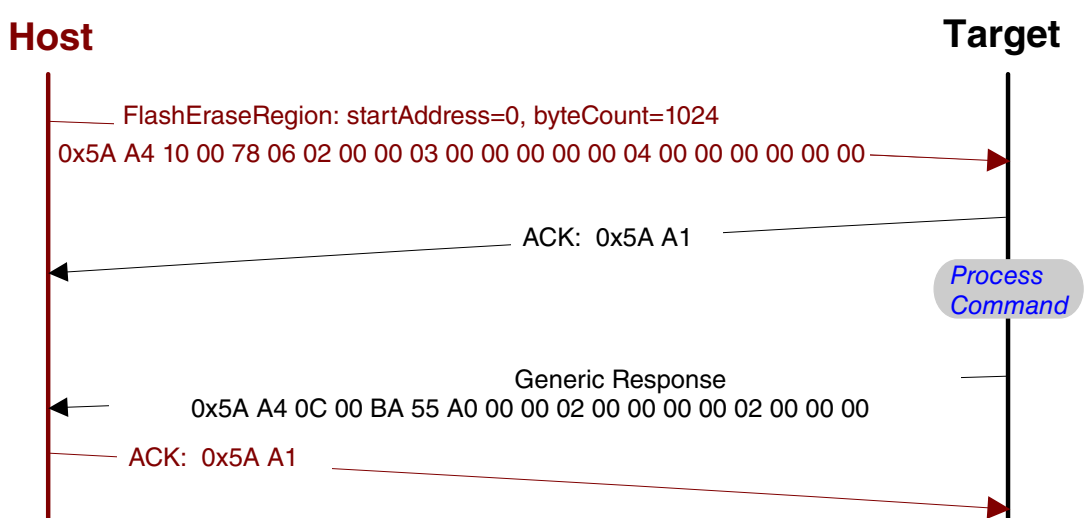


Figure 14-10. Protocol Sequence for FlashEraseRegion Command

Table 14-23. FlashEraseRegion Command Packet Format (Example)

FlashEraseRegion	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0x78 0x06
Command packet	commandTag	0x02, kCommandTag_FlashEraseRegion
	flags	0x00
	reserved	0x00
	parameterCount	0x03
	startAddress	0x00 0x00 0x00 0x00 (0x0000_0000)
	byte count	0x00 0x04 0x00 0x00 (0x400)
	memory_id	0x00 0x00 0x00 0x00 (internal flash)

The FlashEraseRegion command has no data phase.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with one of following error status codes.

Table 14-24. FlashEraseRegion Response Status Codes

Status Code
kStatus_Success (0x0)
kStatus_MemoryRangeInvalid (0x10200)
kStatus_FlashAlignmentError (0x101)
kStatus_FlashAddressError (0x102)
kStatus_FlashAccessError (0x103)
kStatus_FlashProtectionViolation (0x104)
kStatus_FlashCommandFailure (0x105)

14.3.6.5 FillMemory command

The FillMemory command fills a range of bytes in memory with a data pattern. It follows the same rules as the WriteMemory command. The difference between FillMemory and WriteMemory is that a data pattern is included in FillMemory command parameter, and there is no data phase for the FillMemory command, while WriteMemory does have a data phase.

Table 14-25. Parameters for FillMemory Command

Byte #	Command
0 - 3	Start address of memory to fill
4 - 7	Number of bytes to write with the pattern • The start address should be 32-bit aligned. • The number of bytes must be evenly divisible by 4.
8 - 11	32-bit pattern

- To fill with a byte pattern (8-bit), the byte must be replicated 4 times in the 32-bit pattern.
- To fill with a short pattern (16-bit), the short value must be replicated 2 times in the 32-bit pattern.

For example, to fill a byte value with 0xFE, the word pattern would be 0xFEFEFEFE; to fill a short value 0x5AFE, the word pattern would be 0x5AFE5AFE.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll or FlashEraseRegion command.
- Writing to flash requires the start address to be 32-bit aligned.
- If the VerifyWrites property is set to true, then writes to flash will also perform a flash verify program operation.

When writing to RAM, the start address need not be aligned, and the data will not be padded.

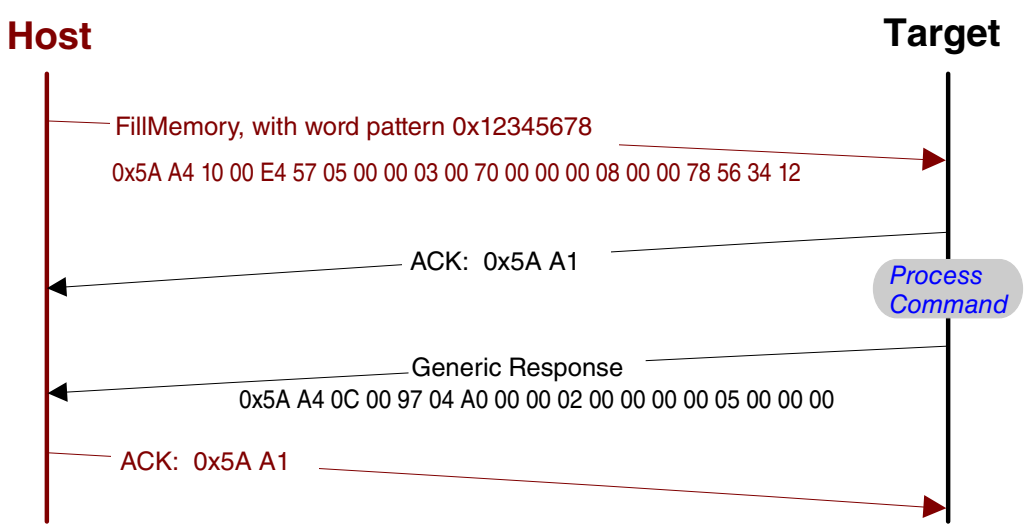


Figure 14-11. Protocol Sequence for FillMemory Command

Table 14-26. FillMemory Command Packet Format (Example)

FillMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x10 0x00
	crc16	0xE4 0x57
Command packet	commandTag	0x05 – FillMemory
	flags	0x00
	Reserved	0x00
	parameterCount	0x03
	startAddress	0x00007000
	byteCount	0x00000800
	patternWord	0x12345678

The FillMemory command has no data phase.

Response: upon successful execution of the command, the target (Kinetis Flashloader) will return a GenericResponse packet with a status code set to kStatus_Success, or to an appropriate error status code.

14.3.6.6 WriteMemory command

The WriteMemory command writes data provided in the data phase to a specified range of bytes in memory (flash or RAM). However, if flash protection is enabled, then writes to protected sectors will fail.

Special care must be taken when writing to flash.

- First, any flash sector written to must have been previously erased with a FlashEraseAll or FlashEraseRegion command.
- Writing to flash requires the start address to be .
- If the VerifyWrites property is set to true, then writes to flash will also perform a flash verify program operation.

When writing to RAM, the start address need not be aligned, and the data will not be padded.

The start address and number of bytes are the 2 parameters required for WriteMemory command.

Table 14-27. Parameters for WriteMemory Command

Byte #	Command
0 - 3	Start address
4 - 7	Byte count

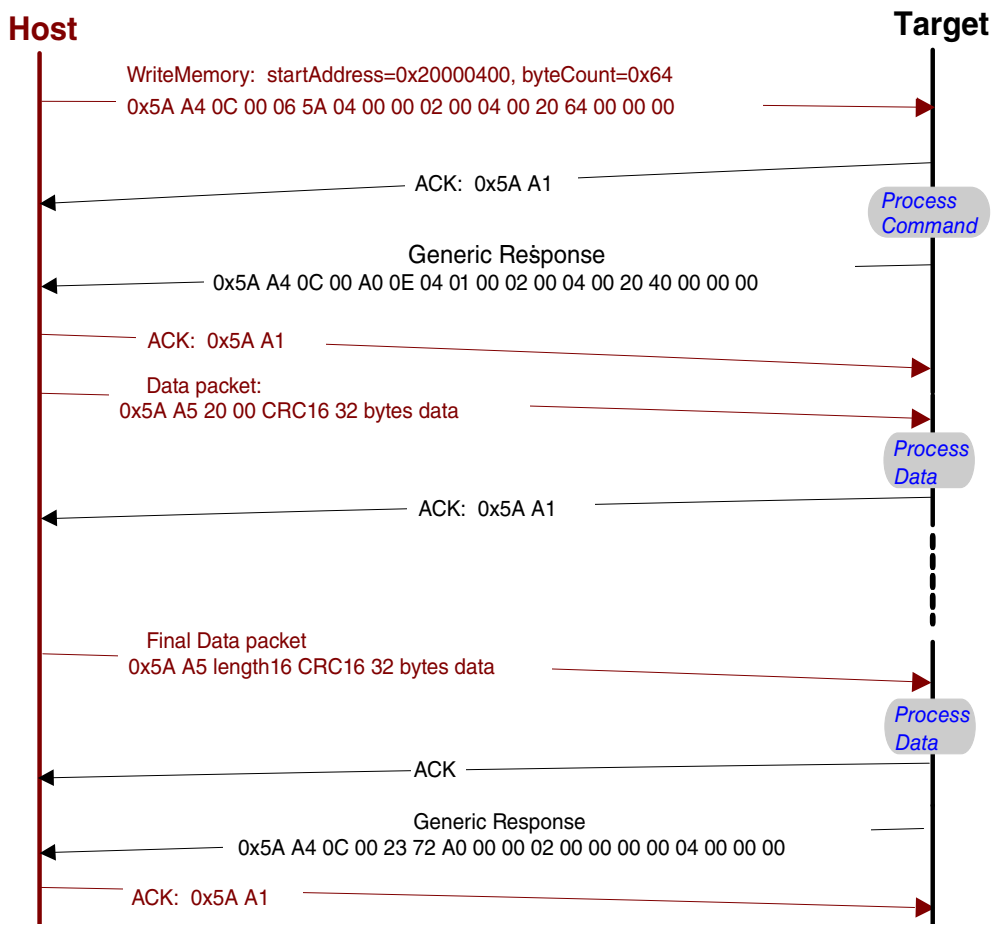


Figure 14-12. Protocol Sequence for WriteMemory Command

Table 14-28. WriteMemory Command Packet Format (Example)

WriteMemory	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x06 0x5A
Command packet	commandTag	0x04 - writeMemory
	flags	0x00
	reserved	0x00
	parameterCount	0x02

Table continues on the next page...

Table 14-28. WriteMemory Command Packet Format (Example) (continued)

WriteMemory	Parameter	Value
	startAddress	0x20000400
	byteCount	0x00000064

Data Phase: The WriteMemory command has a data phase; the host will send data packets until the number of bytes of data specified in the byteCount parameter of the WriteMemory command are received by the target.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with a status code set to kStatus_Success upon successful execution of the command, or to an appropriate error status code.

14.3.6.7 Read memory command

The ReadMemory command returns the contents of memory at the given address, for a specified number of bytes. This command can read any region of Flash memory, SRAM_L and SRAM_U memory accessible by the CPU and not protected by security.

The start address and number of bytes are the 2 parameters required for ReadMemory command.

Table 14-29. Parameters for read memory command

Byte	Parameter	Description
0-3	Start address	Start address of memory to read from
4-7	Byte count	Number of bytes to read and return to caller

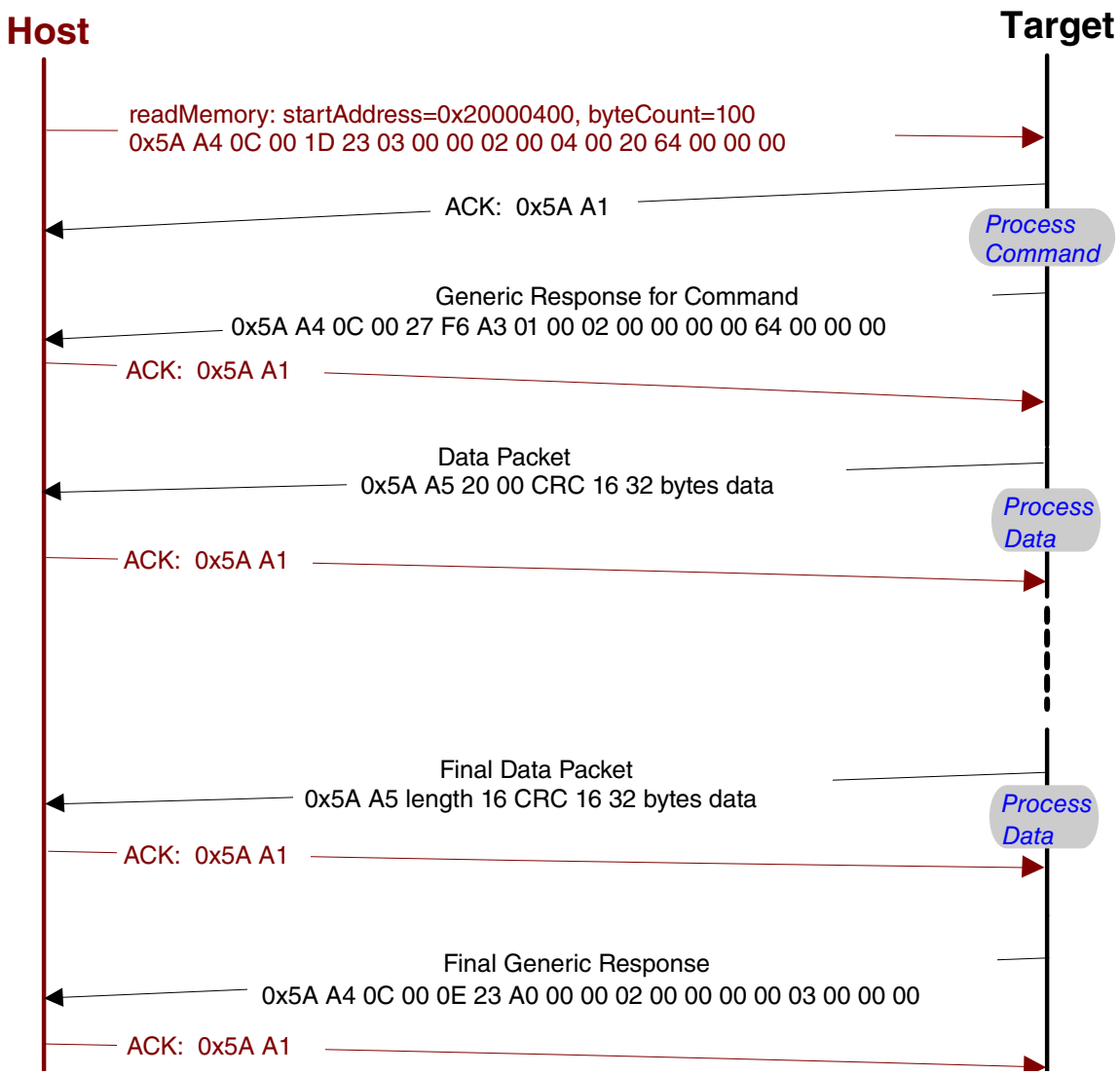


Figure 14-13. Command sequence for read memory

ReadMemory	Parameter	Value
Framing packet	Start byte	0x5A0xA4,
	packetType	kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x1D 0x23
Command packet	commandTag	0x03 - readMemory
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	startAddress	0x20000400
	byteCount	0x00000064

Data Phase: The ReadMemory command has a data phase. Since the target (Kinetis Flashloader) works in slave mode, the host need pull data packets until the number of bytes of data specified in the byteCount parameter of ReadMemory command are received by host.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

14.3.6.8 FlashSecurityDisable command

The FlashSecurityDisable command performs the flash security disable operation, by comparing the 8-byte backdoor key (provided in the command) against the backdoor key stored in the flash configuration field (at address 0x400 in the flash).

The backdoor low and high words are the only parameters required for FlashSecurityDisable command.

Table 14-30. Parameters for FlashSecurityDisable Command

Byte #	Command
0 - 3	Backdoor key low word
4 - 7	Backdoor key high word

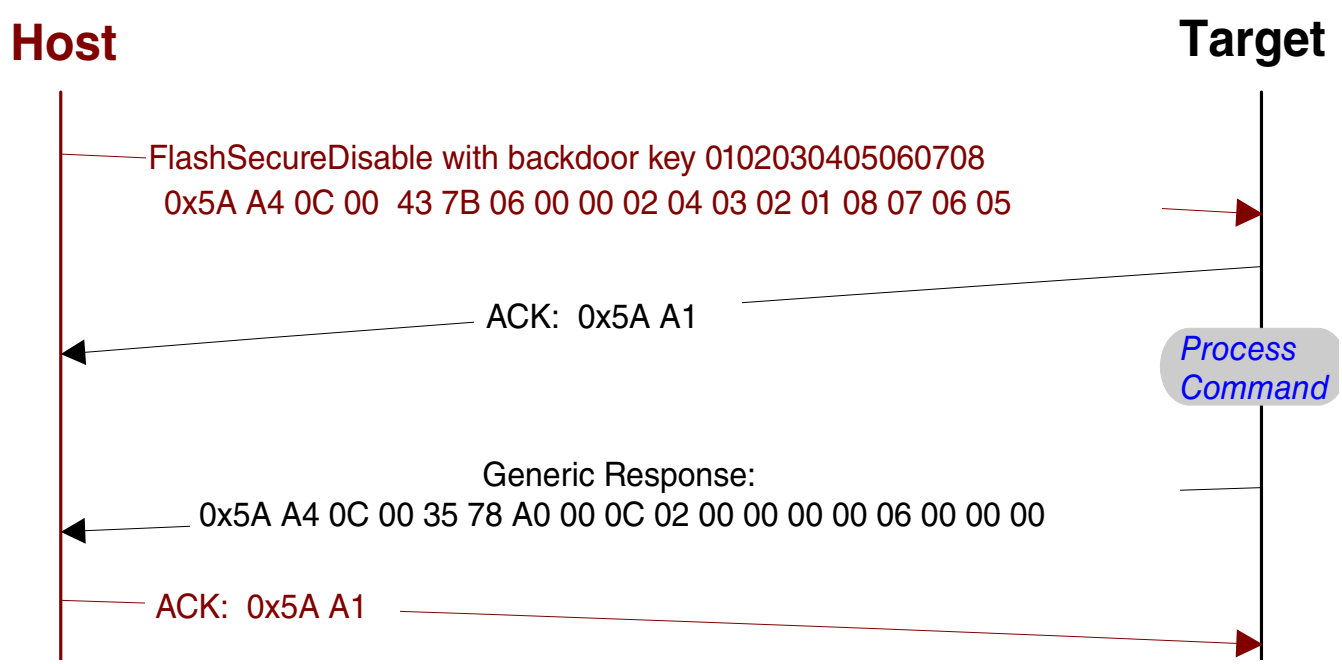


Figure 14-14. Protocol Sequence for FlashSecurityDisable Command

Table 14-31. FlashSecurityDisable Command Packet Format (Example)

FlashSecurityDisable	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x0C 0x00
	crc16	0x43 0x7B
Command packet	commandTag	0x06 - FlashSecurityDisable
	flags	0x00
	reserved	0x00
	parameterCount	0x02
	Backdoorkey_low	0x04 0x03 0x02 0x01
	Backdoorkey_high	0x08 0x07 0x06 0x05

The FlashSecurityDisable command has no data phase.

Response: The target (KinetisFlashloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

14.3.6.9 Execute command

The execute command results in the flashloader setting the program counter to the code at the provided jump address, R0 to the provided argument, and a Stack pointer to the provided stack pointer address. Prior to the jump, the system is returned to the reset state.

The Jump address, function argument pointer, and stack pointer are the parameters required for the Execute command.

Table 14-32. Parameters for Execute Command

Byte #	Command
0 - 3	Jump address
4 - 7	Argument word
8 - 11	Stack pointer address

The Execute command has no data phase.

Response: Before executing the Execute command, the target (Kinetis Flashloader) will validate the parameters and return a GenericResponse packet with a status code either set to kStatus_Success or an appropriate error status code.

14.3.6.10 Reset command

The Reset command will result in flashloader resetting the chip.

The Reset command requires no parameters.

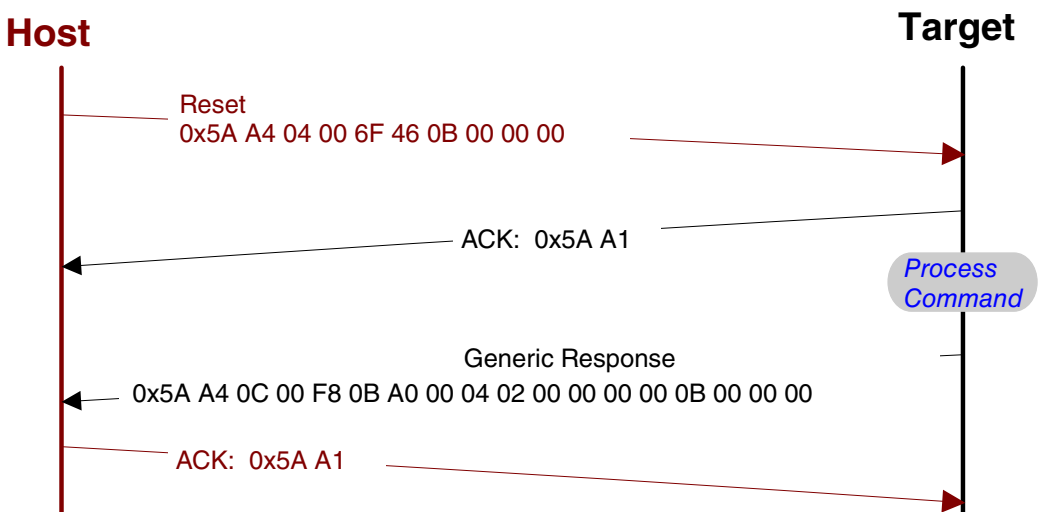


Figure 14-15. Protocol Sequence for Reset Command

Table 14-33. Reset Command Packet Format (Example)

Reset	Parameter	Value
Framing packet	start byte	0x5A
	packetType	0xA4, kFramingPacketType_Command
	length	0x04 0x00
	crc16	0x6F 0x46
Command packet	commandTag	0x0B - reset
	flags	0x00
	reserved	0x00
	parameterCount	0x00

The Reset command has no data phase.

Response: The target (Kinetis Flashloader) will return a GenericResponse packet with status code set to kStatus_Success, before resetting the chip.

14.4 Peripherals Supported

This section describes the peripherals supported by the Kinetis Flashloader.

14.4.1 I2C Peripheral

The Kinetis Flashloader supports loading data into flash via the I2C peripheral, where the I2C peripheral serves as the I2C slave. A 7-bit slave address is used during the transfer.

The Kinetis Flashloader uses 0x10 as the I2C slave address, and supports 400 kbps as the I2C baud rate.

Because the I2C peripheral serves as an I2C slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

- An incoming packet is sent by the host with a selected I2C slave address and the direction bit is set as write.
- An outgoing packet is read by the host with a selected I2C slave address and the direction bit is set as read.
- 0x00 will be sent as the response to host if the target is busy with processing or preparing data.

The following flow charts demonstrate the communication flow of how the host reads ping packet, ACK and response from the target.

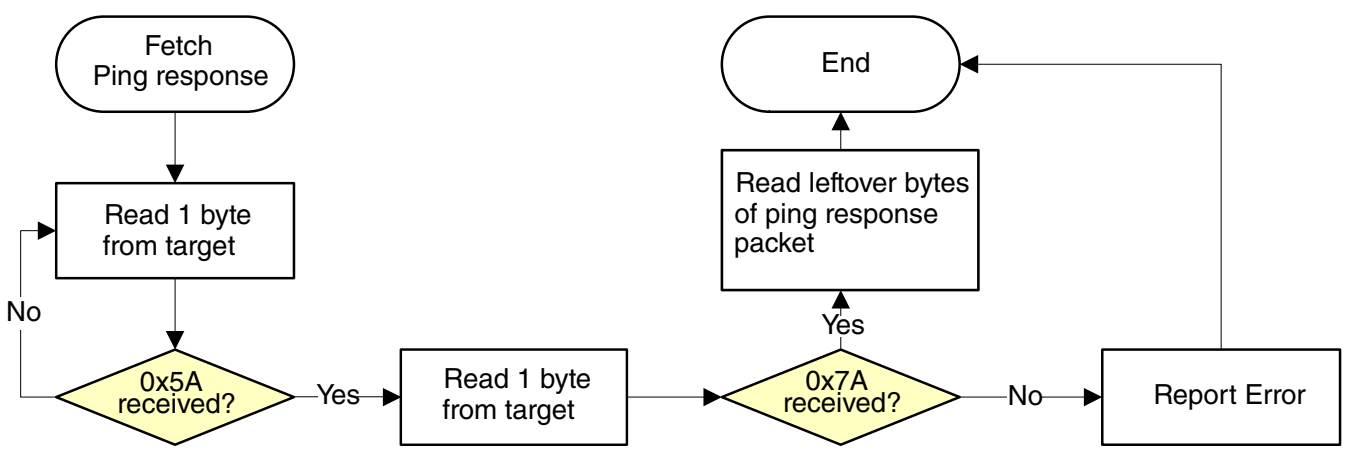


Figure 14-16. Host reads ping response from target via I2C

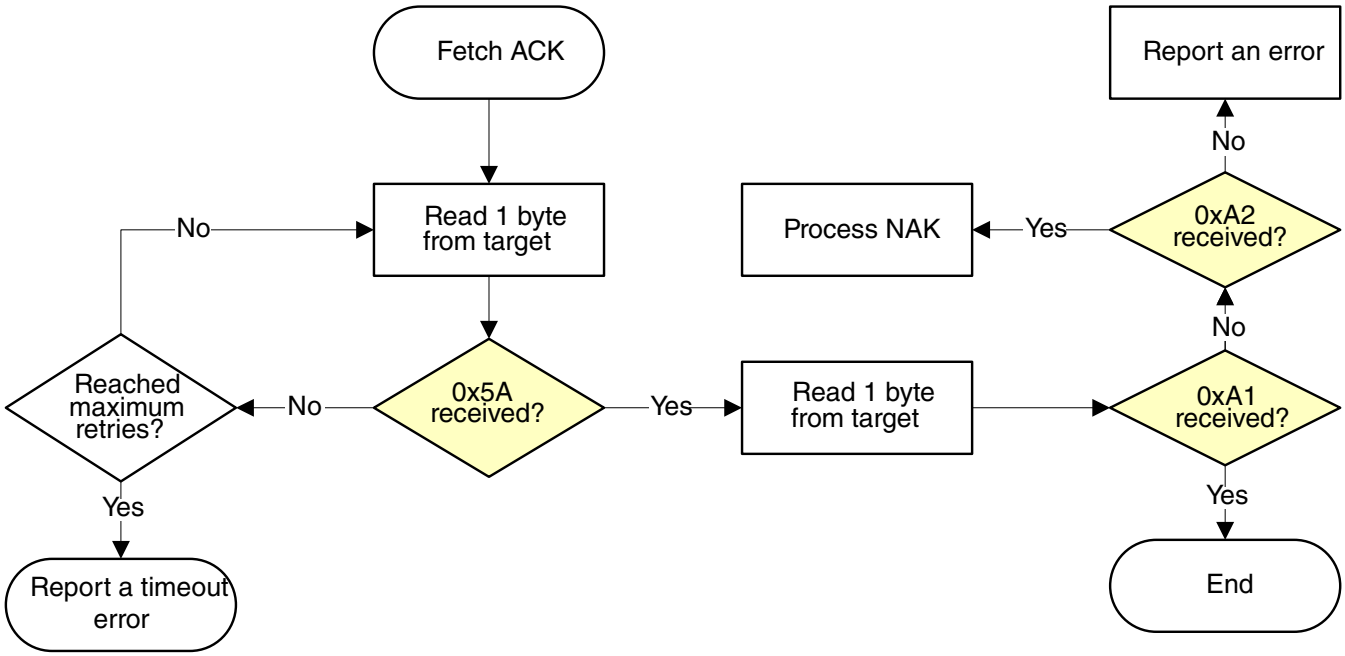


Figure 14-17. Host reads ACK packet from target via I2C

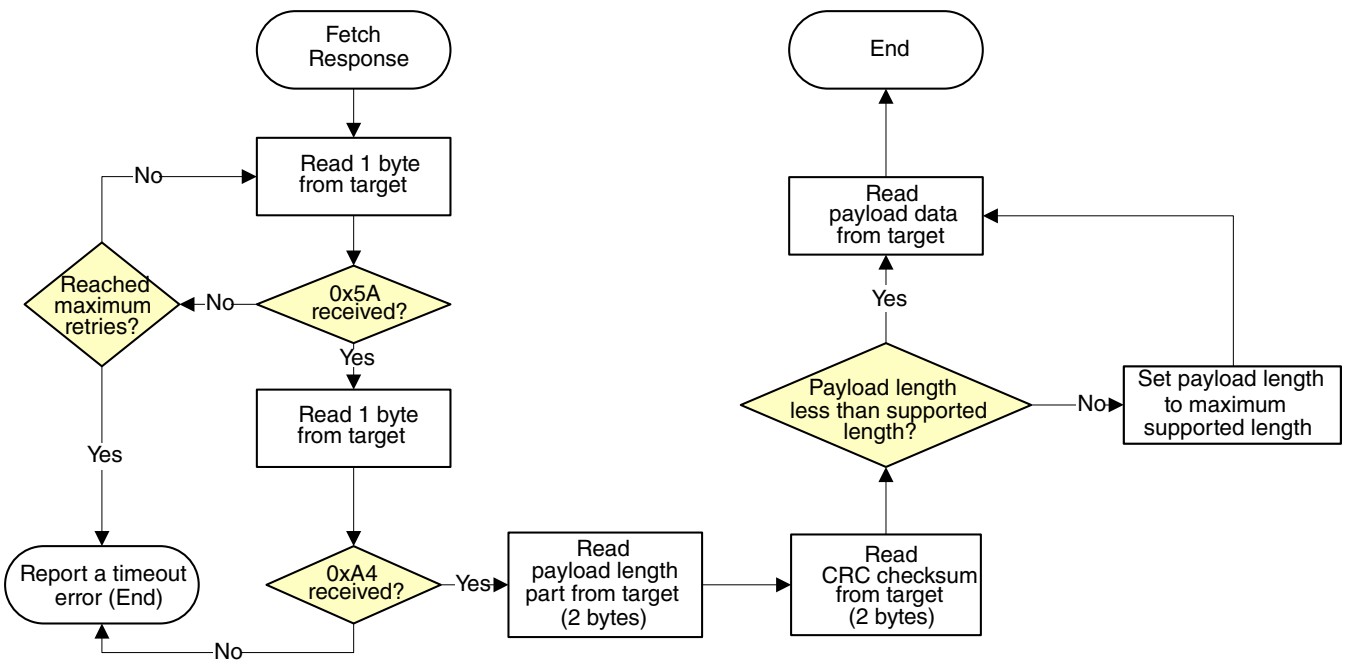


Figure 14-18. Host reads response from target via I2C

14.4.2 SPI Peripheral

The Kinetis Flashloader supports loading data into flash via the SPI peripheral, where the SPI peripheral serves as a SPI slave.

The Kinetis Flashloader supports 400 kbps as the SPI baud rate.

The SPI peripheral uses the following bus attributes:
- Clock Phase = 1 (Second Edge)
- Clock Polarity = 1 (Active Low)

Because the SPI peripheral serves as a SPI slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

The transfer on SPI is slightly different from I2C:
- Host will receive 1 byte after it sends out any byte.
- Received bytes should be ignored when host is sending out bytes to target
- Host starts reading bytes by sending 0x00s to target
- The byte 0x00 will be sent as response to host if target is under the following conditions:
  - Processing incoming packet
  - Preparing outgoing data
  - Received invalid data

The SPI bus configuration is:
- Phase = 1; data is sampled on rising edges
- Polarity = 1; idle is high
- MSB is transmitted first

For any transfer where the target does not have actual data to send, the target (slave) is responsible for ensuring that 0x00 bytes will be returned to the host (master). The host uses framing packets to identify real data and not "dummy" 0x00 bytes (which do not have framing packets).

The following flowcharts demonstrate how the host reads a ping response, an ACK and a command response from target via SPI.

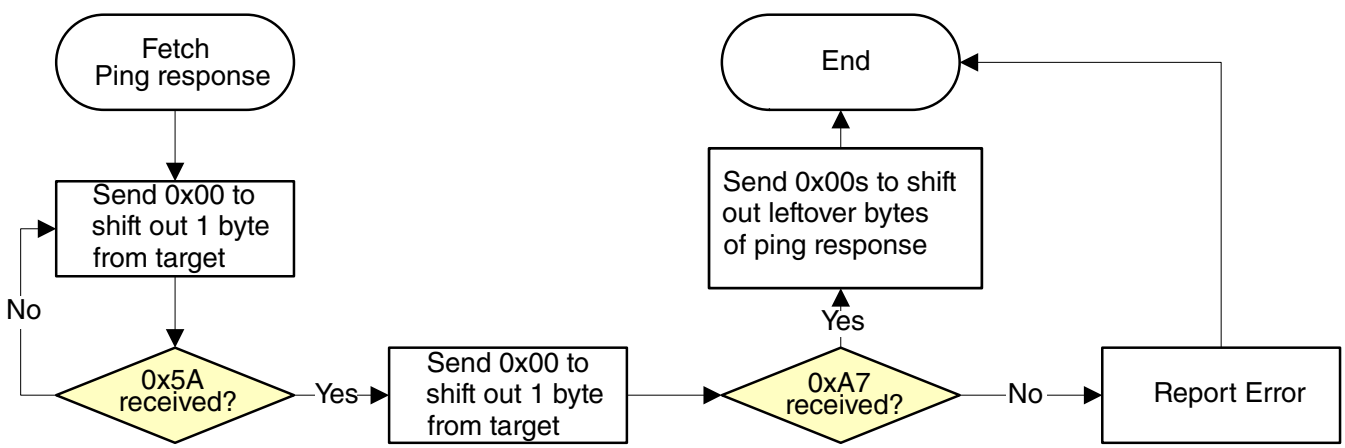


Figure 14-19. Host reads ping packet from target via SPI

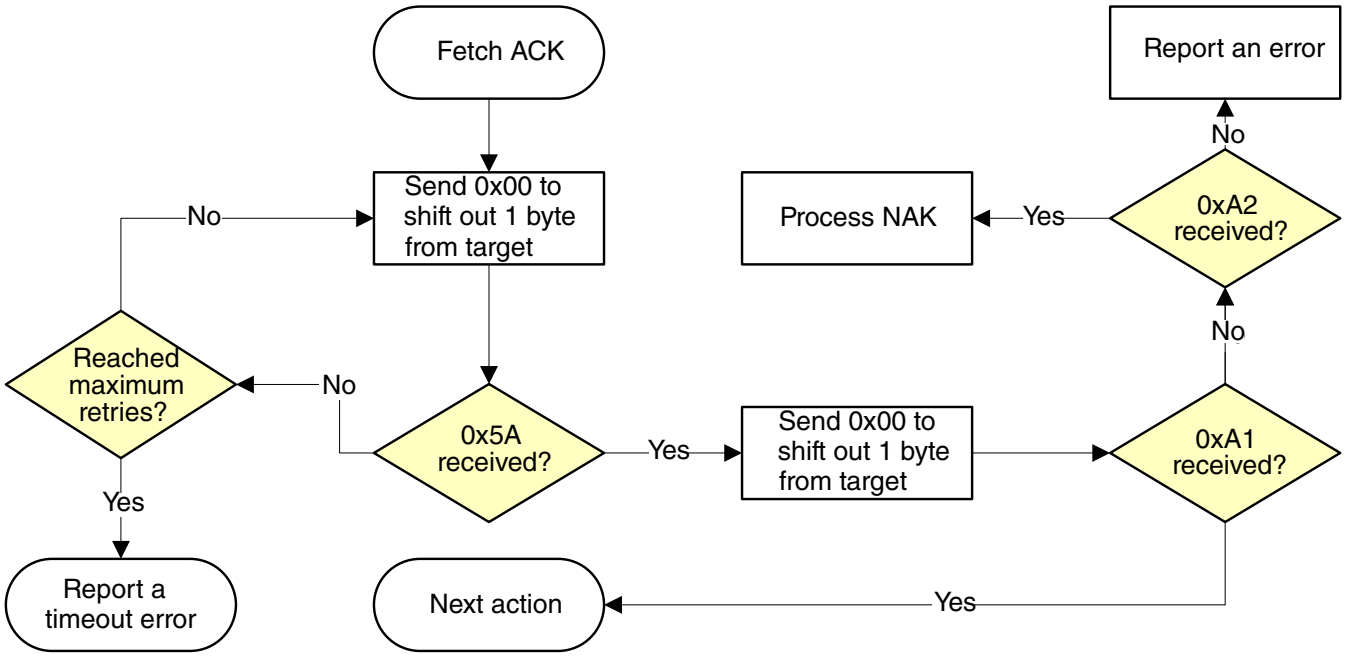


Figure 14-20. Host reads ACK from target via SPI

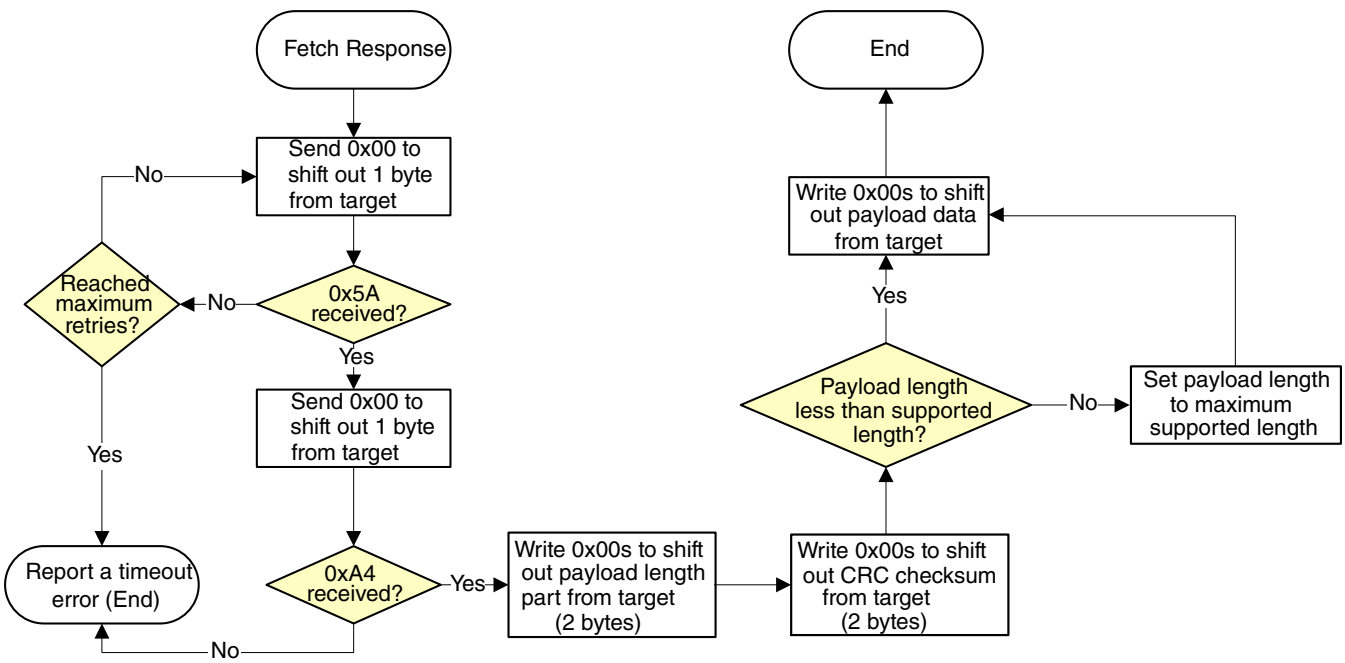


Figure 14-21. Host reads response from target via SPI

14.4.3 UART Peripheral

The Kinetis Flashloader integrates an autobaud detection algorithm for the UART peripheral, thereby providing flexible baud rate choices.

Autobaud feature: If UART*n* is used to connect to the flashloader, then the UART*n*_RX pin must be kept high and not left floating during the detection phase in order to comply with the autobaud detection algorithm. After the flashloader detects the ping packet (0x5A 0xA6) on UART*n*_RX, the flashloader firmware executes the autobaud sequence. If the baudrate is successfully detected, then the flashloader will send a ping packet response [(0x5A 0xA7), protocol version (4 bytes), protocol version options (2 bytes) and crc16 (2 bytes)] at the detected baudrate. The Kinetis Flashloader then enters a loop, waiting for flashloader commands via the UART peripheral.

NOTE

- The autobaud feature requires a ping packet with a higher accuracy (+/-3%), or the ping packet will be ignored as noise.
- The data bytes of the ping packet must be sent continuously (with no more than 80 ms between bytes) in a fixed UART transmission mode (8-bit data, no parity bit and 1 stop bit). If the bytes of the ping packet are sent one-by-one with more than 80 ms delay between them, then the autobaud detection algorithm may calculate an incorrect baud rate. In this case, the autobaud detection state machine should be reset.

Supported baud rates: The baud rate is closely related to the MCU core and system clock frequencies. Typical baud rates supported are 9600, 19200, 38400, 57600, and 115200.

Packet transfer: After autobaud detection succeeds, flashloader communications can take place over the UART peripheral. The following flow charts show:

- How the host detects an ACK from the target
- How the host detects a ping response from the target
- How the host detects a command response from the target

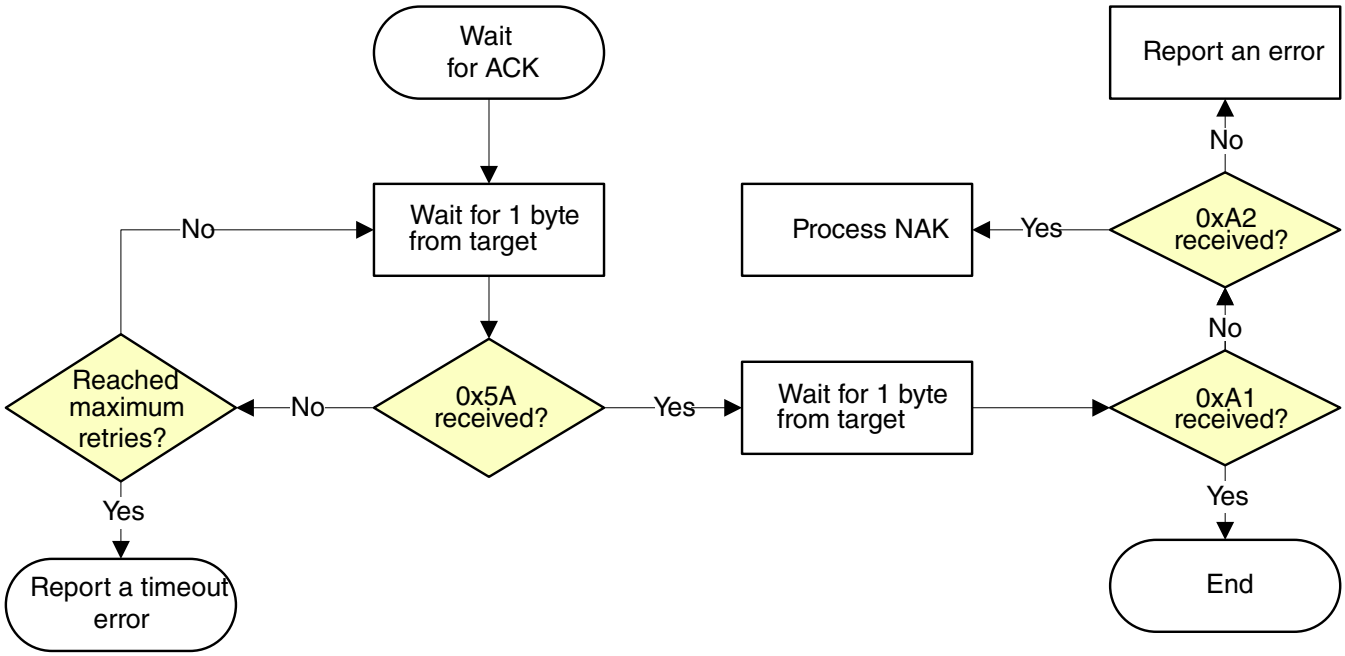


Figure 14-22. Host reads an ACK from target via UART

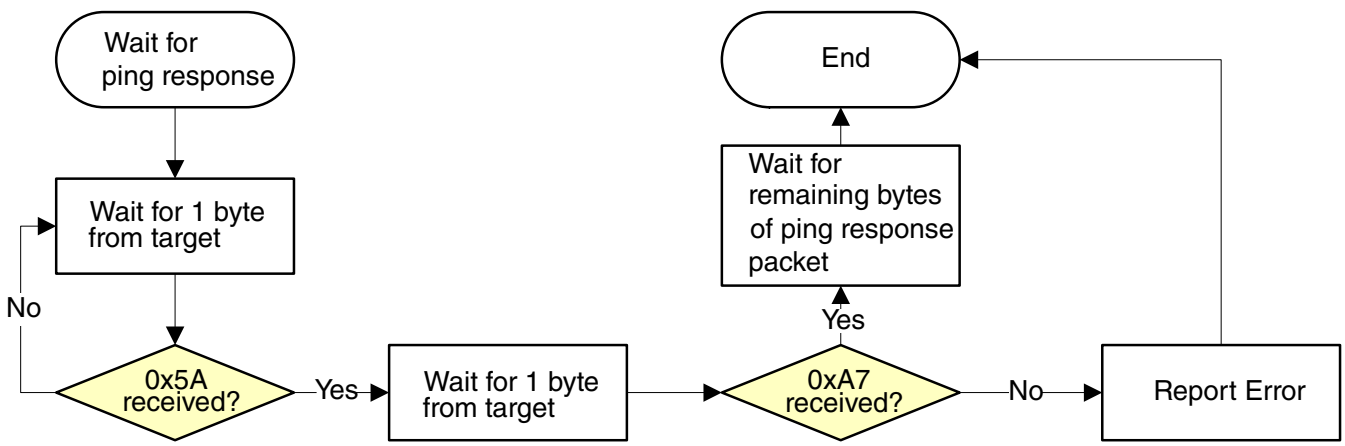


Figure 14-23. Host reads a ping response from target via UART

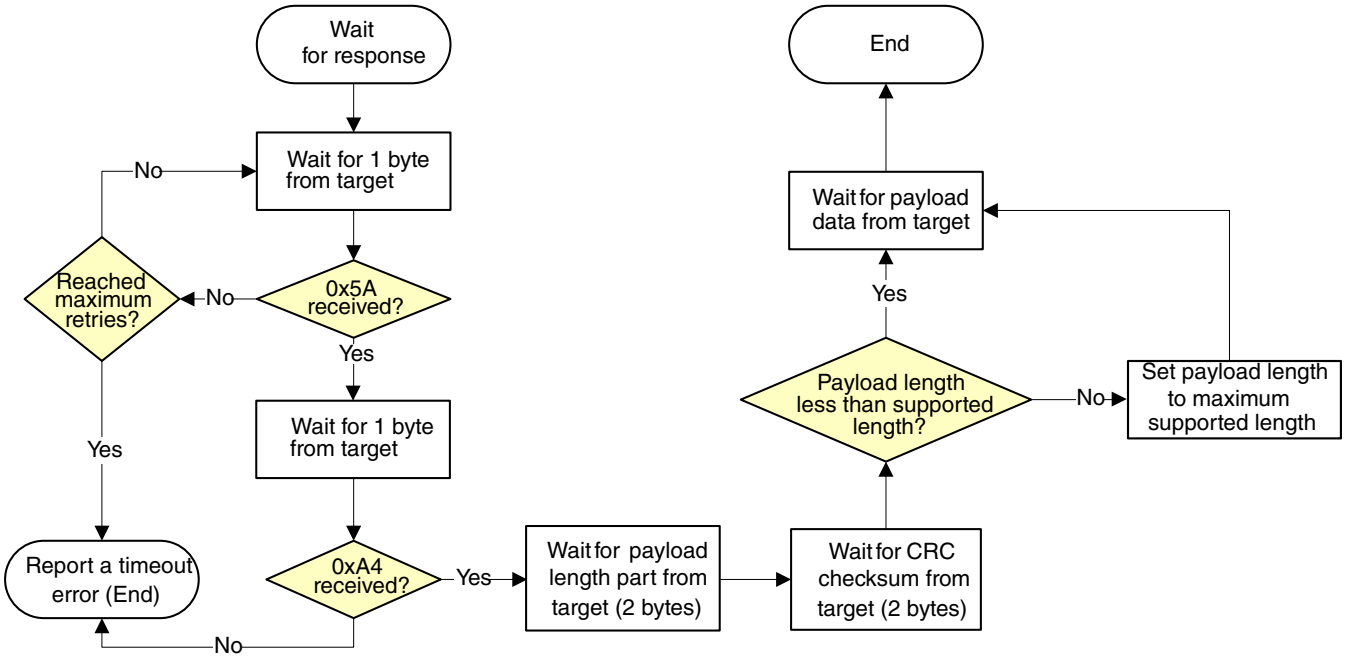


Figure 14-24. Host reads a command response from target via UART

14.4.4 CAN (or FlexCAN) Peripheral

The Kinetis Flashloader supports loading data into flash via the FlexCAN peripheral. Transfers to FlexCAN are supported at several predefined speeds:

- 125 kHz
- 250 kHz
- 500 kHz
- 1 MHz (the default transfer rate)

The host application must use one of the several supported speeds for FlexCAN. In Flashloader, it supports automatic speed detection within supported speeds. The Flashloader will enter the listen mode in the beginning with the initial speed (default speed 1 MHz). Once the host sends a ping to a specific node, it will generate traffic on the FlexCAN bus. Because the Flashloader is in a listen mode, it will be able to check if the local node speed is correct, by detecting errors.

- If there is an error, then some transfers may not be at the right speed.
- The Flashloader will change the speed setting and check again.
- If there is no error, it means that the transfer speed is correct, and it changes the settings back to normal receiving mode, to see if there is a package for this node.
- The host side should also have reasonable time tolerance during the automatic speed detection period. If there is a timeout, it means that there is no response from the specific node, or there is a real error (and it should report the error to the application).

The following flowcharts show how the host reads a ping packet, ACK and response from the target.

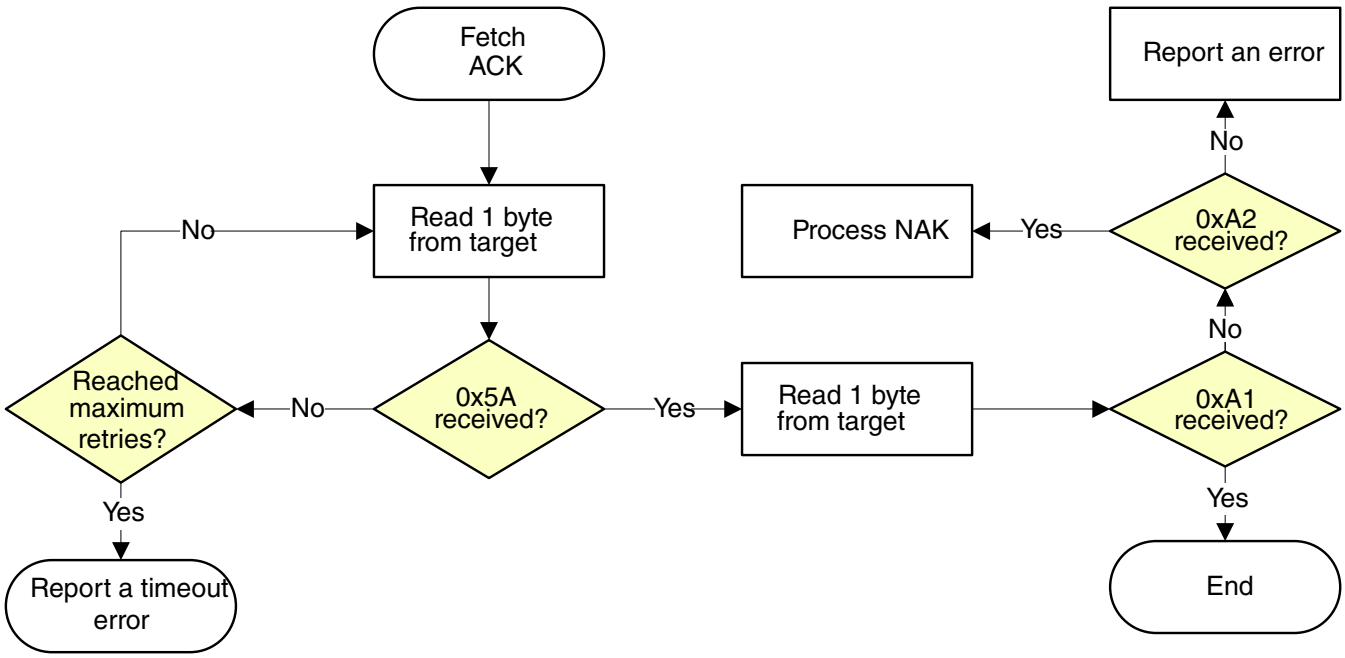


Figure 14-25. Host reads an ACK from target via FlexCAN

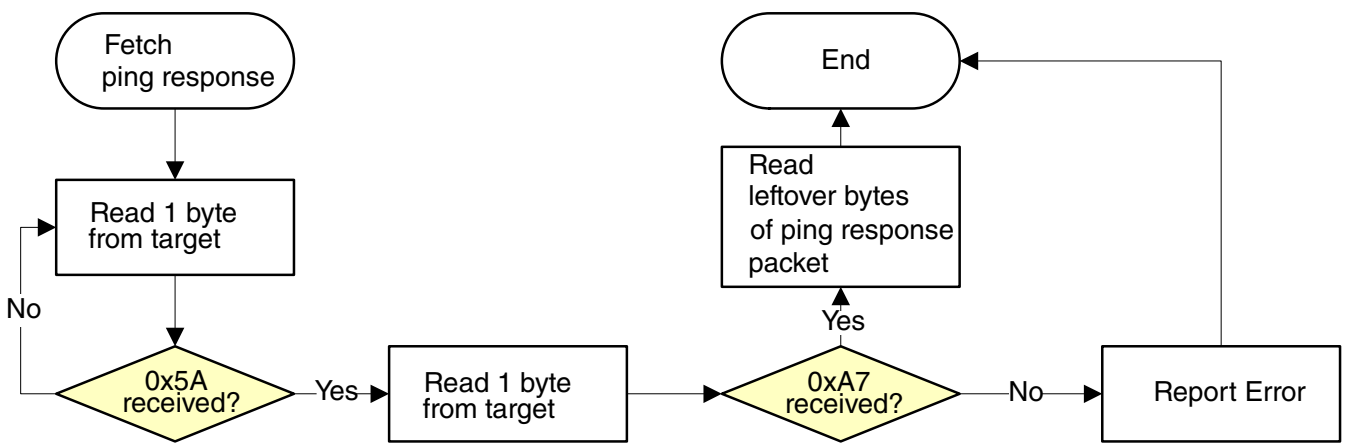


Figure 14-26. Host reads a ping response from target via FlexCAN

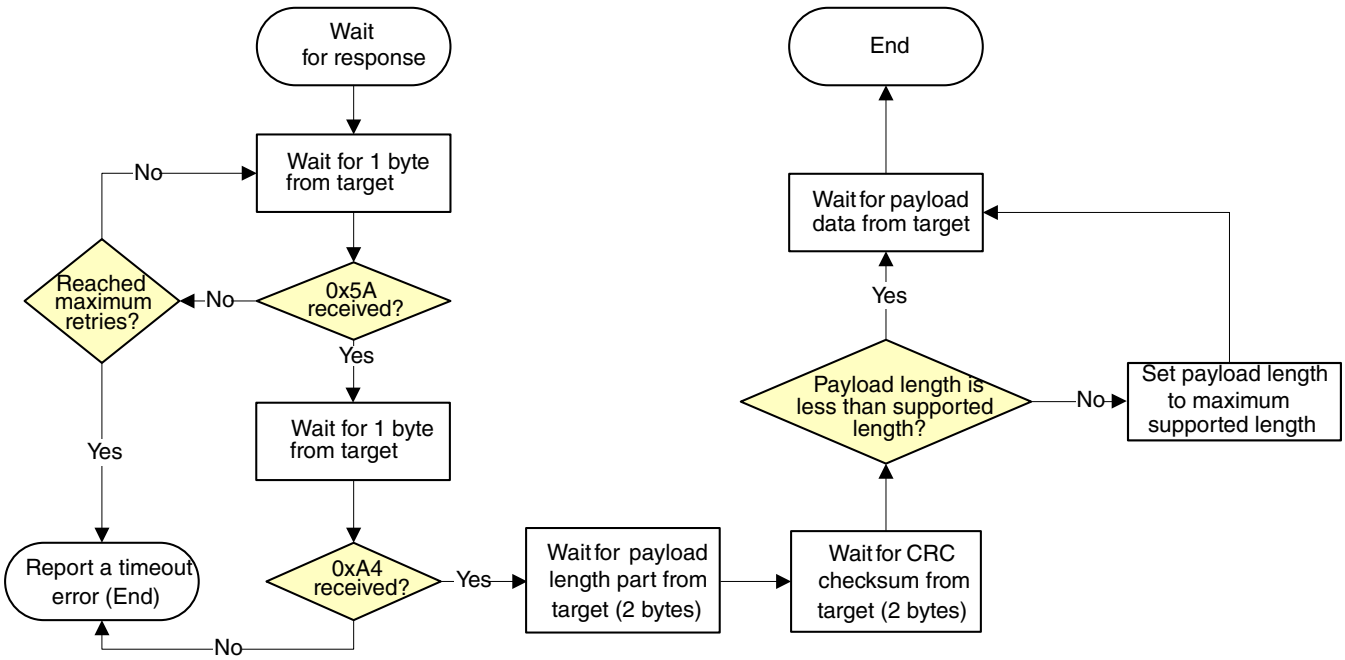


Figure 14-27. Host reads a command response from target via FlexCAN

14.5 Get/SetProperty Command Properties

This section lists the properties of the GetProperty and SetProperty commands.

Table 14-34. Properties used by Get/SetProperty Commands, sorted by Value

Property	Writable	Tag Value	Size	Description
CurrentVersion	No	01h	4	Current flashloader version.
AvailablePeripherals	No	02h	4	The set of peripherals supported on this chip.
FlashStartAddress	No	03h	4	Start address of program flash.
FlashSizeInBytes	No	04h	4	Size in bytes of program flash.
FlashSectorSize	No	05h	4	The size in bytes of one sector of program flash. This is the minimum erase size.
FlashBlockCount	No	06h	4	Number of blocks in the flash array.
AvailableCommands	No	07h	4	The set of commands supported by the flashloader.
VerifyWrites	Yes	0Ah	4	Controls whether the flashloader will verify writes to flash. VerifyWrites feature is enabled by default. 0 - No verification is done. 1 - Enable verification.
MaxPacketSize	No	0Bh	4	Maximum supported packet size for the currently active peripheral interface.

Table continues on the next page...

Table 14-34. Properties used by Get/SetProperty Commands, sorted by Value (continued)

Property	Writable	Tag Value	Size	Description
ReservedRegions	No	0Ch	16	List of memory regions reserved by the flashloader. Returned as value pairs (<start-address-of-region>, <end-address-of-region>). • If HasDataPhase flag is not set, then the Response packet parameter count indicates the number of pairs. • If HasDataPhase flag is set, then the second parameter is the number of bytes in the data phase.
RAMStartAddress	No	0Eh	4	Start address of RAM segment. The first parameter to GetProperty command identifies the segment. See the device specific memory map for number of RAM segments the device contains.
RAMSizeInBytes	No	0Fh	4	Size in bytes of RAM segment. The first parameter to GetProperty command identifies the segment. See the device specific memory map for number of RAM segments the device contains.
SystemDeviceId	No	10h	4	Value of the Kinetis System Device Identification register.
FlashSecurityState	No	11h	4	Indicates whether Flash security is enabled 0 - Flash security is disabled 1 - Flash security is enabled
UniqueDeviceId	No	12h	16	Unique device identification, value of Kinetis Unique Identification registers (16 for K series devices, 12 for KL series devices)
FlashReadMargin	Yes	16h	4	The margin level setting for flash erase and program verify commands. 0 = Normal 1 = User (default) 2 = Factory

14.5.1 Property Definitions

Get/Set property definitions are provided in this section.

14.5.1.1 CurrentVersion Property

The value of this property is a 4-byte structure containing the current version of the flashloader.

Table 14-35. Fields of CurrentVersion property:

Bits	[31:24]	[23:16]	[15:8]	[7:0]
Field	Name = 'K' (0x4B)	Major version	Minor version	Bugfix version

14.5.1.2 AvailablePeripherals Property

The value of this property is a bitfield that lists the peripherals supported by the flashloader and the hardware on which it is running.

Table 14-36. Peripheral bits:

Bit	[31:7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
Peripheral	Reserved	Reserved	Reserved	Reserved	CAN Slave	SPI Slave	I2C Slave	UART

If the peripheral is available, then the corresponding bit will be set in the property value. All reserved bits must be set to 0.

14.5.1.3 AvailableCommands Property

This property value is a bitfield with set bits indicating the commands enabled in the flashloader. Only commands that can be sent from the host to the target are listed in the bitfield. Response commands such as GenericResponse are excluded.

The bit number that identifies whether a command is present is the command's tag value minus 1. 1 is subtracted from the command tag because the lowest command tag value is 0x01. To get the bit mask for a given command, use this expression:

```
mask = 1 << (tag - 1)
```

Table 14-37. Command bits:

Bit	[31:18]	[17]	[16]	[15]	[14]	[13]	[12]	[11]	[10]	[9]	[8]	[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]

Table continues on the next page...

Table 14-37. Command bits: (continued)

Command	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	SetProperty	Reset	Reserved	Execute	Reserved	GetProperty	FlashSecurityDisable	FillMemory	WriteMemory	ReadMemory	FlashEraseRegion	FlashEraseAll

14.6 Kinetis Flashloader Status Error Codes

This section describes the status error codes that the Kinetis Flashloader returns to the host.

Table 14-38. Kinetis Flashloader Status Error Codes, sorted by Value

Error Code	Value	Description
kStatus_Success	0	Operation succeeded without error.
kStatus_Fail	1	Operation failed with a generic error.
kStatus_ReadOnly	2	Requested value cannot be changed because it is read-only.
kStatus_OutOfRange	3	Requested value is out of range.
kStatus_InvalidArgument	4	The requested command's argument is undefined.
kStatus_Timeout	5	A timeout occurred.
kStatus_FlashSizeError	100	Not used.
kStatus_FlashAlignmentError	101	Address or length does not meet required alignment.
kStatus_FlashAddressError	102	Address or length is outside addressable memory.
kStatus_FlashAccessError	103	The FTFA_FSTAT[ACCERR] bit is set.
kStatus_FlashProtectionViolation	104	The FTFA_FSTAT[FPVIOL] bit is set.
kStatus_FlashCommandFailure	105	The FTFA_FSTAT[MGSTAT0] bit is set.
kStatus_FlashUnknownProperty	106	Unknown Flash property.
kStatus_FlashEraseKeyError	107	The key provided does not match the programmed flash key.
kStatus_FlashRegionExecuteOnly	108	The area of flash is protected as execute only.
kStatus_I2C_SlaveTxUnderrun	200	I2C Slave TX Underrun error.
kStatus_I2C_SlaveRxOverrun	201	I2C Slave RX Overrun error.
kStatus_I2C_AribtrationLost	202	I2C Arbitration Lost error.
kStatus_SPI_SlaveTxUnderrun	300	SPI Slave TX Underrun error.
kStatus_SPI_SlaveRxOverrun	301	SPI Slave RX Overrun error.
kStatus_SPI_Timeout	302	SPI tranfser timed out.

Table continues on the next page...

Table 14-38. Kinetis Flashloader Status Error Codes, sorted by Value (continued)

Error Code	Value	Description
kStatus_SPI_Busy	303	SPI instance is already busy performing a transfer.
kStatus_SPI_NoTransferInProgress	304	Attempt to abort a transfer when no transfer was in progress.
kStatus_UnknownCommand	10000	The requested command value is undefined.
kStatus_SecurityViolation	10001	Command is disallowed because flash security is enabled.
kStatus_AbortDataPhase	10002	Abort the data phase early.
kStatusMemoryRangeInvalid	10200	Memory range conflicts with a protected region.
kStatus_UnknownProperty	10300	The requested property value is undefined.
kStatus_ReadOnlyProperty	10301	The requested property value cannot be written.
kStatus_InvalidPropertyValue	10302	The specified property value is invalid.
kStatus_AppCrcCheckPassed	10400	CRC check is valid and passed.
kStatus_AppCrcCheckFailed	10401	CRC check is valid but failed.
kStatus_AppCrcCheckInactive	10402	CRC check is inactive.
kStatus_AppCrcCheckInvalid	10403	CRC check is invalid, because the BCA is invalid or the CRC parameters are unset (all 0xFF bytes).
kStatus_AppCrcCheckOutOfRange	10404	CRC check is valid but addresses are out of range.

Chapter 15
Reset Control Module (RCM)

15.1 Introduction

Information found here describes the registers of the Reset Control Module (RCM). The RCM implements many of the reset functions for the chip. See the chip's reset chapter for more information.

See AN4503: Power Management for Kinetis MCUs for further details on using the RCM.

15.2 Reset memory map and register descriptions

The RCM Memory Map/Register Definition can be found here.

The Reset Control Module (RCM) registers provide reset status information and reset filter control.

NOTE

The RCM registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

RCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_F000	System Reset Status Register 0 (RCM_SRS0)	8	R	82h	15.2.1/254
4007_F001	System Reset Status Register 1 (RCM_SRS1)	8	R	00h	15.2.2/255
4007_F004	Reset Pin Filter Control register (RCM_RPFC)	8	R/W	00h	15.2.3/257
4007_F005	Reset Pin Filter Width register (RCM_RPFW)	8	R/W	00h	15.2.4/258

Table continues on the next page...

RCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_F008	Sticky System Reset Status Register 0 (RCM_SSRS0)	8	R/W	82h	15.2.5/259
4007_F009	Sticky System Reset Status Register 1 (RCM_SSRS1)	8	R/W	00h	15.2.6/260

15.2.1 System Reset Status Register 0 (RCM_SRS0)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:
- POR (including LVD) — 0x82
- LVD (without POR) — 0x02
- VLLS mode wakeup due to $\overline{\text{RESET}}$ pin assertion — 0x41
- VLLS mode wakeup due to other wakeup sources — 0x01
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007_F000h base + 0h offset = 4007_F000h

Bit	7	6	5	4	3	2	1	0
Read	POR	PIN	WDOG	0	LOL	LOC	LVD	WAKEUP
Write								
Reset	1	0	0	0	0	0	1	0

RCM_SRS0 field descriptions

Field	Description
7 POR	Power-On Reset Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 PIN	External Reset Pin Indicates a reset has been caused by an active-low level on the external $\overline{\text{RESET}}$ pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog

Table continues on the next page...

RCM_SRS0 field descriptions (continued)

Field	Description
	Indicates a reset has been caused by the watchdog timer timing out. This reset source can be blocked by disabling the watchdog. 0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LOL	Loss-of-Lock Reset Indicates a reset has been caused by a loss of lock in the MCG PLL. See the MCG description for information on the loss-of-clock event. 0 Reset not caused by a loss of lock in the PLL 1 Reset caused by a loss of lock in the PLL
2 LOC	Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The MCG clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed MCG description for information on enabling the clock monitor. 0 Reset not caused by a loss of external clock. 1 Reset caused by a loss of external clock.
1 LVD	Low-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR
0 WAKEUP	Low Leakage Wakeup Reset Indicates a reset has been caused by an enabled LLWU module wakeup source while the chip was in a low leakage mode. Any enabled wakeup source in a VLLSx mode causes a reset. This bit is cleared by any reset except WAKEUP. 0 Reset not caused by LLWU module wakeup source 1 Reset caused by LLWU module wakeup source

15.2.2 System Reset Status Register 1 (RCM_SRS1)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x00
- LVD (without POR) — 0x00

- VLLS mode wakeup — 0x00
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007_F000h base + 1h offset = 4007_F001h

Bit	7	6	5	4	3	2	1	0
Read	0	0	SACKERR	0	MDM_AP	SW	LOCKUP	0
Write								
Reset	0	0	0	0	0	0	0	0

RCM_SRS1 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SACKERR	Stop Mode Acknowledge Error Reset Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 MDM_AP	MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit
2 SW	Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the Arm core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
1 LOCKUP	Core Lockup Indicates a reset has been caused by the Arm core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

15.2.3 Reset Pin Filter Control register (RCM_RPFC)

NOTE

The reset values of bits 2-0 are for Chip POR only. They are unaffected by other reset types.

NOTE

The bus clock filter is reset when disabled or when entering stop mode. The LPO filter is reset when disabled .

Address: 4007_F000h base + 4h offset = 4007_F004h

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

RCM_RPFC field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RSTFLTSS	Reset Pin Filter Select in Stop Mode Selects how the reset pin filter is enabled in Stop and VLPS modes , and also during VLLS mode. On exit from VLLS mode, this bit should be reconfigured before clearing PMC_REGSC[ACKISO]. 0 All filtering disabled 1 LPO clock filter enabled
RSTFLTSRW	Reset Pin Filter Select in Run and Wait Modes Selects how the reset pin filter is enabled in run and wait modes. 00 All filtering disabled 01 Bus clock filter enabled for normal operation 10 LPO clock filter enabled for normal operation 11 Reserved

15.2.4 Reset Pin Filter Width register (RCM_RPFW)

NOTE

The reset values of the bits in the RSTFLTSEL field are for Chip POR only. They are unaffected by other reset types.

Address: 4007_F000h base + 5h offset = 4007_F005h

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

RCM_RPFW field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
RSTFLTSEL	Reset Pin Filter Bus Clock Select Selects the reset pin bus clock filter width. 00000 Bus clock filter count is 1 00001 Bus clock filter count is 2 00010 Bus clock filter count is 3 00011 Bus clock filter count is 4 00100 Bus clock filter count is 5 00101 Bus clock filter count is 6 00110 Bus clock filter count is 7 00111 Bus clock filter count is 8 01000 Bus clock filter count is 9 01001 Bus clock filter count is 10 01010 Bus clock filter count is 11 01011 Bus clock filter count is 12 01100 Bus clock filter count is 13 01101 Bus clock filter count is 14 01110 Bus clock filter count is 15 01111 Bus clock filter count is 16 10000 Bus clock filter count is 17 10001 Bus clock filter count is 18 10010 Bus clock filter count is 19 10011 Bus clock filter count is 20 10100 Bus clock filter count is 21 10101 Bus clock filter count is 22 10110 Bus clock filter count is 23 10111 Bus clock filter count is 24 11000 Bus clock filter count is 25

Table continues on the next page...

RCM_RPFW field descriptions (continued)

Field	Description
	11001 Bus clock filter count is 26 11010 Bus clock filter count is 27 11011 Bus clock filter count is 28 11100 Bus clock filter count is 29 11101 Bus clock filter count is 30 11110 Bus clock filter count is 31 11111 Bus clock filter count is 32

15.2.5 Sticky System Reset Status Register 0 (RCM_SSRS0)

This register includes status flags to indicate all reset sources since the last POR, LVD or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_F000h base + 8h offset = 4007_F008h

Bit	7	6	5	4	3	2	1	0
Read	SPOR	SPIN	SWDOG	0	SLOL	SLOC	SLVD	SWAKEUP
Write	w1c	w1c	w1c		w1c	w1c	w1c	w1c
Reset	1	0	0	0	0	0	1	0

RCM_SSRS0 field descriptions

Field	Description
7 SPOR	Sticky Power-On Reset Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 SPIN	Sticky External Reset Pin Indicates a reset has been caused by an active-low level on the external $\overline{\text{RESET}}$ pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 SWDOG	Sticky Watchdog Indicates a reset has been caused by the watchdog timer timing out.This reset source can be blocked by disabling the watchdog.

Table continues on the next page...

RCM_SSRS0 field descriptions (continued)

Field	Description
	0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SLOL	Sticky Loss-of-Lock Reset Indicates a reset has been caused by a loss of lock in the MCG PLL. See the MCG description for information on the loss-of-clock event. 0 Reset not caused by a loss of lock in the PLL 1 Reset caused by a loss of lock in the PLL
2 SLOC	Sticky Loss-of-Clock Reset Indicates a reset has been caused by a loss of external clock. The MCG clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed MCG description for information on enabling the clock monitor. 0 Reset not caused by a loss of external clock. 1 Reset caused by a loss of external clock.
1 SLVD	Sticky Low-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR
0 SWAKEUP	Sticky Low Leakage Wakeup Reset Indicates a reset has been caused by an enabled LLWU modulewakeup source while the chip was in a low leakage mode. Any enabled wakeup source in a VLLSx mode causes a reset. 0 Reset not caused by LLWU module wakeup source 1 Reset caused by LLWU module wakeup source

15.2.6 Sticky System Reset Status Register 1 (RCM_SSRS1)

This register includes status flags to indicate all reset sources since the last POR, LVD or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_F000h base + 9h offset = 4007_F009h

Bit	7	6	5	4	3	2	1	0
Read	0	0	SSACKERR	0	SMDM_AP	SSW	SLOCKUP	0
Write			w1c		w1c	w1c	w1c	
Reset	0	0	0	0	0	0	0	0

RCM_SSRS1 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SSACKERR	Sticky Stop Mode Acknowledge Error Reset Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SMDM_AP	Sticky MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit
2 SSW	Sticky Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the Arm core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
1 SLOCKUP	Sticky Core Lockup Indicates a reset has been caused by the Arm core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Chapter 16
System Mode Controller (SMC)

16.1 Introduction

The System Mode Controller (SMC) is responsible for sequencing the system into and out of all low-power Stop and Run modes.

Specifically, it monitors events to trigger transitions between power modes while controlling the power, clocks, and memories of the system to achieve the power consumption and functionality of that mode.

This chapter describes all the available low-power modes, the sequence followed to enter/exit each mode, and the functionality available while in each of the modes.

The SMC is able to function during even the deepest low power modes.

See AN4503: Power Management for Kinetis MCUs for further details on using the SMC.

16.2 Modes of operation

The Arm CPU has three primary modes of operation:

- Run
- Sleep
- Deep Sleep

The WFI or WFE instruction is used to invoke Sleep and Deep Sleep modes. Run, Wait, and Stop are the common terms used for the primary operating modes of Kinetis microcontrollers.

The following table shows the translation between the Arm CPU modes and the Kinetis MCU power modes.

Arm CPU mode	MCU mode
Sleep	Wait
Deep Sleep	Stop

Accordingly, the Arm CPU documentation refers to sleep and deep sleep, while the Kinetis MCU documentation normally uses wait and stop.

In addition, Kinetis MCUs also augment Stop, Wait, and Run modes in a number of ways. The power management controller (PMC) contains a run and a stop mode regulator. Run regulation is used in normal run, wait and stop modes. Stop mode regulation is used during all very low power and low leakage modes. During stop mode regulation, the bus frequencies are limited in the very low power modes.

The SMC provides the user with multiple power options. The Very Low Power Run (VLPR) mode can drastically reduce run time power when maximum bus frequency is not required to handle the application needs. From Normal Run mode, the Run Mode (RUNM) field can be modified to change the MCU into VLPR mode when limited frequency is sufficient for the application. From VLPR mode, a corresponding wait (VLPW) and stop (VLPS) mode can be entered.

Depending on the needs of the user application, a variety of stop modes are available that allow the state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. Several registers are used to configure the various modes of operation for the device.

The following table describes the power modes available for the device.

Table 16-1. Power modes

Mode	Description
RUN	The MCU can be run at full speed and the internal supply is fully regulated, that is, in run regulation. This mode is also referred to as Normal Run mode.
HSRUN	The MCU can be run at a faster frequency compared with RUN mode and the internal supply is fully regulated. See the Power Management chapter for details about the maximum allowable frequencies.
WAIT	The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate. Run regulation is maintained.
STOP	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
VLPR	The core, system, bus, and flash clock maximum frequencies are restricted in this mode. See the Power Management chapter for details about the maximum allowable frequencies.
VLPW	The core clock is gated off. The system, bus, and flash clocks continue to operate, although their maximum frequency is restricted. See the Power Management chapter for details on the maximum allowable frequencies.
VLPS	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.

Table continues on the next page...

Table 16-1. Power modes (continued)

Mode	Description
VLLS3	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic. All system RAM contents are retained and I/O states are held. Internal logic states are not retained.
VLLS2	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and the system RAM3 partition. The system RAM2 partition can be optionally retained using STOPCTRL[RAM2PO].
VLLS1	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained.
VLLS0	The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained. The 1kHz LPO clock is disabled and the power on reset (POR) circuit can be optionally enabled using STOPCTRL[PORPO].

16.3 Memory map and register descriptions

Information about the registers related to the system mode controller can be found here.

Different SMC registers reset on different reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

NOTE

The SMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

NOTE

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

SMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_E000	Power Mode Protection register (SMC_PMPROT)	8	R/W	00h	16.3.1/266
4007_E001	Power Mode Control register (SMC_PMCTRL)	8	R/W	00h	16.3.2/267
4007_E002	Stop Control Register (SMC_STOPCTRL)	8	R/W	03h	16.3.3/269
4007_E003	Power Mode Status register (SMC_PMSTAT)	8	R	01h	16.3.4/270

16.3.1 Power Mode Protection register (SMC_PMPROT)

This register provides protection for entry into any low-power run or stop mode. The enabling of the low-power run or stop mode occurs by configuring the Power Mode Control register (PMCTRL).

The PMPROT register can be written only once after any system reset.

If the MCU is configured for a disallowed or reserved power mode, the MCU remains in its current power mode. For example, if the MCU is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and PMCTRL[RUNM] remains 00b, indicating the MCU is still in Normal Run mode.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 0h offset = 4007_E000h

Bit	7	6	5	4	3	2	1	0
Read	AHSRUN	0	AVLP	0	0	0	AVLLS	0
Write								
Reset	0	0	0	0	0	0	0	0

SMC_PMPROT field descriptions

Field	Description
7 AHSRUN	Allow High Speed Run mode Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter High Speed Run mode (HSRUN). 0 HSRUN is not allowed 1 HSRUN is allowed
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SMC_PMPROT field descriptions (continued)

Field	Description
5 AVLP	Allow Very-Low-Power Modes Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter any very-low-power mode (VLPR, VLPW, and VLPS). 0 VLPR, VLPW, and VLPS are not allowed. 1 VLPR, VLPW, and VLPS are allowed.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 AVLLS	Allow Very-Low-Leakage Stop Mode Provided the appropriate control bits are set up in PMCTRL, this write once bit allows the MCU to enter any very-low-leakage stop mode (VLLSx). 0 Any VLLSx mode is not allowed 1 Any VLLSx mode is allowed
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

16.3.2 Power Mode Control register (SMC_PMCTRL)

The PMCTRL register controls entry into low-power Run and Stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 1h offset = 4007_E001h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	RUNM		0	STOPA	STOPM		
Write								
Reset	0	0	0	0	0	0	0	0

SMC_PMCTRL field descriptions

Field	Description
7 Reserved	This field is reserved. This bit is reserved for future expansion. Software should write 0 to this bit to maintain compatibility.
6–5 RUNM	Run Mode Control When written, causes entry into the selected run mode. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. **NOTE:** RUNM may be set to VLPR only when PMSTAT=RUN. After being written to VLPR, RUNM should not be written back to RUN until PMSTAT=VLPR. **NOTE:** RUNM may be set to HSRUN only when PMSTAT=RUN. After being programmed to HSRUN, RUNM should not be programmed back to RUN until PMSTAT=HSRUN. Also, stop mode entry should not be attempted while RUNM=HSRUN or PMSTAT=HSRUN. 00 Normal Run mode (RUN) 01 Reserved 10 Very-Low-Power Run mode (VLPR) 11 High Speed Run mode (HSRUN)
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 STOPA	Stop Aborted When set, this read-only status bit indicates an interrupt or reset occured during the previous stop mode entry sequence, preventing the system from entering that mode. This field is cleared by hardware at the beginning of any stop mode entry sequence and is set if the sequence was aborted. 0 The previous stop mode entry was successful. 1 The previous stop mode entry was aborted.
STOPM	Stop Mode Control When written, controls entry into the selected stop mode when Sleep-Now or Sleep-On-Exit mode is entered with SLEEPDEEP=1 . Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register. **NOTE:** When set to VLLSx, the VLLSM field in the STOPCTRL register is used to further select the particular VLLS submode which will be entered. **NOTE:** When set to STOP, the PSTOPO bits in the STOPCTRL register can be used to select a Partial Stop mode if desired. 000 Normal Stop (STOP) 001 Reserved 010 Very-Low-Power Stop (VLPS) 011 Reserved 100 Very-Low-Leakage Stop (VLLSx) 101 Reserved 110 Reseved 111 Reserved

16.3.3 Stop Control Register (SMC_STOPCTRL)

The STOPCTRL register provides various control bits allowing the user to fine tune power consumption during the stop mode selected by the STOPM field.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 2h offset = 4007_E002h

Bit	7	6	5	4	3	2	1	0
Read Write	PSTOPO		PORPO	RAM2PO	LPOPO	VLLSM		
Reset	0	0	0	0	0	0	1	1

SMC_STOPCTRL field descriptions

Field	Description
7–6 PSTOPO	Partial Stop Option These bits control whether a Partial Stop mode is entered when STOPM=STOP. When entering a Partial Stop mode from RUN (or VLPR) mode, the PMC, MCG and flash remain fully powered, allowing the device to wakeup almost instantaneously at the expense of higher power consumption. In PSTOP2, only system clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated. 00 STOP - Normal Stop mode 01 PSTOP1 - Partial Stop with both system and bus clocks disabled 10 PSTOP2 - Partial Stop with system clock disabled and bus clock enabled 11 Reserved
5 PORPO	POR Power Option This bit controls whether the POR detect circuit is enabled in VLLS0 mode. 0 POR detect circuit is enabled in VLLS0 1 POR detect circuit is disabled in VLLS0
4 RAM2PO	RAM2 Power Option This bit controls powering of RAM partition 2 in VLLS2 mode. **NOTE:** See the device's Chip Configuration details for the size and location of RAM partition 2 0 RAM2 not powered in VLLS2 1 RAM2 powered in VLLS2
3 LPOPO	LPO Power Option Controls whether the 1 kHz LPO clock is enabled in VLLSx modes. **NOTE:** During VLLS0 mode, the LPO clock is disabled by hardware and this bit has no effect.

Table continues on the next page...

SMC_STOPCTRL field descriptions (continued)

Field	Description
	0 LPO clock is enabled in VLLSx 1 LPO clock is disabled in VLLSx
VLLSM	VLLS Mode Control This field controls which VLLS sub-mode to enter if STOPM = VLLSx. 000 VLLS0 001 VLLS1 010 VLLS2 011 VLLS3 100 Reserved 101 Reserved 110 Reserved 111 Reserved

16.3.4 Power Mode Status register (SMC_PMSTAT)

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 3h offset = 4007_E003h

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

SMC_PMSTAT field descriptions

Field	Description
PMSTAT	Power Mode Status **NOTE:** When debug is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** When a PSTOP mode is enabled, the PMSTAT will not update to STOP or VLPS 0000_0001 Current power mode is RUN. 0000_0010 Current power mode is STOP. 0000_0100 Current power mode is VLPR.

SMC_PMSTAT field descriptions (continued)

Field	Description
	0000_1000 Current power mode is VLPW. 0001_0000 Current power mode is VLPS. 0010_0000 Reserved 0100_0000 Current power mode is VLLS. 1000_0000 Current power mode is HSRUN

16.4 Functional description

16.4.1 Power mode transitions

The following figure shows the power mode state transitions available on the chip. Any reset always brings the MCU back to the normal RUN state.

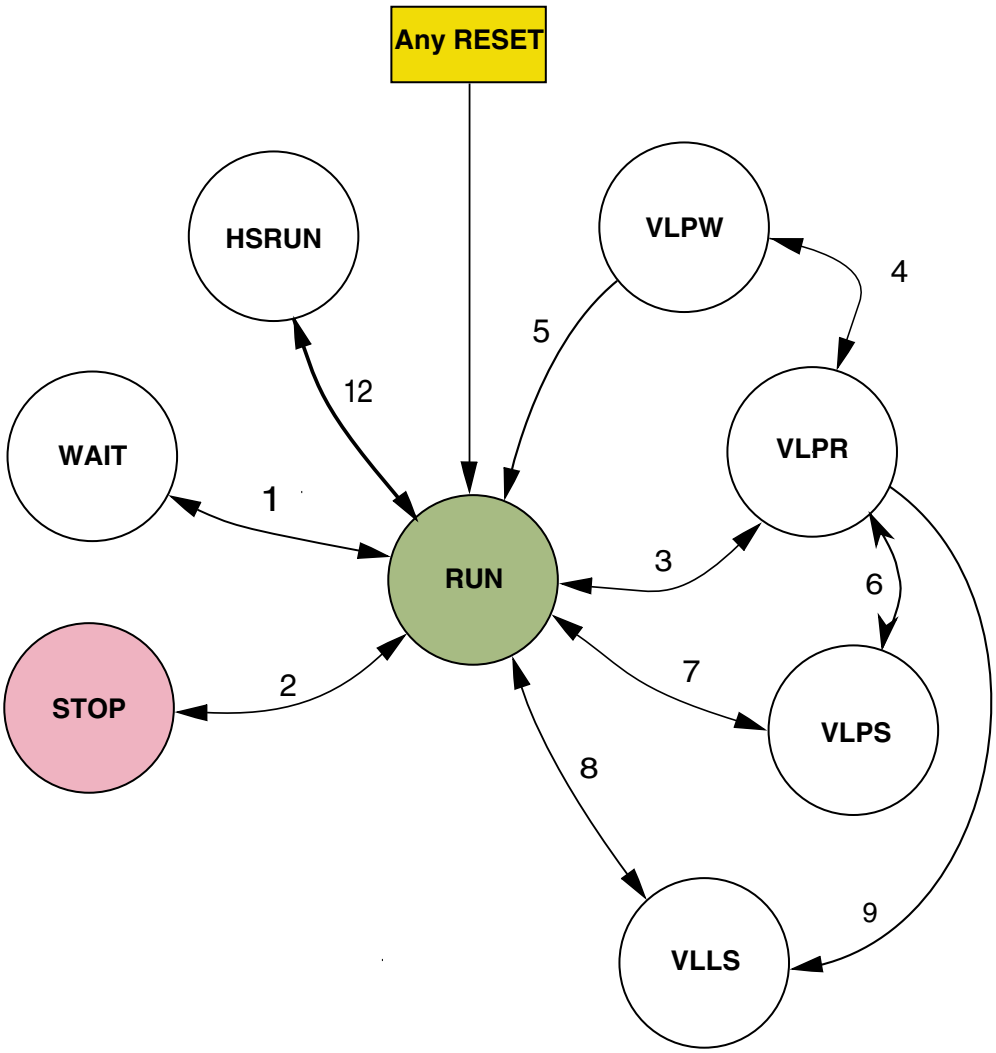


Figure 16-1. Power mode state diagram

The following table defines triggers for the various state transitions shown in the previous figure.

Table 16-2. Power mode transition triggers

Transition #	From	To	Trigger conditions
1	RUN	WAIT	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, controlled in System Control Register in Arm core. See note.[1]
	WAIT	RUN	Interrupt or Reset
2	RUN	STOP	2

Table continues on the next page...

Table 16-2. Power mode transition triggers (continued)

Transition #	From	To	Trigger conditions
			Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in Arm core. See note.[1]
	STOP	RUN	Interrupt or Reset
3	RUN	VLPR	The core, system, bus and flash clock frequencies and MCG clocking mode are restricted in this mode. See the Power Management chapter for the maximum allowable frequencies and MCG modes supported.
	VLPR	RUN	Set or Reset.
4	VLPR	VLPW	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, which is controlled in System Control Register in Arm core. See note.[1]
	VLPW	VLPR	Interrupt
5	VLPW	RUN	Reset
6	VLPR	VLPS	[3] or 010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in Arm core. See note.[1]
	VLPS	VLPR	Interrupt **NOTE:** If VLPS was entered directly from RUN (transition #7), hardware forces exit back to RUN and does not allow a transition to VLPR.
7	RUN	VLPS	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in Arm core. See note.[1]
	VLPS	RUN	Interrupt and VLPS mode was entered directly from RUN or Reset
8	RUN	VLLSx	STOPCTRL[LLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in Arm core.
	VLLSx	RUN	Wakeup from enabled LLWU input source or RESET pin
9	VLPR	VLLSx	STOPCTRL[LLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in Arm core.
12	RUN	HSRUN	
	HSRUN	RUN	Reset

1. If debug is enabled, the core clock remains to support debug.
2. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of STOP

3. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=00, then VLPS mode is entered instead of STOP. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of VLPS

16.4.2 Power mode entry/exit sequencing

When entering or exiting low-power modes, the system must conform to an orderly sequence to manage transitions safely.

The SMC manages the system's entry into and exit from all power modes. This diagram illustrates the connections of the SMC with other system components in the chip that are necessary to sequence the system through all power modes.

16.4.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS, VLLSx) is initiated by a CPU executing the WFI instruction. After the instruction is executed, the following sequence occurs:

1. The CPU clock is gated off immediately.
2. Requests are made to all non-CPU bus masters to enter Stop mode.
3. After all masters have acknowledged they are ready to enter Stop mode, requests are made to all bus slaves to enter Stop mode.
4. After all slaves have acknowledged they are ready to enter Stop mode, all system and bus clocks are gated off.
5. Clock generators are disabled in the MCG.
6. The on-chip regulator in the PMC and internal power switches are configured to meet the power consumption goals for the targeted low-power mode.

16.4.2.2 Stop mode exit sequence

Exit from a low-power stop mode is initiated either by a reset or an interrupt event. The following sequence then executes to restore the system to a run mode (RUN or VLPR):

1. The on-chip regulator in the PMC and internal power switches are restored.
2. Clock generators are enabled in the MCG.
3. System and bus clocks are enabled to all masters and slaves.
4. The CPU clock is enabled and the CPU begins servicing the reset or interrupt that initiated the exit from the low-power stop mode.

16.4.2.3 Aborted stop mode entry

If an interrupt or a reset occurs during a stop entry sequence, the SMC can abort the transition early and return to RUN mode without completely entering the stop mode. An aborted entry is possible only if the reset or interrupt occurs before the PMC begins the transition to stop mode regulation. After this point, the interrupt or reset is ignored until the PMC has completed its transition to stop mode regulation. When an aborted stop mode entry sequence occurs, SMC_PMCTRL[STOPA] is set to 1.

16.4.2.4 Transition to wait modes

For wait modes (WAIT and VLPW), the CPU clock is gated off while all other clocking continues, as in RUN and VLPR mode operation. Some modules that support stop-in-wait functionality have their clocks disabled in these configurations.

16.4.2.5 Transition from stop modes to Debug mode

The debugger module supports a transition from STOP, WAIT, VLPS, and VLPW back to a Halted state when the debugger has been enabled. As part of this transition, system clocking is re-established and is equivalent to the normal RUN and VLPR mode clocking configuration.

16.4.3 Run modes

The run modes supported by this device can be found here.

- Run (RUN)
- Very Low-Power Run (VLPR)
- High Speed Run (HSRUN)

16.4.3.1 RUN mode

This is the normal operating mode for the device.

This mode is selected after any reset. When the Arm processor exits reset, it sets up the stack, program counter (PC), and link register (LR):

- The processor reads the start SP (SP_main) from vector-table offset 0x000
- The processor reads the start PC from vector-table offset 0x004
- LR is set to 0xFFFF_FFFF.

To reduce power in this mode, disable the clocks to unused modules.

16.4.3.2 Very-Low Power Run (VLPR) mode

In VLPR mode, the on-chip voltage regulator is put into a stop mode regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the SIM's registers.

Before entering this mode, the following conditions must be met:

- The MCG must be configured in a mode which is supported during VLPR. See the Power Management details for information about these MCG modes.
- All clock monitors in the MCG must be disabled.
- The maximum frequencies of the system, bus, flash, and core are restricted. See the Power Management details about which frequencies are supported.
- Mode protection must be set to allow VLP modes, that is, PMPROT[AVLP] is 1.
- PMCTRL[RUNM] must be set to 10b to enter VLPR.
- Flash programming/erasing is not allowed.

NOTE

Do not increase the clock frequency while in VLPR mode, because the regulator is slow in responding and cannot manage fast load transitions. In addition, do not modify the clock source in the MCG module or any clock divider registers. Module clock enables in the SIM can be set, but not cleared.

To reenter Normal Run mode, clear PMCTRL[RUNM]. PMSTAT is a read-only status register that can be used to determine when the system has completed an exit to RUN mode. When PMSTAT=RUN, the system is in run regulation and the MCU can run at full speed in any clock mode. If a higher execution frequency is desired, poll PMSTAT until it is set to RUN when returning from VLPR mode.

Any reset always causes an exit from VLPR and returns the device to RUN mode after the MCU exits its reset flow.

16.4.3.3 High Speed Run (HSRUN) mode

In HSRUN mode, the on-chip voltage regulator remains in a run regulation state, but with a slightly elevated voltage output. In this state, the MCU is able to operate at a faster frequency compared to normal RUN mode. For the maximum allowable frequencies, see the Power Management chapter.

While in this mode, the following restrictions must be adhered to:

- The maximum allowable change in frequency of the system, bus, flash or core clocks is restricted to 2x (double the frequency).
- Before exiting HSRUN mode, clock frequencies should be reduced back down to those acceptable in RUN mode.
- Stop mode entry is not supported from HSRUN.
- Modifications to clock gating control bits are prohibited.
- Flash programming/erasing is not allowed.

To enter HSRUN mode, set PMPROT[AHSRUN] to allow HSRUN and then set PMCTRL[RUNM]=HSRUN. Before increasing clock frequencies, the PMSTAT register should be polled to determine when the system has completed entry into HSRUN mode. To reenter normal RUN mode, clear PMCTRL[RUNM]. Any reset also clears PMCTRL[RUNM] and causes the system to exit to normal RUN mode after the MCU exits its reset flow.

16.4.4 Wait modes

This device contains two different wait modes which are listed here.

- Wait
- Very-Low Power Wait (VLPW)

16.4.4.1 WAIT mode

WAIT mode is entered when the Arm core enters the Sleep-Now or Sleep-On-Exit modes while SLEEPDEEP is cleared. The Arm CPU enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled.

When an interrupt request occurs, the CPU exits WAIT mode and resumes processing in RUN mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset causes an exit from WAIT mode, returning the device to normal RUN mode.

16.4.4.2 Very-Low-Power Wait (VLPW) mode

VLPW mode is entered by entering the Sleep-Now or Sleep-On-Exit mode while SLEEPDEEP is cleared and the device is in VLPR mode.

In VLPW, the on-chip voltage regulator remains in its stop regulation state. In this state, the regulator is designed to supply enough current to the device at a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules.

VLPR mode restrictions also apply to VLPW.

When an interrupt from VLPW occurs, the device returns to VLPR mode to execute the interrupt service routine.

A system reset causes an exit from VLPW mode, returning the device to normal RUN mode.

16.4.5 Stop modes

This device contains a variety of stop modes to meet your application needs.

The stop modes range from:

- a stopped CPU, with all I/O, logic, and memory states retained, and certain asynchronous mode peripherals operating

to:

- a powered down CPU, with only I/O and a small register file retained, very few asynchronous mode peripherals operating, while the remainder of the MCU is powered down.

The choice of stop mode depends upon the user's application, and how power usage and state retention versus functional needs and recovery time may be traded off.

The various stop modes are selected by setting the appropriate fields in PMPROT and PMCTRL. The selected stop mode is entered during the sleep-now or sleep-on-exit entry with the SLEEPDEEP bit set in the System Control Register in the Arm core.

The available stop modes are:

- Normal Stop (STOP)
- Very-Low Power Stop (VLPS)
- Very-Low-Leakage Stop (VLLSx)

16.4.5.1 STOP mode

STOP mode is entered via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the Arm core.

The MCG module can be configured to leave the reference clocks running.

A module capable of providing an asynchronous interrupt to the device takes the device out of STOP mode and returns the device to normal RUN mode. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode. When an interrupt request occurs, the CPU exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from STOP mode, returning the device to normal RUN mode via an MCU reset.

16.4.5.2 Very-Low-Power Stop (VLPS) mode

The two ways in which VLPS mode can be entered are listed here.

- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the Arm core while the MCU is in VLPR mode and PMCTRL[STOPM] = 010 or 000.
- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the Arm core while the MCU is in normal RUN mode and PMCTRL[STOPM] = 010. When VLPS is entered directly from RUN mode, exit to VLPR is disabled by hardware and the system will always exit back to RUN.

In VLPS, the on-chip voltage regulator remains in its stop regulation state as in VLPR.

A module capable of providing an asynchronous interrupt to the device takes the device out of VLPS and returns the device to VLPR mode.

A system reset will also cause a VLPS exit, returning the device to normal RUN mode.

16.4.5.3 Very-Low-Leakage Stop (VLLSx) modes

This device contains these very low leakage modes:

- VLLS3
- VLLS2
- VLLS1
- VLLS0

VLLSx is often used in this document to refer to all of these modes.

All VLLSx modes can be entered from normal RUN or VLPR modes.

The MCU enters the configured VLLS mode if:

- In Sleep-Now or Sleep-On-Exit mode, the SLEEPDEEP bit is set in the System Control Register in the Arm core, and
- The device is configured as shown in Table 16-2.

In VLLS, the on-chip voltage regulator is in its stop-regulation state while most digital logic is powered off.

Before entering VLLS mode, the user should configure the Low-Leakage Wake-up (LLWU) module to enable the desired wakeup sources. The available wake-up sources in VLLS are detailed in the chip configuration details for this device.

After wakeup from VLLS, the device returns to normal RUN mode with a pending LLWU interrupt. In the LLWU interrupt service routine (ISR), the user can poll the LLWU module wake-up flags to determine the source of the wake-up.

When entering VLLS, each I/O pin is latched as configured before executing VLLS. Because all digital logic in the MCU is powered off, all port and peripheral data is lost during VLLS. This information must be restored before PMC_REGSC[ACKISO] is set.

An asserted $\overline{\text{RESET}}$ pin will cause an exit from any VLLS mode, returning the device to normal RUN mode. When exiting VLLS via the $\overline{\text{RESET}}$ pin, RCM_SRS[PIN] and RCM_SRS[WAKEUP] are set.

16.4.6 Debug in low power modes

When the MCU is secure, the device disables/limits debugger operation. When the MCU is unsecure, the Arm debugger can assert two power-up request signals:

- System power up, via SYSPWR in the Debug Port Control/Stat register
- Debug power up, via CDBGPWRUPREQ in the Debug Port Control/Stat register

When asserted while in RUN, WAIT, VLPR, or VLPW the mode controller drives a corresponding acknowledge for each signal, that is, both CDBGPWRUPACK and CSYSPWRUPACK. When both requests are asserted, the mode controller handles attempts to enter STOP and VLPS by entering an emulated stop state. In this emulated stop state:

- the regulator is in run regulation,
- the MCG-generated clock source is enabled,
- all system clocks, except the core clock, are disabled,
- the debug module has access to core registers, and
- access to the on-chip peripherals is blocked.

No debug is available while the MCU is in LLS or VLLS modes. LLS is a state-retention mode and all debug operation can continue after waking from LLS, even in cases where system wakeup is due to a system reset event.

Entering into a VLLS mode causes all of the debug controls and settings to be powered off. To give time to the debugger to sync with the MCU, the MDM AP Control Register includes a Very-Low-Leakage Debug Request (VLLDBGREQ) bit that is set to configure the Reset Controller logic to hold the system in reset after the next recovery from a VLLS mode. This bit allows the debugger time to reinitialize the debug module before the debug session continues.

The MDM AP Control Register also includes a Very Low Leakage Debug Acknowledge (VLLDBGACK) bit that is set to release the Arm core being held in reset following a VLLS recovery. The debugger reinitializes all debug IP, and then asserts the VLLDBGACK control bit to allow the RCM to release the Arm core from reset and allow CPU operation to begin.

The VLLDBGACK bit is cleared by the debugger (or can be left set as is) or clears automatically due to the reset generated as part of the next VLLS recovery.

Chapter 17
Miscellaneous Control Module (MCM)

17.1 Introduction

The Miscellaneous Control Module (MCM) provides miscellaneous control functions and contains Cortex-M4 local memory descriptors.

17.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision
- Control and counting logic for embedded trace buffer (ETB) almost full
- Local memory descriptors (ITCM, D0TCM, D1TCM, ICACHE, and DCACHE)

17.2 Memory map/register descriptions

The memory map and register descriptions below describe the registers using byte addresses.

MCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_0008	Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)	16	R	000Fh	17.2.1/284
E008_000A	Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)	16	R	0007h	17.2.2/284
E008_000C	Control Register (MCM_CR)	32	R/W	0000_00F0h	17.2.3/285
E008_0010	Interrupt Status and Control Register (MCM_ISCR)	32	R	0002_0000h	17.2.4/287
E008_0040	Compute Operation Control Register (MCM_CPO)	32	R/W	0000_0000h	17.2.5/289

17.2.1 Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)

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

Address: E008_0000h base + 8h offset = E008_0008h

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

MCM_PLASC field descriptions

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

17.2.2 Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)

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

Address: E008_0000h base + Ah offset = E008_000Ah

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

MCM_PLAMC field descriptions

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

Table continues on the next page...

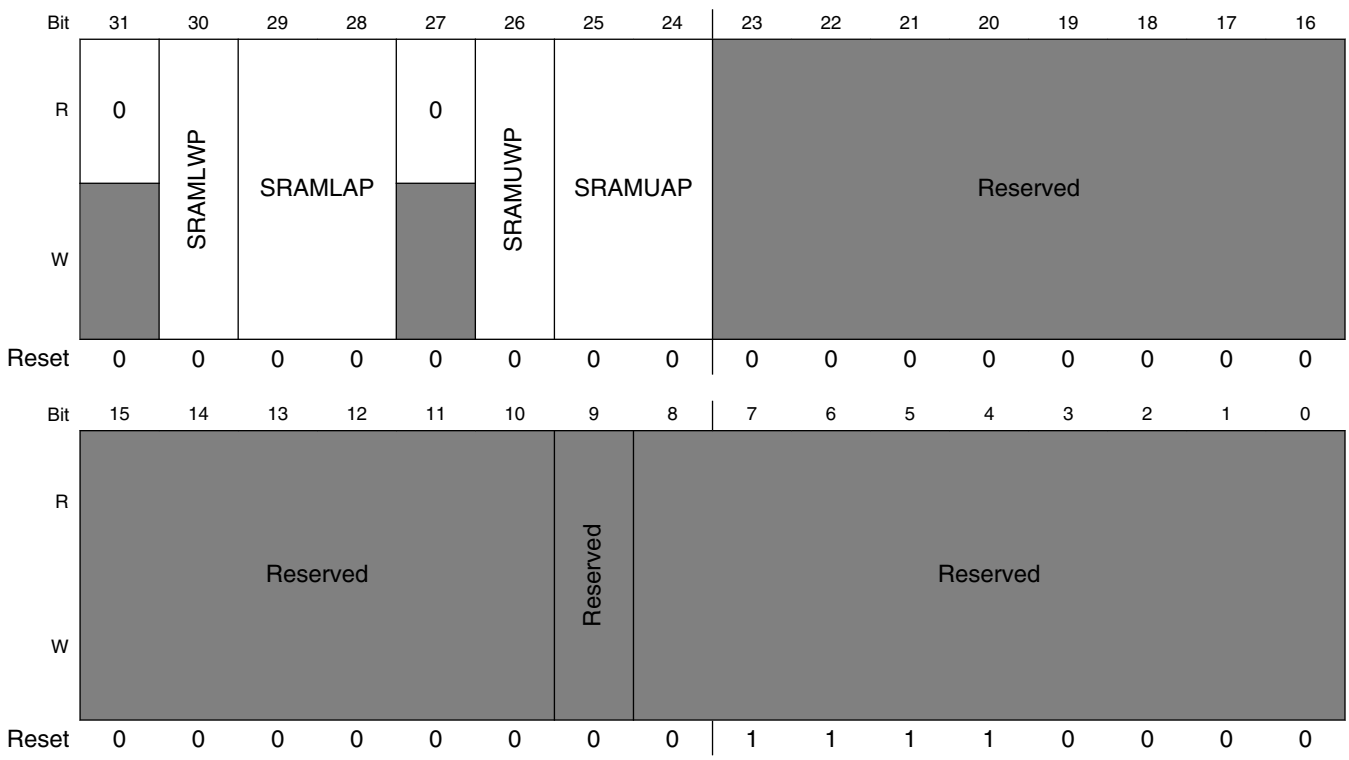
MCM_PLAMC field descriptions (continued)

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

17.2.3 Control Register (MCM_CR)

CR defines the arbitration and protection schemes for the two system RAM arrays.

Address: E008_0000h base + Ch offset = E008_000Ch



MCM_CR field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SRAMLWP	SRAM_L Write Protect When this bit is set, writes to SRAM_L array generates a bus error.
29–28 SRAMLAP	SRAM_L arbitration priority Defines the arbitration scheme and priority for the processor and SRAM backdoor accesses to the SRAM_L array.

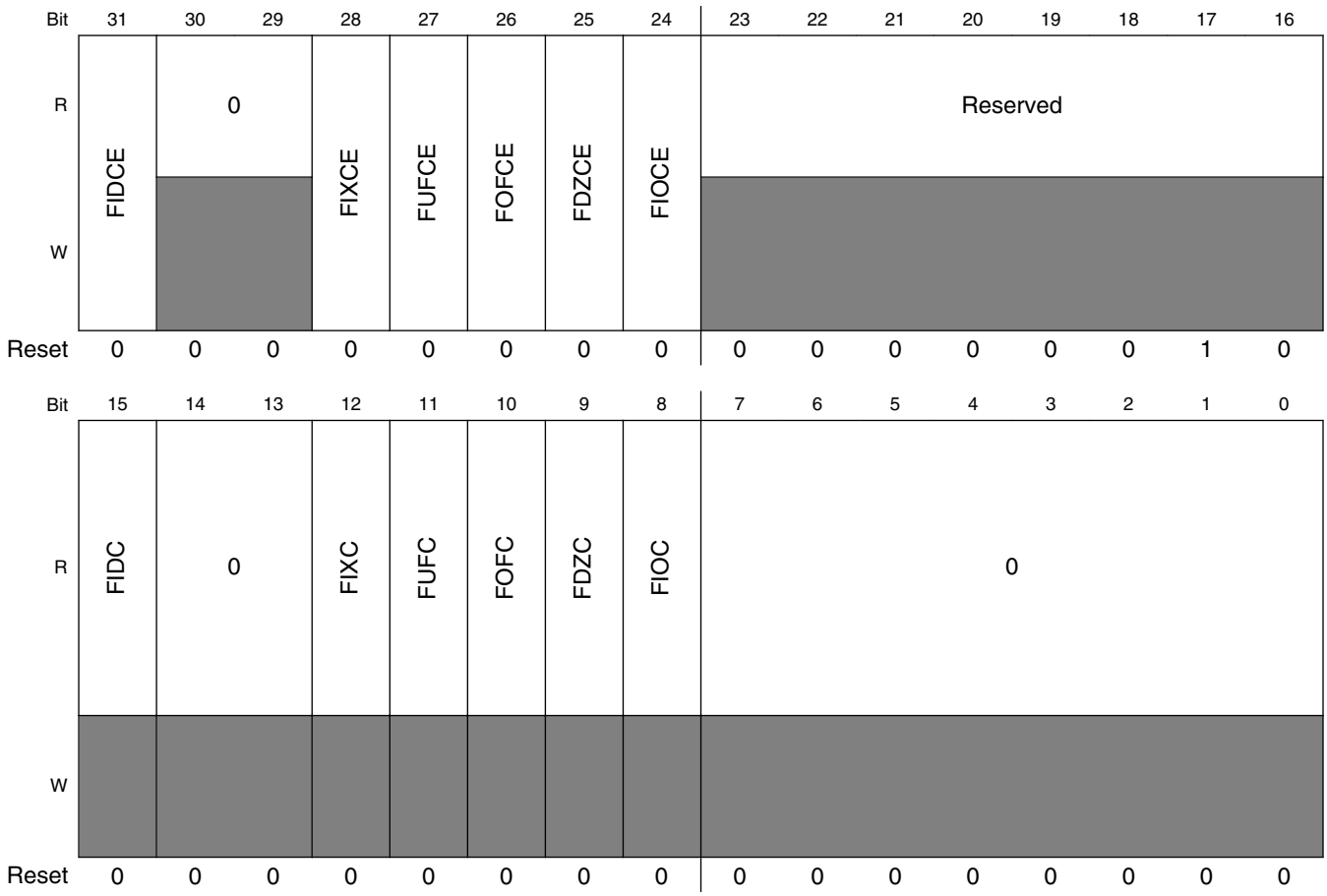
Table continues on the next page...

MCM_CR field descriptions (continued)

Field	Description
	00 Round robin 01 Special round robin (favors SRAM backoor accesses over the processor) 10 Fixed priority. Processor has highest, backdoor has lowest 11 Fixed priority. Backdoor has highest, processor has lowest
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SRAMUWP	SRAM_U write protect When this bit is set, writes to SRAM_U array generates a bus error.
25–24 SRAMUAP	SRAM_U arbitration priority Defines the arbitration scheme and priority for the processor and SRAM backdoor accesses to the SRAM_U array. 00 Round robin 01 Special round robin (favors SRAM backoor accesses over the processor) 10 Fixed priority. Processor has highest, backdoor has lowest 11 Fixed priority. Backdoor has highest, processor has lowest
23–10 Reserved	This field is reserved.
9 Reserved	This field is reserved.
Reserved	This field is reserved.

17.2.4 Interrupt Status and Control Register (MCM_ISCR)

Address: E008_0000h base + 10h offset = E008_0010h



MCM_ISCR field descriptions

Field	Description
31 FIDCE	FPU input denormal interrupt enable 0 Disable interrupt 1 Enable interrupt
30–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 FIXCE	FPU inexact interrupt enable 0 Disable interrupt 1 Enable interrupt
27 FUFCE	FPU underflow interrupt enable 0 Disable interrupt 1 Enable interrupt

Table continues on the next page...

MCM_ISCR field descriptions (continued)

Field	Description
26 FOFCE	FPU overflow interrupt enable 0 Disable interrupt 1 Enable interrupt
25 FDZCE	FPU divide-by-zero interrupt enable 0 Disable interrupt 1 Enable interrupt
24 FIOCE	FPU invalid operation interrupt enable 0 Disable interrupt 1 Enable interrupt
23–16 Reserved	This field is reserved.
15 FIDC	FPU input denormal interrupt status This read-only bit is a copy of the core's FPSCR[IDC] bit and signals input denormalized number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IDC] bit. 0 No interrupt 1 Interrupt occurred
14–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 FIXC	FPU inexact interrupt status This read-only bit is a copy of the core's FPSCR[IXC] bit and signals an inexact number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IXC] bit. 0 No interrupt 1 Interrupt occurred
11 FUFC	FPU underflow interrupt status This read-only bit is a copy of the core's FPSCR[UFC] bit and signals an underflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[UFC] bit. 0 No interrupt 1 Interrupt occurred
10 FOFC	FPU overflow interrupt status This read-only bit is a copy of the core's FPSCR[OFC] bit and signals an overflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[OFC] bit. 0 No interrupt 1 Interrupt occurred
9 FDZC	FPU divide-by-zero interrupt status This read-only bit is a copy of the core's FPSCR[DZC] bit and signals a divide by zero has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[DZC] bit. 0 No interrupt 1 Interrupt occurred

Table continues on the next page...

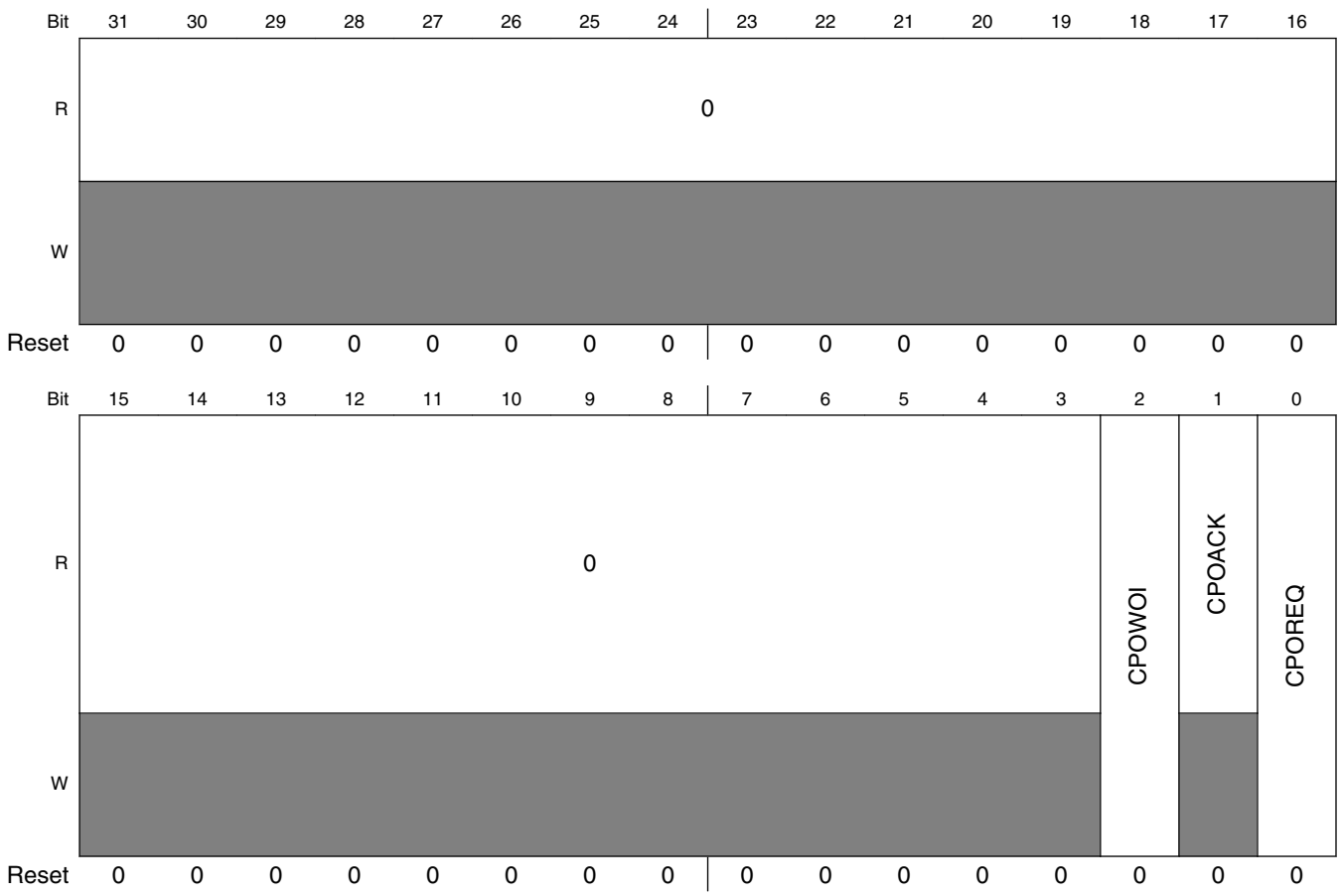
MCM_ISCR field descriptions (continued)

Field	Description
8 FIOC	FPU invalid operation interrupt status This read-only bit is a copy of the core's FPSCR[IOC] bit and signals an illegal operation has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IOC] bit. 0 No interrupt 1 Interrupt occurred
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

17.2.5 Compute Operation Control Register (MCM_CPO)

This register controls the Compute Operation.

Address: E008_0000h base + 40h offset = E008_0040h



MCM_CPO field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CPOWOI	Compute Operation wakeup on interrupt 0 No effect. 1 When set, the CPOREQ is cleared on any interrupt or exception vector fetch.
1 CPOACK	Compute Operation acknowledge 0 Compute operation entry has not completed or compute operation exit has completed. 1 Compute operation entry has completed or compute operation exit has not completed.
0 CPOREQ	Compute Operation request This bit is auto-cleared by vector fetching if CPOWOI = 1. 0 Request is cleared. 1 Request Compute Operation.

17.3 Functional description

This section describes the functional description of MCM module.

17.3.1 Interrupts

The MCM generates two interrupt requests:

- Non-maskable interrupt
- Normal interrupt

17.3.1.1 Non-maskable interrupt

The MCM's NMI is generated if:

- ISCR[ETBN] is set, when
  - The ETB counter is enabled, ETBCC[CNTEN] = 1
  - The ETB count expires
  - The response to counter expiration is an NMI, MCM_ETBCC[RSPT] = 10

17.3.1.2 Normal interrupt

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

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

17.4 Functional description

This section describes the functional description of MCM module.

17.4.1 Interrupts

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

- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC)
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC)
- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC)
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC)
- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC)
- FPU invalid operation interrupt is enabled (FDZCE) and an invalid occurs (FDZC)

17.4.1.1 Determining source of the interrupt

To determine the exact source of the interrupt qualify the interrupt status flags with the corresponding interrupt enable bits.

1. From MCM_ISCR[31:16] && MCM_ISCR[15:0]
2. Search the result for asserted flags, which indicate the exact interrupt sources

Chapter 18
Power Management Controller (PMC)

18.1 Introduction

The power management controller (PMC) contains the internal voltage regulator, power on reset (POR), and low voltage detect system (LVD) for the VDD domain.

See AN4503: Power Management for Kinetis MCUs for further details on using the PMC.

18.2 Features

A list of included PMC features can be found here.

- Internal voltage regulator
- Active POR providing brown-out detect
- Low-voltage detect (LVD) on VDD supporting two low-voltage trip points with four warning levels per trip point

18.3 Low-voltage detect (LVD) system

This device includes a system to guard against low-voltage conditions on the VDD supply. When the VDD supply falls below a specific trip point, the LVD circuit puts the device into a reset state, preventing the device from attempting to operate below its operating voltage range.

The system is comprised of a power-on reset (POR) circuit and a LVD circuit with a user-selectable trip voltage: high (V_{LVDH}) or low (V_{LVDL}). The trip voltage is selected by LVDSC1[LVDV]. The LVD is disabled upon entering VLPx and VLLSx modes.

Two flags are available to indicate the status of the low-voltage detect system:

- The Low Voltage Detect Flag in the Low Voltage Status and Control 1 Register (LVDSC1[LVDF]) operates in a level sensitive manner. LVDSC1[LVDF] is set when the supply voltage falls below the selected trip point (VLVD). LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC1[LVDF] remains set.
- The Low Voltage Warning Flag (LVWF) in the Low Voltage Status and Control 2 Register (LVDSC2[LVWF]) operates in a level sensitive manner. LVDSC2[LVWF] is set when the supply voltage falls below the selected monitor trip point (VLVW). LVDSC2[LVWF] is cleared by writing one to LVDSC2[LVWACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC2[LVWF] remains set.

18.3.1 LVD reset operation

By setting LVDSC1[LVDRE], the LVD generates a reset upon detection of a low-voltage condition. The low-voltage detection threshold is determined by LVDSC1[LVDV]. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above this threshold. The LVD field in the SRS register of the RCM module (RCM_SRS[LVD]) is set following an LVD or power-on reset.

18.3.2 LVD interrupt operation

By configuring the LVD circuit for interrupt operation (LVDSC1[LVDIE] set and LVDSC1[LVDRE] clear), LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK].

18.3.3 Low-voltage warning (LVW) interrupt operation

The LVD system that monitors the VDD supply contains a Low-Voltage Warning Flag (LVWF) in the Low Voltage Detect Status and Control 2 Register to indicate that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt, which is enabled by setting LVDSC2[LVWIE]. If enabled, an LVW interrupt request occurs when LVDSC2[LVWF] is set. LVDSC2[LVWF] is cleared by writing 1 to LVDSC2[LVWACK].

LVDSC2[LVWV] selects one of the four trip voltages:

- Highest: V_{LVW4}
- Two mid-levels: V_{LVW3} and V_{LVW2}
- Lowest: V_{LVW1}

18.4 I/O retention

When in VLLS modes, the I/O states are held on a wake-up event (with the exception of wake-up by reset event) until the wake-up has been acknowledged via a write to REGSC[ACKISO]. In the case of VLLS exit via a RESET pin, the I/O are released and default to their reset state. In this case, no write to REGSC[ACKISO] is needed.

18.5 Memory map and register descriptions

Details about the PMC registers can be found here.

NOTE

Different portions of PMC registers are reset only by particular reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

The PMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

PMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_D000	Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)	8	R/W	10h	18.5.1/296
4007_D001	Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)	8	R/W	00h	18.5.2/297
4007_D002	Regulator Status And Control register (PMC_REGSC)	8	R/W	24h	18.5.3/298

18.5.1 Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)

This register contains status and control bits to support the low voltage detect function. This register should be written during the reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC1 settings. To protect systems that must have LVD always on, configure the Power Mode Protection (PMPROT) register of the SMC module (SMC_PMPROT) to disallow any very low power or low leakage modes from being enabled.

See the device's data sheet for the exact LVD trip voltages.

NOTE

The LVDV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + 0h offset = 4007_D000h

Bit	7	6	5	4	3	2	1	0
Read	LVDF	0	LVDIE	LVDRE	0		LVDV	
Write		LVDACK						
Reset	0	0	0	1	0	0	0	0

PMC_LVDSC1 field descriptions

Field	Description
7 LVDF	Low-Voltage Detect Flag This read-only status field indicates a low-voltage detect event. 0 Low-voltage event not detected 1 Low-voltage event detected
6 LVDACK	Low-Voltage Detect Acknowledge This write-only field is used to acknowledge low voltage detection errors. Write 1 to clear LVDF. Reads always return 0.
5 LVDIE	Low-Voltage Detect Interrupt Enable Enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1

Table continues on the next page...

PMC_LVDSC1 field descriptions (continued)

Field	Description
4 LVDRE	Low-Voltage Detect Reset Enable This write-once bit enables LVDF events to generate a hardware reset. Additional writes are ignored. 0 LVDF does not generate hardware resets 1 Force an MCU reset when LVDF = 1
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVDV	Low-Voltage Detect Voltage Select Selects the LVD trip point voltage (V_{LVD}). 00 Low trip point selected ($V_{LVD} = V_{LVDL}$) 01 High trip point selected ($V_{LVD} = V_{LVDH}$) 10 Reserved 11 Reserved

18.5.2 Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)

This register contains status and control bits to support the low voltage warning function.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC2 settings.

See the device's data sheet for the exact LVD trip voltages.

NOTE

The LVW trip voltages depend on LVWV and LVDV.

NOTE

LVWV is reset solely on a POR Only event. The other fields of the register are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007_D000h base + 1h offset = 4007_D001h

Bit	7	6	5	4	3	2	1	0
Read	LVWF	0	LVWIE	0			LVWV	
Write		LVWACK						
Reset	0	0	0	0	0	0	0	0

PMC_LVDSC2 field descriptions

Field	Description
7 LVWF	Low-Voltage Warning Flag This read-only status field indicates a low-voltage warning event. LVWF is set when V_{Supply} transitions below the trip point, or after reset and V_{Supply} is already below V_{LVW}. LVWF may be 1 after power-on reset, therefore, to use LVW interrupt function, before enabling LVWIE, LVWF must be cleared by writing LVWACK first. 0 Low-voltage warning event not detected 1 Low-voltage warning event detected
6 LVWACK	Low-Voltage Warning Acknowledge This write-only field is used to acknowledge low voltage warning errors. Write 1 to clear LVWF. Reads always return 0.
5 LVWIE	Low-Voltage Warning Interrupt Enable Enables hardware interrupt requests for LVWF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVWF = 1
4–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVWV	Low-Voltage Warning Voltage Select Selects the LVW trip point voltage (V_{LVW}). The actual voltage for the warning depends on LVDSC1[LVDV]. 00 Low trip point selected ($V_{LVW} = V_{LVW1}$) 01 Mid 1 trip point selected ($V_{LVW} = V_{LVW2}$) 10 Mid 2 trip point selected ($V_{LVW} = V_{LVW3}$) 11 High trip point selected ($V_{LVW} = V_{LVW4}$)

18.5.3 Regulator Status And Control register (PMC_REGSC)

The PMC contains an internal voltage regulator. The voltage regulator design uses a bandgap reference that is also available through a buffer as input to certain internal peripherals, such as the CMP and ADC. The internal regulator provides a status bit (REGONS) indicating the regulator is in run regulation.

NOTE

This register is reset on Chip Reset Not VLLS and by reset types that trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007_D000h base + 2h offset = 4007_D002h

Bit	7	6	5	4	3	2	1	0
Read	0	0	Reserved	BGEN	ACKISO	REGONS	BGBDS	BGBE
Write					w1c			
Reset	0	0	1	0	0	1	0	0

PMC_REGSC field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved.
4 BGEN	Bandgap Enable In VLPx Operation BGEN controls whether the bandgap is enabled in lower power modes of operation (VLPx, and VLLSx). When on-chip peripherals require the bandgap voltage reference in low power modes of operation, set BGEN to continue to enable the bandgap operation. **NOTE:** When the bandgap voltage reference is not needed in low power modes, clear BGEN to avoid excess power consumption. 0 Bandgap voltage reference is disabled in VLPx , and VLLSx modes. 1 Bandgap voltage reference is enabled in VLPx , and VLLSx modes.
3 ACKISO	Acknowledge Isolation Reading this field indicates whether certain peripherals and the I/O pads are in a latched state as a result of having been in a VLLS mode. Writing 1 to this field when it is set releases the I/O pads and certain peripherals to their normal run mode state. **NOTE:** After recovering from a VLLS mode, user should restore chip configuration before clearing ACKISO. In particular, pin configuration for enabled LLWU wakeup pins should be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared. 0 Peripherals and I/O pads are in normal run state. 1 Certain peripherals and I/O pads are in an isolated and latched state.
2 REGONS	Regulator In Run Regulation Status This read-only field provides the current status of the internal voltage regulator. 0 Regulator is in stop regulation or in transition to/from it 1 Regulator is in run regulation
1 BGBDS	Bandgap Buffer Drive Select Selects the bandgap buffer drive. **NOTE:** If the bandgap buffer is disabled (BGBE is clear), clear BGBDS. 0 Low drive 1 High drive
0 BGBE	Bandgap Buffer Enable Enables the bandgap buffer.

Table continues on the next page...

PMC_REGSC field descriptions (continued)

Field	Description
	0 Bandgap buffer not enabled 1 Bandgap buffer enabled

Chapter 19
Low-Leakage Wakeup Unit (LLWU)

19.1 Chip-specific LLWU information

This chip uses the following internal peripheral and external pin inputs as wakeup sources to the LLWU module:

- LLWU_P0-P16, P19-P21 are external pin inputs. Any digital function multiplexed on the pin can be selected as the wakeup source. See the chip's signal multiplexing table for the digital signal options.
- LLWU_M0IF-M7IF are connections to the internal peripheral interrupt flags.

NOTE

$\overline{\text{RESET}}$ is also a wakeup source, in addition to LLWU pins, and can be diagnosed by reading the System Reset Status Registers within the RCM.

Table 19-1. Wakeup sources for LLWU inputs

Input	Wakeup source		Input	Wakeup source
LLWU_P0	PTE1/LLWU_P0 pin		LLWU_P12	PTD0/LLWU_P12 pin
LLWU_P1	PTE2/LLWU_P1 pin		LLWU_P13	PTD2/LLWU_P13 pin
LLWU_P2	PTE4/LLWU_P2 pin		LLWU_P14	PTD4/LLWU_P14 pin
LLWU_P3	PTA4/LLWU_P3 pin		LLWU_P15	PTD6/LLWU_P15 pin
LLWU_P4	PTA13/LLWU_P4 pin		LLWU_M0IF	LPTMR[1]
LLWU_P5	PTB0/LLWU_P5 pin		LLWU_M1IF	CMP0
LLWU_P6	PTC1/LLWU_P6 pin		LLWU_M2IF	CMP1
LLWU_P7	PTC3/LLWU_P7 pin		LLWU_M3IF	CMP2/3
LLWU_P8	PTC4/LLWU_P8 pin		LLWU_M4IF	
LLWU_P9	PTC5/LLWU_P9 pin		LLWU_M5IF	
LLWU_P10	PTC6/LLWU_P10 pin		LLWU_M6IF	Reserved
LLWU_P11	PTC11/LLWU_P11 pin		LLWU_M7IF	
LLWU_P16	PTE6/LLWU_P16 pin		LLWU_P20	PTE18/LLWU_P20 pin
LLWU_P19	PTE17/LLWU_P19 pin		LLWU_P21	PTE25/LLWU_P21 pin

1. Requires the peripheral and the peripheral interrupt to be enabled. The LLWU's WUME bit enables the internal module flag as a wakeup input. After wakeup, the flags are cleared based on the peripheral clearing mechanism.

19.2 Introduction

The LLWU module allows the user to select up to 32 external pins and up to 8 internal modules as interrupt wake-up sources from low-leakage power modes.

The input sources are described in the device's chip configuration details. Each of the available wake-up sources can be individually enabled.

The $\overline{\text{RESET}}$ pin is an additional source for triggering an exit from low-leakage power modes, and causes the MCU to exit VLLS through a reset flow.

The LLWU module also includes four optional digital pin filters for the external wakeup pins.

See AN4503: Power Management for Kinetis MCUs for further details on using the LLWU.

19.2.1 Features

The LLWU module features include:

- Support for up to 32 external input pins and up to 8 internal modules with individual enable bits for MCU interrupt from low leakage modes
- Input sources may be external pins or from internal peripherals capable of running in VLLS. See the chip configuration information for wakeup input sources for this device.
- External pin wake-up inputs, each of which is programmable as falling-edge, rising-edge, or any change
- Wake-up inputs that are activated after MCU enters a low-leakage power mode
- Optional digital filters provided to qualify an external pin detect. Note that when the LPO clock is disabled, the filters are disabled and bypassed.

19.2.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from VLLS, the LLWU continues to detect wake-up events until the user has acknowledged the wake-up via a write to PMC_REGSC[ACKISO].

19.2.2.1 VLLS modes

All wakeup and reset events result in VLLS exit via a reset flow.

19.2.2.2 Non-low leakage modes

The LLWU is not active in all non-low leakage modes where detection and control logic are in a static state. The LLWU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

When the wake-up pin filters are enabled, filter operation begins immediately. If a low leakage mode is entered within five LPO clock cycles of an active edge, the edge event will be detected by the LLWU.

19.2.2.3 Debug mode

When the chip is in Debug mode and then enters a VLLSx mode, no debug logic works in the fully-functional low-leakage mode. Upon an exit from the VLLSx mode, the LLWU becomes inactive.

19.2.3 Block diagram

The following figure is the block diagram for the LLWU module.

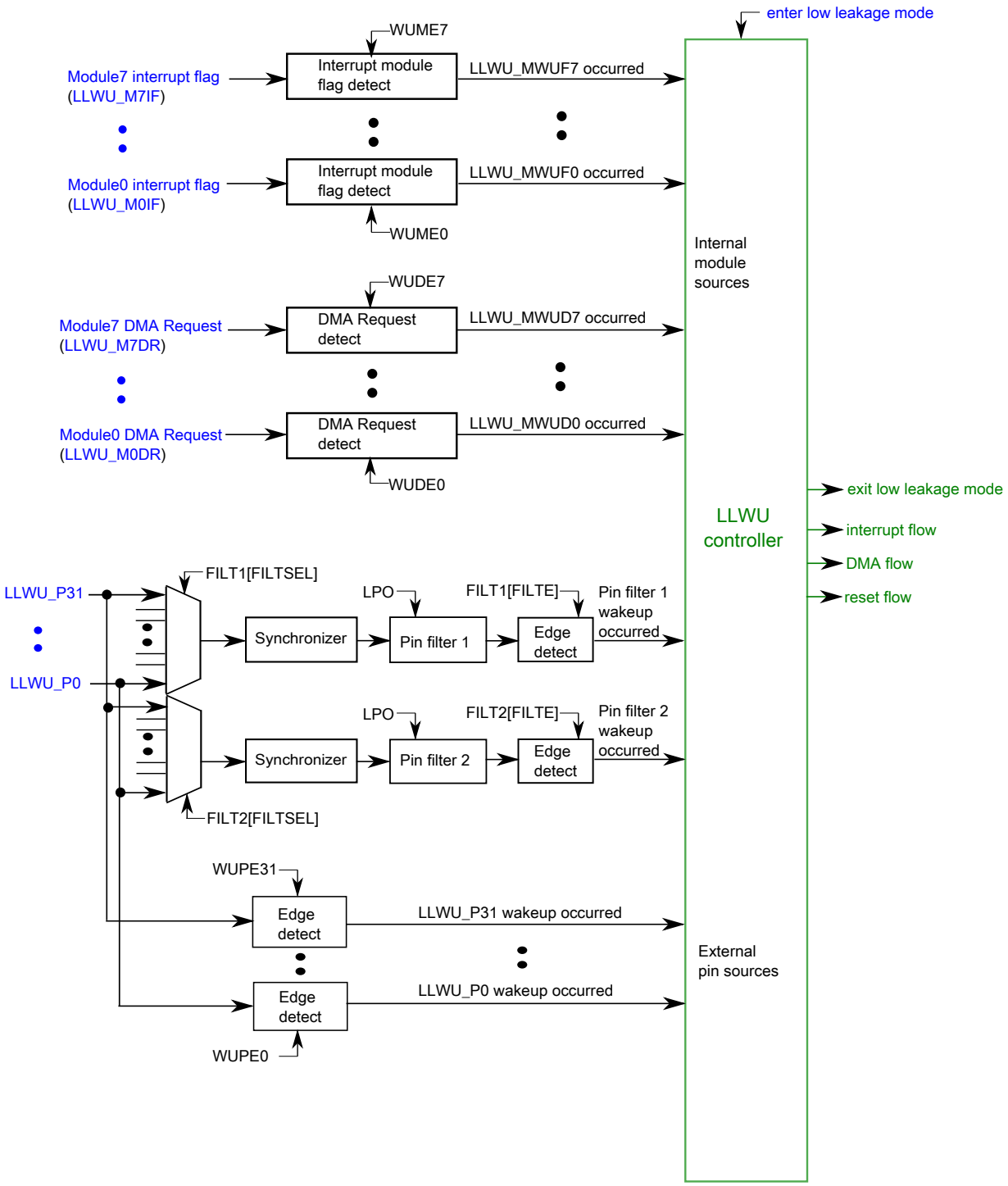


Figure 19-1. LLWU block diagram

19.3 LLWU signal descriptions

The signal properties of LLWU are shown in the table found here.

The external wakeup input pins can be enabled to detect either rising-edge, falling-edge, or on any change.

Table 19-2. LLWU signal descriptions

Signal	Description	I/O
LLWU_Pn	Wakeup inputs (n = 0- 31)	I

19.4 Memory map/register definition

The LLWU includes the following registers:

- Wake-up source enable registers
  - Enable external pin input sources
  - Enable internal peripheral interrupt sources
- Wake-up flag registers
  - Indication of wakeup source that caused exit from a low-leakage power mode includes external pin or internal module interrupt
- Wake-up pin filter enable registers

NOTE

The LLWU registers can be written only in supervisor mode. Write accesses in user mode are blocked and result in a bus error.

The LLWU registers maintain their values during VLLS because the module remains powered.

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect register bitfield values. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the register values for this module. For more information about the types of reset on this chip, see the Reset section.

LLWU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_C000	LLWU Pin Enable 1 register (LLWU_PE1)	8	R/W	00h	19.4.1/306

Table continues on the next page...

LLWU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_C001	LLWU Pin Enable 2 register (LLWU_PE2)	8	R/W	00h	19.4.2/307
4007_C002	LLWU Pin Enable 3 register (LLWU_PE3)	8	R/W	00h	19.4.3/308
4007_C003	LLWU Pin Enable 4 register (LLWU_PE4)	8	R/W	00h	19.4.4/309
4007_C004	LLWU Pin Enable 5 register (LLWU_PE5)	8	R/W	00h	19.4.5/311
4007_C005	LLWU Pin Enable 6 register (LLWU_PE6)	8	R/W	00h	19.4.6/312
4007_C006	LLWU Pin Enable 7 register (LLWU_PE7)	8	R/W	00h	19.4.7/313
4007_C007	LLWU Pin Enable 8 register (LLWU_PE8)	8	R/W	00h	19.4.8/314
4007_C008	LLWU Module Enable register (LLWU_ME)	8	R/W	00h	19.4.9/315
4007_C009	LLWU Pin Flag 1 register (LLWU_PF1)	8	R/W	00h	19.4.10/317
4007_C00A	LLWU Pin Flag 2 register (LLWU_PF2)	8	R/W	00h	19.4.11/318
4007_C00B	LLWU Pin Flag 3 register (LLWU_PF3)	8	R/W	00h	19.4.12/320
4007_C00C	LLWU Pin Flag 4 register (LLWU_PF4)	8	R/W	00h	19.4.13/322
4007_C00D	LLWU Module Flag 5 register (LLWU_MF5)	8	R	00h	19.4.14/324
4007_C00E	LLWU Pin Filter 1 register (LLWU_FILT1)	8	R/W	00h	19.4.15/325
4007_C00F	LLWU Pin Filter 2 register (LLWU_FILT2)	8	R/W	00h	19.4.16/326

19.4.1 LLWU Pin Enable 1 register (LLWU_PE1)

LLWU_PE1 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P3-LLWU_P0.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 0h offset = 4007_C000h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE3		WUPE2		WUPE1		WUPE0	
Reset	0	0	0	0	0	0	0	0

LLWU_PE1 field descriptions

Field	Description
7–6 WUPE3	Wakeup Pin Enable For LLWU_P3 Enables and configures the edge detection for the wakeup pin.

Table continues on the next page...

LLWU_PE1 field descriptions (continued)

Field	Description
	00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE2	Wakeup Pin Enable For LLWU_P2 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE1	Wakeup Pin Enable For LLWU_P1 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE0	Wakeup Pin Enable For LLWU_P0 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.2 LLWU Pin Enable 2 register (LLWU_PE2)

LLWU_PE2 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P7-LLWU_P4.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 1h offset = 4007_C001h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE7		WUPE6		WUPE5		WUPE4	
Reset	0	0	0	0	0	0	0	0

LLWU_PE2 field descriptions

Field	Description
7–6 WUPE7	Wakeup Pin Enable For LLWU_P7 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE6	Wakeup Pin Enable For LLWU_P6 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE5	Wakeup Pin Enable For LLWU_P5 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE4	Wakeup Pin Enable For LLWU_P4 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.3 LLWU Pin Enable 3 register (LLWU_PE3)

LLWU_PE3 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P11-LLWU_P8.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of

reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 2h offset = 4007_C002h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE11		WUPE10		WUPE9		WUPE8	
Reset	0	0	0	0	0	0	0	0

LLWU_PE3 field descriptions

Field	Description
7–6 WUPE11	Wakeup Pin Enable For LLWU_P11 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE10	Wakeup Pin Enable For LLWU_P10 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE9	Wakeup Pin Enable For LLWU_P9 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE8	Wakeup Pin Enable For LLWU_P8 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.4 LLWU Pin Enable 4 register (LLWU_PE4)

LLWU_PE4 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P15-LLWU_P12.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 3h offset = 4007_C003h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE15		WUPE14		WUPE13		WUPE12	
Reset	0	0	0	0	0	0	0	0

LLWU_PE4 field descriptions

Field	Description
7–6 WUPE15	Wakeup Pin Enable For LLWU_P15 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE14	Wakeup Pin Enable For LLWU_P14 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE13	Wakeup Pin Enable For LLWU_P13 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE12	Wakeup Pin Enable For LLWU_P12 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.5 LLWU Pin Enable 5 register (LLWU_PE5)

LLWU_PE5 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P19-LLWU_P16.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 4h offset = 4007_C004h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE19		WUPE18		WUPE17		WUPE16	
Reset	0	0	0	0	0	0	0	0

LLWU_PE5 field descriptions

Field	Description
7–6 WUPE19	Wakeup Pin Enable For LLWU_P19 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE18	Wakeup Pin Enable For LLWU_P18 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE17	Wakeup Pin Enable For LLWU_P17 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE16	Wakeup Pin Enable For LLWU_P16 Enables and configures the edge detection for the wakeup pin.

Table continues on the next page...

LLWU_PE5 field descriptions (continued)

Field	Description
	00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.6 LLWU Pin Enable 6 register (LLWU_PE6)

LLWU_PE6 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P23-LLWU_P20.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 5h offset = 4007_C005h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE23		WUPE22		WUPE21		WUPE20	
Reset	0	0	0	0	0	0	0	0

LLWU_PE6 field descriptions

Field	Description
7–6 WUPE23	Wakeup Pin Enable For LLWU_P23 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE22	Wakeup Pin Enable For LLWU_P22 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE21	Wakeup Pin Enable For LLWU_P21

Table continues on the next page...

LLWU_PE6 field descriptions (continued)

Field	Description
	Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE20	Wakeup Pin Enable For LLWU_P20 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.7 LLWU Pin Enable 7 register (LLWU_PE7)

LLWU_PE7 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P27-LLWU_P24.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 6h offset = 4007_C006h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE27		WUPE26		WUPE25		WUPE24	
Reset	0	0	0	0	0	0	0	0

LLWU_PE7 field descriptions

Field	Description
7–6 WUPE27	Wakeup Pin Enable For LLWU_P27 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

Table continues on the next page...

LLWU_PE7 field descriptions (continued)

Field	Description
5–4 WUPE26	Wakeup Pin Enable For LLWU_P26 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE25	Wakeup Pin Enable For LLWU_P25 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE24	Wakeup Pin Enable For LLWU_P24 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.8 LLWU Pin Enable 8 register (LLWU_PE8)

LLWU_PE8 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P31-LLWU_P28.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 7h offset = 4007_C007h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE31		WUPE30		WUPE29		WUPE28	
Reset	0	0	0	0	0	0	0	0

LLWU_PE8 field descriptions

Field	Description
7–6 WUPE31	Wakeup Pin Enable For LLWU_P31 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE30	Wakeup Pin Enable For LLWU_P30 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE29	Wakeup Pin Enable For LLWU_P29 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
WUPE28	Wakeup Pin Enable For LLWU_P28 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

19.4.9 LLWU Module Enable register (LLWU_ME)

LLWU_ME contains the bits to enable the internal module flag as a wakeup input source for inputs MWUF7-MWUF0.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 8h offset = 4007_C008h

Bit	7	6	5	4	3	2	1	0
Read Write	WUME7	WUME6	WUME5	WUME4	WUME3	WUME2	WUME1	WUME0
Reset	0	0	0	0	0	0	0	0

LLWU_ME field descriptions

Field	Description
7 WUME7	Wakeup Module Enable For Module 7 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
6 WUME6	Wakeup Module Enable For Module 6 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
5 WUME5	Wakeup Module Enable For Module 5 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
4 WUME4	Wakeup Module Enable For Module 4 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
3 WUME3	Wakeup Module Enable For Module 3 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
2 WUME2	Wakeup Module Enable For Module 2 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
1 WUME1	Wakeup Module Enable for Module 1 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
0 WUME0	Wakeup Module Enable For Module 0 Enables an internal module as a wakeup source input.

Table continues on the next page...

LLWU_ME field descriptions (continued)

Field	Description
	0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source

19.4.10 LLWU Pin Flag 1 register (LLWU_PF1)

LLWU_PF1 contains the wakeup flags indicating which wakeup source caused the MCU to exit VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + 9h offset = 4007_C009h

Bit	7	6	5	4	3	2	1	0
Read	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF1 field descriptions

Field	Description
7 WUF7	Wakeup Flag For LLWU_P7 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF7. 0 LLWU_P7 input was not a wakeup source 1 LLWU_P7 input was a wakeup source
6 WUF6	Wakeup Flag For LLWU_P6 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF6. 0 LLWU_P6 input was not a wakeup source 1 LLWU_P6 input was a wakeup source

Table continues on the next page...

LLWU_PF1 field descriptions (continued)

Field	Description
5 WUF5	Wakeup Flag For LLWU_P5 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF5. 0 LLWU_P5 input was not a wakeup source 1 LLWU_P5 input was a wakeup source
4 WUF4	Wakeup Flag For LLWU_P4 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF4. 0 LLWU_P4 input was not a wakeup source 1 LLWU_P4 input was a wakeup source
3 WUF3	Wakeup Flag For LLWU_P3 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF3. 0 LLWU_P3 input was not a wakeup source 1 LLWU_P3 input was a wakeup source
2 WUF2	Wakeup Flag For LLWU_P2 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF2. 0 LLWU_P2 input was not a wakeup source 1 LLWU_P2 input was a wakeup source
1 WUF1	Wakeup Flag For LLWU_P1 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF1. 0 LLWU_P1 input was not a wakeup source 1 LLWU_P1 input was a wakeup source
0 WUF0	Wakeup Flag For LLWU_P0 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF0. 0 LLWU_P0 input was not a wakeup source 1 LLWU_P0 input was a wakeup source

19.4.11 LLWU Pin Flag 2 register (LLWU_PF2)

LLWU_PF2 contains the wakeup flags indicating which wakeup source caused the MCU to exit or VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + Ah offset = 4007_C00Ah

Bit	7	6	5	4	3	2	1	0
Read	WUF15	WUF14	WUF13	WUF12	WUF11	WUF10	WUF9	WUF8
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF2 field descriptions

Field	Description
7 WUF15	Wakeup Flag For LLWU_P15 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF15. 0 LLWU_P15 input was not a wakeup source 1 LLWU_P15 input was a wakeup source
6 WUF14	Wakeup Flag For LLWU_P14 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF14. 0 LLWU_P14 input was not a wakeup source 1 LLWU_P14 input was a wakeup source
5 WUF13	Wakeup Flag For LLWU_P13 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF13. 0 LLWU_P13 input was not a wakeup source 1 LLWU_P13 input was a wakeup source
4 WUF12	Wakeup Flag For LLWU_P12 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF12. 0 LLWU_P12 input was not a wakeup source 1 LLWU_P12 input was a wakeup source

Table continues on the next page...

LLWU_PF2 field descriptions (continued)

Field	Description
3 WUF11	Wakeup Flag For LLWU_P11 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF11. 0 LLWU_P11 input was not a wakeup source 1 LLWU_P11 input was a wakeup source
2 WUF10	Wakeup Flag For LLWU_P10 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF10. 0 LLWU_P10 input was not a wakeup source 1 LLWU_P10 input was a wakeup source
1 WUF9	Wakeup Flag For LLWU_P9 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF9. 0 LLWU_P9 input was not a wakeup source 1 LLWU_P9 input was a wakeup source
0 WUF8	Wakeup Flag For LLWU_P8 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF8. 0 LLWU_P8 input was not a wakeup source 1 LLWU_P8 input was a wakeup source

19.4.12 LLWU Pin Flag 3 register (LLWU_PF3)

LLWU_PF3 contains the wakeup flags indicating which wakeup source caused the MCU to exit VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + Bh offset = 4007_C00Bh

Bit	7	6	5	4	3	2	1	0
Read	WUF23	WUF22	WUF21	WUF20	WUF19	WUF18	WUF17	WUF16
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF3 field descriptions

Field	Description
7 WUF23	Wakeup Flag For LLWU_P23 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF23. 0 LLWU_P23 input was not a wakeup source 1 LLWU_P23 input was a wakeup source
6 WUF22	Wakeup Flag For LLWU_P22 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF22. 0 LLWU_P22 input was not a wakeup source 1 LLWU_P22 input was a wakeup source
5 WUF21	Wakeup Flag For LLWU_P21 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF21. 0 LLWU_P21 input was not a wakeup source 1 LLWU_P21 input was a wakeup source
4 WUF20	Wakeup Flag For LLWU_P20 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF20. 0 LLWU_P20 input was not a wakeup source 1 LLWU_P20 input was a wakeup source
3 WUF19	Wakeup Flag For LLWU_P19 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF19. 0 LLWU_P19 input was not a wakeup source 1 LLWU_P19 input was a wakeup source
2 WUF18	Wakeup Flag For LLWU_P18 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF18. 0 LLWU_P18 input was not a wakeup source 1 LLWU_P18 input was a wakeup source
1 WUF17	Wakeup Flag For LLWU_P17

Table continues on the next page...

LLWU_PF3 field descriptions (continued)

Field	Description
	Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF17. 0 LLWU_P17 input was not a wakeup source 1 LLWU_P17 input was a wakeup source
0 WUF16	Wakeup Flag For LLWU_P16 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF16. 0 LLWU_P16 input was not a wakeup source 1 LLWU_P16 input was a wakeup source

19.4.13 LLWU Pin Flag 4 register (LLWU_PF4)

LLWU_PF4 contains the wakeup flags indicating which wakeup source caused the MCU to exit or VLLS mode. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + Ch offset = 4007_C00Ch

Bit	7	6	5	4	3	2	1	0
Read	WUF31	WUF30	WUF29	WUF28	WUF27	WUF26	WUF25	WUF24
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_PF4 field descriptions

Field	Description
7 WUF31	Wakeup Flag For LLWU_P31 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF31.

Table continues on the next page...

LLWU_PF4 field descriptions (continued)

Field	Description
	0 LLWU_P31 input was not a wakeup source 1 LLWU_P31 input was a wakeup source
6 WUF30	Wakeup Flag For LLWU_P30 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF30. 0 LLWU_P30 input was not a wakeup source 1 LLWU_P30 input was a wakeup source
5 WUF29	Wakeup Flag For LLWU_P29 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF29. 0 LLWU_P29 input was not a wakeup source 1 LLWU_P29 input was a wakeup source
4 WUF28	Wakeup Flag For LLWU_P28 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF28. 0 LLWU_P28 input was not a wakeup source 1 LLWU_P28 input was a wakeup source
3 WUF27	Wakeup Flag For LLWU_P27 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF27. 0 LLWU_P27 input was not a wakeup source 1 LLWU_P27 input was a wakeup source
2 WUF26	Wakeup Flag For LLWU_P26 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF26. 0 LLWU_P26 input was not a wakeup source 1 LLWU_P26 input was a wakeup source
1 WUF25	Wakeup Flag For LLWU_P25 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF25. 0 LLWU_P25 input was not a wakeup source 1 LLWU_P25 input was a wakeup source
0 WUF24	Wakeup Flag For LLWU_P24 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF24. 0 LLWU_P24 input was not a wakeup source 1 LLWU_P24 input was a wakeup source

19.4.14 LLWU Module Flag 5 register (LLWU_MF5)

LLWU_MF5 contains the wakeup flags indicating which internal wakeup source caused the MCU to exit VLLS mode. For VLLS, this is the source causing the MCU reset flow.

For internal peripherals that are capable of running in a low-leakage power mode, such as a real time clock module or CMP module, the flag from the associated peripheral is accessible as the MWUFx bit. The flag will need to be cleared in the peripheral instead of writing a 1 to the MWUFx bit.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + Dh offset = 4007_C00Dh

Bit	7	6	5	4	3	2	1	0
Read	MWUF7	MWUF6	MWUF5	MWUF4	MWUF3	MWUF2	MWUF1	MWUF0
Write								
Reset	0	0	0	0	0	0	0	0

LLWU_MF5 field descriptions

Field	Description
7 MWUF7	Wakeup flag For module 7 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 7 input was not a wakeup source 1 Module 7 input was a wakeup source
6 MWUF6	Wakeup flag For module 6 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 6 input was not a wakeup source 1 Module 6 input was a wakeup source
5 MWUF5	Wakeup flag For module 5 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.

Table continues on the next page...

LLWU_MF5 field descriptions (continued)

Field	Description
	0 Module 5 input was not a wakeup source 1 Module 5 input was a wakeup source
4 MWUF4	Wakeup flag For module 4 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 4 input was not a wakeup source 1 Module 4 input was a wakeup source
3 MWUF3	Wakeup flag For module 3 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 3 input was not a wakeup source 1 Module 3 input was a wakeup source
2 MWUF2	Wakeup flag For module 2 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 2 input was not a wakeup source 1 Module 2 input was a wakeup source
1 MWUF1	Wakeup flag For module 1 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 1 input was not a wakeup source 1 Module 1 input was a wakeup source
0 MWUF0	Wakeup flag For module 0 Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0 Module 0 input was not a wakeup source 1 Module 0 input was a wakeup source

19.4.15 LLWU Pin Filter 1 register (LLWU_FILT1)

LLWU_FILT1 is a control and status register that is used to enable/disable the digital filter 1 features for an external pin.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the

bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + Eh offset = 4007_C00Eh

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE			FILTSEL			
Write	w1c							
Reset	0	0	0	0	0	0	0	0

LLWU_FILT1 field descriptions

Field	Description
7 FILTF	Filter Detect Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
6–5 FILTE	Digital Filter On External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
FILTSEL	Filter Pin Select Selects 1 of the wakeup pins to be muxed into the filter. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter

19.4.16 LLWU Pin Filter 2 register (LLWU_FILT2)

LLWU_FILT2 is a control and status register that is used to enable/disable the digital filter 2 features for an external pin.

NOTE

When the chip is in VLLS mode, resets that are triggered by an LLWU wakeup source (not including the $\overline{\text{RESET}}$ pin) do not affect the bitfield values for this register. All other types of reset (collectively known as *Chip Reset not VLLS*) do affect the bitfield values for this register. For more information about the types of reset on this chip, see the Reset section.

Address: 4007_C000h base + Fh offset = 4007_C00Fh

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE			FILTSEL			
Write	w1c							
Reset	0	0	0	0	0	0	0	0

LLWU_FILT2 field descriptions

Field	Description
7 FILTF	Filter Detect Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0 Pin Filter 2 was not a wakeup source 1 Pin Filter 2 was a wakeup source
6–5 FILTE	Digital Filter On External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
FILTSEL	Filter Pin Select Selects 1 of the wakeup pins to be muxed into the filter. 00000 Select LLWU_P0 for filter ... ... 11111 Select LLWU_P31 for filter

19.5 Functional description

Thie low-leakage wakeup unit (LLWU) module allows internal peripherals and external input pins as a source of wakeup from low-leakage modes.

It is operational only in VLLSx modes.

The LLWU module contains pin enables for each external pin and internal module. For each external pin, the user can disable or select the edge type for the wakeup with the following options:

- Falling-edge
- Rising-edge
- Either-edge

When an external pin is enabled as a wakeup source, the pin must be configured as an input pin.

The LLWU implements optional 3-cycle glitch filters, based on the LPO clock. A detected external pin is required to remain asserted until the enabled glitch filter times out. Additional latency of up to 2 cycles is due to synchronization, which results in a total of up to 5 cycles of delay before the detect circuit alerts the system to the wakeup or reset event when the filter function is enabled. Four wakeup detect filters are available for selected external pins. Glitch filtering is not provided on the internal modules.

For internal module interrupts, the WUMEx bit enables the associated module interrupt as a wakeup source.

19.5.1 VLLS modes

For any wakeup from VLLS, recovery is always via a reset flow and RCM_SRS[WAKEUP] is set indicating the low-leakage mode was active. State retention data is lost and I/O will be restored after PMC_REGSC[ACKISO] has been written.

A VLLS exit event due to $\overline{\text{RESET}}$ pin assertion causes an exit via a system reset. State retention data is lost and the I/O states immediately return to their reset state. The RCM_SRS[WAKEUP] and RCM_SRS[PIN] bits are set and the system executes a reset flow before CPU operation begins with a reset vector fetch.

19.5.2 Initialization

For an enabled peripheral wakeup input, the peripheral flag must be cleared by software before entering VLLSx mode to avoid an immediate exit from the mode.

Flags associated with external input pins, filtered and unfiltered, must also be cleared by software prior to entry to VLLSx mode.

After enabling an external pin filter or changing the source pin, wait at least five LPO clock cycles before entering VLLSx mode to allow the filter to initialize.

NOTE

After recovering from a VLLS mode, user must restore chip configuration before clearing PMC_REGSC[ACKISO]. In particular, pin configuration for enabled LLWU wake-up pins must be restored to avoid any LLWU flag from being falsely set when PMC_REGSC[ACKISO] is cleared.

The signal selected as a wake-up source pin must be a digital pin, as selected in the pin mux control.

Chapter 20
Crossbar Switch Lite (AXBS-Lite)

20.1 Crossbar-Light Switch Configuration

This section summarizes how the module has been configured in the chip.

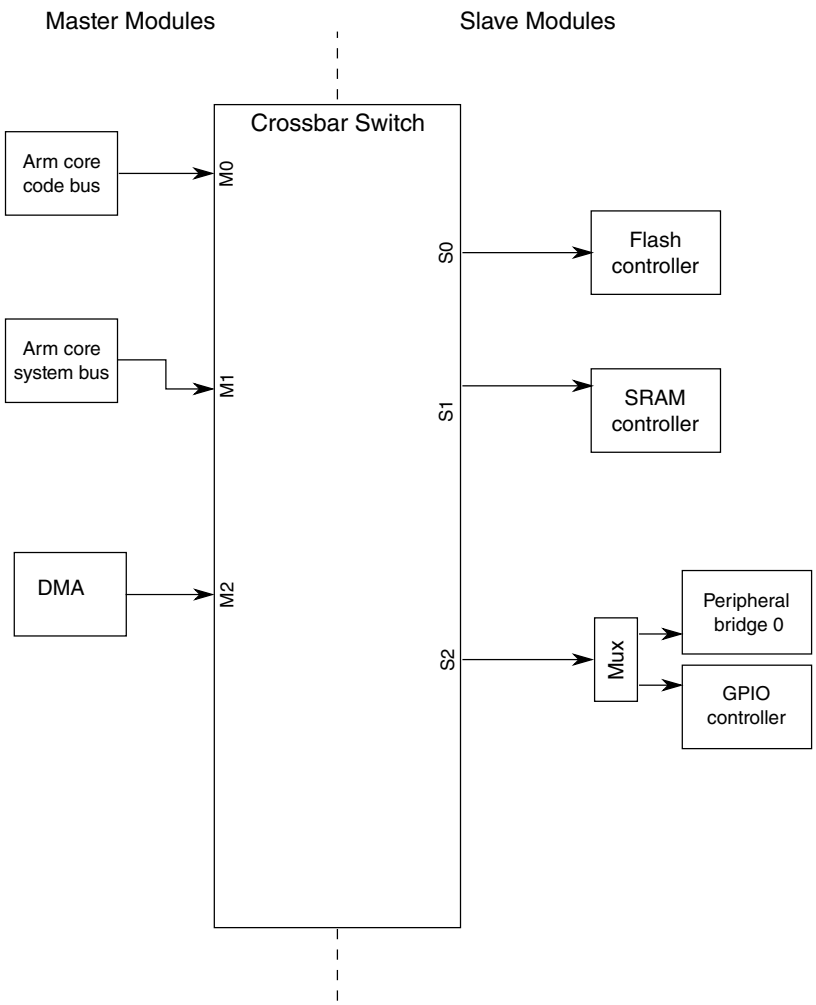


Figure 20-1. Crossbar-Light switch integration

20.1.1 Crossbar Switch Master Assignments

The masters connected to the crossbar switch are assigned as follows:

Master module	Master port number
Arm core code bus	0
Arm core system bus	1
DMA	2

20.1.2 Crossbar Switch Slave Assignments

The slaves connected to the crossbar switch are assigned as follows:

Slave module	Slave port number
Flash memory controller	S0
SRAM controller	S1
AIPS_Lite bus controller/GPIO	S2

20.2 Introduction

The information found here provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows up to four bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave.

20.2.1 Features

The crossbar switch includes these features:

- Symmetric crossbar bus switch implementation
    - Allows concurrent accesses from different masters to different slaves

- Up to single-clock 32-bit transfer
- Programmable configuration for fixed-priority or round-robin slave port arbitration (see the chip-specific information).

20.3 Memory Map / Register Definition

This crossbar switch is designed for minimal gate count. It, therefore, has no memory-mapped configuration registers.

Please see the chip-specific information for information on whether the arbitration method in the crossbar switch is programmable, and by which module.

20.4 Functional Description

20.4.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master simply sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device has no knowledge of whether it actually owns the slave port it is targeting. While the master does not have control of the slave port it is targeting, it simply waits.

After the master has control of the slave port it is targeting, the master remains in control of the slave port until it relinquishes the slave port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it remains parked with the last master to use the slave port. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port.

20.4.2 Arbitration

The crossbar switch supports two arbitration algorithms:

- Fixed priority
- Round-robin

20.4.2.1 Arbitration during undefined length bursts

All lengths of burst accesses lock out arbitration until the last beat of the burst.

20.4.2.2 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level with the highest numbered master having the highest priority (for example, in a system with 5 masters, master 1 has lower priority than master 3). If two masters request access to the same slave port, the master with the highest priority gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing accesses from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the proper master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port:

Table 20-1. How the Crossbar Switch grants control of a slave port to a master

When	Then the Crossbar Switch grants control to the requesting master
Both of the following are true:	At the next clock edge

Table continues on the next page...

Table 20-1. How the Crossbar Switch grants control of a slave port to a master (continued)

When	Then the Crossbar Switch grants control to the requesting master
• The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

20.4.2.3 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requesting master becomes owner of the slave bus at the next transfer boundary. Priority is based on how far ahead the ID of the requesting master is to the ID of the last master.

After granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and master 0, 4, and 5 make simultaneous requests, they are serviced in the order: 4 then 5 then 0.

The round-robin arbitration mode generally provides a more fair allocation of the available slave-port bandwidth (compared to fixed priority) as the fixed master priority does not affect the master selection.

20.5 Initialization/application information

No initialization is required for the crossbar switch. See the chip-specific crossbar switch information for the reset state of the arbitration scheme.

Chapter 21
Peripheral Bridge (AIPS-Lite)

21.1 Number of peripheral bridges

This device contains one peripheral bridge AIPS-Lite with registers.

21.2 Memory map

The peripheral bridge is used to access the registers of most of the modules on this device. See Peripheral Memory Map for the memory slot assignment of each module.

21.3 PACR registers

The single peripheral bridge supports up to 128 peripherals each assigned to a PACRx field within the PACRA-PACRP registers. However, fewer peripherals are supported on this device. See Peripheral Memory Map for details of the peripheral slot assignments for this device. Unused PACRx fields are reserved.

21.4 AIPS_Lite PACRE-P register reset values

The AIPSx_PACRE-P register reset values depend on the availability of a module on the chip. For each populated slot in slots 32-127 in Peripheral Bridge 0 (AIPS-Lite 0) Memory Map, the corresponding module's PACR[32:127] field resets to 0x4.

21.5 Introduction

The peripheral bridge converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

The peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 4 KB. (It might be possible that all the peripheral slots are not used. See the memory map chapter for details on slot assignments.) The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

21.5.1 Features

Key features of the peripheral bridge are:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width
- Programming model provides memory protection functionality

21.5.2 General operation

The slave devices connected to the peripheral bridge are modules which contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge.

The register maps of the peripherals are located on 4-KB boundaries. Each peripheral is allocated one or more 4-KB block(s) of the memory map.

21.6 Memory map/register definition

The 32-bit peripheral bridge registers can be accessed only in supervisor mode by trusted bus masters. Additionally, these registers must be read from or written to only by a 32-bit aligned access. The peripheral bridge registers are mapped into the Peripheral Access Control Register A PACRA[PACR0] address space.

AIPS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_0000	Master Privilege Register A (AIPS_MPRA)	32	R/W	See section	21.6.1/339
4000_0020	Peripheral Access Control Register (AIPS_PACRA)	32	R/W	See section	21.6.2/342
4000_0024	Peripheral Access Control Register (AIPS_PACRB)	32	R/W	See section	21.6.2/342
4000_0028	Peripheral Access Control Register (AIPS_PACRC)	32	R/W	See section	21.6.2/342
4000_002C	Peripheral Access Control Register (AIPS_PACRD)	32	R/W	See section	21.6.2/342
4000_0040	Peripheral Access Control Register (AIPS_PACRE)	32	R/W	See section	21.6.3/347
4000_0044	Peripheral Access Control Register (AIPS_PACRF)	32	R/W	See section	21.6.3/347
4000_0048	Peripheral Access Control Register (AIPS_PACRG)	32	R/W	See section	21.6.3/347
4000_004C	Peripheral Access Control Register (AIPS_PACRH)	32	R/W	See section	21.6.3/347
4000_0050	Peripheral Access Control Register (AIPS_PACRI)	32	R/W	See section	21.6.3/347
4000_0054	Peripheral Access Control Register (AIPS_PACRJ)	32	R/W	See section	21.6.3/347
4000_0058	Peripheral Access Control Register (AIPS_PACRK)	32	R/W	See section	21.6.3/347
4000_005C	Peripheral Access Control Register (AIPS_PACRL)	32	R/W	See section	21.6.3/347
4000_0060	Peripheral Access Control Register (AIPS_PACRM)	32	R/W	See section	21.6.3/347
4000_0064	Peripheral Access Control Register (AIPS_PACRN)	32	R/W	See section	21.6.3/347
4000_0068	Peripheral Access Control Register (AIPS_PACRO)	32	R/W	See section	21.6.3/347
4000_006C	Peripheral Access Control Register (AIPS_PACRP)	32	R/W	See section	21.6.3/347

21.6.1 Master Privilege Register A (AIPS_MPRA)

The MPRA specifies identical 4-bit fields defining the access-privilege level associated with a bus master to various peripherals on the chip. The register provides one field per bus master.

NOTE

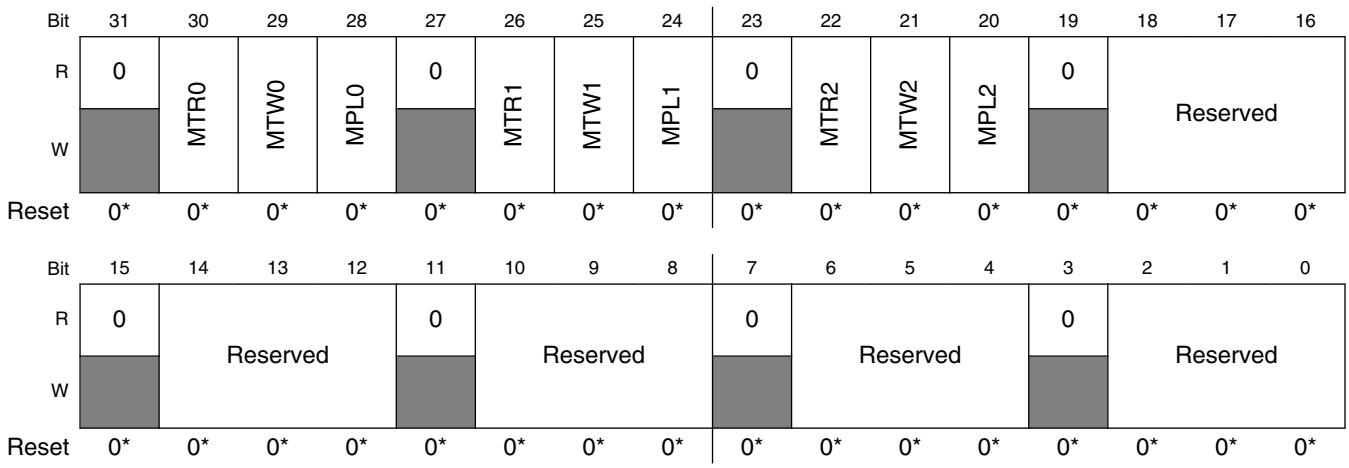
At reset, the default value loaded into the MPRA fields is chip-specific. See the chip configuration details for the value of a particular device.

A register field that maps to an unimplemented master or peripheral behaves as read-only-zero.

Each master is assigned a logical ID from 0 to 15. See the master logical ID assignment table in the chip-specific AIPS information.

Memory map/register definition

Address: 4000_0000h base + 0h offset = 4000_0000h



* Notes:
- The reset value is chip-dependent and can be found in the chip-specific AIPS information.

AIPS_MPRA field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 MTR0	Master 0 Trusted For Read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
29 MTW0	Master 0 Trusted For Writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
28 MPL0	Master 0 Privilege Level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 MTR1	Master 1 Trusted for Read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
25 MTW1	Master 1 Trusted for Writes Determines whether the master is trusted for write accesses.

Table continues on the next page...

AIPS_MPRA field descriptions (continued)

Field	Description
	0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
24 MPL1	Master 1 Privilege Level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 MTR2	Master 2 Trusted For Read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
21 MTW2	Master 2 Trusted For Writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
20 MPL2	Master 2 Privilege Level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 Reserved	This field is reserved.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–12 Reserved	This field is reserved.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 Reserved	This field is reserved.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 Reserved	This field is reserved.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved.

21.6.2 Peripheral Access Control Register (AIPS_PACR*n*)

Each PACR register consists of eight 4-bit PACR fields. Each PACR field defines the access levels for a particular peripheral. The mapping between a peripheral and its PACR field is shown in the table below. The peripheral assignment to each PACR is defined by the memory map slot that the peripheral is assigned to. See this chip's memory map for the assignment of a particular peripheral.

The following table shows the location of each peripheral slot's PACR field in the PACR registers.

Offset	Register	[31:28]	[27:24]	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
0x20	PACRA	PACR0	PACR1	PACR2	PACR3	PACR4	PACR5	PACR6	PACR7
0x24	PACRB	PACR8	PACR9	PACR10	PACR11	PACR12	PACR13	PACR14	PACR15
0x28	PACRC	PACR16	PACR17	PACR18	PACR19	PACR20	PACR21	PACR22	PACR23
0x2C	PACRD	PACR24	PACR25	PACR26	PACR27	PACR28	PACR29	PACR30	PACR31
0x30	Reserved								
0x34	Reserved								
0x38	Reserved								
0x3C	Reserved								
0x40	PACRE	PACR32	PACR33	PACR34	PACR35	PACR36	PACR37	PACR38	PACR39
0x44	PACRF	PACR40	PACR41	PACR42	PACR43	PACR44	PACR45	PACR46	PACR47
0x48	PACRG	PACR48	PACR49	PACR50	PACR51	PACR52	PACR53	PACR54	PACR55
0x4C	PACRH	PACR56	PACR57	PACR58	PACR59	PACR60	PACR61	PACR62	PACR63
0x50	PACRI	PACR64	PACR65	PACR66	PACR67	PACR68	PACR69	PACR70	PACR71
0x54	PACRJ	PACR72	PACR73	PACR74	PACR75	PACR76	PACR77	PACR78	PACR79
0x58	PACRK	PACR80	PACR81	PACR82	PACR83	PACR84	PACR85	PACR86	PACR87
0x5C	PACRL	PACR88	PACR89	PACR90	PACR91	PACR92	PACR93	PACR94	PACR95
0x60	PACRM	PACR96	PACR97	PACR98	PACR99	PACR100	PACR101	PACR102	PACR103
0x64	PACRN	PACR104	PACR105	PACR106	PACR107	PACR108	PACR109	PACR110	PACR111
0x68	PACRO	PACR112	PACR113	PACR114	PACR115	PACR116	PACR117	PACR118	PACR119
0x6C	PACRP	PACR120	PACR121	PACR122	PACR123	PACR124	PACR125	PACR126	PACR127

NOTE

The register field descriptions for PACR A-D, which control peripheral slots 0-31, are shown below. The following section, Peripheral Access Control Register (AIPS_PACR*n*), shows the register field descriptions for PACR E-P. All PACR registers are identical. They are divided into two sections because they occupy two non-contiguous address spaces.

Address: 4000_0000h base + 20h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SP0	WP0	TP0	0	SP1	WP1	TP1	0	SP2	WP2	TP2	0	SP3	WP3	TP3
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SP4	WP4	TP4	0	SP5	WP5	TP5	0	SP6	WP6	TP6	0	SP7	WP7	TP7
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- The reset value is chip-dependent and can be found in the AIPS chip-specific information.

AIPS_PACR*n* field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SP0	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SP1	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
25 WP1	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 SP2	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
21 WP2	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
20 TP2	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 SP3	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
17 WP3	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
16 TP3	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SP4	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
13 WP4	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
12 TP4	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 SP5	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
9 WP5	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
8 TP5	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 SP6	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
5 WP6	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
4 TP6	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SP7	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
1 WP7	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
0 TP7	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

21.6.3 Peripheral Access Control Register (AIPS_PACR*n*)

This section describes PACR registers E-P, which control peripheral slots 32-127. See Peripheral Access Control Register (AIPS_PACR*n*) for the description of these registers.

Address: 4000_0000h base + 40h offset + (4d × i), where i=0d to 11d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SP0	WP0	TP0	0	SP1	WP1	TP1	0	SP2	WP2	TP2	0	SP3	WP3	TP3
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SP4	WP4	TP4	0	SP5	WP5	TP5	0	SP6	WP6	TP6	0	SP7	WP7	TP7
W																
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- The reset value is chip-dependent and can be found in the AIPS chip-specific information.

AIPS_PACR*n* field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SP0	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SP1	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
25 WP1	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 SP2	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
21 WP2	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
20 TP2	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18 SP3	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
17 WP3	Write Protect Determines whether the peripheral allows write access. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
16 TP3	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 SP4	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
13 WP4	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
12 TP4	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 SP5	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
9 WP5	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
8 TP5	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 SP6	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPS_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
5 WP6	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
4 TP6	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SP7	Supervisor Protect Determines whether the peripheral requires supervisor privilege level for accesses. When this field is set, the master privilege level must indicate the supervisor access attribute, and the MPR*x*[MPL*n*] control field for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
1 WP7	Write Protect Determines whether the peripheral allows write accesses. When this field is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
0 TP7	Trusted Protect Determines whether the peripheral allows accesses from an untrusted master. When this field is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

21.7 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus.

The peripheral bridge manages all transactions destined for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

21.7.1 Access support

All combinations of access size and peripheral data port width are supported. An access that is larger than the target peripheral's data width will be decomposed to multiple, smaller accesses. Bus decomposition is terminated by a transfer error caused by an access to an empty register area.

Chapter 22
Direct memory access multiplexer (DMAMUX)

22.1 Chip-specific DMAMUX information

22.1.1 DMA MUX request sources

This device includes a DMA request mux that allows up to 63 DMA request signals to be mapped to any of the 16 DMA channels. Because of the mux there is not a hard correlation between any of the DMA request sources and a specific DMA channel.

Table 22-1. DMA request sources

Source Number	Peripheral Assignment	Description
0	Channel disabled	—
1	Reserved Not used	—
2	UART0 Receive	Receive
3	UART0 Transmit	Transmit
4	UART1 Recieve	Receive
5	UART1 Transmit	Transmit
6	flexPWM_WR0	Submodule 0 DMA request to update PWM buffers
7	flexPWM_WR1	Submodule 1 DMA request to update PWM buffers
8	flexPWM_WR2	Submodule 2 DMA request to update PWM buffers
9	flexPWM_WR3	Submodule 3 DMA request to update PWM buffers
10	flexPWMA_CP0	Submodule 0 DMA request for input capture on any of the Capture FIFO (OR of 6 capture circuits X0,X1,A0,A1,B0,B1)
11	flexPWMA_CP1	Submodule 1 DMA request for input capture on any of the Capture FIFO (OR of 6 capture circuits X0,X1,A0,A1,B0,B1)

Table continues on the next page...

Table 22-1. DMA request sources (continued)

Source Number	Peripheral Assignment	Description
12	flexPWMA_CP2	Submodule 2 DMA request for input capture on any of the Capture FIFO (OR of 6 capture circuits X0,X1,A0,A1,B0,B1)
13	flexPWMA_CP3	Submodule 3 DMA request for input capture on any of the Capture FIFO (OR of 6 capture circuits X0,X1,A0,A1,B0,B1)
14	FlexCAN0	DMA request on Receive mailbox level
15	FlexCAN1	DMA request on Receive mailbox level
16	SPI0 Receive	Receive
17	SPI0 transmit	Transmit
18	XBARA_OUT0	—
19	XBARA_OUT1	—
20	XBARA_OUT2	—
21	XBARA_OUT3	—
22	I2C0	—
23	—	—
24	FTM0 Ch0	Channel 0
25	FTM0 Ch1	Channel 1
26	FTM0 Ch2	Channel 2
27	FTM0 Ch3	Channel 3
28	FTM0 Ch4	Channel 4
29	FTM0 Ch5	Channel 5
30	FTM0 Ch6	Channel 6
31	FTM0 Ch7	Channel 7
32	FTM1 Ch0	Channel 0
33	FTM1 Ch1	Channel 1
34	CMP3	—
35	—	—
36	FTM3 Ch0	Channel 0
37	FTM3 Ch1	Channel 1
38	FTM3 Ch2	Channel 2
39	FTM3 Ch3	Channel 3
40	ADCA Scan complete	—
41	ADCB Scan complete	—
42	CMP0	—
43	CMP1	—
44	CMP2	—
45	DAC0	—
46	-	—

Table continues on the next page...

Table 22-1. DMA request sources (continued)

Source Number	Peripheral Assignment	Description
47	PDB1	—
48	PDB0	—
49	control module Port A	—
50	control module Port B	—
51	control module Port C	—
52	control module Port D	—
53	control module Port E	—
54	FTM3 Ch4	Channel 4
55	FTM3 Ch5	Channel 5
56	FTM3 Ch6	Channel 6
57	FTM3 Ch7	Channel 7
58	DMAX MUX always enabled	—
59	DMAX MUX always enabled	—
60	DMAX MUX always enabled	—
61	DMAX MUX always enabled	—
62	DMAX MUX always enabled	—
63	DMAX MUX always enabled	—

22.1.2 DMA transfers via PIT trigger

The PIT module can trigger a DMA transfer on the first four DMA channels. The assignments are detailed at PIT/DMA Periodic Trigger Assignments .

22.2 Introduction

22.2.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the 16 DMA channels. This process is illustrated in the following figure.

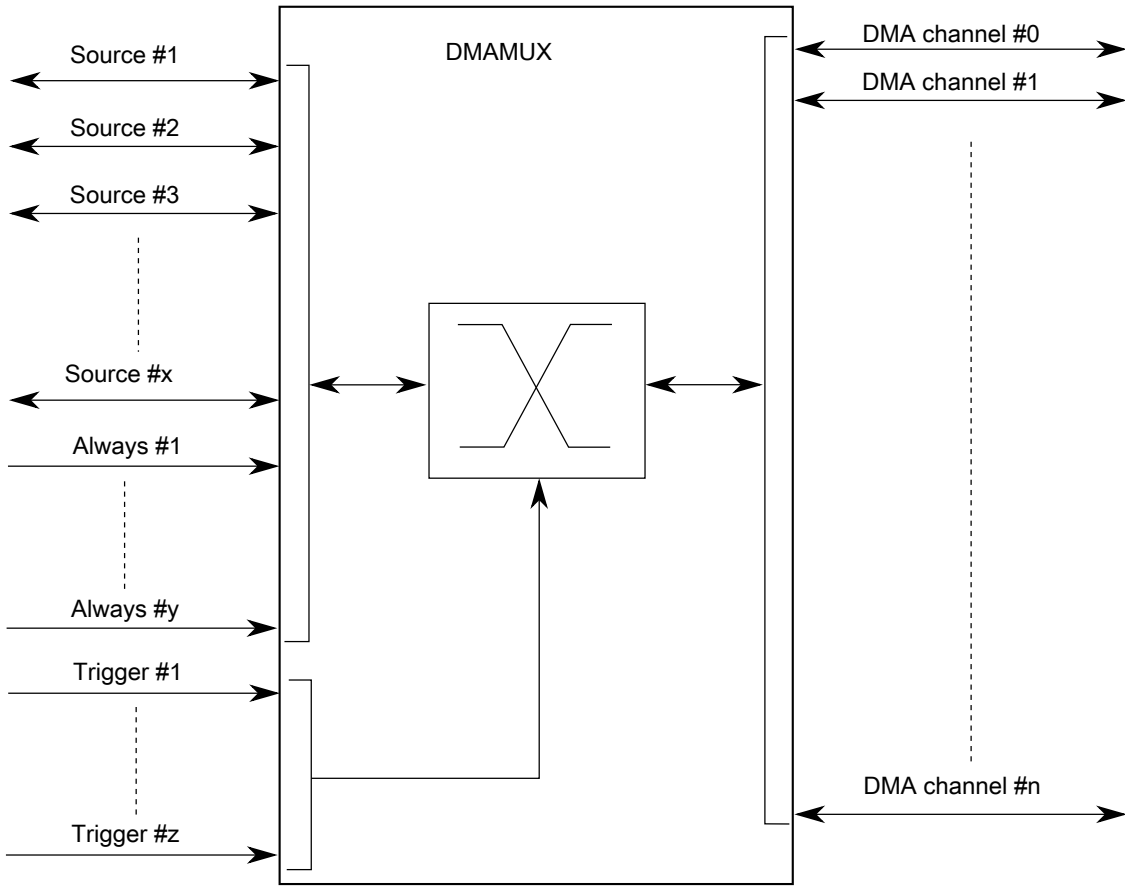


Figure 22-1. DMAMUX block diagram

22.2.2 Features

The DMAMUX module provides these features:

- Up to 63 peripheral slots and up to six always-on slots can be routed to 16 channels.
- 16 independently selectable DMA channel routers.
  - The first two channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the possible peripheral DMA slots or to one of the always-on slots.

22.2.3 Modes of operation

The following operating modes are available:

- Disabled mode

In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.

- Normal mode

  In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.

- Periodic Trigger mode

  In this mode, a DMA source may only request a DMA transfer, such as when a transmit buffer becomes empty or a receive buffer becomes full, periodically.

  Configuration of the period is done in the registers of the periodic interrupt timer (PIT). This mode is available only for channels 0–1.

22.3 External signal description

The DMAMUX has no external pins.

22.4 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

DMAMUX memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_1000	Channel Configuration register (DMAMUX_CHCFG0)	8	R/W	00h	22.4.1/358
4002_1001	Channel Configuration register (DMAMUX_CHCFG1)	8	R/W	00h	22.4.1/358
4002_1002	Channel Configuration register (DMAMUX_CHCFG2)	8	R/W	00h	22.4.1/358
4002_1003	Channel Configuration register (DMAMUX_CHCFG3)	8	R/W	00h	22.4.1/358
4002_1004	Channel Configuration register (DMAMUX_CHCFG4)	8	R/W	00h	22.4.1/358
4002_1005	Channel Configuration register (DMAMUX_CHCFG5)	8	R/W	00h	22.4.1/358
4002_1006	Channel Configuration register (DMAMUX_CHCFG6)	8	R/W	00h	22.4.1/358
4002_1007	Channel Configuration register (DMAMUX_CHCFG7)	8	R/W	00h	22.4.1/358
4002_1008	Channel Configuration register (DMAMUX_CHCFG8)	8	R/W	00h	22.4.1/358

Table continues on the next page...

DMAMUX memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_1009	Channel Configuration register (DMAMUX_CHCFG9)	8	R/W	00h	22.4.1/358
4002_100A	Channel Configuration register (DMAMUX_CHCFG10)	8	R/W	00h	22.4.1/358
4002_100B	Channel Configuration register (DMAMUX_CHCFG11)	8	R/W	00h	22.4.1/358
4002_100C	Channel Configuration register (DMAMUX_CHCFG12)	8	R/W	00h	22.4.1/358
4002_100D	Channel Configuration register (DMAMUX_CHCFG13)	8	R/W	00h	22.4.1/358
4002_100E	Channel Configuration register (DMAMUX_CHCFG14)	8	R/W	00h	22.4.1/358
4002_100F	Channel Configuration register (DMAMUX_CHCFG15)	8	R/W	00h	22.4.1/358

22.4.1 Channel Configuration register (DMAMUX_CHCFG*n*)

Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Setting multiple CHCFG registers with the same source value will result in unpredictable behavior. This is true, even if a channel is disabled (ENBL==0).

Before changing the trigger or source settings, a DMA channel must be disabled via CHCFGn[ENBL].

Address: 4002_1000h base + 0h offset + (1d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read Write	ENBL	TRIG	SOURCE					
Reset	0	0	0	0	0	0	0	0

DMAMUX_CHCFG*n* field descriptions

Field	Description
7 ENBL	DMA Channel Enable Enables the DMA channel. 0 DMA channel is disabled. This mode is primarily used during configuration of the DMAMux. The DMA has separate channel enables/disables, which should be used to disable or reconfigure a DMA channel. 1 DMA channel is enabled
6 TRIG	DMA Channel Trigger Enable Enables the periodic trigger capability for the triggered DMA channel.

Table continues on the next page...

DMAMUX_CHCFG*n* field descriptions (continued)

Field	Description
	0 Triggering is disabled. If triggering is disabled and ENBL is set, the DMA Channel will simply route the specified source to the DMA channel. (Normal mode) 1 Triggering is enabled. If triggering is enabled and ENBL is set, the DMAMUX is in Periodic Trigger mode.
SOURCE	DMA Channel Source (Slot) Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about the peripherals and their slot numbers.

22.5 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels.

As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in Enabling and configuring sources is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

Functionally, the DMAMUX channels may be divided into two classes:

- Channels that implement the normal routing functionality plus periodic triggering capability
- Channels that implement only the normal routing functionality

22.5.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first 2 channels of the DMAMUX provide a special periodic triggering capability that can be used to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention.

The trigger is generated by the periodic interrupt timer (PIT); as such, the configuration of the periodic triggering interval is done via configuration registers in the PIT. See the section on periodic interrupt timer for more information on this topic.

Note

Because of the dynamic nature of the system (due to DMA channel priorities, bus arbitration, interrupt service routine lengths, etc.), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

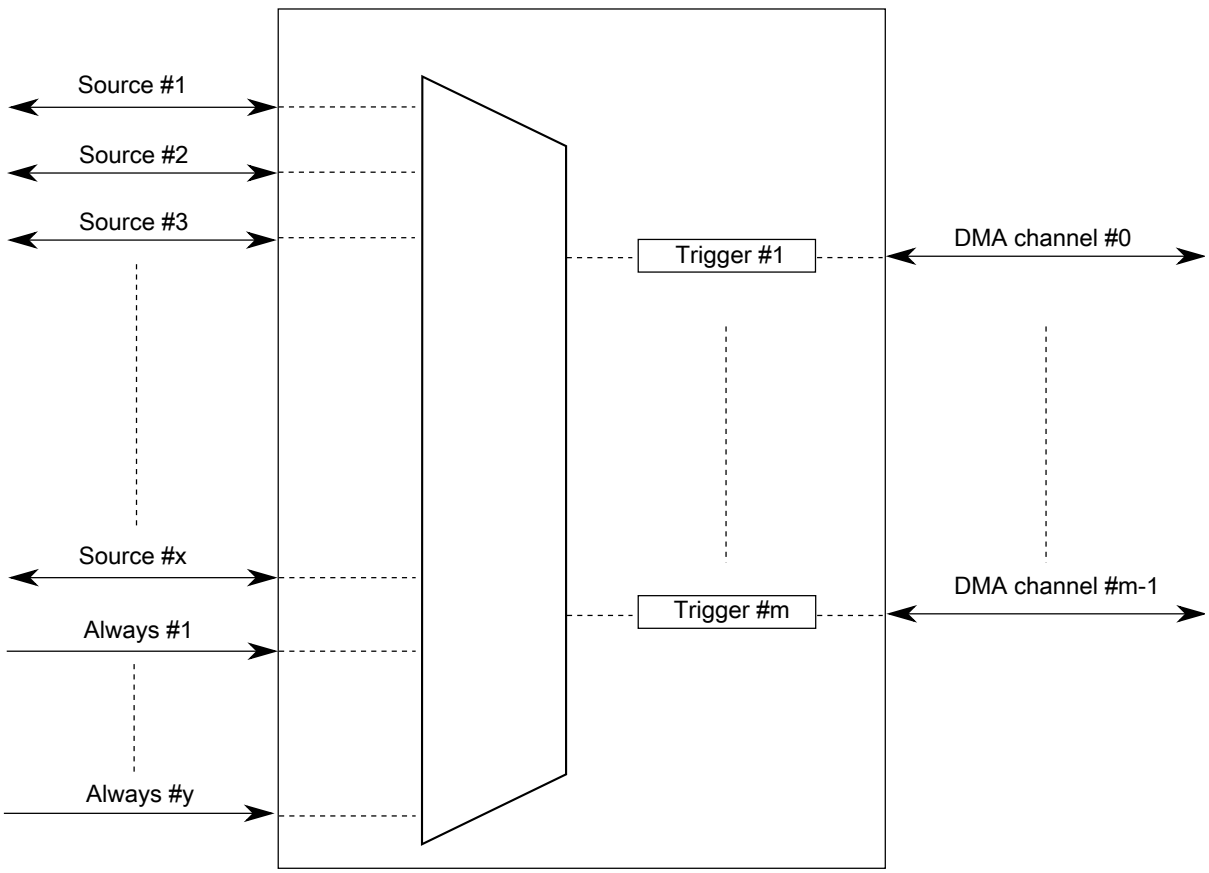


Figure 22-2. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.

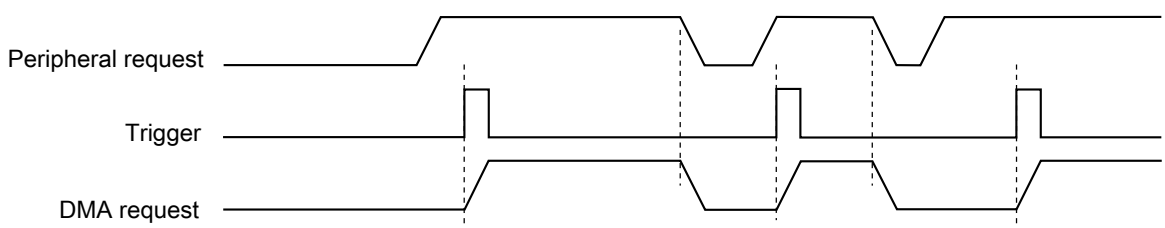


Figure 22-3. DMAMUX channel triggering: normal operation

After the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.

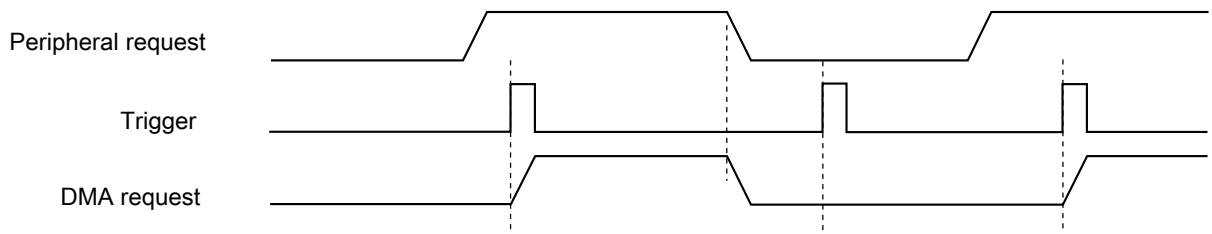


Figure 22-4. DMAMUX channel triggering: ignored trigger

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful for two types of situations:

- Periodically polling external devices on a particular bus

  As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI will request DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.

- Using the GPIO ports to drive or sample waveforms

  By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

A more detailed description of the capability of each trigger, including resolution, range of values, and so on, may be found in the periodic interrupt timer section.

22.5.2 DMA channels with no triggering capability

The other channels of the DMAMUX provide the normal routing functionality as described in Modes of operation.

22.5.3 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are six additional DMA sources that are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability).
- Performing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Performing DMA transfers from memory to the external bus, or vice-versa—Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation—Any DMA transfer that should be explicitly started by software.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are:

- Transfer all data in a single minor loop.

  By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use explicit software reactivation.

  In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers *after every minor loop*. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use an always-enabled DMA source.

In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and pointing to an "always enabled" source. Note that the reactivation of the channel can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of packets of data from one source to another, without processor intervention.

22.6 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

22.6.1 Reset

The reset state of each individual bit is shown in Memory map/register definition. In summary, after reset, all channels are disabled and must be explicitly enabled before use.

22.6.2 Enabling and configuring sources

To enable a source with periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 2 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0x00 to CHCFG1.

2. Configure channel 1 in the DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write 0xC5 to CHCFG1.

The following code example illustrates steps 1 and 4 above:

```
void DMAMUX_Init(uint8_t DMA_CH, uint8_t DMAMUX_SOURCE)
{
    DMAMUX_0.CHCFG[DMA_CH].B.SOURCE = DMAMUX_SOURCE;
    DMAMUX_0.CHCFG[DMA_CH].B.ENBL   = 1;
    DMAMUX_0.CHCFG[DMA_CH].B.TRIG   = 1;
}
```

To enable a source, without periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 2 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is set while CHCFG[TRIG] is cleared.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0x00 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
```

```
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;
*CHCFG1 = 0x85;
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the appropriate section for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00 to CHCFG8.
3. Write 0x87 to CHCFG8. (In this example, setting CHCFG[TRIG] would have no effect due to the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
```

```
In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;
*CHCFG8 = 0x87;
```

Chapter 23
Direct Memory Access Controller (eDMA)

23.1 Introduction

The enhanced direct memory access (eDMA) controller is a second-generation module capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 16 channels

23.1.1 eDMA system block diagram

Figure 23-1 illustrates the components of the eDMA system, including the eDMA module ("engine").

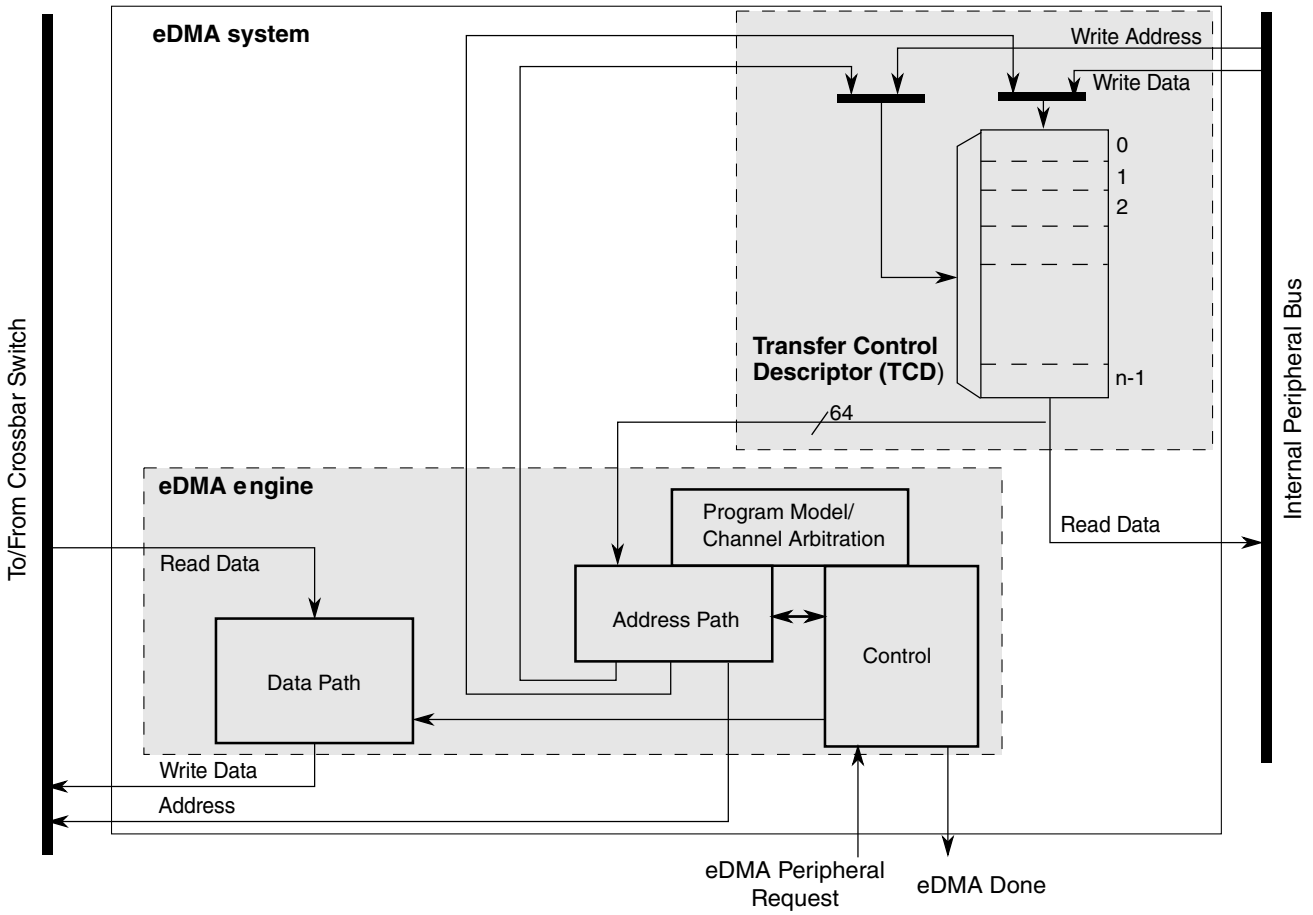


Figure 23-1. eDMA system block diagram

23.1.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer-control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 23-1. eDMA engine submodules

Submodule	Function
Address path	This block implements registered versions of two channel transfer control descriptors, channel x and channel y, and manages all master bus-address calculations. All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the first channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by DCHPRI*n*[ECP]) where a large data move operation can be preempted to minimize the time another channel is blocked from execution. When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes

Table continues on the next page...

Table 23-1. eDMA engine submodules (continued)

Submodule	Function
	the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of the next TCD*n* from memory as part of a scatter/gather operation.
Data path	This block implements the bus master read/write datapath. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the 1st stage of the bus pipeline (address phase), while the data path module implements the 2nd stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/ destination write operations until the number of bytes specified in the minor loop byte count has moved. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, two reads are performed, then one 32-bit write.

The transfer-control descriptor local memory is further partitioned into:

Table 23-2. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in the event of simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

23.1.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for enhanced addressing modes

- 16-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage to support 16- and 32-byte transfers
  - Connections to the crossbar switch for bus mastering the data movement
- Transfer control descriptor (TCD) organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

23.2 Modes of operation

The eDMA operates in the following modes:

Table 23-3. Modes of operation

Mode	Description
Normal	In Normal mode, the eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with the eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the transfer control descriptor (TCD). The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	DMA operation is configurable in Debug mode via the control register: • If CR[EDBG] is cleared, the DMA continues to operate. • If CR[EDBG] is set, the eDMA stops transferring data. If Debug mode is entered while a channel is active, the eDMA continues operation until the channel retires.
Wait	Before entering Wait mode, the DMA attempts to complete its current transfer. After the transfer completes, the device enters Wait mode.

23.3 Memory map/register definition

The eDMA's programming model is partitioned into two regions:

- The first region defines a number of registers providing control functions
- The second region corresponds to the local transfer control descriptor (TCD) memory

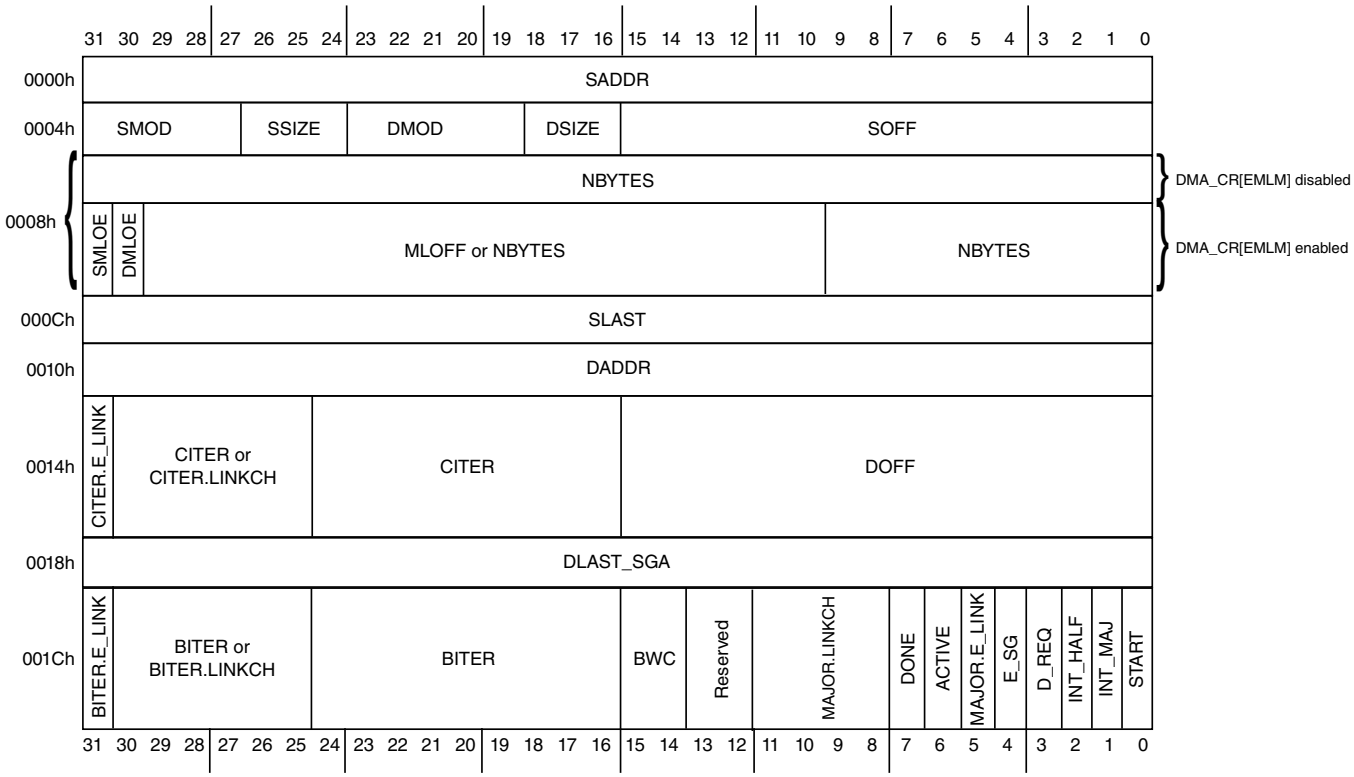
23.3.1 TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the desired data movement operation. The channel descriptors are stored in the local memory in sequential order: channel 0, channel 1, ... channel 15. Each TCD*n* definition is presented as 11 registers of 16 or 32 bits.

23.3.2 TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

23.3.3 TCD structure



23.3.4 Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

DMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_8000	Control Register (DMA_CR)	32	R/W	0000_0000h	23.3.5/383
4000_8004	Error Status Register (DMA_ES)	32	R	0000_0000h	23.3.6/386
4000_800C	Enable Request Register (DMA_ERQ)	32	R/W	0000_0000h	23.3.7/388
4000_8014	Enable Error Interrupt Register (DMA_EEI)	32	R/W	0000_0000h	23.3.8/390
4000_8018	Clear Enable Error Interrupt Register (DMA_CEEI)	8	W (always reads 0)	00h	23.3.9/393
4000_8019	Set Enable Error Interrupt Register (DMA_SEEI)	8	W (always reads 0)	00h	23.3.10/394

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_801A	Clear Enable Request Register (DMA_CERQ)	8	W (always reads 0)	00h	23.3.11/394
4000_801B	Set Enable Request Register (DMA_SERQ)	8	W (always reads 0)	00h	23.3.12/395
4000_801C	Clear DONE Status Bit Register (DMA_CDNE)	8	W (always reads 0)	00h	23.3.13/396
4000_801D	Set START Bit Register (DMA_SSRT)	8	W (always reads 0)	00h	23.3.14/397
4000_801E	Clear Error Register (DMA_CERR)	8	W (always reads 0)	00h	23.3.15/398
4000_801F	Clear Interrupt Request Register (DMA_CINT)	8	W (always reads 0)	00h	23.3.16/399
4000_8024	Interrupt Request Register (DMA_INT)	32	R/W	0000_0000h	23.3.17/400
4000_802C	Error Register (DMA_ERR)	32	R/W	0000_0000h	23.3.18/402
4000_8034	Hardware Request Status Register (DMA_HRS)	32	R	0000_0000h	23.3.19/405
4000_8044	Enable Asynchronous Request in Stop Register (DMA_EARS)	32	R/W	0000_0000h	23.3.20/408
4000_8100	Channel n Priority Register (DMA_DCHPRI3)	8	R/W	See section	23.3.21/410
4000_8101	Channel n Priority Register (DMA_DCHPRI2)	8	R/W	See section	23.3.21/410
4000_8102	Channel n Priority Register (DMA_DCHPRI1)	8	R/W	See section	23.3.21/410
4000_8103	Channel n Priority Register (DMA_DCHPRI0)	8	R/W	See section	23.3.21/410
4000_8104	Channel n Priority Register (DMA_DCHPRI7)	8	R/W	See section	23.3.21/410
4000_8105	Channel n Priority Register (DMA_DCHPRI6)	8	R/W	See section	23.3.21/410
4000_8106	Channel n Priority Register (DMA_DCHPRI5)	8	R/W	See section	23.3.21/410
4000_8107	Channel n Priority Register (DMA_DCHPRI4)	8	R/W	See section	23.3.21/410
4000_8108	Channel n Priority Register (DMA_DCHPRI11)	8	R/W	See section	23.3.21/410
4000_8109	Channel n Priority Register (DMA_DCHPRI10)	8	R/W	See section	23.3.21/410
4000_810A	Channel n Priority Register (DMA_DCHPRI9)	8	R/W	See section	23.3.21/410
4000_810B	Channel n Priority Register (DMA_DCHPRI8)	8	R/W	See section	23.3.21/410
4000_810C	Channel n Priority Register (DMA_DCHPRI15)	8	R/W	See section	23.3.21/410
4000_810D	Channel n Priority Register (DMA_DCHPRI14)	8	R/W	See section	23.3.21/410
4000_810E	Channel n Priority Register (DMA_DCHPRI13)	8	R/W	See section	23.3.21/410
4000_810F	Channel n Priority Register (DMA_DCHPRI12)	8	R/W	See section	23.3.21/410
4000_9000	TCD Source Address (DMA_TCD0_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9004	TCD Signed Source Address Offset (DMA_TCD0_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9006	TCD Transfer Attributes (DMA_TCD0_ATTR)	16	R/W	Undefined	23.3.24/412

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9008	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD0_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD0_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD0_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_900C	TCD Last Source Address Adjustment (DMA_TCD0_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9010	TCD Destination Address (DMA_TCD0_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9014	TCD Signed Destination Address Offset (DMA_TCD0_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9016	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9016	DMA_TCD0_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9018	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD0_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_901C	TCD Control and Status (DMA_TCD0_CSR)	16	R/W	Undefined	23.3.34/421
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD0_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9020	TCD Source Address (DMA_TCD1_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9024	TCD Signed Source Address Offset (DMA_TCD1_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9026	TCD Transfer Attributes (DMA_TCD1_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9028	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD1_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD1_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD1_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_902C	TCD Last Source Address Adjustment (DMA_TCD1_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9030	TCD Destination Address (DMA_TCD1_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9034	TCD Signed Destination Address Offset (DMA_TCD1_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9036	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9036	DMA_TCD1_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9038	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD1_DLASTSGA)	32	R/W	Undefined	23.3.33/420

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_903C	TCD Control and Status (DMA_TCD1_CSR)	16	R/W	Undefined	23.3.34/421
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD1_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9040	TCD Source Address (DMA_TCD2_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9044	TCD Signed Source Address Offset (DMA_TCD2_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9046	TCD Transfer Attributes (DMA_TCD2_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9048	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD2_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD2_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD2_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_904C	TCD Last Source Address Adjustment (DMA_TCD2_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9050	TCD Destination Address (DMA_TCD2_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9054	TCD Signed Destination Address Offset (DMA_TCD2_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9056	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9056	DMA_TCD2_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9058	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD2_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_905C	TCD Control and Status (DMA_TCD2_CSR)	16	R/W	Undefined	23.3.34/421
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD2_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9060	TCD Source Address (DMA_TCD3_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9064	TCD Signed Source Address Offset (DMA_TCD3_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9066	TCD Transfer Attributes (DMA_TCD3_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9068	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD3_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD3_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD3_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_906C	TCD Last Source Address Adjustment (DMA_TCD3_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9070	TCD Destination Address (DMA_TCD3_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9074	TCD Signed Destination Address Offset (DMA_TCD3_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9076	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9076	DMA_TCD3_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9078	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD3_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_907C	TCD Control and Status (DMA_TCD3_CSR)	16	R/W	Undefined	23.3.34/421
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD3_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9080	TCD Source Address (DMA_TCD4_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9084	TCD Signed Source Address Offset (DMA_TCD4_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9086	TCD Transfer Attributes (DMA_TCD4_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9088	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD4_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD4_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD4_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_908C	TCD Last Source Address Adjustment (DMA_TCD4_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9090	TCD Destination Address (DMA_TCD4_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9094	TCD Signed Destination Address Offset (DMA_TCD4_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9096	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9096	DMA_TCD4_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9098	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD4_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_909C	TCD Control and Status (DMA_TCD4_CSR)	16	R/W	Undefined	23.3.34/421
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD4_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_90A0	TCD Source Address (DMA_TCD5_SADDR)	32	R/W	Undefined	23.3.22/411

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90A4	TCD Signed Source Address Offset (DMA_TCD5_SOFF)	16	R/W	Undefined	23.3.23/411
4000_90A6	TCD Transfer Attributes (DMA_TCD5_ATTR)	16	R/W	Undefined	23.3.24/412
4000_90A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD5_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD5_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD5_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_90AC	TCD Last Source Address Adjustment (DMA_TCD5_SLAST)	32	R/W	Undefined	23.3.28/416
4000_90B0	TCD Destination Address (DMA_TCD5_DADDR)	32	R/W	Undefined	23.3.29/417
4000_90B4	TCD Signed Destination Address Offset (DMA_TCD5_DOFF)	16	R/W	Undefined	23.3.30/417
4000_90B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_90B6	DMA_TCD5_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_90B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD5_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_90BC	TCD Control and Status (DMA_TCD5_CSR)	16	R/W	Undefined	23.3.34/421
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD5_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_90C0	TCD Source Address (DMA_TCD6_SADDR)	32	R/W	Undefined	23.3.22/411
4000_90C4	TCD Signed Source Address Offset (DMA_TCD6_SOFF)	16	R/W	Undefined	23.3.23/411
4000_90C6	TCD Transfer Attributes (DMA_TCD6_ATTR)	16	R/W	Undefined	23.3.24/412
4000_90C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD6_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD6_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD6_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_90CC	TCD Last Source Address Adjustment (DMA_TCD6_SLAST)	32	R/W	Undefined	23.3.28/416
4000_90D0	TCD Destination Address (DMA_TCD6_DADDR)	32	R/W	Undefined	23.3.29/417
4000_90D4	TCD Signed Destination Address Offset (DMA_TCD6_DOFF)	16	R/W	Undefined	23.3.30/417
4000_90D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_90D6	DMA_TCD6_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD6_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_90DC	TCD Control and Status (DMA_TCD6_CSR)	16	R/W	Undefined	23.3.34/421
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD6_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_90E0	TCD Source Address (DMA_TCD7_SADDR)	32	R/W	Undefined	23.3.22/411
4000_90E4	TCD Signed Source Address Offset (DMA_TCD7_SOFF)	16	R/W	Undefined	23.3.23/411
4000_90E6	TCD Transfer Attributes (DMA_TCD7_ATTR)	16	R/W	Undefined	23.3.24/412
4000_90E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD7_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD7_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD7_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_90EC	TCD Last Source Address Adjustment (DMA_TCD7_SLAST)	32	R/W	Undefined	23.3.28/416
4000_90F0	TCD Destination Address (DMA_TCD7_DADDR)	32	R/W	Undefined	23.3.29/417
4000_90F4	TCD Signed Destination Address Offset (DMA_TCD7_DOFF)	16	R/W	Undefined	23.3.30/417
4000_90F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_90F6	DMA_TCD7_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_90F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD7_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_90FC	TCD Control and Status (DMA_TCD7_CSR)	16	R/W	Undefined	23.3.34/421
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD7_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9100	TCD Source Address (DMA_TCD8_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9104	TCD Signed Source Address Offset (DMA_TCD8_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9106	TCD Transfer Attributes (DMA_TCD8_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9108	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD8_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9108	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD8_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9108	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD8_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_910C	TCD Last Source Address Adjustment (DMA_TCD8_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9110	TCD Destination Address (DMA_TCD8_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9114	TCD Signed Destination Address Offset (DMA_TCD8_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9116	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9116	DMA_TCD8_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9118	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD8_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_911C	TCD Control and Status (DMA_TCD8_CSR)	16	R/W	Undefined	23.3.34/421
4000_911E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_911E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD8_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9120	TCD Source Address (DMA_TCD9_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9124	TCD Signed Source Address Offset (DMA_TCD9_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9126	TCD Transfer Attributes (DMA_TCD9_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9128	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD9_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9128	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD9_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9128	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD9_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_912C	TCD Last Source Address Adjustment (DMA_TCD9_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9130	TCD Destination Address (DMA_TCD9_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9134	TCD Signed Destination Address Offset (DMA_TCD9_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9136	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9136	DMA_TCD9_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9138	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD9_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_913C	TCD Control and Status (DMA_TCD9_CSR)	16	R/W	Undefined	23.3.34/421
4000_913E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_913E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD9_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9140	TCD Source Address (DMA_TCD10_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9144	TCD Signed Source Address Offset (DMA_TCD10_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9146	TCD Transfer Attributes (DMA_TCD10_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9148	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD10_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9148	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD10_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9148	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD10_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_914C	TCD Last Source Address Adjustment (DMA_TCD10_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9150	TCD Destination Address (DMA_TCD10_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9154	TCD Signed Destination Address Offset (DMA_TCD10_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9156	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9156	DMA_TCD10_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9158	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD10_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_915C	TCD Control and Status (DMA_TCD10_CSR)	16	R/W	Undefined	23.3.34/421
4000_915E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_915E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD10_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9160	TCD Source Address (DMA_TCD11_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9164	TCD Signed Source Address Offset (DMA_TCD11_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9166	TCD Transfer Attributes (DMA_TCD11_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9168	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD11_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9168	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD11_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9168	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD11_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_916C	TCD Last Source Address Adjustment (DMA_TCD11_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9170	TCD Destination Address (DMA_TCD11_DADDR)	32	R/W	Undefined	23.3.29/417

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9174	TCD Signed Destination Address Offset (DMA_TCD11_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9176	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9176	DMA_TCD11_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9178	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD11_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_917C	TCD Control and Status (DMA_TCD11_CSR)	16	R/W	Undefined	23.3.34/421
4000_917E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_917E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD11_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_9180	TCD Source Address (DMA_TCD12_SADDR)	32	R/W	Undefined	23.3.22/411
4000_9184	TCD Signed Source Address Offset (DMA_TCD12_SOFF)	16	R/W	Undefined	23.3.23/411
4000_9186	TCD Transfer Attributes (DMA_TCD12_ATTR)	16	R/W	Undefined	23.3.24/412
4000_9188	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD12_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_9188	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD12_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_9188	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD12_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_918C	TCD Last Source Address Adjustment (DMA_TCD12_SLAST)	32	R/W	Undefined	23.3.28/416
4000_9190	TCD Destination Address (DMA_TCD12_DADDR)	32	R/W	Undefined	23.3.29/417
4000_9194	TCD Signed Destination Address Offset (DMA_TCD12_DOFF)	16	R/W	Undefined	23.3.30/417
4000_9196	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_9196	DMA_TCD12_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_9198	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD12_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_919C	TCD Control and Status (DMA_TCD12_CSR)	16	R/W	Undefined	23.3.34/421
4000_919E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_919E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD12_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_91A0	TCD Source Address (DMA_TCD13_SADDR)	32	R/W	Undefined	23.3.22/411
4000_91A4	TCD Signed Source Address Offset (DMA_TCD13_SOFF)	16	R/W	Undefined	23.3.23/411

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91A6	TCD Transfer Attributes (DMA_TCD13_ATTR)	16	R/W	Undefined	23.3.24/412
4000_91A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD13_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_91A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD13_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_91A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD13_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_91AC	TCD Last Source Address Adjustment (DMA_TCD13_SLAST)	32	R/W	Undefined	23.3.28/416
4000_91B0	TCD Destination Address (DMA_TCD13_DADDR)	32	R/W	Undefined	23.3.29/417
4000_91B4	TCD Signed Destination Address Offset (DMA_TCD13_DOFF)	16	R/W	Undefined	23.3.30/417
4000_91B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_91B6	DMA_TCD13_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_91B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD13_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_91BC	TCD Control and Status (DMA_TCD13_CSR)	16	R/W	Undefined	23.3.34/421
4000_91BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_91BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD13_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_91C0	TCD Source Address (DMA_TCD14_SADDR)	32	R/W	Undefined	23.3.22/411
4000_91C4	TCD Signed Source Address Offset (DMA_TCD14_SOFF)	16	R/W	Undefined	23.3.23/411
4000_91C6	TCD Transfer Attributes (DMA_TCD14_ATTR)	16	R/W	Undefined	23.3.24/412
4000_91C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD14_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_91C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD14_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_91C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD14_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_91CC	TCD Last Source Address Adjustment (DMA_TCD14_SLAST)	32	R/W	Undefined	23.3.28/416
4000_91D0	TCD Destination Address (DMA_TCD14_DADDR)	32	R/W	Undefined	23.3.29/417
4000_91D4	TCD Signed Destination Address Offset (DMA_TCD14_DOFF)	16	R/W	Undefined	23.3.30/417
4000_91D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_91D6	DMA_TCD14_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD14_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_91DC	TCD Control and Status (DMA_TCD14_CSR)	16	R/W	Undefined	23.3.34/421
4000_91DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_91DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD14_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424
4000_91E0	TCD Source Address (DMA_TCD15_SADDR)	32	R/W	Undefined	23.3.22/411
4000_91E4	TCD Signed Source Address Offset (DMA_TCD15_SOFF)	16	R/W	Undefined	23.3.23/411
4000_91E6	TCD Transfer Attributes (DMA_TCD15_ATTR)	16	R/W	Undefined	23.3.24/412
4000_91E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD15_NBYTES_MLNO)	32	R/W	Undefined	23.3.25/413
4000_91E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD15_NBYTES_MLOFFNO)	32	R/W	Undefined	23.3.26/414
4000_91E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD15_NBYTES_MLOFFYES)	32	R/W	Undefined	23.3.27/415
4000_91EC	TCD Last Source Address Adjustment (DMA_TCD15_SLAST)	32	R/W	Undefined	23.3.28/416
4000_91F0	TCD Destination Address (DMA_TCD15_DADDR)	32	R/W	Undefined	23.3.29/417
4000_91F4	TCD Signed Destination Address Offset (DMA_TCD15_DOFF)	16	R/W	Undefined	23.3.30/417
4000_91F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_CITER_ELINKYES)	16	R/W	Undefined	23.3.31/418
4000_91F6	DMA_TCD15_CITER_ELINKNO	16	R/W	Undefined	23.3.32/419
4000_91F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD15_DLASTSGA)	32	R/W	Undefined	23.3.33/420
4000_91FC	TCD Control and Status (DMA_TCD15_CSR)	16	R/W	Undefined	23.3.34/421
4000_91FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_BITER_ELINKYES)	16	R/W	Undefined	23.3.35/423
4000_91FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD15_BITER_ELINKNO)	16	R/W	Undefined	23.3.36/424

23.3.5 Control Register (DMA_CR)

The CR defines the basic operating configuration of the DMA.

Arbitration can be configured to use either a fixed-priority or a round-robin scheme. For fixed-priority arbitration, the highest priority channel requesting service is selected to execute. The channel priority registers assign the priorities; see the DCHPRIn registers. For round-robin arbitration, the channel priorities are ignored and channels are cycled through (from high to low channel number) without regard to priority.

NOTE

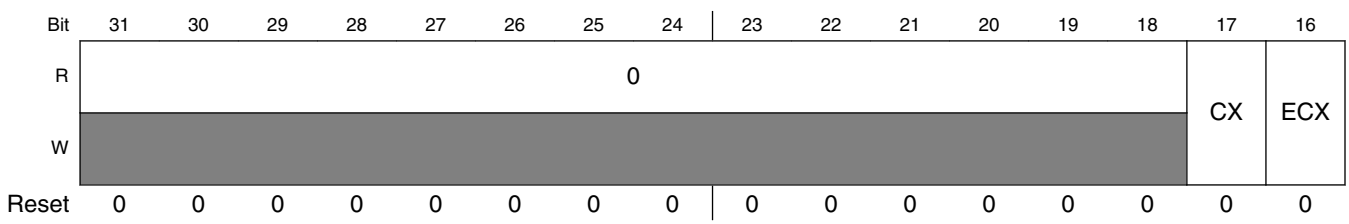
For correct operation, writes to the CR register must be performed only when the DMA channels are inactive; that is, when TCDn_CSR[ACTIVE] bits are cleared.

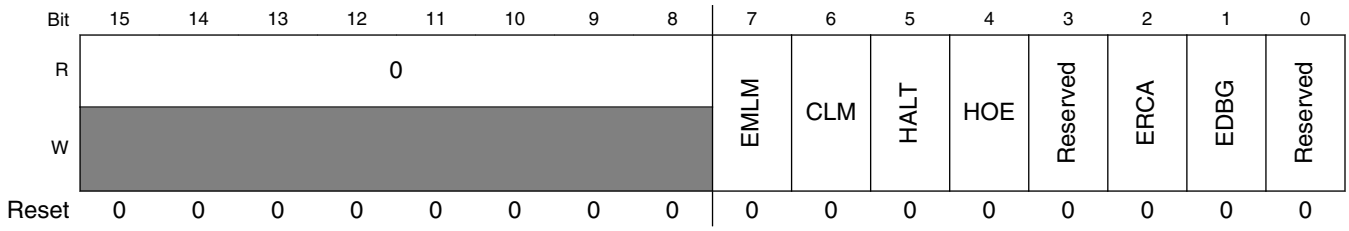
Minor loop offsets are address offset values added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. When minor loop offsets are enabled, the minor loop offset (MLOFF) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both prior to the addresses being written back into the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (EMLM is 1), TCDn word2 is redefined. A portion of TCDn word2 is used to specify multiple fields: a source enable bit (SMLOE) to specify the minor loop offset should be applied to the source address (TCDn_SADDR) upon minor loop completion, a destination enable bit (DMLOE) to specify the minor loop offset should be applied to the destination address (TCDn_DADDR) upon minor loop completion, and the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

When minor loop mapping is disabled (EMLM is 0), all 32 bits of TCDn word2 are assigned to the NBYTES field.

Address: 4000_8000h base + 0h offset = 4000_8000h





DMA_CR field descriptions

Field	Description
31–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 CX	Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The CX bit clears itself after the cancel has been honored. This cancel retires the channel normally as if the minor loop was completed.
16 ECX	Error Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer in the same fashion as the CX bit. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The ECX bit clears itself after the cancel is honored. In addition to cancelling the transfer, ECX treats the cancel as an error condition, thus updating the Error Status register (DMAx_ES) and generating an optional error interrupt.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 EMLM	Enable Minor Loop Mapping 0 Disabled. TCDn.word2 is defined as a 32-bit NBYTES field. 1 Enabled. TCDn.word2 is redefined to include individual enable fields, an offset field, and the NBYTES field. The individual enable fields allow the minor loop offset to be applied to the source address, the destination address, or both. The NBYTES field is reduced when either offset is enabled.
6 CLM	Continuous Link Mode **NOTE:** Do not use continuous link mode with a channel linking to itself if there is only one minor loop iteration per service request, e.g., if the channel's NBYTES value is the same as either the source or destination size. The same data transfer profile can be achieved by simply increasing the NBYTES value, which provides more efficient, faster processing. 0 A minor loop channel link made to itself goes through channel arbitration before being activated again. 1 A minor loop channel link made to itself does not go through channel arbitration before being activated again. Upon minor loop completion, the channel activates again if that channel has a minor loop channel link enabled and the link channel is itself. This effectively applies the minor loop offsets and restarts the next minor loop.
5 HALT	Halt DMA Operations 0 Normal operation 1 Stall the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this bit is cleared.

Table continues on the next page...

DMA_CR field descriptions (continued)

Field	Description
4 HOE	Halt On Error 0 Normal operation 1 Any error causes the HALT bit to set. Subsequently, all service requests are ignored until the HALT bit is cleared.
3 Reserved	This field is reserved. Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0 Fixed priority arbitration is used for channel selection . 1 Round robin arbitration is used for channel selection .
1 EDBG	Enable Debug 0 When in debug mode, the DMA continues to operate. 1 When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the system exits debug mode or the EDBG bit is cleared.
0 Reserved	This field is reserved. Reserved

23.3.6 Error Status Register (DMA_ES)

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- A configuration error, that is:
  - An illegal setting in the transfer-control descriptor, or
  - An illegal priority register setting in fixed-arbitration
- An error termination to a bus master read or write cycle
- A cancel transfer with error bit that will be set when a transfer is canceled via the corresponding cancel transfer control bit

See Fault reporting and handling for more details.

Address: 4000_8000h base + 4h offset = 4000_8004h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CPE	0		ERRCHN				SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_ES field descriptions

Field	Description
31 VLD	Logical OR of all ERR status bits 0 No ERR bits are set. 1 At least one ERR bit is set indicating a valid error exists that has not been cleared.
30–17 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 ECX	Transfer Canceled 0 No canceled transfers 1 The last recorded entry was a canceled transfer by the error cancel transfer input
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 CPE	Channel Priority Error 0 No channel priority error 1 The last recorded error was a configuration error in the channel priorities . Channel priorities are not unique.
13–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error, excluding CPE errors, or last recorded error canceled transfer.
7 SAE	Source Address Error 0 No source address configuration error. 1 The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].
6 SOE	Source Offset Error 0 No source offset configuration error 1 The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].
5 DAE	Destination Address Error 0 No destination address configuration error 1 The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].
4 DOE	Destination Offset Error 0 No destination offset configuration error 1 The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].
3 NCE	NBYTES/CITER Configuration Error 0 No NBYTES/CITER configuration error 1 The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields. • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], or

Table continues on the next page...

DMA_ES field descriptions (continued)

Field	Description
	• TCDn_CITER[CITER] is equal to zero, or • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0 No scatter/gather configuration error 1 The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.
1 SBE	Source Bus Error 0 No source bus error 1 The last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0 No destination bus error 1 The last recorded error was a bus error on a destination write

23.3.7 Enable Request Register (DMA_ERQ)

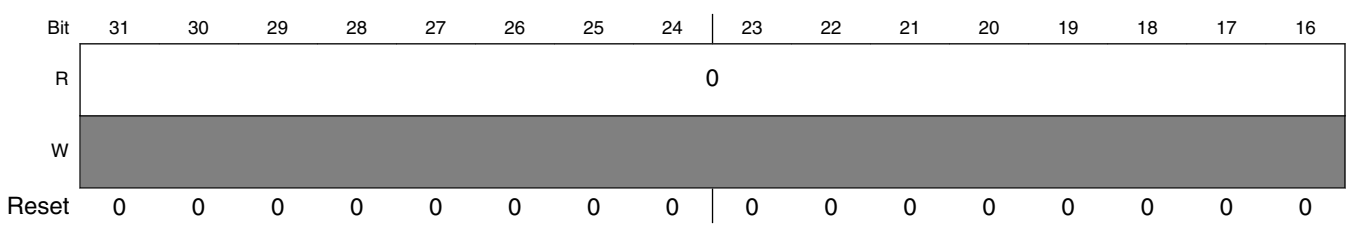
The ERQ register provides a bit map for the 16 channels to enable the request signal for each channel. The state of any given channel enable is directly affected by writes to this register; it is also affected by writes to the SERQ and CERQ registers. These registers are provided so the request enable for a single channel can easily be modified without needing to perform a read-modify-write sequence to the ERQ.

DMA request input signals and this enable request flag must be asserted before a channel's hardware service request is accepted. The state of the DMA enable request flag does not affect a channel service request made explicitly through software or a linked channel request.

NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

Address: 4000_8000h base + Ch offset = 4000_800Ch



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ERQ15	ERQ14	ERQ13	ERQ12	ERQ11	ERQ10	ERQ9	ERQ8	ERQ7	ERQ6	ERQ5	ERQ4	ERQ3	ERQ2	ERQ1	ERQ0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_ERQ field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 ERQ15	Enable DMA Request 15 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
14 ERQ14	Enable DMA Request 14 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
13 ERQ13	Enable DMA Request 13 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
12 ERQ12	Enable DMA Request 12 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
11 ERQ11	Enable DMA Request 11 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
10 ERQ10	Enable DMA Request 10 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
9 ERQ9	Enable DMA Request 9 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
8 ERQ8	Enable DMA Request 8 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
7 ERQ7	Enable DMA Request 7 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
6 ERQ6	Enable DMA Request 6

Table continues on the next page...

DMA_ERQ field descriptions (continued)

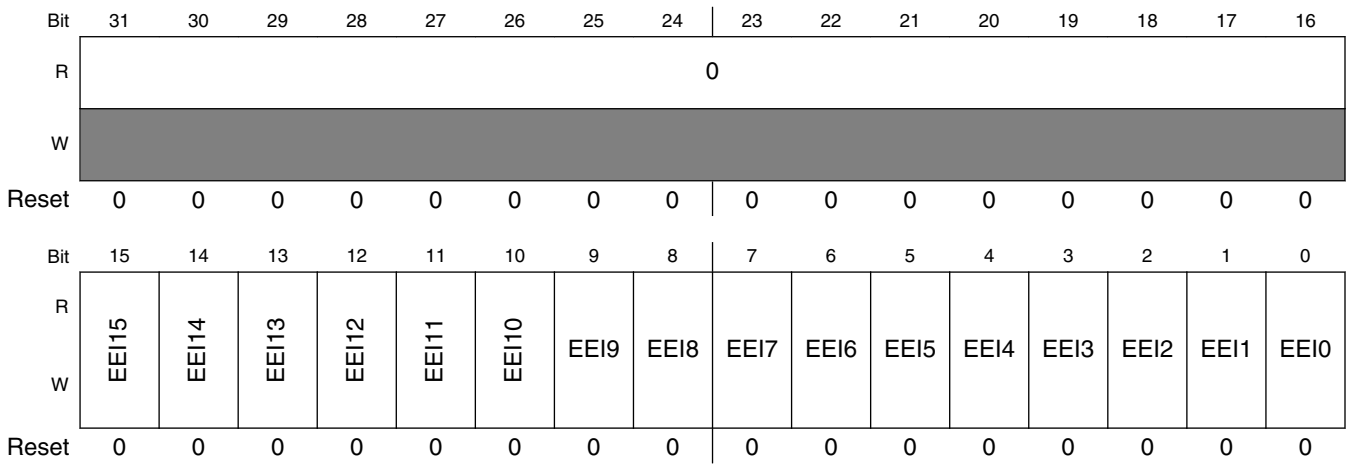
Field	Description
	0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
5 ERQ5	Enable DMA Request 5 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
4 ERQ4	Enable DMA Request 4 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
3 ERQ3	Enable DMA Request 3 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
2 ERQ2	Enable DMA Request 2 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
1 ERQ1	Enable DMA Request 1 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
0 ERQ0	Enable DMA Request 0 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

23.3.8 Enable Error Interrupt Register (DMA_EEI)

The EEI register provides a bit map for the 16 channels to enable the error interrupt signal for each channel. The state of any given channel's error interrupt enable is directly affected by writes to this register; it is also affected by writes to the SEEI and CEEI. These registers are provided so that the error interrupt enable for a single channel can easily be modified without the need to perform a read-modify-write sequence to the EEI register.

The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller.

Address: 4000_8000h base + 14h offset = 4000_8014h



DMA_EEI field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 EEI15	Enable Error Interrupt 15 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
14 EEI14	Enable Error Interrupt 14 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
13 EEI13	Enable Error Interrupt 13 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
12 EEI12	Enable Error Interrupt 12 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
11 EEI11	Enable Error Interrupt 11 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
10 EEI10	Enable Error Interrupt 10 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
9 EEI9	Enable Error Interrupt 9 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
8 EEI8	Enable Error Interrupt 8

Table continues on the next page...

DMA_EEI field descriptions (continued)

Field	Description
	0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
7 EEI7	Enable Error Interrupt 7 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
6 EEI6	Enable Error Interrupt 6 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
5 EEI5	Enable Error Interrupt 5 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
4 EEI4	Enable Error Interrupt 4 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
3 EEI3	Enable Error Interrupt 3 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
2 EEI2	Enable Error Interrupt 2 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
1 EEI1	Enable Error Interrupt 1 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
0 EEI0	Enable Error Interrupt 0 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

23.3.9 Clear Enable Error Interrupt Register (DMA_CEEI)

The CEEI provides a simple memory-mapped mechanism to clear a given bit in the EEI to disable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be cleared. Setting the CAEE bit provides a global clear function, forcing the EEI contents to be cleared, disabling all DMA request inputs. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 18h offset = 4000_8018h

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CAEE	0		CEEI			
Reset	0	0	0	0	0	0	0	0

DMA_CEEI field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEE	Clear All Enable Error Interrupts 0 Clear only the EEI bit specified in the CEEI field 1 Clear all bits in EEI
5–4 Reserved	This field is reserved.
CEEI	Clear Enable Error Interrupt Clears the corresponding bit in EEI

23.3.10 Set Enable Error Interrupt Register (DMA_SEEI)

The SEEI provides a simple memory-mapped mechanism to set a given bit in the EEI to enable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be set. Setting the SAEE bit provides a global set function, forcing the entire EEI contents to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 19h offset = 4000_8019h

Bit	7	6	5	4	3	2	1	0
Read	0	0	0					
Write	NOP	SAEE	0		SEEI			
Reset	0	0	0	0	0	0	0	0

DMA_SEEI field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAEE	Sets All Enable Error Interrupts 0 Set only the EEI bit specified in the SEEI field. 1 Sets all bits in EEI
5–4 Reserved	This field is reserved.
SEEI	Set Enable Error Interrupt Sets the corresponding bit in EEI

23.3.11 Clear Enable Request Register (DMA_CERQ)

The CERQ provides a simple memory-mapped mechanism to clear a given bit in the ERQ to disable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be cleared. Setting the CAER bit provides a global clear function, forcing the entire contents of the ERQ to be cleared, disabling all DMA request inputs. If NOP is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

Address: 4000_8000h base + 1Ah offset = 4000_801Ah

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CAER	0		CERQ			
Reset	0	0	0	0	0	0	0	0

DMA_CERQ field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAER	Clear All Enable Requests 0 Clear only the ERQ bit specified in the CERQ field 1 Clear all bits in ERQ
5–4 Reserved	This field is reserved.
CERQ	Clear Enable Request Clears the corresponding bit in ERQ.

23.3.12 Set Enable Request Register (DMA_SERQ)

The SERQ provides a simple memory-mapped mechanism to set a given bit in the ERQ to enable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be set. Setting the SAER bit provides a global set function, forcing the entire contents of ERQ to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Bh offset = 4000_801Bh

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	SAER	0		SERQ			
Reset	0	0	0	0	0	0	0	0

DMA_SERQ field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAER	Set All Enable Requests 0 Set only the ERQ bit specified in the SERQ field 1 Set all bits in ERQ
5–4 Reserved	This field is reserved.
SERQ	Set Enable Request Sets the corresponding bit in ERQ.

23.3.13 Clear DONE Status Bit Register (DMA_CDNE)

The CDNE provides a simple memory-mapped mechanism to clear the DONE bit in the TCD of the given channel. The data value on a register write causes the DONE bit in the corresponding transfer control descriptor to be cleared. Setting the CADN bit provides a global clear function, forcing all DONE bits to be cleared. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Ch offset = 4000_801Ch

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CADN	0		CDNE			
Reset	0	0	0	0	0	0	0	0

DMA_CDNE field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CADN	Clears All DONE Bits 0 Clears only the TCDn_CSR[DONE] bit specified in the CDNE field 1 Clears all bits in TCDn_CSR[DONE]
5–4 Reserved	This field is reserved.

Table continues on the next page...

DMA_CDNE field descriptions (continued)

Field	Description
CDNE	Clear DONE Bit Clears the corresponding bit in TCDn_CSR[DONE]

23.3.14 Set START Bit Register (DMA_SSRT)

The SSRT provides a simple memory-mapped mechanism to set the START bit in the TCD of the given channel. The data value on a register write causes the START bit in the corresponding transfer control descriptor to be set. Setting the SAST bit provides a global set function, forcing all START bits to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Dh offset = 4000_801Dh

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	SAST	0		SSRT			
Reset	0	0	0	0	0	0	0	0

DMA_SSRT field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 SAST	Set All START Bits (activates all channels) 0 Set only the TCDn_CSR[START] bit specified in the SSRT field 1 Set all bits in TCDn_CSR[START]
5–4 Reserved	This field is reserved.
SSRT	Set START Bit Sets the corresponding bit in TCDn_CSR[START]

23.3.15 Clear Error Register (DMA_CERR)

The CERR provides a simple memory-mapped mechanism to clear a given bit in the ERR to disable the error condition flag for a given channel. The given value on a register write causes the corresponding bit in the ERR to be cleared. Setting the CAEI bit provides a global clear function, forcing the ERR contents to be cleared, clearing all channel error indicators. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Eh offset = 4000_801Eh

Bit	7	6	5	4	3	2	1	0
Read	0	0			0			
Write	NOP	CAEI	0		CERR			
Reset	0	0	0	0	0	0	0	0

DMA_CERR field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEI	Clear All Error Indicators 0 Clear only the ERR bit specified in the CERR field 1 Clear all bits in ERR
5–4 Reserved	This field is reserved.
CERR	Clear Error Indicator Clears the corresponding bit in ERR

23.3.16 Clear Interrupt Request Register (DMA_CINT)

The CINT provides a simple, memory-mapped mechanism to clear a given bit in the INT to disable the interrupt request for a given channel. The given value on a register write causes the corresponding bit in the INT to be cleared. Setting the CAIR bit provides a global clear function, forcing the entire contents of the INT to be cleared, disabling all DMA interrupt requests. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 4000_8000h base + 1Fh offset = 4000_801Fh

Bit	7	6	5	4	3	2	1	0
Read	0	0	0					
Write	NOP	CAIR	0		CINT			
Reset	0	0	0	0	0	0	0	0

DMA_CINT field descriptions

Field	Description
7 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
6 CAIR	Clear All Interrupt Requests 0 Clear only the INT bit specified in the CINT field 1 Clear all bits in INT
5–4 Reserved	This field is reserved.
CINT	Clear Interrupt Request Clears the corresponding bit in INT

23.3.17 Interrupt Request Register (DMA_INT)

The INT register provides a bit map for the 16 channels signaling the presence of an interrupt request for each channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion. The outputs of this register are directly routed to the interrupt controller. During the interrupt-service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. Typically, a write to the CINT register in the interrupt service routine is used for this purpose.

The state of any given channel's interrupt request is directly affected by writes to this register; it is also affected by writes to the CINT register. On writes to INT, a 1 in any bit position clears the corresponding channel's interrupt request. A zero in any bit position has no affect on the corresponding channel's current interrupt status. The CINT register is provided so the interrupt request for a single channel can easily be cleared without the need to perform a read-modify-write sequence to the INT register.

Address: 4000_8000h base + 24h offset = 4000_8024h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_INT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

DMA_INT field descriptions (continued)

Field	Description
15 INT15	Interrupt Request 15 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
14 INT14	Interrupt Request 14 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
13 INT13	Interrupt Request 13 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
12 INT12	Interrupt Request 12 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
11 INT11	Interrupt Request 11 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
10 INT10	Interrupt Request 10 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
9 INT9	Interrupt Request 9 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
8 INT8	Interrupt Request 8 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
7 INT7	Interrupt Request 7 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
6 INT6	Interrupt Request 6 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
5 INT5	Interrupt Request 5 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
4 INT4	Interrupt Request 4 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

Table continues on the next page...

DMA_INT field descriptions (continued)

Field	Description
3 INT3	Interrupt Request 3 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
2 INT2	Interrupt Request 2 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
1 INT1	Interrupt Request 1 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
0 INT0	Interrupt Request 0 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

23.3.18 Error Register (DMA_ERR)

The ERR provides a bit map for the 16 channels, signaling the presence of an error for each channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate bit in this register. The outputs of this register are enabled by the contents of the EEI, and then routed to the interrupt controller. During the execution of the interrupt-service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. Typically, a write to the CERR in the interrupt-service routine is used for this purpose. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when an error is detected.

The contents of this register can also be polled because a non-zero value indicates the presence of a channel error regardless of the state of the EEI. The state of any given channel's error indicators is affected by writes to this register; it is also affected by writes to the CERR. On writes to the ERR, a one in any bit position clears the corresponding channel's error status. A zero in any bit position has no affect on the corresponding channel's current error status. The CERR is provided so the error indicator for a single channel can easily be cleared.

Address: 4000_8000h base + 2Ch offset = 4000_802Ch

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR15	ERR14	ERR13	ERR12	ERR11	ERR10	ERR9	ERR8	ERR7	ERR6	ERR5	ERR4	ERR3	ERR2	ERR1	ERR0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_ERR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 ERR15	Error In Channel 15 0 An error in this channel has not occurred 1 An error in this channel has occurred
14 ERR14	Error In Channel 14 0 An error in this channel has not occurred 1 An error in this channel has occurred
13 ERR13	Error In Channel 13 0 An error in this channel has not occurred 1 An error in this channel has occurred
12 ERR12	Error In Channel 12 0 An error in this channel has not occurred 1 An error in this channel has occurred
11 ERR11	Error In Channel 11 0 An error in this channel has not occurred 1 An error in this channel has occurred
10 ERR10	Error In Channel 10 0 An error in this channel has not occurred 1 An error in this channel has occurred

Table continues on the next page...

DMA_ERR field descriptions (continued)

Field	Description
9 ERR9	Error In Channel 9 0 An error in this channel has not occurred 1 An error in this channel has occurred
8 ERR8	Error In Channel 8 0 An error in this channel has not occurred 1 An error in this channel has occurred
7 ERR7	Error In Channel 7 0 An error in this channel has not occurred 1 An error in this channel has occurred
6 ERR6	Error In Channel 6 0 An error in this channel has not occurred 1 An error in this channel has occurred
5 ERR5	Error In Channel 5 0 An error in this channel has not occurred 1 An error in this channel has occurred
4 ERR4	Error In Channel 4 0 An error in this channel has not occurred 1 An error in this channel has occurred
3 ERR3	Error In Channel 3 0 An error in this channel has not occurred 1 An error in this channel has occurred
2 ERR2	Error In Channel 2 0 An error in this channel has not occurred 1 An error in this channel has occurred
1 ERR1	Error In Channel 1 0 An error in this channel has not occurred 1 An error in this channel has occurred
0 ERR0	Error In Channel 0 0 An error in this channel has not occurred 1 An error in this channel has occurred

23.3.19 Hardware Request Status Register (DMA_HRS)

The HRS register provides a bit map for the DMA channels, signaling the presence of a hardware request for each channel. The hardware request status bits reflect the current state of the register and qualified (via the ERQ fields) DMA request signals as seen by the DMA's arbitration logic. This view into the hardware request signals may be used for debug purposes.

NOTE

These bits reflect the state of the request as seen by the arbitration logic. Therefore, this status is affected by the ERQ bits.

Address: 4000_8000h base + 34h offset = 4000_8034h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HRS15	HRS14	HRS13	HRS12	HRS11	HRS10	HRS9	HRS8	HRS7	HRS6	HRS5	HRS4	HRS3	HRS2	HRS1	HRS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_HRS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 HRS15	Hardware Request Status Channel 15

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 15 is not present 1 A hardware service request for channel 15 is present
14 HRS14	Hardware Request Status Channel 14 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 14 is not present 1 A hardware service request for channel 14 is present
13 HRS13	Hardware Request Status Channel 13 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 13 is not present 1 A hardware service request for channel 13 is present
12 HRS12	Hardware Request Status Channel 12 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 12 is not present 1 A hardware service request for channel 12 is present
11 HRS11	Hardware Request Status Channel 11 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 11 is not present 1 A hardware service request for channel 11 is present
10 HRS10	Hardware Request Status Channel 10 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 10 is not present 1 A hardware service request for channel 10 is present
9 HRS9	Hardware Request Status Channel 9 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	0 A hardware service request for channel 9 is not present 1 A hardware service request for channel 9 is present
8 HRS8	Hardware Request Status Channel 8 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 8 is not present 1 A hardware service request for channel 8 is present
7 HRS7	Hardware Request Status Channel 7 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 7 is not present 1 A hardware service request for channel 7 is present
6 HRS6	Hardware Request Status Channel 6 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 6 is not present 1 A hardware service request for channel 6 is present
5 HRS5	Hardware Request Status Channel 5 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 5 is not present 1 A hardware service request for channel 5 is present
4 HRS4	Hardware Request Status Channel 4 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 4 is not present 1 A hardware service request for channel 4 is present
3 HRS3	Hardware Request Status Channel 3 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 3 is not present 1 A hardware service request for channel 3 is present
2 HRS2	Hardware Request Status Channel 2

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 2 is not present 1 A hardware service request for channel 2 is present
1 HRS1	Hardware Request Status Channel 1 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 1 is not present 1 A hardware service request for channel 1 is present
0 HRS0	Hardware Request Status Channel 0 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0 A hardware service request for channel 0 is not present 1 A hardware service request for channel 0 is present

23.3.20 Enable Asynchronous Request in Stop Register (DMA_EARS)

Address: 4000_8000h base + 44h offset = 4000_8044h

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EDREQ_15	EDREQ_14	EDREQ_13	EDREQ_12	EDREQ_11	EDREQ_10	EDREQ_9	EDREQ_8	EDREQ_7	EDREQ_6	EDREQ_5	EDREQ_4	EDREQ_3	EDREQ_2	EDREQ_1	EDREQ_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_EARS field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

DMA_EARS field descriptions (continued)

Field	Description
15 EDREQ_15	Enable asynchronous DMA request in stop mode for channel 15 0 Disable asynchronous DMA request for channel 15. 1 Enable asynchronous DMA request for channel 15.
14 EDREQ_14	Enable asynchronous DMA request in stop mode for channel 14 0 Disable asynchronous DMA request for channel 14. 1 Enable asynchronous DMA request for channel 14.
13 EDREQ_13	Enable asynchronous DMA request in stop mode for channel 13 0 Disable asynchronous DMA request for channel 13. 1 Enable asynchronous DMA request for channel 13.
12 EDREQ_12	Enable asynchronous DMA request in stop mode for channel 12 0 Disable asynchronous DMA request for channel 12. 1 Enable asynchronous DMA request for channel 12.
11 EDREQ_11	Enable asynchronous DMA request in stop mode for channel 11 0 Disable asynchronous DMA request for channel 11. 1 Enable asynchronous DMA request for channel 11.
10 EDREQ_10	Enable asynchronous DMA request in stop mode for channel 10 0 Disable asynchronous DMA request for channel 10. 1 Enable asynchronous DMA request for channel 10.
9 EDREQ_9	Enable asynchronous DMA request in stop mode for channel 9 0 Disable asynchronous DMA request for channel 9. 1 Enable asynchronous DMA request for channel 9.
8 EDREQ_8	Enable asynchronous DMA request in stop mode for channel 8 0 Disable asynchronous DMA request for channel 8. 1 Enable asynchronous DMA request for channel 8.
7 EDREQ_7	Enable asynchronous DMA request in stop mode for channel 7 0 Disable asynchronous DMA request for channel 7. 1 Enable asynchronous DMA request for channel 7.
6 EDREQ_6	Enable asynchronous DMA request in stop mode for channel 6 0 Disable asynchronous DMA request for channel 6. 1 Enable asynchronous DMA request for channel 6.
5 EDREQ_5	Enable asynchronous DMA request in stop mode for channel 5 0 Disable asynchronous DMA request for channel 5. 1 Enable asynchronous DMA request for channel 5.
4 EDREQ_4	Enable asynchronous DMA request in stop mode for channel 4 0 Disable asynchronous DMA request for channel 4. 1 Enable asynchronous DMA request for channel 4.

Table continues on the next page...

DMA_EARS field descriptions (continued)

Field	Description
3 EDREQ_3	Enable asynchronous DMA request in stop mode for channel 3. 0 Disable asynchronous DMA request for channel 3. 1 Enable asynchronous DMA request for channel 3.
2 EDREQ_2	Enable asynchronous DMA request in stop mode for channel 2. 0 Disable asynchronous DMA request for channel 2. 1 Enable asynchronous DMA request for channel 2.
1 EDREQ_1	Enable asynchronous DMA request in stop mode for channel 1. 0 Disable asynchronous DMA request for channel 1 1 Enable asynchronous DMA request for channel 1.
0 EDREQ_0	Enable asynchronous DMA request in stop mode for channel 0. 0 Disable asynchronous DMA request for channel 0. 1 Enable asynchronous DMA request for channel 0.

23.3.21 Channel n Priority Register (DMA_DCHPRI*n*)

When fixed-priority channel arbitration is enabled (CR[ERCA] = 0), the contents of these registers define the unique priorities associated with each channel. The channel priorities are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. Software must program the channel priorities with unique values; otherwise, a configuration error is reported. The range of the priority value is limited to the values of 0 through 15.

Address: 4000_8000h base + 100h offset + (1d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	ECP	DPA	0		CHPRI			
Write								
Reset	0	0	0	0	*	*	*	*

* Notes:
- CHPRI field: See bit field description.

DMA_DCHPRI*n* field descriptions

Field	Description
7 ECP	Enable Channel Preemption. 0 Channel n cannot be suspended by a higher priority channel's service request. 1 Channel n can be temporarily suspended by the service request of a higher priority channel.

Table continues on the next page...

DMA_DCHPRI*n* field descriptions (continued)

Field	Description
6 DPA	Disable Preempt Ability. 0 Channel n can suspend a lower priority channel. 1 Channel n cannot suspend any channel, regardless of channel priority.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CHPRI	Channel n Arbitration Priority Channel priority when fixed-priority arbitration is enabled **NOTE:** Reset value for the channel priority field, CHPRI, is equal to the corresponding channel number for each priority register, that is, DCHPRI15[CHPRI] = 0b1111.

23.3.22 TCD Source Address (DMA_TCD*n*_SADDR)

Address: 4000_8000h base + 1000h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_SADDR field descriptions

Field	Description
SADDR	Source Address Memory address pointing to the source data.

23.3.23 TCD Signed Source Address Offset (DMA_TCD*n*_SOFF)

Address: 4000_8000h base + 1004h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_SOFF field descriptions

Field	Description
SOFF	Source address signed offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

23.3.24 TCD Transfer Attributes (DMA_TCD*n*_ATTR)

Address: 4000_8000h base + 1006h offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SMOD					SSIZE			DMOD					DSIZE		
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_ATTR field descriptions

Field	Description
15–11 SMOD	Source Address Modulo 0 Source address modulo feature is disabled ≠0 This value defines a specific address range specified to be the value after SADDR + SOFF calculation is performed on the original register value. Setting this field provides the ability to implement a circular data queue easily. For data queues requiring power-of-2 size bytes, the queue should start at a 0-modulo-size address and the SMOD field should be set to the appropriate value for the queue, freezing the desired number of upper address bits. The value programmed into this field specifies the number of lower address bits allowed to change. For a circular queue application, the SOFF is typically set to the transfer size to implement post-increment addressing with the SMOD function constraining the addresses to a 0-modulo-size range.
10–8 SSIZE	Source data transfer size **NOTE:** Using a Reserved value causes a configuration error. The eDMA defaults to privileged data access for all transactions. 000 8-bit 001 16-bit 010 32-bit 011 Reserved 100 16-byte burst 101 32-byte burst 110 Reserved 111 Reserved
7–3 DMOD	Destination Address Modulo See the SMOD definition

Table continues on the next page...

DMA_TCD*n*_ATTR field descriptions (continued)

Field	Description
DSIZE	Destination data transfer size See the SSIZE definition

23.3.25 TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD*n*_NBYTES_MLNO)

This register, or one of the next two registers (TCD_NBYTES_MLOFFNO, TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is disabled (CR[EMLM] = 0)

If minor loop mapping is enabled, see the TCD_NBYTES_MLOFFNO and TCD_NBYTES_MLOFFYES register descriptions for the definition of TCD word 2.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 15d

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_NBYTES_MLNO field descriptions

Field	Description
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed. **NOTE:** An NBYTES value of 0x0000_0000 is interpreted as a 4 GB transfer.

23.3.26 TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD*n*_NBYTES_MLOFFNO)

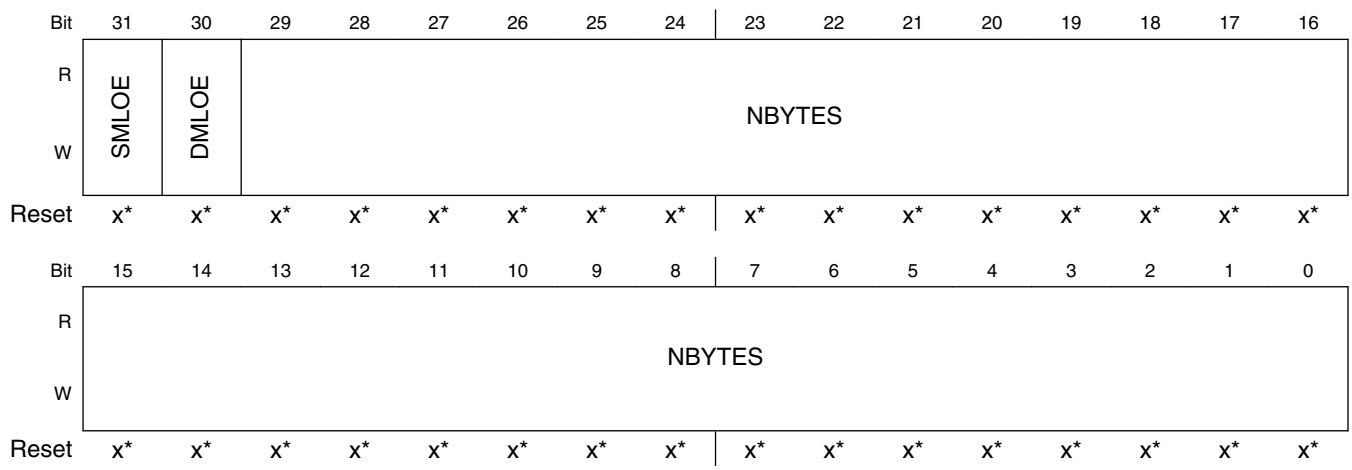
One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- SMLOE = 0 and DMLOE = 0

If minor loop mapping is enabled and SMLOE or DMLOE is set, then refer to the TCD_NBYTES_MLOFFYES register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 15d



* Notes:
- x = Undefined at reset.

DMA_TCD*n*_NBYTES_MLOFFNO field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion.

Table continues on the next page...

DMA_TCD*n*_NBYTES_MLOFFNO field descriptions (continued)

Field	Description
	0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

23.3.27 TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD*n*_NBYTES_MLOFFYES)

One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFNO), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- Minor loop offset is enabled (SMLOE or DMLOE = 1)

If minor loop mapping is enabled and SMLOE and DMLOE are cleared, then refer to the TCD_NBYTES_MLOFFNO register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

Address: 4000_8000h base + 1008h offset + (32d × i), where i=0d to 15d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SMLOE	DMLOE	MLOFF													
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MLOFF						NBYTES									
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_NBYTES_MLOFFYES field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
29–10 MLOFF	If SMLOE or DMLOE is set, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

23.3.28 TCD Last Source Address Adjustment (DMA_TCD*n*_SLAST)

Address: 4000_8000h base + 100Ch offset + (32d × i), where i=0d to 15d

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_SLAST field descriptions

Field	Description
SLAST	Last Source Address Adjustment Adjustment value added to the source address at the completion of the major iteration count. This value can be applied to restore the source address to the initial value, or adjust the address to reference the next data structure. This register uses two's complement notation; the overflow bit is discarded.

23.3.29 TCD Destination Address (DMA_TCD*n*_DADDR)

Address: 4000_8000h base + 1010h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DADDR field descriptions

Field	Description
DADDR	Destination Address Memory address pointing to the destination data.

23.3.30 TCD Signed Destination Address Offset (DMA_TCD*n*_DOFF)

Address: 4000_8000h base + 1014h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DOFF field descriptions

Field	Description
DOFF	Destination Address Signed Offset Sign-extended offset applied to the current destination address to form the next-state value as each destination write is completed.

23.3.31 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD*n*_CITER_ELINKYES)

If TCDn_CITER[ELINK] is set, the TCDn_CITER register is defined as follows.

Address: 4000_8000h base + 1016h offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	0			LINKCH			CITER
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CITER_ELINKYES field descriptions

Field	Description
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–13 Reserved	This field is reserved.
12–9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request to the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field.

Table continues on the next page...

DMA_TCD*n*_CITER_ELINKYES field descriptions (continued)

Field	Description
	NOTE: When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

23.3.32 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD*n*_CITER_ELINKNO)

If TCDn_CITER[ELINK] is cleared, the TCDn_CITER register is defined as follows.

Address: 4000_8000h base + 1016h offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	CITER						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CITER_ELINKNO field descriptions

Field	Description
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for

Table continues on the next page...

DMA_TCD*n*_CITER_ELINKNO field descriptions (continued)

Field	Description
	example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

23.3.33 TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD*n*_DLASTSGA)

Address: 4000_8000h base + 1018h offset + (32d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DLASTSGA field descriptions

Field	Description
DLASTSGA	Destination last address adjustment or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo-32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo-32-byte, otherwise a configuration error is reported.

23.3.34 TCD Control and Status (DMA_TCD*n*_CSR)

Address: 4000_8000h base + 101Ch offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read	BWC				MAJORLINKCH			
Write			0					
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read	DONE	ACTIVE	MAJORELINK	ESG	DREQ	INTHALF	INTMAJOR	START
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CSR field descriptions

Field	Description
15–14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces the eDMA to stall after the completion of each read/write access to control the bus request bandwidth seen by the crossbar switch. **NOTE:** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00 No eDMA engine stalls. 01 Reserved 10 eDMA engine stalls for 4 cycles after each R/W. 11 eDMA engine stalls for 8 cycles after each R/W.
13–12 Reserved	This field is reserved.
11–8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
7 DONE	Channel Done This flag indicates the eDMA has completed the major loop. The eDMA engine sets it as the CITER count reaches zero. The software clears it, or the hardware when the channel is activated. The access of this field is W0C, write-zero-to-clear.

Table continues on the next page...

DMA_TCD*n*_CSR field descriptions (continued)

Field	Description
	NOTE: This bit must be cleared to write the MAJORELINK or ESG bits.
6 ACTIVE	Channel Active This flag signals the channel is currently in execution. It is set when channel service begins, and is cleared by the eDMA as the minor loop completes or when any error condition is detected.
5 MAJORELINK	Enable channel-to-channel linking on major loop complete As the channel completes the major loop, this flag enables the linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. **NOTE:** To support the dynamic linking coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0 The channel-to-channel linking is disabled. 1 The channel-to-channel linking is enabled.
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo-32 address containing a 32-byte data structure loaded as the transfer control descriptor into the local memory. **NOTE:** To support the dynamic scatter/gather coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0 The current channel's TCD is normal format. 1 The current channel's TCD specifies a scatter gather format. The DLASTSGA field provides a memory pointer to the next TCD to be loaded into this channel after the major loop completes its execution.
3 DREQ	Disable Request If this flag is set, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches zero. 0 The channel's ERQ bit is not affected. 1 The channel's ERQ bit is cleared when the major loop is complete.
2 INTHALF	Enable an interrupt when major counter is half complete. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT register when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER == (BITER >> 1)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE:** If BITER = 1, do not use INTHALF. Use INTMAJOR instead. 0 The half-point interrupt is disabled. 1 The half-point interrupt is enabled.
1 INTMAJOR	Enable an interrupt when major iteration count completes. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT when the current major iteration count reaches zero. 0 The end-of-major loop interrupt is disabled. 1 The end-of-major loop interrupt is enabled.

Table continues on the next page...

DMA_TCD*n*_CSR field descriptions (continued)

Field	Description
0 START	Channel Start If this flag is set, the channel is requesting service. The eDMA hardware automatically clears this flag after the channel begins execution. 0 The channel is not explicitly started. 1 The channel is explicitly started via a software initiated service request.

23.3.35 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD*n*_BITER_ELINKYES)

If the TCDn_BITER[ELINK] bit is set, the TCDn_BITER register is defined as follows.

Address: 4000_8000h base + 101Eh offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	0			LINKCH			BITER
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_BITER_ELINKYES field descriptions

Field	Description
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–13 Reserved	This field is reserved.
12–9 LINKCH	Link Channel Number

Table continues on the next page...

DMA_TCD*n*_BITER_ELINKYES field descriptions (continued)

Field	Description
	If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.
BITER	Starting major iteration count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

23.3.36 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD*n*_BITER_ELINKNO)

If the TCDn_BITER[ELINK] bit is cleared, the TCDn_BITER register is defined as follows.

Address: 4000_8000h base + 101Eh offset + (32d × i), where i=0d to 15d

Bit	15	14	13	12	11	10	9	8
Read Write	ELINK	BITER						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

Bit	7	6	5	4	3	2	1	0
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_BITER_ELINKNO field descriptions

Field	Description
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.

Table continues on the next page...

DMA_TCD*n*_BITER_ELINKNO field descriptions (continued)

Field	Description
	NOTE: When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field is reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

23.4 Functional description

The operation of the eDMA is described in the following subsections.

23.4.1 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

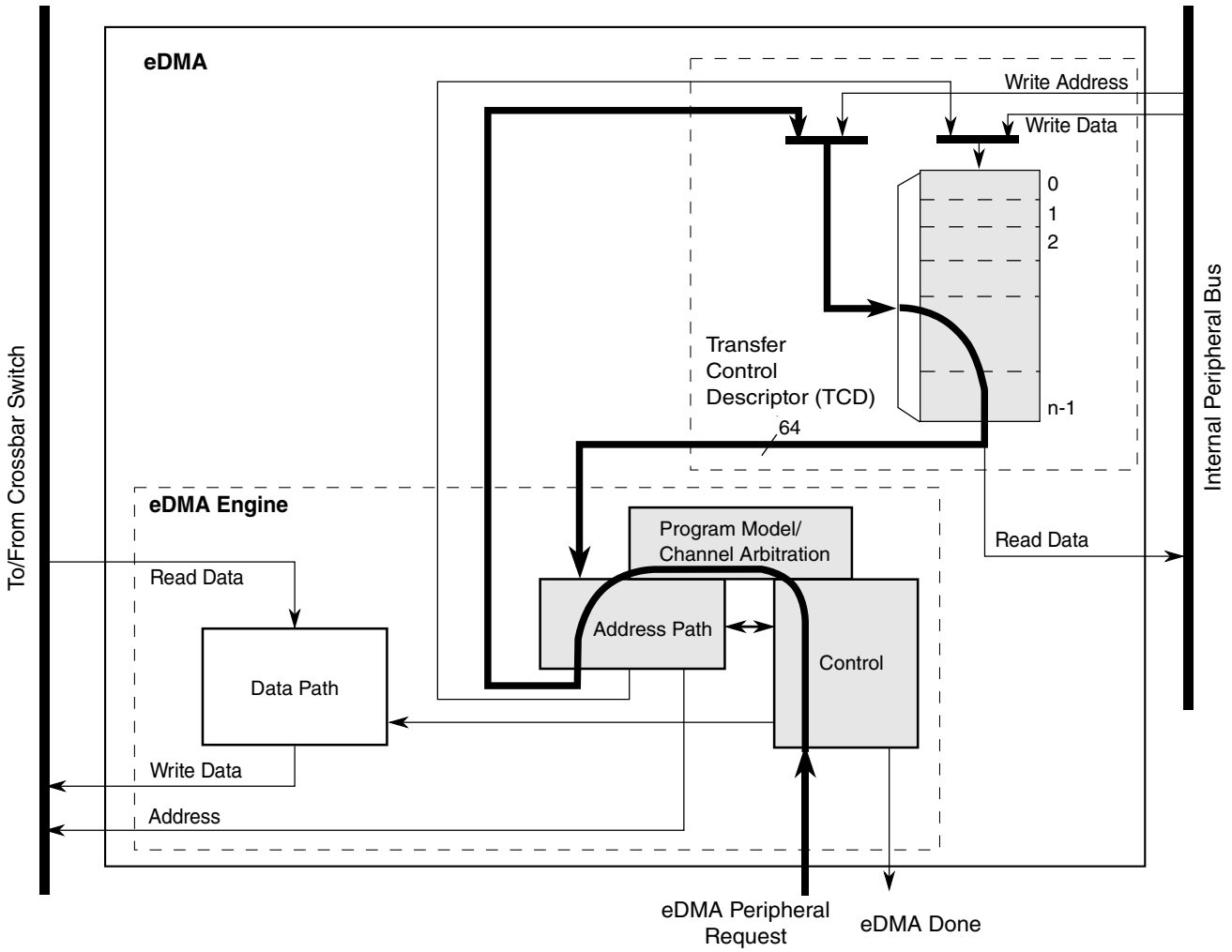


Figure 23-2. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] bit follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration. In the next cycle, the channel arbitration performs, using the fixed-priority or round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor read from the local memory and loaded into the eDMA engine address path channel x or y registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path channel x or y registers.

The following diagram illustrates the second part of the basic data flow:

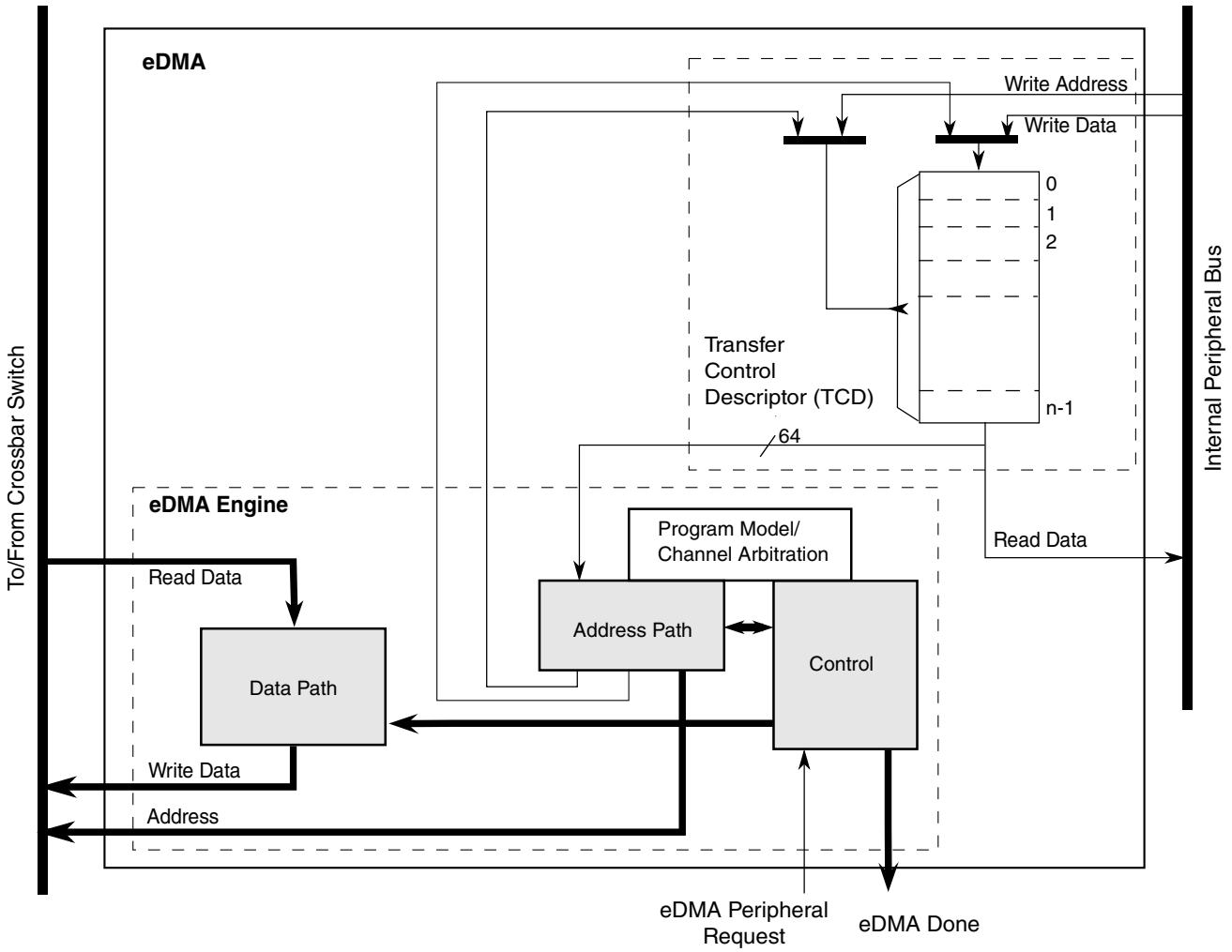


Figure 23-3. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) sequence through the required source reads and destination writes to perform the actual data movement. The source reads are initiated and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the minor byte count has transferred.

After the minor byte count has moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD, for example, SADDR, DADDR, CITER. If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

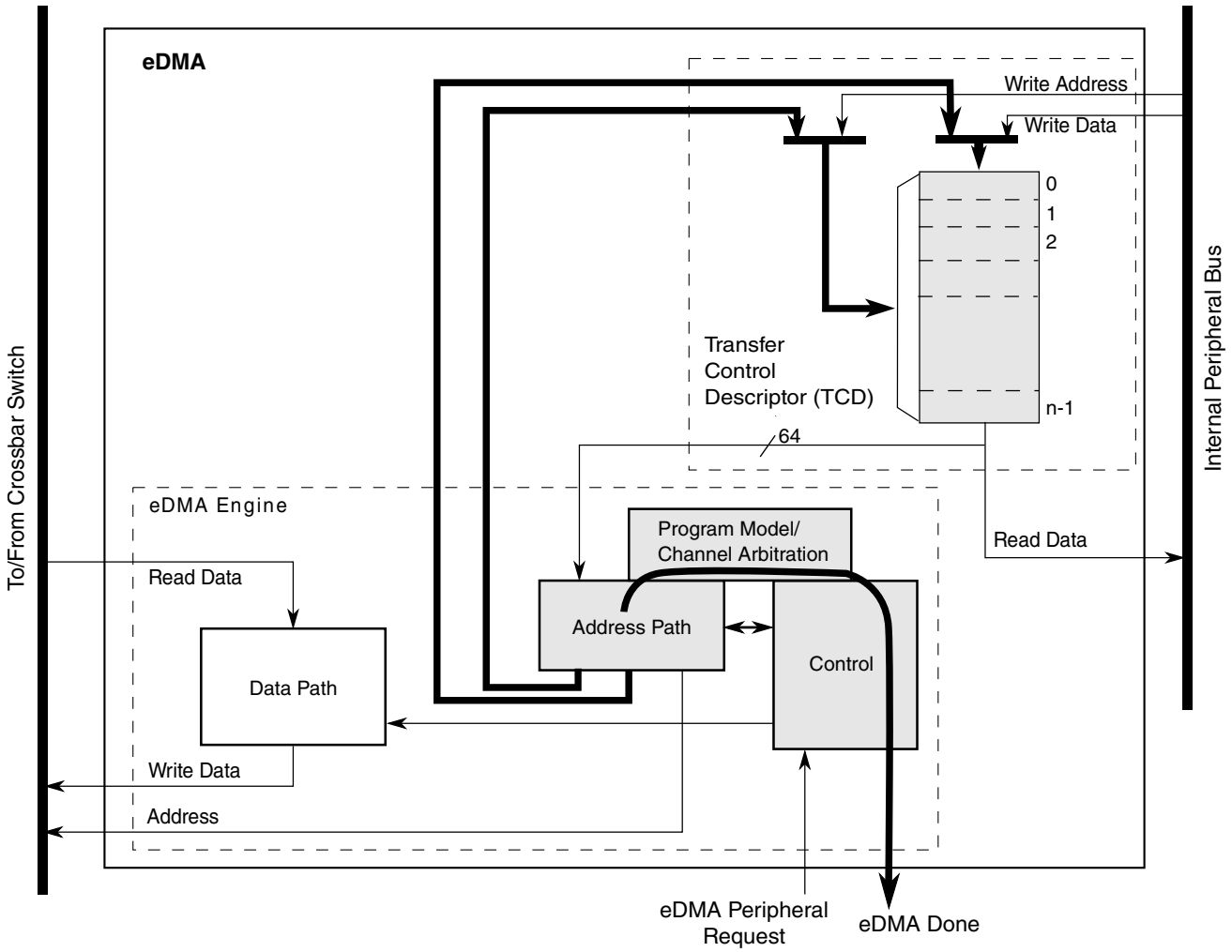


Figure 23-4. eDMA operation, part 3

23.4.2 Fault reporting and handling

Channel errors are reported in the Error Status register (DMAx_ES) and can be caused by:

- A configuration error, which is an illegal setting in the transfer-control descriptor or an illegal priority register setting in Fixed-Arbitration mode, or
- An error termination to a bus master read or write cycle

A configuration error is reported when the starting source or destination address, source or destination offsets, minor loop byte count, or the transfer size represent an inconsistent state. Each of these possible causes are detailed below:

- The addresses and offsets must be aligned on 0-modulo-transfer-size boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.

- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size respectively.
- In fixed arbitration mode, a configuration error is caused by any two channel priorities being equal. All channel priority levels must be unique when fixed arbitration mode is enabled.

NOTE

When two channels have the same priority, a channel priority error exists and will be reported in the Error Status register. However, the channel number will not be reported in the Error Status register. When all of the channel priorities within a group are not unique, the channel number selected by arbitration is undetermined.

To aid in Channel Priority Error (CPE) debug, set the Halt On Error bit in the DMA's Control Register. If all of the channel priorities within a group are not unique, the DMA will be halted after the CPE error is recorded. The DMA will remain halted and will not process any channel service requests. Once all of the channel priorities are set to unique numbers, the DMA may be enabled again by clearing the Halt bit.

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[E_LINK] bit does not equal the TCDn_BITER[E_LINK] bit.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, report as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion when properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error

occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

A transfer may be cancelled by software with the CR[CX] bit. When a cancel transfer request is recognized, the DMA engine stops processing the channel. The current read-write sequence is allowed to finish. If the cancel occurs on the last read-write sequence of a major or minor loop, the cancel request is discarded and the channel retires normally.

The error cancel transfer is the same as a cancel transfer except the Error Status register (DMAx_ES) is updated with the cancelled channel number and ECX is set. The TCD of a cancelled channel contains the source and destination addresses of the last transfer saved in the TCD. If the channel needs to be restarted, you must re-initialize the TCD because the aforementioned fields no longer represent the original parameters. When a transfer is cancelled by the error cancel transfer mechanism, the channel number is loaded into DMA_ES[ERRCHN] and ECX and VLD are set. In addition, an error interrupt may be generated if enabled.

NOTE

The cancel transfer request allows the user to stop a large data transfer in the event the full data transfer is no longer needed. The cancel transfer bit does not abort the channel. It simply stops the transferring of data and then retires the channel through its normal shutdown sequence. The application software must handle the context of the cancel. If an interrupt is desired (or not), then the interrupt should be enabled (or disabled) before the cancel request. The application software must clean up the transfer control descriptor since the full transfer did not occur.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel bit in the eDMA error register is asserted. At the same time, the details of the error condition are loaded into the Error Status register (DMAx_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request, are not affected when an error is detected. After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

23.4.3 Channel preemption

Channel preemption is enabled on a per-channel basis by setting the DCHPRIn[ECP] bit. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher priority channel. After the preempting channel has completed all its minor loop data transfers, the preempted channel is restored and resumes execution. After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended and the higher priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted. Preemption is available only when fixed arbitration is selected.

A channel's ability to preempt another channel can be disabled by setting DCHPRIn[DPA]. When a channel's preempt ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This allows for a pool of low priority, large data-moving channels to be defined. These low priority channels can be configured to not preempt each other, thus preventing a low priority channel from consuming the preempt slot normally available to a true, high priority channel.

23.4.4 Performance

This section addresses the performance of the eDMA module, focusing on two separate metrics:

- In the traditional data movement context, performance is best expressed as the peak data transfer rates achieved using the eDMA. In most implementations, this transfer rate is limited by the speed of the source and destination address spaces.
- In a second context where device-paced movement of single data values to/from peripherals is dominant, a measure of the requests that can be serviced in a fixed time is a more relevant metric. In this environment, the speed of the source and destination address spaces remains important. However, the microarchitecture of the eDMA also factors significantly into the resulting metric.

23.4.4.1 Peak transfer rates

The peak transfer rates for several different source and destination transfers are shown in the following tables. These tables assume:

- Internal SRAM can be accessed with zero wait-states when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states, when viewed from the system bus data phase
- All internal peripheral bus accesses are 32-bits in size

NOTE

All architectures will not meet the assumptions listed above. See the SRAM configuration section for more information.

This table compares peak transfer rates based on different possible system speeds. Specific chips/devices may not support all system speeds listed.

Table 23-4. eDMA peak transfer rates (Mbytes/sec)

System Speed, Width	Internal SRAM-to-Internal SRAM	32 bit internal peripheral bus-to-Internal SRAM	Internal SRAM-to-32 bit internal peripheral bus
48 MHz, 32 bit	96.0	48.0	38.4
66.7 MHz, 32 bit	133.3	66.7	53.3
83.3 MHz, 32 bit	166.7	83.3	66.7
100.0 MHz, 32 bit	200.0	100.0	80.0
133.3 MHz, 32 bit	266.7	133.3	106.7
150.0 MHz, 32 bit	300.0	150.0	120.0

Internal-SRAM-to-internal-SRAM transfers occur at the core's datapath width. For all transfers involving the internal peripheral bus, 32-bit transfer sizes are used. In all cases, the transfer rate includes the time to read the source plus the time to write the destination.

23.4.4.2 Peak request rates

The second performance metric is a measure of the number of DMA requests that can be serviced in a given amount of time. For this metric, assume that the peripheral request causes the channel to move a single internal peripheral bus-mapped operand to/from internal SRAM. The same timing assumptions used in the previous example apply to this calculation. In particular, this metric also reflects the time required to activate the channel.

The eDMA design supports the following hardware service request sequence. Note that the exact timing from Cycle 7 is a function of the response times for the channel's read and write accesses. In the case of an internal peripheral bus read and internal SRAM write, the combined data phase time is 4 cycles. For an SRAM read and internal peripheral bus write, it is 5 cycles.

Table 23-5. Hardware service request process

Cycle		Description
With internal peripheral bus read and internal SRAM write	With SRAM read and internal peripheral bus write	
1		eDMA peripheral request is asserted.
2		The eDMA peripheral request is registered locally in the eDMA module and qualified. TCD*n*_CSR[START] bit initiated requests start at this point with the registering of the user write to TCD*n* word 7.
3		Channel arbitration begins.
4		Channel arbitration completes. The transfer control descriptor local memory read is initiated.
5–6		The first two parts of the activated channel's TCD is read from the local memory. The memory width to the eDMA engine is 64 bits, so the entire descriptor can be accessed in four cycles
7		The first system bus read cycle is initiated, as the third part of the channel's TCD is read from the local memory. Depending on the state of the crossbar switch, arbitration at the system bus may insert an additional cycle of delay here.
8–11	8–12	The last part of the TCD is read in. This cycle represents the first data phase for the read, and the address phase for the destination write.
12	13	This cycle represents the data phase of the last destination write.
13	14	The eDMA engine completes the execution of the inner minor loop and prepares to write back the required TCD*n* fields into the local memory. The TCD*n* word 7 is read and checked for channel linking or scatter/gather requests.
14	15	The appropriate fields in the first part of the TCD*n* are written back into the local memory.
15	16	The fields in the second part of the TCD*n* are written back into the local memory. This cycle coincides with the next channel arbitration cycle start.
16	17	The next channel to be activated performs the read of the first part of its TCD from the local memory. This is equivalent to Cycle 4 for the first channel's service request.

Assuming zero wait states on the system bus, DMA requests can be processed every 9 cycles. Assuming an average of the access times associated with internal peripheral bus-to-SRAM (4 cycles) and SRAM-to-internal peripheral bus (5 cycles), DMA requests can

be processed every 11.5 cycles (4 + (4+5)/2 + 3). This is the time from Cycle 4 to Cycle x +5. The resulting peak request rate, as a function of the system frequency, is shown in the following table.

Table 23-6. eDMA peak request rate (MReq/sec)

System frequency (MHz)	Request rate with zero wait states	Request rate with wait states
48.0	5.3	4.2
66.6	7.4	5.8
83.3	9.2	7.2
100.0	11.1	8.7
133.3	14.8	11.6
150.0	16.6	13.0

A general formula to compute the peak request rate with overlapping requests is:

PEAKreq = freq / [entry + (1 + read_ws) + (1 + write_ws) + exit]

where:

Table 23-7. Peak request formula operands

Operand	Description
PEAKreq	Peak request rate
freq	System frequency
entry	Channel startup (4 cycles)
read_ws	Wait states seen during the system bus read data phase
write_ws	Wait states seen during the system bus write data phase
exit	Channel shutdown (3 cycles)

23.4.4.3 eDMA performance example

Consider a system with the following characteristics:

- Internal SRAM can be accessed with one wait-state when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states viewed from the system bus data phase
- System operates at 150 MHz

For an SRAM to internal peripheral bus transfer,

PEAKreq = 150 MHz / [4 + (1 + 1) + (1 + 3) + 3] cycles = 11.5 Mreq/sec

For an internal peripheral bus to SRAM transfer,

PEAKreq = 150 MHz / [4 + (1 + 2) + (1 + 1) + 3] cycles = 12.5 Mreq/sec

Assuming an even distribution of the two transfer types, the average peak request rate would be:

PEAKreq = (11.5 Mreq/sec + 12.5 Mreq/sec) / 2 = 12.0 Mreq/sec

The minimum number of cycles to perform a single read/write, zero wait states on the system bus, from a cold start where no channel is executing and eDMA is idle are:

- 11 cycles for a software, that is, a TCD*n*_CSR[START] bit, request
- 12 cycles for a hardware, that is, an eDMA peripheral request signal, request

Two cycles account for the arbitration pipeline and one extra cycle on the hardware request resulting from the internal registering of the eDMA peripheral request signals. For the peak request rate calculations above, the arbitration and request registering is absorbed in or overlaps the previous executing channel.

Note

When channel linking or scatter/gather is enabled, a two cycle delay is imposed on the next channel selection and startup. This allows the link channel or the scatter/gather channel to be eligible and considered in the arbitration pool for next channel selection.

23.5 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

23.5.1 eDMA initialization

To initialize the eDMA:

1. Write to the CR if a configuration other than the default is desired.
2. Write the channel priority levels to the DCHPRI*n* registers if a configuration other than the default is desired.

3. Enable error interrupts in the EEI register if so desired.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the ERQ register.
6. Request channel service via either:
   - Software: setting the TCD*n*_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in the following table, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, as defined by TCD*n*_SADDR, to the destination, as defined by TCD*n*_DADDR, continue until the number of bytes specified by TCD*n*_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, further post processing executes, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 23-8. TCD Control and Status fields

TCD*n*_CSR field name	Description
START	Control bit to start channel explicitly when using a software initiated DMA service (Automatically cleared by hardware)
ACTIVE	Status bit indicating the channel is currently in execution
DONE	Status bit indicating major loop completion (cleared by software when using a software initiated DMA service)
D_REQ	Control bit to disable DMA request at end of major loop completion when using a hardware initiated DMA service
BWC	Control bits for throttling bandwidth control of a channel
E_SG	Control bit to enable scatter-gather feature
INT_HALF	Control bit to enable interrupt when major loop is half complete
INT_MAJ	Control bit to enable interrupt when major loop completes

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

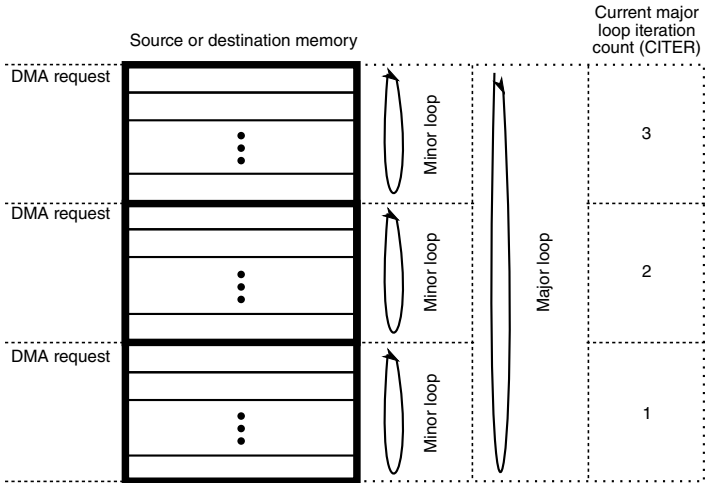


Figure 23-5. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

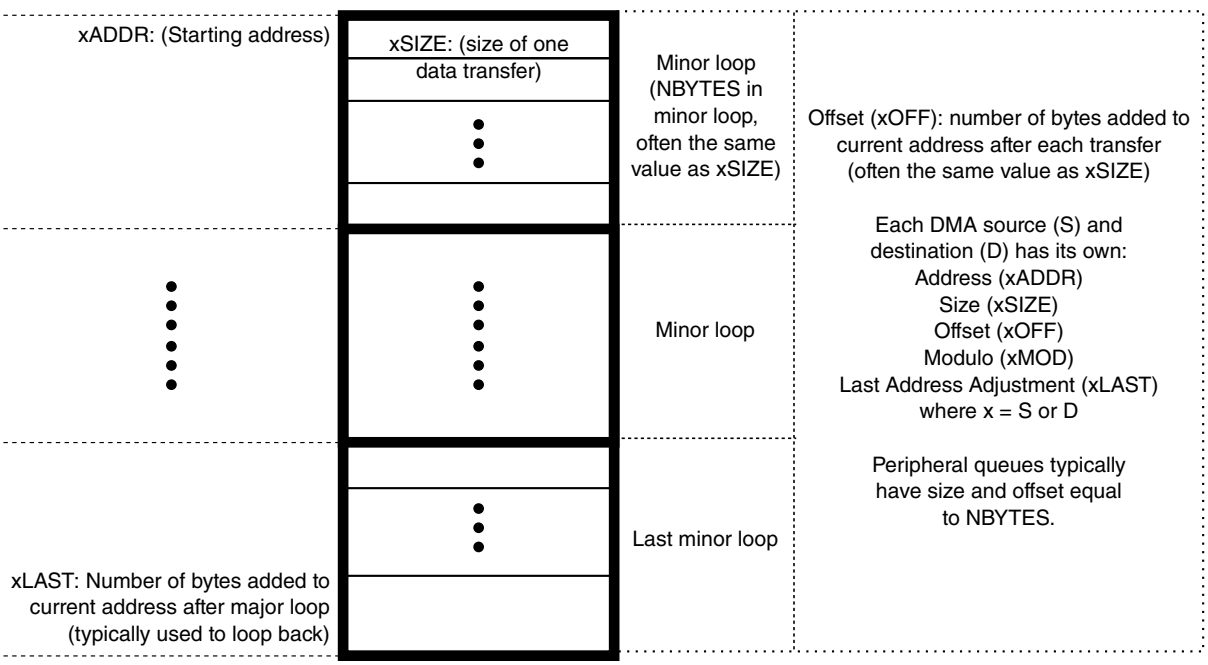


Figure 23-6. Memory array terms

23.5.2 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data. Most programming errors are reported on a per channel basis with the exception of channel priority error (ES[CPE]).

For all error types other than channel priority error, the channel number causing the error is recorded in the Error Status register (DMAx_ES). If the error source is not removed before the next activation of the problem channel, the error is detected and recorded again.

If priority levels are not unique, when any channel requests service, a channel priority error is reported. The highest channel priority with an active request is selected, but the lowest numbered channel with that priority is selected by arbitration and executed by the eDMA engine. The hardware service request handshake signals, error interrupts, and error reporting is associated with the selected channel.

23.5.3 Arbitration mode considerations

This section discusses arbitration considerations for the eDMA.

23.5.3.1 Fixed channel arbitration

In this mode, the channel service request from the highest priority channel is selected to execute.

23.5.3.2 Round-robin channel arbitration

Channels are serviced starting with the highest channel number and rotating through to the lowest channel number without regard to the channel priority levels.

23.5.4 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

23.5.4.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the TCD*n*_CSR[DONE] bit is set and an interrupt generates if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INT_MAJ] = 1
TCDn_CSR[START] = 1 (Should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] bit requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
    b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
    c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
    d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
    e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
    f. Write 32-bits to location 0x2008 → third iteration of the minor loop.
    g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.

h. Write 32-bits to location 0x200C → last iteration of the minor loop → major loop complete.

6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[*n*] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

23.5.4.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop transferring 16 bytes per iteration. After the channel's hardware requests are enabled in the ERQ register, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware, that is, eDMA peripheral, request for channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source to destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.

f. Write 32-bits to location 0x2008 → third iteration of the minor loop.

g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.

h. Write 32-bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0.
8. The channel retires → one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware, that is, eDMA peripheral, requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: channel TCD data from local memory to internal register file.
13. The source to destination transfers are executed as follows:

    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.

    b. Write 32-bits to location 0x2010 → first iteration of the minor loop.

    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.

    d. Write 32-bits to location 0x2014 → second iteration of the minor loop.

    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.

    f. Write 32-bits to location 0x2018 → third iteration of the minor loop.

    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.

    h. Write 32-bits to location 0x201C → last iteration of the minor loop → major loop complete.

14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).

15. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[n] = 1.
16. The channel retires → major loop complete. The eDMA goes idle or services the next channel.

23.5.4.3 Using the modulo feature

The modulo feature of the eDMA provides the ability to implement a circular data queue in which the size of the queue is a power of 2. MOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value while the 28 upper address bits (0x1234567*x*) retain their original value. In this example the source address is set to 0x12345670, the offset is set to 4 bytes and the MOD field is set to 4, allowing for a 2^4 byte (16-byte) size queue.

Table 23-9. Modulo example

Transfer Number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

23.5.5 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

23.5.5.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software initiated service requests. The first is to read the TCD*n*_CITER field and test for a change. Another method may be extracted from the sequence shown below. The second method is

to test the TCD*n*_CSR[START] bit and the TCD*n*_CSR[ACTIVE] bit. The minor-loop-complete condition is indicated by both bits reading zero after the TCD*n*_CSR[START] was set. Polling the TCD*n*_CSR[ACTIVE] bit may be inconclusive, because the active status may be missed if the channel execution is short in duration.

The TCD status bits execute the following sequence for a software activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	1	0	0	Channel service request via software
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

The best method to test for minor-loop completion when using hardware, that is, peripheral, initiated service requests is to read the TCD*n*_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status bits execute the following sequence for a hardware-activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	0	0	0	Channel service request via hardware (peripheral request asserted)
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

For both activation types, the major-loop-complete status is explicitly indicated via the TCD*n*_CSR[DONE] bit.

The TCD*n*_CSR[START] bit is cleared automatically when the channel begins execution regardless of how the channel activates.

23.5.5.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_NBYTES values if read while a channel executes. The true values of the SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file and not the values in the TCD local memory for that channel. The addresses, SADDR and

DADDR, and NBYTES, which decrement to zero as the transfer progresses, can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

23.5.5.3 Checking channel preemption status

Preemption is available only when fixed arbitration is selected as the channel arbitration mode. A preemptive situation is one in which a preempt-enabled channel runs and a higher priority request becomes active. When the eDMA engine is not operating in fixed channel arbitration mode, the determination of the actively running relative priority outstanding requests become undefined. Channel priorities are treated as equal, that is, constantly rotating, when Round-Robin Arbitration mode is selected.

The TCD*n*_CSR[ACTIVE] bit for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended while the preempting channel executes one major loop iteration. If two TCD*n*_CSR[ACTIVE] bits are set simultaneously in the global TCD map, a higher priority channel is actively preempting a lower priority channel.

23.5.6 Channel Linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD*n*_CSR[START] bit of another channel (or itself), therefore initiating a service request for that channel. When properly enabled, the EDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCD*n*_CITER[E_LINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, the initial fields of:

```
TCDn_CITER[E_LINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJOR_E_LINK] = 1
TCDn_CSR[MAJOR_LINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] bit

2. Minor loop done → set TCD12_CSR[START] bit
3. Minor loop done → set TCD12_CSR[START] bit
4. Minor loop done, major loop done→ set TCD7_CSR[START] bit

When minor loop linking is enabled (TCD*n*_CITER[E_LINK] = 1), the TCD*n*_CITER[CITER] field uses a nine bit vector to form the current iteration count. When minor loop linking is disabled (TCD*n*_CITER[E_LINK] = 0), the TCD*n*_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCD*n*_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

Note

The TCD*n*_CITER[E_LINK] bit and the TCD*n*_BITER[E_LINK] bit must equal or a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop, half-way done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, i.e, use another channel's TCD, at the end of a loop.

Table 23-10. Channel Linking Parameters

Desired Link Behavior	TCD Control Field Name	Description
Link at end of Minor Loop	CITER[E_LINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of Major Loop	CSR[MAJOR_E_LINK]	Enable channel-to-channel linking on major loop completion
	CSR[MAJOR_LINKCH]	Link channel number when linking at end of major loop

23.5.7 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

23.5.7.1 Dynamically changing the channel priority

The following two options are recommended for dynamically changing channel priority levels:

1. Switch to Round-Robin Channel Arbitration mode, change the channel priorities, then switch back to Fixed Arbitration mode,
2. Disable all the channels, change the channel priorities, then enable the appropriate channels.

23.5.7.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCD.major.e_link bit during channel execution (see the diagram in TCD structure). This bit is read from the TCD local memory at the end of channel execution, thus allowing the user to enable the feature during channel execution.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic channel link by enabling the TCD.major.e_link bit at the same time the eDMA engine is retiring the channel. The TCD.major.e_link would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

The following coherency model is recommended when executing a dynamic channel link request.

1. Write 1 to the TCD.major.e_link bit.
2. Read back the TCD.major.e_link bit.
3. Test the TCD.major.e_link request status:
   - If TCD.major.e_link = 1, the dynamic link attempt was successful.
   - If TCD.major.e_link = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCD.major.e_link bit to zero on any writes to a channel's TCD.word7 after that channel's TCD.done bit is set, indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link bit. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

23.5.7.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources.When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in eDMA programmer's model, thus replacing the current descriptor.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic scatter/gather operation by enabling the TCD.e_sg bit at the same time the eDMA engine is retiring the channel. The TCD.e_sg would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods for this coherency model are shown in the following subsections. Method 1 has the advantage of reading the major.linkch field and the e_sg bit with a single read. For both dynamic channel linking and scatter/gather requests, the TCD local memory controller forces the TCD.major.e_link and TCD.e_sg bits to zero on any writes to a channel's TCD.word7 if that channel's TCD.done bit is set indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link or TCD.e_sg bits. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

23.5.7.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCD.major.e_link bit is zero, the TCD.major.linkch field is not used by the eDMA. In this case, the TCD.major.linkch bits may be used for other purposes. This method uses the TCD.major.linkch field as a TCD indentification (ID).

1. When the descriptors are built, write a unique TCD ID in the TCD.major.linkch field for each TCD associated with a channel using dynamic scatter/gather.
2. Write 1b to the TCD.d_req bit.

Should a dynamic scatter/gather attempt fail, setting the TCD.d_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offest value.

3. Write the TCD.dlast_sga field with the scatter/gather address.
4. Write 1b to the TCD.e_sg bit.
5. Read back the 16 bit TCD control/status field.
6. Test the TCD.e_sg request status and TCD.major.linkch value:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b and the major.linkch (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If e_sg = 0b and the major.linkch (ID) changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

23.5.7.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCD.dlast_sga field as a TCD indentification (ID).

1. Write 1b to the TCD.d_req bit.

   Should a dynamic scatter/gather attempt fail, setting the d_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offest value.

2. Write theTCD.dlast_sga field with the scatter/gather address.
3. Write 1b to the TCD.e_sg bit.
4. Read back the TCD.e_sg bit.
5. Test the TCD.e_sg request status:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b, read the 32 bit TCD dlast_sga field.

   If e_sg = 0b and the dlast_sga did not change, the attempted dynamic link did not succeed (the channel was already retiring).

If e_sg = 0b and the dlast_sga changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

23.5.8 Suspend/resume a DMA channel with active hardware service requests

The DMA allows the user to move data from memory or peripheral registers to another location in memory or peripheral registers without CPU interaction. Once the DMA and peripherals have been configured and are active, it is rare to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, a specific procedure must be followed. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (SPI), ADC, or other module.

23.5.8.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check Hardware Request Status Register (DMA_HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ bit on the appropriate DMA channel.

23.5.8.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ bit.
2. Enable the DMA service request at the peripheral.

For example, assume a SPI module is set as a master for transmitting data via a DMA service request when the SPI TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If the user needs to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to SPI_RSER[TFFF_RE]. Confirm that SPI_RSER[TFFF_RE] is 0.

2. Ensure there is no DMA service request from the SPI by verifying that DMA_HRS[HRS*n*] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ bit. If a service request is present, wait until the request has been processed and the HRS bit reads zero.

Chapter 24
External Watchdog Monitor (EWM)

24.1 Chip-specific EWM information

24.1.1 EWM clocks

This table shows the EWM clocks and the corresponding chip clocks.

Table 24-1. EWM clock connections

Module clock	Chip clock
Low Power Clock	1 kHz LPO Clock

24.1.2 EWM low-power modes

This table shows the EWM low-power modes and the corresponding chip low-power modes.

Table 24-2. EWM low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS

24.1.3 $\overline{\text{EWM\_OUT}}$ pin state in low power modes

When the CPU enters a Run mode from Wait or Stop recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode from Power Down, the pin returns to its reset state.

24.2 Introduction

For safety, a redundant watchdog system, External Watchdog Monitor (EWM), is designed to monitor external circuits, as well as the MCU software flow. This provides a back-up mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The watchdog is generally used to monitor the flow and execution of embedded software within an MCU. The watchdog consists of a counter that if allowed to overflow, forces an internal reset (asynchronous) to all on-chip peripherals and optionally assert the $\overline{\text{RESET}}$ pin to reset external devices/circuits. The overflow of the watchdog counter must not occur if the software code works well and services the watchdog to re-start the actual counter.

The EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM provides an independent $\overline{\text{EWM\_out}}$ signal that when asserted resets or places an external circuit into a safe mode. The $\overline{\text{EWM\_out}}$ signal is asserted upon the EWM counter time-out. An optional external input EWM_in is provided to allow additional control of the assertion of $\overline{\text{EWM\_out}}$ signal.

24.2.1 Features

Features of EWM module include:

- Independent LPO_CLK clock source
- Programmable time-out period specified in terms of number of EWM LPO_CLK clock cycles.
- Windowed refresh option
  - Provides robust check that program flow is faster than expected.
  - Programmable window.
  - Refresh outside window leads to assertion of $\overline{\text{EWM\_out}}$.
- Robust refresh mechanism
  - Write values of 0xB4 and 0x2C to EWM Refresh Register within 15 (*EWM_refresh_time*) peripheral bus clock cycles.

- One output port, $\overline{\text{EWM\_out}}$, when asserted is used to reset or place the external circuit into safe mode.
- One Input port, EWM_in, allows an external circuit to control the assertion of the $\overline{\text{EWM\_out}}$ signal.

24.2.2 Modes of Operation

This section describes the module's operating modes.

24.2.2.1 Stop Mode

When the EWM is in stop mode, the CPU refreshes to the EWM cannot occur. On entry to stop mode, the EWM's counter freezes.

There are two possible ways to exit from Stop mode:

- On exit from stop mode through a reset, the EWM remains disabled.
- On exit from stop mode by an interrupt, the EWM is re-enabled, and the counter continues to be clocked from the same value prior to entry to stop mode.

Note the following if the EWM enters the stop mode during CPU refresh mechanism: At the exit from stop mode by an interrupt, refresh mechanism state machine starts from the previous state which means, if first refresh command is written correctly and EWM enters the stop mode immediately, the next command has to be written within the next 15 (*EWM_refresh_time*) peripheral bus clocks after exiting from stop mode. User must mask all interrupts prior to executing EWM refresh instructions.

24.2.2.2 Debug Mode

Entry to debug mode has no effect on the EWM.

- If the EWM is enabled prior to entry of debug mode, it remains enabled.
- If the EWM is disabled prior to entry of debug mode, it remains disabled.

24.2.3 Block Diagram

This figure shows the EWM block diagram.

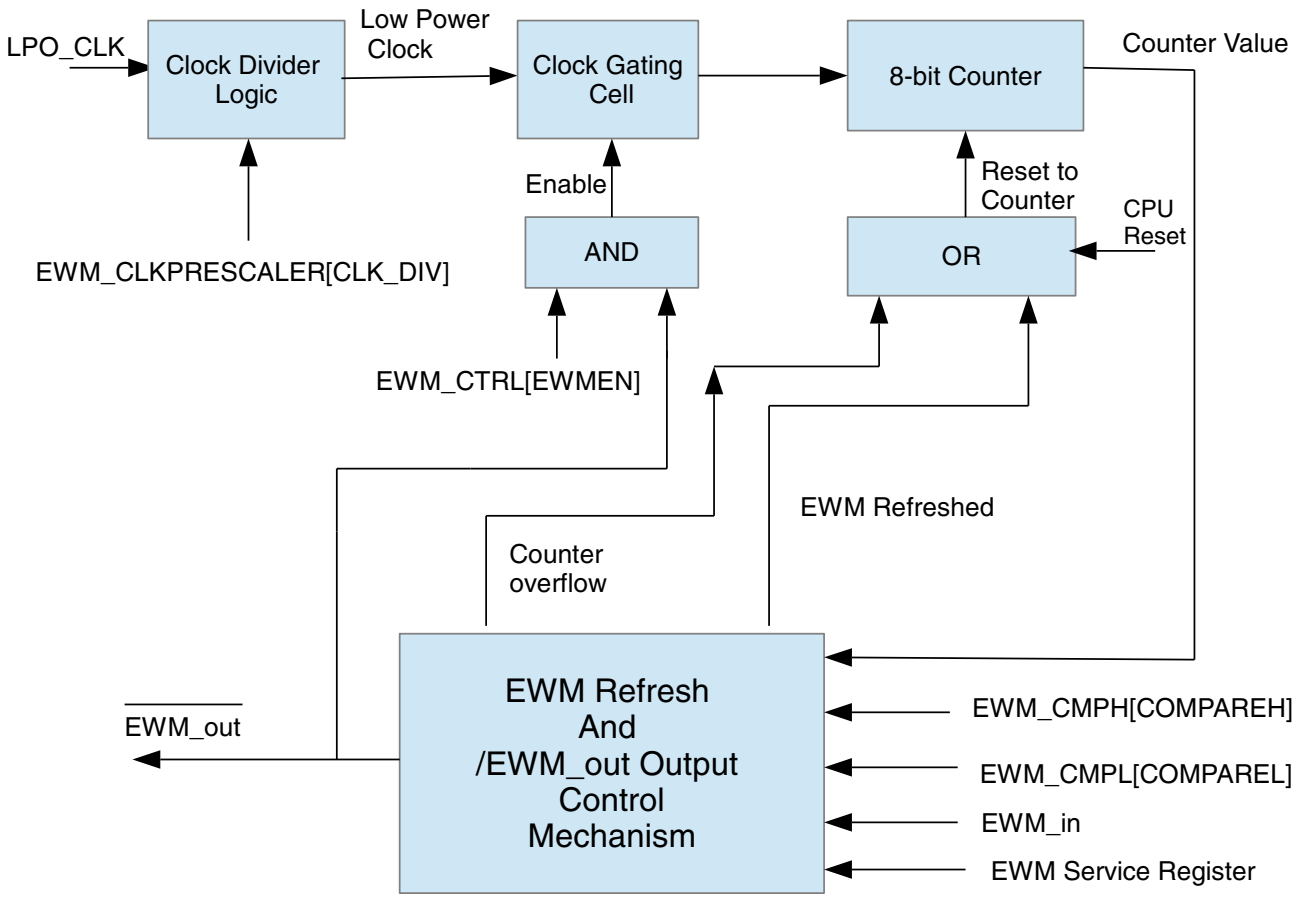


Figure 24-1. EWM Block Diagram

24.3 EWM Signal Descriptions

The EWM has two external signals, as shown in the following table.

Table 24-3. EWM Signal Descriptions

Signal	Description	I/O
EWM_in	EWM input for safety status of external safety circuits. The polarity of EWM_in is programmable using the EWM_CTRL[ASSIN] bit. The default polarity is active-low.	I
$\overline{\text{EWM\_out}}$	EWM reset out signal	O

24.4 Memory Map/Register Definition

This section contains the module memory map and registers.

EWM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_1000	Control Register (EWM_CTRL)	8	R/W	00h	24.4.1/455
4006_1001	Service Register (EWM_SERV)	8	W (always reads 0)	00h	24.4.2/456
4006_1002	Compare Low Register (EWM_CMPL)	8	R/W	00h	24.4.3/456
4006_1003	Compare High Register (EWM_CMPH)	8	R/W	FFh	24.4.4/457

24.4.1 Control Register (EWM_CTRL)

The CTRL register is cleared by any reset.

NOTE

INEN, ASSIN and EWMEN bits can be written once after a CPU reset. Modifying these bits more than once, generates a bus transfer error.

Address: 4006_1000h base + 0h offset = 4006_1000h

Bit	7	6	5	4	3	2	1	0
Read	0				INTEN	INEN	ASSIN	EWMEN
Write								
Reset	0	0	0	0	0	0	0	0

EWM_CTRL field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 INTEN	Interrupt Enable. This bit when set and $\overline{\text{EWM\_out}}$ is asserted, an interrupt request is generated. To de-assert interrupt request, user should clear this bit by writing 0.
2 INEN	Input Enable. This bit when set, enables the EWM_in port.
1 ASSIN	EWM_in's Assertion State Select.

Table continues on the next page...

EWM_CTRL field descriptions (continued)

Field	Description
	Default assert state of the EWM_in signal is logic zero. Setting the ASSIN bit inverts the assert state of EWM_in signal to a logic one.
0 EWMEN	EWM enable. This bit when set, enables the EWM module. This resets the EWM counter to zero and deasserts the $\overline{\text{EWM\_out}}$ signal. This bit when unset, keeps the EWM module disabled. It cannot be re-enabled until a next reset, due to the write-once nature of this bit.

24.4.2 Service Register (EWM_SERV)

The SERV register provides the interface from the CPU to the EWM module. It is write-only and reads of this register return zero.

Address: 4006_1000h base + 1h offset = 4006_1001h

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

EWM_SERV field descriptions

Field	Description
SERVICE	The EWM refresh mechanism requires the CPU to write two values to the SERV register: a first data byte of 0xB4, followed by a second data byte of 0x2C. The EWM refresh is invalid if either of the following conditions is true. • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte. This fixed number of cycles is called *EWM_refresh_time*.

24.4.3 Compare Low Register (EWM_CMPL)

The CMPL register is reset to zero after a CPU reset. This provides no minimum time for the CPU to refresh the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

Address: 4006_1000h base + 2h offset = 4006_1002h

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

EWM_CMPL field descriptions

Field	Description
COMPAREL	To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) minimum refresh time is required.

24.4.4 Compare High Register (EWM_CMPH)

The CMPH register is reset to 0xFF after a CPU reset. This provides a maximum of 256 clocks time, for the CPU to refresh the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

The valid values for CMPH are up to 0xFE because the EWM counter never expires when CMPH = 0xFF. The expiration happens only if EWM counter is greater than CMPH.

Address: 4006_1000h base + 3h offset = 4006_1003h

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

EWM_CMPH field descriptions

Field	Description
COMPAREH	To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) maximum refresh time is required.

24.5 Functional Description

The following sections describe functional details of the EWM module.

NOTE

When the BUS_CLK is lost, then EWM module doesn't generate the $\overline{\text{EWM\_out}}$ signal and no refresh operation is possible

24.5.1 The $\overline{\text{EWM\_out}}$ Signal

The $\overline{\text{EWM\_out}}$ is a digital output signal used to gate an external circuit (application specific) that controls critical safety functions. For example, the $\overline{\text{EWM\_out}}$ could be connected to the high voltage transistors circuits.

The $\overline{\text{EWM\_out}}$ signal remains deasserted when the EWM is being regularly refreshed by the CPU within the programmable refresh window, indicating that the application code is executed as expected.

The $\overline{\text{EWM\_out}}$ signal is asserted in any of the following conditions:

- The EWM refresh occurs when the counter value is less than CMPL value.
- The EWM counter value reaches the CMPH value, and no EWM refresh has occurred.
- If functionality of EWM_in pin is enabled and EWM_in pin is asserted while refreshing the EWM.
- After any reset (by the virtue of the external pull-down mechanism on the $\overline{\text{EWM\_out}}$ pin)

The $\overline{\text{EWM\_out}}$ is asserted after any reset by the virtue of the external pull-down mechanism on the $\overline{\text{EWM\_out}}$ signal. Then, to deassert the $\overline{\text{EWM\_out}}$ signal, set EWMEN bit in the CTRL register to enable the EWM.

If the $\overline{\text{EWM\_out}}$ signal shares its pad with a digital I/O pin, on reset this actual pad defers to being an input signal. The pad state is controlled by the $\overline{\text{EWM\_out}}$ signal only after the EWM is enabled by the EWMEN bit in the CTRL register.

Note

$\overline{\text{EWM\_out}}$ pad must be in pull down state when EWM functionality is used and when EWM is under Reset.

24.5.2 The EWM_in Signal

The EWM_in is a digital input signal for safety status of external safety circuits, that allows an external circuit to control the assertion of the EWM_out signal. For example, in the application, an external circuit monitors a critical safety function, and if there is fault with safety function, the external circuit can then actively initiate the EWM_out signal that controls the gating circuit.

The EWM_in signal is ignored if the EWM is disabled, or if INEN bit of CTRL register is cleared, as after any reset.

On enabling the EWM (setting the CTRL[EWMEN] bit) and enabling EWM_in functionality (setting the CTRL[INEN] bit), the EWM_in signal must be in the deasserted state prior to the CPU start refreshing the EWM. This ensures that the EWM_out stays in the deasserted state; otherwise, the EWM_out output signal is asserted.

Note

The user must update the CMPH and CMPL registers prior to enabling the EWM. After enabling the EWM, the counter resets to zero, therefore the user shall provide a reasonable time after a power-on reset for the external monitoring circuit to stabilize. The user shall also ensure that the EWM_in pin is deasserted.

24.5.3 EWM Counter

It is an 8-bit ripple counter fed from a clock source that is independent of the peripheral bus clock source. As the preferred time-out is between 1 ms and 100 ms the actual clock source should be in the kHz range.

The counter is reset to zero after the CPU reset, or when EWM refresh action completes, or at counter overflow. The counter value is not accessible to the CPU.

24.5.4 EWM Compare Registers

The compare registers CMPL and CMPH are write-once after a CPU reset and cannot be modified until another CPU reset occurs.

The EWM compare registers are used to create a refresh window to refresh the EWM module.

It is illegal to program CMPL and CMPH with same value. In this case, as soon as counter reaches (CMPL + 1), EWM_out is asserted.

24.5.5 EWM Refresh Mechanism

Other than the initial configuration of the EWM, the CPU can only access the EWM by the EWM Service Register. The CPU must access the EWM service register with correct write of unique data within the windowed time frame as determined by the CMPL and CMPH registers for correct EWM refresh operation. Therefore, three possible conditions can occur:

Table 24-4. EWM Refresh Mechanisms

Condition	Mechanism
An EWM refresh action completes when: CMPL < Counter < CMPH.	The software behaves as expected and the EWM counter is reset to zero. The $\overline{\text{EWM\_out}}$ output signal remains in the deasserted state if, during the EWM refresh action, the EWM_in input has been in deasserted state..
An EWM refresh action completes when Counter < CMPL	The software refreshes the EWM before the windowed time frame, the counter is reset to zero and the $\overline{\text{EWM\_out}}$ output signal is asserted irrespective of the input $\overline{\text{EWM\_in}}$.
Counter value reaches CMPH prior to completion of EWM refresh action.	Software has not refreshed the EWM. The EWM counter is reset to zero and the $\overline{\text{EWM\_out}}$ output signal is asserted irrespective of the input $\overline{\text{EWM\_in}}$.

24.5.6 EWM Interrupt

When $\overline{\text{EWM\_out}}$ is asserted, an interrupt request is generated to indicate the assertion of the EWM reset out signal. This interrupt is enabled when CTRL[INTEN] is set. Clearing this bit clears the interrupt request but does not affect $\overline{\text{EWM\_out}}$. The $\overline{\text{EWM\_out}}$ signal can be deasserted only by forcing a system reset.

Chapter 25
Watchdog Timer (WDOG)

25.1 Chip-specific WDOG information

25.1.1 WDOG clocks

This table shows the WDOG module clocks and the corresponding chip clocks.

Table 25-1. WDOG clock connections

Module clock	Chip clock
LPO Oscillator	1 kHz LPO Clock or MCGIRCLK depending on SIM_WDOGC[WDOGCLKS]
Alt Clock	Bus Clock
Fast Test Clock	Bus Clock
System Bus Clock	Bus Clock

25.1.2 WDOG low-power modes

This table shows the WDOG low-power modes and the corresponding chip low-power modes.

Table 25-2. WDOG low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS
Power Down	VLLSx

25.2 Introduction

The Watchdog Timer (WDOG) keeps a watch on the system functioning and resets it in case of its failure. Reasons for failure include run-away software code and the stoppage of the system clock that in a safety critical system can lead to serious consequences. In such cases, the watchdog brings the system into a safe state of operation. The watchdog monitors the operation of the system by expecting periodic communication from the software, generally known as servicing or refreshing the watchdog. If this periodic refreshing does not occur, the watchdog resets the system.

25.3 Features

The features of the Watchdog Timer (WDOG) include:

- Clock source input independent from CPU/bus clock. Choice between two clock sources:
  - Low-power oscillator (LPO)
  - External system clock
- Unlock sequence for allowing updates to write-once WDOG control/configuration bits.
- All WDOG control/configuration bits are writable once only within 256 bus clock cycles of being unlocked.
  - You need to always update these bits after unlocking within 256 bus clock cycles. Failure to update these bits resets the system.
- Programmable time-out period specified in terms of number of WDOG clock cycles.
- Ability to test WDOG timer and reset with a flag indicating watchdog test.
  - Quick test—Small time-out value programmed for quick test.
  - Byte test—Individual bytes of timer tested one at a time.
  - Read-only access to the WDOG timer—Allows dynamic check that WDOG timer is operational.

NOTE

Reading the watchdog timer counter while running the watchdog on the bus clock might not give the accurate counter value.

- Windowed refresh option
  - Provides robust check that program flow is faster than expected.
  - Programmable window.
  - Refresh outside window leads to reset.
- Robust refresh mechanism
  - Write values of 0xA602 and 0xB480 to WDOG Refresh Register within 20 bus clock cycles.
- Count of WDOG resets as they occur.
- Configurable interrupt on time-out to provide debug breadcrumbs. This is followed by a reset after 256 bus clock cycles.

25.4 Functional overview

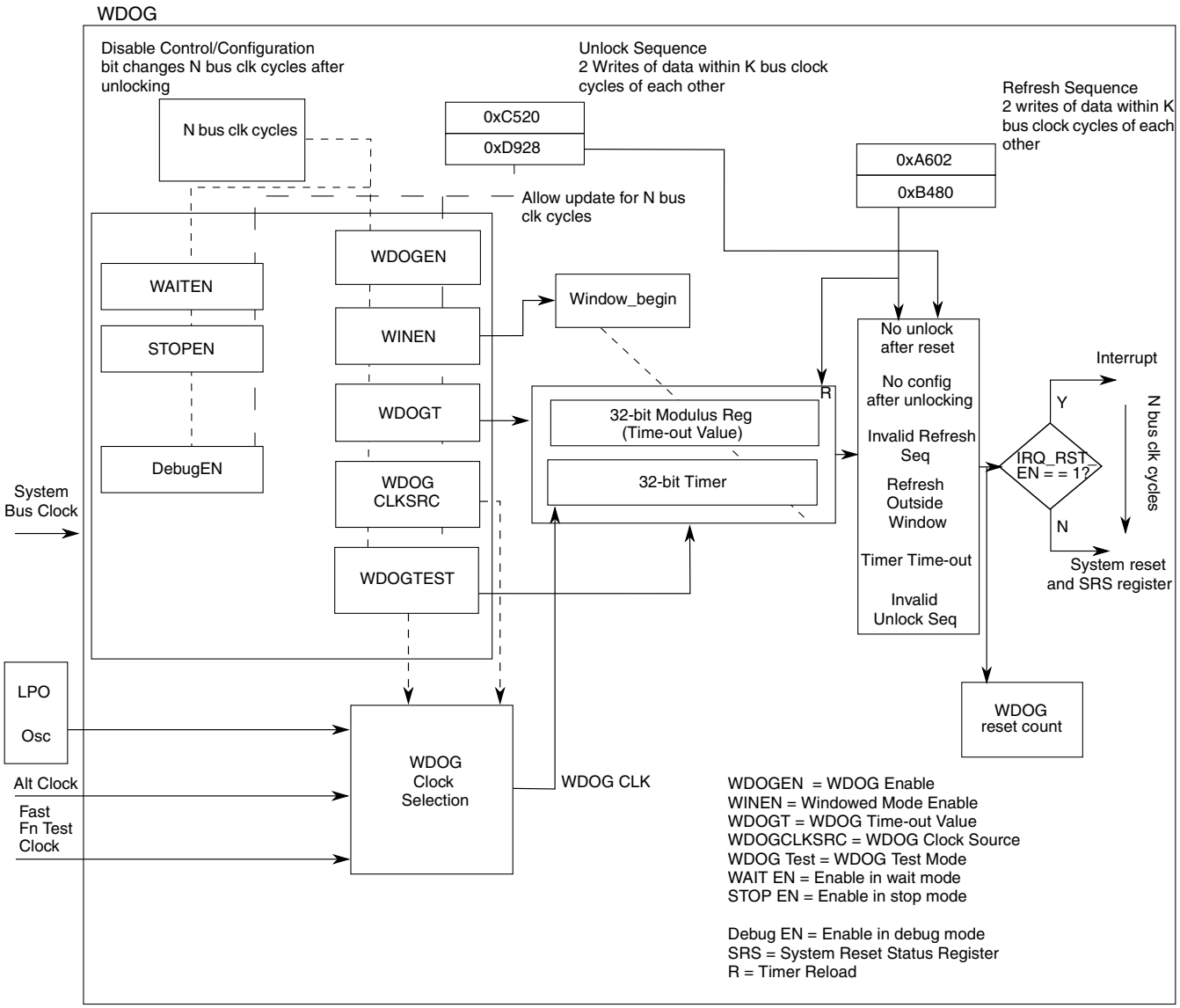


Figure 25-1. WDOG operation

The preceding figure shows the operation of the watchdog. The values for N and K are:

- N = 256
- K = 20

The watchdog is a fail safe mechanism that brings the system into a known initial state in case of its failure due to CPU clock stopping or a run-away condition in code execution. In its simplest form, the watchdog timer runs continuously off a clock source and expects to be serviced periodically, failing which it resets the system. This ensures that the software is executing correctly and has not run away in an unintended direction. Software can adjust the period of servicing or the time-out value for the watchdog timer to meet the needs of the application.

You can select a windowed mode of operation that expects the servicing to be done only in a particular window of the time-out period. An attempted servicing of the watchdog outside this window results in a reset. By operating in this mode, you can get an indication of whether the code is running faster than expected. The window length is also user programmable.

If a system fails to update/refresh the watchdog due to an unknown and persistent cause, it will be caught in an endless cycle of resets from the watchdog. To analyze the cause of such conditions, you can program the watchdog to first issue an interrupt, followed by a reset. In the interrupt service routine, the software can analyze the system stack to aid debugging.

To enhance the independence of watchdog from the system, it runs off an independent LPO oscillator clock. You can also switch over to an alternate clock source if required, through a control register bit.

25.4.1 Unlocking and updating the watchdog

As long as ALLOW_UPDATE in the watchdog control register is set, you can unlock and modify the write-once-only control and configuration registers:

1. Write 0xC520 followed by 0xD928 within 20 bus clock cycles to a specific unlock register (WDOG_UNLOCK).
2. Wait one bus clock cycle. You cannot update registers on the bus clock cycle immediately following the write of the unlock sequence.
3. An update window equal in length to the watchdog configuration time (WCT) opens. Within this window, you can update the configuration and control register bits.

These register bits can be modified only once after unlocking.

If none of the configuration and control registers is updated within the update window, the watchdog issues a reset, that is, interrupt-then-reset, to the system. Trying to unlock the watchdog within the WCT after an initial unlock has no effect. During the update operation, the watchdog timer is not paused and continues running in the background. After the update window closes, the watchdog timer restarts and the watchdog functions according to the new configuration.

The update feature is useful for applications that have an initial, non-safety critical part, where the watchdog is kept disabled or with a conveniently long time-out period. This means the application coder does not have to frequently service the watchdog. After the critical part of the application begins, the watchdog can be reconfigured as needed.

The watchdog issues a reset, that is, interrupt-then-reset if enabled, to the system for any of these invalid unlock sequences:

- Write any value other than 0xC520 or 0xD928 to the unlock register.
- ALLOW_UPDATE is set and a gap of more than 20 bus clock cycles is inserted between the writing of the unlock sequence values.

An attempted refresh operation between the two writes of the unlock sequence and in the WCT time following a successful unlock, goes undetected. Also, see Watchdog Operation with 8-bit access for guidelines related to 8-bit accesses to the unlock register.

Note

A context switch during unlocking and refreshing may lead to a watchdog reset. Therefore, before starting an update or refresh sequence, disable global interrupts to ensure that system interrupts do not cause too much delay, invalidating the sequence. After the sequence finishes, restore global interrupts.

25.4.2 Watchdog configuration time (WCT)

To prevent unintended modification of the watchdog's control and configuration register bits, you are allowed to update them only within a period of 256 bus clock cycles after unlocking. This period is known as the watchdog configuration time (WCT). In addition, these register bits can be modified only once after unlocking them for editing, even after reset.

You must unlock the registers within WCT after system reset, failing which the WDOG issues a reset to the system. In other words, you must write at least the first word of the unlocking sequence within the WCT after reset. After this is done, you have a further 20 bus clock cycles, the maximum allowed gap between the words of the unlock sequence, to complete the unlocking operation. Thereafter, to make sure that you do not forget to configure the watchdog, the watchdog issues a reset if none of the WDOG control and configuration registers is updated in the WCT after unlock. After the close of this window or after the first write, these register bits are locked out from any further changes.

The watchdog timer keeps running according to its default configuration through unlocking and update operations that can extend up to a maximum total of 2xWCT + 20 bus clock cycles. Therefore, it must be ensured that the time-out value for the watchdog is always greater than 2xWCT time + 20 bus clock cycles.

Updates in the write-once registers take effect only after the WCT window closes with the following exceptions for which changes take effect immediately:

- Stop, Wait, and Debug mode enable
- IRQ_RST_EN

The operations of refreshing the watchdog goes undetected during the WCT.

25.4.3 Refreshing the watchdog

A robust refreshing mechanism has been chosen for the watchdog. A valid refresh is a write of 0xA602 followed by 0xB480 within 20 bus clock cycles to watchdog refresh register. If these two values are written more than 20 bus cycles apart or if something other than these two values is written to the register, a watchdog reset, or interrupt-then-reset if enabled, is issued to the system. A valid refresh makes the watchdog timer restart on the next bus clock. Also, an attempted unlock operation in between the two writes of the refresh sequence goes undetected. See Watchdog Operation with 8-bit access for guidelines related to 8-bit accesses to the refresh register.

25.4.4 Windowed mode of operation

In this mode of operation, a restriction is placed on the point in time within the time-out period at which the watchdog can be refreshed. The refresh is considered valid only when the watchdog timer increments beyond a certain count as specified by the watchdog window register. This is known as refreshing the watchdog within a window of the total time-out period. If a refresh is attempted before the timer reaches the window value, the watchdog generates a reset, or interrupt-then-reset if enabled. If there is no refresh at all, the watchdog times out and generates a reset or interrupt-then-reset if enabled.

25.4.5 Watchdog disabled mode of operation

When the watchdog is disabled through the WDOG_EN bit in the watchdog status and control register, the watchdog timer is reset to zero and is disabled from counting until you enable it or it is enabled again by the system reset. In this mode, the watchdog timer cannot be refreshed–there is no requirement to do so while the timer is disabled. However, the watchdog still generates a reset, or interrupt-then-reset if enabled, on a non-time-out exception. See Generated Resets and Interrupts. You need to unlock the watchdog before enabling it. A system reset brings the watchdog out of the disabled mode.

25.4.6 Low-power modes of operation

The low-power modes of operation of the watchdog are described in the following table:

Table 25-3. Low-power modes of operation

Mode	Behavior
Wait	If the WDOG is enabled (WAIT_EN = 1), it can run on bus clock or low-power oscillator clock (CLK_SRC = x) to generate interrupt (IRQ_RST_EN=1) followed by a reset on time-out. After reset the WDOG reset counter increments by one.
Stop	Where the bus clock is gated, the WDOG can run only on low-power oscillator clock (CLK_SRC=0) if it is enabled in stop (STOP_EN=1). In this case, the WDOG runs to time-out twice, and then generates a reset from its backup circuitry. Therefore, if you program the watchdog to time-out after 100 ms and then enter such a stop mode, the reset will occur after 200 ms. Also, in this case, no interrupt will be generated irrespective of the value of IRQ_RST_EN bit. After WDOG reset, the WDOG reset counter will also not increment.
Power-Down	The watchdog is • powered off in VLLSx mode

25.4.7 Debug modes of operation

You can program the watchdog to disable in debug modes through DBG_EN in the watchdog control register. This results in the watchdog timer pausing for the duration of the mode. Register read/writes are still allowed, which means that operations like refresh, unlock, and so on are allowed. Upon exit from the mode, the timer resumes its operation from the point of pausing.

The entry of the system into the debug mode does not excuse it from compulsorily configuring the watchdog in the WCT time after unlock, unless the system bus clock is gated off, in which case the internal state machine pauses too. Failing to do so still results in a reset, or interrupt-then-reset, if enabled, to the system. Also, all of the exception conditions that result in a reset to the system, as described in Generated Resets and Interrupts, are still valid in this mode. So, if an exception condition occurs and the system bus clock is on, a reset occurs, or interrupt-then-reset, if enabled.

The entry into Debug mode within WCT after reset is treated differently. The WDOG timer is kept reset to zero and there is no need to unlock and configure it within WCT. You must not try to refresh or unlock the WDOG in this state or unknown behavior may result. Upon exit from this mode, the WDOG timer restarts and the WDOG has to be unlocked and configured within WCT.

25.5 Testing the watchdog

For IEC 60730 and other safety standards, the expectation is that anything that monitors a safety function must be tested, and this test is required to be fault tolerant. To test the watchdog, its main timer and its associated compare and reset logic must be tested. To this end, two tests are implemented for the watchdog, as described in Quick Test and Byte Test. A control bit is provided to put the watchdog into functional test mode. There is also an overriding test-disable control bit which allows the functional test mode to be disabled permanently. After it is set, this test-disable bit can only be cleared by a reset.

These two tests achieve the overall aim of testing the counter functioning and the compare and reset logic.

Note

Do not enable the watchdog interrupt during these tests. If required, you must ensure that the effective time-out value is greater than WCT time. See Generated Resets and Interrupts for more details.

To run a particular test:

1. Select either quick test or byte test..
2. Set a certain test mode bit to put the watchdog in the functional test mode. Setting this bit automatically switches the watchdog timer to a fast clock source. The switching of the clock source is done to achieve a faster time-out and hence a faster test.

In a successful test, the timer times out after reaching the programmed time-out value and generates a system reset.

Note

After emerging from a reset due to a watchdog test, unlock and configure the watchdog. The refresh and unlock operations and interrupt are not automatically disabled in the test mode.

25.5.1 Quick test

In this test, the time-out value of watchdog timer is programmed to a very low value to achieve quick time-out. The only difference between the quick test and the normal mode of the watchdog is that TESTWDOG is set for the quick test. This allows for a faster test of the watchdog reset mechanism.

25.5.2 Byte test

The byte test is a more thorough a test of the watchdog timer. In this test, the timer is split up into its constituent byte-wide stages that are run independently and tested for time-out against the corresponding byte of the time-out value register. The following figure explains the splitting concept:

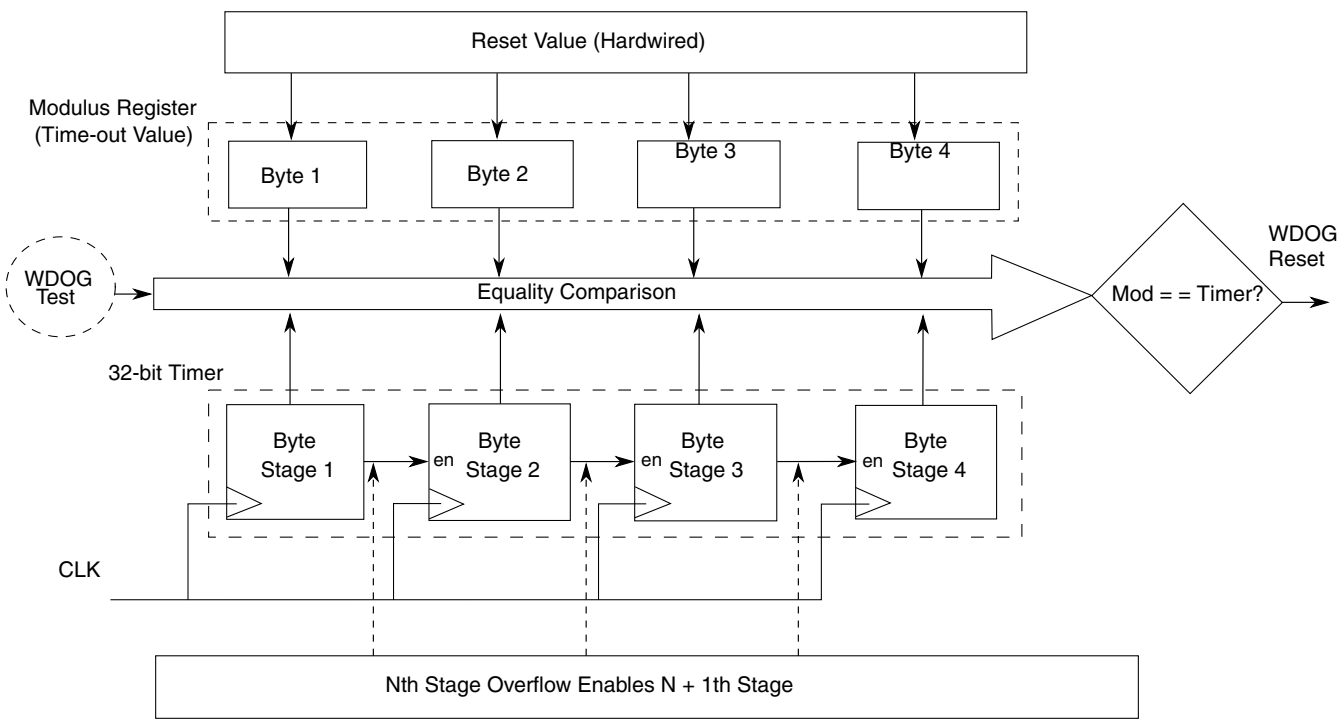


Figure 25-2. Watchdog timer byte splitting

Each stage is an 8-bit synchronous counter followed by combinational logic that generates an overflow signal. The overflow signal acts as an enable to the N + 1th stage.

In the test mode, when an individual byte, N, is tested, byte N – 1 is loaded forcefully with 0xFF, and both these bytes are allowed to run off the clock source. By doing so, the overflow signal from stage N – 1 is generated immediately, enabling counter stage N. The Nth stage runs and compares with the Nth byte of the time-out value register. In this way, the byte N is also tested along with the link between it and the preceding stage. No

other stages, N – 2, N – 3... and N + 1, N + 2... are enabled for the test on byte N. These disabled stages, except the most significant stage of the counter, are loaded with a value of 0xFF.

25.6 Backup reset generator

The backup reset generator generates the final reset which goes out to the system. It has a backup mechanism which ensures that in case the bus clock stops and prevents the main state machine from generating a reset exception/interrupt, the watchdog timer's time-out is separately routed out as a reset to the system. Two successive timer time-outs without an intervening system reset result in the backup reset generator routing out the time-out signal as a reset to the system.

25.7 Generated resets and interrupts

The watchdog generates a reset in the following events, also referred to as exceptions:

- A watchdog time-out
- Failure to unlock the watchdog within WCT time after system reset deassertion
- No update of the control and configuration registers within the WCT window after unlocking. At least one of the following registers must be written to within the WCT window to avoid reset:
  - WDOG_ST_CTRL_H, WDOG_ST_CTRL_L
  - WDOG_TO_VAL_H, WDOG_TO_VAL_L
  - WDOG_WIN_H, WDOG_WIN_L
  - WDOG_PRESCALER
- A value other than the unlock sequence or the refresh sequence is written to the unlock and/or refresh registers, respectively.
- A gap of more than 20 bus cycles exists between the writes of two values of the unlock sequence.
- A gap of more than 20 bus cycles exists between the writes of two values of the refresh sequence.

The watchdog can also generate an interrupt. If IRQ_RST_EN is set, then on the above mentioned events WDOG_ST_CTRL_L[INT_FLG] is set, generating an interrupt. A watchdog reset is also generated WCT time later to ensure the watchdog is fault tolerant. The interrupt can be cleared by writing 1 to INT_FLG.

The gap of WCT between interrupt and reset means that the WDOG time-out value must be greater than WCT. Otherwise, if the interrupt was generated due to a time-out, a second consecutive time-out will occur in that WCT gap. This will trigger the backup reset generator to generate a reset to the system, prematurely ending the interrupt service routine execution. Also, jobs such as counting the number of watchdog resets would not be done.

25.8 Memory map and register definition

This section consists of the memory map and register descriptions.

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_2000	Watchdog Status and Control Register High (WDOG_STCTRLH)	16	R/W	01D2h	25.8.1/473
4005_2002	Watchdog Status and Control Register Low (WDOG_STCTRLL)	16	R/W	0001h	25.8.2/474
4005_2004	Watchdog Time-out Value Register High (WDOG_TOVALH)	16	R/W	004Ch	25.8.3/475
4005_2006	Watchdog Time-out Value Register Low (WDOG_TOVALL)	16	R/W	4B4Ch	25.8.4/475
4005_2008	Watchdog Window Register High (WDOG_WINH)	16	R/W	0000h	25.8.5/476
4005_200A	Watchdog Window Register Low (WDOG_WINL)	16	R/W	0010h	25.8.6/476
4005_200C	Watchdog Refresh register (WDOG_REFRESH)	16	R/W	B480h	25.8.7/477
4005_200E	Watchdog Unlock register (WDOG_UNLOCK)	16	R/W	D928h	25.8.8/477
4005_2010	Watchdog Timer Output Register High (WDOG_TMROUTH)	16	R/W	0000h	25.8.9/477
4005_2012	Watchdog Timer Output Register Low (WDOG_TMROUTL)	16	R/W	0000h	25.8.10/478
4005_2014	Watchdog Reset Count register (WDOG_RSTCNT)	16	R/W	0000h	25.8.11/478
4005_2016	Watchdog Prescaler register (WDOG_PRESC)	16	R/W	0400h	25.8.12/478

25.8.1 Watchdog Status and Control Register High (WDOG_STCTRLH)

Address: 4005_2000h base + 0h offset = 4005_2000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	DISTESTWDOG	BYTESEL[1:0]		TESTSEL	TESTWDOG	0	Reserved	WAITEN	STOPEN	DBGEN	ALLOWUPDATE	WINEN	IRQRSTEN	CLKSRC	WDOGEN
Write																
Reset	0	0	0	0	0	0	0	1	1	1	0	1	0	0	1	0

WDOG_STCTRLH field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 DISTESTWDOG	Allows the WDOG's functional test mode to be disabled permanently. After it is set, it can only be cleared by a reset. It cannot be unlocked for editing after it is set. 0 WDOG functional test mode is not disabled. 1 WDOG functional test mode is disabled permanently until reset.
13–12 BYTESEL[1:0]	This 2-bit field selects the byte to be tested when the watchdog is in the byte test mode. 00 Byte 0 selected 01 Byte 1 selected 10 Byte 2 selected 11 Byte 3 selected
11 TESTSEL	Effective only if TESTWDOG is set. Selects the test to be run on the watchdog timer. 0 Quick test. The timer runs in normal operation. You can load a small time-out value to do a quick test. 1 Byte test. Puts the timer in the byte test mode where individual bytes of the timer are enabled for operation and are compared for time-out against the corresponding byte of the programmed time-out value. Select the byte through BYTESEL[1:0] for testing.
10 TESTWDOG	Puts the watchdog in the functional test mode. In this mode, the watchdog timer and the associated compare and reset generation logic is tested for correct operation. The clock for the timer is switched from the main watchdog clock to the fast clock input for watchdog functional test. The TESTSEL bit selects the test to be run.
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 Reserved	This field is reserved.
7 WAITEN	Enables or disables WDOG in Wait mode. 0 WDOG is disabled in CPU Wait mode. 1 WDOG is enabled in CPU Wait mode.
6 STOPEN	Enables or disables WDOG in Stop mode.

Table continues on the next page...

WDOG_STCTRLH field descriptions (continued)

Field	Description
	0 WDOG is disabled in CPU Stop mode. 1 WDOG is enabled in CPU Stop mode.
5 DBGEN	Enables or disables WDOG in Debug mode. 0 WDOG is disabled in CPU Debug mode. 1 WDOG is enabled in CPU Debug mode.
4 ALLOWUPDATE	Enables updates to watchdog write-once registers, after the reset-triggered initial configuration window (WCT) closes, through unlock sequence. 0 No further updates allowed to WDOG write-once registers. 1 WDOG write-once registers can be unlocked for updating.
3 WINEN	Enables Windowing mode. 0 Windowing mode is disabled. 1 Windowing mode is enabled.
2 IRQRSTEN	Used to enable the debug breadcrumbs feature. A change in this bit is updated immediately, as opposed to updating after WCT. 0 WDOG time-out generates reset only. 1 WDOG time-out initially generates an interrupt. After WCT, it generates a reset.
1 CLKSRC	Selects clock source for the WDOG timer and other internal timing operations. 0 WDOG clock sourced from LPO . 1 WDOG clock sourced from alternate clock source.
0 WDOGEN	Enables or disables the WDOG's operation. In the disabled state, the watchdog timer is kept in the reset state, but the other exception conditions can still trigger a reset/interrupt. A change in the value of this bit must be held for more than one WDOG_CLK cycle for the WDOG to be enabled or disabled. 0 WDOG is disabled. 1 WDOG is enabled.

25.8.2 Watchdog Status and Control Register Low (WDOG_STCTRLL)

Address: 4005_2000h base + 2h offset = 4005_2002h

Bit	15	14	13	12	11	10	9	8
Read Write	INTFLG	Reserved						
Reset	0	0	0	0	0	0	0	0

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

WDOG_STCTRLL field descriptions

Field	Description
15 INTFLG	Interrupt flag. It is set when an exception occurs. IRQRSTEN = 1 is a precondition to set this flag. INTFLG = 1 results in an interrupt being issued followed by a reset, WCT later. The interrupt can be cleared by writing 1 to this bit. It also gets cleared on a system reset.
Reserved	This field is reserved. **NOTE:** Do not modify this field value.

25.8.3 Watchdog Time-out Value Register High (WDOG_TOVALH)

Address: 4005_2000h base + 4h offset = 4005_2004h

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

WDOG_TOVALH field descriptions

Field	Description
TOVALHIGH	Defines the upper 16 bits of the 32-bit time-out value for the watchdog timer. It is defined in terms of cycles of the watchdog clock.

25.8.4 Watchdog Time-out Value Register Low (WDOG_TOVALL)

The time-out value of the watchdog must be set to a minimum of four watchdog clock cycles. This is to take into account the delay in new settings taking effect in the watchdog clock domain.

Address: 4005_2000h base + 6h offset = 4005_2006h

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

WDOG_TOVALL field descriptions

Field	Description
TOVALLOW	Defines the lower 16 bits of the 32-bit time-out value for the watchdog timer. It is defined in terms of cycles of the watchdog clock.

25.8.5 Watchdog Window Register High (WDOG_WINH)

NOTE

You must set the Window Register value lower than the Time-out Value Register.

Address: 4005_2000h base + 8h offset = 4005_2008h

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

WDOG_WINH field descriptions

Field	Description
WINHIGH	Defines the upper 16 bits of the 32-bit window for the windowed mode of operation of the watchdog. It is defined in terms of cycles of the watchdog clock. In this mode, the watchdog can be refreshed only when the timer has reached a value greater than or equal to this window length. A refresh outside this window resets the system or if IRQRSTEN is set, it interrupts and then resets the system.

25.8.6 Watchdog Window Register Low (WDOG_WINL)

NOTE

You must set the Window Register value lower than the Time-out Value Register.

Address: 4005_2000h base + Ah offset = 4005_200Ah

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

WDOG_WINL field descriptions

Field	Description
WINLOW	Defines the lower 16 bits of the 32-bit window for the windowed mode of operation of the watchdog. It is defined in terms of cycles of the pre-scaled watchdog clock. In this mode, the watchdog can be refreshed only when the timer reaches a value greater than or equal to this window length value. A refresh outside of this window resets the system or if IRQRSTEN is set, it interrupts and then resets the system.

25.8.7 Watchdog Refresh register (WDOG_REFRESH)

Address: 4005_2000h base + Ch offset = 4005_200Ch

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

WDOG_REFRESH field descriptions

Field	Description
WDOGREFRESH	Watchdog refresh register. A sequence of 0xA602 followed by 0xB480 within 20 bus clock cycles written to this register refreshes the WDOG and prevents it from resetting the system. Writing a value other than the above mentioned sequence or if the sequence is longer than 20 bus cycles, resets the system, or if IRQRSTEN is set, it interrupts and then resets the system.

25.8.8 Watchdog Unlock register (WDOG_UNLOCK)

Address: 4005_2000h base + Eh offset = 4005_200Eh

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

WDOG_UNLOCK field descriptions

Field	Description
WDOGUNLOCK	Writing the unlock sequence values to this register to makes the watchdog write-once registers writable again. The required unlock sequence is 0xC520 followed by 0xD928 within 20 bus clock cycles. A valid unlock sequence opens a window equal in length to the WCT within which you can update the registers. Writing a value other than the above mentioned sequence or if the sequence is longer than 20 bus cycles, resets the system or if IRQRSTEN is set, it interrupts and then resets the system. The unlock sequence is effective only if ALLOWUPDATE is set.

25.8.9 Watchdog Timer Output Register High (WDOG_TMROUTH)

Address: 4005_2000h base + 10h offset = 4005_2010h

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

WDOG_TMROUTH field descriptions

Field	Description
TIMEROUTHIGH	Shows the value of the upper 16 bits of the watchdog timer.

25.8.10 Watchdog Timer Output Register Low (WDOG_TMROUTL)

During Stop mode, the WDOG_TIMER_OUT will be caught at the pre-stop value of the watchdog timer. After exiting Stop mode, a maximum delay of 1 WDOG_CLK cycle + 3 bus clock cycles will occur before the WDOG_TIMER_OUT starts following the watchdog timer.

Address: 4005_2000h base + 12h offset = 4005_2012h

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

WDOG_TMROUTL field descriptions

Field	Description
TIMEROUTLOW	Shows the value of the lower 16 bits of the watchdog timer.

25.8.11 Watchdog Reset Count register (WDOG_RSTCNT)

Address: 4005_2000h base + 14h offset = 4005_2014h

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

WDOG_RSTCNT field descriptions

Field	Description
RSTCNT	Counts the number of times the watchdog resets the system. This register is reset only on a POR. Writing 1 to the bit to be cleared enables you to clear the contents of this register.

25.8.12 Watchdog Prescaler register (WDOG_PRESC)

Address: 4005_2000h base + 16h offset = 4005_2016h

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

WDOG_PRESC field descriptions

Field	Description
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 PRESCVAL	3-bit prescaler for the watchdog clock source. A value of zero indicates no division of the input WDOG clock. The watchdog clock is divided by (PRESCVAL + 1) to provide the prescaled WDOG_CLK.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

25.9 Watchdog operation with 8-bit access

25.9.1 General guideline

When performing 8-bit accesses to the watchdog's 16-bit registers where the intention is to access both the bytes of a register, place the two 8-bit accesses one after the other in your code.

25.9.2 Refresh and unlock operations with 8-bit access

One exception condition that generates a reset to the system is the write of any value other than those required for a legal refresh/update sequence to the respective refresh and unlock registers.

For an 8-bit access to these registers, writing a correct value requires at least two bus clock cycles, resulting in an invalid value in the registers for one cycle. Therefore, the system is reset even if the intention is to write a correct value to the refresh/unlock register. Keeping this in mind, the exception condition for 8-bit accesses is slightly modified.

Whereas the match for a correct value for a refresh/unlock sequence is as according to the original definition, the match for an incorrect value is done byte-wise on the refresh/unlock rather than for the whole 16-bit value. This means that if the high byte of the refresh/unlock register contains any value other than high bytes of the two values that make up the sequence, it is treated as an exception condition, leading to a reset or interrupt-then-reset. The same holds true for the lower byte of the refresh or unlock register. Take the refresh operation that expects a write of 0xA602 followed by 0xB480 to the refresh register, as an example.

Table 25-4. Refresh for 8-bit access

	WDOG_REFRESH[15:8]	WDOG_REFRESH[7:0]	Sequence value1 or value2 match	Mismatch exception
Current Value	0xB4	0x80	Value2 match	No
Write 1	0xB4	0x02	No match	No
Write 2	0xA6	0x02	Value1 match	No
Write 3	0xB4	0x02	No match	No
Write 4	0xB4	0x80	Value2 match. Sequence complete.	No
Write 5	0x02	0x80	No match	Yes

As shown in the preceding table, the refresh register holds its reset value initially. Thereafter, two 8-bit accesses are performed on the register to write the first value of the refresh sequence. No mismatch exception is registered on the intermediate write, Write1. The sequence is completed by performing two more 8-bit accesses, writing in the second value of the sequence for a successful refresh. It must be noted that the match of value2 takes place only when the complete 16-bit value is correctly written, write4. Hence, the requirement of writing value2 of the sequence within 20 bus clock cycles of value1 is checked by measuring the gap between write2 and write4.

It is reiterated that the condition for matching values 1 and 2 of the refresh or unlock sequence remains unchanged. The difference for 8-bit accesses is that the criterion for detecting a mismatch is less strict. Any 16-bit access still needs to adhere to the original guidelines, mentioned in the sections Refreshing the Watchdog.

25.10 Restrictions on watchdog operation

This section mentions some exceptions to the watchdog operation that may not be apparent to you.

- Restriction on unlock/refresh operations—In the period between the closure of the WCT window after unlock and the actual reload of the watchdog timer, unlock and refresh operations need not be attempted.
- The update and reload of the watchdog timer happens two to three watchdog clocks after WCT window closes, following a successful configuration on unlock.
- Clock Switching Delay—The watchdog uses glitch-free multiplexers at two places – one to choose between the LPO oscillator input and alternate clock input, and the other to choose between the watchdog functional clock and fast clock input for

watchdog functional test. A maximum time period of ~2 clock A cycles plus ~2 clock B cycles elapses from the time a switch is requested to the occurrence of the actual clock switch, where clock A and B are the two input clocks to the clock mux.

- For the windowed mode, there is a two to three bus clock latency between the watchdog counter going past the window value and the same registering in the bus clock domain.
- For proper operation of the watchdog, the watchdog clock must be at least five times slower than the system bus clock at all times. An exception is when the watchdog clock is synchronous to the bus clock wherein the watchdog clock can be as fast as the bus clock.
- WCT must be equivalent to at least three watchdog clock cycles. If not ensured, this means that even after the close of the WCT window, you have to wait for the synchronized system reset to deassert in the watchdog clock domain, before expecting the configuration updates to take effect.
- The time-out value of the watchdog should be set to a minimum of four watchdog clock cycles. This is to take into account the delay in new settings taking effect in the watchdog clock domain.
- You must take care not only to refresh the watchdog within the watchdog timer's actual time-out period, but also provide enough allowance for the time it takes for the refresh sequence to be detected by the watchdog timer, on the watchdog clock.
- Updates cannot be made in the bus clock cycle immediately following the write of the unlock sequence, but one bus clock cycle later.
- It should be ensured that the time-out value for the watchdog is always greater than 2xWCT time + 20 bus clock cycles.
- An attempted refresh operation, in between the two writes of the unlock sequence and in the WCT time following a successful unlock, will go undetected.
- Trying to unlock the watchdog within the WCT time after an initial unlock has no effect.
- The refresh and unlock operations and interrupt are not automatically disabled in the watchdog functional test mode.
- After emerging from a reset due to a watchdog functional test, you are still expected to go through the mandatory steps of unlocking and configuring the watchdog. The watchdog continues to be in its functional test mode and therefore you should pull the watchdog out of the functional test mode within WCT time of reset.

- After emerging from a reset due to a watchdog functional test, you still need to go through the mandatory steps of unlocking and configuring the watchdog.
- You must ensure that both the clock inputs to the glitchless clock multiplexers are alive during the switching of clocks. Failure to do so results in a loss of clock at their outputs.
- There is a gap of two to three watchdog clock cycles from the point that stop mode is entered to the watchdog timer actually pausing, due to synchronization. The same holds true for an exit from the stop mode, this time resulting in a two to three watchdog clock cycle delay in the timer restarting. In case the duration of the stop mode is less than one watchdog clock cycle, the watchdog timer is not guaranteed to pause.
- Consider the case when the first refresh value is written, following which the system enters stop mode with system bus clk still on. If the second refresh value is not written within 20 bus cycles of the first value, the system is reset, or interrupt-then-reset if enabled.

Chapter 26
Inter-Peripheral Crossbar Switch A (XBARA)

26.1 Chip-specific XBARA information

26.1.1 XBARA input signal assignment

Table 26-1. XBARA input assignment

XBARA Input	Assigned Output
XBARA_IN0	VSS
XBARA_IN1	VDD
XBARA_IN2	XB_IN2
XBARA_IN3	XB_IN3
XBARA_IN4	XB_IN4
XBARA_IN5	XB_IN5
XBARA_IN6	XB_IN6
XBARA_IN7	XB_IN7
XBARA_IN8	XB_IN8
XBARA_IN9	XB_IN9
XBARA_IN10	XB_IN10
XBARA_IN11	XB_IN11
XBARA_IN12	CMP0_OUT
XBARA_IN13	CMP1_OUT
XBARA_IN14	CMP2_OUT
XBARA_IN15	CMP3_OUT
XBARA_IN16	FTM0_CH_allTRIG
XBARA_IN17	FTM0_INIT
XBARA_IN18	FTM3_CH_allTRIG
XBARA_IN19	FTM3_INIT
XBARA_IN20	PWMA0_TRG0
XBARA_IN21	PWMA0_TRG1
XBARA_IN22	PWMA1_TRG0

Table continues on the next page...

Table 26-1. XBARA input assignment (continued)

XBARA Input	Assigned Output
XBARA_IN23	PWMA1_TRG1
XBARA_IN24	PWMA2_TRG0
XBARA_IN25	PWMA2_TRG1
XBARA_IN26	PWMA3_TRG0
XBARA_IN27	PWMA3_TRG1
XBARA_IN28	—
XBARA_IN29	PDB0_OUT
XBARA_IN30	—
XBARA_IN31	PDB1_OUT
XBARA_IN32	—
XBARA_IN33	ADCA Scan complete
XBARA_IN34	—
XBARA_IN35	ADCB Scan complete
XBARA_IN36	FTM1_allTRIG
XBARA_IN37	FTM1_INIT
XBARA_IN38	DMA ch0_done[1]
XBARA_IN39	DMA ch1_done[1]
XBARA_IN40	DMA ch6_done[1]
XBARA_IN41	DMA ch7_done[1]
XBARA_IN42	PIT ch0
XBARA_IN43	PIT ch1
XBARA_IN44	—
XBARA_IN45	ENC_CMP/POS_MATCH
XBARA_IN46	AND_OR_INVERT_0
XBARA_IN47	AND_OR_INVERT_1
XBARA_IN48	AND_OR_INVERT_2
XBARA_IN49	AND_OR_INVERT_3
XBARA_IN50	PIT ch2
XBARA_IN51	PIT ch3

1. It works when DMA is triggered by hardware. Software trigger (DMA_TCDn_CSR, START) wouldn't generate the DMA_done for XBARA

26.1.2 XBARA signal output assignment

Table 26-2. XBARA signal output assignment

XBARA Output	Assigned Input
XBARA_OUT0	DMAMUX18
XBARA_OUT1	DMAMUX19

Table continues on the next page...

Table 26-2. XBARA signal output assignment (continued)

XBARA Output	Assigned Input
XBARA_OUT2	DMAMUX20
XBARA_OUT3	DMAMUX21
XBARA_OUT4	XB_OUT4
XBARA_OUT5	XB_OUT5
XBARA_OUT6	XB_OUT6
XBARA_OUT7	XB_OUT7
XBARA_OUT8	XB_OUT8
XBARA_OUT9	XB_OUT9
XBARA_OUT10	XB_OUT10
XBARA_OUT11	XB_OUT11
XBARA_OUT12	ADCA_TRIG
XBARA_OUT13	ADCB_TRIG
XBARA_OUT14	—
XBARA_OUT15	DAC0_12B_SYNC
XBARA_OUT16	CMP0 window/sample input
XBARA_OUT17	CMP1 window/sample input
XBARA_OUT18	CMP2 window/sample input
XBARA_OUT19	CMP3 window/sample input
XBARA_OUT20	PWMA0_EXTA
XBARA_OUT21	PWMA1_EXTA
XBARA_OUT22	PWMA2_EXTA
XBARA_OUT23	PWMA3_EXTA
XBARA_OUT24	PWMA0_EXT_SYNC
XBARA_OUT25	PWMA1_EXT_SYNC
XBARA_OUT26	PWMA2_EXT_SYNC
XBARA_OUT27	PWMA3_EXT_SYNC
XBARA_OUT28	PWMA_EXT_CLK
XBARA_OUT29	PWMA_FAULT0
XBARA_OUT30	PWMA_FAULT1
XBARA_OUT31	PWMA_FAULT2
XBARA_OUT32	PWMA_FAULT3
XBARA_OUT33	PWMA_FORCE
XBARA_OUT34	FTM0_TRIG2
XBARA_OUT35	FTM1_TRIG2
XBARA_OUT36	—
XBARA_OUT37	FTM3_TRIG2
XBARA_OUT38	PDB0_IN_CH_1100
XBARA_OUT39	—
XBARA_OUT40	—

Table continues on the next page...

Table 26-2. XBARA signal output assignment (continued)

XBARA Output	Assigned Input
XBARA_OUT41	PDB1_IN_CH_1100
XBARA_OUT42	FTM1 channel 1 input capture
XBARA_OUT43	—
XBARA_OUT44	ENC_PHA
XBARA_OUT45	ENC_PHB
XBARA_OUT46	ENC_INDEX
XBARA_OUT47	ENC_HOME
XBARA_OUT48	ENC_CAP/Trigger
XBARA_OUT49	FTM0_FAULT3
XBARA_OUT50	FTM1_FAULT1
XBARA_OUT51	—
XBARA_OUT52	FTM3_FAULT3
XBARA_OUT53	—
XBARA_OUT54	—
XBARA_OUT55	—
XBARA_OUT56	—
XBARA_OUT57	—
XBARA_OUT58	EWM_IN

26.2 Introduction

26.2.1 Overview

This module implements an array of M N-input combinational muxes. All muxes share the same N inputs in the same order, but each mux has its own independent select field.

The intended application of this module is to provide a flexible crossbar switch function that allows any input (typically from external GPIO or internal module outputs) to be connected to any output (typically to external GPIO or internal module inputs) under user control. This is used to allow user configuration of data paths between internal modules and between internal modules and GPIO.

A subset of the muxes can be configured to support edge detection and either interrupt or DMA request generation based on detected signal edges on the mux output. This allows signal transitions on the signals feeding the crossbar to trigger interrupts or initiate data transfers via DMA into or out of other system modules.

26.2.2 Features

The XBAR module design includes these distinctive features:

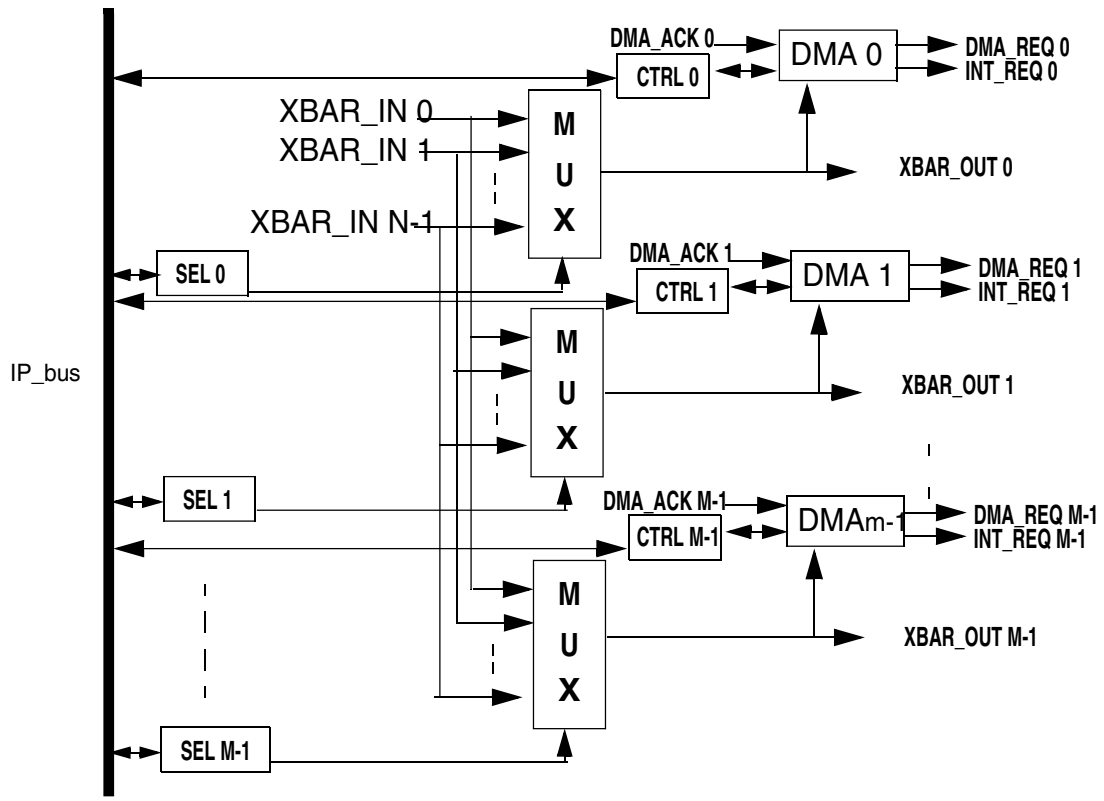
- M identical N-input muxes with individual select fields.
- Edge detection with associated interrupt or DMA request generation for a subset of mux outputs.
- Memory mapped registers with IPBus interface for select and control fields.
- Register write protection input signal.

26.2.3 Modes of Operation

The XBAR module design operates in only a single mode of operation: Functional Mode. The various counting modes are detailed in Functional Mode.

26.2.4 Block Diagram

The block diagram for XBAR is shown in Figure 26-1.



Note: Control field support can be restricted to a subset of outputs. CTRLn represents the memory mapped control fields in the XBAR_CTRL registers for each corresponding XBAR_OUT output. Control fields provide edge status, edge selection, and interrupt/DMA generation controls.

Figure 26-1. XBAR Block Diagram

26.3 Signal Descriptions

The following table summarizes the module's external signals.

Table 26-3. Control Signal Properties

Name	I/O Type	Function	Reset State	Notes
XBAR_OUT [0:NUMOUT-1]	O	Mux Outputs with configurable width	*	
XBAR_IN [0:NUMIN-1]	I	Mux Inputs with configurable width	*	
DMA_REQ	O	DMA request	0	
INT_REQ	O	Interrupt request	0	
DMA_ACK	I	DMA acknowledge	0	

At reset, each output XBAR_OUT[*] contains the reset value of the signal driving XBAR_IN[0].

26.3.1 XBAR_OUT[0:NUM_OUT-1] - MUX Outputs

This is a one-dimensional array of the mux outputs. The value on each output XBAR_OUT[n] is determined by the setting of the corresponding memory mapped register SELn such that XBAR_OUT[n] = XBAR_IN[SELn].

26.3.2 XBAR_IN[0:NUM_IN-1] - MUX Inputs

This is a one-dimensional array consisting of the inputs shared by each mux. All muxes share the same inputs in the same order.

26.3.3 DMA_REQ[n] - DMA Request Output(s)

DMA_REQ[n] is a DMA request to the DMA controller.

26.3.4 DMA_ACK[n] - DMA Acknowledge Input(s)

DMA_ACK[n] is a DMA acknowledge input from the DMA controller.

26.3.5 INT_REQ[n] - Interrupt Request Output(s)

INT_REQ[n] is an interrupt request output to the interrupt controller.

26.4 Memory Map and Register Descriptions

The XBAR module has select registers and control registers.

In the XBAR select registers, the SELn fields select which of the shared inputs (XBAR_IN[*]) is muxed to each mux output (XBAR_OUT[*]). There is one SELn field per mux and therefore one per XBAR_OUT output. Crossbar output XBAR_OUT[n]

presents the value of XBAR_IN[SELn]. Each select register contains two SELn fields. In the first select register, the LSBs contain the select field for mux 0, and the MSBs contain the select field for mux 1. The pattern repeats in subsequent select registers.

The actual signals connected to XBAR_IN and XBAR_OUT are application specific and are described in the Chip Configuration details.

The XBAR control registers configure edge detection, interrupt, and DMA features for a subset of the XBAR_OUT[*] outputs.

XBARA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_9000	Crossbar A Select Register 0 (XBARA_SEL0)	16	R/W	0000h	26.4.1/491
4005_9002	Crossbar A Select Register 1 (XBARA_SEL1)	16	R/W	0000h	26.4.2/491
4005_9004	Crossbar A Select Register 2 (XBARA_SEL2)	16	R/W	0000h	26.4.3/492
4005_9006	Crossbar A Select Register 3 (XBARA_SEL3)	16	R/W	0000h	26.4.4/492
4005_9008	Crossbar A Select Register 4 (XBARA_SEL4)	16	R/W	0000h	26.4.5/493
4005_900A	Crossbar A Select Register 5 (XBARA_SEL5)	16	R/W	0000h	26.4.6/493
4005_900C	Crossbar A Select Register 6 (XBARA_SEL6)	16	R/W	0000h	26.4.7/494
4005_900E	Crossbar A Select Register 7 (XBARA_SEL7)	16	R/W	0000h	26.4.8/494
4005_9010	Crossbar A Select Register 8 (XBARA_SEL8)	16	R/W	0000h	26.4.9/495
4005_9012	Crossbar A Select Register 9 (XBARA_SEL9)	16	R/W	0000h	26.4.10/495
4005_9014	Crossbar A Select Register 10 (XBARA_SEL10)	16	R/W	0000h	26.4.11/496
4005_9016	Crossbar A Select Register 11 (XBARA_SEL11)	16	R/W	0000h	26.4.12/496
4005_9018	Crossbar A Select Register 12 (XBARA_SEL12)	16	R/W	0000h	26.4.13/497
4005_901A	Crossbar A Select Register 13 (XBARA_SEL13)	16	R/W	0000h	26.4.14/497
4005_901C	Crossbar A Select Register 14 (XBARA_SEL14)	16	R/W	0000h	26.4.15/498
4005_901E	Crossbar A Select Register 15 (XBARA_SEL15)	16	R/W	0000h	26.4.16/498
4005_9020	Crossbar A Select Register 16 (XBARA_SEL16)	16	R/W	0000h	26.4.17/499
4005_9022	Crossbar A Select Register 17 (XBARA_SEL17)	16	R/W	0000h	26.4.18/499
4005_9024	Crossbar A Select Register 18 (XBARA_SEL18)	16	R/W	0000h	26.4.19/500
4005_9026	Crossbar A Select Register 19 (XBARA_SEL19)	16	R/W	0000h	26.4.20/500
4005_9028	Crossbar A Select Register 20 (XBARA_SEL20)	16	R/W	0000h	26.4.21/501
4005_902A	Crossbar A Select Register 21 (XBARA_SEL21)	16	R/W	0000h	26.4.22/501
4005_902C	Crossbar A Select Register 22 (XBARA_SEL22)	16	R/W	0000h	26.4.23/502
4005_902E	Crossbar A Select Register 23 (XBARA_SEL23)	16	R/W	0000h	26.4.24/502
4005_9030	Crossbar A Select Register 24 (XBARA_SEL24)	16	R/W	0000h	26.4.25/503
4005_9032	Crossbar A Select Register 25 (XBARA_SEL25)	16	R/W	0000h	26.4.26/503
4005_9034	Crossbar A Select Register 26 (XBARA_SEL26)	16	R/W	0000h	26.4.27/504
4005_9036	Crossbar A Select Register 27 (XBARA_SEL27)	16	R/W	0000h	26.4.28/504
4005_9038	Crossbar A Select Register 28 (XBARA_SEL28)	16	R/W	0000h	26.4.29/505

Table continues on the next page...

XBARA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_903A	Crossbar A Select Register 29 (XBARA_SEL29)	16	R/W	0000h	26.4.30/505
4005_903C	Crossbar A Control Register 0 (XBARA_CTRL0)	16	R/W	0000h	26.4.31/506
4005_903E	Crossbar A Control Register 1 (XBARA_CTRL1)	16	R/W	0000h	26.4.32/508

26.4.1 Crossbar A Select Register 0 (XBARA_SEL0)

Address: 4005_9000h base + 0h offset = 4005_9000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL1						0		SEL0					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL0 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL1	Input (XBARA_INn) to be muxed to XBARA_OUT1 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL0	Input (XBARA_INn) to be muxed to XBARA_OUT0 (refer to Functional Description section for input/output assignment)

26.4.2 Crossbar A Select Register 1 (XBARA_SEL1)

Address: 4005_9000h base + 2h offset = 4005_9002h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL3						0		SEL2					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL1 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL3	Input (XBARA_INn) to be muxed to XBARA_OUT3 (refer to Functional Description section for input/output assignment)

Table continues on the next page...

XBARA_SEL1 field descriptions (continued)

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL2	Input (XBARA_INn) to be muxed to XBARA_OUT2 (refer to Functional Description section for input/output assignment)

26.4.3 Crossbar A Select Register 2 (XBARA_SEL2)

Address: 4005_9000h base + 4h offset = 4005_9004h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL5						0		SEL4					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL2 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL5	Input (XBARA_INn) to be muxed to XBARA_OUT5 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL4	Input (XBARA_INn) to be muxed to XBARA_OUT4 (refer to Functional Description section for input/output assignment)

26.4.4 Crossbar A Select Register 3 (XBARA_SEL3)

Address: 4005_9000h base + 6h offset = 4005_9006h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL7						0		SEL6					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL3 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL7	Input (XBARA_INn) to be muxed to XBARA_OUT7 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

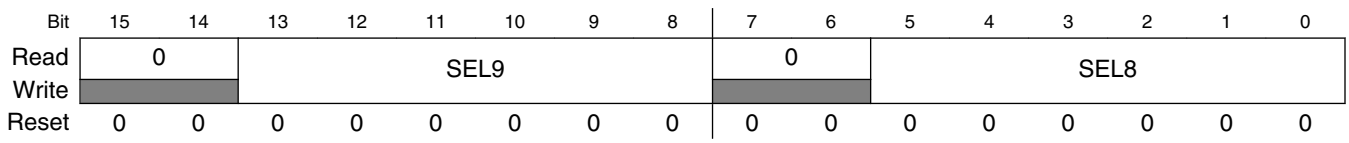
Table continues on the next page...

XBARA_SEL3 field descriptions (continued)

Field	Description
SEL6	Input (XBARA_INn) to be muxed to XBARA_OUT6 (refer to Functional Description section for input/output assignment)

26.4.5 Crossbar A Select Register 4 (XBARA_SEL4)

Address: 4005_9000h base + 8h offset = 4005_9008h

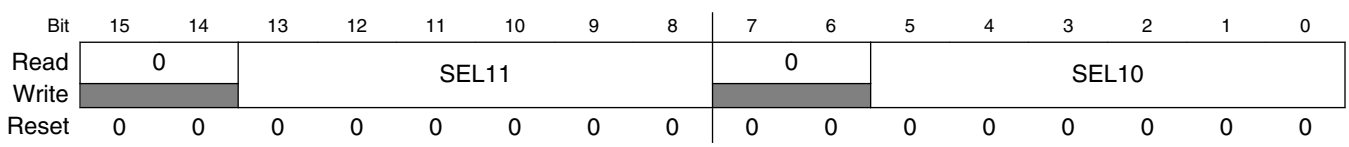


XBARA_SEL4 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL9	Input (XBARA_INn) to be muxed to XBARA_OUT9 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL8	Input (XBARA_INn) to be muxed to XBARA_OUT8 (refer to Functional Description section for input/output assignment)

26.4.6 Crossbar A Select Register 5 (XBARA_SEL5)

Address: 4005_9000h base + Ah offset = 4005_900Ah



XBARA_SEL5 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL11	Input (XBARA_INn) to be muxed to XBARA_OUT11 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL10	Input (XBARA_INn) to be muxed to XBARA_OUT10 (refer to Functional Description section for input/output assignment)

26.4.7 Crossbar A Select Register 6 (XBARA_SEL6)

Address: 4005_9000h base + Ch offset = 4005_900Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL13						0		SEL12					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL6 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL13	Input (XBARA_INn) to be muxed to XBARA_OUT13 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL12	Input (XBARA_INn) to be muxed to XBARA_OUT12 (refer to Functional Description section for input/output assignment)

26.4.8 Crossbar A Select Register 7 (XBARA_SEL7)

Address: 4005_9000h base + Eh offset = 4005_900Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL15						0		SEL14					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL7 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL15	Input (XBARA_INn) to be muxed to XBARA_OUT15 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL14	Input (XBARA_INn) to be muxed to XBARA_OUT14 (refer to Functional Description section for input/output assignment)

26.4.9 Crossbar A Select Register 8 (XBARA_SEL8)

Address: 4005_9000h base + 10h offset = 4005_9010h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL17						0		SEL16					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL8 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL17	Input (XBARA_INn) to be muxed to XBARA_OUT17 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL16	Input (XBARA_INn) to be muxed to XBARA_OUT16 (refer to Functional Description section for input/output assignment)

26.4.10 Crossbar A Select Register 9 (XBARA_SEL9)

Address: 4005_9000h base + 12h offset = 4005_9012h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL19						0		SEL18					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL9 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL19	Input (XBARA_INn) to be muxed to XBARA_OUT19 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL18	Input (XBARA_INn) to be muxed to XBARA_OUT18 (refer to Functional Description section for input/output assignment)

26.4.11 Crossbar A Select Register 10 (XBARA_SEL10)

Address: 4005_9000h base + 14h offset = 4005_9014h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL21						0		SEL20					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL10 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL21	Input (XBARA_INn) to be muxed to XBARA_OUT21 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL20	Input (XBARA_INn) to be muxed to XBARA_OUT20 (refer to Functional Description section for input/output assignment)

26.4.12 Crossbar A Select Register 11 (XBARA_SEL11)

Address: 4005_9000h base + 16h offset = 4005_9016h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL23						0		SEL22					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL11 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL23	Input (XBARA_INn) to be muxed to XBARA_OUT23 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL22	Input (XBARA_INn) to be muxed to XBARA_OUT22 (refer to Functional Description section for input/output assignment)

26.4.13 Crossbar A Select Register 12 (XBARA_SEL12)

Address: 4005_9000h base + 18h offset = 4005_9018h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL25						0		SEL24					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL12 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL25	Input (XBARA_INn) to be muxed to XBARA_OUT25 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL24	Input (XBARA_INn) to be muxed to XBARA_OUT24 (refer to Functional Description section for input/output assignment)

26.4.14 Crossbar A Select Register 13 (XBARA_SEL13)

Address: 4005_9000h base + 1Ah offset = 4005_901Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL27						0		SEL26					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL13 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL27	Input (XBARA_INn) to be muxed to XBARA_OUT27 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL26	Input (XBARA_INn) to be muxed to XBARA_OUT26 (refer to Functional Description section for input/output assignment)

26.4.15 Crossbar A Select Register 14 (XBARA_SEL14)

Address: 4005_9000h base + 1Ch offset = 4005_901Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL29						0		SEL28					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL14 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL29	Input (XBARA_INn) to be muxed to XBARA_OUT29 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL28	Input (XBARA_INn) to be muxed to XBARA_OUT28 (refer to Functional Description section for input/output assignment)

26.4.16 Crossbar A Select Register 15 (XBARA_SEL15)

Address: 4005_9000h base + 1Eh offset = 4005_901Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL31						0		SEL30					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL15 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL31	Input (XBARA_INn) to be muxed to XBARA_OUT31 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL30	Input (XBARA_INn) to be muxed to XBARA_OUT30 (refer to Functional Description section for input/output assignment)

26.4.17 Crossbar A Select Register 16 (XBARA_SEL16)

Address: 4005_9000h base + 20h offset = 4005_9020h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL33						0		SEL32					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL16 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL33	Input (XBARA_INn) to be muxed to XBARA_OUT33 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL32	Input (XBARA_INn) to be muxed to XBARA_OUT32 (refer to Functional Description section for input/output assignment)

26.4.18 Crossbar A Select Register 17 (XBARA_SEL17)

Address: 4005_9000h base + 22h offset = 4005_9022h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL35						0		SEL34					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL17 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL35	Input (XBARA_INn) to be muxed to XBARA_OUT35 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL34	Input (XBARA_INn) to be muxed to XBARA_OUT34 (refer to Functional Description section for input/output assignment)

26.4.19 Crossbar A Select Register 18 (XBARA_SEL18)

Address: 4005_9000h base + 24h offset = 4005_9024h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL37						0		SEL36					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL18 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL37	Input (XBARA_INn) to be muxed to XBARA_OUT37 (refer to Functional Description section for input/ output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL36	Input (XBARA_INn) to be muxed to XBARA_OUT36 (refer to Functional Description section for input/ output assignment)

26.4.20 Crossbar A Select Register 19 (XBARA_SEL19)

Address: 4005_9000h base + 26h offset = 4005_9026h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL39						0		SEL38					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL19 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL39	Input (XBARA_INn) to be muxed to XBARA_OUT39 (refer to Functional Description section for input/ output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL38	Input (XBARA_INn) to be muxed to XBARA_OUT38 (refer to Functional Description section for input/ output assignment)

26.4.21 Crossbar A Select Register 20 (XBARA_SEL20)

Address: 4005_9000h base + 28h offset = 4005_9028h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL41						0		SEL40					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL20 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL41	Input (XBARA_INn) to be muxed to XBARA_OUT41 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL40	Input (XBARA_INn) to be muxed to XBARA_OUT40 (refer to Functional Description section for input/output assignment)

26.4.22 Crossbar A Select Register 21 (XBARA_SEL21)

Address: 4005_9000h base + 2Ah offset = 4005_902Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL43						0		SEL42					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL21 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL43	Input (XBARA_INn) to be muxed to XBARA_OUT43 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL42	Input (XBARA_INn) to be muxed to XBARA_OUT42 (refer to Functional Description section for input/output assignment)

26.4.23 Crossbar A Select Register 22 (XBARA_SEL22)

Address: 4005_9000h base + 2Ch offset = 4005_902Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL45						0		SEL44					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL22 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL45	Input (XBARA_INn) to be muxed to XBARA_OUT45 (refer to Functional Description section for input/ output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL44	Input (XBARA_INn) to be muxed to XBARA_OUT44 (refer to Functional Description section for input/ output assignment)

26.4.24 Crossbar A Select Register 23 (XBARA_SEL23)

Address: 4005_9000h base + 2Eh offset = 4005_902Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL47						0		SEL46					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL23 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL47	Input (XBARA_INn) to be muxed to XBARA_OUT47 (refer to Functional Description section for input/ output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL46	Input (XBARA_INn) to be muxed to XBARA_OUT46 (refer to Functional Description section for input/ output assignment)

26.4.25 Crossbar A Select Register 24 (XBARA_SEL24)

Address: 4005_9000h base + 30h offset = 4005_9030h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL49						0		SEL48					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL24 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL49	Input (XBARA_INn) to be muxed to XBARA_OUT49 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL48	Input (XBARA_INn) to be muxed to XBARA_OUT48 (refer to Functional Description section for input/output assignment)

26.4.26 Crossbar A Select Register 25 (XBARA_SEL25)

Address: 4005_9000h base + 32h offset = 4005_9032h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL51						0		SEL50					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL25 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL51	Input (XBARA_INn) to be muxed to XBARA_OUT51 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL50	Input (XBARA_INn) to be muxed to XBARA_OUT50 (refer to Functional Description section for input/output assignment)

26.4.27 Crossbar A Select Register 26 (XBARA_SEL26)

Address: 4005_9000h base + 34h offset = 4005_9034h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL53						0		SEL52					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL26 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL53	Input (XBARA_INn) to be muxed to XBARA_OUT53 (refer to Functional Description section for input/ output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL52	Input (XBARA_INn) to be muxed to XBARA_OUT52 (refer to Functional Description section for input/ output assignment)

26.4.28 Crossbar A Select Register 27 (XBARA_SEL27)

Address: 4005_9000h base + 36h offset = 4005_9036h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL55						0		SEL54					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL27 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL55	Input (XBARA_INn) to be muxed to XBARA_OUT55 (refer to Functional Description section for input/ output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL54	Input (XBARA_INn) to be muxed to XBARA_OUT54 (refer to Functional Description section for input/ output assignment)

26.4.29 Crossbar A Select Register 28 (XBARA_SEL28)

Address: 4005_9000h base + 38h offset = 4005_9038h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		SEL57						0		SEL56					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL28 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 SEL57	Input (XBARA_INn) to be muxed to XBARA_OUT57 (refer to Functional Description section for input/output assignment)
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL56	Input (XBARA_INn) to be muxed to XBARA_OUT56 (refer to Functional Description section for input/output assignment)

26.4.30 Crossbar A Select Register 29 (XBARA_SEL29)

Address: 4005_9000h base + 3Ah offset = 4005_903Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		0						0		SEL58					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARA_SEL29 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL58	Input (XBARA_INn) to be muxed to XBARA_OUT58 (refer to Functional Description section for input/output assignment)

26.4.31 Crossbar A Control Register 0 (XBARA_CTRL0)

Use this register to configure edge detection, interrupt, and DMA features for the XBAR_OUT0 and XBAR_OUT1 outputs.

The XBAR_CTRL registers are organized similarly to the XBAR_SEL registers, with control fields for two XBAR_OUT outputs in each register. In control register 0, the LSBs contain the control fields for XBAR_OUT0, and the MSBs contain the control fields for XBAR_OUT1.

Address: 4005_9000h base + 3Ch offset = 4005_903Ch

Bit	15	14	13	12	11	10	9	8
Read	0			STS1	EDGE1		IEN1	DEN1
Write				w1c				
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0			STS0	EDGE0		IEN0	DEN0
Write				w1c				
Reset	0	0	0	0	0	0	0	0

XBARA_CTRL0 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 STS1	Edge detection status for XBAR_OUT1 This bit reflects the results of edge detection for XBAR_OUT1. This field is set to 1 when an edge consistent with the current setting of EDGE1 is detected on XBAR_OUT1. This field is cleared by writing 1 to it or by a DMA_ACK1 reception when DEN1 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT1, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT1 1 Active edge detected on XBAR_OUT1
11–10 EDGE1	Active edge for edge detection on XBAR_OUT1 This field selects which edges on XBAR_OUT1 cause STS1 to assert. 00 STS1 never asserts 01 STS1 asserts on rising edges of XBAR_OUT1 10 STS1 asserts on falling edges of XBAR_OUT1 11 STS1 asserts on rising and falling edges of XBAR_OUT1
9 IEN1	Interrupt Enable for XBAR_OUT1

Table continues on the next page...

XBARA_CTRL0 field descriptions (continued)

Field	Description
	This bit enables the interrupt function on the corresponding XBAR_OUT1 output. When the interrupt is enabled, the output INT_REQ1 reflects the value STS1. When the interrupt is disabled, INT_REQ1 remains low. The interrupt request is cleared by writing a 1 to STS1. **Restriction:** IEN1 and DEN1 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
8 DEN1	DMA Enable for XBAR_OUT1 This bit enables the DMA function on the corresponding XBAR_OUT1 output. When enabled, DMA_REQ1 presents the value STS1. When disabled, the DMA_REQ1 output remains low. **Restriction:** IEN1 and DEN1 should not both be set to 1. 0 DMA disabled 1 DMA enabled
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 STS0	Edge detection status for XBAR_OUT0 This bit reflects the results of edge detection for XBAR_OUT0. This field is set to 1 when an edge consistent with the current setting of EDGE0 is detected on XBAR_OUT0. This field is cleared by writing 1 to it or by a DMA_ACK0 reception when DEN0 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT0, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT0 1 Active edge detected on XBAR_OUT0
3–2 EDGE0	Active edge for edge detection on XBAR_OUT0 This field selects which edges on XBAR_OUT0 cause STS0 to assert. 00 STS0 never asserts 01 STS0 asserts on rising edges of XBAR_OUT0 10 STS0 asserts on falling edges of XBAR_OUT0 11 STS0 asserts on rising and falling edges of XBAR_OUT0
1 IEN0	Interrupt Enable for XBAR_OUT0 This bit enables the interrupt function on the corresponding XBAR_OUT0 output. When the interrupt is enabled, the output INT_REQ0 reflects the value STS0. When the interrupt is disabled, INT_REQ0 remains low. The interrupt request is cleared by writing a 1 to STS0. **Restriction:** IEN0 and DEN0 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
0 DEN0	DMA Enable for XBAR_OUT0 This bit enables the DMA function on the corresponding XBAR_OUT0 output. When enabled, DMA_REQ0 presents the value STS0. When disabled, the DMA_REQ0 output remains low.

Table continues on the next page...

XBARA_CTRL0 field descriptions (continued)

Field	Description
	Restriction: IEN0 and DEN0 should not both be set to 1. 0 DMA disabled 1 DMA enabled

26.4.32 Crossbar A Control Register 1 (XBARA_CTRL1)

Use this register to configure edge detection, interrupt, and DMA features for the XBAR_OUT2 and XBAR_OUT3 outputs.

The XBAR_CTRL registers are organized similarly to the XBAR_SEL registers, with control fields for two XBAR_OUT outputs in each register. In control register 1, the LSBs contain the control fields for XBAR_OUT2, and the MSBs contain the control fields for XBAR_OUT3.

Address: 4005_9000h base + 3Eh offset = 4005_903Eh

Bit	15	14	13	12	11	10	9	8
Read	0			STS3	EDGE3		IEN3	DEN3
Write				w1c				
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0			STS2	EDGE2		IEN2	DEN2
Write				w1c				
Reset	0	0	0	0	0	0	0	0

XBARA_CTRL1 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 STS3	Edge detection status for XBAR_OUT3 This bit reflects the results of edge detection for XBAR_OUT3. This field is set to 1 when an edge consistent with the current setting of EDGE3 is detected on XBAR_OUT3. This field is cleared by writing 1 to it or by a DMA_ACK3 reception when DEN3 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT3, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT3 1 Active edge detected on XBAR_OUT3

Table continues on the next page...

XBARA_CTRL1 field descriptions (continued)

Field	Description
11–10 EDGE3	Active edge for edge detection on XBAR_OUT3 This field selects which edges on XBAR_OUT3 cause STS3 to assert. 00 STS3 never asserts 01 STS3 asserts on rising edges of XBAR_OUT3 10 STS3 asserts on falling edges of XBAR_OUT3 11 STS3 asserts on rising and falling edges of XBAR_OUT3
9 IEN3	Interrupt Enable for XBAR_OUT3 This bit enables the interrupt function on the corresponding XBAR_OUT3 output. When the interrupt is enabled, the output INT_REQ3 reflects the value STS3. When the interrupt is disabled, INT_REQ3 remains low. The interrupt request is cleared by writing a 1 to STS3. **Restriction:** IEN3 and DEN3 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
8 DEN3	DMA Enable for XBAR_OUT3 This bit enables the DMA function on the corresponding XBAR_OUT3 output. When enabled, DMA_REQ3 presents the value STS3. When disabled, the DMA_REQ3 output remains low. **Restriction:** IEN3 and DEN3 should not both be set to 1. 0 DMA disabled 1 DMA enabled
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 STS2	Edge detection status for XBAR_OUT2 This bit reflects the results of edge detection for XBAR_OUT2. This field is set to 1 when an edge consistent with the current setting of EDGE2 is detected on XBAR_OUT2. This field is cleared by writing 1 to it or by a DMA_ACK2 reception when DEN2 is set. Writing 0 to the field has no effect. When interrupt or DMA functionality is enabled for XBAR_OUT2, this field is 1 when the interrupt or DMA request is asserted and 0 when the interrupt or DMA request has been cleared. 0 Active edge not yet detected on XBAR_OUT2 1 Active edge detected on XBAR_OUT2
3–2 EDGE2	Active edge for edge detection on XBAR_OUT2 This field selects which edges on XBAR_OUT2 cause STS2 to assert. 00 STS2 never asserts 01 STS2 asserts on rising edges of XBAR_OUT2 10 STS2 asserts on falling edges of XBAR_OUT2 11 STS2 asserts on rising and falling edges of XBAR_OUT2
1 IEN2	Interrupt Enable for XBAR_OUT2

Table continues on the next page...

XBARA_CTRL1 field descriptions (continued)

Field	Description
	This bit enables the interrupt function on the corresponding XBAR_OUT2 output. When the interrupt is enabled, the output INT_REQ2 reflects the value STS2. When the interrupt is disabled, INT_REQ2 remains low. The interrupt request is cleared by writing a 1 to STS2. **Restriction:** IEN2 and DEN2 should not both be set to 1. 0 Interrupt disabled 1 Interrupt enabled
0 DEN2	DMA Enable for XBAR_OUT2 This bit enables the DMA function on the corresponding XBAR_OUT2 output. When enabled, DMA_REQ2 presents the value STS2. When disabled, the DMA_REQ2 output remains low. **Restriction:** IEN2 and DEN2 should not both be set to 1. 0 DMA disabled 1 DMA enabled

26.5 Functional Description

26.5.1 General

The XBAR module has only one mode of operation, functional mode.

26.5.2 Functional Mode

The value of each mux output is XBAR_OUT[n] = XBAR_IN[SELn]. The SELn select values are configured in the XBAR_SEL registers. All muxes share the same inputs in the same order.

A subset of XBAR_OUT[*] outputs has dedicated control fields in a Crossbar Control (XBAR_CTRL) register. Control fields provide the ability to perform edge detection on the corresponding XBAR_OUT output. Edge detection in turn can optionally be used to trigger an interrupt or DMA request. The intention is that, by detecting specified edges on signals propagating through the Crossbar, interrupts or DMA requests can be triggered to perform data transfers to or from other system components.

Control fields include an edge status field (STS), an detected edge type field (EDGE), and interrupt and DMA enable fields (DEN and IEN). STSn is set to 1 when an edge consistent with EDGEn occurs on XBAR_OUT[n]. STSn is cleared by writing 1 to it. Writing 0 as no effect. See Interrupts and DMA Requests for details on the use of STSn for DMA and interrupt request generation.

26.6 Resets

The XBAR module can be reset by only a hard reset, which forces all registers to their reset state.

26.7 Clocks

All sequential functionality is controlled by the Bus Clock.

26.8 Interrupts and DMA Requests

For each XBAR_OUT[*] output with XBAR_CTRL register support, DMA or interrupt functionality can be enabled by setting the corresponding XBAR_CTRL register bit DENn or IENn to 1. DENn and IENn should not be set to 1 at the same time for the same output XBAR_OUT[n].

Setting DENn to 1 enables DMA functionality for XBAR_OUT[n]. When DMA functionality is enabled, the output DMA_REQ[n] reflects the value of STSn. Thus the DMA request asserts when the edge specified by EDGEn is detected on XBAR_OUT[n]. Also, a rising edge on DMA_ACK[n] sets STSn to zero and thus clears the DMA request. When DEN is 0, DMA_REQ[n] is held low and DMA_ACK[n] is ignored.

Setting IENn to 1 enables interrupt functionality for XBAR_OUT[n]. When interrupt functionality is enabled, the output INT_REQ[n] reflects the value of STSn. Thus the interrupt request asserts when the edge specified by EDGEDENn is detected on XBAR_OUT[n]. The interrupt request is cleared by writing a 1 to STSn. When IENn is 0, INT_REQ[n] is held low.

DENn and IENn should not be set to 1 at the same time for the same output XBAR_OUT[n].

Chapter 27
Inter-Peripheral Crossbar Switch B (XBARB)

27.1 chip-specific XBARB information

27.1.1 XBARB signal input assignment

Table 27-1. XBARB signal input assignment

XBARB Input	Assigned Output
XBARB_IN0	CMP0_OUT
XBARB_IN1	CMP1_OUT
XBARB_IN2	CMP2_OUT
XBARB_IN3	CMP3_OUT
XBARB_IN4	FTM0_CH_allTRIG
XBARB_IN5	FTM0_INIT
XBARB_IN6	FTM3_CH_allTRIG
XBARB_IN7	FTM3_INIT
XBARB_IN8	PWMA0_TRG0 /PWMA0_TRIG1
XBARB_IN9	PWMA1_TRG0 /PWMA1_TRG1
XBARB_IN10	PWMA2_TRG0 /PWMA2_TRG1
XBARB_IN11	PWMA3_TRG0 /PWMA3_TRG1
XBARB_IN12	PDB0_OUT
XBARB_IN13	ADCA Scan complete
XBARB_IN14	XB_IN2
XBARB_IN15	XB_IN3
XBARB_IN16	FTM1_allTRIG
XBARB_IN17	FTM1_INIT
XBARB_IN18	DMA ch0_done
XBARB_IN19	DMA ch1_done
XBARB_IN20	XB_IN10
XBARB_IN21	XB_IN11
XBARB_IN22	DMA ch6_done

Table continues on the next page...

Table 27-1. XBARB signal input assignment (continued)

XBARB Input	Assigned Output
XBARB_IN23	DMA ch7_done
XBARB_IN24	PIT ch0
XBARB_IN25	PIT ch1
XBARB_IN26	PDB1_OUT
XBARB_IN27	ADCB Scan complete

27.1.2 XBARB signal output assignment

Table 27-2. XBARB signal output assignment

XBARB Output	Assigned Input
XBARB_OUT0	AOI_IN0
XBARB_OUT1	AOI_IN1
XBARB_OUT2	AOI_IN2
XBARB_OUT3	AOI_IN3
XBARB_OUT4	AOI_IN4
XBARB_OUT5	AOI_IN5
XBARB_OUT6	AOI_IN6
XBARB_OUT7	AOI_IN7
XBARB_OUT8	AOI_IN8
XBARB_OUT9	AOI_IN9
XBARB_OUT10	AOI_IN10
XBARB_OUT11	AOI_IN11
XBARB_OUT12	AOI_IN12
XBARB_OUT13	AOI_IN13
XBARB_OUT14	AOI_IN14
XBARB_OUT15	AOI_IN15

27.2 Introduction

For a description of the general features and functionality of this module, refer to the XBARA description.

27.3 Memory Map and Register Descriptions

This section provides information about the XBARB instance of the inter-peripheral crossbar switch. Refer to the XBARA register details for information about that instance of the module.

This XBAR module has only select registers.

In the XBAR select registers, the SELn fields select which of the shared inputs (XBAR_IN[*]) is muxed to each mux output (XBAR_OUT[*]). There is one SELn field per mux and therefore one per XBAR_OUT output. Crossbar output XBAR_OUT[n] presents the value of XBAR_IN[SELn]. Each select register contains two SELn fields. In the first select register, the LSBs contain the select field for mux 0, and the MSBs contain the select field for mux 1. The pattern repeats in subsequent select registers.

The actual signals connected to XBAR_IN and XBAR_OUT are application specific and are described in the Chip Configuration details.

XBARB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_A000	Crossbar B Select Register 0 (XBARB_SEL0)	16	R/W	0000h	27.3.1/515
4005_A002	Crossbar B Select Register 1 (XBARB_SEL1)	16	R/W	0000h	27.3.2/516
4005_A004	Crossbar B Select Register 2 (XBARB_SEL2)	16	R/W	0000h	27.3.3/516
4005_A006	Crossbar B Select Register 3 (XBARB_SEL3)	16	R/W	0000h	27.3.4/517
4005_A008	Crossbar B Select Register 4 (XBARB_SEL4)	16	R/W	0000h	27.3.5/517
4005_A00A	Crossbar B Select Register 5 (XBARB_SEL5)	16	R/W	0000h	27.3.6/518
4005_A00C	Crossbar B Select Register 6 (XBARB_SEL6)	16	R/W	0000h	27.3.7/518
4005_A00E	Crossbar B Select Register 7 (XBARB_SEL7)	16	R/W	0000h	27.3.8/519

27.3.1 Crossbar B Select Register 0 (XBARB_SEL0)

Address: 4005_A000h base + 0h offset = 4005_A000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL1					0			SEL0				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL0 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL1	Input (XBARB_INn) to be muxed to XBARB_OUT1 (refer to Functional Description section for input/output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL0	Input (XBARB_INn) to be muxed to XBARB_OUT0 (refer to Functional Description section for input/output assignment)

27.3.2 Crossbar B Select Register 1 (XBARB_SEL1)

Address: 4005_A000h base + 2h offset = 4005_A002h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL3					0			SEL2				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL1 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL3	Input (XBARB_INn) to be muxed to XBARB_OUT3 (refer to Functional Description section for input/output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL2	Input (XBARB_INn) to be muxed to XBARB_OUT2 (refer to Functional Description section for input/output assignment)

27.3.3 Crossbar B Select Register 2 (XBARB_SEL2)

Address: 4005_A000h base + 4h offset = 4005_A004h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL5					0			SEL4				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL2 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

XBARB_SEL2 field descriptions (continued)

Field	Description
12–8 SEL5	Input (XBARB_INn) to be muxed to XBARB_OUT5 (refer to Functional Description section for input/output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL4	Input (XBARB_INn) to be muxed to XBARB_OUT4 (refer to Functional Description section for input/output assignment)

27.3.4 Crossbar B Select Register 3 (XBARB_SEL3)

Address: 4005_A000h base + 6h offset = 4005_A006h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL7					0			SEL6				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL3 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL7	Input (XBARB_INn) to be muxed to XBARB_OUT7 (refer to Functional Description section for input/output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL6	Input (XBARB_INn) to be muxed to XBARB_OUT6 (refer to Functional Description section for input/output assignment)

27.3.5 Crossbar B Select Register 4 (XBARB_SEL4)

Address: 4005_A000h base + 8h offset = 4005_A008h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL9					0			SEL8				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL4 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL9	Input (XBARB_INn) to be muxed to XBARB_OUT9 (refer to Functional Description section for input/output assignment)

Table continues on the next page...

XBARB_SEL4 field descriptions (continued)

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL8	Input (XBARB_INn) to be muxed to XBARB_OUT8 (refer to Functional Description section for input/output assignment)

27.3.6 Crossbar B Select Register 5 (XBARB_SEL5)

Address: 4005_A000h base + Ah offset = 4005_A00Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL11					0			SEL10				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL5 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL11	Input (XBARB_INn) to be muxed to XBARB_OUT11 (refer to Functional Description section for input/output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL10	Input (XBARB_INn) to be muxed to XBARB_OUT10 (refer to Functional Description section for input/output assignment)

27.3.7 Crossbar B Select Register 6 (XBARB_SEL6)

Address: 4005_A000h base + Ch offset = 4005_A00Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0			SEL13					0			SEL12				
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

XBARB_SEL6 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL13	Input (XBARB_INn) to be muxed to XBARB_OUT13 (refer to Functional Description section for input/output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

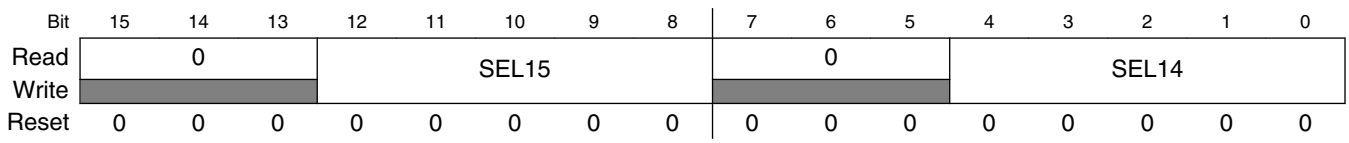
Table continues on the next page...

XBARB_SEL6 field descriptions (continued)

Field	Description
SEL12	Input (XBARB_INn) to be muxed to XBARB_OUT12 (refer to Functional Description section for input/ output assignment)

27.3.8 Crossbar B Select Register 7 (XBARB_SEL7)

Address: 4005_A000h base + Eh offset = 4005_A00Eh



XBARB_SEL7 field descriptions

Field	Description
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12–8 SEL15	Input (XBARB_INn) to be muxed to XBARB_OUT15 (refer to Functional Description section for input/ output assignment)
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL14	Input (XBARB_INn) to be muxed to XBARB_OUT14 (refer to Functional Description section for input/ output assignment)

Chapter 28
Crossbar AND/OR/INVERT (AOI) Module

28.1 Chip-specific AOI information

28.1.1 AOI signal assignment

The AOI block has 16 input signals that are connected directly to the 16 output signals of XBARB. The AOI has 4 output signals that are connected to XBARA inputs.

Table 28-1. AOI signal assignment

AOI Input	Assigned Output	AOI Output	Assigned Input
AOI_IN0	XBARB_OUT0	—	—
AOI_IN1	XBARB_OUT1	—	—
AOI_IN2	XBARB_OUT2	AND_OR_INVERT_0	XBARA_IN46
AOI_IN3	XBARB_OUT3	—	—
AOI_IN4	XBARB_OUT4	—	—
AOI_IN5	XBARB_OUT5	AND_OR_INVERT_1	XBARA_IN47
AOI_IN6	XBARB_OUT6	—	—
AOI_IN7	XBARB_OUT7	—	—
AOI_IN8	XBARB_OUT8	—	—
AOI_IN9	XBARB_OUT9	—	—
AOI_IN10	XBARB_OUT10	AND_OR_INVERT_2	XBARA_IN48
AOI_IN11	XBARB_OUT11	—	—
AOI_IN12	XBARB_OUT12	—	—
AOI_IN13	XBARB_OUT13	AND_OR_INVERT_3	XBARA_IN49
AOI_IN14	XBARB_OUT14	—	—
AOI_IN15	XBARB_OUT15	—	—

28.2 Introduction

The AND/OR/INVERT module (known simply as the AOI module) supports the generation of a configurable number of EVENT signals. Each output EVENTn is a configurable and/or/invert function of four associated AOI inputs: An, Bn, Cn, and Dn.

This module is designed to be integrated in conjuction with one or more inter-peripheral crossbar switch (XBAR) modules. A crossbar switch is typically used to select the 4*n AOI inputs from among available peripheral outputs and GPIO signals. The n EVENTn outputs from the AOI module are typically used as additional inputs to a second crossbar switch, adding to it the ability to connect to its outputs an arbitrary 4-input boolean function of its other inputs.

The AOI controller is a slave peripheral module connecting event input indicators from a variety of device modules and generating event output signals that can be routed to an inter-peripheral crossbar switch or other peripherals. Its programming model is accessed through the standard IPS (Sky Blue) slave interface. The module is designed to be very configurable in terms of the functionality of its integrated AOI functions.

28.2.1 Overview

The AOI module supports a configurable number of event outputs, where each event output represents a user-programmed combinational boolean function based on four event inputs. The key features of this module include:

- Four dedicated inputs for each event output
- User-programmable combinational boolean function evaluation for each event output
- Memory-mapped device connected to a slave peripheral (IPS) bus
- Configurable number of event outputs

NOTE

The connections from the AOI module outputs to other functions is SoC-specific.

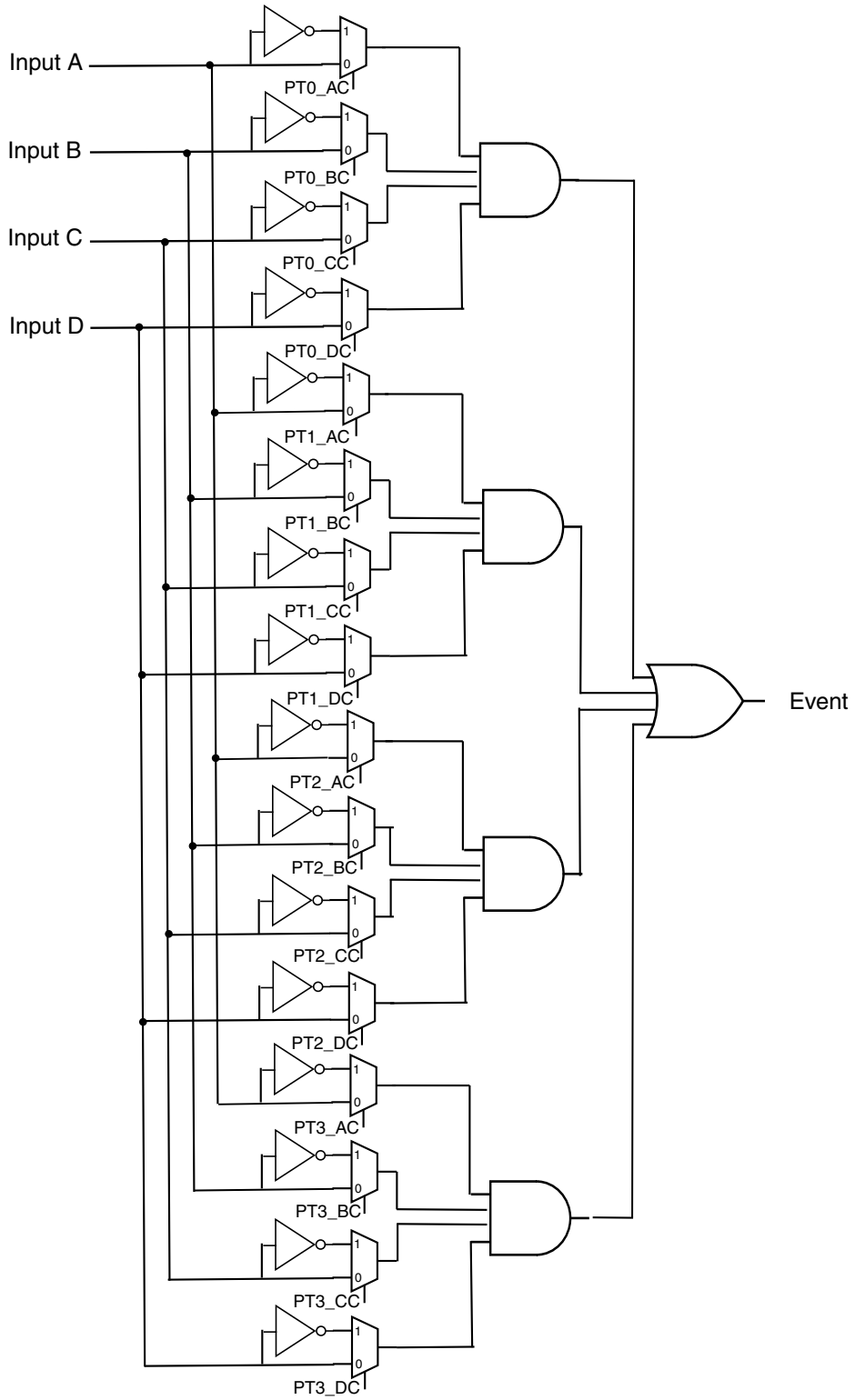


Figure 28-1. Simplified AOI Block Diagram

28.2.2 Features

The major features of the AOI module are summarized below:

- Highly programmable module for creating combinational boolean events for use as hardware triggers
  - Each channel has four event inputs and one output
  - Evaluates a combinational boolean expression as the sum of four products where each product term includes all four selected input sources available as true or complement values
  - Event output is formed as purely combinational logic and operates as a hardware trigger
- Memory-mapped device connected to the slave peripheral (IPS) bus
  - Programming model organized per channel for simplified software

28.2.3 Modes of Operation

The AOI module does not support any special modes of operation. As shown in Figure 28-1, its operation is primarily controlled by the selected event inputs and outputs. Additionally, as a memory-mapped device located on the slave peripheral bus, it responds based strictly on memory address for accesses to its programming model.

The AOI module resides in the slave peripheral *bus clock domain*.

28.3 External Signal Description

The AOI module does not directly support any external interfaces. There may be package input signals (indirectly) connected to the module as event inputs, but since the *AOI does not include any input synchronization hardware*, this function must be handled before the event input signals are routed into the module.

28.4 Memory Map and Register Descriptions

The AOI module supports access to its programming model via a 16-bit peripheral bus connection. The module is designed to support 16-bit accesses only. Functionality for accesses of other widths is undefined.

The AOI module supports a specific number of event outputs. Each output EVENTn outputs a four-term AOI function of four binary inputs: An, Bn, Cn, and Dn. A pair of 16-bit registers configures this four-term AOI function: The two registers BFCRT01n and BFCRT23n define the configuration for the evaluation of the Boolean function defining EVENTn, where *n* is the event output channel number. The BFCRT01n register defines the configuration of product terms 0 and 1, and the BFCRT23n register defines the configuration of product terms 2 and 3.

The AOI module provides a universal Boolean function generator using a four-term sum of products expression with each product term containing true or complement values of the four selected event inputs (An, Bn, Cn, Dn). Specifically, the EVENTn output is defined by the following "4 x 4" Boolean expression:

```
EVENTn
    = (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 0
    | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 1
    | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 2
    | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 3
```

where each selected input of each product term can be configured to produce a logical 0 or 1 or pass the true or complement of the selected event input. Each product termuses 8 bits of configuration information, 2 bits for each of the four selected event inputs. The resulting logic provides a simple yet powerful Boolean function evaluation for defining an event output.

These AOI functions are combinational in nature and are intended to be sampled and used synchronously.

AOI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_B000	Boolean Function Term 0 and 1 Configuration Register for EVENTn (AOI_BFCRT010)	16	R/W	0000h	28.4.1/526
4005_B002	Boolean Function Term 2 and 3 Configuration Register for EVENTn (AOI_BFCRT230)	16	R/W	0000h	28.4.2/527
4005_B004	Boolean Function Term 0 and 1 Configuration Register for EVENTn (AOI_BFCRT011)	16	R/W	0000h	28.4.1/526
4005_B006	Boolean Function Term 2 and 3 Configuration Register for EVENTn (AOI_BFCRT231)	16	R/W	0000h	28.4.2/527
4005_B008	Boolean Function Term 0 and 1 Configuration Register for EVENTn (AOI_BFCRT012)	16	R/W	0000h	28.4.1/526
4005_B00A	Boolean Function Term 2 and 3 Configuration Register for EVENTn (AOI_BFCRT232)	16	R/W	0000h	28.4.2/527
4005_B00C	Boolean Function Term 0 and 1 Configuration Register for EVENTn (AOI_BFCRT013)	16	R/W	0000h	28.4.1/526
4005_B00E	Boolean Function Term 2 and 3 Configuration Register for EVENTn (AOI_BFCRT233)	16	R/W	0000h	28.4.2/527

28.4.1 Boolean Function Term 0 and 1 Configuration Register for EVENTn (AOI_BFCRT01*n*)

Address: 4005_B000h base + 0h offset + (4d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	PT0_AC		PT0_BC		PT0_CC		PT0_DC		PT1_AC		PT1_BC		PT1_CC		PT1_DC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

AOI_BFCRT01*n* field descriptions

Field	Description
15–14 PT0_AC	Product term 0, A input configuration This 2-bit field defines the Boolean evaluation associated with the selected input A in product term 0. 00 Force the A input in this product term to a logical zero 01 Pass the A input in this product term 10 Complement the A input in this product term 11 Force the A input in this product term to a logical one
13–12 PT0_BC	Product term 0, B input configuration This 2-bit field defines the Boolean evaluation associated with the selected input B in product term 0. 00 Force the B input in this product term to a logical zero 01 Pass the B input in this product term 10 Complement the B input in this product term 11 Force the B input in this product term to a logical one
11–10 PT0_CC	Product term 0, C input configuration This 2-bit field defines the Boolean evaluation associated with the selected input C in product term 0. 00 Force the C input in this product term to a logical zero 01 Pass the C input in this product term 10 Complement the C input in this product term 11 Force the C input in this product term to a logical one
9–8 PT0_DC	Product term 0, D input configuration This 2-bit field defines the Boolean evaluation associated with the selected input D in product term 0. 00 Force the D input in this product term to a logical zero 01 Pass the D input in this product term 10 Complement the D input in this product term 11 Force the D input in this product term to a logical one
7–6 PT1_AC	Product term 1, A input configuration This 2-bit field defines the Boolean evaluation associated with the selected input A in product term 1. 00 Force the A input in this product term to a logical zero 01 Pass the A input in this product term

Table continues on the next page...

AOI_BFCRT01*n* field descriptions (continued)

Field	Description
	10 Complement the A input in this product term 11 Force the A input in this product term to a logical one
5–4 PT1_BC	Product term 1, B input configuration This 2-bit field defines the Boolean evaluation associated with the selected input B in product term 1. 00 Force the B input in this product term to a logical zero 01 Pass the B input in this product term 10 Complement the B input in this product term 11 Force the B input in this product term to a logical one
3–2 PT1_CC	Product term 1, C input configuration This 2-bit field defines the Boolean evaluation associated with the selected input C in product term 1. 00 Force the C input in this product term to a logical zero 01 Pass the C input in this product term 10 Complement the C input in this product term 11 Force the C input in this product term to a logical one
PT1_DC	Product term 1, D input configuration This 2-bit field defines the Boolean evaluation associated with the selected input D in product term 1. 00 Force the D input in this product term to a logical zero 01 Pass the D input in this product term 10 Complement the D input in this product term 11 Force the D input in this product term to a logical one

28.4.2 Boolean Function Term 2 and 3 Configuration Register for EVENTn (AOI_BFCRT23*n*)

Address: 4005_B000h base + 2h offset + (4d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	PT2_AC		PT2_BC		PT2_CC		PT2_DC		PT3_AC		PT3_BC		PT3_CC		PT3_DC	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

AOI_BFCRT23*n* field descriptions

Field	Description
15–14 PT2_AC	Product term 2, A input configuration This 2-bit field defines the Boolean evaluation associated with the selected input A in product term 2. 00 Force the A input in this product term to a logical zero 01 Pass the A input in this product term 10 Complement the A input in this product term 11 Force the A input in this product term to a logical one

Table continues on the next page...

AOI_BFCRT23*n* field descriptions (continued)

Field	Description
13–12 PT2_BC	Product term 2, B input configuration This 2-bit field defines the Boolean evaluation associated with the selected input B in product term 2. 00 Force the B input in this product term to a logical zero 01 Pass the B input in this product term 10 Complement the B input in this product term 11 Force the B input in this product term to a logical one
11–10 PT2_CC	Product term 2, C input configuration This 2-bit field defines the Boolean evaluation associated with the selected input C in product term 2. 00 Force the C input in this product term to a logical zero 01 Pass the C input in this product term 10 Complement the C input in this product term 11 Force the C input in this product term to a logical one
9–8 PT2_DC	Product term 2, D input configuration This 2-bit field defines the Boolean evaluation associated with the selected input D in product term 2. 00 Force the D input in this product term to a logical zero 01 Pass the D input in this product term 10 Complement the D input in this product term 11 Force the D input in this product term to a logical one
7–6 PT3_AC	Product term 3, A input configuration This 2-bit field defines the Boolean evaluation associated with the selected input A in product term 3. 00 Force the A input in this product term to a logical zero 01 Pass the A input in this product term 10 Complement the A input in this product term 11 Force the A input in this product term to a logical one
5–4 PT3_BC	Product term 3, B input configuration This 2-bit field defines the Boolean evaluation associated with the selected input B in product term 3. 00 Force the B input in this product term to a logical zero 01 Pass the B input in this product term 10 Complement the B input in this product term 11 Force the B input in this product term to a logical one
3–2 PT3_CC	Product term 3, C input configuration This 2-bit field defines the Boolean evaluation associated with the selected input C in product term 3. 00 Force the C input in this product term to a logical zero 01 Pass the C input in this product term 10 Complement the C input in this product term 11 Force the C input in this product term to a logical one
PT3_DC	Product term 3, D input configuration This 2-bit field defines the Boolean evaluation associated with the selected input D in product term 3.

Table continues on the next page...

AOI_BFCRT23*n* field descriptions (continued)

Field	Description
	00 Force the D input in this product term to a logical zero 01 Pass the D input in this product term 10 Complement the D input in this product term 11 Force the D input in this product term to a logical one

28.5 Functional Description

The AOI is a highly programmable module for creating combinational boolean outputs for use as hardware triggers. Each AOI output channel, as shown in Figure 28-1, has one logic function:

- Evaluation of a combinational boolean expression as a sum of four products where each product term includes all four selected input sources available as true or complement values

A typical application of the AOI module is to be integrated with one or more inter-peripheral crossbar switch modules as illustrated in the following figure. The 20 external inputs are shared by two crossbar switch modules. The crossbar switch on the top is used to select the inputs to four 4-input AOI functions in the AOI module. The outputs of these four AOI functions are output from the AOI module and are added to the original 20 external inputs to provide a total of 24 inputs to the bottom crossbar switch. As a result, the bottom crossbar can not only direct any of the original 20 external inputs to any of its outputs, it can also now direct any one of four 4-input AOI functions of those external inputs to any of its outputs.

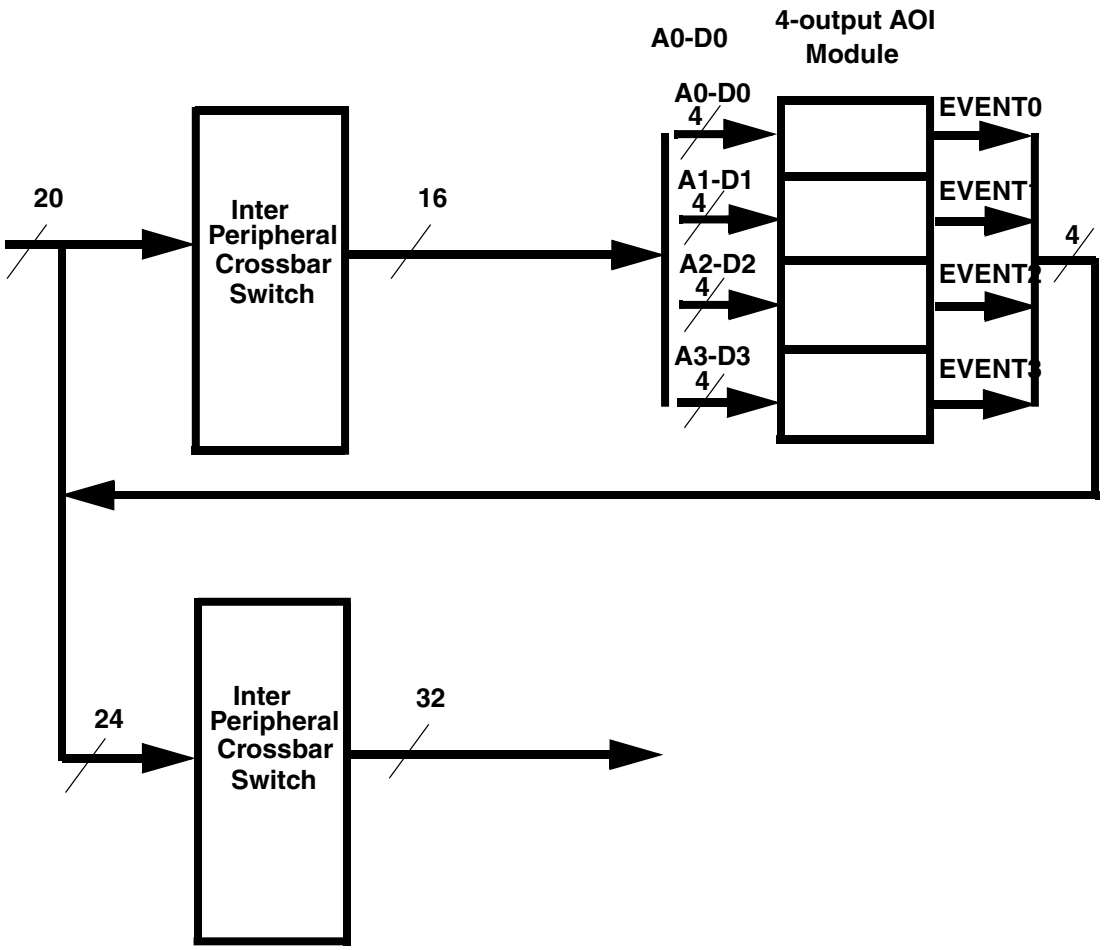


Figure 28-2. Integration Example of AOI with two Inter-Peripheral Crossbar Switches

28.5.1 Configuration Examples for the Boolean Function Evaluation

This section presents examples of the programming model configuration for simple boolean expressions.

The AOI module provides a universal boolean function generator using a four-term sum of products expression with each product term containing true or complement values of the four selected event inputs (A, B, C, D). Specifically, the event output is defined by the following “4 x 4” boolean expression:

```
EVENTn
    = (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 0
    | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 1
```

```
    | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 2
    | (0,An,~An,1) & (0,Bn,~Bn,1) & (0,Cn,~Cn,1) & (0,Dn,~Dn,1)// product term 3
```

where each selected input term in each product term can be configured to produce a logical 0 or 1 or pass the true or complement of the selected event input. Each product term uses eight bits of configuration information, two bits for each of the four selected event inputs. The actual boolean expression implemented in each channel is:

```
EVENTn
    = (PT0_AC[0] & A |  PT0_AC[1] & ~A)// product term 0
                     & (PT0_BC[0] &  B | PTO_BC[1] & ~B)
                     & (PT0_CC[0] &  C | PTO_CC[1] & ~C)
                     & (PT0_DC[0] &  D | PTO_DC[1] & ~D)

    | (PT1_AC[0] & A |  PT1_AC[1] & ~A)// product term 1
                     & (PT1_BC[0] &  B | PT1_BC[1] & ~B)
                     & (PT1_CC[0] &  C | PT1_CC[1] & ~C)
                     & (PT1_DC[0] &  D | PT1_DC[1] & ~D)

    | (PT2_AC[0] & A |  PT2_AC[1] & ~A)// product term 2
                     & (PT2_BC[0] &  B | PT2_BC[1] & ~B)
                     & (PT2_CC[0] &  C | PT2_CC[1] & ~C)
                     & (PT2_DC[0] &  D | PT2_DC[1] & ~D)

    | (PT3_AC[0] & A |  PT3_AC[1] & ~A)// product term 3
                     & (PT3_BC[0] &  B | PT3_BC[1] & ~B)
                     & (PT3_CC[0] &  C | PT3_CC[1] & ~C)
                     & (PT3_DC[0] &  D | PT3_DC[1] & ~D)
```

where the bits of the combined {BFECRT01n,BFCRT23n} registers correspond to the `PT{0-3}_{A,B,C,D}C[1:0]` terms in the equation.

Consider the settings of the combined 32-bit {BFECRT01n,BFCRT23n} registers for several simple boolean expressions as shown in Table 28-2.

Table 28-2. IEVENT_BFECRn Values for Simple Boolean Expressions

Event Output Expression	PT0	PT1	PT2	PT3	{BFCRT01, BFCRT23}
A & B	A & B	0	0	0	01011111_00000000_00000000_00000000
A & B & C	A & B & C	0	0	0	01010111_00000000_00000000_00000000
(A & B & C) + D	A & B & C	D	0	0	01010111_11111101_00000000_00000000
A + B + C + D	A	B	C	D	01111111_11011111_11110111_11111101
(A & ~B) + (~A & B)	A & ~B	~A & B	0	0	01101111_10011111_00000000_00000000

As can be seen in these examples, the resulting logic provides a simple yet powerful boolean function evaluation for defining an event output.

28.5.2 AOI Timing Between Inputs and Outputs

Each EVENTn output of the AOI module is a combination function of its four dedicated inputs An, Bn, Cn, and Dn. Propagation through the AOI and any associated inter-peripheral crossbar switch modules is intended to be single bus clock cycle.

Chapter 29
Oscillator (OSC)

29.1 Introduction

The OSC module is a crystal oscillator. The module, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

29.2 Features and Modes

Key features of the module are listed here.

- Supports 32 kHz crystals (Low Range mode)
- Supports 3–8 MHz, 8–32 MHz crystals and resonators (High Range mode)
- Automatic Gain Control (AGC) to optimize power consumption in high frequency ranges 3–8 MHz, 8–32 MHz using low-power mode
- High gain option in frequency ranges: 32 kHz, 3–8 MHz, and 8–32 MHz
- Voltage and frequency filtering to guarantee clock frequency and stability
- Optionally external input bypass clock from EXTAL signal directly
- One clock for MCU clock system
- Two clocks for on-chip peripherals that can work in Stop modes

Functional Description describes the module's operation in more detail.

29.3 Block Diagram

The OSC module uses a crystal or resonator to generate three filtered oscillator clock signals.Three clocks are output from OSC module: OSCCLK for MCU system, OSCERCLK for on-chip peripherals, and . The OSCCLK can only work in run mode. OSCERCLK and can work in low power modes. For the clock source assignments, refer to the clock distribution information of this MCU.

Refer to the chip configuration details for the external reference clock source in this MCU.

The figure found here shows the block diagram of the OSC module.

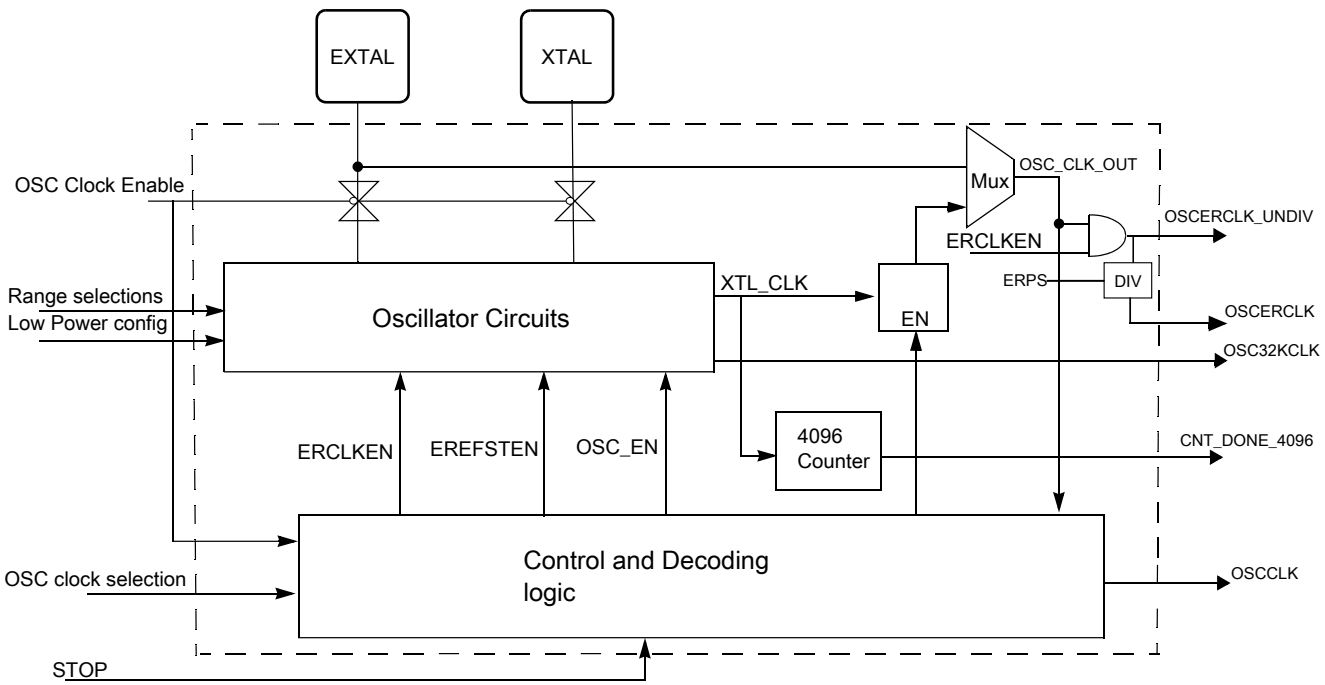


Figure 29-1. OSC Module Block Diagram

29.4 OSC Signal Descriptions

The table found here shows the user-accessible signals available for the OSC module.

Refer to signal multiplexing information for this MCU for more details.

Table 29-1. OSC Signal Descriptions

Signal	Description	I/O
EXTAL	External clock/Oscillator input	I
XTAL	Oscillator output	O

29.5 External Crystal / Resonator Connections

The connections for a crystal/resonator frequency reference are shown in the figures found here.

When using low-frequency, low-power mode, the only external component is the crystal or ceramic resonator itself. In the other oscillator modes, load capacitors (C_x, C_y) and feedback resistor (R_F) are required. The following table shows all possible connections.

Table 29-2. External Caystal/Resonator Connections

Oscillator Mode	Connections
Low-frequency (32 kHz), low-power	Connection 1[1]
Low-frequency (32 kHz), high-gain	Connection 2/Connection 3[2]
High-frequency (3~32 MHz), low-power	Connection 3[1]
High-frequency (3~32 MHz), high-gain	Connection 3

1. With the low-power mode, the oscillator has the internal feedback resistor R_F. Therefore, the feedback resistor must not be externally with the Connection 3.
2. When the load capacitors (Cx, Cy) are greater than 30 pF, use Connection 3.

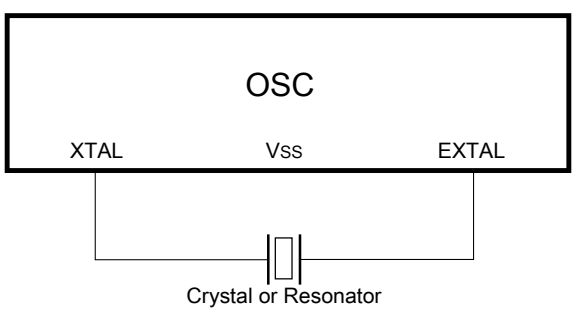


Figure 29-2. Crystal/Ceramic Resonator Connections - Connection 1

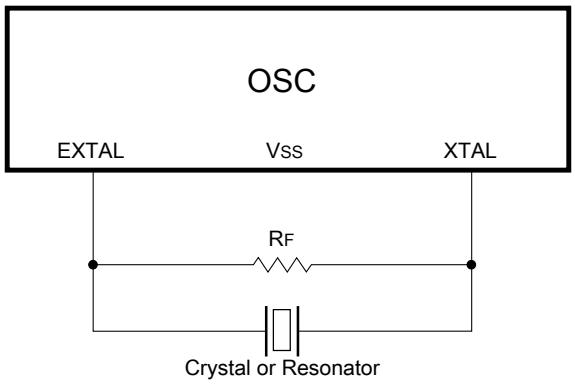


Figure 29-3. Crystal/Ceramic Resonator Connections - Connection 2

NOTE

Connection 1 and Connection 2 should use internal capacitors as the load of the oscillator by configuring the CR[SCxP] bits.

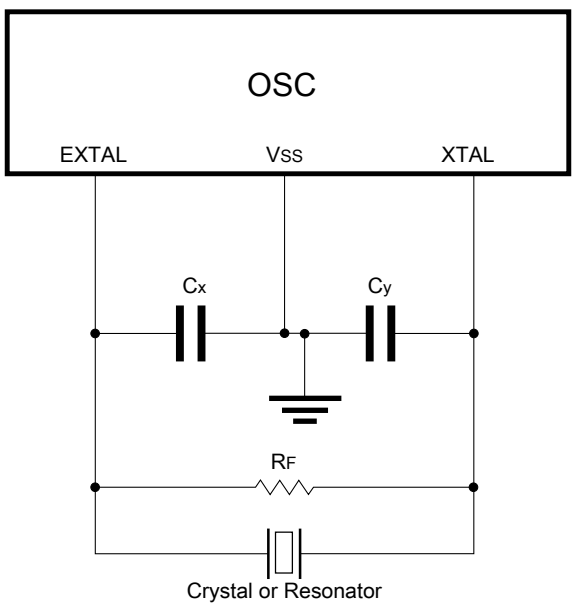


Figure 29-4. Crystal/Ceramic Resonator Connections - Connection 3

29.6 External Clock Connections

In external clock mode, the pins can be connected as shown in the figure found here.

NOTE

XTAL can be used as a GPIO when the GPIO alternate function is configured for it.

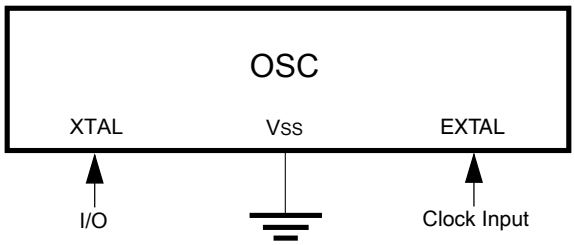


Figure 29-5. External Clock Connections

29.7 Memory Map/Register Definitions

Some oscillator module register bits are typically incorporated into other peripherals such as MCG or SIM.

29.7.1 OSC Memory Map/Register Definition

OSC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_5000	OSC Control Register (OSC_CR)	8	R/W	00h	29.7.1.1/ 537
4006_5002	OSC_DIV (OSC_OSC_DIV)	8	R/W	00h	29.7.1.2/ 539

29.7.1.1 OSC Control Register (OSC_CR)

NOTE

After OSC is enabled and starts generating the clocks, the configurations such as low power and frequency range, must not be changed.

Address: 4006_5000h base + 0h offset = 4006_5000h

Bit	7	6	5	4	3	2	1	0
Read	ERCLKEN	0	EREFSTEN	0	SC2P	SC4P	SC8P	SC16P
Write								
Reset	0	0	0	0	0	0	0	0

OSC_CR field descriptions

Field	Description
7 ERCLKEN	External Reference Enable Enables external reference clock (OSCERCLK) . 0 External reference clock is inactive. 1 External reference clock is enabled.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 EREFSTEN	External Reference Stop Enable Controls whether or not the external reference clock (OSCERCLK) remains enabled when MCU enters Stop mode. 0 External reference clock is disabled in Stop mode. 1 External reference clock stays enabled in Stop mode if ERCLKEN is set before entering Stop mode.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 SC2P	Oscillator 2 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 2 pF capacitor to the oscillator load.
2 SC4P	Oscillator 4 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 4 pF capacitor to the oscillator load.
1 SC8P	Oscillator 8 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 8 pF capacitor to the oscillator load.
0 SC16P	Oscillator 16 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 16 pF capacitor to the oscillator load.

29.7.1.2 OSC_DIV (OSC_OSC_DIV)

OSC Clock divider register.

Address: 4006_5000h base + 2h offset = 4006_5002h

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

OSC_OSC_DIV field descriptions

Field	Description
7–6 ERPS	ERCLK prescaler. These two bits are used to divide the ERCLK output. The un-divided ERCLK output is not affected by these two bits. 00 The divisor ratio is 1. 01 The divisor ratio is 2. 10 The divisor ratio is 4. 11 The divisor ratio is 8.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

29.8 Functional Description

Functional details of the module can be found here.

29.8.1 OSC module states

The states of the OSC module are shown in the following figure. The states and their transitions between each other are described in this section.

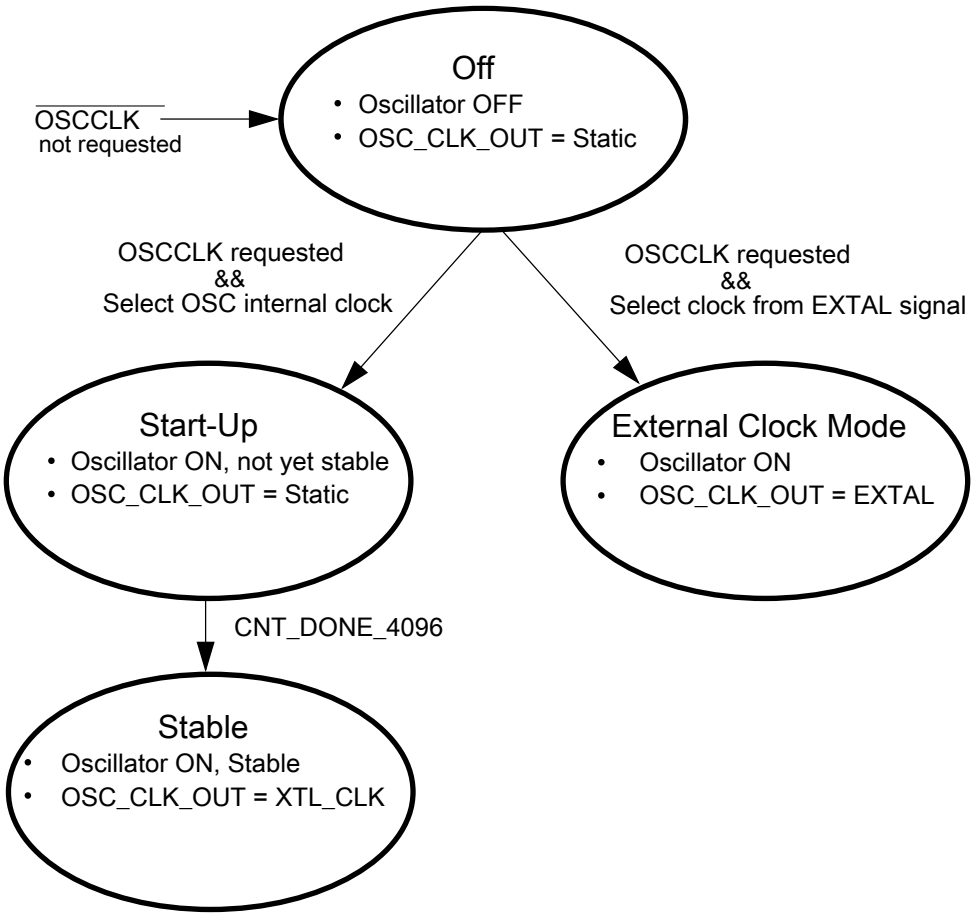


Figure 29-6. OSC Module state diagram

NOTE

XTL_CLK is the clock generated internally from OSC circuits.

29.8.1.1 Off

The OSC enters the Off state when the system does not require OSC clocks. Upon entering this state, XTL_CLK is static unless OSC is configured to select the clock from the EXTAL pad by clearing the external reference clock selection bit. For details regarding the external reference clock source in this MCU, refer to the chip configuration details. The EXTAL and XTAL pins are also decoupled from all other oscillator circuitry in this state. The OSC module circuitry is configured to draw minimal current.

29.8.1.2 Oscillator startup

The OSC enters startup state when it is configured to generate clocks (internally the OSC_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit. In this state, the OSC module is enabled and oscillations are starting up, but have not yet stabilized. When the oscillation amplitude becomes large enough to pass through the input buffer, XTL_CLK begins clocking the counter. When the counter reaches 4096 cycles of XTL_CLK, the oscillator is considered stable and XTL_CLK is passed to the output clock OSC_CLK_OUT.

29.8.1.3 Oscillator Stable

The OSC enters stable state when it is configured to generate clocks (internally the OSC_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit and the counter reaches 4096 cycles of XTL_CLK (when CNT_DONE_4096 is high). In this state, the OSC module is producing a stable output clock on OSC_CLK_OUT. Its frequency is determined by the external components being used.

29.8.1.4 External Clock mode

The OSC enters external clock state when it is enabled and external reference clock selection bit is cleared. For details regarding external reference clock source in this MCU, see the chip configuration details. In this state, the OSC module is set to buffer (with hysteresis) a clock from EXTAL onto the OSC_CLK_OUT. Its frequency is determined by the external clock being supplied.

29.8.2 OSC module modes

The OSC is a pierce-type oscillator that supports external crystals or resonators operating over the frequency ranges shown in Table 29-3. These modes assume the following conditions: OSC is enabled to generate clocks (OSC_EN=1), configured to generate clocks internally (MCG_C2[EREFS] = 1), and some or one of the other peripherals (MCG, Timer, and so on) is configured to use the oscillator output clock (OSC_CLK_OUT).

Table 29-3. Oscillator modes

Mode	Frequency Range
Low-frequency, high-gain	f_{osc_lo} (32.768 kHz) up to f_{osc_lo} (39.0625 kHz)

Table continues on the next page...

Table 29-3. Oscillator modes (continued)

Mode	Frequency Range
High-frequency mode1, high-gain	$f_{osc\_hi\_1}$ (3 MHz) up to $f_{osc\_hi\_1}$ (8 MHz)
High-frequency mode1, low-power	
High-frequency mode2, high-gain	$f_{osc\_hi\_2}$ (8 MHz) up to $f_{osc\_hi\_2}$ (32 MHz)
High-frequency mode2, low-power	

NOTE

For information about low power modes of operation used in this chip and their alignment with some OSC modes, see the chip's Power Management details.

29.8.2.1 Low-Frequency, High-Gain Mode

In Low-frequency, high-gain mode, the oscillator uses a simple inverter-style amplifier. The gain is set to achieve rail-to-rail oscillation amplitudes.

The oscillator input buffer in this mode is single-ended. It provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used.

29.8.2.2 Low-Frequency, Low-Power Mode

In low-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used, the internal feedback resistor is connected, and no external resistor should be used.

In this mode, the amplifier inputs, gain-control input, and input buffer input are all capacitively coupled for leakage tolerance (not sensitive to the DC level of EXTAL).

Also in this mode, all external components except for the resonator itself are integrated, which includes the load capacitors and feeback resistor that biases EXTAL.

29.8.2.3 High-Frequency, High-Gain Mode

In high-frequency, high-gain mode, the oscillator uses a simple inverter-style amplifier. The gain is set to achieve rail-to-rail oscillation amplitudes. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels.

29.8.2.4 High-Frequency, Low-Power Mode

In high-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. In this mode, no external resistor should be used.

The oscillator input buffer in this mode is differential. It provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels.

29.8.3 Counter

The oscillator output clock (OSC_CLK_OUT) is gated off until the counter has detected 4096 cycles of its input clock (XTL_CLK). After 4096 cycles are completed, the counter passes XTL_CLK onto OSC_CLK_OUT. This counting timeout is used to guarantee output clock stability.

29.8.4 Reference clock pin requirements

The OSC module requires use of both the EXTAL and XTAL pins to generate an output clock in Oscillator mode, but requires only the EXTAL pin in External clock mode. The EXTAL and XTAL pins are available for I/O. For the implementation of these pins on this device, refer to the Signal Multiplexing chapter.

29.9 Reset

There is no reset state associated with the OSC module. The counter logic is reset when the OSC is not configured to generate clocks.

There are no sources of reset requests for the OSC module.

29.10 Low power modes operation

When the MCU enters Stop modes, the OSC is functional depending on CR[ERCLKEN] and CR[EREFSETN] bit settings. If both these bits are set, the OSC is in operation.

After waking up from Very Low Leakage Stop (VLLSx) modes, all OSC register bits are reset and initialization is required through software.

29.11 Interrupts

The OSC module does not generate any interrupts.

Chapter 30
Multipurpose Clock Generator (MCG)

30.1 Introduction

The multipurpose clock generator (MCG) module provides several clock source choices for the MCU.

The module contains a frequency-locked loop (FLL) and a phase-locked loop (PLL). The FLL is controllable by either an internal or an external reference clock. The PLL is controllable by the external reference clock. The module can select either an FLL or PLL output clock, or a reference clock (internal or external) as a source for the MCU system clock. The MCG operates in conjuction with a crystal oscillator, which allows an external crystal, ceramic resonator, or another external clock source to produce the external reference clock.

30.1.1 Features

Key features of the MCG module are:

- Frequency-locked loop (FLL):
  - Digitally-controlled oscillator (DCO)
  - DCO frequency range is programmable for up to four different frequency ranges.
  - Option to program and maximize DCO output frequency for a low frequency external reference clock source.
  - Option to prevent FLL from resetting its current locked frequency when switching clock modes if FLL reference frequency is not changed.
  - Internal or external reference clock can be used as the FLL source.
  - Can be used as a clock source for other on-chip peripherals.
- Phase-locked loop (PLL):

  - Voltage-controlled oscillator (VCO)
  - External reference clock is used as the PLL source.
  - Modulo VCO frequency divider
  - Phase/Frequency detector
  - Integrated loop filter
  - Can be used as a clock source for other on-chip peripherals.
- Internal reference clock generator:
  - Slow clock with nine trim bits for accuracy
  - Fast clock with five trim bits
  - Can be used as source clock for the FLL. In FEI mode, only the slow Internal Reference Clock (IRC) can be used as the FLL source.
  - Either the slow or the fast clock can be selected as the clock source for the MCU.
  - Can be used as a clock source for other on-chip peripherals.
- External clock from the Oscillator :
  - Can be used as a source for the FLL and/or the PLL.
  - Can be selected as the clock source for the MCU.
- External clock monitor with reset and interrupt request capability to check for external clock failure when running in FBE, PEE, BLPE, or FEE modes
- Lock detector with interrupt request capability for use with the PLL
- Internal Reference Clocks Auto Trim Machine (ATM) capability using an external clock as a reference
- Reference dividers for both the FLL and the PLL are provided
- Reference dividers for the Fast Internal Reference Clock are provided
-
- MCG FLL Clock (MCGFLLCLK) is provided as a clock source for other on-chip peripherals
- MCG Internal Reference Clock (MCGIRCLK) is provided as a clock source for other on-chip peripherals

This figure presents the block diagram of the MCG module.

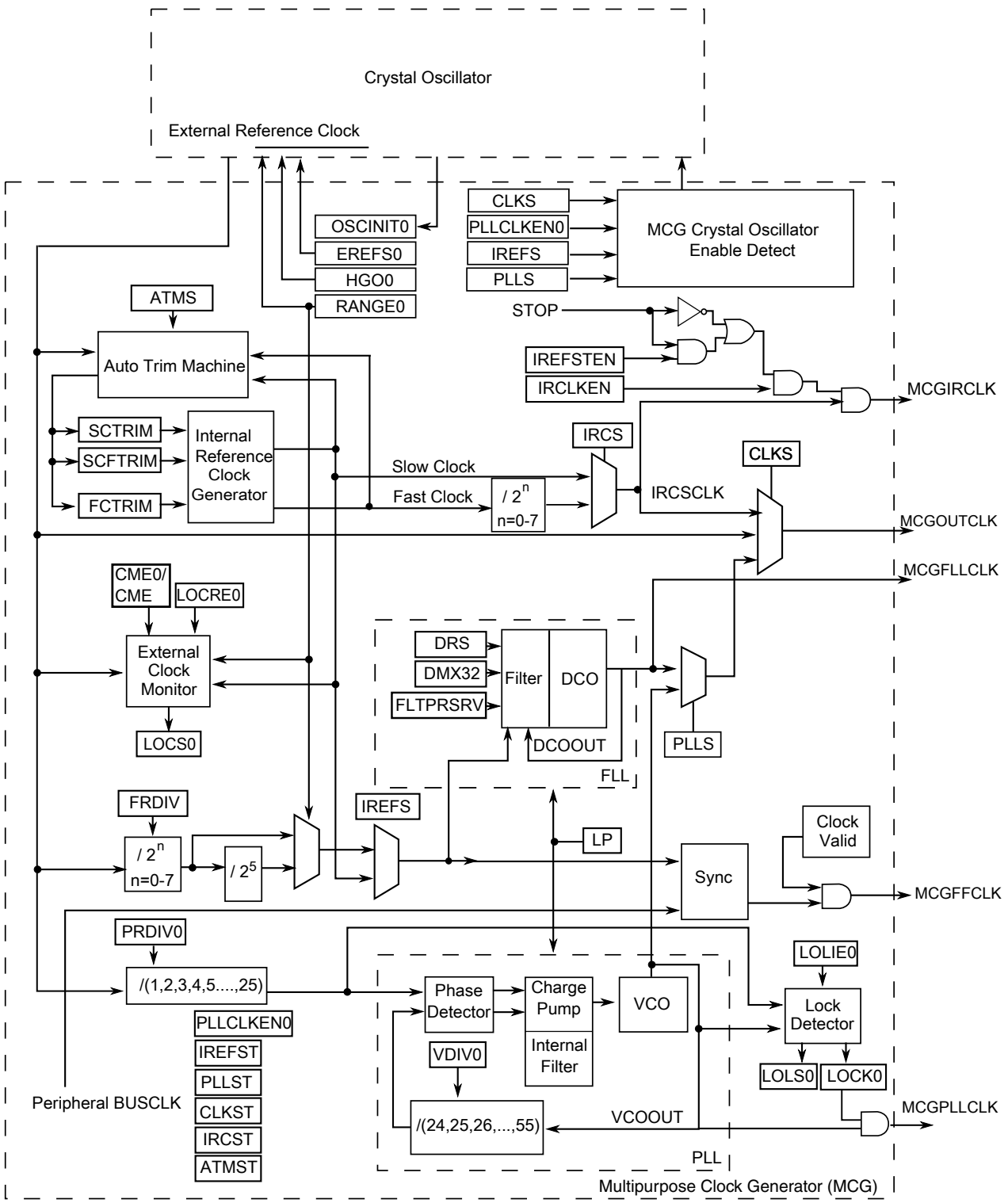


Figure 30-1. Multipurpose Clock Generator (MCG) block diagram

30.1.2 Modes of Operation

The MCG has the following modes of operation: FEI, FEE, FBI, FBE, PBE, PEE, BLPI, BLPE, and Stop. For details, see MCG modes of operation.

30.2 External Signal Description

There are no MCG signals that connect off chip.

30.3 Memory Map/Register Definition

This section includes the memory map and register definition.

The MCG registers can only be written when in supervisor mode. Write accesses when in user mode will result in a bus error. Read accesses may be performed in both supervisor and user mode.

MCG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_4000	MCG Control 1 Register (MCG_C1)	8	R/W	04h	30.3.1/549
4006_4001	MCG Control 2 Register (MCG_C2)	8	R/W	80h	30.3.2/550
4006_4002	MCG Control 3 Register (MCG_C3)	8	R/W	Undefined	30.3.3/551
4006_4003	MCG Control 4 Register (MCG_C4)	8	R/W	See section	30.3.4/551
4006_4004	MCG Control 5 Register (MCG_C5)	8	R/W	00h	30.3.5/553
4006_4005	MCG Control 6 Register (MCG_C6)	8	R/W	00h	30.3.6/554
4006_4006	MCG Status Register (MCG_S)	8	R	10h	30.3.7/555
4006_4008	MCG Status and Control Register (MCG_SC)	8	R/W	02h	30.3.8/557
4006_400A	MCG Auto Trim Compare Value High Register (MCG_ATCVH)	8	R/W	00h	30.3.9/558
4006_400B	MCG Auto Trim Compare Value Low Register (MCG_ATCVL)	8	R/W	00h	30.3.10/ 558
4006_400D	MCG Control 8 Register (MCG_C8)	8	R/W	80h	30.3.11/ 559

30.3.1 MCG Control 1 Register (MCG_C1)

Address: 4006_4000h base + 0h offset = 4006_4000h

Bit	7	6	5	4	3	2	1	0
Read Write	CLKS		FRDIV			IREFS	IRCLKEN	IREFSTEN
Reset	0	0	0	0	0	1	0	0

MCG_C1 field descriptions

Field	Description
7–6 CLKS	Clock Source Select Selects the clock source for MCGOUTCLK . 00 Encoding 0 — Output of FLL or PLL is selected (depends on PLLS control bit). 01 Encoding 1 — Internal reference clock is selected. 10 Encoding 2 — External reference clock is selected. 11 Encoding 3 — Reserved.
5–3 FRDIV	FLL External Reference Divider Selects the amount to divide down the external reference clock for the FLL. The resulting frequency must be in the range 31.25 kHz to 39.0625 kHz (This is required when FLL/DCO is the clock source for MCGOUTCLK . In FBE mode, it is not required to meet this range, but it is recommended in the cases when trying to enter a FLL mode from FBE). 000 If RANGE = 0 , Divide Factor is 1; for all other RANGE values, Divide Factor is 32. 001 If RANGE = 0 , Divide Factor is 2; for all other RANGE values, Divide Factor is 64. 010 If RANGE = 0 , Divide Factor is 4; for all other RANGE values, Divide Factor is 128. 011 If RANGE = 0 , Divide Factor is 8; for all other RANGE values, Divide Factor is 256. 100 If RANGE = 0 , Divide Factor is 16; for all other RANGE values, Divide Factor is 512. 101 If RANGE = 0 , Divide Factor is 32; for all other RANGE values, Divide Factor is 1024. 110 If RANGE = 0 , Divide Factor is 64; for all other RANGE values, Divide Factor is 1280 . 111 If RANGE = 0 , Divide Factor is 128; for all other RANGE values, Divide Factor is 1536 .
2 IREFS	Internal Reference Select Selects the reference clock source for the FLL. 0 External reference clock is selected. 1 The slow internal reference clock is selected.
1 IRCLKEN	Internal Reference Clock Enable Enables the internal reference clock for use as MCGIRCLK. 0 MCGIRCLK inactive. 1 MCGIRCLK active.
0 IREFSTEN	Internal Reference Stop Enable Controls whether or not the internal reference clock remains enabled when the MCG enters Stop mode. 0 Internal reference clock is disabled in Stop mode. 1 Internal reference clock is enabled in Stop mode if IRCLKEN is set or if MCG is in FEI, FBI, or BLPI modes before entering Stop mode.

30.3.2 MCG Control 2 Register (MCG_C2)

Address: 4006_4000h base + 1h offset = 4006_4001h

Bit	7	6	5	4	3	2	1	0
Read Write	LOCRE0	FCFTRIM	RANGE		HGO	EREFS	LP	IRCS
Reset	1	0	0	0	0	0	0	0

MCG_C2 field descriptions

Field	Description
7 LOCRE0	Loss of Clock Reset Enable Determines whether an interrupt or a reset request is made following a loss of OSC0 external reference clock. The LOCRE0 only has an affect when CME0 is set. 0 Interrupt request is generated on a loss of OSC0 external reference clock. 1 Generate a reset request on a loss of OSC0 external reference clock.
6 FCFTRIM	Fast Internal Reference Clock Fine Trim FCFTRIM controls the smallest adjustment of the fast internal reference clock frequency. Setting FCFTRIM increases the period and clearing FCFTRIM decreases the period by the smallest amount possible. If an FCFTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this bit.
5–4 RANGE	Frequency Range Select Selects the frequency range for the crystal oscillator or external clock source. See the Oscillator (OSC) chapter for more details and the device data sheet for the frequency ranges used. 00 Encoding 0 — Low frequency range selected for the crystal oscillator . 01 Encoding 1 — High frequency range selected for the crystal oscillator . 1X Encoding 2 — Very high frequency range selected for the crystal oscillator .
3 HGO	High Gain Oscillator Select Controls the crystal oscillator mode of operation. See the Oscillator (OSC) chapter for more details. 0 Configure crystal oscillator for low-power operation. 1 Configure crystal oscillator for high-gain operation.
2 EREFS	External Reference Select Selects the source for the external reference clock. See the Oscillator (OSC) chapter for more details. 0 External reference clock requested. 1 Oscillator requested.
1 LP	Low Power Select Controls whether the FLL or PLL is disabled in BLPI and BLPE modes. In FBE or PBE modes, setting this bit to 1 will transition the MCG into BLPE mode; in FBI mode, setting this bit to 1 will transition the MCG into BLPI mode. In any other MCG mode, LP bit has no affect. 0 FLL or PLL is not disabled in bypass modes. 1 FLL or PLL is disabled in bypass modes (lower power)

Table continues on the next page...

MCG_C2 field descriptions (continued)

Field	Description
0 IRCS	Internal Reference Clock Select Selects between the fast or slow internal reference clock source. 0 Slow internal reference clock selected. 1 Fast internal reference clock selected.

30.3.3 MCG Control 3 Register (MCG_C3)

Address: 4006_4000h base + 2h offset = 4006_4002h

Bit	7	6	5	4	3	2	1	0
Read Write	SCTRIM							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCG_C3 field descriptions

Field	Description
SCTRIM	Slow Internal Reference Clock Trim Setting SCTRIM [1] controls the slow internal reference clock frequency by controlling the slow internal reference clock period. The SCTRIM bits are binary weighted, that is, bit 1 adjusts twice as much as bit 0. Increasing the binary value increases the period, and decreasing the value decreases the period. An additional fine trim bit is available in C4 register as the SCFTRIM bit. Upon reset, this value is loaded with a factory trim value. If an SCTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.

1. A value for SCTRIM is loaded during reset from a factory programmed location.

30.3.4 MCG Control 4 Register (MCG_C4)

Address: 4006_4000h base + 3h offset = 4006_4003h

Bit	7	6	5	4	3	2	1	0
Read Write	DMX32	DRST_DRS		FCTRIM				SCFTRIM
Reset	0	0	0	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCG_C4 field descriptions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>7
DMX32</td><td>DCO Maximum Frequency with 32.768 kHz Reference

The DMX32 bit controls whether the DCO frequency range is narrowed to its maximum frequency with a 32.768 kHz reference.

The following table identifies settings for the DCO frequency range.

NOTE: The system clocks derived from this source should not exceed their specified maximums. Refer to the device datasheet for supported maximum frequency.

<table>
<tr><th>DRST_DRS</th><th>DMX32</th><th>Reference Range</th><th>FLL Factor</th><th>DCO Range</th></tr>
<tr><td rowspan="2">00</td><td>0</td><td>31.25–39.0625 kHz</td><td>640</td><td>20–25 MHz</td></tr>
<tr><td>1</td><td>32.768 kHz</td><td>732</td><td>24 MHz</td></tr>
<tr><td rowspan="2">01</td><td>0</td><td>31.25–39.0625 kHz</td><td>1280</td><td>40–50 MHz</td></tr>
<tr><td>1</td><td>32.768 kHz</td><td>1464</td><td>48 MHz</td></tr>
<tr><td rowspan="2">10</td><td>0</td><td>31.25–39.0625 kHz</td><td>1920</td><td>60–75 MHz</td></tr>
<tr><td>1</td><td>32.768 kHz</td><td>2197</td><td>72 MHz</td></tr>
<tr><td rowspan="2">11</td><td>0</td><td>31.25–39.0625 kHz</td><td>2560</td><td>80–100 MHz</td></tr>
<tr><td>1</td><td>32.768 kHz</td><td>2929</td><td>96 MHz</td></tr>
</table>

0 DCO has a default range of 25%.
1 DCO is fine-tuned for maximum frequency with 32.768 kHz reference.</td></tr>
<tr><td>6–5
DRST_DRS</td><td>DCO Range Select

The DRS bits select the frequency range for the FLL output, DCOOUT. When the LP bit is set, writes to the DRS bits are ignored. The DRST read field indicates the current frequency range for DCOOUT. The DRST field does not update immediately after a write to the DRS field due to internal synchronization between clock domains. See the DCO Frequency Range table for more details.

00 Encoding 0 — Low range (reset default).
01 Encoding 1 — Mid range.
10 Encoding 2 — Mid-high range.
11 Encoding 3 — High range.</td></tr>
<tr><td>4–1
FCTRIM</td><td>Fast Internal Reference Clock Trim Setting

FCTRIM [1] controls the fast internal reference clock frequency by controlling the fast internal reference clock period. The FCTRIM bits are binary weighted, that is, bit 1 adjusts twice as much as bit 0. Increasing the binary value increases the period, and decreasing the value decreases the period.

If an FCTRIM[3:0] value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.</td></tr>
<tr><td>0
SCFTRIM</td><td>Slow Internal Reference Clock Fine Trim

SCFTRIM [2] controls the smallest adjustment of the slow internal reference clock frequency. Setting SCFTRIM increases the period and clearing SCFTRIM decreases the period by the smallest amount possible.

If an SCFTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this bit.</td></tr>
</table>

1. A value for FCTRIM is loaded during reset from a factory programmed location.
2. A value for SCFTRIM is loaded during reset from a factory programmed location.

30.3.5 MCG Control 5 Register (MCG_C5)

Address: 4006_4000h base + 4h offset = 4006_4004h

Bit	7	6	5	4	3	2	1	0
Read	0	PLLCLKEN	PLLSTEN	0		PRDIV		
Write								
Reset	0	0	0	0	0	0	0	0

MCG_C5 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 PLLCLKEN	PLL Clock Enable Enables PLL independent of PLLS and enables the PLL clock for use as MCGPLLCLK. (PRDIV needs to be programmed to the correct divider to generate a PLL reference clock in a valid reference range prior to setting the PLLCLKEN bit). Setting PLLCLKEN will enable the external oscillator if not already enabled. Whenever the PLL is being enabled by means of the PLLCLKEN bit, and the external oscillator is being used as the reference clock, the OSCINIT 0 bit should be checked to make sure it is set. 0 MCGPLLCLK is inactive. 1 MCGPLLCLK is active.
5 PLLSTEN	PLL Stop Enable Enables the PLL Clock during Normal Stop (In Low Power Stop mode, the PLL clock gets disabled even if PLLSTEN=1). All other power modes, PLLSTEN bit has no affect and does not enable the PLL Clock to run if it is written to 1. 0 MCGPLLCLK and MCGPLLCLK2X are disabled in any of the Stop modes. 1 MCGPLLCLK and MCGPLLCLK2X are enabled if system is in Normal Stop mode.
4–3 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
PRDIV	PLL External Reference Divider Selects the amount to divide down the external reference clock for the PLL0. The resulting frequency must be in the range of 8 MHz to 16 MHz. After the PLL0 is enabled (by setting either PLLCLKEN0 or PLLS), the PRDIV0 value must not be changed when LOCK0 is zero.

Table 30-1. PLL External Reference Divide Factor

PRDIV	Divide Factor
000	1
001	2
010	3
011	4

Table continues on the next page...

MCG_C5 field descriptions (continued)

Field	Description
	Table 30-1. PLL External Reference Divide Factor (continued)

100	5
101	6
110	7
111	8

30.3.6 MCG Control 6 Register (MCG_C6)

Address: 4006_4000h base + 5h offset = 4006_4005h

Bit	7	6	5	4	3	2	1	0
Read Write	LOLIE0	PLLS	CME0	VDIV				
Reset	0	0	0	0	0	0	0	0

MCG_C6 field descriptions

Field	Description
7 LOLIE0	Loss of Lock Interrrupt Enable Determines if an interrupt request is made following a loss of lock indication. This bit only has an effect when LOLS 0 is set. 0 No interrupt request is generated on loss of lock. 1 Generate an interrupt request on loss of lock.
6 PLLS	PLL Select Controls whether the PLL or FLL output is selected as the MCG source when CLKS[1:0]=00. If the PLLS bit is cleared and PLLCLKEN 0 is not set, the PLL is disabled in all modes. If the PLLS is set, the FLL is disabled in all modes. 0 FLL is selected. 1 PLL is selected (PRDIV 0 need to be programmed to the correct divider to generate a PLL reference clock in the range of 8–16 MHz prior to setting the PLLS bit).
5 CME0	Clock Monitor Enable Enables the loss of clock monitoring circuit for the OSC0 external reference mux select. The LOCRE0 bit will determine if a interrupt or a reset request is generated following a loss of OSC0 indication. The CME0 bit must only be set to a logic 1 when the MCG is in an operational mode that uses the external clock (FEE, FBE, PEE, PBE, or BLPE) . Whenever the CME0 bit is set to a logic 1, the value of the RANGE0 bits in the C2 register should not be changed. CME0 bit should be set to a logic 0 before the MCG enters any Stop mode. Otherwise, a reset request may occur while in Stop mode. CME0 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode. 0 External clock monitor is disabled for OSC0. 1 External clock monitor is enabled for OSC0.

Table continues on the next page...

MCG_C6 field descriptions (continued)

Field	Description
VDIV	VCO Divider Selects the amount to divide the VCO output of the PLL. The VDIV bits establish the multiplication factor (M) applied to the reference clock frequency. After the PLL is enabled (by setting either PLLCLKEN or PLLS), the VDIV value must not be changed when LOCK is zero.

Table 30-2. PLL VCO Divide Factor

VDIV	Multiply Factor		VDIV	Multiply Factor		VDIV	Multiply Factor		VDIV	Multiply Factor
00000	16		01000	24		10000	32		11000	40
00001	17		01001	25		10001	33		11001	41
00010	18		01010	26		10010	34		11010	42
00011	19		01011	27		10011	35		11011	43
00100	20		01100	28		10100	36		11100	44
00101	21		01101	29		10101	37		11101	45
00110	22		01110	30		10110	38		11110	46
00111	23		01111	31		10111	39		11111	47

30.3.7 MCG Status Register (MCG_S)

Address: 4006_4000h base + 6h offset = 4006_4006h

Bit	7	6	5	4	3	2	1	0
Read	LOLS0	LOCK0	PLLST	IREFST	CLKST		OSCINIT0	IRCST
Write								
Reset	0	0	0	1	0	0	0	0

MCG_S field descriptions

Field	Description
7 LOLS0	Loss of Lock Status This bit is a sticky bit indicating the lock status for the PLL. LOLS is set if after acquiring lock, the PLL output frequency has fallen outside the lock exit frequency tolerance, D_{unl}. LOLIE determines whether an interrupt request is made when LOLS is set. LOLRE determines whether a reset request is made when LOLS is set. This bit is cleared by reset or by writing a logic 1 to it when set. Writing a logic 0 to this bit has no effect. 0 PLL has not lost lock since LOLS 0 was last cleared. 1 PLL has lost lock since LOLS 0 was last cleared.
6 LOCK0	Lock Status

Table continues on the next page...

MCG_S field descriptions (continued)

Field	Description
	This bit indicates whether the PLL has acquired lock. Lock detection is disabled when not operating in either PBE or PEE mode unless PLLCLKEN=1 and the MCG is not configured in BLPI or BLPE mode. While the PLL clock is locking to the desired frequency, MCGPLLCLK and MCGPLLCLK2X will be gated off until the LOCK bit gets asserted. If the lock status bit is set, changing the value of the PRDIV[2:0] bits in the C5 register or the VDIV[4:0] bits in the C6 register causes the lock status bit to clear and stay cleared until the PLL has reacquired lock. Loss of PLL reference clock will also cause the LOCK bit to clear until PLL has reacquired lock Entry into VLPS, or regular Stop with PLLSTEN=0 also causes the lock status bit to clear and stay cleared until the Stop mode is exited and the PLL has reacquired lock. Any time the PLL is enabled and the LOCK bit is cleared, the MCGPLLCLK and MCGPLLCLK2X will be gated off until the LOCK bit is asserted again. 0 PLL is currently unlocked. 1 PLL is currently locked.
5 PLLST	PLL Select Status This bit indicates the clock source selected by PLLS . The PLLST bit does not update immediately after a write to the PLLS bit due to internal synchronization between clock domains. 0 Source of PLLS clock is FLL clock. 1 Source of PLLS clock is PLL output clock.
4 IREFST	Internal Reference Status This bit indicates the current source for the FLL reference clock. The IREFST bit does not update immediately after a write to the IREFS bit due to internal synchronization between clock domains. 0 Source of FLL reference clock is the external reference clock. 1 Source of FLL reference clock is the internal reference clock.
3–2 CLKST	Clock Mode Status These bits indicate the current clock mode. The CLKST bits do not update immediately after a write to the CLKS bits due to internal synchronization between clock domains. 00 Encoding 0 — Output of the FLL is selected (reset default). 01 Encoding 1 — Internal reference clock is selected. 10 Encoding 2 — External reference clock is selected. 11 Encoding 3 — Output of the PLL is selected.
1 OSCINIT0	OSC Initialization This bit, which resets to 0, is set to 1 after the initialization cycles of the oscillator clock have completed. After being set, the bit is cleared to 0 if the OSC is subsequently disabled. See the OSC module's detailed description for more information.
0 IRCST	Internal Reference Clock Status The IRCST bit indicates the current source for the internal reference clock select clock (IRCSCLK). The IRCST bit does not update immediately after a write to the IRCS bit due to internal synchronization between clock domains. The IRCST bit will only be updated if the internal reference clock is enabled, either by the MCG being in a mode that uses the IRC or by setting the C1[IRCLKEN] bit . 0 Source of internal reference clock is the slow clock (32 kHz IRC). 1 Source of internal reference clock is the fast clock (4 MHz IRC).

30.3.8 MCG Status and Control Register (MCG_SC)

Address: 4006_4000h base + 8h offset = 4006_4008h

Bit	7	6	5	4	3	2	1	0
Read	ATME	ATMS	ATMF	FLTPRSRV	FCRDIV			LOCS0
Write			w1c					w1c
Reset	0	0	0	0	0	0	1	0

MCG_SC field descriptions

Field	Description
7 ATME	Automatic Trim Machine Enable Enables the Auto Trim Machine to start automatically trimming the selected Internal Reference Clock. **NOTE:** ATME deasserts after the Auto Trim Machine has completed trimming all trim bits of the IRCS clock selected by the ATMS bit. Writing to C1, C3, C4, and SC registers or entering Stop mode aborts the auto trim operation and clears this bit. 0 Auto Trim Machine disabled. 1 Auto Trim Machine enabled.
6 ATMS	Automatic Trim Machine Select Selects the IRCS clock for Auto Trim Test. 0 32 kHz Internal Reference Clock selected. 1 4 MHz Internal Reference Clock selected.
5 ATMF	Automatic Trim Machine Fail Flag Fail flag for the Automatic Trim Machine (ATM). This bit asserts when the Automatic Trim Machine is enabled, ATME=1, and a write to the C1, C3, C4, and SC registers is detected or the MCG enters into any Stop mode. A write to ATMF clears the flag. 0 Automatic Trim Machine completed normally. 1 Automatic Trim Machine failed.
4 FLTPRSRV	FLL Filter Preserve Enable This bit will prevent the FLL filter values from resetting allowing the FLL output frequency to remain the same during clock mode changes where the FLL/DCO output is still valid. (Note: This requires that the FLL reference frequency to remain the same as what it was prior to the new clock mode switch. Otherwise FLL filter and frequency values will change.) 0 FLL filter and FLL frequency will reset on changes to currect clock mode. 1 Fll filter and FLL frequency retain their previous values during new clock mode change.
3–1 FCRDIV	Fast Clock Internal Reference Divider Selects the amount to divide down the fast internal reference clock. The resulting frequency will be in the range 31.25 kHz to 4 MHz (Note: Changing the divider when the Fast IRC is enabled is not supported). 000 Divide Factor is 1 001 Divide Factor is 2.

Table continues on the next page...

MCG_SC field descriptions (continued)

Field	Description
	010 Divide Factor is 4. 011 Divide Factor is 8. 100 Divide Factor is 16 101 Divide Factor is 32 110 Divide Factor is 64 111 Divide Factor is 128.
0 LOCS0	OSC0 Loss of Clock Status The LOCS0 indicates when a loss of OSC0 reference clock has occurred. The LOCS0 bit only has an effect when CME0 is set. This bit is cleared by writing a logic 1 to it when set. 0 Loss of OSC0 has not occurred. 1 Loss of OSC0 has occurred.

30.3.9 MCG Auto Trim Compare Value High Register (MCG_ATCVH)

Address: 4006_4000h base + Ah offset = 4006_400Ah

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

MCG_ATCVH field descriptions

Field	Description
ATCVH	ATM Compare Value High Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.

30.3.10 MCG Auto Trim Compare Value Low Register (MCG_ATCVL)

Address: 4006_4000h base + Bh offset = 4006_400Bh

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

MCG_ATCVL field descriptions

Field	Description
ATCVL	ATM Compare Value Low Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.

30.3.11 MCG Control 8 Register (MCG_C8)

Address: 4006_4000h base + Dh offset = 4006_400Dh

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

MCG_C8 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 LOLRE	PLL Loss of Lock Reset Enable Determines if an interrupt or a reset request is made following a PLL loss of lock. 0 Interrupt request is generated on a PLL loss of lock indication. The PLL loss of lock interrupt enable bit must also be set to generate the interrupt request. 1 Generate a reset request on a PLL loss of lock indication.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

30.4 Functional description

30.4.1 MCG mode state diagram

The nine states of the MCG are shown in the following figure and are described in Table 30-3. The arrows indicate the permitted MCG mode transitions.

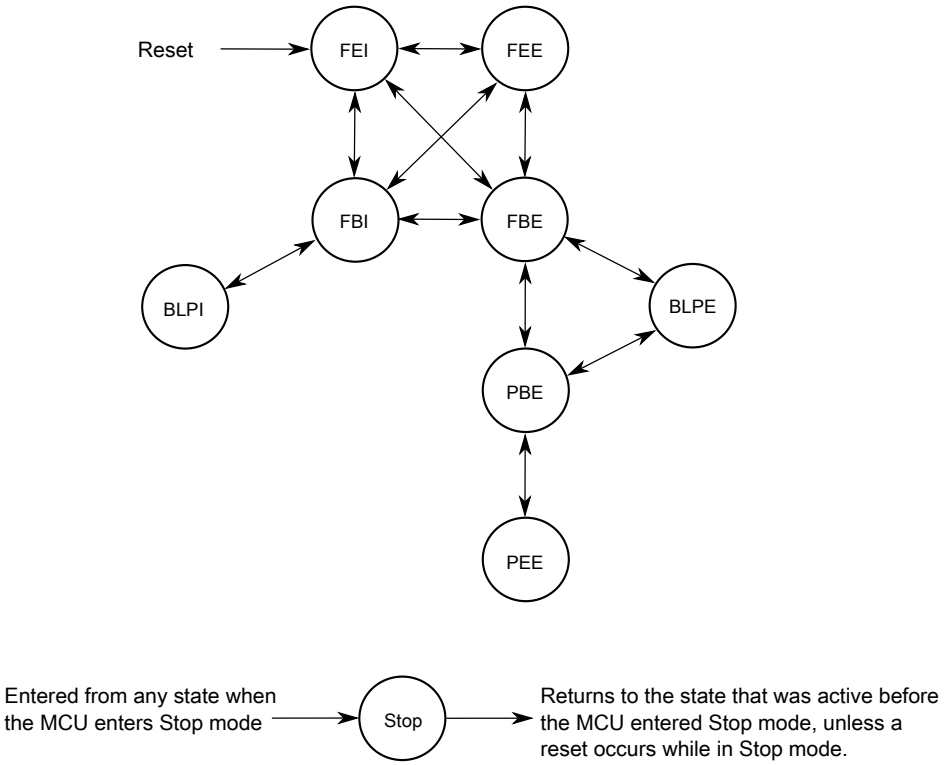


Figure 30-2. MCG mode state diagram

NOTE

- During exits from VLPS when the MCG is in PEE mode, the MCG will reset to PBE clock mode and the C1[CLKS] and S[CLKST] will automatically be set to 2’b10.
- If entering Normal Stop mode when the MCG is in PEE mode with PLLSTEN=0, the MCG will reset to PBE clock mode and C1[CLKS] and S[CLKST] will automatically be set to 2’b10.

30.4.1.1 MCG modes of operation

The MCG operates in one of the following modes.

Note

The MCG restricts transitions between modes. For the permitted transitions, see Figure 30-2.

Table 30-3. MCG modes of operation

Mode	Description
FLL Engaged Internal (FEI)	FLL engaged internal (FEI) is the default mode of operation and is entered when all the following condtions occur: • 00 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 0 is written to C6[PLLS]. In FEI mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the 32 kHz Internal Reference Clock (IRC). The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the internal reference frequency. See the C4[DMX32] bit description for more details. In FEI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
FLL Engaged External (FEE)	FLL engaged external (FEE) mode is entered when all the following conditions occur: • 00 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz • 0 is written to C6[PLLS]. In FEE mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the external reference clock. The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the external reference frequency, as specified by C1[FRDIV] and C2[RANGE]. See the C4[DMX32] bit description for more details. In FEE mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
FLL Bypassed Internal (FBI)	FLL bypassed internal (FBI) mode is entered when all the following conditions occur: • 01 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 0 is written to C6[PLLS] • 0 is written to C2[LP]. In FBI mode, the MCGOUTCLK is derived either from the slow (32 kHz IRC) or fast (4 MHz IRC) internal reference clock, as selected by the C2[IRCS] bit. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the C2[IRCS] selected internal reference clock. The FLL clock (DCOCLK) is controlled by the slow internal reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the internal reference frequency. See the C4[DMX32] bit description for more details. In FBI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
FLL Bypassed External (FBE)	FLL bypassed external (FBE) mode is entered when all the following conditions occur: • 10 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz. • 0 is written to C6[PLLS]. • 0 is written to C2[LP].

Table continues on the next page...

Table 30-3. MCG modes of operation (continued)

Mode	Description
	In FBE mode, the MCGOUTCLK is derived from the external reference clock. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the external reference clock. The FLL clock (DCOCLK) is controlled by the external reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the divided external reference frequency. See the C4[DMX32] bit description for more details. In FBE mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN] is set .
PLL Engaged External (PEE)	PLL Engaged External (PEE) mode is entered when all the following conditions occur: • 00 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • 1 is written to C6[PLLS]. In PEE mode, the MCGOUTCLK is derived from the output of PLL which is controlled by a external reference clock. The PLL clock frequency locks to a multiplication factor, as specified by its corresponding VDIV, times the selected PLL reference frequency, as specified by its corresponding PRDIV. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.
PLL Bypassed External (PBE)	PLL Bypassed External (PBE) mode is entered when all the following conditions occur: • 10 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • 1 is written to C6[PLLS]. • 0 is written to C2[LP]. In PBE mode, MCGOUTCLK is derived from the external reference clock; the PLL is operational, but its output clock is not used. This mode is useful to allow the PLL to acquire its target frequency while MCGOUTCLK is driven from the external reference clock. The PLL clock frequency locks to a multiplication factor, as specified by its [VDIV], times the PLL reference frequency, as specified by its [PRDIV]. In preparation for transition to PEE, the PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.
Bypassed Low Power Internal (BLPI) [1]	Bypassed Low Power Internal (BLPI) mode is entered when all the following conditions occur: • 01 is written to C1[CLKS]. • 1 is written to C1[IREFS]. • 0 is written to C6[PLLS]. • 1 is written to C2[LP]. In BLPI mode, MCGOUTCLK is derived from the internal reference clock. The FLL is disabled and PLL is disabled even if C5[PLLCLKEN] is set to 1.
Bypassed Low Power External (BLPE) [1]	Bypassed Low Power External (BLPE) mode is entered when all the following conditions occur: • 10 is written to C1[CLKS]. • 0 is written to C1[IREFS]. • 1 is written to C2[LP]. In BLPE mode, MCGOUTCLK is derived from the external reference clock. The FLL is disabled and PLL is disabled even if the C5[PLLCLKEN] is set to 1.

Table continues on the next page...

Table 30-3. MCG modes of operation (continued)

Mode	Description
Stop	Entered whenever the MCU enters a Stop state. The power modes are chip specific. For power mode assignments, see the Power management chapter that describes how modules are configured and MCG behavior during Stop recovery. Entering Stop mode, the FLL is disabled, and all MCG clock signals are static except in the following case: MCGPLLCLK is active in Normal Stop mode when PLLSTEN=1 MCGIRCLK is active in Normal Stop mode when all the following conditions become true: • C1[IRCLKEN] = 1 • C1[IREFSTEN] = 1 **NOTE:** • In VLPS Stop Mode, the MCGIRCLK can be programmed to stay enabled and continue running if C1[IRCLKEN] = 1, C1[IREFSTEN]=1, and Fast IRC clock is selected (C2[IRCS] = 1) **NOTE:** • When entering Low Power Stop mode (VLPS) from PEE mode, on exit the MCG clock mode is forced to PBE clock mode. C1[CLKS] and S[CLKST] will be configured to 2'b10if entering from PEE mode or to 2'b01 if entering from PEI mode, C5[PLLSTEN0] will be force to 1'b0 and S[LOCK] bit will be cleared without setting S[LOLS]. • When entering Normal Stop mode from PEE mode and if C5[PLLSTEN]=0, on exit the MCG clock mode is forced to PBE mode, the C1[CLKS] and S[CLKST] will be configured to 2'b10 and S[LOCK] bit will clear without setting S[LOLS]. If C5[PLLSTEN]=1, the S[LOCK] bit will not get cleared and on exit the MCG will continue to run in PEE mode.

1. **Caution:** If entering VLPR mode, MCG has to be configured and enter BLPE mode or BLPI mode with the Fast IRC clock selected (C2[IRCS]=1). After it enters VLPR mode, writes to any of the MCG control registers that can cause an MCG clock mode switch to a non low power clock mode must be avoided.

NOTE

For the chip-specific modes of operation, see the power management chapter of this MCU.

30.4.1.2 MCG mode switching

C1[IREFS] can be changed at any time, but the actual switch to the newly selected reference clocks is shown by S[IREFST]. When switching between engaged internal and engaged external modes, the FLL will begin locking again after the switch is completed.

C1[CLKS] can also be changed at any time, but the actual switch to the newly selected clock is shown by S[CLKST]. If the newly selected clock is not available, the previous clock will remain selected.

The C4[DRST_DRS] write bits can be changed at any time except when C2[LP] bit is 1. If C4[DRST_DRS] write bits are changed while in FLL engaged internal (FEI) or FLL engaged external (FEE) mode, the MCGOUTCLK switches to the new selected DCO

range within three clocks of the selected DCO clock. After switching to the new DCO (indicated by the updated C4[DRST_DRS] read bits), the FLL remains unlocked for several reference cycles. The FLL lock time is provided in the device data sheet as $t_{fll\_acquire}$.

30.4.2 Low-power bit usage

C2[LP] is provided to allow the FLL or PLL to be disabled and thus conserve power when these systems are not being used. C4[DRST_DRS] can not be written while C2[LP] is 1. However, in some applications, it may be desirable to enable the FLL or PLL and allow it to lock for maximum accuracy before switching to an engaged mode. Do this by writing 0 to C2[LP].

30.4.3 MCG Internal Reference Clocks

This module supports two internal reference clocks with nominal frequencies of 32 kHz (slow IRC) and 4 MHz (fast IRC). The fast IRC frequency can be divided down by programming of the FCRDIV to produce a frequency range of 32 kHz to 4 MHz.

30.4.3.1 MCG Internal Reference Clock

The MCG Internal Reference Clock (MCGIRCLK) provides a clock source for other on-chip peripherals and is enabled when C1[IRCLKEN]=1. When enabled, MCGIRCLK is driven by either the fast internal reference clock (4 MHz IRC which can be divided down by the FCRDIV factors) or the slow internal reference clock (32 kHz IRC). The IRCS clock frequency can be re-targeted by trimming the period of its IRCS selected internal reference clock. This can be done by writing a new trim value to the C3[SCTRIM]:C4[SCFTRIM] bits when the slow IRC clock is selected or by writing a new trim value to C4[FCTRIM]:C2[FCFTRIM] when the fast IRC clock is selected. The internal reference clock period is proportional to the trim value written. C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) and C4[FCTRIM]:C2[FCFTRIM] (if C2[IRCS]=1) bits affect the MCGOUTCLK frequency if the MCG is in FBI or BLPI modes. C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) bits also affect the MCGOUTCLK frequency if the MCG is in FEI mode.

Additionally, this clock can be enabled in Stop mode by setting C1[IRCLKEN] and C1[IREFSTEN], otherwise this clock is disabled in Stop mode.

30.4.4 External Reference Clock

The MCG module can support an external reference clock in all modes. See the device datasheet for external reference frequency range. When C1[IREFS] is set, the external reference clock will not be used by the FLL or PLL. In these modes, the frequency can be equal to the maximum frequency the chip-level timing specifications will support.

If any of the CME bits are asserted the slow internal reference clock is enabled along with the enabled external clock monitor. For the case when C6[CME0]=1, a loss of clock is detected if the OSC0 external reference falls below a minimum frequency (f_{loc_high} or f_{loc_low} depending on C2[RANGE0]).

NOTE

All clock monitors must be disabled before entering these low-power modes: Stop, VLPS, VLPR, VLPW, and VLLSx.

On detecting a loss-of-clock event, the MCU generates a system reset if the respective LOCRE bit is set. Otherwise the MCG sets the respective LOCS bit and the MCG generates a LOCS interrupt request. In the case where a OSC loss of clock is detected, the PLL LOCK status bit is cleared.

30.4.5 MCG PLL clock

The MCG PLL Clock (MCGPLLCLK) is available depending on the device's configuration of the MCG module. For more details, see the clock distribution chapter of this MCU. The MCGPLLCLK is prevented from coming out of the MCG until it is enabled and S[LOCK0] is set.

30.4.6 MCG Auto TRIM (ATM)

The MCG Auto Trim (ATM) is a MCG feature that when enabled, it configures the MCG hardware to automatically trim the MCG Internal Reference Clocks using an external clock as a reference. The selection between which MCG IRC clock gets tested and enabled is controlled by the ATC[ATMS] control bit (ATC[ATMS]=0 selects the 32 kHz IRC and ATC[ATMS]=1 selects the 4 MHz IRC). If 4 MHz IRC is selected for the ATM, a divide by 128 is enabled to divide down the 4 MHz IRC to a range of 31.250 kHz.

When MCG ATM is enabled by writing ATC[ATME] bit to 1, The ATM machine will start auto trimming the selected IRC clock. During the autotrim process, ATC[ATME] will remain asserted and will deassert after ATM is completed or an abort occurs. The

MCG ATM is aborted if a write to any of the following control registers is detected : C1, C3, C4, or ATC or if Stop mode is entered. If an abort occurs, ATC[ATMF] fail flag is asserted.

The ATM machine uses the bus clock as the external reference clock to perform the IRC auto-trim. Therefore, it is required that the MCG is configured in a clock mode where the reference clock used to generate the system clock is the external reference clock such as FBE clock mode. The MCG must not be configured in a clock mode where selected IRC ATM clock is used to generate the system clock. The bus clock is also required to be running with in the range of 8–16 MHz.

To perform the ATM on the selected IRC, the ATM machine uses the successive approximation technique to adjust the IRC trim bits to generate the desired IRC trimmed frequency. The ATM SARs each of the ATM IRC trim bits starting with the MSB. For each trim bit test, the ATM uses a pulse that is generated by the ATM selected IRC clock to enable a counter that counts number of ATM external clocks. At end of each trim bit, the ATM external counter value is compared to the ATCV[15:0] register value. Based on the comparison result, the ATM trim bit under test will get cleared or stay asserted. This is done until all trim bits have been tested by ATM SAR machine.

Before the ATM can be enabled, the ATM expected count needs to be derived and stored into the ATCV register. The ATCV expected count is derived based on the required target Internal Reference Clock (IRC) frequency, and the frequency of the external reference clock using the following formula:

$$\text{ATCV Expected Count Value} = 21 * (Fe / Fr)$$

- Fr = Target Internal Reference Clock (IRC) Trimmed Frequency
- Fe = External Clock Frequency

If the auto trim is being performed on the 4 MHz IRC, the calculated expected count value must be multiplied by 128 before storing it in the ATCV register. Therefore, the ATCV Expected Count Value for trimming the 4 MHz IRC is calculated using the following formula.

$$\text{Expected Count Value} = (Fe / Fr) * 21 * (128)$$

30.5 Initialization / Application information

This section describes how to initialize and configure the MCG module in an application.

The following sections include examples on how to initialize the MCG and properly switch between the various available modes.

30.5.1 MCG module initialization sequence

The MCG comes out of reset configured for FEI mode.

The internal reference will stabilize in $t_{irefsts}$ microseconds before the FLL can acquire lock. As soon as the internal reference is stable, the FLL will acquire lock in $t_{fll\_acquire}$ milliseconds.

30.5.1.1 Initializing the MCG

Because the MCG comes out of reset in FEI mode, the only MCG modes that can be directly switched to upon reset are FEE, FBE, and FBI modes (see Figure 30-2). Reaching any of the other modes requires first configuring the MCG for one of these three intermediate modes. Care must be taken to check relevant status bits in the MCG status register reflecting all configuration changes within each mode.

To change from FEI mode to FEE or FBE modes, follow this procedure:

1. Enable the external clock source by setting the appropriate bits in C2 register.
2. Write to C1 register to select the clock mode.
   - If entering FEE mode, set C1[FRDIV] appropriately, clear C1[IREFS] bit to switch to the external reference, and leave C1[CLKS] at 2'b00 so that the output of the FLL is selected as the system clock source.
   - If entering FBE, clear C1[IREFS] to switch to the external reference and change C1[CLKS] to 2'b10 so that the external reference clock is selected as the system clock source. The C1[FRDIV] bits should also be set appropriately here according to the external reference frequency to keep the FLL reference clock in the range of 31.25 kHz to 39.0625 kHz. Although the FLL is bypassed, it is still on in FBE mode.
   - The internal reference can optionally be kept running by setting C1[IRCLKEN]. This is useful if the application will switch back and forth between internal and external modes. For minimum power consumption, leave the internal reference disabled while in an external clock mode.
3. Once the proper configuration bits have been set, wait for the affected bits in the MCG status register to be changed appropriately, reflecting that the MCG has moved into the proper mode.

- If the MCG is in FEE, FBE, PEE, PBE, or BLPE mode, and C2[EREFS] was also set in step 1, wait here for S[OSCINIT0] bit to become set indicating that the external clock source has finished its initialization cycles and stabilized.
- If in FEE mode, check to make sure S[IREFST] is cleared before moving on.
- If in FBE mode, check to make sure S[IREFST] is cleared and S[CLKST] bits have changed to 2'b10 indicating the external reference clock has been appropriately selected. Although the FLL is bypassed, it is still on in FBE mode.

4. Write to the C4 register to determine the DCO output (MCGFLLCLK) frequency range.
   - By default, with C4[DMX32] cleared to 0, the FLL multiplier for the DCO output is 640. For greater flexibility, if a mid-low-range FLL multiplier of 1280 is desired instead, set C4[DRST_DRS] bits to 2'b01 for a DCO output frequency of 40 MHz. If a mid high-range FLL multiplier of 1920 is desired instead, set the C4[DRST_DRS] bits to 2'b10 for a DCO output frequency of 60 MHz. If a high-range FLL multiplier of 2560 is desired instead, set the C4[DRST_DRS] bits to 2'b11 for a DCO output frequency of 80 MHz.
   - When using a 32.768 kHz external reference, if the maximum low-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b00 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 732 will be 24 MHz.
   - When using a 32.768 kHz external reference, if the maximum mid-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b01 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 1464 will be 48 MHz.
   - When using a 32.768 kHz external reference, if the maximum mid high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b10 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2197 will be 72 MHz.
   - When using a 32.768 kHz external reference, if the maximum high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b11 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2929 will be 96 MHz.

5. Wait for the FLL lock time to guarantee FLL is running at new C4[DRST_DRS] and C4[DMX32] programmed frequency.

To change from FEI clock mode to FBI clock mode, follow this procedure:

1. Change C1[CLKS] bits in C1 register to 2'b01 so that the internal reference clock is selected as the system clock source.
2. Wait for S[CLKST] bits in the MCG status register to change to 2'b01, indicating that the internal reference clock has been appropriately selected.
3. Write to the C2 register to determine the IRCS output (IRCSCLK) frequency range.
    - By default, with C2[IRCS] cleared to 0, the IRCS selected output clock is the slow internal reference clock (32 kHz IRC). If the faster IRC is desired, set C2[IRCS] to 1 for a IRCS clock derived from the 4 MHz IRC source.

30.5.2 Using a 32.768 kHz reference

In FEE and FBE modes, if using a 32.768 kHz external reference, at the default FLL multiplication factor of 640, the DCO output (MCGFLLCLK) frequency is 20.97 MHz at low-range.

If C4[DRST_DRS] bits are set to 2'b01, the multiplication factor is doubled to 1280, and the resulting DCO output frequency is 41.94 MHz at mid-low-range. If C4[DRST_DRS] bits are set to 2'b10, the multiplication factor is set to 1920, and the resulting DCO output frequency is 62.91 MHz at mid high-range. If C4[DRST_DRS] bits are set to 2'b11, the multiplication factor is set to 2560, and the resulting DCO output frequency is 83.89 MHz at high-range.

In FBI and FEI modes, setting C4[DMX32] bit is not recommended. If the internal reference is trimmed to a frequency above 32.768 kHz, the greater FLL multiplication factor could potentially push the microcontroller system clock out of specification, potentially resulting in unpredictable behavior.

30.5.3 MCG mode switching

When switching between operational modes of the MCG, certain configuration bits must be changed in order to properly move from one mode to another.

Each time any of these bits are changed (C6[PLLS], C1[IREFS], C1[CLKS], C2[IRCS], or C2[EREFS], the corresponding bits in the MCG status register (PLLST, IREFST, CLKST, IRCST, or OSCINIT) must be checked before moving on in the application software.

Additionally, care must be taken to ensure that the reference clock divider (C1[FRDIV] and C5[PRDIV0]) is set properly for the mode being switched to. For instance, in PEE mode, if using a 16 MHz crystal, C5[PRDIV0] must be set to 3'b000 (divide-by-1) or 3'b001 (divide-by-2) to divide the external reference down to the required frequency between 8 and 16 MHz

In FBE, FEE, FBI, and FEI modes, at any time, the application can switch the FLL multiplication factor between 640, 1280, 1920, and 2560 with C4[DRST_DRS] bits. Writes to C4[DRST_DRS] bits will be ignored if C2[LP]=1.

The table below shows MCGOUTCLK frequency calculations using C1[FRDIV], C5[PRDIV0], and C6[VDIV0] settings for each clock mode.

Table 30-4. MCGOUTCLK Frequency Calculation Options

Clock Mode	$f_{MCGOUTCLK}$[1]	Note
FEI (FLL engaged internal)	$f_{int} \times F$	Typical $f_{MCGOUTCLK}$ = 21 MHz immediately after reset.
FEE (FLL engaged external)	$(f_{ext} / FLL\_R) \times F$	f_{ext} / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz
FBE (FLL bypassed external)	OSCCLK	OSCCLK / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz
FBI (FLL bypassed internal)	MCGIRCLK	Selectable between slow and fast IRC
PEE (PLL engaged external)	(OSCCLK / PLL_R) × M / 2	OSCCLK / PLL_R must be in the range of 8 – 16 MHz
PBE (PLL bypassed external)	OSCCLK	OSCCLK / PLL_R must be in the range of 8 – 16 MHz
BLPI (Bypassed low power internal)	MCGIRCLK	Selectable between slow and fast IRC
BLPE (Bypassed low power external)	OSCCLK	

1. FLL_R is the reference divider selected by the C1[FRDIV] bits, F is the FLL factor selected by C4[DRST_DRS] and C4[DMX32] bits , PLL_R is the reference divider selected by C5[PRDIV0] bits, and M is the multiplier selected by C6[VDIV0] bits.

This section will include several mode switching examples, using an 16 MHz external crystal. If using an external clock source less than 8 MHz, the MCG must not be configured for any of the PLL modes (PEE and PBE).

30.5.3.1 Example 1: Moving from FEI to PEE Mode with OSC0 as the source for the external crystal clock: External Crystal = 16 MHz, MCGOUTCLK frequency = 120 MHz

In this example, the MCG will move through the proper operational modes from FEI to PEE to achieve 120 MHz MCGOUTCLK frequency from 16 MHz external crystal reference. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, FEI must transition to FBE mode:
   a. C2 = 0x2C
      - C2[RANGE] set to 2'b10 because the frequency of 16 MHz is within the high frequency range.
      - C2[HGO] set to 1 to configure the crystal oscillator for high gain operation.
      - C2[EREFS] set to 1, because a crystal is being used.
   b. C1 = 0xA0
      - C1[CLKS] set to 2'b10 to select external reference clock as system clock source
      - C1[FRDIV] set to 3'b100, or divide-by-512 because 8 MHz / 512 = 31.25 kHz which is in the 31.25 kHz to 39.0625 kHz range required by the FLL
      - C1[IREFS] cleared to 0, selecting the external reference clock and enabling the external oscillator.
   c. Loop until S[OSCINIT0] is 1, indicating the crystal selected by C2[EREFS0] has been initialized.
   d. Loop until S[IREFST] is 0, indicating the external reference is the current source for the reference clock.
   e. Loop until S[CLKST] is 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK.
2. Then configure C5[PRDIV0] to generate correct PLL reference frequency.
   a. C5 = 0x01
      - C5[PRDIV] set to 3'b001, or divide-by-2 resulting in a pll reference frequency of 16MHz/2 = 8 MHz.
3. Then, FBE must transition either directly to PBE mode or first through BLPE mode and then to PBE mode:

a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.

b. BLPE/PBE: C6 = 0x4E

- C6[PLLS] set to 1, selects the PLL. At this time, with a C1[PRDIV] value of 2'b001, the PLL reference divider is 2 (see PLL External Reference Divide Factor table), resulting in a reference frequency of 16 MHz/ 2 = 8 MHz. In BLPE mode, changing the C6[PLLS] bit only prepares the MCG for PLL usage in PBE mode.
- C6[VDIV] set to 5'b01110, or multiply-by-30 because 8 MHz reference * 30 = 240 MHz. This is the MCGPLL0CLK2X frequency, which is the frequency of the VCO. This is divided by 2 to achieve the MCGPLL0CLK frequency of 120 MHz which is later used as the MCGOUTCLK frequency. In BLPE mode, the configuration of the VDIV bits does not matter because the PLL is disabled. Changing them only sets up the multiply value for PLL usage in PBE mode.

c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to PBE mode.

d. PBE: Loop until S[PLLST] is set, indicating that the current source for the PLLS clock is the PLL.

e. PBE: Then loop until S[LOCK0] is set, indicating that the PLL has acquired lock.

4. Lastly, PBE mode transitions into PEE mode:

a. C1 = 0x20

- C1[CLKS] set to 2'b00 to select the output of the PLL as the system clock source.

b. Loop until S[CLKST] are 2'b11, indicating that the PLL output is selected to feed MCGOUTCLK in the current clock mode.

- Now, with PRDIV of divide-by-2, and C6[VDIV] of multiply-by-30, MCGOUTCLK = [(16 MHz / 2) * 30] / 2 = 120 MHz.

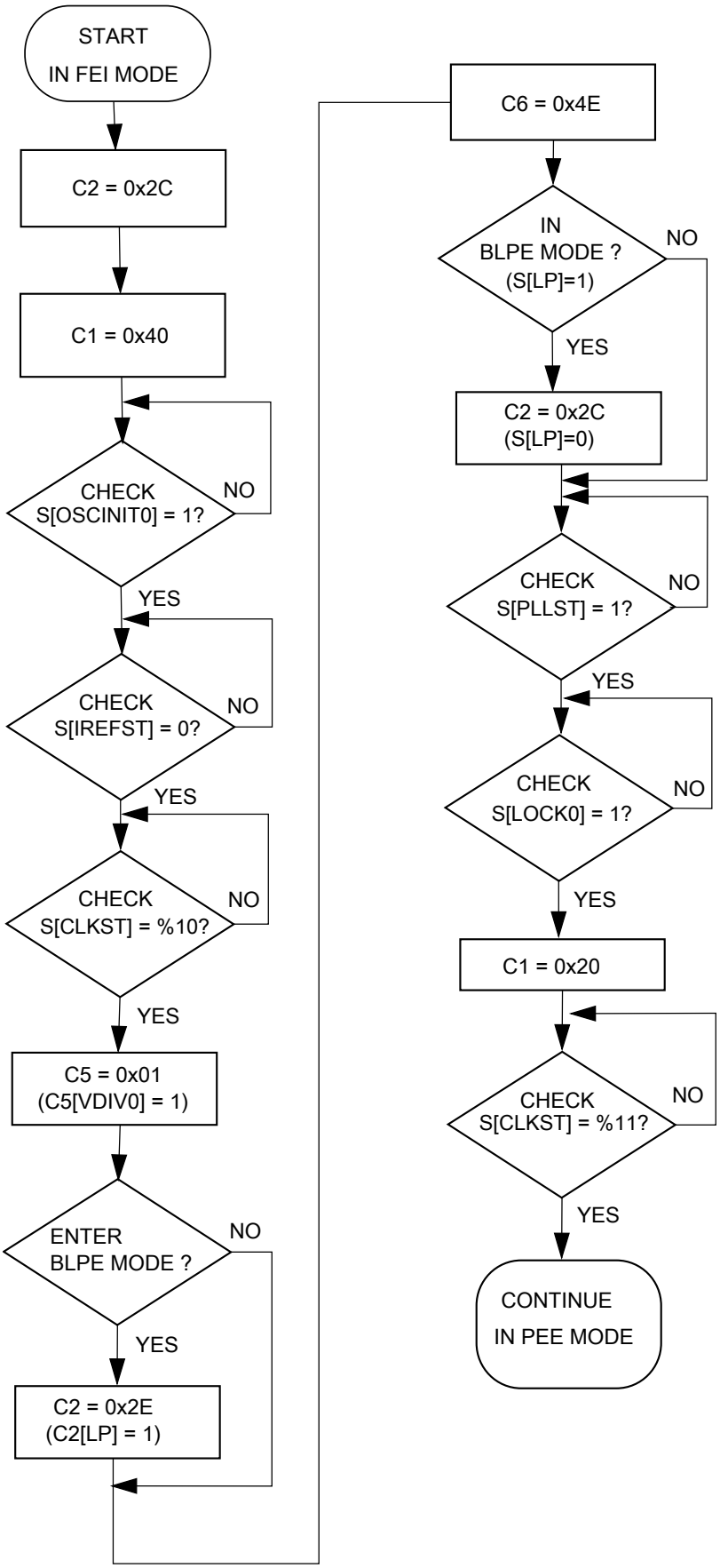


Figure 30-3. Flowchart of FEI to PEE mode transition using an 16 MHz crystal

30.5.3.2 Example 2: Moving from PEE to BLPI mode: MCGOUTCLK frequency =32 kHz

In this example, the MCG will move through the proper operational modes from PEE mode with a 16 MHz crystal configured for a 120 MHz MCGOUTCLK frequency (see previous example) to BLPI mode with a 32 kHz MCGOUTCLK frequency. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, PEE must transition to PBE mode:
   a. C1 = 0x40
      - C1[CLKS] set to 2'b10 to switch the system clock source to the external reference clock.
   b. Loop until S[CLKST] are 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK.
2. Then, PBE must transition either directly to FBE mode or first through BLPE mode and then to FBE mode:
   a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.
   b. BLPE/FBE: C6 = 0x00
      - C6[PLLS] clear to 0 to select the FLL. At this time, with C1[FRDIV] value of 3'b100, the FLL divider is set to 512, resulting in a reference frequency of 16 MHz / 512 = 31.25 kHz. If C1[FRDIV] was not previously set to 3'b100 (necessary to achieve required 31.25–39.06 kHz FLL reference frequency with an 16 MHz external source frequency), it must be changed prior to clearing C6[PLLS] bit. In BLPE mode,changing this bit only prepares the MCG for FLL usage in FBE mode. With C6[PLLS] = 0, the C6[VDIV] value does not matter.
   c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to FBE mode.
   d. FBE: Loop until S[PLLST] is cleared, indicating that the current source for the PLLS clock is the FLL.
3. Next, FBE mode transitions into FBI mode:
   a. C1 = 0x64
      - C1[CLKS] set to 2'b01 to switch the system clock to the internal reference clock.

- C1[IREFS] set to 1 to select the internal reference clock as the reference clock source.
- C1[FRDIV] remain unchanged because the reference divider does not affect the internal reference.

b. Loop until S[IREFST] is 1, indicating the internal reference clock has been selected as the reference clock source.

c. Loop until S[CLKST] are 2'b01, indicating that the internal reference clock is selected to feed MCGOUTCLK.

4. Lastly, FBI transitions into BLPI mode.

a. C2 = 0x22

- C2[LP] is 1
- C2[RANGE], C2[HGO], C2[EREFS], C1[IRCLKEN], and C1[IREFSTEN] bits are ignored when the C1[IREFS] bit is set. They can remain set, or be cleared at this point.

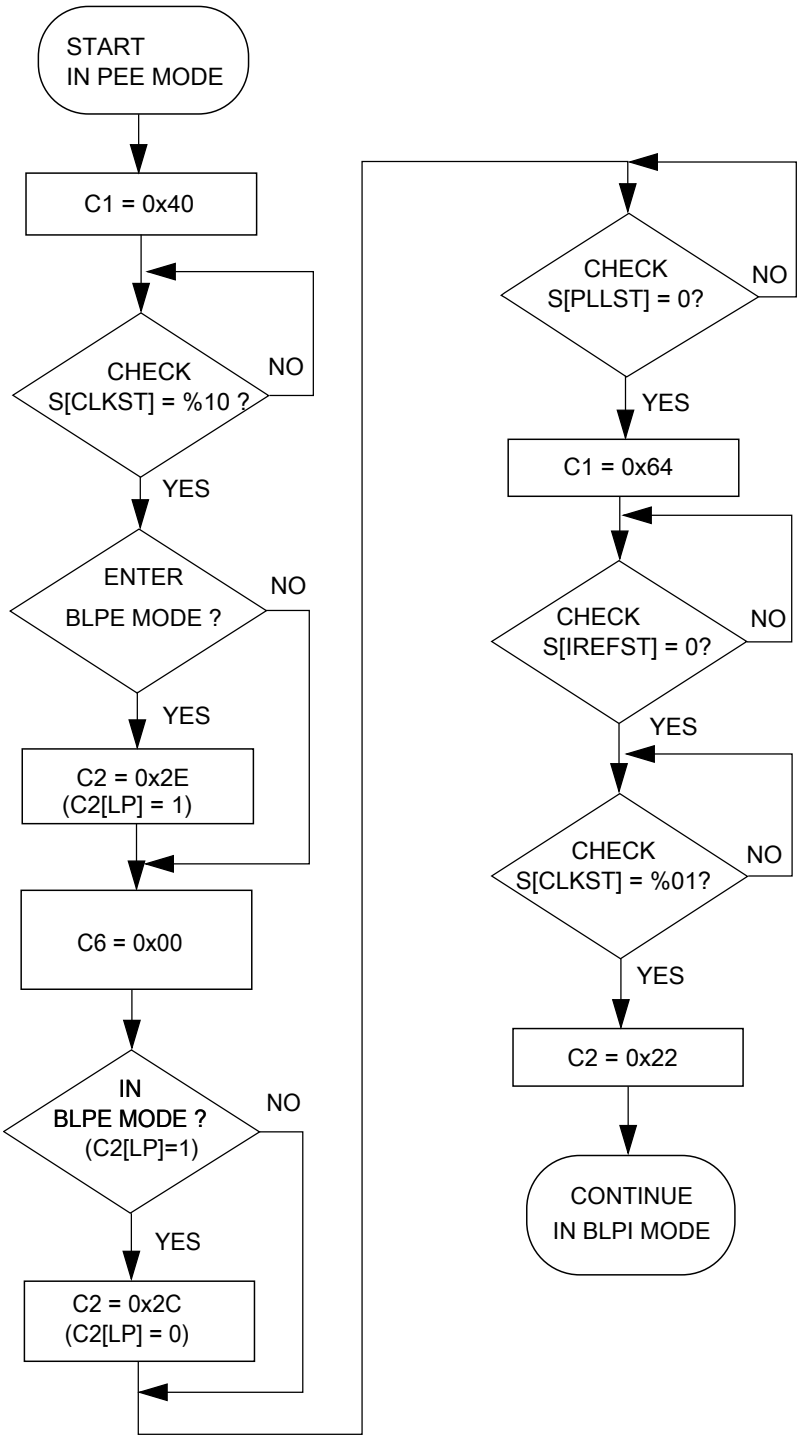


Figure 30-4. Flowchart of PEE to BLPI mode transition using an 16 MHz crystal

Chapter 31
Flash Memory Controller (FMC)

31.1 Introduction

The Flash Memory Controller (FMC) is a memory acceleration unit that provides:

- an interface between the device and the nonvolatile memory.
- buffers that can accelerate flash memory transfers.

31.1.1 Overview

The Flash Memory Controller manages the interface between the device and the flash memory. The FMC receives status information detailing the configuration of the memory and uses this information to ensure a proper interface. The following table shows the supported read/write operations.

Flash memory type	Read	Write
Program flash memory	8-bit, 16-bit, and 32-bit reads	—[1]

1. A write operation to program flash memory results in a bus error.

In addition, for bank 0, the FMC provides three separate mechanisms for accelerating the interface between the device and the flash memory. A 128-bit speculation buffer can prefetch the next 128-bit flash memory location, and both a 4-way, 2-set cache and a single-entry 128-bit buffer can store previously accessed flash memory data for quick access times.

31.1.2 Features

The FMC's features include:

- Interface between the device and the flash memory:
  - 8-bit, 16-bit, and 32-bit read operations to program flash memory.

- For bank 0: Read accesses to consecutive 32-bit spaces in memory return the second, third, and fourth read data with no wait states. The memory returns 128 bits via the 32-bit bus access.
- Crossbar master access protection for setting no access, read-only access, write-only access, or read/write access for each crossbar master.

- For bank 0: Acceleration of data transfer from program flash memory to the device:
  - 128-bit prefetch speculation buffer with controls for instruction/data access per master
  - 4-way, 2-set, 128-bit line size cache for a total of eight 128-bit entries with controls for replacement algorithm and lock per way
  - Single-entry buffer
  - Invalidation control for the speculation buffer and the single-entry buffer

31.2 Modes of operation

The FMC only operates when a bus master accesses the flash memory.

For any device power mode where the flash memory cannot be accessed, the FMC is disabled.

31.3 External signal description

The FMC has no external signals.

31.4 Memory map and register descriptions

The programming model consists of the FMC control registers and the program visible cache (data and tag/valid entries).

NOTE

Program the registers only while the flash controller is idle (for example, execute from RAM). Changing configuration settings while a flash access is in progress can lead to non-deterministic behavior.

Table 31-1. FMC register access

Registers	Read access		Write access	
	Mode	Length	Mode	Length
Control registers: PFAPR, PFB0CR, PFB1CR	Supervisor (privileged) mode or user mode	32 bits	Supervisor (privileged) mode only	32 bits
Cache registers	Supervisor (privileged) mode or user mode	32 bits	Supervisor (privileged) mode only	32 bits

NOTE

Accesses to unimplemented registers within the FMC's 4 KB address space return a bus error.

The cache entries, both data and tag/valid, can be read at any time.

NOTE

System software is required to maintain memory coherence when any segment of the flash cache is programmed. For example, all buffer data associated with the reprogrammed flash should be invalidated. Accordingly, cache program visible writes must occur after a programming or erase event is completed and before the new memory image is accessed.

The cache is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. The following table elaborates on the tag/valid and data entries.

Table 31-2. Program visible cache registers

Cache storage	Based at offset	Contents of 32-bit read	Nomenclature	Nomenclature example
Tag	100h	13'h0, tag[18:6], 5'h0, valid	In TAGVDWxSy, x denotes the way and y denotes the set.	TAGVDW2S0 is the 13-bit tag and 1-bit valid for cache entry way 2, set 0.
Data	200h	One of the four longwords in a 128-bit cache entry	In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost).	For data entry way 1, set 3, DATAW1S3UM represents bits [127:96], DATAW1S3MU represents bits [95:64], DATAW1S3ML represents bits [63:32], and DATAW1S3LM represents bits [31:0].

FMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F000	Flash Access Protection Register (FMC_PFAPR)	32	R/W	00F8_003Fh	31.4.1/582
4001_F004	Flash Bank 0 Control Register (FMC_PFB0CR)	32	R/W	3004_001Fh	31.4.2/584
4001_F008	Reserved (FMC_Reserved)	32	R/W	3000_0000h	31.4.3/586
4001_F100	Cache Tag Storage (FMC_TAGVDW0S0)	32	R/W	0000_0000h	31.4.4/587
4001_F104	Cache Tag Storage (FMC_TAGVDW0S1)	32	R/W	0000_0000h	31.4.4/587
4001_F108	Cache Tag Storage (FMC_TAGVDW1S0)	32	R/W	0000_0000h	31.4.5/588
4001_F10C	Cache Tag Storage (FMC_TAGVDW1S1)	32	R/W	0000_0000h	31.4.5/588
4001_F110	Cache Tag Storage (FMC_TAGVDW2S0)	32	R/W	0000_0000h	31.4.6/589
4001_F114	Cache Tag Storage (FMC_TAGVDW2S1)	32	R/W	0000_0000h	31.4.6/589
4001_F118	Cache Tag Storage (FMC_TAGVDW3S0)	32	R/W	0000_0000h	31.4.7/590
4001_F11C	Cache Tag Storage (FMC_TAGVDW3S1)	32	R/W	0000_0000h	31.4.7/590
4001_F200	Cache Data Storage (uppermost word) (FMC_DATAW0S0UM)	32	R/W	0000_0000h	31.4.8/591
4001_F204	Cache Data Storage (mid-upper word) (FMC_DATAW0S0MU)	32	R/W	0000_0000h	31.4.9/591
4001_F208	Cache Data Storage (mid-lower word) (FMC_DATAW0S0ML)	32	R/W	0000_0000h	31.4.10/ 592
4001_F20C	Cache Data Storage (lowermost word) (FMC_DATAW0S0LM)	32	R/W	0000_0000h	31.4.11/ 592
4001_F210	Cache Data Storage (uppermost word) (FMC_DATAW0S1UM)	32	R/W	0000_0000h	31.4.8/591
4001_F214	Cache Data Storage (mid-upper word) (FMC_DATAW0S1MU)	32	R/W	0000_0000h	31.4.9/591
4001_F218	Cache Data Storage (mid-lower word) (FMC_DATAW0S1ML)	32	R/W	0000_0000h	31.4.10/ 592
4001_F21C	Cache Data Storage (lowermost word) (FMC_DATAW0S1LM)	32	R/W	0000_0000h	31.4.11/ 592
4001_F240	Cache Data Storage (uppermost word) (FMC_DATAW1S0UM)	32	R/W	0000_0000h	31.4.12/ 593
4001_F244	Cache Data Storage (mid-upper word) (FMC_DATAW1S0MU)	32	R/W	0000_0000h	31.4.13/ 593
4001_F248	Cache Data Storage (mid-lower word) (FMC_DATAW1S0ML)	32	R/W	0000_0000h	31.4.14/ 594
4001_F24C	Cache Data Storage (lowermost word) (FMC_DATAW1S0LM)	32	R/W	0000_0000h	31.4.15/ 594
4001_F250	Cache Data Storage (uppermost word) (FMC_DATAW1S1UM)	32	R/W	0000_0000h	31.4.12/ 593
4001_F254	Cache Data Storage (mid-upper word) (FMC_DATAW1S1MU)	32	R/W	0000_0000h	31.4.13/ 593
4001_F258	Cache Data Storage (mid-lower word) (FMC_DATAW1S1ML)	32	R/W	0000_0000h	31.4.14/ 594
4001_F25C	Cache Data Storage (lowermost word) (FMC_DATAW1S1LM)	32	R/W	0000_0000h	31.4.15/ 594

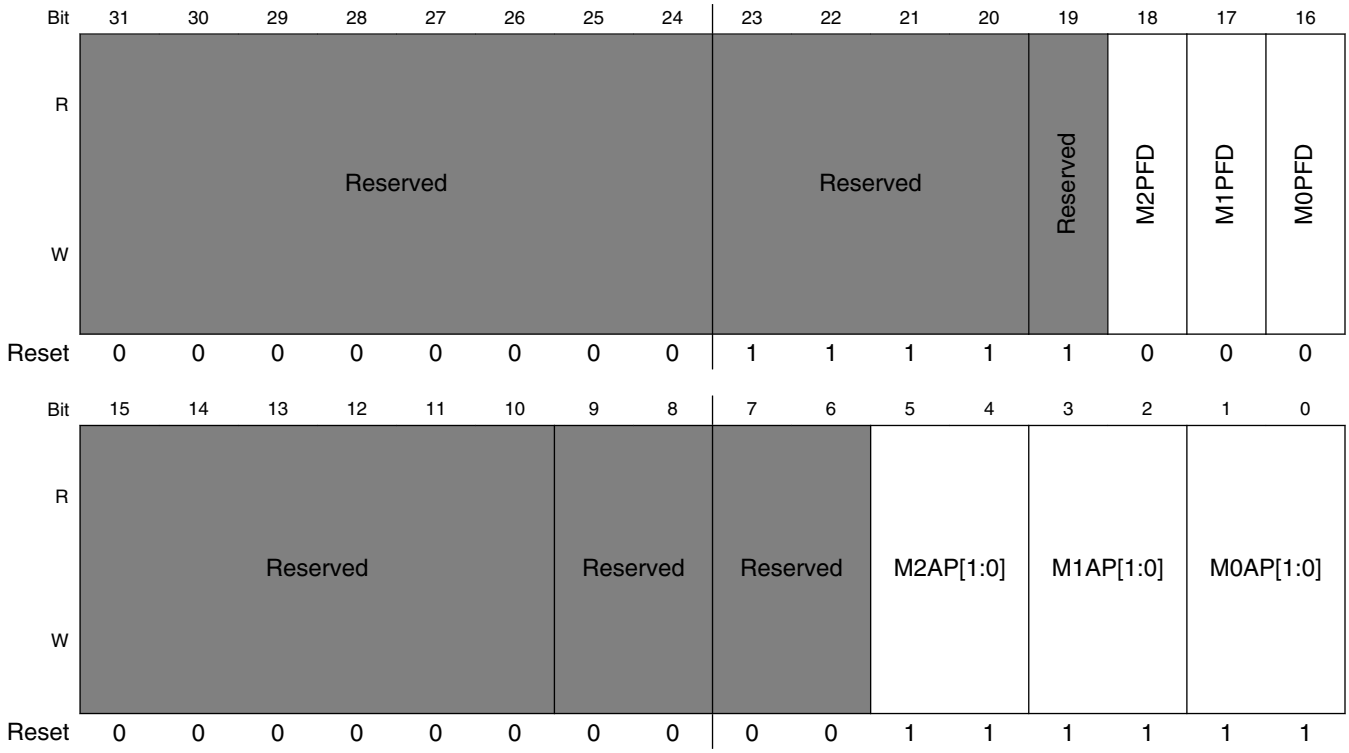
Table continues on the next page...

FMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F280	Cache Data Storage (uppermost word) (FMC_DATAW2S0UM)	32	R/W	0000_0000h	31.4.16/ 595
4001_F284	Cache Data Storage (mid-upper word) (FMC_DATAW2S0MU)	32	R/W	0000_0000h	31.4.17/ 595
4001_F288	Cache Data Storage (mid-lower word) (FMC_DATAW2S0ML)	32	R/W	0000_0000h	31.4.18/ 596
4001_F28C	Cache Data Storage (lowermost word) (FMC_DATAW2S0LM)	32	R/W	0000_0000h	31.4.19/ 596
4001_F290	Cache Data Storage (uppermost word) (FMC_DATAW2S1UM)	32	R/W	0000_0000h	31.4.16/ 595
4001_F294	Cache Data Storage (mid-upper word) (FMC_DATAW2S1MU)	32	R/W	0000_0000h	31.4.17/ 595
4001_F298	Cache Data Storage (mid-lower word) (FMC_DATAW2S1ML)	32	R/W	0000_0000h	31.4.18/ 596
4001_F29C	Cache Data Storage (lowermost word) (FMC_DATAW2S1LM)	32	R/W	0000_0000h	31.4.19/ 596
4001_F2C0	Cache Data Storage (uppermost word) (FMC_DATAW3S0UM)	32	R/W	0000_0000h	31.4.20/ 597
4001_F2C4	Cache Data Storage (mid-upper word) (FMC_DATAW3S0MU)	32	R/W	0000_0000h	31.4.21/ 597
4001_F2C8	Cache Data Storage (mid-lower word) (FMC_DATAW3S0ML)	32	R/W	0000_0000h	31.4.22/ 598
4001_F2CC	Cache Data Storage (lowermost word) (FMC_DATAW3S0LM)	32	R/W	0000_0000h	31.4.23/ 598
4001_F2D0	Cache Data Storage (uppermost word) (FMC_DATAW3S1UM)	32	R/W	0000_0000h	31.4.20/ 597
4001_F2D4	Cache Data Storage (mid-upper word) (FMC_DATAW3S1MU)	32	R/W	0000_0000h	31.4.21/ 597
4001_F2D8	Cache Data Storage (mid-lower word) (FMC_DATAW3S1ML)	32	R/W	0000_0000h	31.4.22/ 598
4001_F2DC	Cache Data Storage (lowermost word) (FMC_DATAW3S1LM)	32	R/W	0000_0000h	31.4.23/ 598

31.4.1 Flash Access Protection Register (FMC_PFAPR)

Address: 4001_F000h base + 0h offset = 4001_F000h



FMC_PFAPR field descriptions

Field	Description
31–24 Reserved	This field is reserved.
23–20 Reserved	This field is reserved. This read-only bitfield is reserved. Do not write to this bitfield or indeterminate results will occur.
19 Reserved	This field is reserved.
18 M2PFD	Master 2 Prefetch Disable These bits control whether prefetching is enabled, based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
17 M1PFD	Master 1 Prefetch Disable These bits control whether prefetching is enabled, based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.

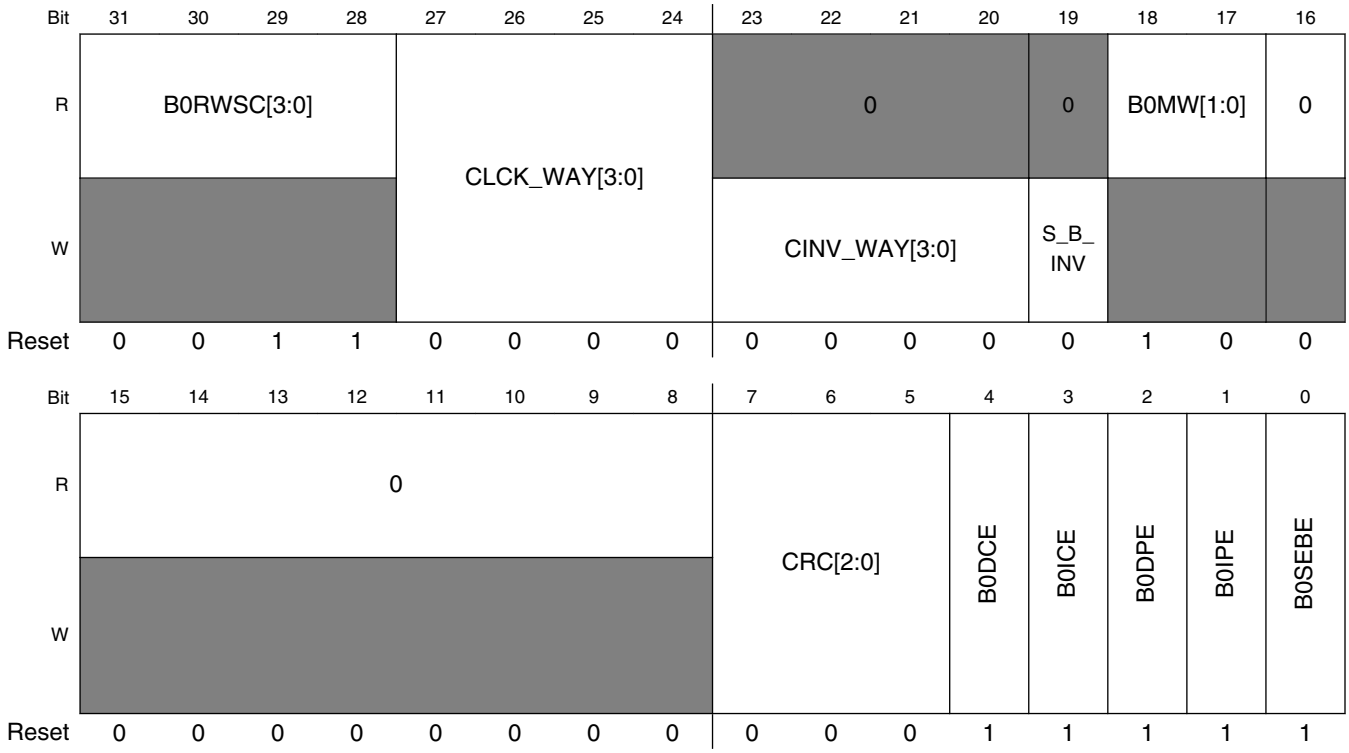
Table continues on the next page...

FMC_PFAPR field descriptions (continued)

Field	Description
16 M0PFD	Master 0 Prefetch Disable These bits control whether prefetching is enabled, based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
15–10 Reserved	This field is reserved. This read-only bitfield is reserved and is reset to zero. Do not write to this bitfield or indeterminate results will occur.
9–8 Reserved	This field is reserved.
7–6 Reserved	This field is reserved.
5–4 M2AP[1:0]	Master 2 Access Protection This field controls whether read and write access to the flash is allowed, based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
3–2 M1AP[1:0]	Master 1 Access Protection This field controls whether read and write access to the flash is allowed, based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
M0AP[1:0]	Master 0 Access Protection This field controls whether read and write access to the flash is allowed, based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master

31.4.2 Flash Bank 0 Control Register (FMC_PFB0CR)

Address: 4001_F000h base + 4h offset = 4001_F004h



FMC_PFB0CR field descriptions

Field	Description
31–28 B0RWSC[3:0]	Bank 0 Read Wait State Control This read-only field defines the number of wait states required to access the bank 0 flash memory. The relationship between the read access time of the flash array (expressed in system clock cycles) and RWSC is defined as: Access time of flash array [system clocks] = RWSC + 1 The FMC automatically calculates this value based on the ratio of the system clock speed to the flash clock speed. For example, when this ratio is 4:1, the field's value is 3h.
27–24 CLCK_WAY[3:0]	Cache Lock Way x These bits determine if the given cache way is locked such that its contents will not be displaced by future misses. The bit setting definitions are for each bit in the field. 0 Cache way is unlocked and may be displaced 1 Cache way is locked and its contents are not displaced
23–20 CINV_WAY[3:0]	Cache Invalidate Way x

Table continues on the next page...

FMC_PFB0CR field descriptions (continued)

Field	Description
	These bits determine if the given cache way is to be invalidated (cleared). When a bit within this field is written, the corresponding cache way is immediately invalidated: the way's tag, data, and valid contents are cleared. This field always reads as zero. Cache invalidation takes precedence over locking. The cache is invalidated by system reset. System software is required to maintain memory coherency when any segment of the flash memory is programmed or erased. Accordingly, cache invalidations must occur after a programming or erase event is completed and before the new memory image is accessed. The bit setting definitions are for each bit in the field. 0 No cache way invalidation for the corresponding cache 1 Invalidate cache way for the corresponding cache: clear the tag, data, and vld bits of ways selected
19 S_B_INV	Invalidate Prefetch Speculation Buffer This bit determines if the FMC's prefetch speculation buffer and the single entry page buffer are to be invalidated (cleared). When this bit is written, the speculation buffer and single entry buffer are immediately cleared. This bit always reads as zero. 0 Speculation buffer and single entry buffer are not affected. 1 Invalidate (clear) speculation buffer and single entry buffer.
18–17 B0MW[1:0]	Bank 0 Memory Width This read-only field defines the width of the bank 0 memory. 00 32 bits 01 64 bits 10 128 bits 11 Reserved
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–5 CRC[2:0]	Cache Replacement Control This 3-bit field defines the replacement algorithm for accesses that are cached. 000 LRU replacement algorithm per set across all four ways 001 Reserved 010 Independent LRU with ways [0-1] for ifetches, [2-3] for data 011 Independent LRU with ways [0-2] for ifetches, [3] for data 1xx Reserved
4 B0DCE	Bank 0 Data Cache Enable This bit controls whether data references are loaded into the cache. 0 Do not cache data references. 1 Cache data references.
3 B0ICE	Bank 0 Instruction Cache Enable This bit controls whether instruction fetches are loaded into the cache.

Table continues on the next page...

FMC_PFB0CR field descriptions (continued)

Field	Description
	0 Do not cache instruction fetches. 1 Cache instruction fetches.
2 B0DPE	Bank 0 Data Prefetch Enable This bit controls whether prefetches (or speculative accesses) are initiated in response to data references. 0 Do not prefetch in response to data references. 1 Enable prefetches in response to data references.
1 B0IPE	Bank 0 Instruction Prefetch Enable This bit controls whether prefetches (or speculative accesses) are initiated in response to instruction fetches. 0 Do not prefetch in response to instruction fetches. 1 Enable prefetches in response to instruction fetches.
0 B0SEBE	Bank 0 Single Entry Buffer Enable This bit controls whether the single entry page buffer is enabled in response to flash read accesses. A high-to-low transition of this enable forces the page buffer to be invalidated. 0 Single entry buffer is disabled. 1 Single entry buffer is enabled.

31.4.3 Reserved (FMC_Reserved)

This register address is reserved.

Address: 4001_F000h base + 8h offset = 4001_F008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				0									Reserved		0
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

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

FMC_Reserved field descriptions

Field	Description
31–28 Reserved	This field is reserved.
27–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

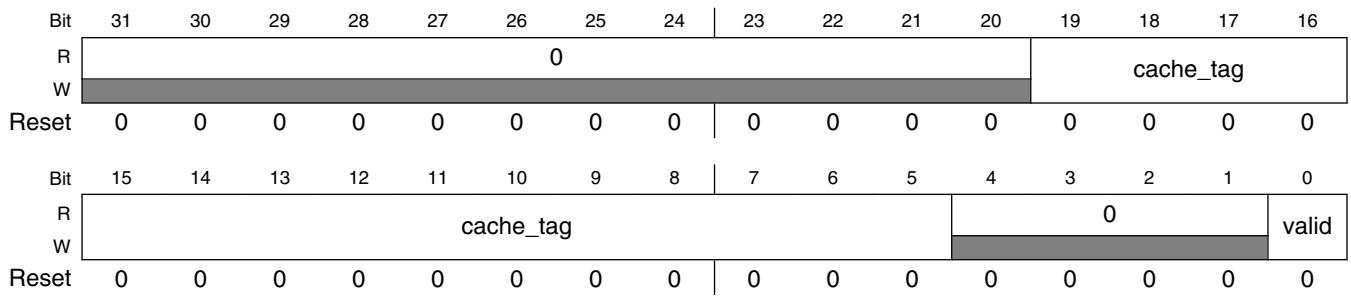
FMC_Reserved field descriptions (continued)

Field	Description
18–17 Reserved	This field is reserved.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

31.4.4 Cache Tag Storage (FMC_TAGVDW0S*n*)

The 128-entry cache is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all sets (n=0-1) in way 0.

Address: 4001_F000h base + 100h offset + (4d × i), where i=0d to 1d



FMC_TAGVDW0S*n* field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–5 cache_tag	the tag for cache entry
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 valid	1-bit valid for cache entry

31.4.5 Cache Tag Storage (FMC_TAGVDW1S*n*)

The 128-entry cache is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all sets (n=0-1) in way 1.

Address: 4001_F000h base + 108h offset + (4d × i), where i=0d to 1d

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

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

FMC_TAGVDW1S*n* field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–5 cache_tag	the tag for cache entry
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 valid	1-bit valid for cache entry

31.4.6 Cache Tag Storage (FMC_TAGVDW2S*n*)

The 128-entry cache is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all sets (n=0-1) in way 2.

Address: 4001_F000h base + 110h offset + (4d × i), where i=0d to 1d

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

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

FMC_TAGVDW2S*n* field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–5 cache_tag	the tag for cache entry
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 valid	1-bit valid for cache entry

31.4.7 Cache Tag Storage (FMC_TAGVDW3S*n*)

The 128-entry cache is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all sets (n=0-1) in way 3.

Address: 4001_F000h base + 118h offset + (4d × i), where i=0d to 1d

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

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

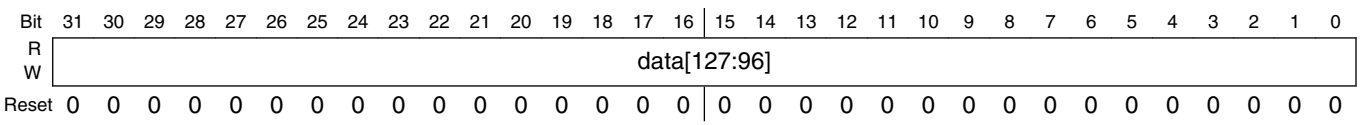
FMC_TAGVDW3S*n* field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–5 cache_tag	the tag for cache entry
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 valid	1-bit valid for cache entry

31.4.8 Cache Data Storage (uppermost word) (FMC_DATAW0S*n*UM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 200h offset + (16d × i), where i=0d to 1d



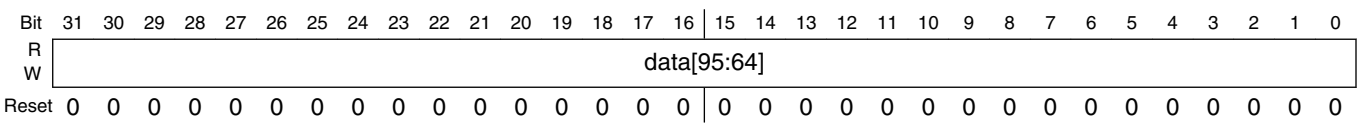
FMC_DATAW0S*n*UM field descriptions

Field	Description
data[127:96]	Bits [127:96] of data entry

31.4.9 Cache Data Storage (mid-upper word) (FMC_DATAW0S*n*MU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 204h offset + (16d × i), where i=0d to 1d



FMC_DATAW0S*n*MU field descriptions

Field	Description
data[95:64]	Bits [95:64] of data entry

31.4.10 Cache Data Storage (mid-lower word) (FMC_DATAW0S*n*ML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 208h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[63:32]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW0S*n*ML field descriptions

Field	Description
data[63:32]	Bits [63:32] of data entry

31.4.11 Cache Data Storage (lowermost word) (FMC_DATAW0S*n*LM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 20Ch offset + (16d × i), where i=0d to 1d

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

FMC_DATAW0S*n*LM field descriptions

Field	Description
data[31:0]	Bits [31:0] of data entry

31.4.12 Cache Data Storage (uppermost word) (FMC_DATAW1S*n*UM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 240h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[127:96]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW1S*n*UM field descriptions

Field	Description
data[127:96]	Bits [127:96] of data entry

31.4.13 Cache Data Storage (mid-upper word) (FMC_DATAW1S*n*MU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 244h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[95:64]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW1S*n*MU field descriptions

Field	Description
data[95:64]	Bits [95:64] of data entry

31.4.14 Cache Data Storage (mid-lower word) (FMC_DATAW1S*n*ML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 248h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[63:32]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW1S*n*ML field descriptions

Field	Description
data[63:32]	Bits [63:32] of data entry

31.4.15 Cache Data Storage (lowermost word) (FMC_DATAW1S*n*LM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 24Ch offset + (16d × i), where i=0d to 1d

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

FMC_DATAW1S*n*LM field descriptions

Field	Description
data[31:0]	Bits [31:0] of data entry

31.4.16 Cache Data Storage (uppermost word) (FMC_DATAW2S*n*UM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 280h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[127:96]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW2S*n*UM field descriptions

Field	Description
data[127:96]	Bits [127:96] of data entry

31.4.17 Cache Data Storage (mid-upper word) (FMC_DATAW2S*n*MU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 284h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[95:64]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW2S*n*MU field descriptions

Field	Description
data[95:64]	Bits [95:64] of data entry

31.4.18 Cache Data Storage (mid-lower word) (FMC_DATAW2S*n*ML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 288h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[63:32]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW2S*n*ML field descriptions

Field	Description
data[63:32]	Bits [63:32] of data entry

31.4.19 Cache Data Storage (lowermost word) (FMC_DATAW2S*n*LM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 28Ch offset + (16d × i), where i=0d to 1d

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

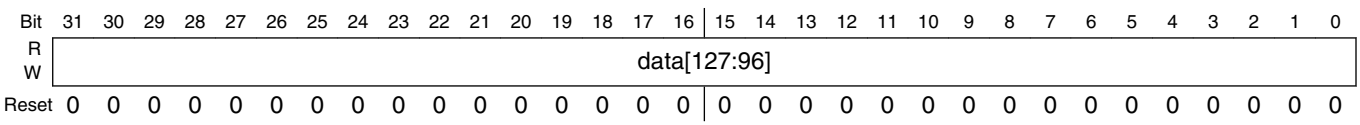
FMC_DATAW2S*n*LM field descriptions

Field	Description
data[31:0]	Bits [31:0] of data entry

31.4.20 Cache Data Storage (uppermost word) (FMC_DATAW3S*n*UM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 2C0h offset + (16d × i), where i=0d to 1d



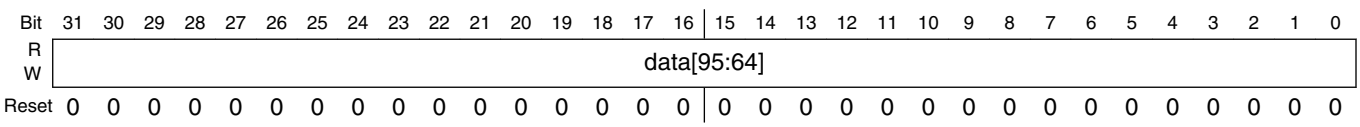
FMC_DATAW3S*n*UM field descriptions

Field	Description
data[127:96]	Bits [127:96] of data entry

31.4.21 Cache Data Storage (mid-upper word) (FMC_DATAW3S*n*MU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1 . In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 2C4h offset + (16d × i), where i=0d to 1d



FMC_DATAW3S*n*MU field descriptions

Field	Description
data[95:64]	Bits [95:64] of data entry

31.4.22 Cache Data Storage (mid-lower word) (FMC_DATAW3S*n*ML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1 . In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 2 sets (n=0-1) in way 0.

Address: 4001_F000h base + 2C8h offset + (16d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	data[63:32]																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FMC_DATAW3S*n*ML field descriptions

Field	Description
data[63:32]	Bits [63:32] of data entry

31.4.23 Cache Data Storage (lowermost word) (FMC_DATAW3S*n*LM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 2 sets. The ways are numbered 0-3 and the sets are numbered 0-1 . In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 24 sets (n=0-1) in way 0.

Address: 4001_F000h base + 2CCh offset + (16d × i), where i=0d to 1d

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

FMC_DATAW3S*n*LM field descriptions

Field	Description
data[31:0]	Bits [31:0] of data entry

31.5 Functional description

The FMC is a flash acceleration unit with flexible buffers for user configuration. Besides managing the interface between the device and the flash memory, the FMC can be used to restrict access from crossbar switch masters and customize the cache and buffers to provide single-cycle system-clock data-access times. Whenever a hit occurs for the prefetch speculation buffer, the cache, or the single-entry buffer, the requested data is transferred within a single system clock.

31.5.1 Default configuration

Upon system reset, the FMC is configured to provide a significant level of buffering for transfers from the flash memory:

- Crossbar masters 0, 1, 2 have read access to bank 0.
- For bank 0:
  - Prefetch support for data and instructions is enabled for crossbar masters 0, 1, 2.
  - The cache is configured for least recently used (LRU) replacement for all four ways.
  - The cache is configured for data or instruction replacement.
  - The single-entry buffer is enabled.

31.5.2 Configuration options

Though the default configuration provides a high degree of flash acceleration, advanced users may desire to customize the FMC buffer configurations to maximize throughput for their use cases. When reconfiguring the FMC for custom use cases, do not program the FMC's control registers while the flash memory is being accessed. Instead, change the control registers with a routine executing from RAM in supervisor mode.

The FMC's cache and buffering controls within PFB0CR allow the tuning of resources to suit particular applications' needs. The cache and buffer are each controlled individually. The register controls enable buffering and prefetching per access type (instruction fetch or data reference). The cache also supports 3 types of LRU replacement algorithms:

- LRU per set across all 4 ways,
- LRU with ways [0-1] for instruction fetches and ways [2-3] for data fetches, and
- LRU with ways [0-2] for instruction fetches and way [3] for data fetches.

As an application example: if both instruction fetches and data references are accessing flash memory, then control is available to send instruction fetches, data references, or both to the cache or the single-entry buffer. Likewise, speculation can be enabled or disabled for either type of access. If both instruction fetches and data references are cached, then the cache's way resources may be divided in several ways between the instruction fetches and data references.

31.5.3 Speculative reads

The FMC has a single buffer that reads ahead to the next word in the flash memory if there is an idle cycle. Speculative prefetching is programmable for each bank for instruction and/or data accesses using the B0DPE and B0IPE fields of PFB0CR. Because many code accesses are sequential, using the speculative prefetch buffer improves performance in most cases.

When speculative reads are enabled, the FMC immediately requests the next sequential address after a read completes. By requesting the next word immediately, speculative reads can help to reduce or even eliminate wait states when accessing sequential code and/or data.

For example, consider the following scenario:

- Assume a system with a 4:1 core-to-flash clock ratio and with speculative reads enabled.
- The core requests eight sequential longwords in back-to-back requests, meaning there are no core cycle delays except for stalls waiting for flash memory data to be returned.
- None of the data is already stored in the cache or speculation buffer.

In this scenario, the sequence of events for accessing the eight longwords is as follows:

1. The first longword read requires 4 to 7 core clocks. See Wait states for more information.
2. Due to the 128-bit data bus of the flash memory, the second longword read takes only 1 core clock because the data is already available inside the FMC. For the same reason, the third and fourth longword reads each take only 1 core clock.
3. Accessing the fifth longword requires 1 core clock cycle. The flash memory read itself takes 4 clocks, but the access starts immediately after the first read. As a result, 3 clocks for this access overlap with the second, third, and fourth longword reads from the core.
4. Reading the sixth, seventh, and eighth longwords takes only 1 clock each because the data is already available inside the FMC.

31.6 Initialization and application information

The FMC does not require user initialization. Flash acceleration features are enabled by default.

The FMC has no visibility into flash memory erase and program cycles because the Flash Memory module manages them directly. As a result, if an application is executing flash memory commands, the FMC's cache might need to be disabled and/or flushed to prevent the possibility of returning stale data. Use the PFB0CR[CINV_WAY] field to invalidate the cache in this manner.

Chapter 32
Flash Memory Module (FTFA)

32.1 Chip-specific FTFA information

32.1.1 HSRUN mode

In this device, the HSRUN mode allows the CPU to run at high speed in high power mode. In this mode, the CPU power consumption is higher than normal RUN mode. The user has no access to the flash command set when the device is in HSRUN mode.

32.2 Introduction

The flash memory module includes the following accessible memory regions:

- Program flash memory for vector space and code store

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The flash memory module includes a memory controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased with security disabled. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that each flash block or sector be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

32.2.1 Features

The flash memory module includes the following features.

NOTE

See the device's Chip Configuration details for the exact amount of flash memory available on your device.

32.2.1.1 Program Flash Memory Features

- Sector size of 4 KB
- Program flash protection scheme prevents accidental program or erase of stored data
- Automated, built-in, program and erase algorithms with verify

32.2.1.2 Other Flash Memory Module Features

- Internal high-voltage supply generator for flash memory program and erase operations
- Optional interrupt generation upon flash command completion
- Supports MCU security mechanisms which prevent unauthorized access to the flash memory contents

32.2.2 Block Diagram

The block diagram of the flash memory module is shown in the following figure.

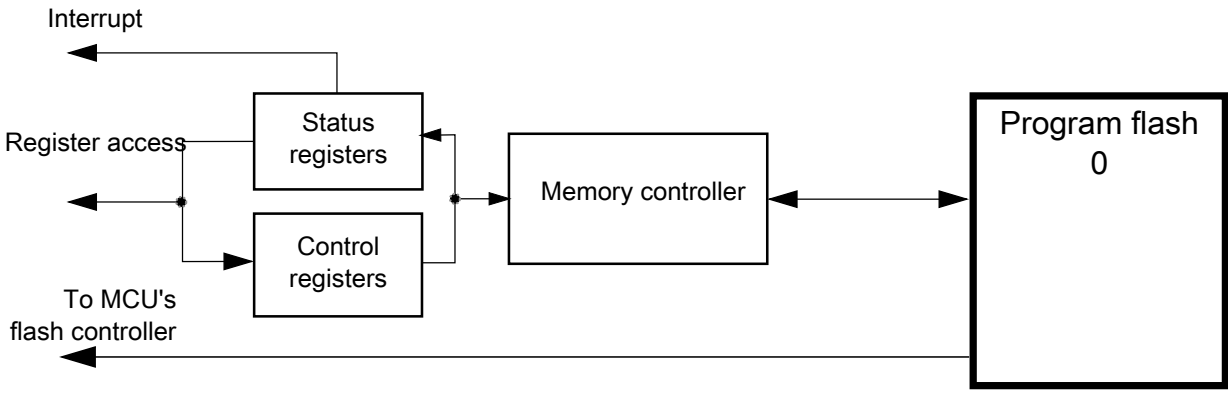


Figure 32-1. Flash Block Diagram

32.2.3 Glossary

Command write sequence — A series of MCU writes to the flash FCCOB register group that initiates and controls the execution of flash algorithms that are built into the flash memory module.

Double-Phrase — 128 bits of data with an aligned double-phrase having byte-address[3:0] = 0000.

Endurance — The number of times that a flash memory location can be erased and reprogrammed.

FCCOB (Flash Common Command Object) — A group of flash registers that are used to pass command, address, data, and any associated parameters to the memory controller in the flash memory module.

Flash block — A macro within the flash memory module which provides the nonvolatile memory storage.

Flash Memory Module — All flash blocks plus a flash management unit providing high-level control and an interface to MCU buses.

HSRUN — An MCU power mode enabling high-speed access to the memory resources in the flash module. The user has no access to the flash command set when the MCU is in HSRUN mode.

IFR — Nonvolatile information register found in each flash block, separate from the main memory array.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

NVM — Nonvolatile memory. A memory technology that maintains stored data during power-off. The flash array is an NVM using NOR-type flash memory technology.

NVM Normal Mode — An NVM mode that provides basic user access to flash memory module resources. The CPU or other bus masters initiate flash program and erase operations (or other flash commands) using writes to the FCCOB register group in the flash memory module.

NVM Special Mode — An NVM mode enabling external, off-chip access to the memory resources in the flash memory module. A reduced flash command set is available when the MCU is secured. See the Chip Configuration details for information on when this mode is used.

Phrase — 64 bits of data with an aligned phrase having byte-address[2:0] = 000.

Program flash — The program flash memory provides nonvolatile storage for vectors and code store.

Program flash Sector — The smallest portion of the program flash memory (consecutive addresses) that can be erased.

Retention — The length of time that data can be kept in the NVM without experiencing errors upon readout. Since erased (1) states are subject to degradation just like programmed (0) states, the data retention limit may be reached from the last erase operation (not from the programming time).

RWW— Read-While-Write. The ability to simultaneously read from one memory resource while commanded operations are active in another memory resource.

Secure — An MCU state conveyed to the flash memory module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

32.3 External Signal Description

The flash memory module contains no signals that connect off-chip.

32.4 Memory Map and Registers

This section describes the memory map and registers for the flash memory module.

Data read from unimplemented memory space in the flash memory module is undefined. Writes to unimplemented or reserved memory space (registers) in the flash memory module are ignored.

32.4.1 Flash Configuration Field Description

The program flash memory contains a 16-byte flash configuration field that stores default protection settings (loaded on reset) and security information that allows the MCU to restrict access to the flash memory module.

Flash Configuration Field Offset Address	Size (Bytes)	Field Description
0x0_0400–0x0_0407	8	Backdoor Comparison Key. Refer to Verify Backdoor Access Key Command and Unsecuring the Chip Using Backdoor Key Access.
0x0_0408–0x0_040B	4	Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3).
0x0_040F	1	Reserved
0x0_040E	1	Reserved
0x0_040D	1	Flash nonvolatile option byte. Refer to the description of the Flash Option Register (FOPT).
0x0_040C	1	Flash security byte. Refer to the description of the Flash Security Register (FSEC).

32.4.2 Program Flash IFR Map

The program flash IFR is nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see the Read Once, Program Once, and Read Resource commands in Read Once Command, Program Once Command and Read Resource Command).

The contents of the program flash IFR are summarized in the table found here and further described in the subsequent paragraphs.

The program flash IFR is located within the program flash 0 memory block .

Address Range	Size (Bytes)	Field Description
0x00 – 0xBF	192	Reserved
0xC0 – 0xFF	64	Program Once Field

32.4.2.1 Program Once Field

The Program Once Field in the program flash IFR provides 64 bytes of user data storage separate from the program flash main array. The user can program the Program Once Field one time only as there is no program flash IFR erase mechanism available to the user. The Program Once Field can be read any number of times. This section of the program flash IFR is accessed in 4-byte records using the Read Once and Program Once commands (see Read Once Command and Program Once Command).

32.4.3 Register Descriptions

The flash memory module contains a set of memory-mapped control and status registers.

NOTE

While a command is running (FSTAT[CCIF]=0), register writes are not accepted to any register except FCNFG and FSTAT. The no-write rule is relaxed during the start-up reset sequence, prior to the initial rise of CCIF. During this initialization period the user may write any register. All register writes are also disabled (except for registers FCNFG and FSTAT) whenever an erase suspend request is active (FCNFG[ERSSUSP]=1).

FTFA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0000	Flash Status Register (FTFA_FSTAT)	8	R/W	00h	32.4.3.1/ 609
4002_0001	Flash Configuration Register (FTFA_FCNFG)	8	R/W	00h	32.4.3.2/ 611
4002_0002	Flash Security Register (FTFA_FSEC)	8	R	Undefined	32.4.3.3/ 612
4002_0003	Flash Option Register (FTFA_FOPT)	8	R	Undefined	32.4.3.4/ 613
4002_0004	Flash Common Command Object Registers (FTFA_FCCOB3)	8	R/W	00h	32.4.3.5/ 614
4002_0005	Flash Common Command Object Registers (FTFA_FCCOB2)	8	R/W	00h	32.4.3.5/ 614
4002_0006	Flash Common Command Object Registers (FTFA_FCCOB1)	8	R/W	00h	32.4.3.5/ 614
4002_0007	Flash Common Command Object Registers (FTFA_FCCOB0)	8	R/W	00h	32.4.3.5/ 614

Table continues on the next page...

FTFA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0008	Flash Common Command Object Registers (FTFA_FCCOB7)	8	R/W	00h	32.4.3.5/ 614
4002_0009	Flash Common Command Object Registers (FTFA_FCCOB6)	8	R/W	00h	32.4.3.5/ 614
4002_000A	Flash Common Command Object Registers (FTFA_FCCOB5)	8	R/W	00h	32.4.3.5/ 614
4002_000B	Flash Common Command Object Registers (FTFA_FCCOB4)	8	R/W	00h	32.4.3.5/ 614
4002_000C	Flash Common Command Object Registers (FTFA_FCCOBB)	8	R/W	00h	32.4.3.5/ 614
4002_000D	Flash Common Command Object Registers (FTFA_FCCOBA)	8	R/W	00h	32.4.3.5/ 614
4002_000E	Flash Common Command Object Registers (FTFA_FCCOB9)	8	R/W	00h	32.4.3.5/ 614
4002_000F	Flash Common Command Object Registers (FTFA_FCCOB8)	8	R/W	00h	32.4.3.5/ 614
4002_0010	Program Flash Protection Registers (FTFA_FPROT3)	8	R/W	Undefined	32.4.3.6/ 615
4002_0011	Program Flash Protection Registers (FTFA_FPROT2)	8	R/W	Undefined	32.4.3.6/ 615
4002_0012	Program Flash Protection Registers (FTFA_FPROT1)	8	R/W	Undefined	32.4.3.6/ 615
4002_0013	Program Flash Protection Registers (FTFA_FPROT0)	8	R/W	Undefined	32.4.3.6/ 615

32.4.3.1 Flash Status Register (FTFA_FSTAT)

The FSTAT register reports the operational status of the flash memory module.

The CCIF, RDCOLERR, ACCERR, and FPVIOL bits are readable and writable. The MGSTAT0 bit is read only. The unassigned bits read 0 and are not writable.

NOTE

When set, the Access Error (ACCERR) and Flash Protection Violation (FPVIOL) bits in this register prevent the launch of any more commands until the flag is cleared (by writing a one to it).

Address: 4002_0000h base + 0h offset = 4002_0000h

Bit	7	6	5	4	3	2	1	0
Read	CCIF	RDCOLERR	ACCERR	FPVIOL	0			MGSTAT0
Write	w1c	w1c	w1c	w1c				
Reset	0	0	0	0	0	0	0	0

FTFA_FSTAT field descriptions

Field	Description
7 CCIF	Command Complete Interrupt Flag Indicates that a flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command, and CCIF stays low until command completion or command violation. CCIF is reset to 0 but is set to 1 by the memory controller at the end of the reset initialization sequence. Depending on how quickly the read occurs after reset release, the user may or may not see the 0 hardware reset value. 0 Flash command in progress 1 Flash command has completed
6 RDCOLERR	Flash Read Collision Error Flag Indicates that the MCU attempted a read from a flash memory resource that was being manipulated by a flash command (CCIF=0). Any simultaneous access is detected as a collision error by the block arbitration logic. The read data in this case cannot be guaranteed. The RDCOLERR bit is cleared by writing a 1 to it. Writing a 0 to RDCOLERR has no effect. 0 No collision error detected 1 Collision error detected
5 ACCERR	Flash Access Error Flag Indicates an illegal access has occurred to a flash memory resource caused by a violation of the command write sequence or issuing an illegal flash command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR while CCIF is set. Writing a 0 to the ACCERR bit has no effect. 0 No access error detected 1 Access error detected
4 FPVIOL	Flash Protection Violation Flag Indicates an attempt was made to program or erase an address in a protected area of program flash memory during a command write sequence . While FPVIOL is set, the CCIF flag cannot be cleared to launch a command. The FPVIOL bit is cleared by writing a 1 to FPVIOL while CCIF is set. Writing a 0 to the FPVIOL bit has no effect. 0 No protection violation detected 1 Protection violation detected
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 MGSTAT0	Memory Controller Command Completion Status Flag The MGSTAT0 status flag is set if an error is detected during execution of a flash command or during the flash reset sequence. As a status flag, this field cannot (and need not) be cleared by the user like the other error flags in this register.

Table continues on the next page...

FTFA_FSTAT field descriptions (continued)

Field	Description
	The value of the MGSTAT0 bit for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the previous result is discarded and any previous error is cleared.

32.4.3.2 Flash Configuration Register (FTFA_FCNFG)

This register provides information on the current functional state of the flash memory module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. The unassigned bits read as noted and are not writable.

Address: 4002_0000h base + 1h offset = 4002_0001h

Bit	7	6	5	4	3	2	1	0
Read	CCIE	RDCOLLIE	ERSAREQ	ERSSUSP	0	0	0	0
Write								
Reset	0	0	0	0	0	0	0	0

FTFA_FCNFG field descriptions

Field	Description
7 CCIE	Command Complete Interrupt Enable Controls interrupt generation when a flash command completes. 0 Command complete interrupt disabled 1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6 RDCOLLIE	Read Collision Error Interrupt Enable Controls interrupt generation when a flash memory read collision error occurs. 0 Read collision error interrupt disabled 1 Read collision error interrupt enabled. An interrupt request is generated whenever a flash memory read collision error is detected (see the description of FSTAT[RDCOLERR]).
5 ERSAREQ	Erase All Request Issues a request to the memory controller to execute the Erase All Blocks command and release security. ERSAREQ is not directly writable but is under indirect user control. Refer to the device's Chip Configuration details on how to request this command. ERSAREQ sets when an erase all request is triggered external to the flash memory module and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the flash memory module when the operation completes. 0 No request or request complete 1 Request to: 1. run the Erase All Blocks command,

Table continues on the next page...

FTFA_FCNFG field descriptions (continued)

Field	Description
	2. verify the erased state, 3. program the security byte in the Flash Configuration Field to the unsecure state, and 4. release MCU security by setting the FSEC[SEC] field to the unsecure state.
4 ERSSUSP	Erase Suspend Allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing. 0 No suspend requested 1 Suspend the current Erase Flash Sector command execution.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

32.4.3.3 Flash Security Register (FTFA_FSEC)

This read-only register holds all bits associated with the security of the MCU and flash memory module.

During the reset sequence, the register is loaded with the contents of the flash security byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002_0000h base + 2h offset = 4002_0002h

Bit	7	6	5	4	3	2	1	0
Read	KEYEN		MEEN		FSLACC		SEC	
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FSEC field descriptions

Field	Description
7–6 KEYEN	Backdoor Key Security Enable Enables or disables backdoor key access to the flash memory module. 00 Backdoor key access disabled 01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access)

Table continues on the next page...

FTFA_FSEC field descriptions (continued)

Field	Description
	10 Backdoor key access enabled 11 Backdoor key access disabled
5–4 MEEN	Mass Erase Enable Enables and disables mass erase capability of the flash memory module at all times in all NVM modes. 00 Mass erase is enabled 01 Mass erase is enabled 10 Mass erase is disabled 11 Mass erase is enabled
3–2 FSLACC	Factory Security Level Access Code Enables or disables access to the flash memory contents during returned part failure analysis at NXP. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed by NXP factory test must begin with a full erase to unsecure the part. When access is granted (SEC is unsecure, or SEC is secure and FSLACC is granted), NXP factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when SEC is set to secure. When SEC is set to unsecure, the FSLACC setting does not matter. 00 NXP factory access granted 01 NXP factory access denied 10 NXP factory access denied 11 NXP factory access granted
SEC	Flash Security Defines the security state of the MCU. In the secure state, the MCU limits access to flash memory module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the flash memory module is unsecured using backdoor key access, SEC is forced to 10b. 00 MCU security status is secure. 01 MCU security status is secure. 10 MCU security status is unsecure. (The standard shipping condition of the flash memory module is unsecure.) 11 MCU security status is secure.

32.4.3.4 Flash Option Register (FTFA_FOPT)

The flash option register allows the MCU to customize its operations by examining the state of these read-only bits, which are loaded from NVM at reset. The function of the bits is defined in the device's Chip Configuration details.

All bits in the register are read-only .

During the reset sequence, the register is loaded from the flash nonvolatile option byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002_0000h base + 3h offset = 4002_0003h

Bit	7	6	5	4	3	2	1	0
Read	OPT							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FOPT field descriptions

Field	Description
OPT	Nonvolatile Option These bits are loaded from flash to this register at reset. Refer to the device's Chip Configuration details for the definition and use of these bits.

32.4.3.5 Flash Common Command Object Registers (FTFA_FCCOB*n*)

The FCCOB register group provides 12 bytes for command codes and parameters. The individual bytes within the set append a 0-B hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOBB.

Address: 4002_0000h base + 4h offset + (1d × i), where i=0d to 11d

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

FTFA_FCCOB*n* field descriptions

Field	Description
CCOBn	The FCCOB register provides a command code and relevant parameters to the memory controller. The individual registers that compose the FCCOB data set can be written in any order, but you must provide all needed values, which vary from command to command. First, set up all required FCCOB fields and then initiate the command's execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit, which locks all FCCOB parameter fields and they cannot be changed by the user until the command completes (CCIF returns to 1). No command buffering or queueing is provided; the next command can be loaded only after the current command completes. Some commands return information to the FCCOB registers. Any values returned to FCCOB are available for reading after the FSTAT[CCIF] flag returns to 1 by the memory controller. The following table shows a generic flash command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific flash command, typically an address and/or data values.

FTFA_FCCOB*n* field descriptions (continued)

Field	Description
	NOTE: The command parameter table is written in terms of FCCOB Number (which is equivalent to the byte number). This number is a reference to the FCCOB register name and is not the register address.

FCCOB Number	Typical Command Parameter Contents [7:0]
0	FCMD (a code that defines the flash command)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0]
4	Data Byte 0
5	Data Byte 1
6	Data Byte 2
7	Data Byte 3
8	Data Byte 4
9	Data Byte 5
A	Data Byte 6
B	Data Byte 7

FCCOB Endianness and Multi-Byte Access :

The FCCOB register group uses a big endian addressing convention. For all command parameter fields larger than 1 byte, the most significant data resides in the lowest FCCOB register number. The FCCOB register group may be read and written as individual bytes, aligned words (2 bytes) or aligned longwords (4 bytes).

32.4.3.6 Program Flash Protection Registers (FTFA_FPROT*n*)

The FPROT registers define which program flash regions are protected from program and erase operations. Protected flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any flash command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow up to 32 protectable regions. Each bit protects a 1/32 region of the program flash memory except for memory configurations with less than 128 KB of program flash where each assigned bit protects 4 KB. For configurations with 96 KB of program flash memory or less, FPROT0 is not used. For configurations with 64 KB of program flash memory or less, FPROT1 is not used. For configurations with 32 KB of program flash memory, FPROT2 is not used. The bitfields are defined in each register as follows:

Program flash protection register	Program flash protection bits
FPROT0	PROT[31:24]
FPROT1	PROT[23:16]
FPROT2	PROT[15:8]
FPROT3	PROT[7:0]

During the reset sequence, the FPROT registers are loaded with the contents of the program flash protection bytes in the Flash Configuration Field as indicated in the following table.

Program flash protection register	Flash Configuration Field offset address
FPROT0	0x000B
FPROT1	0x000A
FPROT2	0x0009
FPROT3	0x0008

To change the program flash protection that is loaded during the reset sequence, unprotect the sector of program flash memory that contains the Flash Configuration Field. Then, reprogram the program flash protection byte.

Address: 4002_0000h base + 10h offset + (1d × i), where i=0d to 3d

Bit	7	6	5	4	3	2	1	0
Read Write	PROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFA_FPROT*n* field descriptions

Field	Description
PROT	Program Flash Region Protect Each program flash region can be protected from program and erase operations by setting the associated PROT bit. **In NVM Normal mode:** The protection can only be increased, meaning that currently unprotected memory can be protected, but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **In NVM Special mode:** All bits of FPROT are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected. **Restriction:** The user must never write to any FPROT register while a command is running (CCIF=0). Trying to alter data in any protected area in the program flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A full block erase of a program flash block is not possible if it contains any protected region.

FTFA_FPROT*n* field descriptions (continued)

Field	Description
	Each bit in the 32-bit protection register represents 1/32 of the total program flash except for memory configurations with less than 128 KB of program flash where each assigned bit protects 4 KB. 0 Program flash region is protected. 1 Program flash region is not protected

32.5 Functional Description

The information found here describes functional details of the flash memory module.

32.5.1 Flash Protection

Individual regions within the flash memory can be protected from program and erase operations.

Protection is controlled by the following registers:

- FPROT*n* —
  - For 2^n program flash sizes, four registers typically protect 32 regions of the program flash memory as shown in the following figure

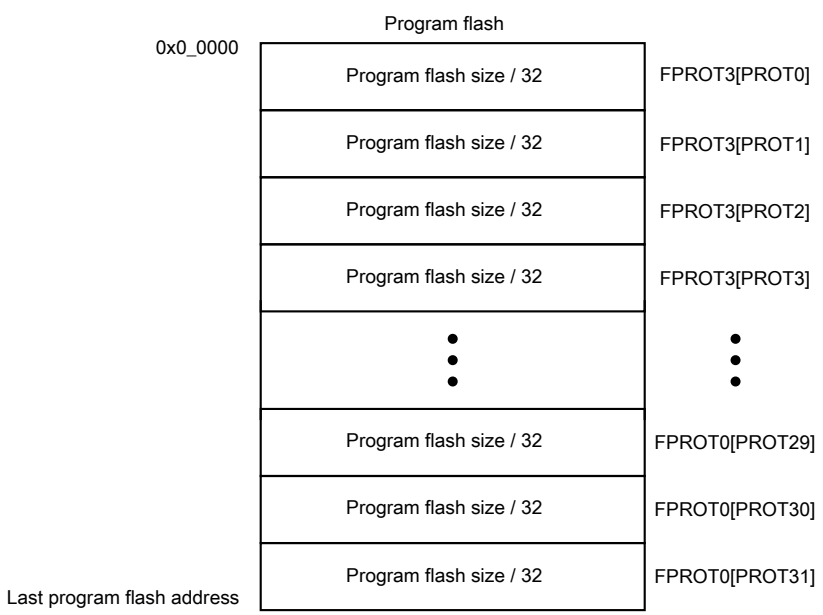


Figure 32-2. Program flash protection

NOTE

Flash protection features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Not all features described in the application note are available on this device.

32.5.2 Interrupts

The flash memory module can generate interrupt requests to the MCU upon the occurrence of various flash events.

These interrupt events and their associated status and control bits are shown in the following table.

Table 32-1. Flash Interrupt Sources

Flash Event	Readable Status Bit	Interrupt Enable Bit
Flash Command Complete	FSTAT[CCIF]	FCNFG[CCIE]
Flash Read Collision Error	FSTAT[RDCOLERR]	FCNFG[RDCOLLIE]

Note

Vector addresses and their relative interrupt priority are determined at the MCU level.

Some devices also generate a bus error response as a result of a Read Collision Error event. See the chip configuration information to determine if a bus error response is also supported.

32.5.3 Flash Operation in Low-Power Modes

32.5.3.1 Wait Mode

When the MCU enters wait mode, the flash memory module is not affected. The flash memory module can recover the MCU from wait via the command complete interrupt (see Interrupts).

32.5.3.2 Stop Mode

When the MCU requests stop mode, if a flash command is active (CCIF = 0) the command execution completes before the MCU is allowed to enter stop mode.

CAUTION

The MCU should never enter stop mode while any flash command is running (CCIF = 0).

NOTE

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the flash memory module does not accept flash commands.

32.5.4 Functional Modes of Operation

The flash memory module has two operating modes: NVM Normal and NVM Special.

The operating mode affects the command set availability (see Table 32-2). Refer to the Chip Configuration details of this device for how to activate each mode.

32.5.5 Flash Reads and Ignored Writes

The flash memory module requires only the flash address to execute a flash memory read.

The MCU must not read from the flash memory while commands are running (as evidenced by CCIF=0) on that block. Read data cannot be guaranteed from a flash block while any command is processing within that block. The block arbitration logic detects any simultaneous access and reports this as a read collision error (see the FSTAT[RDCOLERR] bit).

32.5.6 Read While Write (RWW)

The following simultaneous accesses are not allowed:

- Reading from program flash memory space while a flash command is active (CCIF=0).

32.5.7 Flash Program and Erase

All flash functions except read require the user to setup and launch a flash command through a series of peripheral bus writes.

The user cannot initiate any further flash commands until notified that the current command has completed. The flash command structure and operation are detailed in Flash Command Operations.

32.5.8 Flash Command Operations

Flash command operations are typically used to modify flash memory contents.

The next sections describe:

- The command write sequence used to set flash command parameters and launch execution
- A description of all flash commands available

32.5.8.1 Command Write Sequence

Flash commands are specified using a command write sequence illustrated in Figure 32-3. The flash memory module performs various checks on the command (FCCOB) content and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be zero and the CCIF flag must read 1 to verify that any previous command has completed. If CCIF is zero, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

Attempts to launch a flash command in VLP mode will be ignored. Attempts to launch a flash command in HSRUN mode will be trapped with the ACCERR flag being set.

32.5.8.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired flash command. The individual registers that make up the FCCOB data set can be written in any order.

32.5.8.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing FSTAT[CCIF] by writing a '1' to it. FSTAT[CCIF] remains 0 until the flash command completes.

The FSTAT register contains a blocking mechanism that prevents a new command from launching (can't clear FSTAT[CCIF]) if the previous command resulted in an access error (FSTAT[ACCERR]=1) or a protection violation (FSTAT[FPVIOL]=1). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write clears the error flags, the second write clears CCIF.

32.5.8.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The flash memory module reads the command code and performs a series of parameter checks and protection checks, if applicable, which are unique to each command.

   If the parameter check fails, the FSTAT[ACCERR] (access error) flag is set. FSTAT[ACCERR] reports invalid instruction codes and out-of bounds addresses. Usually, access errors suggest that the command was not set-up with valid parameters in the FCCOB register group.

   Program and erase commands also check the address to determine if the operation is requested to execute on protected areas. If the protection check fails, FSTAT[FPVIOL] (protection error) flag is set.

   Command processing never proceeds to execution when the parameter or protection step fails. Instead, command processing is terminated after setting FSTAT[CCIF].

2. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to erase verify, may occur during the execution phase. Run-time errors are reported in FSTAT[MGSTAT0]. A command may have access errors, protection errors, and run-time errors, but the run-time errors are not seen until all access and protection errors have been corrected.
3. Command execution results, if applicable, are reported back to the user via the FCCOB and FSTAT registers.
4. The flash memory module sets FSTAT[CCIF] signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

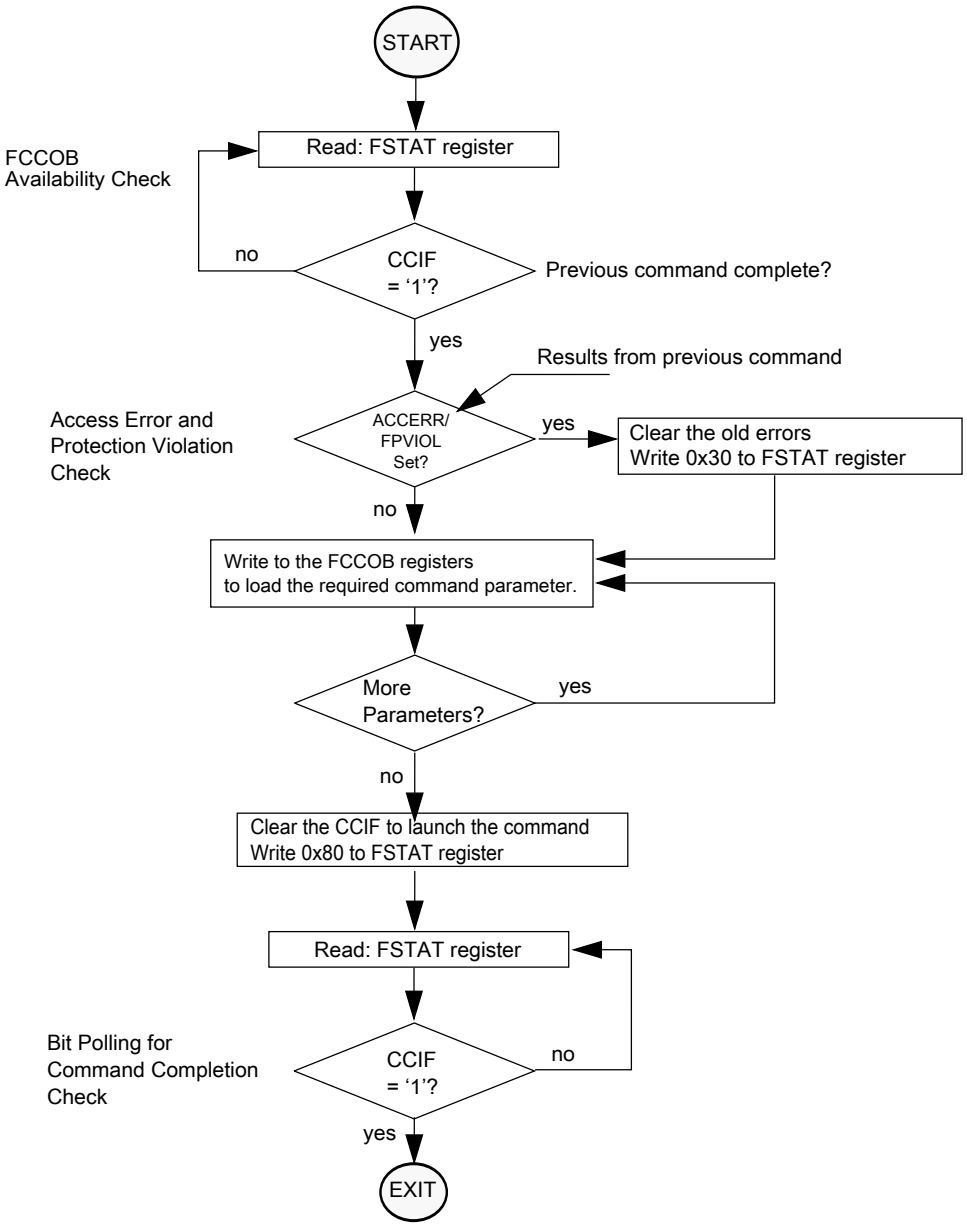


Figure 32-3. Generic flash command write sequence flowchart

32.5.8.2 Flash Commands

The following table summarizes the function of all flash commands.

FCMD	Command	Program flash	Function
0x01	Read 1s Section	×	Verify that a given number of program flash locations from a starting address are erased.

Table continues on the next page...

FCMD	Command	Program flash	Function
0x02	Program Check	×	Tests previously-programmed locations at margin read levels.
0x03	Read Resource	IFR, ID	Read 4 bytes from program flash IFR or version ID.
0x06	Program Longword	×	Program 4 bytes in a program flash block.
0x09	Erase Flash Sector	×	Erase all bytes in a program flash sector.
0x40	Read 1s All Blocks	×	Verify that the program flash block is erased then release MCU security.
0x41	Read Once	IFR	Read 4 bytes of a dedicated 64 byte field in the program flash 0 IFR.
0x43	Program Once	IFR	One-time program of 4 bytes of a dedicated 64-byte field in the program flash 0 IFR.
0x44	Erase All Blocks	×	Erase the program flash block, verify-erase and release MCU security. **NOTE:** An erase is only possible when all memory locations are unprotected.
0x45	Verify Backdoor Access Key	×	Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash.

32.5.8.3 Flash Commands by Mode

The following table shows the flash commands that can be executed in each flash operating mode.

Table 32-2. Flash Commands by Mode

FCMD	Command	NVM Normal			NVM Special		
		Unsecure	Secure	MEEN=10	Unsecure	Secure	MEEN=10
0x01	Read 1s Section	×	×	×	×	—	—
0x02	Program Check	×	×	×	×	—	—
0x03	Read Resource	×	×	×	×	—	—
0x06	Program Longword	×	×	×	×	—	—
0x09	Erase Flash Sector	×	×	×	×	—	—
0x40	Read 1s All Blocks	×	×	—	×	×	—

Table continues on the next page...

Table 32-2. Flash Commands by Mode (continued)

FCMD	Command	NVM Normal			NVM Special		
		Unsecure	Secure	MEEN=10	Unsecure	Secure	MEEN=10
0x41	Read Once	×	×	×	×	—	—
0x43	Program Once	×	×	×	×	—	—
0x44	Erase All Blocks	×	×	—	×	×	—
0x45	Verify Backdoor Access Key	×	×	×	×	—	—

32.5.9 Margin Read Commands

The Read-1s commands (Read 1s All Blocks, Read 1s Section) and the Program Check command have a margin choice parameter that allows the user to apply non-standard read reference levels to the program flash array reads performed by these commands. Using the preset 'user' and 'factory' margin levels, these commands perform their associated read operations at tighter tolerances than a 'normal' read. These non-standard read levels are applied only during the command execution. Basic flash array reads use the standard, un-margined, read reference level.

Only the 'normal' read level should be employed during normal flash usage. The non-standard, 'user' and 'factory' margin levels should be employed only in special cases. They can be used during special diagnostic routines to gain confidence that the device is not suffering from the end-of-life data loss customary of flash memory devices.

Erased ('1') and programmed ('0') bit states can degrade due to elapsed time and data cycling (number of times a bit is erased and re-programmed). The lifetime of the erased states is relative to the last erase operation. The lifetime of the programmed states is measured from the last program time.

The 'user' and 'factory' levels become, in effect, a minimum safety margin; i.e. if the reads pass at the tighter tolerances of the 'user' and 'factory' margins, then the 'normal' reads have at least this much safety margin before they experience data loss.

The 'user' margin is a small delta to the normal read reference level. 'User' margin levels can be employed to check that flash memory contents have adequate margin for normal level read operations. If unexpected read results are encountered when checking flash memory contents at the 'user' margin levels, loss of information might soon occur during 'normal' readout.

The 'factory' margin is a bigger deviation from the norm, a more stringent read criteria that should only be attempted immediately (or very soon) after completion of an erase or program command, early in the cycling life. 'Factory' margin levels can be used to check

that flash memory contents have adequate margin for long-term data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at 'factory' margin levels, the flash memory contents should be erased and reprogrammed.

CAUTION

Factory margin levels must only be used during verify of the initial factory programming.

32.5.10 Flash Command Description

This section describes all flash commands that can be launched by a command write sequence.

The flash memory module sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in a FCCOB field for the specific commands. Refer to the error handling table provided for each command.

Ensure that FSTAT[ACCERR] and FSTAT[FPVIOL] are cleared prior to starting the command write sequence. As described in Launch the Command by Clearing CCIF, a new command cannot be launched while these error flags are set.

Do not attempt to read a flash block while the flash memory module is running a command (FSTAT[CCIF] = 0) on that same block. The flash memory module may return invalid data to the MCU with the collision error flag (FSTAT[RDCOLERR]) set.

CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

32.5.10.1 Read 1s Section Command

The Read 1s Section command checks if a section of program flash memory is erased to the specified read margin level. The Read 1s Section command defines the starting address and the number of double-phrases to be verified.

Table 32-3. Read 1s Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x01 (RD1SEC)
1	Flash address [23:16] of the first double-phrase to be verified
2	Flash address [15:8] of the first double-phrase to be verified
3	Flash address [7:0][1] of the first double-phrase to be verified
4	Number of double-phrases to be verified [15:8]
5	Number of double-phrases to be verified [7:0]
6	Read-1 Margin Choice

1. Must be double-phrase aligned (Flash address [3:0] = 0000).

Upon clearing CCIF to launch the Read 1s Section command, the flash memory module sets the read margin for 1s according to Table 32-4 and then reads all locations within the specified section of flash memory. If the flash memory module fails to read all 1s (that is, the flash section is not erased), FSTAT[MGSTAT0] is set. FSTAT[CCIF] sets after the Read 1s Section operation completes.

Table 32-4. Margin Level Choices for Read 1s Section

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 32-5. Read 1s Section Command Error Handling

Error condition	Error bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin code is supplied.	FSTAT[ACCERR]
An invalid flash address is supplied.	FSTAT[ACCERR]
Flash address is not double-phrase aligned.	FSTAT[ACCERR]
The requested section crosses a Flash block boundary.	FSTAT[ACCERR]
The requested number of double-phrases is 0.	FSTAT[ACCERR]
Read-1s fails.	FSTAT[MGSTAT0]

32.5.10.2 Program Check Command

The Program Check command tests a previously programmed program flash longword to see if it reads correctly at the specified margin level.

Table 32-6. Program Check Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x02 (PGMCHK)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Margin Choice
8	Byte 0 expected data
9	Byte 1 expected data
A	Byte 2 expected data
B	Byte 3 expected data

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Check command, the flash memory module sets the read margin for 1s according to Table 32-7, reads the specified longword, and compares the actual read data to the expected data provided by the FCCOB. If the comparison at margin-1 fails, FSTAT[MGSTAT0] is set.

The flash memory module then sets the read margin for 0s, re-reads, and compares again. If the comparison at margin-0 fails, FSTAT[MGSTAT0] is set. FSTAT[CCIF] is set after the Program Check operation completes.

The supplied address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10,
- Byte 0 data is programmed to byte address start+0b11.

NOTE

See the description of margin reads, Margin Read Commands

Table 32-7. Margin Level Choices for Program Check

Read Margin Choice	Margin Level Description
0x01	Read at 'User' margin-1 and 'User' margin-0
0x02	Read at 'Factory' margin-1 and 'Factory' margin-0

Table 32-8. Program Check Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]

Table continues on the next page...

Table 32-8. Program Check Command Error Handling (continued)

Error Condition	Error Bit
Flash address is not longword aligned	FSTAT[ACCERR]
An invalid margin choice is supplied	FSTAT[ACCERR]
Either of the margin reads does not match the expected data	FSTAT[MGSTAT0]

32.5.10.3 Read Resource Command

The Read Resource command allows the user to read data from special-purpose memory resources located within the flash memory module. The special-purpose memory resources available include program flash IFR space and the Version ID field. Each resource is assigned a select code as shown in Table 32-10.

Table 32-9. Read Resource Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x03 (RDRSRC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
Returned Values	
4	Read Data [31:24]
5	Read Data [23:16]
6	Read Data [15:8]
7	Read Data [7:0]
User-provided values	
8	Resource Select Code (see Table 32-10)

1. Must be longword aligned (Flash address [1:0] = 00).

Table 32-10. Read Resource Select Codes

Resource Select Code	Description	Resource Size	Local Address Range
0x00	Program Flash 0 IFR	256 Bytes	0x00_0000–0x00_00FF
0x01[1]	Version ID	8 Bytes	0x00_0000–0x00_0007

1. Located in program flash 0 reserved space.

After clearing CCIF to launch the Read Resource command, four consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag sets after the Read Resource operation completes. The Read Resource command exits with an access error if an invalid resource code is provided or if the address for the applicable area is out-of-range.

Table 32-11. Read Resource Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid resource code is entered	FSTAT[ACCERR]
Flash address is out-of-range for the targeted resource.	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]

32.5.10.4 Program Longword Command

The Program Longword command programs four previously-erased bytes in the program flash memory using an embedded algorithm.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 32-12. Program Longword Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x06 (PGM4)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Byte 0 program value
5	Byte 1 program value
6	Byte 2 program value
7	Byte 3 program value

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Longword command, the flash memory module programs the data bytes into the flash using the supplied address. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Longword operation.

The programming operation is unidirectional. It can only move NVM bits from the erased state ('1') to the programmed state ('0'). Erased bits that fail to program to the '0' state are flagged as errors in FSTAT[MGSTAT0]. The CCIF flag is set after the Program Longword operation completes.

The supplied address must be longword aligned (flash address [1:0] = 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10, and
- Byte 0 data is programmed to byte address start+0b11.

Table 32-13. Program Longword Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
Flash address points to a protected area	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

32.5.10.5 Erase Flash Sector Command

The Erase Flash Sector operation erases all addresses in a flash sector.

Table 32-14. Erase Flash Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x09 (ERSSCR)
1	Flash address [23:16] in the flash sector to be erased
2	Flash address [15:8] in the flash sector to be erased
3	Flash address [7:0][1] in the flash sector to be erased

1. Must be double-phrase aligned (flash address [3:0] = 0000).

After clearing CCIF to launch the Erase Flash Sector command, the flash memory module erases the selected program flash sector and then verifies that it is erased. The Erase Flash Sector command aborts if the selected sector is protected (see the description

of the FPROT registers). If the erase-verify fails the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase Flash Sector operation completes. The Erase Flash Sector command is suspendable (see the FCNFG[ERSSUSP] bit and Figure 32-4).

Table 32-15. Erase Flash Sector Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid Flash address is supplied	FSTAT[ACCERR]
Flash address is not double-phrase aligned	FSTAT[ACCERR]
The selected program flash sector is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s Section command to verify all bits are erased.

32.5.10.5.1 Suspending an Erase Flash Sector Operation

To suspend an Erase Flash Sector operation set the FCNFG[ERSSUSP] bit when CCIF, ACCERR, and FPVIOL are clear and the CCOB command field holds the code for the Erase Flash Sector command. During the Erase Flash Sector operation (see Erase Flash Sector Command), the flash memory module samples the state of the ERSSUSP bit at convenient points. If the flash memory module detects that the ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the flash memory module sets CCIF. While ERSSUSP is set, all writes to flash registers are ignored except for writes to the FSTAT and FCNFG registers.

If an Erase Flash Sector operation effectively completes before the flash memory module detects that a suspend request has been made, the flash memory module clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the flash memory module sets CCIF and leaves the ERSSUSP bit set. While CCIF is set, the ERSSUSP bit can only be cleared to prevent the withdrawal of a suspend request before the flash memory module has acknowledged it.

32.5.10.5.2 Resuming a Suspended Erase Flash Sector Operation

If the ERSSUSP bit is still set when CCIF is cleared to launch the next command, the previous Erase Flash Sector operation resumes. The flash memory module acknowledges the request to resume a suspended operation by clearing the ERSSUSP bit. A new suspend request can then be made by setting ERSSUSP. A single Erase Flash Sector operation can be suspended and resumed multiple times.

There is a minimum elapsed time limit of 4.3 msec between the request to resume the Erase Flash Sector operation (CCIF is cleared) and the request to suspend the operation again (ERSSUSP is set). This minimum time period is required to ensure that the Erase

Flash Sector operation will eventually complete. If the minimum period is continually violated, i.e. the suspend requests come repeatedly and too quickly, no forward progress is made by the Erase Flash Sector algorithm. The resume/suspend sequence runs indefinitely without completing the erase.

32.5.10.5.3 Aborting a Suspended Erase Flash Sector Operation

The user may choose to abort a suspended Erase Flash Sector operation by clearing the ERSSUSP bit prior to clearing CCIF for the next command launch. When a suspended operation is aborted, the flash memory module starts the new command using the new FCCOB contents.

Note

Aborting the erase leaves the bitcells in an indeterminate, partially-erased state. Data in this sector is not reliable until a new erase command fully completes.

The following figure shows how to suspend and resume the Erase Flash Sector operation.

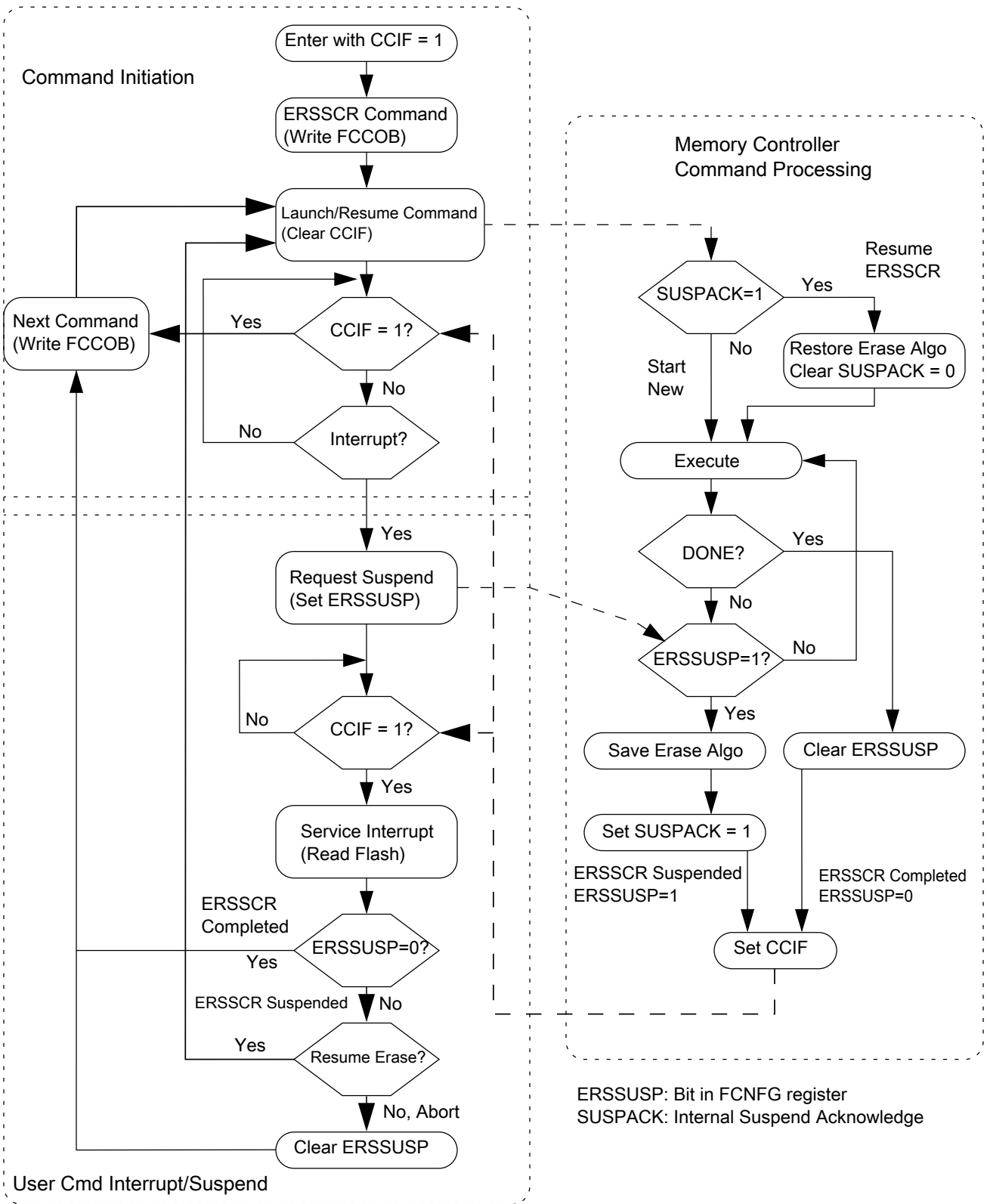


Figure 32-4. Suspend and Resume of Erase Flash Sector Operation

32.5.10.6 Read 1s All Blocks Command

The Read 1s All Blocks command checks if the program flash blocks have been erased to the specified read margin level, if applicable, and releases security if the readout passes, i.e. all data reads as '1'.

Table 32-16. Read 1s All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x40 (RD1ALL)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Blocks command, the flash memory module :

- sets the read margin for 1s according to Table 32-17,
- checks the contents of the program flash are in the erased state.

If the flash memory module confirms that these memory resources are erased, security is released by setting the FSEC[SEC] field to the unsecure state. The security byte in the flash configuration field (see Flash Configuration Field Description) remains unaffected by the Read 1s All Blocks command. If the read fails, i.e. all memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

Table 32-17. Margin Level Choices for Read 1s All Blocks

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 32-18. Read 1s All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

32.5.10.7 Read Once Command

The Read Once command provides read access to special 64-byte fields located in the program flash 0 IFR (see Program Flash IFR Map and Program Once Field). Access to the Program Once field is via 16 records (index values 0x00 - 0x0F), each 4 bytes long. These fields are programmed using the Program Once command described in Program Once Command.

Table 32-19. Read Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x41 (RDONCE)
1	Program Once record index (0x00 - 0x0F)
2	Not used
3	Not used
Returned Values	
4	Program Once byte 0 value
5	Program Once byte 1 value
6	Program Once byte 2 value
7	Program Once byte 3 value

After clearing CCIF to launch the Read Once command, a 4-byte Program Once record is read and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. Valid record index values for the Read Once command range from 0x00 - 0x0F. During execution of the Read Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data. The Read Once command can be executed any number of times.

Table 32-20. Read Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]

32.5.10.8 Program Once Command

The Program Once command enables programming to special 64-byte fields in the program flash 0 IFR (see Program Flash IFR Map and Program Once Field). Access to the Program Once field is via 16 records (index values 0x00 - 0x0F), each 4 bytes long. These records can be read using the Read Once command (see Read Once Command) or using the Read Resource command (see Read Resource Command). These records can be programmed only once since the program flash 0 IFR cannot be erased.

Table 32-21. Program Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x43 (PGMONCE)
1	Program Once record index (0x00 - 0x0F)
2	Not Used
3	Not Used
4	Program Once byte 0 value
5	Program Once byte 1 value
6	Program Once byte 2 value
7	Program Once byte 3 value

After clearing CCIF to launch the Program Once command, the flash memory module first verifies that the selected record is erased. If erased, then the selected record is programmed using the values provided. The Program Once command also verifies that the programmed values read back correctly. The CCIF flag is set after the Program Once operation has completed.

Any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. Valid record index values for the Program Once command range from 0x00 - 0x0F. During execution of the Program Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data.

Table 32-22. Program Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]
The requested record has already been programmed to a non-FFFF value[1]	FSTAT[ACCERR]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

1. If a Program Once record is initially programmed to 0xFFFF_FFFF, the Program Once command is allowed to execute again on that same record.

32.5.10.9 Erase All Blocks Command

The Erase All Blocks operation erases all flash memory, verifies all memory contents, and releases MCU security.

Table 32-23. Erase All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x44 (ERSALL)

After clearing CCIF to launch the Erase All Blocks command, the flash memory module erases all program flash memory, then verifies that all are erased.

If the flash memory module verifies that all flash memories were properly erased, security is released by setting the FSEC[SEC] field to the unsecure state. The Erase All Blocks command aborts if any flash region is protected. The security byte and all other contents of the flash configuration field (see Flash Configuration Field Description) are erased by the Erase All Blocks command. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks operation completes.

Table 32-24. Erase All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any region of the program flash memory is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation[1]	FSTAT[MGSTAT0]

1. User margin read may be run using the Read 1s All Blocks command to verify all bits are erased.

32.5.10.9.1 Triggering an Erase All External to the Flash Memory Module

The functionality of the Erase All Blocks command is also available in an uncommanded fashion outside of the flash memory. Refer to the device's Chip Configuration details for information on this functionality.

Before invoking the external erase all function, the FSTAT[ACCERR and PVIOL] flags must be cleared and the FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory regardless of the protection settings. If the post-erase verify passes, the routine then releases security by setting the FSEC[SEC] field register to the unsecure state. The security byte in the Flash Configuration Field is also programmed to the unsecure state. The status of the erase-all request is reflected in the FCNFG[ERSAREQ] bit. The FCNFG[ERSAREQ] bit is cleared once the operation completes and the normal FSTAT error reporting is available, except FPVIOL, as described in Erase All Blocks Command.

32.5.10.10 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command only executes if the mode and security conditions are satisfied (see Flash Commands by Mode). Execution of the Verify Backdoor Access Key command is further qualified by the FSEC[KEYEN] bits. The Verify Backdoor Access Key command releases security if user-supplied keys in the FCCOB match those stored in the Backdoor Comparison Key bytes of the Flash

Configuration Field (see Flash Configuration Field Description). The column labelled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

Table 32-25. Verify Backdoor Access Key Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]	Flash Configuration Field Offset Address
0	0x45 (VFYKEY)	
1-3	Not Used	
4	Key Byte 0	0x0_0003
5	Key Byte 1	0x0_0002
6	Key Byte 2	0x0_0001
7	Key Byte 3	0x0_0000
8	Key Byte 4	0x0_0007
9	Key Byte 5	0x0_0006
A	Key Byte 6	0x0_0005
B	Key Byte 7	0x0_0004

After clearing CCIF to launch the Verify Backdoor Access Key command, the flash memory module checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the flash memory module sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the flash memory module compares the key provided in FCCOB to the backdoor comparison key in the Flash Configuration Field. If the backdoor keys match, the FSEC[SEC] field is changed to the unsecure state and security is released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are immediately aborted and the FSTAT[ACCERR] bit is (again) set to 1 until a reset of the flash memory module module occurs. If the entire 8-byte key is all zeros or all ones, the Verify Backdoor Access Key command fails with an access error. The CCIF flag is set after the Verify Backdoor Access Key operation completes.

Table 32-26. Verify Backdoor Access Key Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
The supplied key is all-0s or all-Fs	FSTAT[ACCERR]
An incorrect backdoor key is supplied	FSTAT[ACCERR]
Backdoor key access has not been enabled (see the description of the FSEC register)	FSTAT[ACCERR]
This command is launched and the backdoor key has mismatched since the last power down reset	FSTAT[ACCERR]

32.5.11 Security

The flash memory module provides security information to the MCU based on contents of the FSEC security register.

The MCU then limits access to flash memory resources as defined in the device's Chip Configuration details. During reset, the flash memory module initializes the FSEC register using data read from the security byte of the Flash Configuration Field (see Flash Configuration Field Description).

The following fields are available in the FSEC register. The settings are described in the Flash Security Register (FTFA_FSEC) details.

Flash security features are discussed further in AN4507: Using the Kinetis Security and Flash Protection Features . Note that not all features described in the application note are available on this device.

Table 32-27. FSEC register fields

FSEC field	Description
KEYEN	Backdoor Key Access
MEEN	Mass Erase Capability
FSLACC	Factory Security Level Access
SEC	MCU security

32.5.11.1 Flash Memory Access by Mode and Security

The following table summarizes how access to the flash memory module is affected by security and operating mode.

Table 32-28. Flash Memory Access Summary

Operating Mode	Chip Security State	
	Unsecure	Secure
NVM Normal	Full command set	
NVM Special	Full command set	Only the Erase All Blocks and Read 1s All Blocks commands.

32.5.11.2 Changing the Security State

The security state out of reset can be permanently changed by programming the security byte of the flash configuration field. This assumes that you are starting from a mode where the necessary program flash erase and program commands are available and that the region of the program flash containing the flash configuration field is unprotected. If the flash security byte is successfully programmed, its new value takes affect after the next chip reset.

32.5.11.2.1 Unsecuring the Chip Using Backdoor Key Access

The chip can be unsecured by using the backdoor key access feature, which requires knowledge of the contents of the 8-byte backdoor key value stored in the Flash Configuration Field (see Flash Configuration Field Description). If the FSEC[KEYEN] bits are in the enabled state, the Verify Backdoor Access Key command (see Verify Backdoor Access Key Command) can be run; it allows the user to present prospective keys for comparison to the stored keys. If the keys match, the FSEC[SEC] bits are changed to unsecure the chip. The entire 8-byte key cannot be all 0s or all 1s; that is, 0000_0000_0000_0000h and FFFF_FFFF_FFFF_FFFFh are not accepted by the Verify Backdoor Access Key command as valid comparison values. While the Verify Backdoor Access Key command is active, program flash memory is not available for read access and returns invalid data.

The user code stored in the program flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN bits are in the enabled state, the chip can be unsecured by the following backdoor key access sequence:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Verify Backdoor Access Key Command
2. If the Verify Backdoor Access Key command is successful, the chip is unsecured and the FSEC[SEC] bits are forced to the unsecure state

An illegal key provided to the Verify Backdoor Access Key command prohibits further use of the Verify Backdoor Access Key command. A reset of the chip is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the chip is unsecured by changing the FSEC[SEC] bits. A successful execution of the Verify Backdoor Access Key command changes the security in the FSEC register only. It does not alter the security byte or the keys stored in the Flash Configuration Field (Flash Configuration Field Description). After the next reset of the chip, the security state of the flash memory

module reverts back to the flash security byte in the Flash Configuration Field. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the program flash protection registers.

If the backdoor keys successfully match, the unsecured chip has full control of the contents of the Flash Configuration Field. The chip may erase the sector containing the Flash Configuration Field and reprogram the flash security byte to the unsecure state and change the backdoor keys to any desired value.

32.5.12 Reset Sequence

On each system reset the flash memory module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FOPT, and FSEC registers.

FSTAT[CCIF] is cleared throughout the reset sequence. The flash memory module holds off CPU access during the reset sequence. Flash reads are possible when the hold is removed. Completion of the reset sequence is marked by setting CCIF which enables flash user commands.

If a reset occurs while any flash command is in progress, that command is immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed. Commands and operations do not automatically resume after exiting reset.

Chapter 33
Cyclic redundancy check (CRC)

33.1 Introduction

The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial and other parameters required to implement a 16-bit or 32-bit CRC standard.

The 16/32-bit code is calculated for 32 bits of data at a time.

33.1.1 Features

Features of the CRC module include:

- Hardware CRC generator circuit using a 16-bit or 32-bit programmable shift register
- Programmable initial seed value and polynomial
- Option to transpose input data or output data (the CRC result) bitwise or bytewise. This option is required for certain CRC standards. A bytewise transpose operation is not possible when accessing the CRC data register via 8-bit accesses. In this case, the user's software must perform the bytewise transpose function.
- Option for inversion of final CRC result
- 32-bit CPU register programming interface

33.1.2 Block diagram

The following is a block diagram of the CRC.

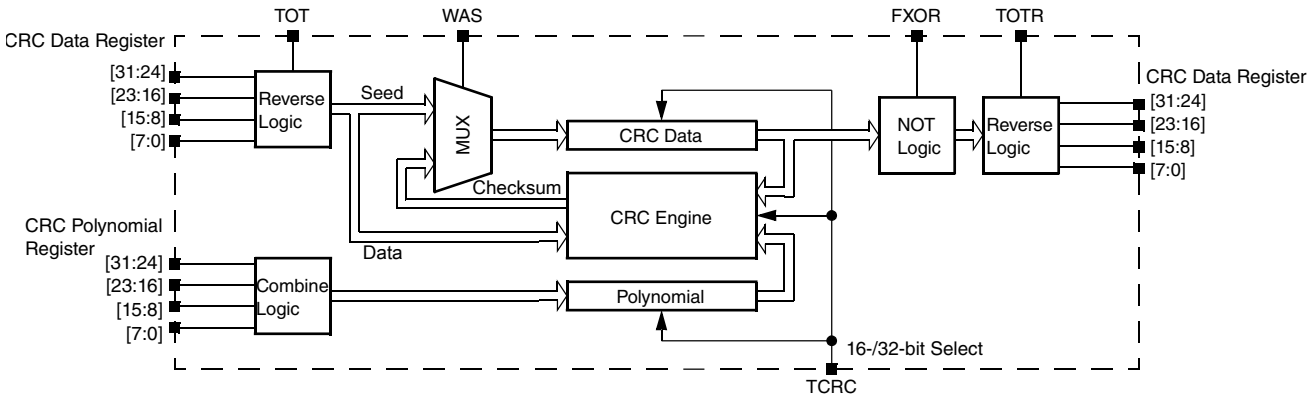


Figure 33-1. Programmable cyclic redundancy check (CRC) block diagram

33.1.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

33.1.3.1 Run mode

This is the basic mode of operation.

33.1.3.2 Low-power modes (Wait or Stop)

Any CRC calculation in progress stops when the MCU enters a low-power mode that disables the module clock. It resumes after the clock is enabled or via the system reset for exiting the low-power mode. Clock gating for this module is dependent on the MCU.

33.2 Memory map and register descriptions

CRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_2000	CRC Data register (CRC_DATA)	32	R/W	FFFF_FFFFh	33.2.1/645
4003_2004	CRC Polynomial register (CRC_GPOLY)	32	R/W	0000_1021h	33.2.2/646
4003_2008	CRC Control register (CRC_CTRL)	32	R/W	0000_0000h	33.2.3/646

33.2.1 CRC Data register (CRC_DATA)

The CRC Data register contains the value of the seed, data, and checksum. When CTRL[WAS] is set, any write to the data register is regarded as the seed value. When CTRL[WAS] is cleared, any write to the data register is regarded as data for general CRC computation.

In 16-bit CRC mode, the HU and HL fields are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous; with MSB of data value written first.

After all data values are written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is available in the LU and LL fields. In 32-bit CRC mode, all fields contain the result. Reads of this register at any time return the intermediate CRC value, provided the CRC module is configured.

Address: 4003_2000h base + 0h offset = 4003_2000h

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

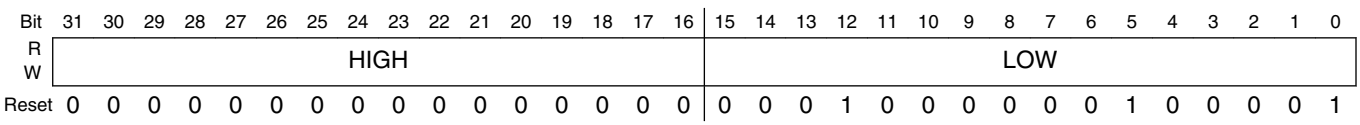
CRC_DATA field descriptions

Field	Description
31–24 HU	CRC High Upper Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
23–16 HL	CRC High Lower Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
15–8 LU	CRC Low Upper Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.
LL	CRC Low Lower Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.

33.2.2 CRC Polynomial register (CRC_GPOLY)

This register contains the value of the polynomial for the CRC calculation. The HIGH field contains the upper 16 bits of the CRC polynomial, which are used only in 32-bit CRC mode. Writes to the HIGH field are ignored in 16-bit CRC mode. The LOW field contains the lower 16 bits of the CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 4003_2000h base + 4h offset = 4003_2004h



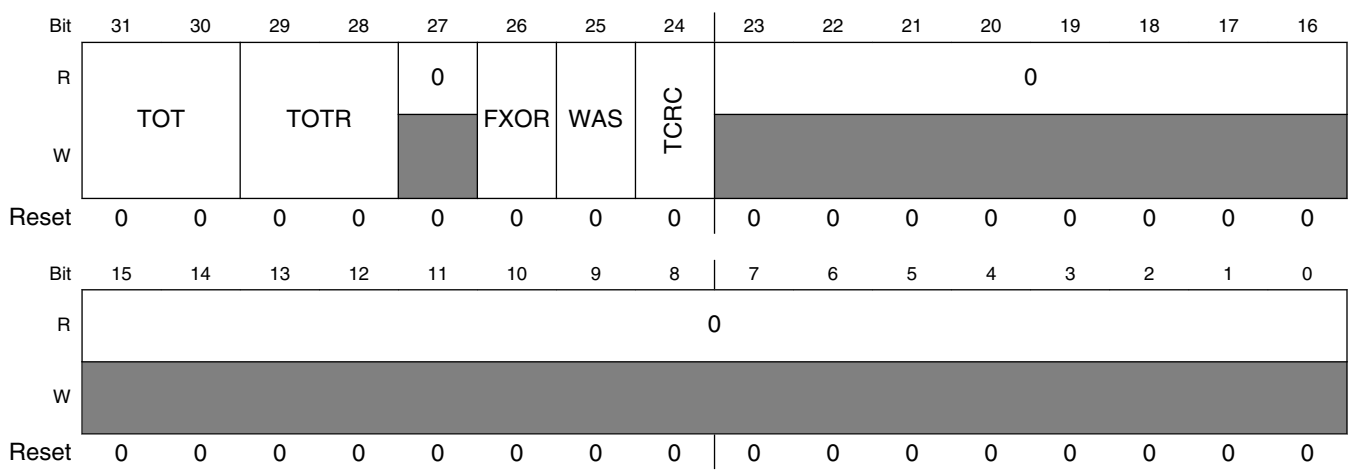
CRC_GPOLY field descriptions

Field	Description
31–16 HIGH	High Polynominal Half-word Writable and readable in 32-bit CRC mode (CTRL[TCRC] is 1). This field is not writable in 16-bit CRC mode (CTRL[TCRC] is 0).
LOW	Low Polynominal Half-word Writable and readable in both 32-bit and 16-bit CRC modes.

33.2.3 CRC Control register (CRC_CTRL)

This register controls the configuration and working of the CRC module. Appropriate bits must be set before starting a new CRC calculation. A new CRC calculation is initialized by asserting CTRL[WAS] and then writing the seed into the CRC data register.

Address: 4003_2000h base + 8h offset = 4003_2008h



CRC_CTRL field descriptions

Field	Description
31–30 TOT	Type Of Transpose For Writes Defines the transpose configuration of the data written to the CRC data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
29–28 TOTR	Type Of Transpose For Read Identifies the transpose configuration of the value read from the CRC Data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 FXOR	Complement Read Of CRC Data Register Some CRC protocols require the final checksum to be XORed with 0xFFFFFFFF or 0xFFFF. Asserting this bit enables on the fly complementing of read data. 0 No XOR on reading. 1 Invert or complement the read value of the CRC Data register.
25 WAS	Write CRC Data Register As Seed When asserted, a value written to the CRC data register is considered a seed value. When deasserted, a value written to the CRC data register is taken as data for CRC computation. 0 Writes to the CRC data register are data values. 1 Writes to the CRC data register are seed values.
24 TCRC	Width of CRC protocol. 0 16-bit CRC protocol. 1 32-bit CRC protocol.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

33.3 Functional description

33.3.1 CRC initialization/reinitialization

To enable the CRC calculation, the user must program CRC_CTRL[WAS], CRC_GPOLY,necessary parameters for transposition and CRC result inversion in the applicable registers. Asserting CRC_CTRL[WAS] enables the programming of the seed value into the CRC_DATA register.

After a completed CRC calculation, the module can be reinitialized for a new CRC computation by reasserting CRC_CTRL[WAS] and programming a new, or previously used, seed value. All other parameters must be set before programming the seed value and subsequent data values.

33.3.2 CRC calculations

In 16-bit and 32-bit CRC modes, data values can be programmed 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes can lead to an incorrect CRC computation.

33.3.2.1 16-bit CRC

To compute a 16-bit CRC:

1. Clear CRC_CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 16-bit polynomial to the CRC_GPOLY[LOW] field. The CRC_GPOLY[HIGH] field is not usable in 16-bit CRC mode.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to CRC_DATA[LU:LL]. CRC_DATA[HU:HL] are not used.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[LU:LL].

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

33.3.2.2 32-bit CRC

To compute a 32-bit CRC:

1. Set CRC_CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 32-bit polynomial to CRC_GPOLY[HIGH:LOW].
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to CRC_DATA[HU:HL:LU:LL].
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[HU:HL:LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[HU:HL:LU:LL]. The CRC is calculated bytewise, and two clocks are required to complete one CRC calculation.

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

33.3.3 Transpose feature

By default, the transpose feature is not enabled. However, some CRC standards require the input data and/or the final checksum to be transposed. The user software has the option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on the fly while being read or written.

Some protocols use little endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

33.3.3.1 Types of transpose

The CRC module provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] or CTRL[TOTR] fields, according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from the CRC data register:

1. CTRL[TOT] or CTRL[TOTR] is 00.

No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 01

   Bits in a byte are transposed, while bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}

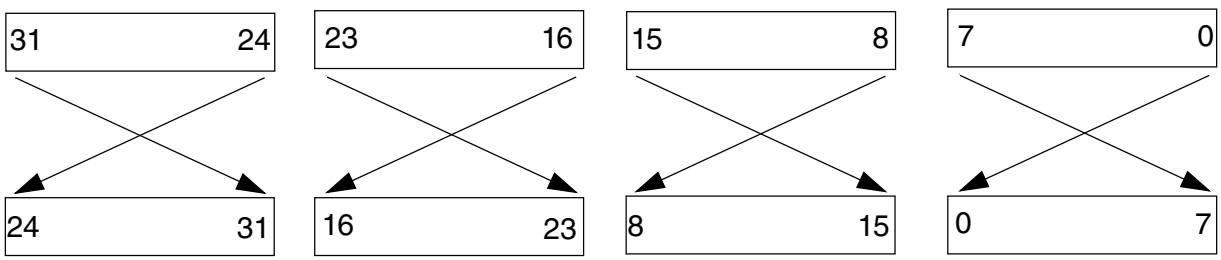


Figure 33-2. Transpose type 01

3. CTRL[TOT] or CTRL[TOTR] is 10.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes = {reg[0:7], reg[8:15],reg[16:23], reg[24:31]}

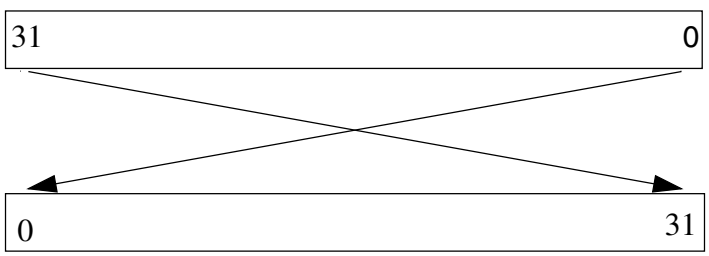


Figure 33-3. Transpose type 10

4. CTRL[TOT] or CTRL[TOTR] is 11.

   Bytes are transposed, but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}

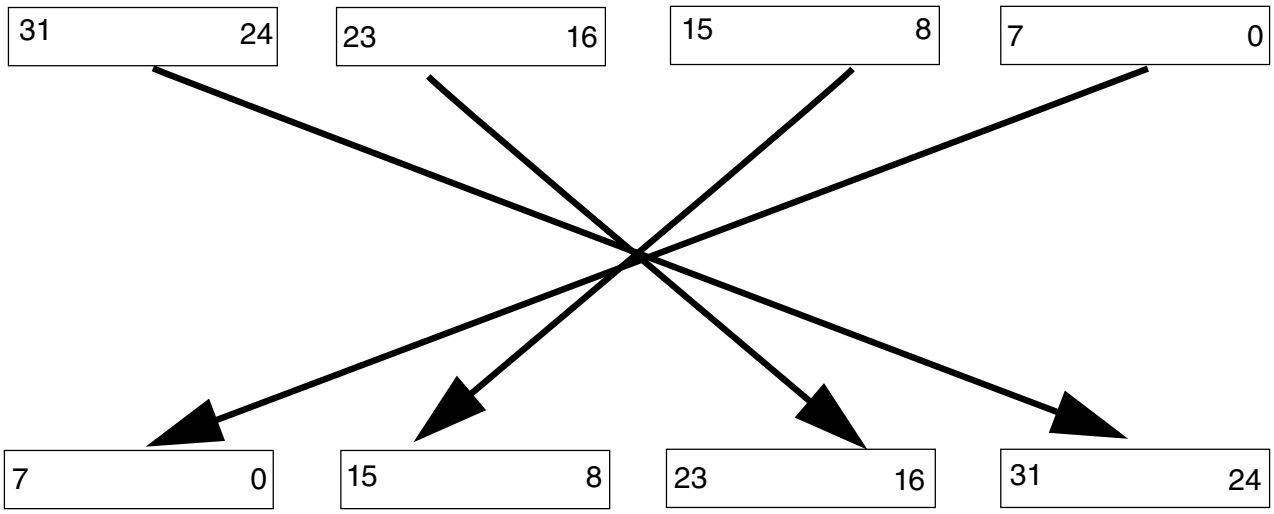


Figure 33-4. Transpose type 11

NOTE

- For 8-bit and 16-bit write accesses to the CRC data register, the data is transposed with zeros on the unused byte or bytes (taking 32 bits as a whole), but the CRC is calculated on the valid byte(s) only.
- When reading the CRC data register for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in the CRC[HU:HL] fields. The user software must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

33.3.4 CRC result complement

When CTRL[FXOR] is set, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in the CRC data register every time the CRC data register is read. When CTRL[FXOR] is cleared, reading the CRC data register accesses the raw checksum value.

Chapter 34
12-bit Cyclic Analog-to-Digital Converter (ADC)

34.1 Chip-specific Cyclic ADC information

34.1.1 Cyclic ADC instantiation

It is a dual ADC. The signals of its first ADC are labeled A, as in ANA, ADCA, VREFLA, and VREFHA. The signals of its second ADC are labeled B, as in ANB, ADCB, VREFLB, and VREFHB.

34.1.2 ADC input channel multiplexing

The cyclic ADC channel mux has been expanded with additional 8:1 multiplexers on four ADC channels, namely ADCA_CH6, ADCA_CH7, ADCB_CH6, and ADCB_CH7. The additional analog multiplexers, external to the cyclic ADC, can be selected via the SIM_ADCOPT register. Four fields in the SIM_ADCOPT register provide additional input analog signals to the cyclic ADC, namely ADCACH6SEL, ADCACH7SEL, ADCBCH6SEL, and ADCBCH7SEL. In certain packages, not all channels are bonded out. See Signal Multiplexing to identify the available channels.

34.1.3 Cyclic ADC SYNC signal connections

XBARA_OUT12 and PDB0 channel trigger outputs can trigger ADCA and ADCB (parallel sampling) conversion via SYNC0 input. XBARA_OUT13 and PDB1 channel trigger outputs can trigger ADCB conversions via SYNC1 input.

Each ADC can be synchronized to another module, such as a PWMA, through the XBARA connections.

Each ADC can be synchronized to PDB0/1 trigger input with a programmable delay through the PDB0/1 connections.

See SIM_SOPT7 and PDB0 Input Trigger Connections.

34.1.4 Cyclic ADC and eFlexPWM connections

Within the chip, the cyclic ADC has internal connections for eFlexPWM control.

Table 34-1. Cyclic ADC and eFlexPWM Connections

Cyclic ADC Outputs	eFlexPWM Inputs
an0_pwm	PWMA0_EXTB
an1_pwm	PWMA1_EXTB
an2_pwm	PWMA2_EXTB
an3_pwm	PWMA3_EXTB

34.1.5 ADC in low power mode

The ADC can be configured to be active in low power mode like STOP/VLPS mode. In this mode, the ADC is clocked with MCGIRCCLK. User should configure the MCG block to make MCGIRCCLK work at 4 MHz before entering low power mode.

34.1.6 ADC channel muxing

Each of ADCA_CH6/7 and ADCB_CH6/7 have analog 8 to 1 muxing to expand the channels. The muxing is controlled by SIM_ADCOPT.

34.2 Introduction

34.2.1 Overview

The analog-to-digital converter (ADC) is one dual 12-bit ADC in which each ADC converter has a separate voltage reference and control block.

34.2.2 Features

The analog-to-digital (ADC) converter function consists of two separate analog-to-digital converters, each with eight analog inputs and its own sample and hold circuit. A common digital control module configures and controls the functioning of the converters. ADC features include:

- 12-bit resolution
- Designed for maximum ADC clock frequency of 25 MHz with 40 ns period
- Sampling rate up to 8.83 million samples per second[1]
- Single conversion time of 8.5 ADC clock cycles (8.5 × 40 ns =340 ns)
- Additional conversion time of 6 ADC clock cycles (6 × 40 ns = 240 ns)
- Eight conversions in 26.5 ADC clock cycles (26.5 × 40 ns = 1.060 µs) using parallel mode
- Can be synchronized to other peripherals that are connected to an internal Inter-Peripheral Crossbar module and PDB, such as the PWM, through the SYNC0/1 input signal
- Sequentially scans and stores up to sixteen measurements
- Scans and stores up to eight measurements each on two ADC converters operating simultaneously and in parallel
- Scans and stores up to eight measurements each on two ADC converters operating asynchronously to each other in parallel
- A scan can pause and await new SYNC input prior to continuing
- Gains the input signal by x1, x2, or x4
- Optional interrupts at end of scan if an out-of-range limit is exceeded or there is a zero crossing
- Optional DMA function to transfer conversion data at the end of a scan or when a sample is ready to be read.
- Optional sample correction by subtracting a pre-programmed offset value

1. In loop mode, the time between each conversion is 6 ADC clock cycles (240 ns). Using simultaneous conversion, two samples can be obtained in 240 ns. Samples per second is calculated according to 240 ns per two samples or 8,333,333 samples per second.

- Signed or unsigned result
- Single-ended or differential inputs
- PWM outputs with hysteresis for three of the analog inputs

34.2.3 Block Diagram

The following figure illustrates the dual ADC configuration.

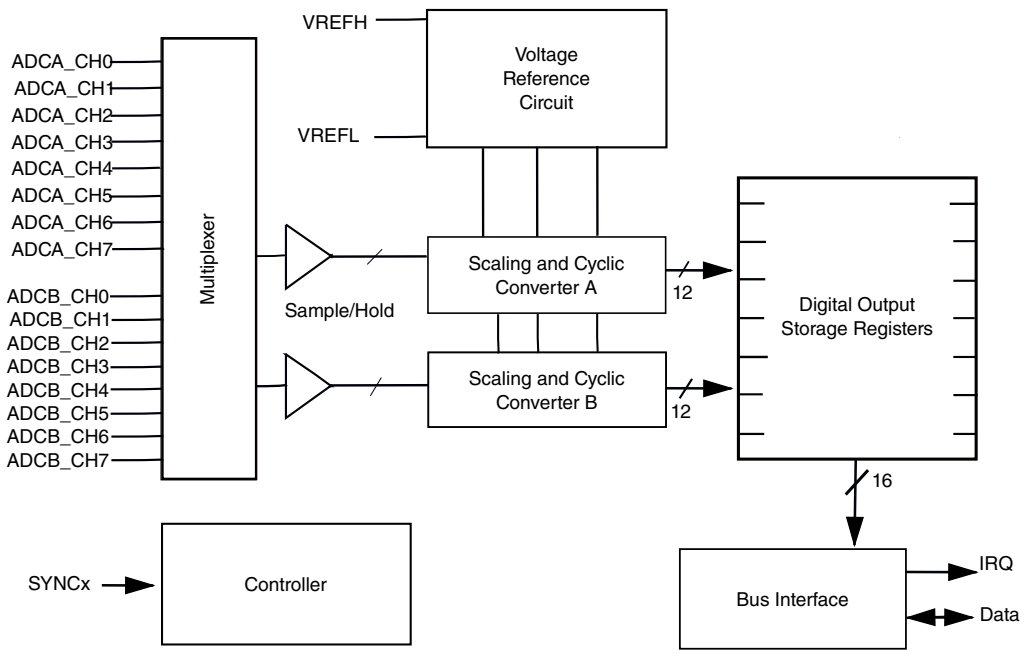


Figure 34-1. Dual ADC Block Diagram

34.3 Signal Descriptions

34.3.1 Overview

Table 34-2. External Signal Properties

Name	I/O Type	Function	Reset State	Notes
ADCA_CH0–ADCB_CH7	I	Analog Input Pins	n/a	—
VREFH	I	Voltage Reference Pin of ADCA	n/a	Selectable between VREFH and ADCA_CH2
VREFL	I	Voltage Reference Pin of ADCA	n/a	Selectable between VREFL and ADCA_CH3

Table continues on the next page...

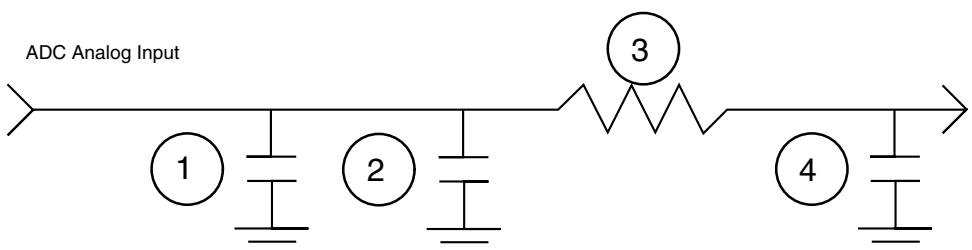
Table 34-2. External Signal Properties (continued)

Name	I/O Type	Function	Reset State	Notes
VREFH	I	Voltage Reference Pin of ADCB	n/a	Selectable between VREFH and ADCB_CH2
VREFL	I	Voltage Reference Pin of ADCB	n/a	Selectable between VREFL and ADCB_CH3
VDDA	Supply	ADC Power	n/a	—
VSSA	Supply	ADC Ground	n/a	—

34.3.2 External Signal Descriptions

34.3.2.1 Analog Input Pins (ADCA_CH[0:7] and ADCB_CH[0:7])

Each ADC module has sixteen analog input pins that are subdivided into two sets of eight (ADCA_CH[0:7] and ADCB_CH[0:7]), each with their own S/H unit and converter. This configuration allows simultaneous sampling of two selected channels, one from each subgroup. Sequential scans have access to all sixteen analog inputs. During parallel scans, each ADC converter has access to its eight analog inputs. An equivalent circuit for an analog input is shown below:



1. Parasitic capacitance due to package, pin-to-pin, and pin-to-package base coupling. 1.8pf
2. Parasitic capacitance due to the chip bond pad, ESD protection devices, and signal routing. 2.04pf
3. Equivalent resistance for the ESD isolation resistor and the channel select multiplexer. 500Ω
4. Sampling capacitor at the sample and hold circuit. 1pf

Figure 34-2. Equivalent Analog Input Circuit

34.3.2.2 Voltage Reference Pins (VREFH and VREFL)

The voltage difference between V_{REFH} and V_{REFL} provides the reference voltage against which all analog inputs are measured. V_{REFH} is nominally set to V_{DDA}. V_{REFL} is nominally set to 0V. Any external reference voltage should come from a low noise

filtered source. The external reference source should provide up to 1mA of reference current. illustrates the internal workings of the ADC voltage reference circuit. V_{REFH} must be noise filtered. A minimum configuration is shown in the figure.

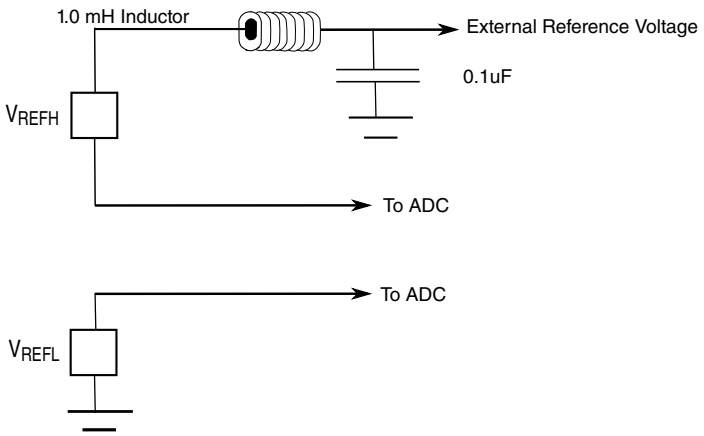


Figure 34-3. ADC Voltage Reference Circuit

Dedicated power supply pins, V_{REFH} and V_{REFL}, are provided to reduce noise coupling and to improve accuracy. The power to these pins should come from a low noise filtered source. Uncoupling capacitors should be connected between V_{REFH} and V_{REFL}.

V_{SS} is shared between both analog and digital circuitry.

34.4 Memory Map and Registers

ADC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_C000	ADC Control Register 1 (ADC_CTRL1)	16	R/W	5005h	34.4.1/662
4005_C002	ADC Control Register 2 (ADC_CTRL2)	16	R/W	5044h	34.4.2/666
4005_C004	ADC Zero Crossing Control 1 Register (ADC_ZXCTRL1)	16	R/W	0000h	34.4.3/668
4005_C006	ADC Zero Crossing Control 2 Register (ADC_ZXCTRL2)	16	R/W	0000h	34.4.4/669
4005_C008	ADC Channel List Register 1 (ADC_CLIST1)	16	R/W	3210h	34.4.5/671
4005_C00A	ADC Channel List Register 2 (ADC_CLIST2)	16	R/W	7654h	34.4.6/672
4005_C00C	ADC Channel List Register 3 (ADC_CLIST3)	16	R/W	BA98h	34.4.7/674
4005_C00E	ADC Channel List Register 4 (ADC_CLIST4)	16	R/W	FEDCh	34.4.8/676
4005_C010	ADC Sample Disable Register (ADC_SDIS)	16	R/W	F0F0h	34.4.9/678
4005_C012	ADC Status Register (ADC_STAT)	16	R	0000h	34.4.10/ 679
4005_C014	ADC Ready Register (ADC_RDY)	16	R	0000h	34.4.11/ 681

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_C016	ADC Low Limit Status Register (ADC_LOLIMSTAT)	16	w1c	0000h	34.4.12/ 681
4005_C018	ADC High Limit Status Register (ADC_HILIMSTAT)	16	w1c	0000h	34.4.13/ 682
4005_C01A	ADC Zero Crossing Status Register (ADC_ZXSTAT)	16	w1c	0000h	34.4.14/ 682
4005_C01C	ADC Result Registers with sign extension (ADC_RSLT0)	16	R/W	0000h	34.4.15/ 683
4005_C01E	ADC Result Registers with sign extension (ADC_RSLT1)	16	R/W	0000h	34.4.15/ 683
4005_C020	ADC Result Registers with sign extension (ADC_RSLT2)	16	R/W	0000h	34.4.15/ 683
4005_C022	ADC Result Registers with sign extension (ADC_RSLT3)	16	R/W	0000h	34.4.15/ 683
4005_C024	ADC Result Registers with sign extension (ADC_RSLT4)	16	R/W	0000h	34.4.15/ 683
4005_C026	ADC Result Registers with sign extension (ADC_RSLT5)	16	R/W	0000h	34.4.15/ 683
4005_C028	ADC Result Registers with sign extension (ADC_RSLT6)	16	R/W	0000h	34.4.15/ 683
4005_C02A	ADC Result Registers with sign extension (ADC_RSLT7)	16	R/W	0000h	34.4.15/ 683
4005_C02C	ADC Result Registers with sign extension (ADC_RSLT8)	16	R/W	0000h	34.4.15/ 683
4005_C02E	ADC Result Registers with sign extension (ADC_RSLT9)	16	R/W	0000h	34.4.15/ 683
4005_C030	ADC Result Registers with sign extension (ADC_RSLT10)	16	R/W	0000h	34.4.15/ 683
4005_C032	ADC Result Registers with sign extension (ADC_RSLT11)	16	R/W	0000h	34.4.15/ 683
4005_C034	ADC Result Registers with sign extension (ADC_RSLT12)	16	R/W	0000h	34.4.15/ 683
4005_C036	ADC Result Registers with sign extension (ADC_RSLT13)	16	R/W	0000h	34.4.15/ 683
4005_C038	ADC Result Registers with sign extension (ADC_RSLT14)	16	R/W	0000h	34.4.15/ 683
4005_C03A	ADC Result Registers with sign extension (ADC_RSLT15)	16	R/W	0000h	34.4.15/ 683
4005_C03C	ADC Low Limit Registers (ADC_LOLIM0)	16	R/W	0000h	34.4.16/ 684
4005_C03E	ADC Low Limit Registers (ADC_LOLIM1)	16	R/W	0000h	34.4.16/ 684
4005_C040	ADC Low Limit Registers (ADC_LOLIM2)	16	R/W	0000h	34.4.16/ 684

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_C042	ADC Low Limit Registers (ADC_LOLIM3)	16	R/W	0000h	34.4.16/ 684
4005_C044	ADC Low Limit Registers (ADC_LOLIM4)	16	R/W	0000h	34.4.16/ 684
4005_C046	ADC Low Limit Registers (ADC_LOLIM5)	16	R/W	0000h	34.4.16/ 684
4005_C048	ADC Low Limit Registers (ADC_LOLIM6)	16	R/W	0000h	34.4.16/ 684
4005_C04A	ADC Low Limit Registers (ADC_LOLIM7)	16	R/W	0000h	34.4.16/ 684
4005_C04C	ADC Low Limit Registers (ADC_LOLIM8)	16	R/W	0000h	34.4.16/ 684
4005_C04E	ADC Low Limit Registers (ADC_LOLIM9)	16	R/W	0000h	34.4.16/ 684
4005_C050	ADC Low Limit Registers (ADC_LOLIM10)	16	R/W	0000h	34.4.16/ 684
4005_C052	ADC Low Limit Registers (ADC_LOLIM11)	16	R/W	0000h	34.4.16/ 684
4005_C054	ADC Low Limit Registers (ADC_LOLIM12)	16	R/W	0000h	34.4.16/ 684
4005_C056	ADC Low Limit Registers (ADC_LOLIM13)	16	R/W	0000h	34.4.16/ 684
4005_C058	ADC Low Limit Registers (ADC_LOLIM14)	16	R/W	0000h	34.4.16/ 684
4005_C05A	ADC Low Limit Registers (ADC_LOLIM15)	16	R/W	0000h	34.4.16/ 684
4005_C05C	ADC High Limit Registers (ADC_HILIM0)	16	R/W	7FF8h	34.4.17/ 685
4005_C05E	ADC High Limit Registers (ADC_HILIM1)	16	R/W	7FF8h	34.4.17/ 685
4005_C060	ADC High Limit Registers (ADC_HILIM2)	16	R/W	7FF8h	34.4.17/ 685
4005_C062	ADC High Limit Registers (ADC_HILIM3)	16	R/W	7FF8h	34.4.17/ 685
4005_C064	ADC High Limit Registers (ADC_HILIM4)	16	R/W	7FF8h	34.4.17/ 685
4005_C066	ADC High Limit Registers (ADC_HILIM5)	16	R/W	7FF8h	34.4.17/ 685
4005_C068	ADC High Limit Registers (ADC_HILIM6)	16	R/W	7FF8h	34.4.17/ 685
4005_C06A	ADC High Limit Registers (ADC_HILIM7)	16	R/W	7FF8h	34.4.17/ 685
4005_C06C	ADC High Limit Registers (ADC_HILIM8)	16	R/W	7FF8h	34.4.17/ 685

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_C06E	ADC High Limit Registers (ADC_HILIM9)	16	R/W	7FF8h	34.4.17/ 685
4005_C070	ADC High Limit Registers (ADC_HILIM10)	16	R/W	7FF8h	34.4.17/ 685
4005_C072	ADC High Limit Registers (ADC_HILIM11)	16	R/W	7FF8h	34.4.17/ 685
4005_C074	ADC High Limit Registers (ADC_HILIM12)	16	R/W	7FF8h	34.4.17/ 685
4005_C076	ADC High Limit Registers (ADC_HILIM13)	16	R/W	7FF8h	34.4.17/ 685
4005_C078	ADC High Limit Registers (ADC_HILIM14)	16	R/W	7FF8h	34.4.17/ 685
4005_C07A	ADC High Limit Registers (ADC_HILIM15)	16	R/W	7FF8h	34.4.17/ 685
4005_C07C	ADC Offset Registers (ADC_OFFST0)	16	R/W	0000h	34.4.18/ 685
4005_C07E	ADC Offset Registers (ADC_OFFST1)	16	R/W	0000h	34.4.18/ 685
4005_C080	ADC Offset Registers (ADC_OFFST2)	16	R/W	0000h	34.4.18/ 685
4005_C082	ADC Offset Registers (ADC_OFFST3)	16	R/W	0000h	34.4.18/ 685
4005_C084	ADC Offset Registers (ADC_OFFST4)	16	R/W	0000h	34.4.18/ 685
4005_C086	ADC Offset Registers (ADC_OFFST5)	16	R/W	0000h	34.4.18/ 685
4005_C088	ADC Offset Registers (ADC_OFFST6)	16	R/W	0000h	34.4.18/ 685
4005_C08A	ADC Offset Registers (ADC_OFFST7)	16	R/W	0000h	34.4.18/ 685
4005_C08C	ADC Offset Registers (ADC_OFFST8)	16	R/W	0000h	34.4.18/ 685
4005_C08E	ADC Offset Registers (ADC_OFFST9)	16	R/W	0000h	34.4.18/ 685
4005_C090	ADC Offset Registers (ADC_OFFST10)	16	R/W	0000h	34.4.18/ 685
4005_C092	ADC Offset Registers (ADC_OFFST11)	16	R/W	0000h	34.4.18/ 685
4005_C094	ADC Offset Registers (ADC_OFFST12)	16	R/W	0000h	34.4.18/ 685
4005_C096	ADC Offset Registers (ADC_OFFST13)	16	R/W	0000h	34.4.18/ 685
4005_C098	ADC Offset Registers (ADC_OFFST14)	16	R/W	0000h	34.4.18/ 685

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_C09A	ADC Offset Registers (ADC_OFFST15)	16	R/W	0000h	34.4.18/ 685
4005_C09C	ADC Power Control Register (ADC_PWR)	16	R/W	1DA7h	34.4.19/ 686
4005_C09E	ADC Calibration Register (ADC_CAL)	16	R/W	0000h	34.4.20/ 689
4005_C0A0	Gain Control 1 Register (ADC_GC1)	16	R/W	0000h	34.4.21/ 690
4005_C0A2	Gain Control 2 Register (ADC_GC2)	16	R/W	0000h	34.4.22/ 691
4005_C0A4	ADC Scan Control Register (ADC_SCTRL)	16	R/W	0000h	34.4.23/ 693
4005_C0A6	ADC Power Control Register (ADC_PWR2)	16	R/W	0400h	34.4.24/ 694
4005_C0A8	ADC Control Register 3 (ADC_CTRL3)	16	R/W	0000h	34.4.25/ 695
4005_C0AA	ADC Scan Interrupt Enable Register (ADC_SCHLTEN)	16	R/W	0000h	34.4.26/ 696

34.4.1 ADC Control Register 1 (ADC_CTRL1)

Bits 14, 13, 12, and 11 in CTRL1 control all types of scans except parallel scans in the B converter when CTRL2[SIMULT]=0. Non-simultaneous parallel scan modes allow independent parallel scanning in the A and B converter. Bits 14, 13, 12, and 11 in CTRL2 are used to control B converter scans in non-simultaneous parallel scan modes.

Address: 4005_C000h base + 0h offset = 4005_C000h

Bit	15	14	13	12	11	10	9	8
Read	DMAEN0	STOP0		SYNC0	EOSIE0	ZCIE	LLMTIE	HLMTIE
Write			START0					
Reset	0	1	0	1	0	0	0	0

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

ADC_CTRL1 field descriptions

Field	Description
15 DMAEN0	DMA enable When this bit is asserted, the DMA source selected by CTRL3[DMASRC] causes the conversion results to be transferred by the DMA controller. Setting this bit blocks EOSI0 interrupts generation. 0 DMA is not enabled. 1 DMA is enabled.
14 STOP0	Stop When this bit is asserted, the current scan is stopped and no further scans can start. Any further SYNC0 input pulses (see CTRL1[SYNC0] bit) or writes to the CTRL1[START0] bit are ignored until this bit has been cleared. After the ADC is in stop mode, the results registers can be modified by the processor. Any changes to the result registers in stop mode are treated as if the analog core supplied the data. Therefore, limit checking, zero crossing, and associated interrupts can occur when authorized. **This is not the same as DSP STOP mode.** 0 Normal operation 1 Stop mode
13 START0	START0 Conversion A scan is started by writing 1 to this bit. This is a write only bit. Writing 1 to it again while the scan remains in process, is ignored. The ADC must be in a stable power configuration prior to writing the start bit. Refer to the functional description of power modes for further details. 0 No action 1 Start command is issued
12 SYNC0	SYNC0 Enable A conversion may be initiated by asserting a positive edge on the SYNC0 input. Any subsequent SYNC0 input pulses while the scan remains in process are ignored unless the scan is awaiting further SYNC inputs due to the SCTRL[SCn] bits. CTRL1[SYNC0] is cleared in ONCE mode, CTRL1[SMODE=000 or 001], when the first SYNC input is detected. This prevents unintentionally starting a new scan after the first scan has completed. The ADC must be in a stable power mode prior to SYNC0 input assertion. Refer to the functional description of power modes for further details. In "once" scan modes, only a first SYNC0 input pulse is honored. CTRL1[SYNC0] is cleared in this mode when the first SYNC input is detected. This prevents unintentionally starting a new scan after the first scan has completed. The CTRL1[SYNC0] bit can be set again at any time including while the scan remains in process 0 Scan is initiated by a write to CTRL1[START0] only 1 Use a SYNC0 input pulse or CTRL1[START0] to initiate a scan
11 EOSIE0	End Of Scan Interrupt Enable This bit enables an EOSI0 interrupt to be generated upon completion of the scan. For looping scan modes, the interrupt will trigger after the completion of each iteration of the loop. 0 **Interrupt disabled** 1 **Interrupt enabled**
10 ZCIE	Zero Crossing Interrupt Enable

Table continues on the next page...

ADC_CTRL1 field descriptions (continued)

Field	Description
	This bit enables the zero crossing interrupt if the current result value has a sign change from the previous result as configured by the ZXCTRL1 and ZXCTRL2 registers. 0 **Interrupt disabled** 1 **Interrupt enabled**
9 LLMTIE	Low Limit Interrupt Enable This bit enables the Low Limit exceeded interrupt when the current result value is less than the low limit register value. The raw result value is compared to LOLIM[LLMT] before the offset register value is subtracted. 0 **Interrupt disabled** 1 **Interrupt enabled**
8 HLMTIE	High Limit Interrupt Enable This bit enables the High Limit exceeded interrupt if the current result value is greater than the high limit register value. The raw result value is compared to HILIM[HLMT] before the offset register value is subtracted. 0 **Interrupt disabled** 1 **Interrupt enabled**
7–4 CHNCFG_L	CHCNF (Channel Configure Low) bits The bits configure the analog inputs for either single ended or differential conversions. Differential measurements return the max value ((2**12)-1) when the + input is V_{REFH} and the - input is V_{REFLO} , return 0 when the + input is at V_{REFLO} and the - input is at V_{REFH} , and scale linearly between based on the voltage difference between the two signals. Single ended measurements return the max value when the input is at V_{REFH} , return 0 when the input is at V_{REFLO}, and scale linearly between based on the amount by which the input exceeds V_{REFLO}. xxx1 Inputs = ADCA_CH0-ADCA_CH1 — Configured as differential pair (ANA0 is + and ANA1 is —) xxx0 Inputs = ADCA_CH0-ADCA_CH1 — Both configured as single ended inputs xx1x Inputs = ADCA_CH2-ADCA_CH3 — Configured as differential pair (ANA2 is + and ANA3 is —) xx0x Inputs = ADCA_CH2-ADCA_CH3 — Both configured as single ended inputs x1xx Inputs = ADCB_CH0-ADCB_CH1 — Configured as differential pair (ADCB_CH0 is + and ADCB_CH1 is —) x0xx Inputs = ADCB_CH0-ADCB_CH1 — Both configured as single ended inputs 1xxx Inputs = ADCB_CH2-ADCB_CH3 — Configured as differential pair (ADCB_CH2 is + and ADCB_CH3 is —) 0xxx Inputs = ADCB_CH2-ADCB_CH3 — Both configured as single ended inputs
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SMODE	ADC Scan Mode Control This field controls the ADC module's scan mode. All scan modes use 16 sample slots defined by the CLIST1-4 registers. A scan is the process of stepping through a subset of these sample slots, converting the input indicated by a slot, and storing the result. Unused slots may be disabled using the SDIS register. Input pairs ADCA_CH0-1, ADCA_CH2-3, ADCA_CH4-5, ADCA_CH6-7, ADCB_CH0-1, ADCB_CH2-3, ADCB_CH4-5, and ADCB_CH6-7 may be configured as differential pairs using the CHNCFG fields. When a slot refers to either member of a differential pair, a differential measurement on that pair is made; otherwise, a single ended measurement is taken on that input. The CTRL*[CHNCFG] fields' descriptions

Table continues on the next page...

ADC_CTRL1 field descriptions (continued)

Field	Description
	detail differential and single ended measurement. The SMODE field determines whether the slots are used to perform one long sequential scan or two shorter parallel scans, each performed by one of the two converters. SMODE controls how these scans are initiated and terminated. It also controls whether the scans are performed once or repetitively. For details, refer to Sequential Versus Parallel Sampling and Scan Sequencing . Parallel scans may be simultaneous (CTRL2[SIMULT] is 1) or non-simultaneous. Simultaneous parallel scans perform the A and B converter scan in lock step using one set of shared controls. Non-simultaneous parallel scans operate the A and B converters independently, with each converter using its own set of controls. Refer to the CTRL2[SIMULT] bit's description for details. Setting any sequential mode overrides the setting of CTRL2[SIMULT]. 000 **Once (single) sequential** — Upon start or an enabled sync signal, samples are taken one at a time starting with CLIST1[SAMPLE0], until the first disabled sample is encountered. If no disabled sample is encountered, conversion concludes after CLIST4[SAMPLE15]. If the scan is initiated by a SYNC signal, only one scan is completed because the CTRL*[SYNC*] bit is cleared automatically by the initial SYNC detection. CTRL*[SYNC*] can be set again at any time during the scan. 001 **Once parallel** — Upon start or an armed and enabled sync signal: In parallel, converter A converts SAMPLEs 0-7, and converter B converts SAMPLEs 8-15. When CTRL2[SIMULT] is 1 (default), scanning stops when either converter encounters a disabled sample or both converters complete all 8 samples. When CTRL2[SIMULT] is 0, a converter stops scanning when it encounters a disabled sample or completes all 8 samples. If the scan is initiated by a SYNC signal, only one scan is completed because the CTRL*[SYNC*] bit is cleared automatically by the initial SYNC detection. CTRL*[SYNC*] can be set again at any time during the scan. If CTRL2[SIMULT] is 0, the B converter must be rearmed by writing the CTRL2[SYNC1] bit. 010 **Loop sequential** — Upon an initial start or enabled sync pulse, up to 16 samples in the order SAMPLEs 0-15 are taken one at a time until a disabled sample is encountered. The process repeats perpetually until the CTRL1[STOP0] bit is set. While a loop mode is running, any additional start commands or sync pulses are ignored unless the scan is paused using the SCTRL[SC*] bits. If PWR[ASB] or PWR[APD] is the selected power mode control, PWR[PUDELAY] is applied only on the first conversion. 011 **Loop parallel** — Upon an initial start or enabled sync pulse, converter A converts SAMPLEs 0-7, and converter B converts SAMPLEs 8-15. Each time a converter completes its current scan, it immediately restarts its scan sequence. This process continues until the CTRL*[STOP*] bit is asserted. While a loop is running, any additional start commands or sync pulses are ignored unless the scan is paused using the SCTRL[SC*] bits. When CTRL2[SIMULT] is 1 (default), scanning restarts when either converter encounters a disabled sample. When CTRL2[SIMULT] is 0, a converter restarts scanning when it encounters a disabled sample. If PWR[ASB] or PWR[APD] is the selected power mode control, PWR[PUDELAY] is applied only on the first conversion. 100 **Triggered sequential** — Upon start or an enabled sync signal, samples are taken one at a time starting with CLIST1[SAMPLE0], until the first disabled sample is encountered. If no disabled sample is encountered, conversion concludes after CLIST4[SAMPLE15]. If external sync is enabled, new scans start for each SYNC pulse that does not overlap with a current scan in progress. 101 **Triggered parallel (default)** — Upon start or an enabled sync signal: In parallel, converter A converts SAMPLEs 0-7, and converter B converts SAMPLEs 8-15. When CTRL2[SIMULT] is 1 (default), scanning stops when either converter encounters a disabled sample. When CTRL2[SIMULT] is 0, a converter stops scanning when it encounters a disabled sample. If external sync is enabled, new scans start for each SYNC pulse that does not overlap with a current scan in progress. 110 **Reserved** 111 **Reserved**

34.4.2 ADC Control Register 2 (ADC_CTRL2)

Address: 4005_C000h base + 2h offset = 4005_C002h

Bit	15	14	13	12	11	10	9	8
Read	DMAEN1	STOP1	0	SYNC1	EOSIE1	CHNCFG_H		
Write			START1					
Reset	0	1	0	1	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	CHNCFG_H	SIMULT	DIV0					
Write								
Reset	0	1	0	0	0	1	0	0

ADC_CTRL2 field descriptions

Field	Description
15 DMAEN1	DMA enable During parallel scan modes when SIMULT=0, this bit enables DMA for converter B. When this bit is asserted, the DMA source selected by CTRL3[DMASRC] causes the conversion results to be transferred by the DMA controller. Setting this bit blocks EOSI1 interrupts generation. 0 DMA is not enabled. 1 DMA is enabled.
14 STOP1	Stop During parallel scan modes when SIMULT = 0, this bit enables stop control of a B converter parallel scan. When this bit is asserted, the current scan is stopped and no further scans can start. Any further SYNC1 input pulses (see CTRL2[SYNC1] bit) or writes to the CTRL2[START1] bit are ignored until this bit has been cleared. After the ADC is in stop mode, the results registers can be modified by the processor. Any changes to the result registers in stop mode are treated as if the analog core supplied the data. Therefore, limit checking, zero crossing, and associated interrupts can occur when authorized. **This is not the same as DSP STOP mode.** 0 Normal operation 1 Stop mode
13 START1	START1 Conversion During parallel scan modes when SIMULT = 0, this bit enables start control of a B converter parallel scan. A scan is started by writing 1 to this bit. This is a write only bit. Writing 1 to it again while the scan remains in process, is ignored. The ADC must be in a stable power configuration prior to writing the start bit. Refer to the functional description of power modes for further details. 0 No action 1 Start command is issued
12 SYNC1	SYNC1 Enable

Table continues on the next page...

ADC_CTRL2 field descriptions (continued)

Field	Description
	During parallel scan modes when CTRL2[SIMULT]=0, setting this bit to 1 permits a B converter parallel scan to be initiated by asserting the SYNC1 input for at least one ADC clock cycle. CTRL2[SYNC1] is cleared in ONCE mode, CTRL1[SMODE=000 or 001], when the first SYNC input is detected. This prevents unintentionally starting a new scan after the first scan has completed. The ADC must be in a stable power mode prior to SYNC1 input assertion. Refer to the functional description of power modes for further details. In "once" scan modes, only a first SYNC1 input pulse is honored. CTRL2[SYNC1] is cleared in this mode when the first SYNC1 input is detected. This prevents unintentionally starting a new scan after the first scan has completed. The CTRL2[SYNC1] bit can be set again at any time including while the scan remains in process. 0 B converter parallel scan is initiated by a write to CTRL2[START1] bit only 1 Use a SYNC1 input pulse or CTRL2[START1] bit to initiate a B converter parallel scan
11 EOSIE1	End Of Scan Interrupt Enable During parallel scan modes when SIMULT = 0, this bit enables interrupt control for a B converter parallel scan. This bit enables an EOSI1 interrupt to be generated upon completion of the scan. For looping scan modes, the interrupt will trigger after the completion of each iteration of the loop. 0 **Interrupt disabled** 1 **Interrupt enabled**
10–7 CHNCFG_H	CHCNF (Channel Configure High) bits The bits configure the analog inputs for either single ended or differential conversions. Differential measurements return the max value ((2**12)-1) when the + input is V_{REFH} and the - input is V_{REFLO}, return 0 when the + input is at V_{REFLO} and the - input is at V_{REFH}, and scale linearly between based on the voltage difference between the two signals. Single ended measurements return the max value when the input is at V_{REFH}, return 0 when the input is at V_{REFLO} , and scale linearly between based on the amount by which the input exceeds V_{REFLO}. xxx1 Inputs = ADCA_CH4-ADCA_CH5 — Configured as differential pair (ADCA_CH4 is + and ADCA_CH5 is —) xxx0 Inputs = ADCA_CH4-ADCA_CH5 — Both configured as single ended inputs xx1x Inputs = ADCA_CH6-ADCA_CH7 — Configured as differential pair (ADCA_CH6 is + and ADCA_CH7 is —) xx0x Inputs = ADCA_CH6-ADCA_CH7 — Both configured as single ended inputs x1xx Inputs = ADCB_CH4-ADCB_CH5 — Configured as differential pair (ADCB_CH4 is + and ADCB_CH5 is —) x0xx Inputs = ADCB_CH4-ADCB_CH5 — Both configured as single ended inputs 1xxx Inputs = ADCB_CH6-ADCB_CH7 — Configured as differential pair (ADCB_CH6 is + and ADCB_CH7 is —) 0xxx Inputs = ADCB_CH6-ADCB_CH7 — Both configured as single ended inputs
6 SIMULT	Simultaneous mode This bit only affects parallel scan modes. By default (CTRL2[SIMULT]=1) parallel scans operate in simultaneous mode. The scans in the A and B converter operate simultaneously and always result in pairs of simultaneous conversions in the A and B converter. CTRL1[STOP0, SYNC0, and START0] control bits and the SYNC0 input are used to start and stop scans in both converters simultaneously. A scan ends in both converters when either converter encounters a disabled sample slot. When the parallel scan

Table continues on the next page...

ADC_CTRL2 field descriptions (continued)

Field	Description
	completes, the STAT[EOSI0] triggers if CTRL1[EOSIE0] is set. The STAT[CIP0] status bit indicates that a parallel scan is in process. When CTRL2[SIMULT]=0, parallel scans in the A and B converters operate independently. The B converter has its own independent set of the above controls (with a 1 suffix) which control its operation and report its status. Each converter's scan continues until its sample list is exhausted (8 samples) or a disabled sample IN ITS LIST is encountered. For looping parallel scan mode, each converter starts its next iteration as soon as the previous iteration in that converter is complete and continues until the CTRL*[STOP*] bit for that converter is asserted. 0 Parallel scans done independently 1 Parallel scans done simultaneously (default)
DIV0	Clock Divisor Select The divider circuit generates the ADC clock by dividing the system clock: • When DIV0 is 0, the divisor is 2. • For all other DIV0 values, the divisor is 1 more than the decimal value of DIV0: (DIV0)Â +Â 1d. A DIV0 value must be chosen so the ADC clock does not exceed the maximum frequency. This clock is used by ADCA during all scans and is used by ADCB during sequential scan modes and during parallel simultaneous scan modes.

34.4.3 ADC Zero Crossing Control 1 Register (ADC_ZXCTRL1)

Address: 4005_C000h base + 4h offset = 4005_C004h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	ZCE7		ZCE6		ZCE5		ZCE4		ZCE3		ZCE2		ZCE1		ZCE0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_ZXCTRL1 field descriptions

Field	Description
15–14 ZCE7	Zero crossing enable 7 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
13–12 ZCE6	Zero crossing enable 6 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
11–10 ZCE5	Zero crossing enable 5 00 **Zero Crossing disabled**

Table continues on the next page...

ADC_ZXCTRL1 field descriptions (continued)

Field	Description
	01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
9–8 ZCE4	Zero crossing enable 4 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
7–6 ZCE3	Zero crossing enable 3 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
5–4 ZCE2	Zero crossing enable 2 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
3–2 ZCE1	Zero crossing enable 1 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
ZCE0	Zero crossing enable 0 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**

34.4.4 ADC Zero Crossing Control 2 Register (ADC_ZXCTRL2)

Address: 4005_C000h base + 6h offset = 4005_C006h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	ZCE15		ZCE14		ZCE13		ZCE12		ZCE11		ZCE10		ZCE9		ZCE8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_ZXCTRL2 field descriptions

Field	Description
15–14 ZCE15	Zero crossing enable 15

Table continues on the next page...

ADC_ZXCTRL2 field descriptions (continued)

Field	Description
	00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
13–12 ZCE14	Zero crossing enable 14 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
11–10 ZCE13	Zero crossing enable 13 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
9–8 ZCE12	Zero crossing enable 12 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
7–6 ZCE11	Zero crossing enable 11 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
5–4 ZCE10	Zero crossing enable 10 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
3–2 ZCE9	Zero crossing enable 9 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**
ZCE8	Zero crossing enable 8 00 **Zero Crossing disabled** 01 **Zero Crossing enabled for positive to negative sign change** 10 **Zero Crossing enabled for negative to positive sign change** 11 **Zero Crossing enabled for any sign change**

34.4.5 ADC Channel List Register 1 (ADC_CLIST1)

Address: 4005_C000h base + 8h offset = 4005_C008h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SAMPLE3				SAMPLE2				SAMPLE1				SAMPLE0			
Reset	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

ADC_CLIST1 field descriptions

Field	Description
15–12 SAMPLE3	Sample Field 3 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
11–8 SAMPLE2	Sample Field 2 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**

Table continues on the next page...

ADC_CLIST1 field descriptions (continued)

Field	Description
7–4 SAMPLE1	Sample Field 1 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
SAMPLE0	Sample Field 0 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**

34.4.6 ADC Channel List Register 2 (ADC_CLIST2)

Address: 4005_C000h base + Ah offset = 4005_C00Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SAMPLE7				SAMPLE6				SAMPLE5				SAMPLE4			
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0

ADC_CLIST2 field descriptions

Field	Description
15–12 SAMPLE7	Sample Field 7 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
11–8 SAMPLE6	Sample Field 6 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
7–4 SAMPLE5	Sample Field 5 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-**

Table continues on the next page...

ADC_CLIST2 field descriptions (continued)

Field	Description
	1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
SAMPLE4	Sample Field 4 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**

34.4.7 ADC Channel List Register 3 (ADC_CLIST3)

Address: 4005_C000h base + Ch offset = 4005_C00Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SAMPLE11				SAMPLE10				SAMPLE9				SAMPLE8			
Reset	1	0	1	1	1	0	1	0	1	0	0	1	1	0	0	0

ADC_CLIST3 field descriptions

Field	Description
15–12 SAMPLE11	Sample Field 11 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-**

Table continues on the next page...

ADC_CLIST3 field descriptions (continued)

Field	Description
	1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
11–8 SAMPLE10	Sample Field 10 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
7–4 SAMPLE9	Sample Field 9 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
SAMPLE8	Sample Field 8 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-**

Table continues on the next page...

ADC_CLIST3 field descriptions (continued)

Field	Description
	0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-**
	0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-**
	0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-**
	0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-**
	0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-**
	0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-**
	0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-**
	1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-**
	1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-**
	1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-**
	1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-**
	1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-**
	1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-**
	1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-**
	1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**

34.4.8 ADC Channel List Register 4 (ADC_CLIST4)

Address: 4005_C000h base + Eh offset = 4005_C00Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SAMPLE15				SAMPLE14				SAMPLE13				SAMPLE12			
Reset	1	1	1	1	1	1	1	0	1	1	0	1	1	1	0	0

ADC_CLIST4 field descriptions

Field	Description
15–12 SAMPLE15	Sample Field 15
	0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-**
	0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-**
	0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-**
	0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-**
	0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-**
	0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-**
	0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-**
	0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-**
	1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-**
	1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-**
	1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-**
	1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-**
	1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-**
	1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-**
	1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-**
	1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**

Table continues on the next page...

ADC_CLIST4 field descriptions (continued)

Field	Description
11–8 SAMPLE14	Sample Field 14 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
7–4 SAMPLE13	Sample Field 13 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-** 1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**
SAMPLE12	Sample Field 12 0000 **Single Ended: ADCA_CH0, Differential: ADCA_CH0+, ADCA_CH1-** 0001 **Single Ended: ADCA_CH1, Differential: ADCA_CH0+, ADCA_CH1-** 0010 **Single Ended: ADCA_CH2, Differential: ADCA_CH2+, ADCA_CH3-** 0011 **Single Ended: ADCA_CH3, Differential: ADCA_CH2+, ADCA_CH3-** 0100 **Single Ended: ADCA_CH4, Differential: ADCA_CH4+, ADCA_CH5-** 0101 **Single Ended: ADCA_CH5, Differential: ADCA_CH4+, ADCA_CH5-** 0110 **Single Ended: ADCA_CH6, Differential: ADCA_CH6+, ADCA_CH7-** 0111 **Single Ended: ADCA_CH7, Differential: ADCA_CH6+, ADCA_CH7-** 1000 **Single Ended: ADCB_CH0, Differential: ADCB_CH0+, ADCB_CH1-**

Table continues on the next page...

ADC_CLIST4 field descriptions (continued)

Field	Description
	1001 **Single Ended: ADCB_CH1, Differential: ADCB_CH0+, ADCB_CH1-** 1010 **Single Ended: ADCB_CH2, Differential: ADCB_CH2+, ADCB_CH3-** 1011 **Single Ended: ADCB_CH3, Differential: ADCB_CH2+, ADCB_CH3-** 1100 **Single Ended: ADCB_CH4, Differential: ADCB_CH4+, ADCB_CH5-** 1101 **Single Ended: ADCB_CH5, Differential: ADCB_CH4+, ADCB_CH5-** 1110 **Single Ended: ADCB_CH6, Differential: ADCB_CH6+, ADCB_CH7-** 1111 **Single Ended: ADCB_CH7, Differential: ADCB_CH6+, ADCB_CH7-**

34.4.9 ADC Sample Disable Register (ADC_SDIS)

Each bit of ADC_SDIS corresponds to a SAMPLEx filed in one or multiple ADC_CLISTn registers. For example, bit 0 of ADC_SDIS enables/disables CLIST0 and bit 15 of ADC_SDIS enables/disables CLIST15.

If an ADC_SDIS bit is clear, then the correspopnding SAMPLEx field is enabled in an ADC scan. If the ADC_SDIS bit is set, then the corresponding SAMPLEx field is disabled and the ADC scan is halted and subsequent ADC acquisition also does not occur. Remembering that the ADC sequentially scans in order SAMPLE0, SAMPLE1, SAMPLE2, etc thus if ADC_SDIS = 0xFFF7, then an ADC scan acquires SAMPLE0 while SAMPLE1, SAMPLE2, and all other acquistions do not occur. The 4-bit wide SAMPLEx fields are found in the ADC_CLISTn registers which select a particular ADC channel and whether it is a differential or single ended conversion.

Address: 4005_C000h base + 10h offset = 4005_C010h

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

ADC_SDIS field descriptions

Field	Description
DS	Disable Sample Bits **Note:** SAMPLEx resides in one of the ADC_CLISTn registers, SAMPLE0-3 inADC_CLIST1 register, SAMPLE4-7 in ADC_CLIST2 register, etc. 0 SAMPLEx channel is enabled for ADC scan. 1 SAMPLEx channel is disabled for ADC scan and corresponding channels after SAMPLEx also doesn not occur in an ADC scan.

34.4.10 ADC Status Register (ADC_STAT)

This register provides the current status of the ADC module. STAT[HLMTI and LLMTI] bits are cleared by writing 1s to all asserted bits in the limit status register, LIMSTAT. Likewise, the STAT[ZCI] bit, is cleared by writing 1s to all asserted bits in the zero crossing status register, ZXSTAT. The STAT[EOSIx] bits are cleared by writing a one to them.

Except for STAT[CIP0 and CIP1] this register's bits are sticky. Once set to a one state, they require some specific action to clear them. They are not cleared automatically on the next scan sequence.

Address: 4005_C000h base + 12h offset = 4005_C012h

Bit	15	14	13	12	11	10	9	8
Read	CIP0	CIP1	0	EOSI1	EOSI0	ZCI	LLMTI	HLMTI
Write				w1c	w1c			
Reset	0	0	0	0	0	0	0	0

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

ADC_STAT field descriptions

Field	Description
15 CIP0	Conversion in Progress This bit indicates whether a scan is in progress. This refers to any scan except a B converter scan in non-simultaneous parallel scan modes. 0 Idle state 1 A scan cycle is in progress. The ADC will ignore all sync pulses or start commands
14 CIP1	Conversion in Progress This bit indicates whether a scan is in progress. This refers only to a B converter scan in non-simultaneous parallel scan modes. 0 Idle state 1 A scan cycle is in progress. The ADC will ignore all sync pulses or start commands
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 EOSI1	End of Scan Interrupt This bit indicates whether a scan of analog inputs have been completed since the last read of the status register or since a reset. This bit is cleared by writing a one to it. This bit cannot be set by software. In looping scan modes, this interrupt is triggered at the completion of each iteration of the loop.

Table continues on the next page...

ADC_STAT field descriptions (continued)

Field	Description
	This interrupt is triggered only by the completion of a B converter scan in non-simultaneous parallel scan modes. 0 A scan cycle has not been completed, no end of scan IRQ pending 1 A scan cycle has been completed, end of scan IRQ pending
11 EOSI0	End of Scan Interrupt This bit indicates whether a scan of analog inputs have been completed since the last read of the status register or since a reset. This bit is cleared by writing a one to it. This bit cannot be set by software. STAT[EOSI0] is the preferred bit to poll for scan completion if interrupts are not enabled. In looping scan modes, this interrupt is triggered at the completion of each iteration of the loop. This interrupt is triggered upon the completion of any scan except for the completion of a B converter scan in non-simultaneous parallel scan modes. 0 A scan cycle has not been completed, no end of scan IRQ pending 1 A scan cycle has been completed, end of scan IRQ pending
10 ZCI	Zero Crossing Interrupt If the respective offset register is configured by having a value greater than 0000h, zero crossing checking is enabled. If the offset register is programmed with 7FF8h, the result will always be less than or equal to zero. On the other hand, if 0000h is programmed into the offset register, the result will always be greater than or equal to zero, and no zero crossing can occur because the sign of the result will not change. This interrupt asserts at the completion of an individual conversion which may or may not be the end of a scan. This bit is cleared by writing a "1" to all active ZXSTAT[ZCS] bits. 0 No zero crossing interrupt request 1 Zero crossing encountered, IRQ pending if CTRL1[ZCIE] is set
9 LLMTI	Low Limit Interrupt If the respective low limit register is enabled by having a value other than 0000h, low limit checking is enabled. This interrupt asserts at the completion of an individual conversion which may or may not be the end of a scan. This bit is cleared by writing a "1" to all active LIMSTAT[LLS] bits. 0 No low limit interrupt request 1 Low limit exceeded, IRQ pending if CTRL1[LLMTIE] is set
8 HLMTI	High Limit Interrupt If the respective high limit register is enabled by having a value other than 7FF8h, high limit checking is enabled. This interrupt asserts at the completion of an individual conversion which may or may not be the end of a scan. This bit is cleared by writing a "1" to all active LIMSTAT[HLS] bits. 0 No high limit interrupt request 1 High limit exceeded, IRQ pending if CTRL1[HLMTIE] is set
UNDEFINED	This read-only bitfield is undefined and will always contain random data.

34.4.11 ADC Ready Register (ADC_RDY)

This register provides the current status of the ADC conversions. RDY[RDYx] bits are cleared by reading their corresponding result registers (RSLTx).

Address: 4005_C000h base + 14h offset = 4005_C014h

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

ADC_RDY field descriptions

Field	Description
RDY[15:0]	Ready Sample These bits indicate samples fifteen through zero are ready to be read. These bits are cleared after a read from the respective results register. The RDY[RDYn] bits are set as the individual channel conversions are completed. Polling the RDY[RDYn] bits can determine if a particular sample is ready to be read. 0 Sample not ready or has been read 1 Sample ready to be read

34.4.12 ADC Low Limit Status Register (ADC_LOLIMSTAT)

The low limit status register latches in the result of the comparison between the result of the sample and the respective low limit register, LOLIM0-15. Here is an example: If the result for the channel programmed in CLIST1[SAMPLE0] is lower than the value programmed into the Low Limit Register zero, then the LIMSTAT[LLS0] bit is set to one. An interrupt is generated if the CTRL1[LLMTIE] bit is set. These bits are sticky. They are not cleared automatically by subsequent conversions. Each bit is cleared only by writing a value of one to that specific bit.

Address: 4005_C000h base + 16h offset = 4005_C016h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	LLS[15:0]															
Write	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_LOLIMSTAT field descriptions

Field	Description
LLS[15:0]	Low Limit Status Bits

34.4.13 ADC High Limit Status Register (ADC_HILIMSTAT)

The high limit status register latches in the result of the comparison between the result of the sample and the respective high limit register, HILIM0-15. Here is an example: If the result for the channel programmed in CLIST1[SAMPLE0] is greater than the value programmed into the High Limit Register zero, then the LIMSTAT[HLS0] bit is set to one. An interrupt is generated if the CTRL1[HLMTIE] bit is set. These bits are sticky. They are not cleared automatically by subsequent conversions. Each bit is cleared only by writing a value of one to that specific bit.

Address: 4005_C000h base + 18h offset = 4005_C018h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	HLS[15:0]															
Write	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_HILIMSTAT field descriptions

Field	Description
HLS[15:0]	High Limit Status Bits

34.4.14 ADC Zero Crossing Status Register (ADC_ZXSTAT)

Address: 4005_C000h base + 1Ah offset = 4005_C01Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ZCS[15:0]															
Write	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_ZXSTAT field descriptions

Field	Description
ZCS[15:0]	Zero Crossing Status The zero crossing condition is determined by examining the ADC value after it has been adjusted by the offset for the result register. Each bit of the register is cleared by writing a one to that register bit.

ADC_ZXSTAT field descriptions (continued)

Field	Description
	0 Either: • A sign change did not occur in a comparison between the current channelx result and the previous channelx result, or • Zero crossing control is disabled for channelx in the zero crossing control register, ZXCTRL 1 In a comparison between the current channelx result and the previous channelx result, a sign change condition occurred as defined in the zero crossing control register (ZXCTRL)

34.4.15 ADC Result Registers with sign extension (ADC_RSLT*n*)

The result registers contain the converted results from a scan. The CLIST1[SAMPLE0] result is loaded into RSLT0, CLIST1[SAMPLE1] result in RSLT1, and so on. In a parallel scan mode, the first channel pair designated by CLIST1[SAMPLE0] and CLIST3[SAMPLE8] are stored in RSLT0 and RSLT8, respectively.

Note

When writing to this register, only the RSLT portion of the value written is used. This value is modified and the result of the subtraction is stored. The SEXT bit is only set as a result of this subtraction and is not directly determined by the value written.

Address: 4005_C000h base + 1Ch offset + (2d × i), where i=0d to 15d

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

ADC_RSLT*n* field descriptions

Field	Description
15 SEXT	Sign Extend This is the sign-extend bit of the result. RSLT*[SEXT] set to one implies a negative result; RSLT*[SEXT] set to zero implies a positive result. If **unsigned** results are required, then the respective offset register must be set to a value of zero.
14–3 RSLT	Digital Result of the Conversion RSLT can be interpreted as either a signed integer or a signed fractional number. As a signed fractional number, the RSLT can be used directly. As a signed integer, it is an option to right shift with sign extend (ASR) three places and interpret the number, or accept the number as presented, knowing there are missing codes. The lower three bits are always going to be zero. Negative results, RSLT*[SEXT] = 1, are always presented in two's complement format. If it is a requirement of your application that the result registers always be positive, the offset registers must always be set to zero.

Table continues on the next page...

ADC_RSLT*n* field descriptions (continued)

Field	Description
	The interpretation of the numbers programmed into the limit and offset registers, LOLIM, HILIM, and OFFST should match your interpretation of the result register.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

34.4.16 ADC Low Limit Registers (ADC_LOLIM*n*)

Each ADC sample is compared against the values in the limit registers. The comparison is based upon the raw conversion value with no offset correction applied. The limit register used corresponds to the result register the value will be written to. The high limit register is used for the comparison of Result > High Limit. The low limit register is used for the comparison of Result < Low Limit. The limit checking can be disabled by programming the respective limit register with 7FF8h for the high limit and 0000h for the low limit. At reset, limit checking is disabled.

Address: 4005_C000h base + 3Ch offset + (2d × i), where i=0d to 15d

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

ADC_LOLIM*n* field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–3 LLMT	Low Limit Bits
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

34.4.17 ADC High Limit Registers (ADC_HILIM*n*)

Each ADC sample is compared against the values in the limit registers. The comparison is based upon the raw conversion value with no offset correction applied. The limit register used corresponds to the result register the value will be written to. The high limit register is used for the comparison of Result > High Limit. The low limit register is used for the comparison of Result < Low Limit. The limit checking can be disabled by programming the respective limit register with 7FF8h for the high limit and 0000h for the low limit. At reset, limit checking is disabled.

Address: 4005_C000h base + 5Ch offset + (2d × i), where i=0d to 15d

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

ADC_HILIM*n* field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–3 HLMT	High Limit Bits
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

34.4.18 ADC Offset Registers (ADC_OFFST*n*)

The value of the offset register is used to correct the ADC result before it is stored in the RSLT registers.

The offset value is subtracted from the ADC result. To obtain unsigned results, program the respective offset register with a value of $0000, thus giving a result range of $0000 to $7FF8.

Address: 4005_C000h base + 7Ch offset + (2d × i), where i=0d to 15d

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

ADC_OFFST*n* field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14–3 OFFSET	ADC Offset Bits
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

34.4.19 ADC Power Control Register (ADC_PWR)

This register controls the power management features of the ADC module. There are individual manual power down controls for the two ADC converters and the voltage reference generators. There are also five distinct power modes. The following terms are used to describe power modes and their related controls.

Power down state	Each converter and voltage reference generator can individually be put into a power down state. When powered down, the unit consumes no power. Results of scans referencing a powered down converter are undefined. At least one converter must be powered up to use the ADC module.
Manual power down controls	Each converter and voltage reference generator have a manual power control bit capable of putting that component into the power down state. Converters have other mechanisms that can automatically put them into the power down state.
Idle state	The ADC module is idle when neither of the two converters has a scan in process.
Active state	The ADC module is active when at least one of the two converters has a scan in process.
Current Mode	Both converters share a common current mode. Normal current mode is used to power the converters at clock rates above 600kHz. Current mode does not affect the number of ADC clock cycles required to do a conversion or the accuracy of a conversion. The ADC module may change the current mode when idle as part of the power saving strategy.
Startup delay	Auto-powerdown and auto-standby power modes cause a startup delay when the ADC module goes between the idle and active states to allow time to switch clocks or power configurations.

Address: 4005_C000h base + 9Ch offset = 4005_C09Ch

Bit	15	14	13	12	11	10	9	8
Read	ASB	0		1	PSTS1	PSTS0	PUDELAY	
Write								
Reset	0	0	0	1	1	1	0	1

Bit	7	6	5	4	3	2	1	0
Read	PUDELAY				APD	1	PD1	PD0
Write								
Reset	1	0	1	0	0	1	1	1

ADC_PWR field descriptions

Field	Description
15 ASB	Auto Standby This bit selects auto-standby mode. PWR[ASB] is ignored if PWR[APD] is 1. When the ADC is idle, auto-standby mode selects the standby clock as the ADC clock source and puts the converters into standby current mode. At the start of any scan, the conversion clock is selected as the ADC clock and then a delay of PWR[PUDELAY] ADC clock cycles is imposed for current levels to stabilize. After this delay, the ADC will initiate the scan. When the ADC returns to the idle state, the standby clock is again selected and the converters revert to the standby current state. This mode is not recommended for conversion clock rates at or below 100kHz. Instead, set PWR[ASB and APD]=0 and use standby power mode (normal mode with a sufficiently slow conversion clock so that standby current mode automatically engages). This provides the advantages of standby current mode while avoiding the clock switching and the PWR[PUDELAY]. Set PWR[ASB] prior to clearning PWR[PD1/0]. 0 Auto standby mode disabled 1 Auto standby mode enabled
14–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
11 PSTS1	ADC Converter B Power Status This bit is asserted immediately following a write of "1" to PWR[PD1]. It is de-asserted PWR[PUDELAY] ADC clock cycles after a write of "0" to PWR[PD1] if PWR[APD] is "0". This bit can be read as a status bit to determine when the ADC is ready for operation. During auto-powerdown mode, this bit indicates the current powered state of converter B. 0 ADC Converter B is currently powered up 1 ADC Converter B is currently powered down
10 PSTS0	ADC Converter A Power Status This bit is asserted immediately following a write of "1" to PWR[PD0]. It is de-asserted PWR[PUDELAY] ADC clock cycles after a write of "0" to PWR[PD0] if PWR[APD] is "0". This bit can be read as a status bit to determine when the ADC is ready for operation. During auto-powerdown mode, this bit indicates the current powered state of converter A. 0 ADC Converter A is currently powered up 1 ADC Converter A is currently powered down
9–4 PUDELAY	Power Up Delay This 6-bit field determines the number of ADC clocks provided to power up an ADC converter (after setting PWR[PD0 or PD1] to 0) before allowing a scan to start. It also determines the number of ADC clocks of delay provided in auto-powerdown (APD) and auto-standby (ASB) modes between when the ADC goes from the idle to active state and when the scan is allowed to start. The default value is 26 ADC clocks. Accuracy of the initial conversions in a scan will be degraded if PWR[PUDELAY] is set to too small a value.

Table continues on the next page...

ADC_PWR field descriptions (continued)

Field	Description
	NOTE: PWR[PUDELAY] defaults to a value that is typically sufficient for any power mode. The latency of a scan can be reduced by reducing PWR[PUDELAY] to the lowest value for which accuracy is not degraded. Refer to the data sheet for further details.
3 APD	Auto Powerdown Auto-powerdown mode powers down converters when not in use for a scan. PWR[APD] takes precedence over PWR[ASB]. When a scan is started in PWR[APD] mode, a delay of PWR[PUDELAY] ADC clock cycles is imposed during which the needed converter(s), if idle, are powered up. The ADC will then initiate a scan equivalent to that done when PWR[APD] is not active. When the scan is completed, the converter(s) are powered down again. **NOTE:** If PWR[ASB or APD] is asserted while a scan is in progress, that scan is unaffected and the ADC will wait to enter its low power state until after all conversions are complete and both ADCs are idle. PWR[ASB and APD] are not useful in looping modes. The continuous nature of scanning means that the low power state can never be entered. 0 Auto Powerdown Mode is not active 1 Auto Powerdown Mode is active
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
1 PD1	Manual Power Down for Converter B This bit forces ADC converter B to power down. Asserting this bit powers down converter B immediately. The results of a scan using converter B will be invalid while PWR[PD1] is asserted. When PWR[PD1] is cleared, converter B is either continuously powered up (PWR[APD] = 0) or automatically powered up when needed (PWR[APD]=1). When clearing this bit in any power mode except auto-powerdown (PWR[APD]=1), wait PWR[PUDELAY] ADC clock cycles before initiating a scan to stabilize power levels within the converter. The PWR[PSTS1] bit can be polled to determine when the PWR[PUDELAY] time has elapsed. Failure to follow this procedure can result in loss of accuracy of the first two samples. 0 Power Up ADC converter B 1 Power Down ADC converter B
0 PD0	Manual Power Down for Converter A This bit forces ADC converter A to power down. Asserting this bit powers down converter A immediately. The results of a scan using converter A will be invalid while PWR[PD0] is asserted. When PWR[PD0] is cleared, converter A is either continuously powered up (PWR[APD] = 0) or automatically powered up when needed (PWR[APD]=1). When clearing this bit in any power mode except auto-powerdown (PWR[APD]=1), wait PWR[PUDELAY] ADC clock cycles before initiating a scan to stabilize power levels within the converter. The PWR[PSTS0] bit can be polled to determine when the PWR[PUDELAY] time has elapsed. Failure to follow this procedure can result in loss of accuracy of the first two samples. 0 Power Up ADC converter A 1 Power Down ADC converter A

34.4.20 ADC Calibration Register (ADC_CAL)

The ADC provides for off-chip references that can be used for ADC conversions.

Address: 4005_C000h base + 9Eh offset = 4005_C09Eh

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

ADC_CAL field descriptions

Field	Description
15 SEL_VREFH_B	Select V_{REFH} Source This bit selects the source of the V_{REFH} reference for all conversions in converter 1. 0 V_{REFH} pad 1 ADCB_CH2
14 SEL_VREFLO_B	Select V_{REFLO} Source This bit selects the source of the V_{REFLO} reference for all conversions in converter 1. 0 V_{REFL} pad 1 ADCB_CH3
13 SEL_VREFH_A	Select V_{REFH} Source This bit selects the source of the V_{REFH} reference for all conversions in converter 0. 0 V_{REFH} pad 1 ADCA_CH2
12 SEL_VREFLO_A	Select V_{REFLO} Source This bit selects the source of the V_{REFLO} reference for all conversions in converter 0. 0 V_{REFL} pad 1 ADCA_CH3
11–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

34.4.21 Gain Control 1 Register (ADC_GC1)

The gain control registers are used to control amplification of each of the 16 input channels. GAIN0-GAIN7 control the amplification of inputs ADCA_CH0-ADCA_CH7 while GAIN8-GAIN15 control the amplification of inputs ADCB_CH0-ADCB_CH7.

Address: 4005_C000h base + A0h offset = 4005_C0A0h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	GAIN7		GAIN6		GAIN5		GAIN4		GAIN3		GAIN2		GAIN1		GAIN0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_GC1 field descriptions

Field	Description
15–14 GAIN7	Gain Control Bit 7 GAIN 7 controls ADCA_CH7 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
13–12 GAIN6	Gain Control Bit 6 GAIN 6 controls ADCA_CH6 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
11–10 GAIN5	Gain Control Bit 5 GAIN 5 controls ADCA_CH5 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
9–8 GAIN4	Gain Control Bit 4 GAIN 4 controls ADCA_CH4 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved

Table continues on the next page...

ADC_GC1 field descriptions (continued)

Field	Description
7–6 GAIN3	Gain Control Bit 3 GAIN 3 controls ADCA_CH3 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
5–4 GAIN2	Gain Control Bit 2 GAIN 2 controls ADCA_CH2 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
3–2 GAIN1	Gain Control Bit 1 GAIN 1 controls ADCA_CH1 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
GAIN0	Gain Control Bit 0 GAIN 0 controls ADCA_CH0 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved

34.4.22 Gain Control 2 Register (ADC_GC2)

The gain control registers are used to control amplification of each of the 16 input channels. GAIN0-GAIN7 control the amplification of inputs ADCA_CH0-ADCA_CH7 while GAIN8-GAIN15 control the amplification of inputs ADCB_CH0-ADCB_CH7.

Address: 4005_C000h base + A2h offset = 4005_C0A2h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	GAIN15		GAIN14		GAIN13		GAIN12		GAIN11		GAIN10		GAIN9		GAIN8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_GC2 field descriptions

Field	Description
15–14 GAIN15	Gain Control Bit 15 GAIN 15 controls ADCB_CH7 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
13–12 GAIN14	Gain Control Bit 14 GAIN 14 controls ADCB_CH6 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
11–10 GAIN13	Gain Control Bit 13 GAIN 13 controls ADCB_CH5 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
9–8 GAIN12	Gain Control Bit 12 GAIN 12 controls ADCB_CH4 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
7–6 GAIN11	Gain Control Bit 11 GAIN 11 controls ADCB_CH3 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
5–4 GAIN10	Gain Control Bit 10 GAIN 10 controls ADCB_CH2 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
3–2 GAIN9	Gain Control Bit 9 GAIN 9 controls ADCB_CH1

Table continues on the next page...

ADC_GC2 field descriptions (continued)

Field	Description
	00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved
GAIN8	Gain Control Bit 8 GAIN 8 controls ADCB_CH0 00 x1 amplification 01 x2 amplification 10 x4 amplification 11 reserved

34.4.23 ADC Scan Control Register (ADC_SCTRL)

This register is an extension to the CLIST1-4 registers, providing the ability to pause and await a new sync while processing samples programmed in the CLIST*[SAMPLE0–SAMPLE15] fields.

These 16 control bits are used to determine whether a sample in a scan occurs immediately or if the sample waits for an enabled sync input to occur. The sync input must occur after the conversion of the current sample completes. During sequential mode scans, the SCTRL[SC] bits are used in order from SC0 to SC15. During simultaneous parallel scan modes, the bits are used in order from SC0 to SC3 and SC8 to SC11. In non-simultaneous parallel scans, ADCA uses the bits in order from SC0 to SC3 followed by SC8 to SC11. ADCB will use bits SC4 to SC7 followed by SC12 to SC15 in non-simultaneous parallel scans.

When setting SCTRL[SC0], don't set CTRL1[START0] or CTRL2[START1]. Just clear CTRL1[STOP0] or CTRL2[STOP1] and the first enabled sync input will start the scan.

Setting SC0 delays sample 0 until a sync pulse occurs. Setting SC1 delays sample 1 until a sync pulse occurs after completing sample 0.

Address: 4005_C000h base + A4h offset = 4005_C0A4h

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

ADC_SCTRL field descriptions

Field	Description
SC[15:0]	Scan Control Bits

ADC_SCTRL field descriptions (continued)

Field	Description
	0 Perform sample immediately after the completion of the current sample. 1 Delay sample until a new sync input occurs.

34.4.24 ADC Power Control Register (ADC_PWR2)

Address: 4005_C000h base + A6h offset = 4005_C0A6h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0		DIV1						0				SPEEDB		SPEEDA	
Write																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

ADC_PWR2 field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–8 DIV1	Clock Divisor Select The divider circuit operates in the same manner as the CTRL2[DIV0] field but is used to generate the clock used by ADCB during parallel non-simultaneous scan modes.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 SPEEDB	ADCB Speed Control Bits These bits configure the clock speed at which the ADCB can operate. Faster conversion speeds require greater current consumption. 00 Conversion clock frequency ≤ 6.25 MHz; current consumption per converter = 6 mA 01 Conversion clock frequency ≤ 12.5 MHz; current consumption per converter = 10.8 mA 10 Conversion clock frequency ≤ 18.75 MHz; current consumption per converter = 18 mA 11 Conversion clock frequency ≤ 25 MHz; current consumption per converter = 25.2 mA
SPEEDA	ADCA Speed Control Bits These bits configure the clock speed at which the ADCA can operate. Faster conversion speeds require greater current consumption. 00 Conversion clock frequency ≤ 6.25 MHz; current consumption per converter = 6 mA 01 Conversion clock frequency ≤ 12.5 MHz; current consumption per converter = 10.8 mA 10 Conversion clock frequency ≤ 18.75 MHz; current consumption per converter = 18 mA 11 Conversion clock frequency ≤ 25 MHz; current consumption per converter = 25.2 mA

34.4.25 ADC Control Register 3 (ADC_CTRL3)

Address: 4005_C000h base + A8h offset = 4005_C0A8h

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0	DMASRC	SCNT1[2:0]			SCNT0[2:0]		
Write								
Reset	0	0	0	0	0	0	0	0

ADC_CTRL3 field descriptions

Field	Description
15–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DMASRC	DMA Trigger Source During sequential and simultaneous parallel scan modes CTRL3[DMASRC] selects between EOSI0 and RDY bits as the DMA source. During non-simultaneous parallel scan mode CTRL3[DMASRC] selects between EOSI0/EOSI1 for converters A and B, and the RDY bits as the DMA source. 0 DMA trigger source is end of scan interrupt 1 DMA trigger source is RDY bits
5–3 SCNT1[2:0]	Sample Window Count 1 During parallel non-simultaneous scan mode (CTRL2[SIMULT]=0) the CTRL3[SCNT1] bits are used to control the sampling time of the first sample after a scan is initiated on converter B. During sequential and parallel simultaneous scan modes, CTRL3[SCNT1] is ignored and the sampling window for converters A and B is controlled by CTRL3[SCNT0]. The default value is 0 which corresponds to a sampling time of 2 ADC clocks. Each increment of CTRL3[SCNT1] corresponds to an additional ADC clock cycle of sampling time with a maximum sampling time of 9 ADC clocks.
SCNT0[2:0]	Sample Window Count 0 During sequential and parallel simultaneous scan modes (CTRL2[SIMULT]=1) the CTRL3[SCNT0] bits control the sampling time of the first sample after a scan is initiated on both converters A and B. In parallel non-simultaneous mode (CTRL2[SIMULT]=0) CTRL3[SCNT0] affects converter A only. The default value is 0 which corresponds to a sampling time of 2 ADC clocks. Each increment of CTRL3[SCNT0] corresponds to an additional ADC clock cycle of sampling time with a maximum sampling time of 9 ADC clocks. In sequential scan mode the CTRL[SCNT0] setting will be ignored whenever the channel selected for the next sample is on the other converter. In other words, during a sequential scan, if a sample converts a converter A channel (ADCA_CH0-ADCA_CH7) and the next sample converts a converter B channel (ADCB_CH0-ADCB_CH7) or vice versa, CTRL3[SCNT0] will be ignored and use the default sampling time for the next sample.

34.4.26 ADC Scan Interrupt Enable Register (ADC_SCHLTEN)

This register is used with ready register (RDY) to select the samples that will generate a scan interrupt.

Address: 4005_C000h base + AAh offset = 4005_C0AAh

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

ADC_SCHLTEN field descriptions

Field	Description
SCHLTEN[15:0]	Scan Interrupt Enable 0 Scan interrupt is not enabled for this sample. 1 Scan interrupt is enabled for this sample.

34.5 Functional Description

The ADC consists of two eight-channel input select functions, which are two independent sample and hold (S/H) circuits feeding two separate 12-bit ADCs. The two separate converters store their results in an accessible buffer, awaiting further processing.

The conversion process is initiated either by a SYNC signal or by writing a 1 to a START bit.

Starting a single conversion actually begins a sequence of conversions, or a scan. The ADC operates in either sequential scan mode or parallel scan mode. In sequential scan mode, scan sequence is determined by defining sixteen sample slots that are processed in order, SAMPLE[0:15]. In parallel scan mode, converter A processes SAMPLE[0:7] in order, and converter B processes SAMPLE[8:15] in order. SAMPLE slots can be disabled using the ADC_SDIS control register to terminate a scan early.

In sequential scan mode, a scan takes up to sixteen single-ended or differential samples, one at a time. See the following figure.

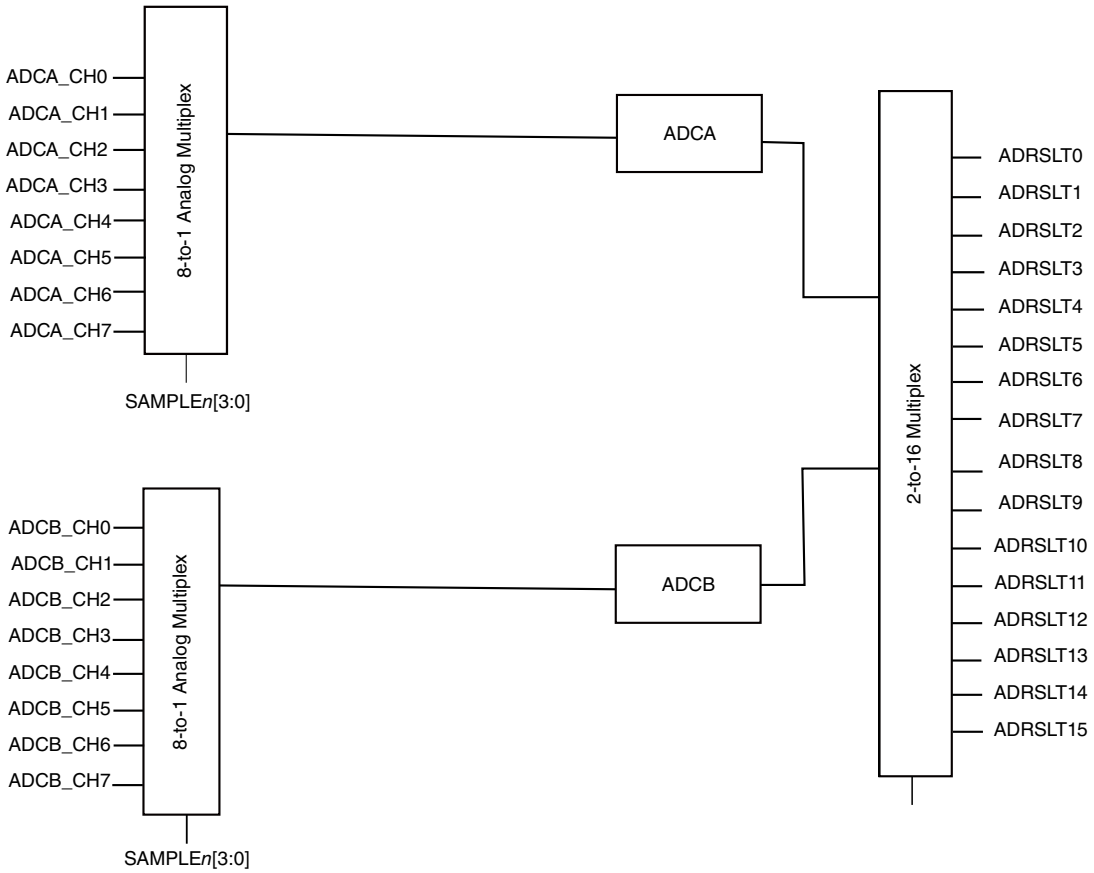


Figure 34-4. ADC Sequential Scan Mode

In parallel scan mode, eight of the sixteen samples are allocated to converter A and eight are allocated to converter B. Two converters operate in parallel, and each can take at most eight samples. Converter A can sample only analog inputs ADCA_CH[0:7], and converter B can sample only analog inputs ADCB_CH[0:7].

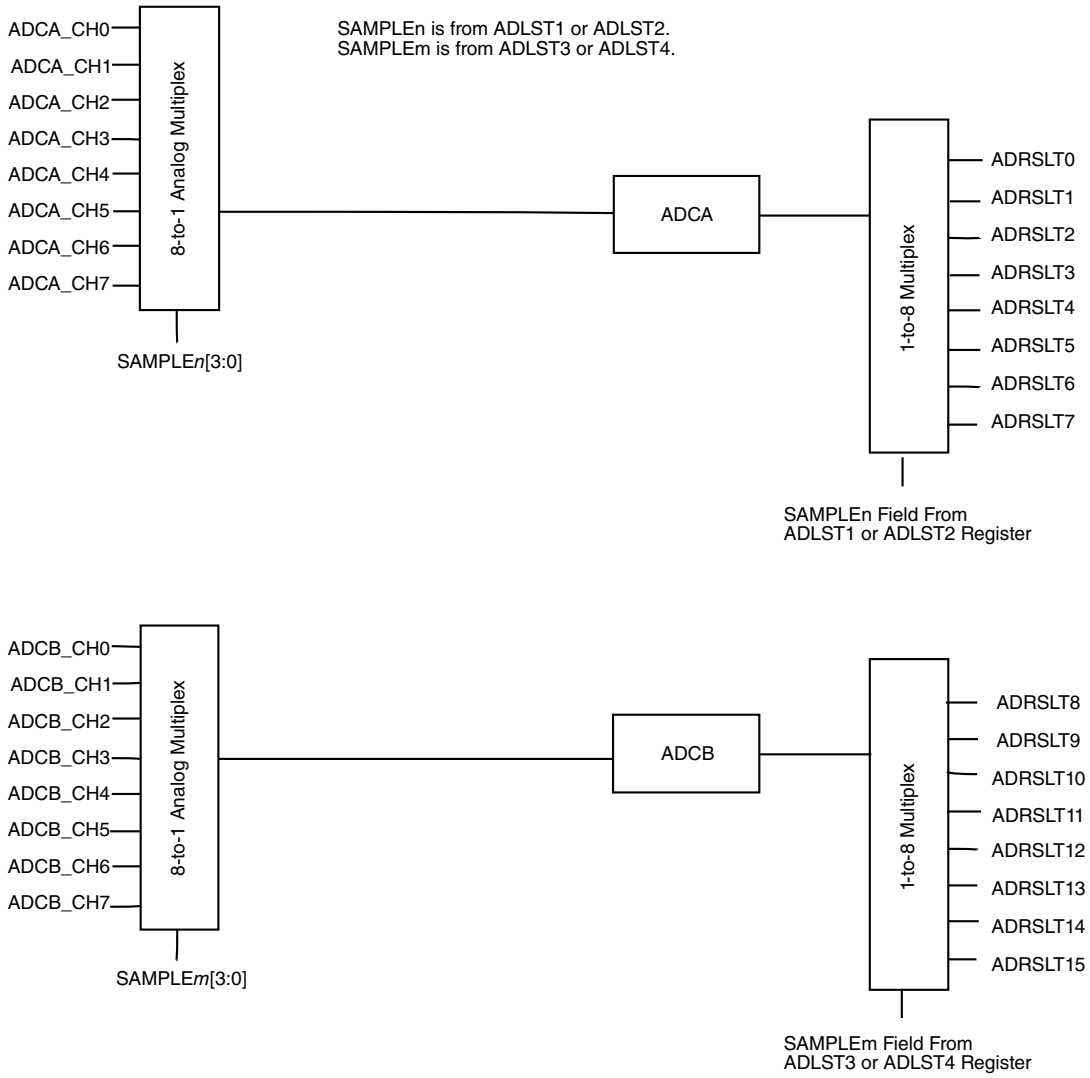


Figure 34-5. ADC Parallel Scan Mode

Each of the following pairs of analog inputs can be configured as a differential pair:

- ADCA_CH[0:1], ADCA_CH[2:3], ADCA_CH[4:5], ADCA_CH[6:7]
- ADCB_CH[0:1], ADCB_CH[2:3], ADCB_CH[4:5], ADCB_CH[6:7]

When they are so configured, a reference to either member of the differential pair by a sample slot results in a differential measurement using that differential pair.

Parallel scan mode can be simultaneous or non-simultaneous. In simultaneous scan mode, the parallel scans in the two converters occur simultaneously and result in simultaneous pairs of conversions, one by converter A and one by converter B. The two converters share the same start, stop, sync, end-of-scan interrupt enable control, and interrupts. Scanning in both converters terminates when either converter encounters a disabled sample. In non-simultaneous scan mode, the parallel scans in the two converters occur

independently. Each converter has its own start, stop, sync, end-of-scan interrupt enable controls, and interrupts. Scanning in either converter terminates only when that converter encounters a disabled sample.

The ADC can be configured to perform a single scan and halt, perform a scan whenever triggered, or perform the scan sequence repeatedly until manually stopped. The single scan (once mode) differs from the triggered mode only in that SYNC input signals must be re-armed after each use and subsequent SYNC inputs are ignored until the SYNC input is re-armed. This arming can occur any time after the SYNC pulse, including while the scan is still in process.

Optional interrupts can be generated at the end of a scan sequence. Interrupts are available simply to indicate that a scan has ended, that a sample is out of range, or several different zero crossing conditions. Range is determined by the high and low limit registers.

34.5.1 Input Multiplex Function

The following figure shows the input multiplex function. The ChannelSelect and SingleEndedDifferential switches are indirectly controlled by settings within the following registers:

- CLIST1, CLIST2, CLIST3, CLIST4, and SDIS registers
- CTRL1[CHNCFG_L]
- CTRL2[CHNCFG_H]

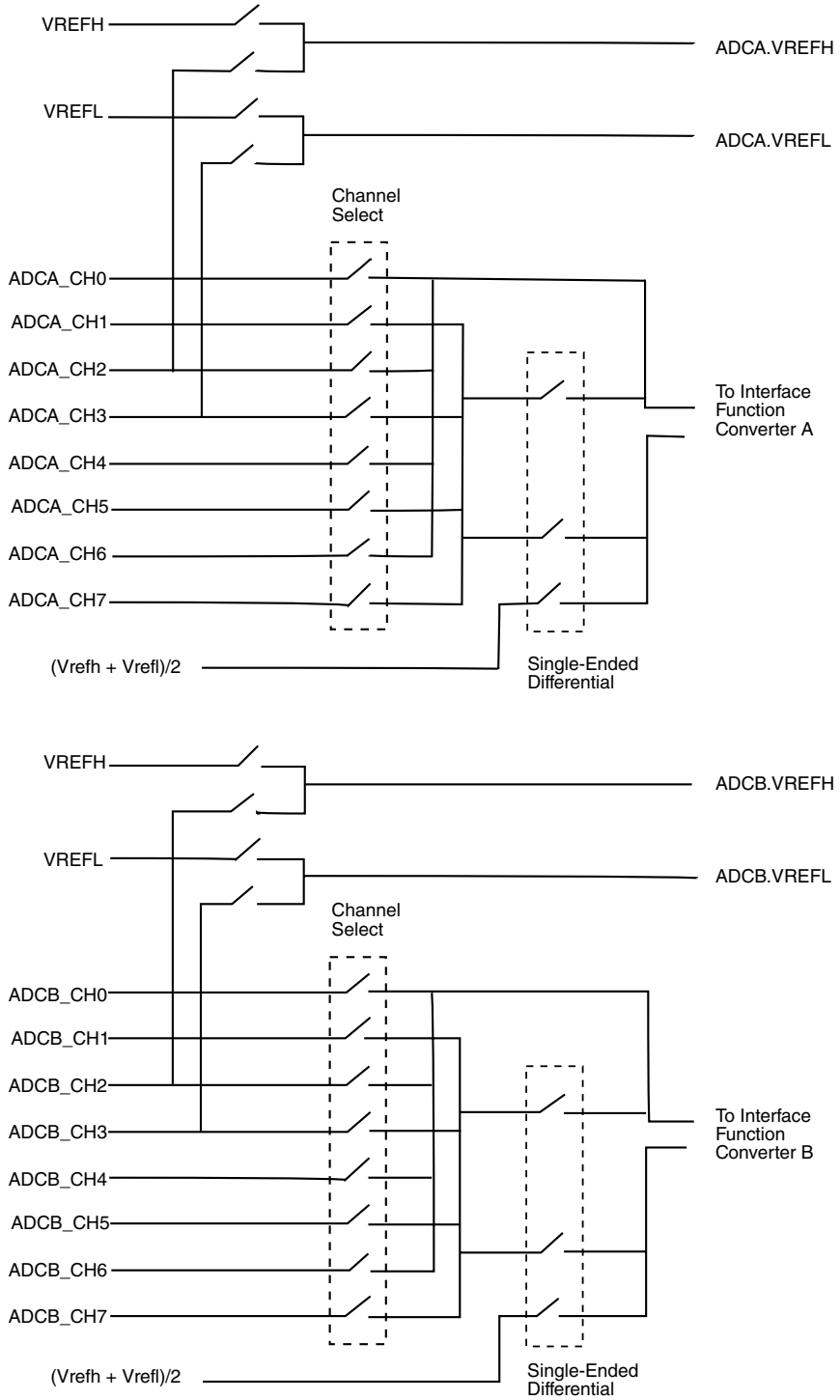


Figure 34-6. Input Select Multiplex

The multiplexing for conversions in different operating modes is as follows:

- Sequential, single-ended mode conversions — During each conversion cycle (sample), any one input of the two four input groups can be directed to its corresponding output.

- Sequential, differential mode conversions — During any conversion cycle (sample), either member of a differential pair may be referenced, resulting in a differential measurement on that pair.
- Parallel, single-ended mode conversions — During any conversion cycle (sample), any of ADCA_CH[0:7] can be directed to the converter A output and any of ADCB_CH[0:7] can be directed to the converter B output.
- Parallel, differential mode conversions — During any conversion cycle (sample), either member of any possible differential pair—ADCA_CH0/1, ADCA_CH2/3, ADCA_CH4/5, ADCA_CH6/7, ADCB_CH0/1, ADCB_CH2/3, ADCB_CH4/5, and ADCB_CH6/7—can be referenced, resulting in a differential measurement of that pair at the converter A output (for ADCA pairs) or converter B output (for ADCB pairs).

The details of single-ended and differential measurement are described under the CHNCFG field. Internally, all measurements are performed differentially.

Table 34-3. Analog Multiplexing Controls for Each Conversion Mode

Conversion Mode	Channel Select Switches	Single-Ended Differential Switches
General		The two lower switches within the dashed box are controlled so that one switch is always closed and the other open.
Sequential, single-ended	The two 1-of-8 select multiplexes can be set for the appropriate input line.	The lower switch is closed, providing $(V_{REFH}+V_{REFL})/2$ to the differential input of the A/D. The upper switch is always closed so that any of the four inputs can get to the A/D input.
Sequential, differential	The channel select switches are turned on in pairs, providing a dual 1-of-4 select function so that either of the two differential channels can be routed to the A/D input.	The upper and lower switches are open and the middle switch is closed, providing the differential channel to the differential input of the A/D.
Parallel, single-ended	The two 1-of-8 select multiplexes can be set for the appropriate input line.	The lower switch is closed, providing $V_{REF}/2$ to the differential input of the A/D. The upper switch is always closed so that any of the four inputs can get to the A/D input.
Parallel, differential	The channel select switches are turned on in pairs, providing a dual 1-of-4 select function so that either of the two differential channels can be routed to the A/D input.	The upper and lower switches are open and the middle switch is closed, providing the differential channel to the differential input of the A/D.

34.5.2 ADC Sample Conversion Operating Modes

The ADC consists of a cyclic, algorithmic architecture using two recursive sub-ranging sections (RSD 1 and RSD 2) as shown in the following figure. Each sub-ranging section resolves a single bit for each conversion clock, resulting in an overall conversion rate of 2 bits per clock cycle. Each sub-ranging section runs at a maximum clock speed of 25 MHz, so a complete 12-bit conversion can be accommodated in 240 ns, not including sample or post-processing time.

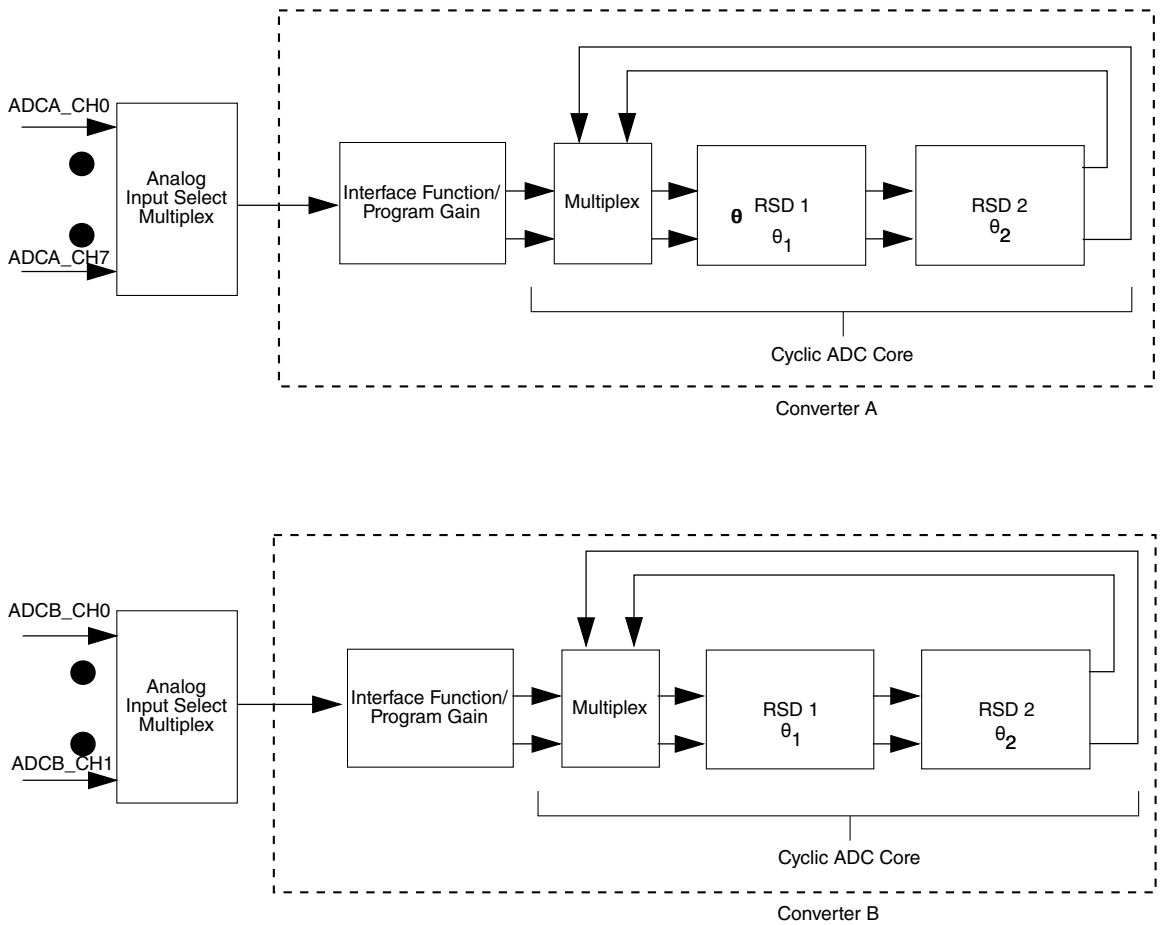


Figure 34-7. Top-Level Diagram of Cyclic ADC

34.5.2.1 Normal Mode Operation

The ADC has two normal operating modes: single-ended mode and differential mode. For a given sample, the mode of operation is determined by the CTRL1[CHNCFG] field:

- Single-ended mode (CHNCFG bit=0). The input multiplex of the ADC selects one of the 8 analog inputs and directs it to the plus terminal of the A/D core. The minus terminal of the A/D core is connected to the V_{REFL} reference. The ADC measures the voltage of the selected analog input and compares it against the (V_{REFH} - V_{REFL}) reference voltage range.
- Differential mode (CHNCFG bit=1). The ADC measures the voltage difference between two analog inputs and compares that value against the (V_{REFH}–V_{REFL}) voltage range. The input is selected as an input pair: ADCA_CH0/1, ADCA_CH2/3, ADCA_CH4/5, ADCA_CH6/7, ADCB_CH0/1, ADCB_CH2/3, ADCB_CH4/5, or ADCB_CH6/7. The plus terminal of the A/D core is connected to the even analog input, and the minus terminal is connected to the odd analog input.

A mix and match combination of single-ended and differential configurations may exist. For example:

- ADCA_CH[0:1] differential; ADCA_CH[2:3] single-ended
- ADCA_CH[4:5] differential; ADCA_CH[6:7] single-ended
- ADCB_CH[0:1] differential; ADCB_CH[2:3] single-ended
- ADCB_CH[4:5] differential; ADCB_CH[6:7] single-ended

34.5.2.1.1 Single-Ended Samples

The ADC module performs a ratio metric conversion. For single-ended measurements, the digital result is proportional to the ratio of the analog input to the reference voltage:

$$SingleEndedValue = round(((V_{IN} - V_{REFL})/(V_{REFH} - V_{REFL})) \times 4096) \times 8$$

V_{IN} is the applied voltage at the input pin.

V_{REFH} and V_{REFL} are the voltages at the external reference pins on the device (typically V_{REFH}=V_{DDA} and V_{REFL}=V_{SSA}).

Note: The 12-bit result is rounded to the nearest LSB.

Note: The ADC is a 12-bit function with 4096 possible states. However, the 12 bits have been left shifted by 3 bits on the 16-bit data bus. As a result, the magnitude of this function, as read from the data bus, is now 32760.

34.5.2.1.2 Differential Samples

For differential measurements, the digital result is proportional to the ratio of the difference in the inputs to the difference in the reference voltages (V_{REFH} and V_{REFL}).

When differential measurements are converted, the following formula is useful:

$$DifferentialValue = round\left(\left(\frac{VIN1-VIN2}{VREFH-VREFL} \times 2048\right) + 2048\right) \times 8$$

V_{IN}=Applied voltage at the input pin

V_{REFH} and V_{REFL}=Voltage at the external reference pins on the device (typically $V_{REFH}=V_{DDA}$ and $V_{REFL}=V_{SSA}$)

Note: The 12-bit result is rounded to the nearest LSB.

Note: The ADC is a 12-bit function with 4096 possible states. However, the 12 bits have been left shifted three bits on the 16-bit data bus, so the magnitude of this function, as read from the data bus, is now 32760.

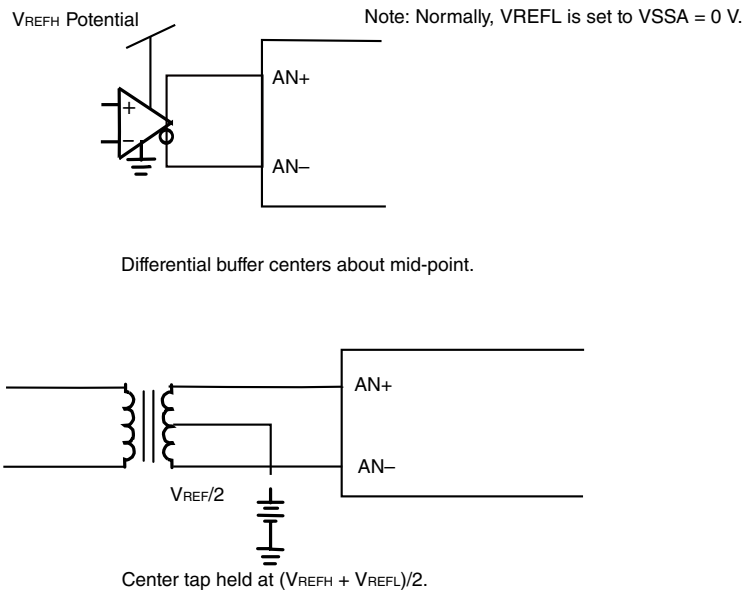


Figure 34-8. Typical Connections for Differential Measurements

34.5.3 ADC Data Processing

The result of an ADC conversion process is normally sent to an adder for offset correction, as the following figure shows. The adder subtracts the ADC_OFFST register value from each sample, and the resultant value is stored in the result register (RSLT). The raw ADC value and the RSLT values are checked for limit violations and zero-crossing, as shown. Appropriate interrupts are asserted, if enabled.

The result value sign is determined from the ADC unsigned result minus the respective offset register value. If the offset register is programmed with a value of zero, the result register value is unsigned and equals the cyclic converter unsigned result. The range of the result (RSLT) is 0000H–7FF8H, assuming that the offset register (OFFST) is cleared to all zeros. This is equal to the raw value of the ADC core.

The processor can write the result registers used for the results of a scan when the STOP bit for that scan is asserted. This write operation is treated as if it comes from the ADC analog core, so the limit checking, zero crossing, and the offset registers function as if in

normal mode. For example, if the STOP bit is set to one and the processor writes to RSLT5, the data written to the RSLT5 is multiplexed to the ADC digital logic inputs, processed, and stored into RSLT5 as if the analog core had provided the data. This test data must be justified, as illustrated by the RSLT register definition and does not include the sign bit.

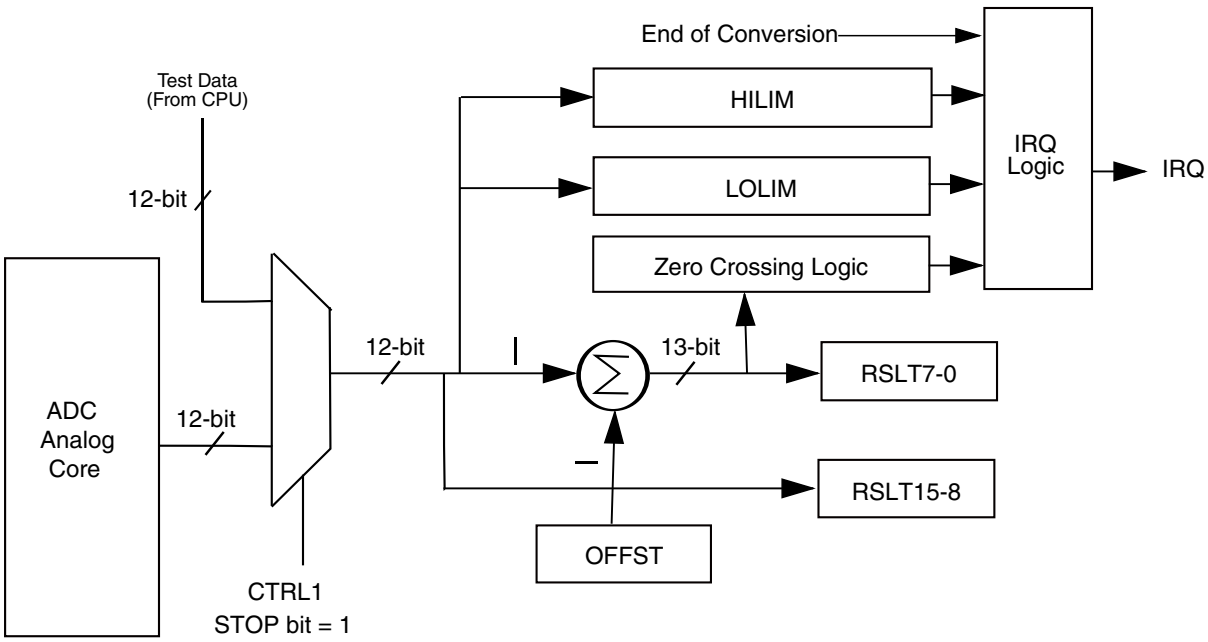


Figure 34-9. Result Register Data Manipulation

34.5.4 Sequential Versus Parallel Sampling

All scan modes use the sixteen sample slots in the CLIST1–4 registers. The slots are used to define which input or differential pair to measure at each step in a scan sequence. The SDIS register defines which sample slots are enabled. Input pairs ADCA_CH0/1, ADCA_CH2/3, ADCA_CH4/5, ADCA_CH6/7, ADCB_CH0/1, ADCB_CH2/3, ADCB_CH4/5, and ADCB_CH6/7 can be set to be measured differentially using the CHNCFG field. If a sample refers to an input that is not configured as a member of a differential pair, a single-ended measurement is made. If a sample refers to either member of a differential pair, a differential measurement is made.

Scan are either sequential or parallel. In sequential scans, up to sixteen sample slots are sampled one at a time in order, SAMPLE [0:15]. Each sample refers to any of the sixteen analog inputs ADCA_CH0–ADCB_CH7, so the same input can be referenced by more than one sample slot. All samples have the full functionality of offset subtraction and high/low limit compare. Scanning is initiated when the CTRL1 [START0] bit is written with a 1 or when the CTRL1[SYNC0] bit is set and the SYNC0 input goes high. A scan ends when the first disabled sample slot is encountered per the SDIS register. Completion

of the scan triggers the STAT[EOSI0] interrupt if the CTRL1[EOSIEN0] interrupt enable is set. If CTRL1[DMAEN0] is set and CTRL3[DMASRC]=0 a DMA transfer of the result data is initiated. The CTRL1[START0] bit and SYNC0 input are ignored while a scan is in process. Scanning stops and cannot be initiated when the CTRL1 [STOP0] bit is set.

Parallel scans differ in that converter A performs up to eight samples (SAMPLE[0:7]) in parallel with converter B (SAMPLE[8:15]). Constraints are as follows:

- SAMPLEs[0:7] can reference only the ADCA_CH[0:7] inputs.
- SAMPLEs[8:15] can reference only the ADCB_CH[0:7] inputs.

Within these constraints, any sample can reference any pin, and more than one sample slot can reference the same sample and the same input. All samples have the full functionality of offset subtraction and high/low limit compare. By default (when CTRL2[SIMULT]=1), the scans in both converters are initiated when the CTRL1[START0] bit is written with a 1 or when the CTRL1[SYNC0] bit has a value of 1 and the SYNC0 input goes high. The scan in both converters terminates when either converter encounters a disabled sample slot. Completion of a scan triggers the STAT[EOSI0] interrupt if the CTRL1[EOSIEN0] interrupt enable is set. If CTRL1[DMAEN0] is set and CTRL3[DMASRC]=0 then a DMA transfer of the result data is initiated. Samples are always taken simultaneously in both the A and B converters. Setting the CTRL1 [STOP0] bit stops and prevents the initiation of scanning in both converters.

Setting CTRL2[SIMULT]=0 (non-simultaneous mode) causes parallel scanning to operate independently in the A and B converter. Each converter has its own set of START, STOP, SYNC, DMAEN, and EOSIEN control bits, SYNC input, EOSI interrupt, and CIP status indicators (suffix 0 for converter A and suffix 1 for converter B). Though still operating in parallel, the scans in the A and B converter start and stop independently according to their own controls and can be simultaneous, phase shifted, or asynchronous depending on when scans are initiated on the respective converters. The A and B converters can be of different length (still up to a maximum of 8) and each converter's scan completes when a disabled sample is encountered in that converters sample list only. CTRL1[STOP0] stops the A converter only and CTRL2[STOP1] stops the B converter only. Looping scan modes iterate independently. Each converter independently restarts its scan after completing its list or encountering a disabled sample slot.

34.5.5 Scan Sequencing

The sequential and parallel scan modes fall into three types based on how they repeat:

- Once scan. A once scan executes a sequential or parallel scan only once each time it is started. It differs from a triggered scan in that sync inputs must be re-armed after each use.
- Triggered scan. Identical to the corresponding once scan modes except that resetting CTRL*[SYNC*] bits is not necessary.
- Looping scan. Automatically restarts a scan, either parallel or sequential, as soon as the previous scan completes. In parallel looping scan modes, the A converter scan restarts as soon as the A converter scan completes and the B converter scan restarts as soon as the B converter scan completes. All subsequent start and sync pulses are ignored after the scan begins unless the scan is paused by the SCTRL[SC] bits. Scanning can only be terminated by setting the STOP bit.

All scan modes ignore sync pulses while a scan is in process unless the scan is paused by the SCTRL[SC] bits. Once scan modes continue to ignore sync pulses even after the scan completes until the CTRL*[SYNC*] bit is set again. However, a reset can occur any time including during the scan. The SYNC0 input is re-armed by setting the CTRL1[SYNC0] bit, and the SYNC1 input is reset by setting the CTRL2[SYNC1] bit. A reset can be performed any time after a scan starts.

34.5.6 Power Management

The five supported power modes are discussed in order from highest to lowest power usage at the expense of increased conversion latency and/or startup delay. See the Clocks section for details on the various clocks referenced here.

34.5.6.1 Low Power Modes

In the following table, ADC's low-power modes are discussed in order from from highest to lowest power usage.

Mode	Description
Normal power	At least one ADC converter is powered up (PWR[PD0 or PD1] is 0), the PWR[APD and ASB] bits are both 0, and the SIM_SCGC5[ADC] bit is 1. The ADC uses the conversion clock as the ADC clock source in either active or idle. The conversion clock should be configured at or near 25 MHz to minimize conversion latency although PWR2[SPEEDn] can be used for reduced power consumption when lower conversion frequencies are acceptable . No startup delay (PWR[PUDELAY]) is imposed.
Auto-standby	At least one ADC converter is powered up (PWR[PD0 or PD1] is 0), PWR[APD] is 0, PWR[ASB] is 1 and MCGIRC enabled. The ADC uses the conversion clock when active and MCGIRC when idle. The standby (low current) state automatically engages when the ADC is idle. The conversion clock should be configured at or near 25 MHz to minimize conversion latency when active although PWR2[SPEEDn] can be used for reduced power consumption when lower conversion frequencies are acceptable. At the start of all scans, there is a startup delay of PWR[PUDELAY]

Table continues on the next page...

Mode	Description
	ADC clocks to engage the conversion clock and revert from standby to normal current mode. Auto-standby is a compromise between normal and auto-powerdown modes. This mode offers moderate power savings at the cost of a moderate latency when leaving the idle state to start a new scan.
Auto-Powerdown	At least one ADC converter is powered up (PWR[PD0 or PD1] is 0), PWR[APD] is 1, and the SIM_SCGC5[ADC] bit is 1. The conversion clock should be configured at or near 20 MHz to minimize conversion latency when active although PWR2[SPEEDn] can be used for reduced power consumption when lower conversion frequencies are acceptable. The ADC uses the conversion clock when active. For maximum power savings, it gates off the conversion clock and powers down the converters when idle. At the start of all scans, there is a startup delay of PWR[PUDELAY] ADC clocks to stabilize normal current mode from a completely powered off condition. This mode saves more power than auto-standby but requires more startup latency when leaving the idle state to start a scan (higher PWR[PUDELAY] value).
Standby	At least one ADC converter is powered up (PWR[PD0 or PD1] is 0), the PWR[ASB] bit is 0, the chip is in STOP/VLPS/PSTOP2 mode and the PLL is bypassed. The ADC clock operates continuously by MCGIRC and standby current mode is enabled continuously without loss of conversion accuracy. Though no startup delay (PWR[PUDELAY]) is imposed, the latency of a scan is affected by the low frequency of the ADC clock. To use auto-powerdown mode and standby mode together, set PWR[APD]. In this hybrid mode ADC clock is derived from MCGIRC when active, and it gates off the ADC clock and powers down the converters when idle. At the start of all scans, there is a startup delay of PWR[PUDELAY] ADC clock cycles to engage the conversion clock and revert power-up the converters and stabilize them in the standby current mode. This is the slowest and lowest power operational configuration of the ADC.
Powerdown	Both ADC converters and voltage references are powered down (PWR[PD0 and PD1] are both 1) and the SIM_PCE[ADC] bit is 0. In this configuration, the clock trees to the ADC and all of its analog components are shut down and power utilization is eliminated.

34.5.6.2 Startup in Different Power Modes

The ADC voltage reference and converters are powered down (PWR[PDn]=1) on reset. Individual converters and voltage references can be manually powered down when not in use (PWR[PD0]=1 or PWR[PD1]=1). When the ADC reference is powered down, the output reference voltages are set to Low (VSSA) and the ADC data output is driven low.

A delay of PWR[PUDELAY] ADC clock cycles is imposed when PWR[PD0 or PD1] are cleared to power up a regulator and also to transition from an idle state in which neither converter has a scan in process to an active state in which at least one converter has a scan in process. ADC data sheets recommend the use of two PWR[PUDELAY] values: a large value for full powerup and a moderate value for transitioning from standby current levels to full powerup.

To start up in normal mode or standby power mode, perform the following steps:

1. Set PWR[PUDELAY] to the large power -up value.
2. Clear PWR[ASB and APD].
3. Clear the PWR[PD0 and/or PD1] bits to power up the required converters.

4. Poll the status bits until all required converters are powered up.
5. Start scan operations. This will provide a full power-up delay before scans begin.

Normal mode does not use PWR[PUDELAY] at start of scan, so no further delay is imposed.

To start up in auto-standby, use the normal mode startup procedure first. Before starting scan operations, set PWR[PUDELAY] to the moderate standby recovery value, and set PWR[ASB]. Auto- standby mode automatically reduces current levels until active and then imposes the PWR[PUDELAY] to allow current levels to rise from standby to full power levels.

To start up in auto powerdown mode, perform the following steps:
1. Set PWR[PUDELAY] to the large power-up value.
2. Clear PWR[ASB] and set PWR[APD].
3. Clear the PWR[PD0 and/or PD1] bits for the required converters.

Converters remain powered off until scanning goes active. Before a scan starts, there is a large PWR [PUDELAY] to go from the powered down to the fully powered state.

To avoid ambiguity and ensure that the proper delays are applied when powering up or starting scans, both regulators should be powered off (PWR[PD0]=PWR[PD1]=1) when the clock or power controls are configured.

Attempts to start a scan during the PWR[PUDELAY] are ignored until the appropriate PWR[PSTSn] bits are cleared.

Any attempt to use a converter when it is powered down or with the voltage references disabled will result in invalid results. It IS possible to read ADC result registers after converter power down for results calculated before power-down. A new scan sequence must be started with a SYNC pulse or a write to the START bit before new valid results are available.

In auto-powerdown mode, when the ADC goes from idle to active, a converter is powered up only if it is required for the scan as determined by the CLIST1-4 and SDIS registers.

34.5.6.3 Stop Mode of Operation

Any conversion sequence can be stopped by setting the relevant STOP bit. Any further sync pulses or writes to the start bit are ignored until the STOP bit is cleared. In stop mode, the results registers can be modified by writes from the processor. Any write to the result register in the ADC-STOP mode is treated as if the analog core supplied the data, so limit checking and zero crossing and associated interrupts can occur if enabled.

34.6 Reset

At reset, all the registers return to the reset state.

34.7 Clocks

The ADC has two external clock inputs to drive two clock domains within the ADC module.

Table 34-4. Clock Summary

Clock input	Source	Characteristics
IP Clock	SIM	Maximun rate is 150 MHz from fast bus clock domain. When the device is in low power mode, the IP clock is from MCGIRCLK.
RC Clock	MCG	MCG provides MCGIRCLK for auto standby power saving mode. MCGIRCLK must be configured to 4 MHz before ADC enters low power mode.

The IP clock is sourced by fast bus clock and the maximum rate is 150 MHz.

The IP_CLK is enabled only when the SIM_SCGC5[ADC] bit is set. This clock enable bit must be set before the ADC can be used.

The conversion clock is the primary source for the ADC clock and is always selected as the ADC clock when conversions are in process. The ADC clock is sourced by fast bus clock in normal mode and MCGIRC in stop mode,, and CTRL2[DIV0] and PWR2[DIV1] should be configured so that conversion clock frequency falls between 100 kHz and 25 MHz. Operating the ADC at out-of-spec conversion clock frequencies or reconfiguring the parameters that affect clock rates or power modes while the regulators are powered up (PWR[PD0]=0 or PWR[PD1]=0) negatively affects conversion accuracy.

The conversion clock that the ADC uses for sampling is calculated using the IP bus clock and the clock divisor bits within the ADC Control Register 2. The ADC clock is active 100 percent of the time in looping modes or in normal power mode. It is also active during all ADC powerup sequences for a period of time determined by the PWR[PUDELAY] field. If a conversion is initiated in power savings mode, then the ADC clock continues until the conversion sequence completes.

The following diagram shows the structure of the clocking system.

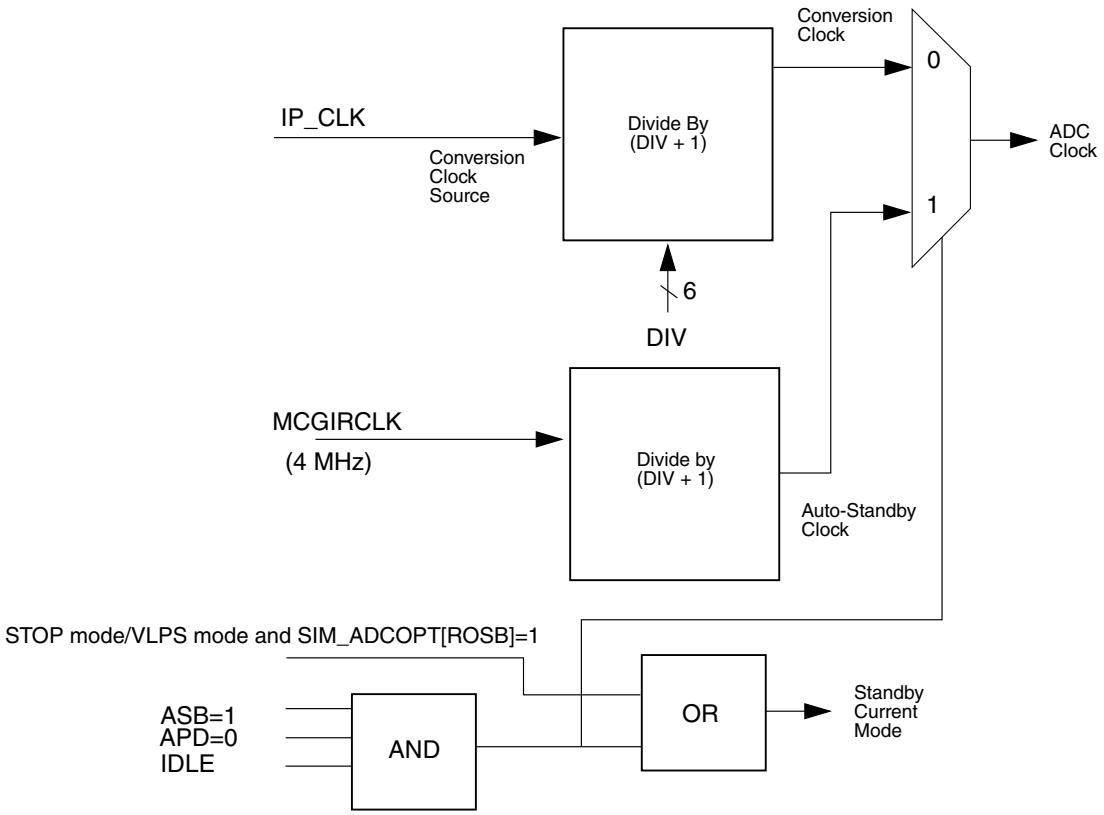


Figure 34-10. ADC Clock Generation

MCGIRCLK clocks a divider to generate the auto-standby clock, which is selected as the ADC clock only during auto-standby power mode and only when both converters are idle. The logic that selects the standby clock as the ADC clock also asserts standby current mode, which is available only at an ADC clock rate of less than 667 kHz. This mode provides substantially power savings yet requires less latency when switching back to the conversion clock at the start of a scan than auto-powerdown mode, which uses only the conversion clock as the ADC clock source but fully powers down the converters when idle.

The standby current mode is also engaged when the chip is in stop mode and SIM_ADCOPT[ROSB]=1. It is necessary to configure DIV0/1 to ensure that ADC conversion clock rate is less than 667 kHz while a conversion is in process.

The ADC_CLK is an output of the gasket used to operate the two converters during scan operations. It is derived by multiplexing the conversion clock (divided version of the conversion clock source) and the standby clock (divided version of MCGIRC). This clock can be selected in the SIM for external output for debug and failure analysis.

34.8 Interrupts

The following table summarizes the ADC interrupts.

Table 34-5. Interrupt Summary

Interrupt	Source	Description
ADC_ERR_INT_B	STAT[[ZCI], STAT[LLMTI], STAT[HLMTI]	Zero Crossing, low Limit, and high limit interrupt
ADC_CC0_INT_B	STAT[EOSI0] RDY[RDY[15:0]]	Conversion Complete and Scan Interrupt for any scan type except converter B scan in non-simultaneous parallel scan mode (see EOSI0)
ADC_CC1_INT_B	STAT[EOSI1] RDY[RDY[7:4]] RDY[RDY[15:12]]	Conversion Complete and Scan Interrupt for converter B scan in non-simultaneous parallel scan mode (see EOSI1)

ADC interrupts fall into three categories:

- Threshold interrupts, which are caused by three different events. All of these interrupts are optional and enabled through control register CTRL1:
  - Zero crossing — occurs if the current result value has a sign change from the previous result as configured by the ZXCTRL register.
  - Low limit exceeded error — occurs when the current result value is less than the low limit register value. The raw result value is compared to LOLIM[LLMT] before the offset register value is subtracted.
  - High limit exceeded error — is asserted if the current result value is greater than the high limit register value. The raw result value is compared to HILIM[HLMT] before the offset register value is subtracted.
- Conversion complete interrupts, which are generated upon completion of any scan and convert sequence when CTRL1[EOSIE0]=1. Additional bits may need to be set in the Interrupt Control Module to enable the CPU to receive the interrupt signal.
- Scan interrupts are generated when a sample is converted. This allows processing of intermediate conversion data during a scan. The interrupt occurs when any sample has its SCTRL[SC] and SCHLTEN[SCHLTEN] bit enabled (set), and the RDY[RDY] bit for that sample is asserted. Use these registers to determine which sample triggered the interrupt.

34.9 Timing Specifications

The following figure shows a timing diagram for the ADC module. The ADC is assumed to be in Once or Triggered mode, so the ADC clock is shown in the OFF state prior to the SYNC pulse or START bit write. The ADC clock restarts (switching high) within 1 to 2 IP bus clocks of that event. ADC_CLK is derived from the ROSC or PLL output. The frequency relationship is programmable. Conversions are pipelined. The second start command is ignored because the ADC is busy with the previous start request. The third start command is recognized and is synchronized to the positive edge of the ADC clock when the conversion process is restarted. The ADC has two possible interrupts that are latched in the ADSTAT register:

- Conversion complete interrupt (End of Scan interrupt, EOSIx)
- Zero crossing or limit error interrupt (ZCI, LLMTI, and HLMTI)

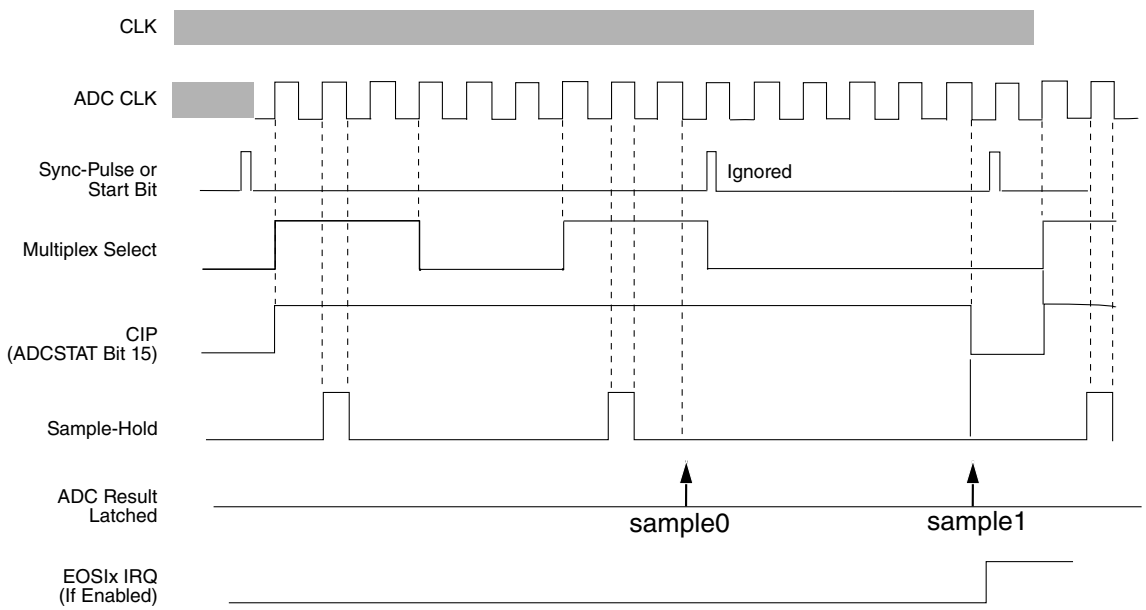


Figure 34-11. ADC Timing

As the figure shows, a conversion is initiated by (1) a sync pulse originating from the timer module or by (2) a write to a start bit. In APD or ASB mode, a delay of PWR [PUDELAY] ADC clock cycles is imposed. The conversion is initiated in the next clock cycle. The ADC clock period is determined by the CTRL2[DIV0] or PWR2[DIV1] value and the fast peripheral clock.

The first conversion takes 8.5 ADC clocks to be valid. Then, each additional sample takes only six ADC clocks. The start conversion command is latched and the real conversion process is synchronized to the positive edge of the ADC clock.

Because the conversion is a pipeline process, after the last sample is in the S/H, the ADC cannot be restarted until the pipeline is emptied. However, the conversion cycle can be aborted by issuing a STOP command.

The figure shown here illustrates the case in which PWR[APD and ASB] are not in use. When the PWR[APD or ASB] bit is set, the sync pulse or start powers up the ADC, waits for a number of ADC clocks (determined by the PWR[PUDELAY] bits) for the ADC circuitry to stabilize, and only then begins the conversion sequence.

Chapter 35
Comparator (CMP)

35.1 Chip-specific CMP information

35.1.1 CMP instantiation information

This chip has four high speed CMP modules each with a optional 6-bit DAC to provide a trigger point.

35.1.2 CMP Signal Assignments

NOTE

For more details see SIM_MISCTRL0 and SIM_SOPT7 registers

Table 35-1. CMP Signal assignments

Instance	Signal	Description	I/O	Connected to	Ext Pin Mux / SIM	Pin No. Of 100
CMP0	IN0	input channel	input	CMP0_IN0 pin	PTC6	pin78 of 100
	IN1	input channel	input	CMP0_IN1 pin	PTC7	pin79 of 100
	IN2	input channel	input	CMP0_IN2 pin	PTC8	pin80 of 100
	IN3	input channel	input	CMP0_IN3 pin	PTC9	pin81 of 100
	IN4	input channel	input	CMP0_IN4 pin	—	pin28 of 100
	IN5	input channel	input	CMP0_IN5 pin	PTE29	pin26 of 100
	IN6	input channel	input	bandgap	—	—
	IN7	input channel	input	6bDAC	—	—
	WindowIN	—	input	PDB0 pulse out, PDB1 pulse out XBARA_OUT16	—	—
	CMP0_OUT	—	output	LPTMR0, PDB0 ch 0001, PDB1_ch 0001, DMA_MUX	SIM: FTM0TRG0SRC, FTM1TRG0SRC,	PTB20 , PTC5, PTA1 via digital signal mux

Table continues on the next page...

Table 35-1. CMP Signal assignments (continued)

Instance	Signal	Description	I/O	Connected to	Ext Pin Mux / SIM	Pin No. Of 100
				source 42,XBARA_IN12,XBARB1_IN0	FTM3TRIG0SRC, FTM1CH0SRC, FTM0FLT0, FTM1FLT0, FTM3FLT0, UART0RXSRC, UART1RXSRC	
CMP1	IN0	input channel	input	CMP1_IN0 pin	PTC2	pin72 of 100
	IN1	input channel	input	CMP1_IN1 pin	PTC3	pin73 of 100
	IN2	input channel	input	CMP1_IN2 pin	ADC0_SE8	pin 36 of 144
	IN3	input channel	input	CMP1_IN3 pin	PTE30, DAC0_OUT	pin27 of 100
	IN4	input channel	input	na	—	—
	IN5	input channel	input	CMP1_IN5 pin	PTE29	pin26 of 100
	IN6	input channel	input	bandgap	—	—
	IN7	input channel	input	6bDAC	—	—
	WindowIN	—	input	PDB0 pulse out, PDB1 pulse out XBARA_OUT17	—	—
	CMP1_OUT	—	output	PDB0 ch 0010, PDB1 ch 0010, XBARA_IN13, XBARB_IN1 , DMA_MUX source 43	SIM: FTM1CH0SRC, FTM0FLT1, UART0RXSRC, UART1RXSRC	PTB21,PTA2, PTC4 via digital signal mux
CMP2	IN0	input channel	input	CMP2_IN0 pin	PTA12	pin 42 of 100
	IN1	input channel	input	CMP2_IN1 pin	PTA13	pin43 of 100
	IN2	input channel	input	CMP2_IN2 pin	PTB2	pin55 of 100
	IN3	input channel	input	CMP2_IN3 pin	—	pin28 of 100
	IN4	input channel	input	CMP2_IN4 pin	PTC6	pin78 of 100
	IN5	input channel	input	CMP2_IN5 pin	ADC0_SE0	pin35 of 100
	IN6	input channel	input	bandgap	—	—
	IN7	input channel	input	6bDAC	—	—
	WindowIN	—	input	PDB0 pulse out, PDB1 pulse out XBARA_OUT18	—	—
	CMP2_OUT	—	output	PDB0 ch 0011, PDB1 ch 0011, XBARA_IN14, XBARB_IN2 , DMA_MUX source 44	SIM: FTM0FLT2	—
CMP3	IN0	input channel	input	CMP3_IN0 pin	PTA14	pin44 of 100
	IN1	input channel	input	CMP3_IN1 pin	PTA15	pin45 of 100
	IN2	input channel	input	CMP3_IN2pin	PTA16	pin46 of 100
	IN3	input channel	input	DAC0_OUT	DAC0_OUT	—

Table continues on the next page...

Table 35-1. CMP Signal assignments (continued)

Instance	Signal	Description	I/O	Connected to	Ext Pin Mux / SIM	Pin No. Of 100
	IN4	input channel	input	CMP3_IN4 pin	PTC7	pin79 of 100
	IN5	input channel	input	CMP3_IN5 pin	PTB3	pin56 of 100
	IN6	input channel	input	bandgap	—	—
	IN7	input channel	input	6bDAC	—	—
	WindowIN	—	input	PDB0 pulse out, PDB1 pulse out XBARA_OUT19	—	—
	CMP3_OUT	—	output	XBARA_IN15, XBARB_IN3 , DMA_MUX source 34	—	—

35.1.3 CMP voltage references

The 6-bit DAC sub-block supports selection of two references. For this device, both theV_{in1} input and the V_{in2} input are connected to V_{DD}.

35.1.4 CMP external references

Each CMP has its own 6-bit DAC sub-block which feeds the IN7 of the comparator. Additionally the output of DAC0 is provided to IN6 of each comparator, providing a 12-bit DC reference.

35.1.5 External window/sample input

Individual PDB pulse-out signals control each CMP Sample/Window timing.

35.2 Introduction

The comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal is provided by the 6-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed. The 64-tap resistor ladder network divides the supply reference V_{in} into 64 voltage levels. A 6-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/64$. V_{in} can be selected from two voltage sources, V_{in1} and V_{in2}. The 6-bit DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

35.2.1 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control
- Selectable interrupt on rising-edge, falling-edge, or both rising or falling edges of the comparator output
- Selectable inversion on comparator output
- Capability to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered:
    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Two software selectable performance levels:
  - Shorter propagation delay at the expense of higher power
  - Low power, with longer propagation delay
- DMA transfer support
  - A comparison event can be selected to trigger a DMA transfer
- Functional in all modes of operation except VLLS0
- The window and filter functions are not available in the following modes:
  - Stop

- VLPS
- VLLSx

35.2.2 6-bit DAC key features

The 6-bit DAC has the following features:

- 6-bit resolution
- Selectable supply reference source
- Power Down mode to conserve power when not in use
- Option to route the output to internal comparator input

35.2.3 ANMUX key features

The ANMUX has the following features:

- Two 8-to-1 channel mux
- Operational over the entire supply range

35.2.4 CMP, DAC and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

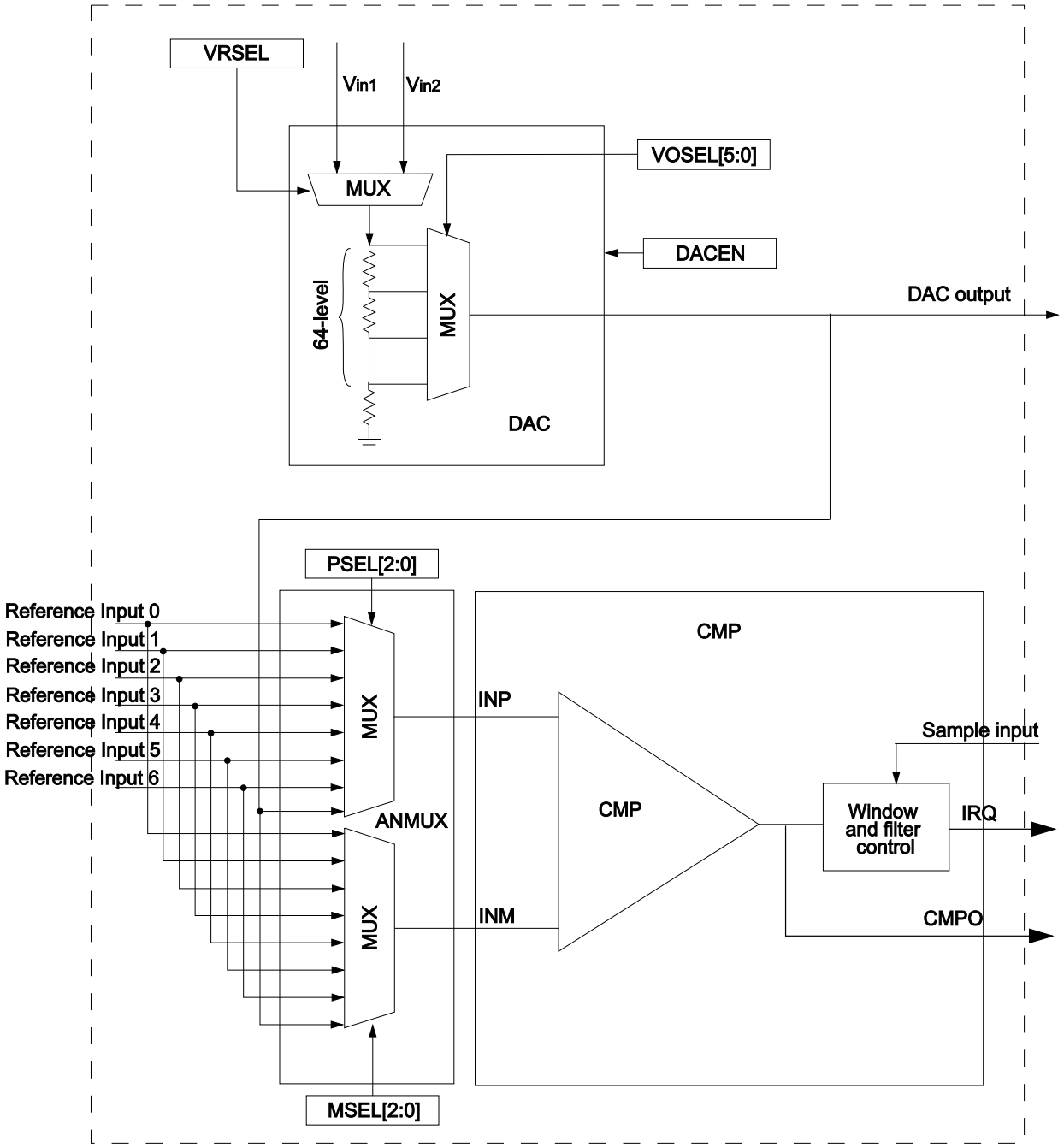


Figure 35-1. CMP, DAC and ANMUX block diagram

35.2.5 CMP block diagram

The following figure shows the block diagram for the CMP module.

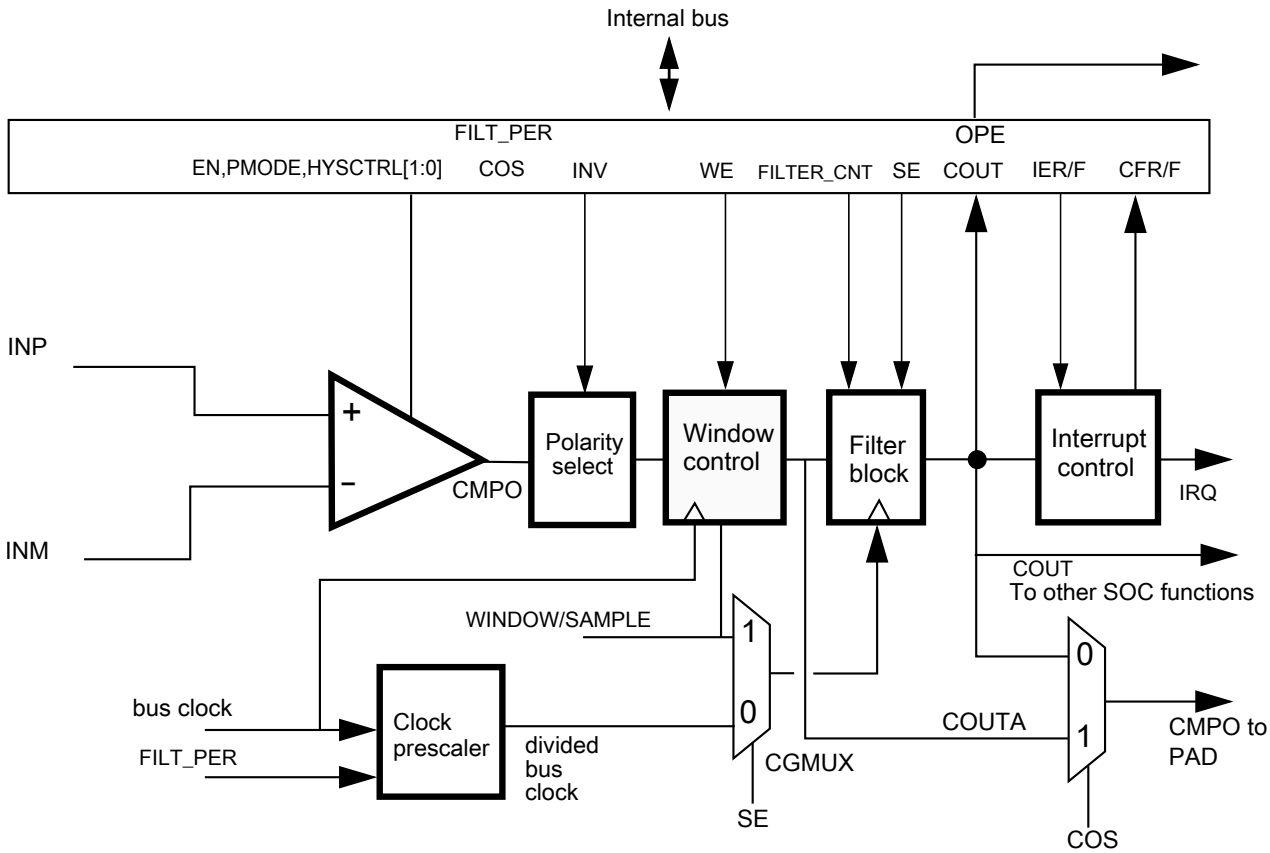


Figure 35-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter block is bypassed when not in use.
- The Filter block acts as a simple sampler if the filter is bypassed and CR0[FILTER_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and CR0[FILTER_CNT] is set greater than 0x01.
  - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
  - If CR1[SE] = 0, the divided bus clock is used as sampling clock

- If enabled, the Filter block will incur up to one bus clock additional latency penalty on COUT due to the fact that COUT, which is crossing clock domain boundaries, must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

35.3 Memory map/register definitions

CMP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_3000	CMP Control Register 0 (CMP0_CR0)	8	R/W	00h	35.3.1/723
4007_3001	CMP Control Register 1 (CMP0_CR1)	8	R/W	00h	35.3.2/723
4007_3002	CMP Filter Period Register (CMP0_FPR)	8	R/W	00h	35.3.3/725
4007_3003	CMP Status and Control Register (CMP0_SCR)	8	R/W	00h	35.3.4/725
4007_3004	DAC Control Register (CMP0_DACCR)	8	R/W	00h	35.3.5/726
4007_3005	MUX Control Register (CMP0_MUXCR)	8	R/W	00h	35.3.6/727
4007_3008	CMP Control Register 0 (CMP1_CR0)	8	R/W	00h	35.3.1/723
4007_3009	CMP Control Register 1 (CMP1_CR1)	8	R/W	00h	35.3.2/723
4007_300A	CMP Filter Period Register (CMP1_FPR)	8	R/W	00h	35.3.3/725
4007_300B	CMP Status and Control Register (CMP1_SCR)	8	R/W	00h	35.3.4/725
4007_300C	DAC Control Register (CMP1_DACCR)	8	R/W	00h	35.3.5/726
4007_300D	MUX Control Register (CMP1_MUXCR)	8	R/W	00h	35.3.6/727
4007_3010	CMP Control Register 0 (CMP2_CR0)	8	R/W	00h	35.3.1/723
4007_3011	CMP Control Register 1 (CMP2_CR1)	8	R/W	00h	35.3.2/723
4007_3012	CMP Filter Period Register (CMP2_FPR)	8	R/W	00h	35.3.3/725
4007_3013	CMP Status and Control Register (CMP2_SCR)	8	R/W	00h	35.3.4/725
4007_3014	DAC Control Register (CMP2_DACCR)	8	R/W	00h	35.3.5/726
4007_3015	MUX Control Register (CMP2_MUXCR)	8	R/W	00h	35.3.6/727
4007_3018	CMP Control Register 0 (CMP3_CR0)	8	R/W	00h	35.3.1/723
4007_3019	CMP Control Register 1 (CMP3_CR1)	8	R/W	00h	35.3.2/723
4007_301A	CMP Filter Period Register (CMP3_FPR)	8	R/W	00h	35.3.3/725
4007_301B	CMP Status and Control Register (CMP3_SCR)	8	R/W	00h	35.3.4/725
4007_301C	DAC Control Register (CMP3_DACCR)	8	R/W	00h	35.3.5/726
4007_301D	MUX Control Register (CMP3_MUXCR)	8	R/W	00h	35.3.6/727

35.3.1 CMP Control Register 0 (CMPx_CR0)

Address: Base address + 0h offset

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

CMPx_CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count Represents the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency, see the Functional description. 000 Filter is disabled. If SE = 1, then COUT is a logic 0. This is not a legal state, and is not recommended. If SE = 0, COUT = COUTA. 001 One sample must agree. The comparator output is simply sampled. 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level are device-specific. See the Data Sheet of the device for the exact values. 00 Level 0 01 Level 1 10 Level 2 11 Level 3

35.3.2 CMP Control Register 1 (CMPx_CR1)

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SE	WE	TRIGM	PMODE	INV	COS	OPE	EN
Reset	0	0	0	0	0	0	0	0

CMPx_CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. It is mandatory request to not set SE and WE both at a given time. 0 Sampling mode is not selected. 1 Sampling mode is selected.
6 WE	Windowing Enable At any given time, either SE or WE can be set. It is mandatory request to not set SE and WE both at a given time. 0 Windowing mode is not selected. 1 Windowing mode is selected.
5 TRIGM	Trigger Mode Enable CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1. In addition, the CMP should be enabled. If the DAC is to be used as a reference to the CMP, it should also be enabled. CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare. Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received. See the chip configuration for details about the external timer resource. 0 Trigger mode is disabled. 1 Trigger mode is enabled.
4 PMODE	Power Mode Select See the electrical specifications table in the device Data Sheet for details. 0 Low-Speed (LS) Comparison mode selected. In this mode, CMP has slower output propagation delay and lower current consumption. 1 High-Speed (HS) Comparison mode selected. In this mode, CMP has faster output propagation delay and higher current consumption.
3 INV	Comparator INVERT Allows selection of the polarity of the analog comparator function. It is also driven to the COUT output, on both the device pin and as SCR[COUT], when OPE=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select 0 Set the filtered comparator output (CMPO) to equal COUT. 1 Set the unfiltered comparator output (CMPO) to equal COUTA.
1 OPE	Comparator Output Pin Enable 0 CMPO is not available on the associated CMPO output pin. If the comparator does not own the pin, this field has no effect. 1 CMPO is available on the associated CMPO output pin.

Table continues on the next page...

CMPx_CR1 field descriptions (continued)

Field	Description
	The comparator output (CMPO) is driven out on the associated CMPO output pin if the comparator owns the pin. If the comparator does not own the field, this bit has no effect.
0 EN	Comparator Module Enable Enables the Analog Comparator module. When the module is not enabled, it remains in the off state, and consumes no power. When the user selects the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator is disabled. 1 Analog Comparator is enabled.

35.3.3 CMP Filter Period Register (CMPx_FPR)

Address: Base address + 2h offset

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

CMPx_FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when CR1[SE]=0. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional description. This field has no effect when CR1[SE]=1. In that case, the external SAMPLE signal is used to determine the sampling period.

35.3.4 CMP Status and Control Register (CMPx_SCR)

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read	0	DMAEN	0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

CMPx_SCR field descriptions

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

Table continues on the next page...

CMPx_SCR field descriptions (continued)

Field	Description
6 DMAEN	DMA Enable Control Enables the DMA transfer triggered from the CMP module. When this field is set, a DMA request is asserted when CFR or CFF is set. 0 DMA is disabled. 1 DMA is enabled.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
3 IEF	Comparator Interrupt Enable Falling Enables the CFF interrupt from the CMP. When this field is set, an interrupt will be asserted when CFF is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 CFR	Analog Comparator Flag Rising Detects a rising-edge on COUT, when set, during normal operation. CFR is cleared by writing 1 to it. During Stop modes, CFR is edge sensitive . 0 Rising-edge on COUT has not been detected. 1 Rising-edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling Detects a falling-edge on COUT, when set, during normal operation. CFF is cleared by writing 1 to it. During Stop modes, CFF is edge sensitive . 0 Falling-edge on COUT has not been detected. 1 Falling-edge on COUT has occurred.
0 COUT	Analog Comparator Output Returns the current value of the Analog Comparator output, when read. The field is reset to 0 and will read as CR1[INV] when the Analog Comparator module is disabled, that is, when CR1[EN] = 0. Writes to this field are ignored.

35.3.5 DAC Control Register (CMPx_DACCR)

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read Write	DACEN	VRSEL	VOSEL					
Reset	0	0	0	0	0	0	0	0

CMPx_DACCR field descriptions

Field	Description
7 DACEN	DAC Enable Enables the DAC. When the DAC is disabled, it is powered down to conserve power. 0 DAC is disabled. 1 DAC is enabled.
6 VRSEL	Supply Voltage Reference Source Select 0 V_{in1} is selected as resistor ladder network supply reference. 1 V_{in2} is selected as resistor ladder network supply reference.
VOSEL	DAC Output Voltage Select Selects an output voltage from one of 64 distinct levels. `DACO = (V in /64) * (VOSEL[5:0] + 1)` , so the DACO range is from V_{in} /64 to V_{in} .

35.3.6 MUX Control Register (CMPx_MUXCR)

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read	Reserved	0	PSEL			MSEL		
Write								
Reset	0	0	0	0	0	0	0	0

CMPx_MUXCR field descriptions

Field	Description
7 Reserved	Bit can be programmed to zero only . This field is reserved.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–3 PSEL	Plus Input Mux Control Determines which input is selected for the plus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams. **NOTE:** When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7

Table continues on the next page...

CMPx_MUXCR field descriptions (continued)

Field	Description
MSEL	Minus Input Mux Control Determines which input is selected for the minus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams. **NOTE:** When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7

35.4 Functional description

The CMP module can be used to compare two analog input voltages applied to INP and INM.

CMPO is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CR1[INV] = 1.

SCR[IER] and SCR[IEF] are used to select the condition which will cause the CMP module to assert an interrupt to the processor. SCR[CFF] is set on a falling-edge and SCR[CFR] is set on rising-edge of the comparator output. The optionally filtered CMPO can be read directly through SCR[COUT].

35.4.1 CMP functional modes

There are the following main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, CR0[FILTER_CNT], can be clocked from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CR1[WE]. When set, the comparator output is sampled only when WINDOW=1. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 35-2. Comparator sample/filter controls

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
1	0	X	X	X	X	**Disabled** See the Disabled mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode** See the Continuous mode (#s 2A & 2B).
2B	1	0	0	X	0x00	
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode** See the Sampled, Non-Filtered mode (#s 3A & 3B).
3B	1	0	0	0x01	> 0x00	
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode** See the Sampled, Filtered mode (#s 4A & 4B).
4B	1	0	0	> 0x01	> 0x00	
5A	1	1	0	0x00	X	**Windowed mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA. See the Windowed mode (#s 5A & 5B).
5B	1	1	0	X	0x00	
6	1	1	0	0x01	0x01–0xFF	**Windowed/Resampled mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled on an interval determined by FILT_PER to generate COUT. See the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01–0xFF	**Windowed/Filtered mode**

Table continues on the next page...

Table 35-2. Comparator sample/filter controls (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
						Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. See the Windowed/Filtered mode (#7).
All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] are illegal.						

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it must be configured to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings must be changed only after setting CR1[SE]=0 and CR0[FILTER_CNT]=0x00. This resets the filter to a known state.

35.4.1.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

35.4.1.2 Continuous mode (#s 2A & 2B)

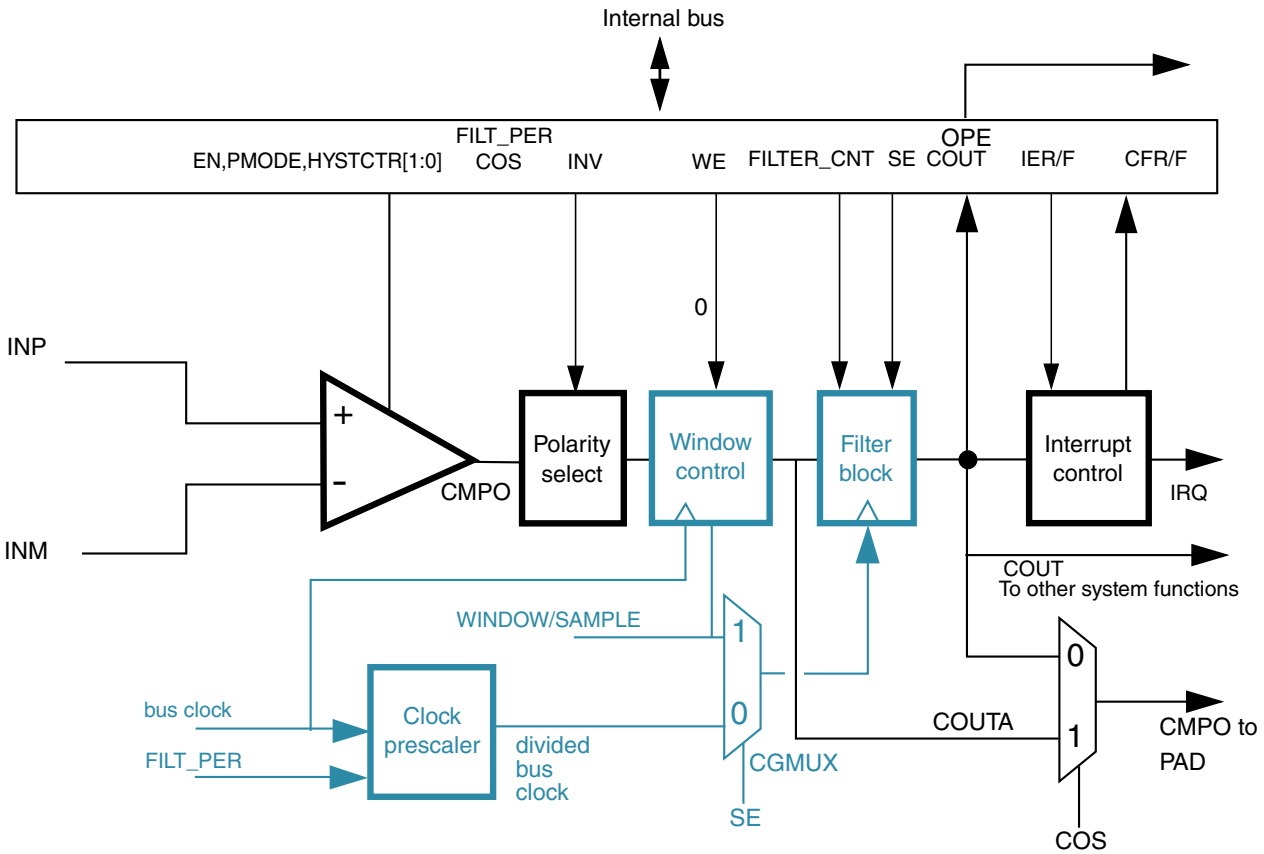


Figure 35-3. Comparator operation in Continuous mode

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed. SCR[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational unclocked mode. COUT and COUTA are identical.

For control configurations which result in disabling the filter block, see the Filter Block Bypass Logic diagram.

35.4.1.3 Sampled, Non-Filtered mode (#s 3A & 3B)

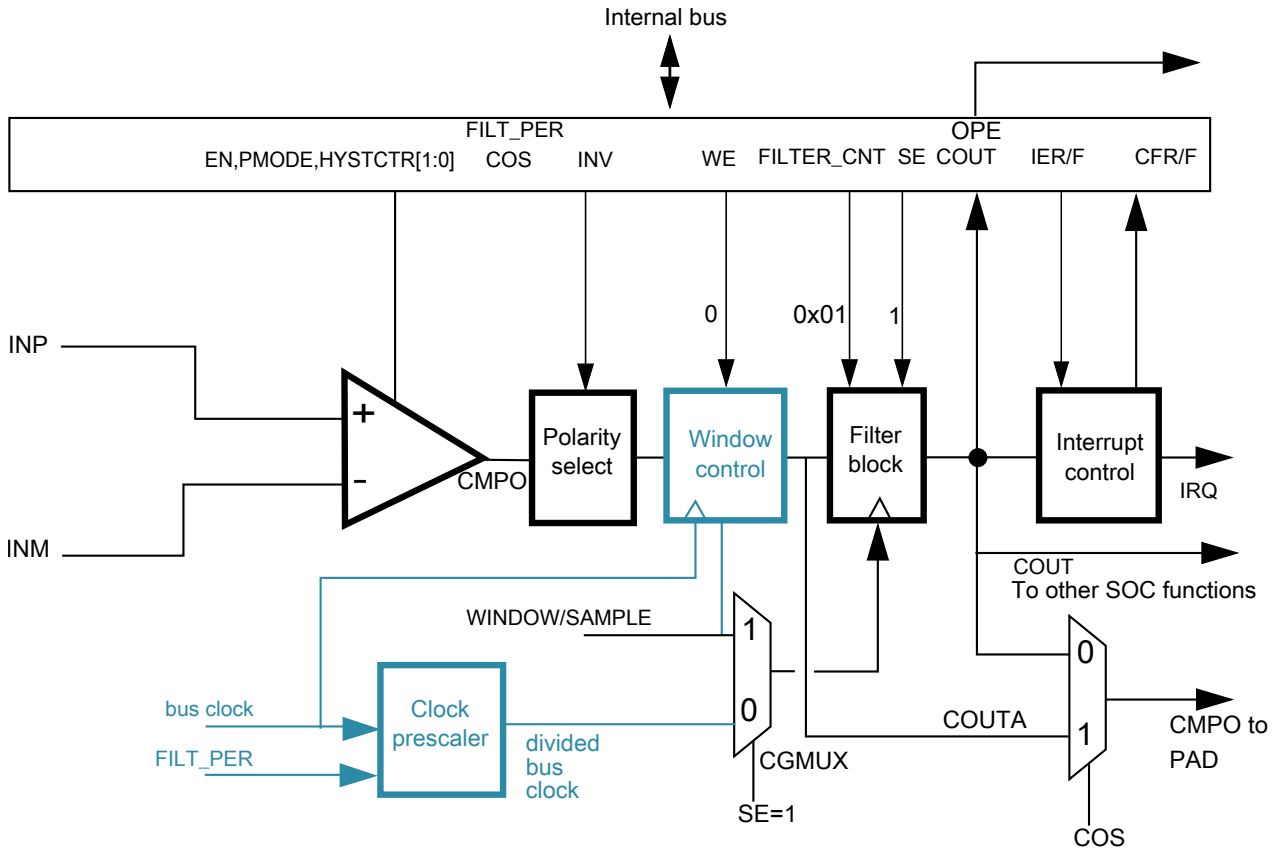


Figure 35-4. Sampled, Non-Filtered (# 3A): sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising-edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

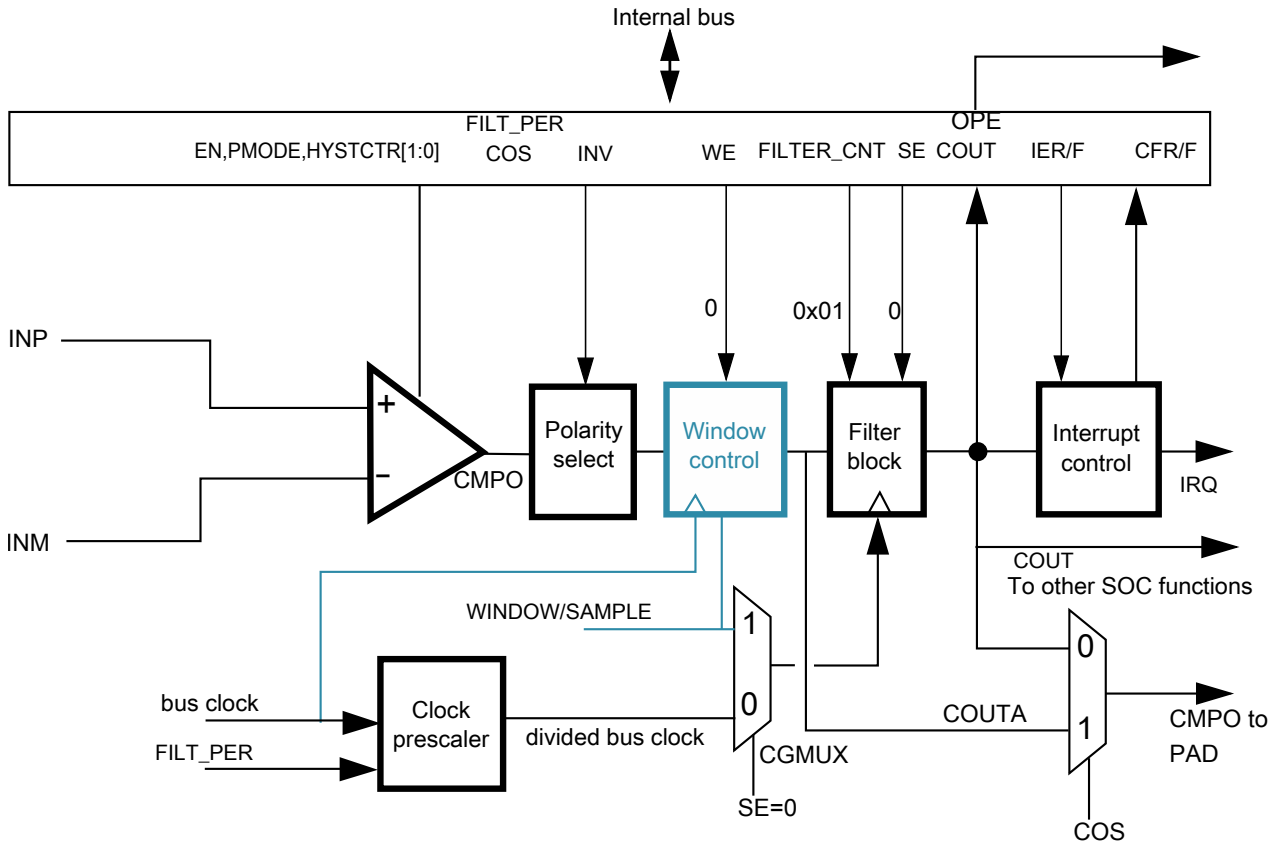


Figure 35-5. Sampled, Non-Filtered (# 3B): sampling interval internally derived

35.4.1.4 Sampled, Filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that, now, CR0[FILTER_CNT]>1, which activates filter operation.

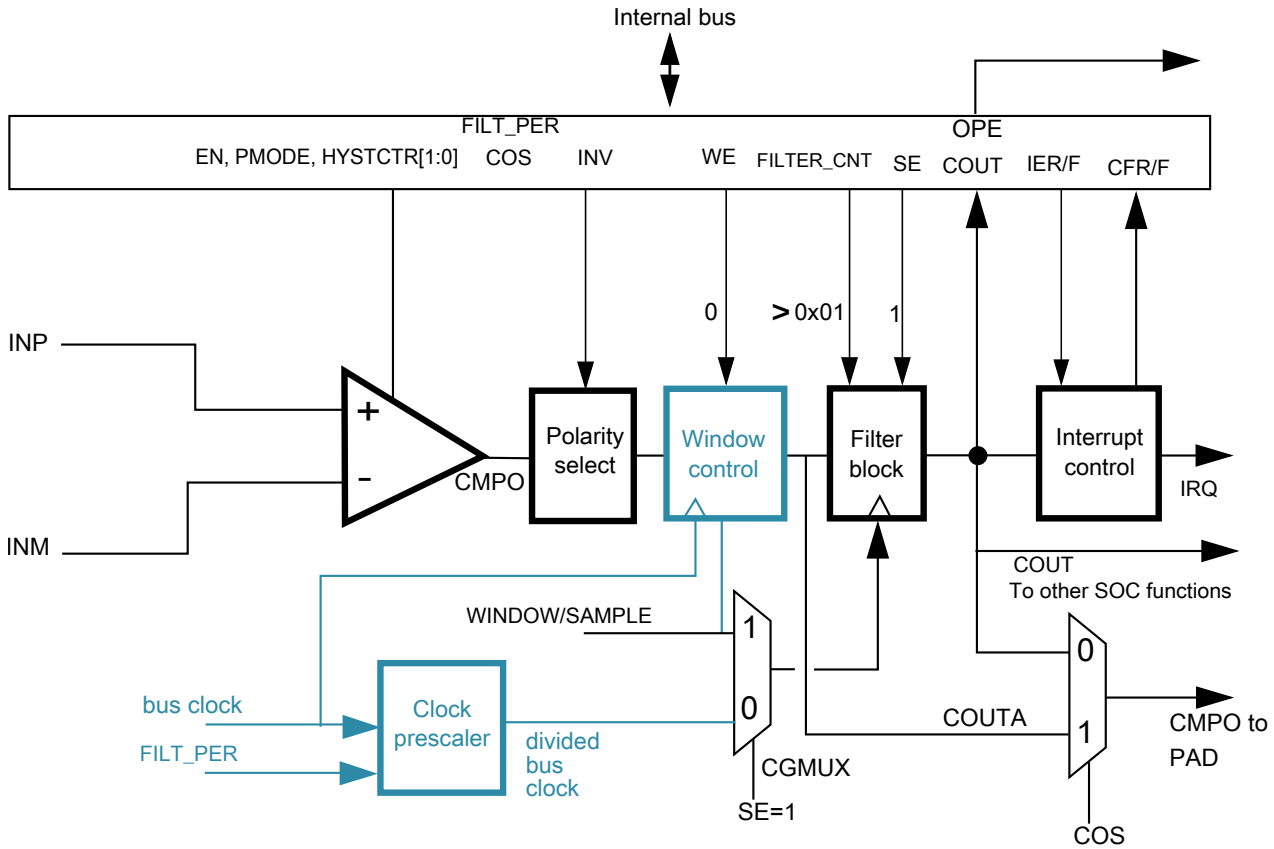


Figure 35-6. Sampled, Filtered (# 4A): sampling point externally driven

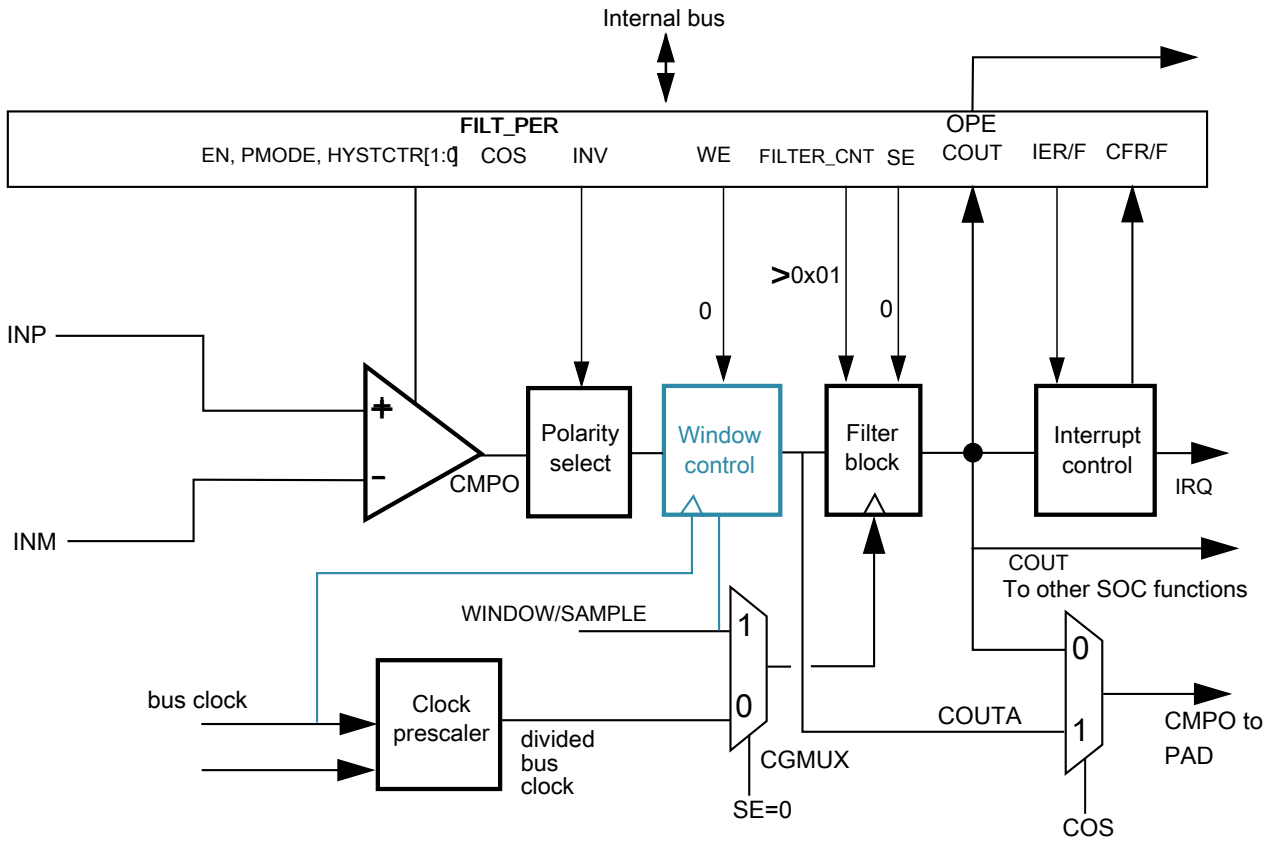


Figure 35-7. Sampled, Filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, CR0[FILTER_CNT]>1, which activates filter operation.

35.4.1.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the Windowed mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

NOTE

The analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

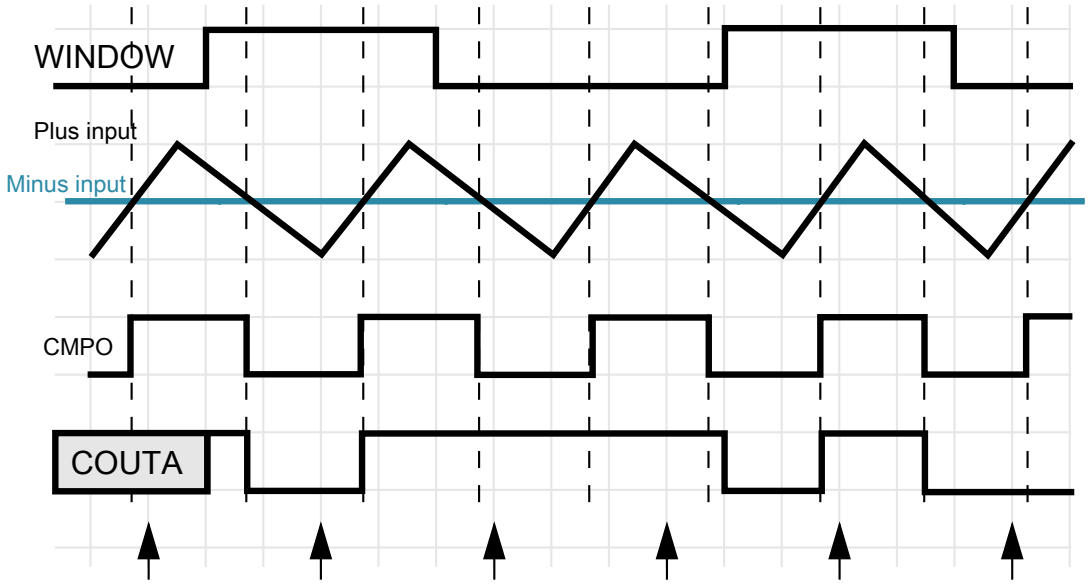


Figure 35-8. Windowed mode operation

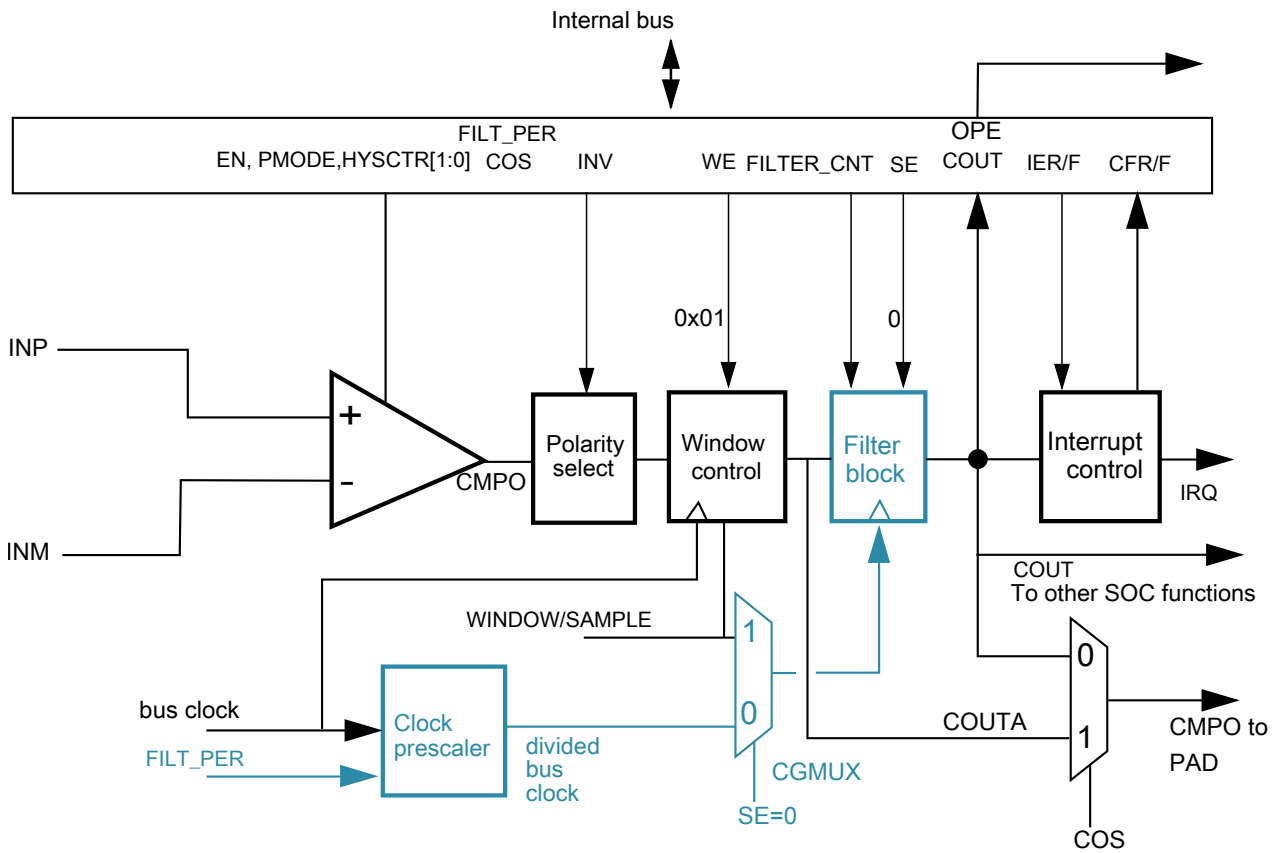


Figure 35-9. Windowed mode

For control configurations which result in disabling the filter block, see Filter Block Bypass Logic diagram.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

35.4.1.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 35-8, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows in the figure. Again, prop delays and latency are ignored for the sake of clarity.

This example was generated solely to demonstrate operation of the comparator in windowed/resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

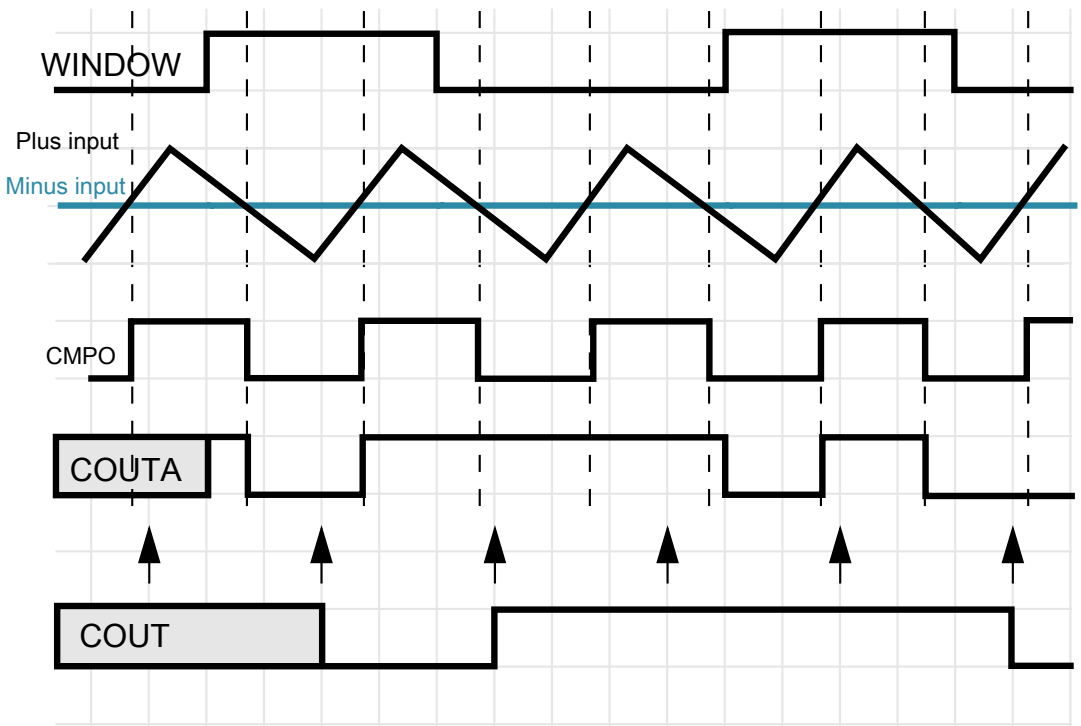


Figure 35-10. Windowed/resampled mode operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of CR0[FILTER_CNT] must be 1.

35.4.1.7 Windowed/Filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it uses both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + ((CR0[FILTER_CNT] * FPR[FILT_PER]) + 1) * bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

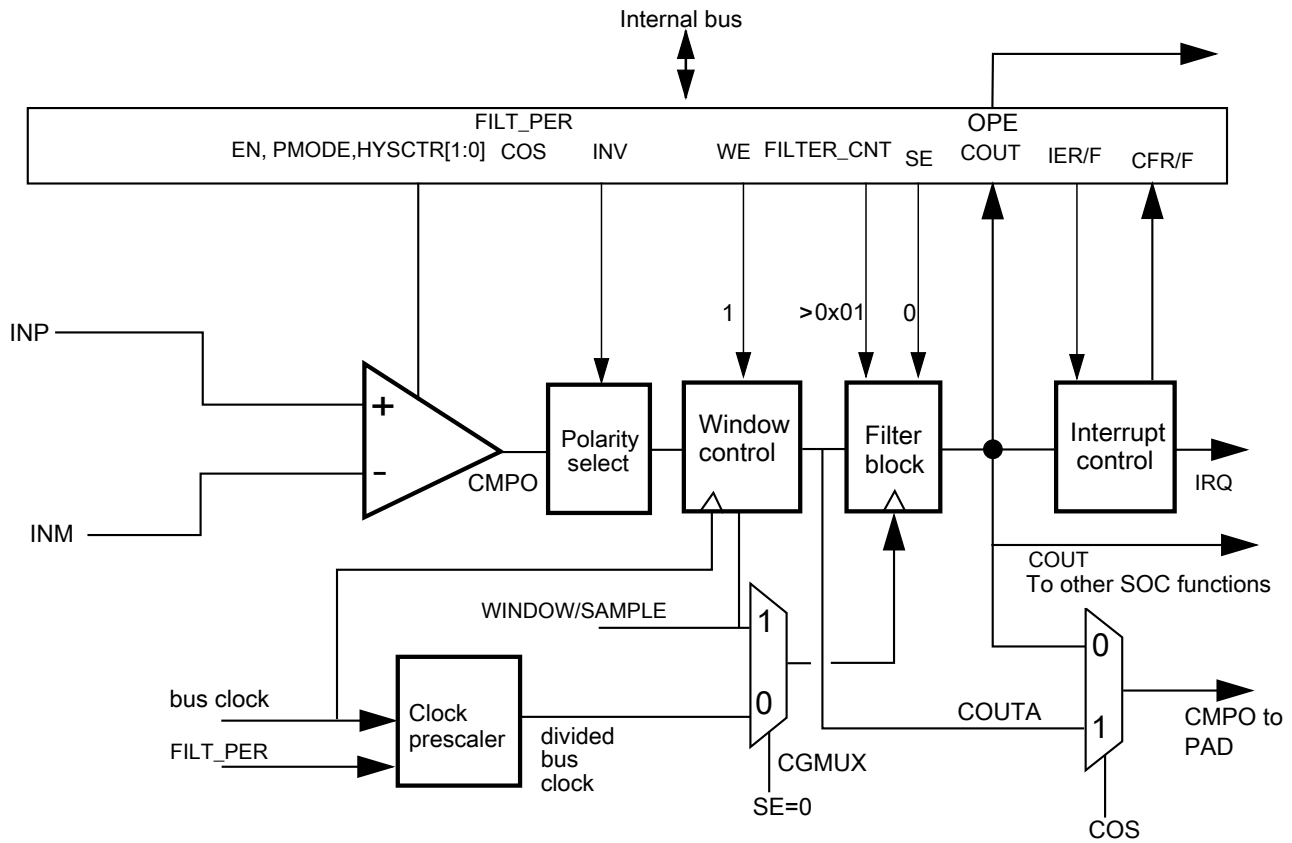


Figure 35-11. Windowed/Filtered mode

35.4.2 Power modes

35.4.2.1 Wait mode operation

During Wait and VLPW modes, the CMP, if enabled, continues to operate normally and a CMP interrupt can wake the MCU.

35.4.2.2 Stop mode operation

Depending on clock restrictions related to the MCU core or core peripherals, the MCU is brought out of stop when a compare event occurs and the corresponding interrupt is enabled. Similarly, if CR1[OPE] is enabled, the comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. In Stop modes, the comparator can be operational in both:

- High-Speed (HS) Comparison mode when CR1[PMODE] = 1
- Low-Speed (LS) Comparison mode when CR1[PMODE] = 0

It is recommended to use the LS mode to minimize power consumption.

If stop is exited with a reset, all comparator registers are put into their reset state.

35.4.2.3 Background Debug Mode Operation

When the microcontroller is in active background debug mode, the CMP continues to operate normally.

35.4.3 Startup and operation

A typical startup sequence is listed here.

- The time required to stabilize COUT will be the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function and filter. See the Data Sheets for power-on delays of the comparators. The windowing function has a maximum of one bus clock period delay. The filter delay is specified in the Low-pass filter.
- During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and SCR[CFR]/SCR[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.
- When programmed for filtering modes, COUT will initially be equal to 0, until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

35.4.4 Low-pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT.

Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by FPR[FILT_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high-frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

35.4.4.1 Enabling filter modes

Filter modes can be enabled by:

- Setting CR0[FILTER_CNT] > 0x01 and
- Setting FPR[FILT_PER] to a nonzero value or setting CR1[SE]=1

If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

The filter output will be at logic 0 when first initalized, and will subsequently change when all the consecutive CR0[FILTER_CNT] samples agree that the output value has changed. In other words, SCR[COUT] will be 0 for some initial period, even when COUTA is at logic 1.

Setting both CR1[SE] and FPR[FILT_PER] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CR0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CR0[FILTER_CNT] samples agree that the output value has changed.

35.4.4.2 Latency issues

The value of FPR[FILT_PER] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of CR0[FILTER_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CR0[FILTER_CNT].

The values of FPR[FILT_PER] or SAMPLE period and CR0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CR0[FILTER_CNT].

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

Table 35-3. Comparator sample/filter maximum latencies

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		T_{PD} + (FPR[FILT_PER] * T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CR0[FILTER_CNT] * T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] x T_{per}) + T_{per}
5A	1	1	0	0x00	X	Windowed mode	$T_{PD} + T_{per}$
5B	1	1	0	X	0x00		$T_{PD} + T_{per}$
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (FPR[FILT_PER] * T_{per}) + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] x T_{per}) + $2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

35.5 CMP interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both.

The following table gives the conditions in which the interrupt request is asserted and deasserted.

When	Then
SCR[IER] and SCR[CFR] are set	The interrupt request is asserted
SCR[IEF] and SCR[CFF] are set	The interrupt request is asserted
SCR[IER] and SCR[CFR] are cleared for a rising-edge interrupt	The interrupt request is deasserted
SCR[IEF] and SCR[CFF] are cleared for a falling-edge interrupt	The interrupt request is deasserted

35.6 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it sends a transfer completing indicator that deasserts the DMA transfer request and clears the flag to allow a subsequent change on comparator output to occur and force another DMA request.

The comparator can remain functional in STOP modes.

When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

35.7 CMP Asynchronous DMA support

The comparator can remain functional in STOP modes.

When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

35.8 Digital-to-analog converter

The figure found here shows the block diagram of the DAC module.

It contains a 64-tap resistor ladder network and a 64-to-1 multiplexer, which selects an output voltage from one of 64 distinct levels that outputs from DACO. It is controlled through the DAC Control Register (DACCR). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down or disabled when not in use. When in Disabled mode, DACO is connected to the analog ground.

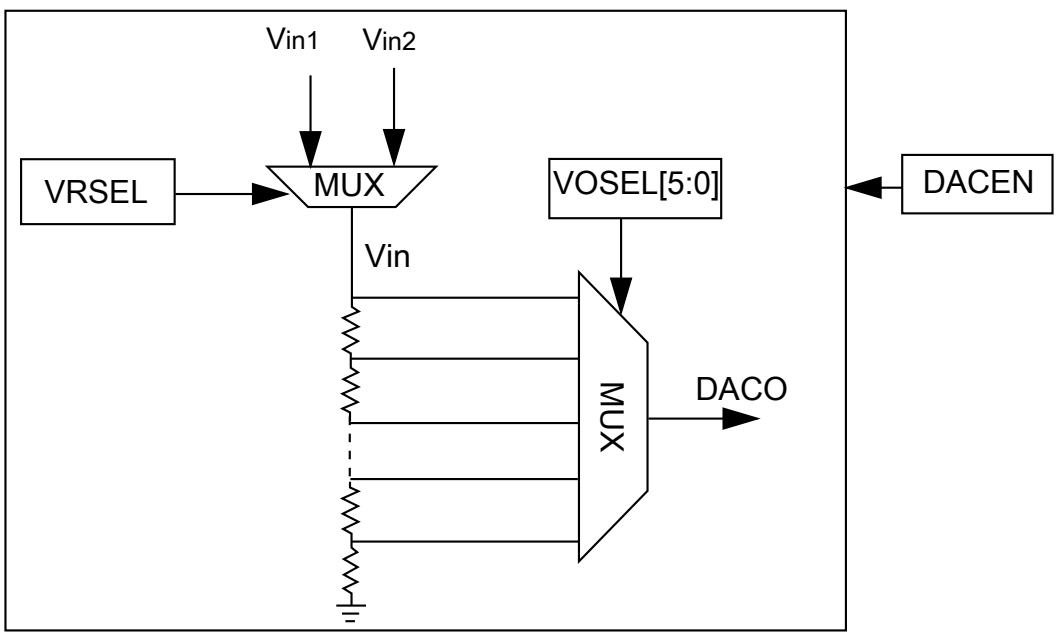


Figure 35-12. 6-bit DAC block diagram

35.9 DAC functional description

This section provides DAC functional description information.

35.9.1 Voltage reference source select

- V_{in1} connects to the primary voltage source as supply reference of 64 tap resistor ladder
- V_{in2} connects to an alternate voltage source

35.10 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

35.11 DAC clocks

This module has a single clock input, the bus clock.

35.12 DAC interrupts

This module has no interrupts.

35.13 CMP Trigger Mode

CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1.

In addition, the CMP must be enabled. If the DAC is to be used as a reference to the CMP, it must also be enabled.

CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare.

Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received.

Chapter 36
12-bit Digital-to-Analog Converter (DAC)

36.1 Chip-specific 12-bit DAC information

36.1.1 12-bit DAC Instantiation Information

This device contains one 12-bit digital-to-analog converter (DAC) with programmable reference generator output. The DAC includes a 16 deep 12bit FIFO for DMA support.

DAC0_OUT signal is connected to CMP3_IN3 and CMP1_IN3/PTE30.

DAC0_buffer_limit output signal feeds DMAUX source 45.

DAC_12B_SYNC input signals is connected to XBARA_OUT15, PDB0_DAC_trig, and PDB1_DAC_trig outputs.

36.1.2 12-bit DAC Output

The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator or ADC.

36.1.3 12-bit DAC Reference

For this device VREFH and VDDA are selectable as the DAC reference. VREFH is connected to the DACREF_1 input and VDDA is connected to the DACREF_2 input. Use DACx_C0[DACRFS] control bit to select between these two options.

NOTE

It must be noted that if the DAC and ADC use the same reference simultaneously, some degradation of ADC accuracy is to be expected due to DAC switching.

36.2 Introduction

The 12-bit digital-to-analog converter (DAC) is a low-power, general-purpose DAC. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator, op-amps, or ADC.

36.3 Features

The features of the DAC module include:

- On-chip programmable reference generator output. The voltage output range is from 1⁄4096 V_{in} to V_{in}, and the step is 1⁄4096 V_{in}, where V_{in} is the input voltage.
- V_{in} can be selected from two reference sources
- Static operation in Normal Stop mode
- 16-word data buffer supported with configurable watermark and multiple operation modes
- DMA support

36.4 Block diagram

The block diagram of the DAC module is as follows:

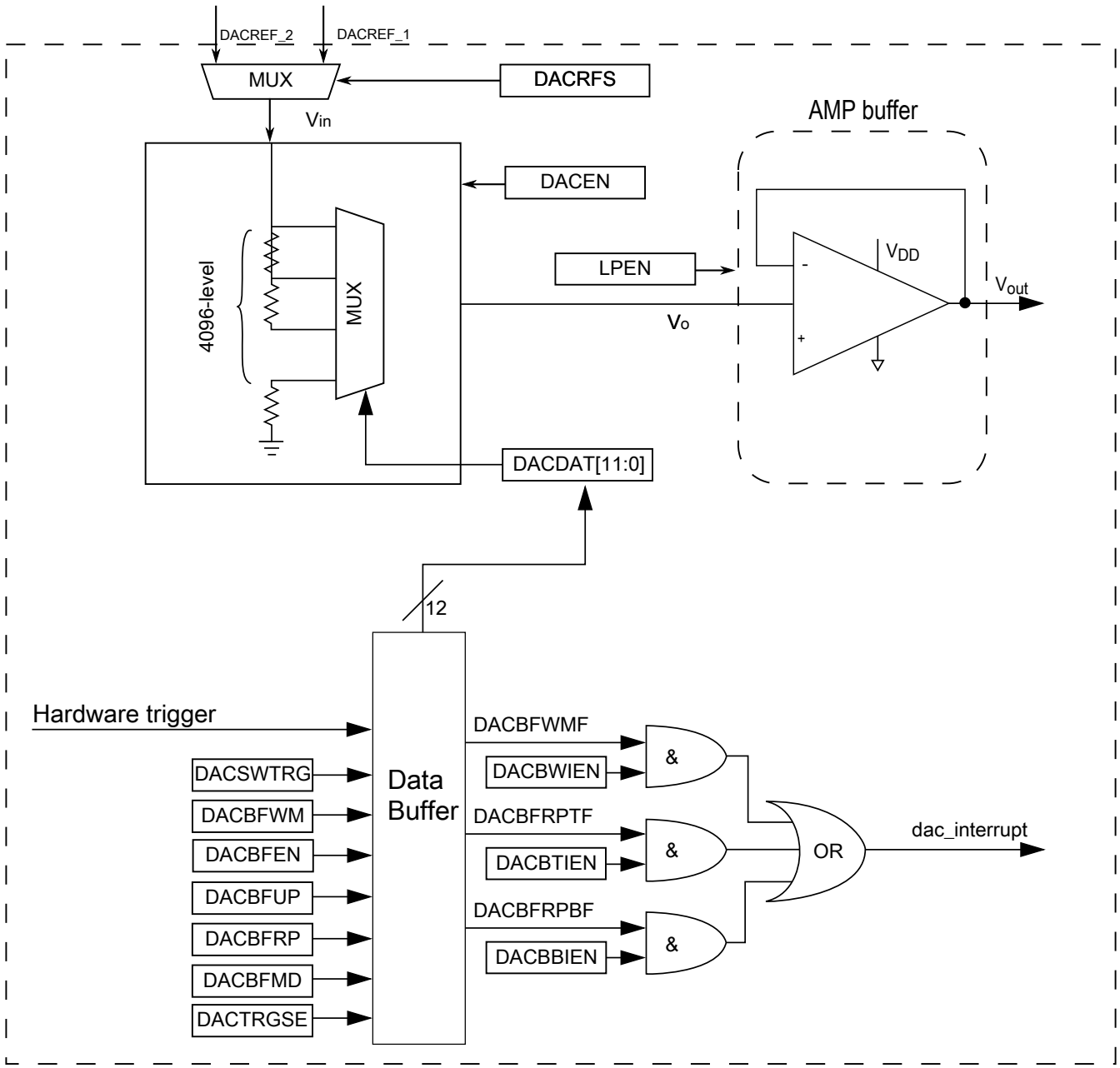


Figure 36-1. DAC block diagram

36.5 Memory map/register definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

DAC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_F000	DAC Data Low Register (DAC0_DAT0L)	8	R/W	00h	36.5.1/749
4003_F001	DAC Data High Register (DAC0_DAT0H)	8	R/W	00h	36.5.2/749
4003_F002	DAC Data Low Register (DAC0_DAT1L)	8	R/W	00h	36.5.1/749
4003_F003	DAC Data High Register (DAC0_DAT1H)	8	R/W	00h	36.5.2/749
4003_F004	DAC Data Low Register (DAC0_DAT2L)	8	R/W	00h	36.5.1/749
4003_F005	DAC Data High Register (DAC0_DAT2H)	8	R/W	00h	36.5.2/749
4003_F006	DAC Data Low Register (DAC0_DAT3L)	8	R/W	00h	36.5.1/749
4003_F007	DAC Data High Register (DAC0_DAT3H)	8	R/W	00h	36.5.2/749
4003_F008	DAC Data Low Register (DAC0_DAT4L)	8	R/W	00h	36.5.1/749
4003_F009	DAC Data High Register (DAC0_DAT4H)	8	R/W	00h	36.5.2/749
4003_F00A	DAC Data Low Register (DAC0_DAT5L)	8	R/W	00h	36.5.1/749
4003_F00B	DAC Data High Register (DAC0_DAT5H)	8	R/W	00h	36.5.2/749
4003_F00C	DAC Data Low Register (DAC0_DAT6L)	8	R/W	00h	36.5.1/749
4003_F00D	DAC Data High Register (DAC0_DAT6H)	8	R/W	00h	36.5.2/749
4003_F00E	DAC Data Low Register (DAC0_DAT7L)	8	R/W	00h	36.5.1/749
4003_F00F	DAC Data High Register (DAC0_DAT7H)	8	R/W	00h	36.5.2/749
4003_F010	DAC Data Low Register (DAC0_DAT8L)	8	R/W	00h	36.5.1/749
4003_F011	DAC Data High Register (DAC0_DAT8H)	8	R/W	00h	36.5.2/749
4003_F012	DAC Data Low Register (DAC0_DAT9L)	8	R/W	00h	36.5.1/749
4003_F013	DAC Data High Register (DAC0_DAT9H)	8	R/W	00h	36.5.2/749
4003_F014	DAC Data Low Register (DAC0_DAT10L)	8	R/W	00h	36.5.1/749
4003_F015	DAC Data High Register (DAC0_DAT10H)	8	R/W	00h	36.5.2/749
4003_F016	DAC Data Low Register (DAC0_DAT11L)	8	R/W	00h	36.5.1/749
4003_F017	DAC Data High Register (DAC0_DAT11H)	8	R/W	00h	36.5.2/749
4003_F018	DAC Data Low Register (DAC0_DAT12L)	8	R/W	00h	36.5.1/749
4003_F019	DAC Data High Register (DAC0_DAT12H)	8	R/W	00h	36.5.2/749
4003_F01A	DAC Data Low Register (DAC0_DAT13L)	8	R/W	00h	36.5.1/749
4003_F01B	DAC Data High Register (DAC0_DAT13H)	8	R/W	00h	36.5.2/749
4003_F01C	DAC Data Low Register (DAC0_DAT14L)	8	R/W	00h	36.5.1/749
4003_F01D	DAC Data High Register (DAC0_DAT14H)	8	R/W	00h	36.5.2/749
4003_F01E	DAC Data Low Register (DAC0_DAT15L)	8	R/W	00h	36.5.1/749
4003_F01F	DAC Data High Register (DAC0_DAT15H)	8	R/W	00h	36.5.2/749
4003_F020	DAC Status Register (DAC0_SR)	8	R/W	02h	36.5.3/750
4003_F021	DAC Control Register (DAC0_C0)	8	R/W	00h	36.5.4/751
4003_F022	DAC Control Register 1 (DAC0_C1)	8	R/W	00h	36.5.5/752
4003_F023	DAC Control Register 2 (DAC0_C2)	8	R/W	0Fh	36.5.6/753

36.5.1 DAC Data Low Register (DAC*x*_DAT*n*L)

Address: 4003_F000h base + 0h offset + (2d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read Write	DATA0							
Reset	0	0	0	0	0	0	0	0

DAC*x*_DAT*n*L field descriptions

Field	Description
DATA0	DATA0 When the DAC buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula: $V_{out} = V_{in} * (1 + DACDAT0[11:0])/4096$ When the DAC buffer is enabled, DATA is mapped to the 16-word buffer.

36.5.2 DAC Data High Register (DAC*x*_DAT*n*H)

Address: 4003_F000h base + 1h offset + (2d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	0				DATA1			
Write								
Reset	0	0	0	0	0	0	0	0

DAC*x*_DAT*n*H field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DATA1	DATA1 When the DAC Buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula. $V_{out} = V_{in} * (1 + DACDAT0[11:0])/4096$ When the DAC buffer is enabled, DATA[11:0] is mapped to the 16-word buffer.

36.5.3 DAC Status Register (DACx_SR)

If DMA is enabled, the flags can be cleared automatically by DMA when the DMA request is done. Writing 0 to a field clears it whereas writing 1 has no effect. After reset, DACBFRPTF is set and can be cleared by software, if needed. The flags are set only when the data buffer status is changed.

Address: 4003_F000h base + 20h offset = 4003_F020h

Bit	7	6	5	4	3	2	1	0
Read	0					DACBFWMF	DACBFRPTF	DACBFRPBF
Write								
Reset	0	0	0	0	0	0	1	0

DACx_SR field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 DACBFWMF	DAC Buffer Watermark Flag This bit is set if the remaining FIFO data is less than the watermark setting. It is cleared automatically by writing data into FIFO by DMA or CPU. Write to this bit is ignored in FIFO mode. 0 The DAC buffer read pointer has not reached the watermark level. 1 The DAC buffer read pointer has reached the watermark level.
1 DACBFRPTF	DAC Buffer Read Pointer Top Position Flag In FIFO mode, it is FIFO nearly empty flag. It is set when only one data remains in FIFO. Any DAC trigger does not increase the Read Pointer if this bit is set to avoid any possible glitch or abrupt change at DAC output. It is cleared automatically if FIFO is not empty. 0 The DAC buffer read pointer is not zero. 1 The DAC buffer read pointer is zero.
0 DACBFRPBF	DAC Buffer Read Pointer Bottom Position Flag In FIFO mode, it is FIFO FULL status bit. It means FIFO read pointer equals Write Pointer because of Write Pointer increase. If this bit is set, any write to FIFO from either DMA or CPU is ignored by DAC. It is cleared if there is any DAC trigger making the DAC read pointer increase. Write to this bit is ignored in FIFO mode. 0 The DAC buffer read pointer is not equal to C2[DACBFUP]. 1 The DAC buffer read pointer is equal to C2[DACBFUP].

36.5.4 DAC Control Register (DACx_C0)

Address: 4003_F000h base + 21h offset = 4003_F021h

Bit	7	6	5	4	3	2	1	0
Read	DACEN	DACRFS	DACTRGSEL	0	LPEN	DACBWIEN	DACBTIEN	DACBBIEN
Write				DACSWTRG				
Reset	0	0	0	0	0	0	0	0

DACx_C0 field descriptions

Field	Description
7 DACEN	DAC Enable Starts the Programmable Reference Generator operation. 0 The DAC system is disabled. 1 The DAC system is enabled.
6 DACRFS	DAC Reference Select 0 The DAC selects DACREF_1 as the reference voltage. 1 The DAC selects DACREF_2 as the reference voltage.
5 DACTRGSEL	DAC Trigger Select 0 The DAC hardware trigger is selected. 1 The DAC software trigger is selected.
4 DACSWTRG	DAC Software Trigger Active high. This is a write-only field, which always reads 0. If DAC software trigger is selected and buffer is enabled, writing 1 to this field will advance the buffer read pointer once. 0 The DAC soft trigger is not valid. 1 The DAC soft trigger is valid.
3 LPEN	DAC Low Power Control **NOTE:** See the 12-bit DAC electrical characteristics of the device data sheet for details on the impact of the modes below. 0 High-Power mode 1 Low-Power mode
2 DACBWIEN	DAC Buffer Watermark Interrupt Enable 0 The DAC buffer watermark interrupt is disabled. 1 The DAC buffer watermark interrupt is enabled.
1 DACBTIEN	DAC Buffer Read Pointer Top Flag Interrupt Enable 0 The DAC buffer read pointer top flag interrupt is disabled. 1 The DAC buffer read pointer top flag interrupt is enabled.
0 DACBBIEN	DAC Buffer Read Pointer Bottom Flag Interrupt Enable

Table continues on the next page...

DACx_C0 field descriptions (continued)

Field	Description
	0 The DAC buffer read pointer bottom flag interrupt is disabled. 1 The DAC buffer read pointer bottom flag interrupt is enabled.

36.5.5 DAC Control Register 1 (DACx_C1)

Address: 4003_F000h base + 22h offset = 4003_F022h

Bit	7	6	5	4	3	2	1	0
Read	DMAEN	0		DACBFWM		DACBFMD		DACBFEN
Write								
Reset	0	0	0	0	0	0	0	0

DACx_C1 field descriptions

Field	Description
7 DMAEN	DMA Enable Select 0 DMA is disabled. 1 DMA is enabled. When DMA is enabled, the DMA request will be generated by original interrupts. The interrupts will not be presented on this module at the same time.
6–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4–3 DACBFWM	DAC Buffer Watermark Select In normal mode it controls when SR[DACBFWMF] is set. When the DAC buffer read pointer reaches the word defined by this field, which is 1–4 words away from the upper limit (DACBUP), SR[DACBFWMF] will be set. This allows user configuration of the watermark interrupt. In FIFO mode, it is FIFO watermark select field. 00 In normal mode, 1 word . In FIFO mode, 2 or less than 2 data remaining in FIFO will set watermark status bit. 01 In normal mode, 2 words . In FIFO mode, Max/4 or less than Max/4 data remaining in FIFO will set watermark status bit. 10 In normal mode, 3 words . In FIFO mode, Max/2 or less than Max/2 data remaining in FIFO will set watermark status bit. 11 In normal mode, 4 words . In FIFO mode, Max-2 or less than Max-2 data remaining in FIFO will set watermark status bit.
2–1 DACBFMD	DAC Buffer Work Mode Select 00 Normal mode 01 Swing mode 10 One-Time Scan mode 11 FIFO mode
0 DACBFEN	DAC Buffer Enable 0 Buffer read pointer is disabled. The converted data is always the first word of the buffer. 1 Buffer read pointer is enabled. The converted data is the word that the read pointer points to. It means converted data can be from any word of the buffer.

36.5.6 DAC Control Register 2 (DAC*x*_C2)

Address: 4003_F000h base + 23h offset = 4003_F023h

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

DAC*x*_C2 field descriptions

Field	Description
7–4 DACBFRP	DAC Buffer Read Pointer In normal mode it keeps the current value of the buffer read pointer. FIFO mode, it is the FIFO read pointer. It is writable in FIFO mode. User can configure it to same address to reset FIFO as empty.
DACBFUP	DAC Buffer Upper Limit In normal mode it selects the upper limit of the DAC buffer. The buffer read pointer cannot exceed it. In FIFO mode it is the FIFO write pointer. User cannot set Buffer Up limit in FIFO mode. In Normal mode its reset value is MAX. When IP is configured to FIFO mode, this register becomes Write_Pointer, and its value is initially set to equal READ_POINTER automatically, and the FIFO status is empty. It is writable and user can configure it to the same address to reset FIFO as empty.

36.6 Functional description

The 12-bit DAC module can select one of the two reference inputs—DACREF_1 and DACREF_2 as the DAC reference voltage, V_{in} by C0 [DACRFS]. See the chip-specific DAC information to determine the source options for DACREF_1 and DACREF_2.

When the DAC is enabled, it converts the data in DACDAT0[11:0] or the data from the DAC data buffer to a stepped analog output voltage. The output voltage range is from V_{in} to $V_{in}/4096$, and the step is $V_{in}/4096$.

36.6.1 DAC data buffer operation

When the DAC is enabled and the buffer is not enabled, the DAC module always converts the data in DAT0 to the analog output voltage.

When both the DAC and the buffer are enabled, the DAC converts the data in the data buffer to analog output voltage. The data buffer read pointer advances to the next word whenever a hardware or software trigger event occurs.

The data buffer can be configured to operate in Normal mode, Swing mode, One-Time Scan mode or FIFO mode. When the buffer operation is switched from one mode to another, the read pointer does not change. The read pointer can be set to any value between 0 and C2[DACBFUP] by writing C2[DACBFRP].

36.6.1.1 DAC data buffer interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer. SR[DACBFRPBF] is set when the DAC buffer read pointer reaches the DAC buffer upper limit, that is, C2[DACBFRP] = C2[DACBFUP]. SR[DACBFRPTF] is set when the DAC read pointer is equal to the start position, 0. Finally, SR[DACBFWMF] is set when the DAC buffer read pointer has reached the position defined by C1[DACBFWM]. C1[DACBFWM] can be used to generate an interrupt when the DAC buffer read pointer is between 1 to 4 words from C2[DACBFUP].

36.6.1.2 Modes of DAC data buffer operation

The following table describes the different modes of data buffer operation for the DAC module.

Table 36-1. Modes of DAC data buffer operation

Modes	Description
Buffer Normal mode	This is the default mode. The buffer works as a circular buffer. The read pointer increases by one, every time the trigger occurs. When the read pointer reaches the upper limit, it goes to 0 directly in the next trigger event.
Buffer Swing mode	This mode is similar to the normal mode. However, when the read pointer reaches the upper limit, it does not go to 0. It will descend by 1 in the next trigger events until 0 is reached.
Buffer One-time Scan mode	The read pointer increases by 1 every time the trigger occurs. When it reaches the upper limit, it stops there. If read pointer is reset to the address other than the upper limit, it will increase to the upper address and stop there again. **NOTE:** If the software set the read pointer to the upper limit, the read pointer will not advance in this mode.
FIFO Mode	In FIFO mode, the buffer is organized as a FIFO. For a valid write to any DACDATx, the data is put into the FIFO, and the write pointer is automatically incremented. The module is connected internally to a 32bit interface. For any 16bit or 8bit FIFO access, address bit[1] needs to be 0; otherwise, the write is ignored. For any 32bit FIFO access, the Write_Pointer needs to be an EVEN number; otherwise, the write is ignored.

Table 36-1. Modes of DAC data buffer operation

Modes	Description
	NOTE: A successful 32bit FIFO write will increase the write pointer by 2. Any write will cause the FIFO over-flow will be ignored, the cases includes: 1.FIFO is full, the write will be ignored. 2.FIFO is nearly full (FIFO_SIZE-1), 32bit write will be ignored. **NOTE:** For 8bit write, address bit[0] determine which byte lane will be written to the FIFO according to little endian alignment. Only both byte lanes are written will the write pointer increase. User need to make sure 8bit access happened in pair and both upper & lower bytes are written. There is no requirement on which byte write first. In FIFO mode, there is no change to read access of DACDATx (from normal mode), read to DACDATx will return the DATA addressed by the access address to the data buffer, and both write pointer and read pointer in FIFO mode will NOT be changed by read access. FIFO write can be happened when DAC is not enabled for 1st data conversion enable. But FIFO mode need to work at buffer Enabled at DACC1[DACBFEN]. In FIFO mode, the DATA BUF will be organized as FIFO.

36.6.2 DMA operation

When DMA is enabled, DMA requests are generated instead of interrupt requests. The DMA Done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

36.6.3 Resets

During reset, the DAC is configured in the default mode and is disabled.

36.6.4 Low-Power mode operation

The following table shows the wait mode and the stop mode operation of the DAC module.

Table 36-2. Modes of operation

Modes of operation	Description
Wait mode	The DAC will operate normally, if enabled.
Stop mode	In low-power stop modes, the DAC is fully shut down.

NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

Chapter 37
Pulse Width Modulator A (PWMA/eFlexPWM)

37.1 Chip-specific eFlexPWM information

37.1.1 eFlexPWM Inputs

For this chip the PWMA has the following signal input connections.

Input synchronisation to the four submodules of the PWMA are fed from the XBARA module, which provides flexible triggering and fault inputs from other peripherals.

PWM0 signal	Connected to
PWM0_EXT0A	XBARA_OUT20
PWM0_EXT1A	XBARA_OUT21
PWM0_EXT2A	XBARA_OUT22
PWM0_EXT3A	XBARA_OUT23
PWM0_EXT_SYNC0	XBARA_OUT24
PWM0_EXT_SYNC1	XBARA_OUT25
PWM0_EXT_SYNC2	XBARA_OUT26
PWM0_EXT_SYNC3	XBARA_OUT27
PWM0_EXT_CLK	XBARA_OUT28
PWM0_FAULT0	XBARA_OUT29
PWM0_FAULT1	XBARA_OUT30
PWM0_FAULT2	XBARA_OUT31
PWM0_FAULT3	XBARA_OUT32
PWM0_FORCE	XBARA_OUT33
PWM0_EXT0B	an0_pwm (ADC output)
PWM0_EXT1B	an1_pwm (ADC output)
PWM0_EXT2B	an2_pwm (ADC output)
PWM0_EXT3B	an3_pwm (ADC output)

37.1.2 eFlexPWM Outputs

For this chip the PWMA has the following signal output connections.

Output trigger events from the PWMA are used to trigger other peripherals to take action via the XBARA and XBARB and AOI module. Each of the four sub modules have two trigger outputs (TRIG0) and TRIG1).

- PWM0_OUT_TRIG00 output is connected to XBARA_IN20 and XBARB_IN8
- PWM0_OUT_TRIG01 output is connected to XBARA_IN21 and XBARB_IN8
- PWM0_OUT_TRIG10 output is connected to XBARA_IN22 and XBARB_IN9
- PWM0_OUT_TRIG11 output is connected to XBARA_IN23 and XBARB_IN9
- PWM0_OUT_TRIG20 output is connected to XBARA_IN24 and XBARB_IN10
- PWM0_OUT_TRIG21 output is connected to XBARA_IN25 and XBARB_IN10
- PWM0_OUT_TRIG30 output is connected to XBARA_IN26 and XBARB_IN11
- PWM0_OUT_TRIG31 output is connected to XBARA_IN27 and XBARB_IN11

Each submodule of the PWMA can request a DMA transfer for updating their double buffered VALx registers.

- Submodule 0 , PWM0_WR0 output is connected to DMAMUX0 ch 6
- Submodule 1 , PWM0_WR1 output is connected to DMAMUX0 ch 7
- Submodule 2 , PWM0_WR2 output is connected to DMAMUX0 ch 8
- Submodule 3 , PWM0_WR3 output is connected to DMAMUX0 ch 9

Each submodule of the PWMA can request a DMA transfer to read VALx registers for captured values when configured as input captures.

- Submodule 0, PWM0_CP0 output is connected to DMAMUX0 ch 10
- Submodule 1, PWM0_CP1 output is connected to DMAMUX0 ch 11
- Submodule 2, PWM0_CP2 output is connected to DMAMUX0 ch 12
- Submodule 3, PWM0_CP3 output is connected to DMAMUX0 ch 13

37.2 Introduction

The pulse width modulator (PWM) module contains PWM submodules, each of which is set up to control a single half-bridge power stage. Fault channel support is provided.

This PWM module can generate various switching patterns, including highly sophisticated waveforms. It can be used to control all known Switched Mode Power Supplies (SMPS) topologies.

37.2.1 Features

- 16 bits of resolution for center, edge-aligned, and asymmetrical PWMs
- Fractional PWM clock generation for enhanced resolution of the PWM period and duty cycle
- Dithering to simulate enhanced resolution when fine edge placement is not available
- PWM outputs that can operate as complementary pairs or independent channels
- Ability to accept signed numbers for PWM generation
- Independent control of both edges of each PWM output
- Support for synchronization to external hardware or other PWM
- Double buffered PWM registers
  - Integral reload rates from 1 to 16
  - Half cycle reload capability
- Multiple output trigger events can be generated per PWM cycle via hardware
- Support for double switching PWM outputs
- Fault inputs can be assigned to control multiple PWM outputs
- Programmable filters for fault inputs
- Independently programmable PWM output polarity
- Independent top and bottom deadtime insertion
- Each complementary pair can operate with its own PWM frequency and deadtime values
- Individual software control for each PWM output
- All outputs can be programmed to change simultaneously via a FORCE_OUT event
- PWM_X pin can optionally output a third PWM signal from each submodule
- Channels not used for PWM generation can be used for buffered output compare functions
- Channels not used for PWM generation can be used for input capture functions
- Enhanced dual edge capture functionality
- The option to supply the source for each complementary PWM signal pair from any of the following:
  - Crossbar module outputs
  - External ADC input, taking into account values set in ADC high and low limit registers

37.2.2 Modes of Operation

Be careful when using this module in stop, wait and debug operating modes.

CAUTION

Some applications require regular software updates for proper operation. Failure to provide regular software updates could result in destroying the hardware setup.

To accommodate this situation, PWM outputs are placed in their inactive states in stop mode, and they can optionally be placed in inactive states in wait and debug (EOnCE) modes. PWM outputs are reactivated (assuming they were active beforehand) when these modes are exited.

Table 37-1. Modes when PWM Operation is Restricted

Mode	Description
Stop	PWM outputs are inactive.
Wait	PWM outputs are driven or inactive as a function of CTRL2[WAITEN].
Debug	CPU and peripheral clocks continue to run, but the CPU may stall for periods of time. PWM outputs are driven or inactive as a function of CTRL2[DBGEN].

37.2.3 Block Diagram

The following figure is a block diagram of the PWM.

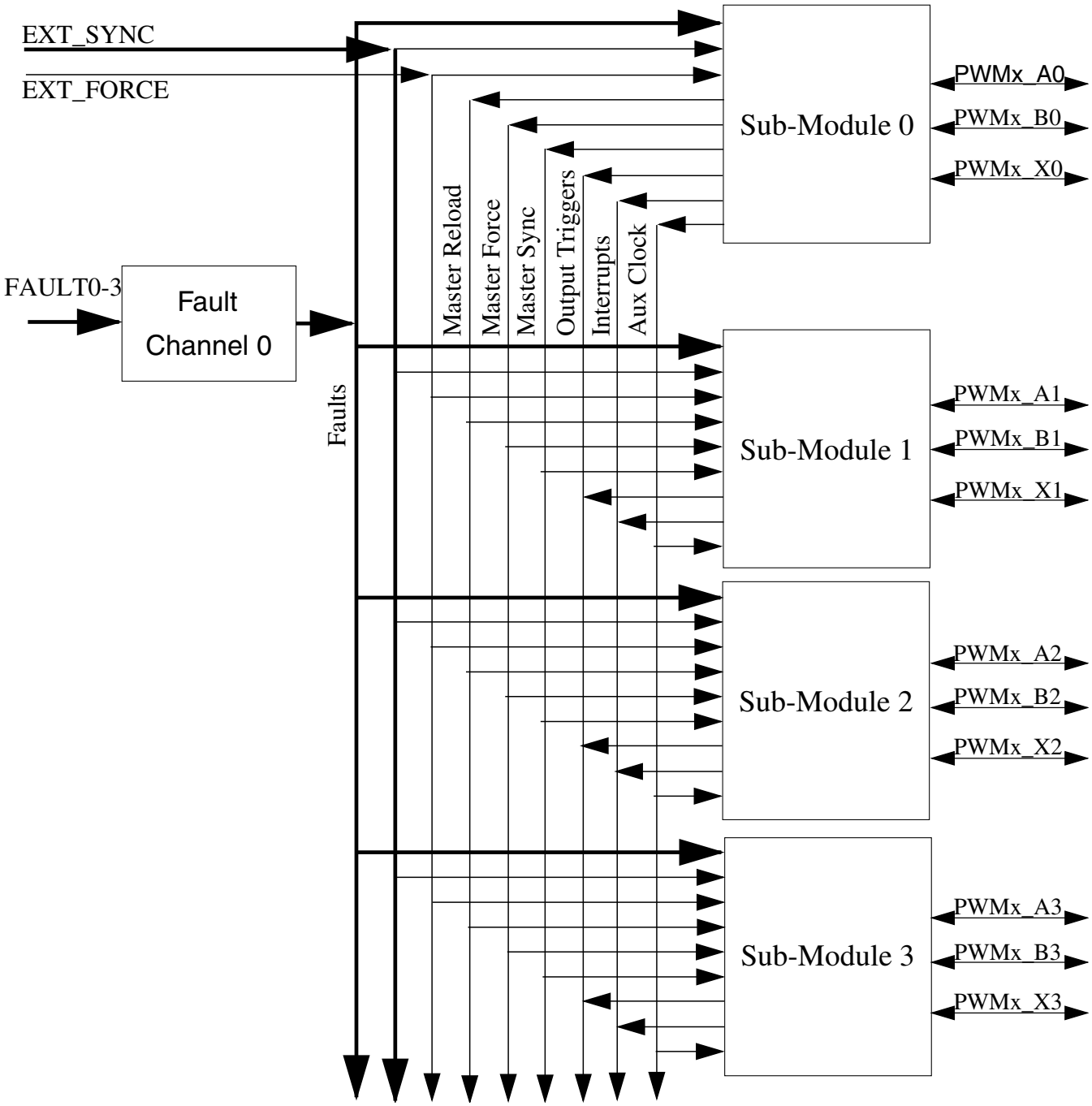


Figure 37-1. PWM Block Diagram

37.2.3.1 PWM Submodule

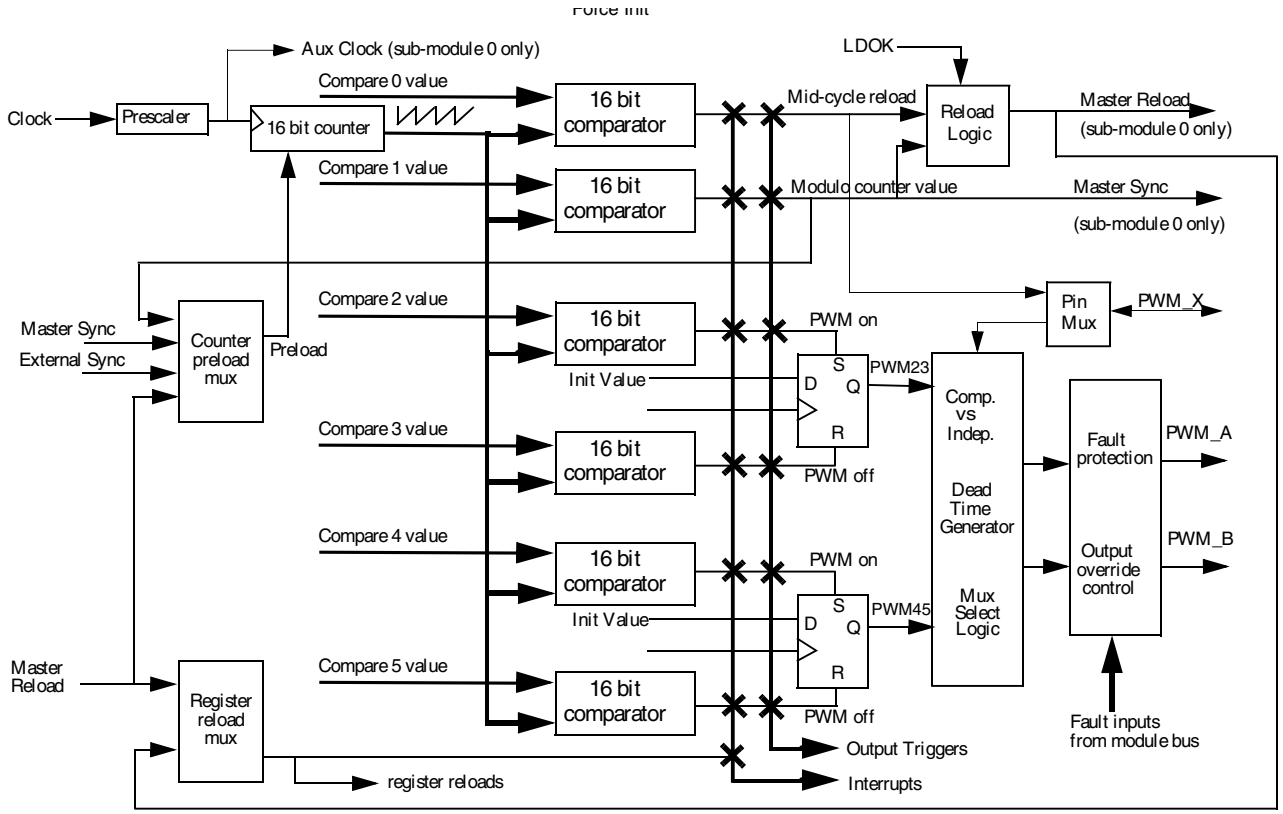


Figure 37-2. PWM Submodule Block Diagram

37.3 Signal Descriptions

The PWM has pins named PWM[n]_A, PWM[n]_B, PWM[n]_X, FAULT[n], PWM[n]_EXT_SYNC, EXT_FORCE, PWM[n]_EXTA, and PWM[n]_EXTB. The PWM also has an on-chip input called EXT_CLK and output signals called PWM[n]_OUT_TRIGx.

37.3.1 PWM[n]_A and PWM[n]_B - External PWM Output Pair

These pins are the output pins of the PWM channels. These pins can be independent PWM signals or a complementary pair. When not needed as an output, they can be used as inputs to the input capture circuitry.

37.3.2 PWM[n]_X - Auxiliary PWM Output signal

These pins are the auxiliary output pins of the PWM channels. They can be independent PWM signals. When not needed as an output, they can be used as inputs to the input capture circuitry or used to detect the polarity of the current flowing through the complementary circuit at deadtime correction.

37.3.3 FAULT[n] - Fault Inputs

These are input pins for disabling selected PWM outputs.

37.3.4 PWM[n]_EXT_SYNC - External Synchronization Signal

These input signals allow a source external to the PWM to initialize the PWM counter. In this manner, the PWM can be synchronized to external circuitry.

37.3.5 EXT_FORCE - External Output Force Signal

This input signal allows a source external to the PWM to force an update of the PWM outputs. In this manner, the PWM can be synchronized to external circuitry.

37.3.6 PWM[n]_EXTA and PWM[n]_EXTB - Alternate PWM Control Signals

These pins allow an alternate source to control the PWM_A and PWM_B outputs. Typically, either the PWM_EXTA or PWM_EXTB input (depending on the state of MCTRL[IPOL]) is used for the generation of a complementary pair. Typical control signals include ADC conversion high/low limits, TMR outputs, GPIO inputs, and comparator outputs.

37.3.7 PWM[n]_OUT_TRIG0 and PWM[n]_OUT_TRIG1 - Output Triggers

These outputs allow the PWM submodules to control timing of ADC conversions. See the description of the Output Trigger Control Register for information about how to enable these outputs and how the compare registers match up to the output triggers.

37.3.8 EXT_CLK - External Clock Signal

This signal allows a source external to the PWM (typically a timer or an off-chip source) to control the PWM clocking. In this manner, the PWM can be synchronized to the timer, or multiple chips can be synchronized to each other.

37.4 Memory Map and Registers

The address of a register is the sum of a base address and an address offset. The base address is defined at the core level, and the address offset is defined at the module level. The PWM module has a set of registers for each PWM submodule, for the configuration logic, and for each fault channel. While the registers are 16-bit wide, they can be accessed in pairs as 32-bit registers.

Submodule registers are repeated for each PWM submodule. To designate which submodule they are in, register names are prefixed with SM0, SM1, SM2, and SM3. The base address of submodule 0 is the same as the base address for the PWM module as a whole. The base address of submodule 1 is offset $60 from the base address for the PWM module as a whole. This $60 offset is based on the number of registers in a submodule. The base address of submodule 2 is equal to the base address of submodule 1 plus this same $60 offset. The pattern repeats for the base address of submodule 3.

The base address of the configuration registers is equal to the base address of the PWM module as a whole plus an offset of $180.

The base address of fault channel is equal to the base address of the PWM module as a whole plus an offset of $18C. Each of the four fields in the fault channel registers corresponds to fault inputs 3-0.

PWMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_3000	Counter Register (PWMA_SM0CNT)	16	R	0000h	37.4.1/772
4003_3002	Initial Count Register (PWMA_SM0INIT)	16	R/W	0000h	37.4.2/772
4003_3004	Control 2 Register (PWMA_SM0CTRL2)	16	R/W	0000h	37.4.3/773
4003_3006	Control Register (PWMA_SM0CTRL)	16	R/W	0400h	37.4.4/775
4003_300A	Value Register 0 (PWMA_SM0VAL0)	16	R/W	0000h	37.4.5/778
4003_300C	Fractional Value Register 1 (PWMA_SM0FRACVAL1)	16	R/W	0000h	37.4.6/778

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_300E	Value Register 1 (PWMA_SM0VAL1)	16	R/W	0000h	37.4.7/779
4003_3010	Fractional Value Register 2 (PWMA_SM0FRACVAL2)	16	R/W	0000h	37.4.8/779
4003_3012	Value Register 2 (PWMA_SM0VAL2)	16	R/W	0000h	37.4.9/780
4003_3014	Fractional Value Register 3 (PWMA_SM0FRACVAL3)	16	R/W	0000h	37.4.10/ 780
4003_3016	Value Register 3 (PWMA_SM0VAL3)	16	R/W	0000h	37.4.11/ 781
4003_3018	Fractional Value Register 4 (PWMA_SM0FRACVAL4)	16	R/W	0000h	37.4.12/ 781
4003_301A	Value Register 4 (PWMA_SM0VAL4)	16	R/W	0000h	37.4.13/ 782
4003_301C	Fractional Value Register 5 (PWMA_SM0FRACVAL5)	16	R/W	0000h	37.4.14/ 782
4003_301E	Value Register 5 (PWMA_SM0VAL5)	16	R/W	0000h	37.4.15/ 783
4003_3020	Fractional Control Register (PWMA_SM0FRCTRL)	16	R/W	0000h	37.4.16/ 783
4003_3022	Output Control Register (PWMA_SM0OCTRL)	16	R/W	0000h	37.4.17/ 785
4003_3024	Status Register (PWMA_SM0STS)	16	w1c	0000h	37.4.18/ 787
4003_3026	Interrupt Enable Register (PWMA_SM0INTEN)	16	R/W	0000h	37.4.19/ 788
4003_3028	DMA Enable Register (PWMA_SM0DMAEN)	16	R/W	0000h	37.4.20/ 790
4003_302A	Output Trigger Control Register (PWMA_SM0TCTRL)	16	R/W	0000h	37.4.21/ 791
4003_302C	Fault Disable Mapping Register 0 (PWMA_SM0DISMAP0)	16	R/W	FFFFh	37.4.22/ 792
4003_3030	Deadtime Count Register 0 (PWMA_SM0DTCNT0)	16	R/W	07FFh	37.4.23/ 793
4003_3032	Deadtime Count Register 1 (PWMA_SM0DTCNT1)	16	R/W	07FFh	37.4.24/ 794
4003_3034	Capture Control A Register (PWMA_SM0CAPTCTRLA)	16	R/W	0000h	37.4.25/ 794
4003_3036	Capture Compare A Register (PWMA_SM0CAPTCOMPA)	16	R/W	0000h	37.4.26/ 796
4003_3038	Capture Control B Register (PWMA_SM0CAPTCTRLB)	16	R/W	0000h	37.4.27/ 797
4003_303A	Capture Compare B Register (PWMA_SM0CAPTCOMPB)	16	R/W	0000h	37.4.28/ 798
4003_303C	Capture Control X Register (PWMA_SM0CAPTCTRLX)	16	R/W	0000h	37.4.29/ 799

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_303E	Capture Compare X Register (PWMA_SM0CAPTCOMPX)	16	R/W	0000h	37.4.30/ 801
4003_3040	Capture Value 0 Register (PWMA_SM0CVAL0)	16	R	0000h	37.4.31/ 801
4003_3042	Capture Value 0 Cycle Register (PWMA_SM0CVAL0CYC)	16	R	0000h	37.4.32/ 801
4003_3044	Capture Value 1 Register (PWMA_SM0CVAL1)	16	R	0000h	37.4.33/ 802
4003_3046	Capture Value 1 Cycle Register (PWMA_SM0CVAL1CYC)	16	R	0000h	37.4.34/ 802
4003_3048	Capture Value 2 Register (PWMA_SM0CVAL2)	16	R	0000h	37.4.35/ 803
4003_304A	Capture Value 2 Cycle Register (PWMA_SM0CVAL2CYC)	16	R	0000h	37.4.36/ 803
4003_304C	Capture Value 3 Register (PWMA_SM0CVAL3)	16	R	0000h	37.4.37/ 803
4003_304E	Capture Value 3 Cycle Register (PWMA_SM0CVAL3CYC)	16	R	0000h	37.4.38/ 804
4003_3050	Capture Value 4 Register (PWMA_SM0CVAL4)	16	R	0000h	37.4.39/ 804
4003_3052	Capture Value 4 Cycle Register (PWMA_SM0CVAL4CYC)	16	R	0000h	37.4.40/ 805
4003_3054	Capture Value 5 Register (PWMA_SM0CVAL5)	16	R	0000h	37.4.41/ 805
4003_3056	Capture Value 5 Cycle Register (PWMA_SM0CVAL5CYC)	16	R	0000h	37.4.42/ 805
4003_3060	Counter Register (PWMA_SM1CNT)	16	R	0000h	37.4.1/772
4003_3062	Initial Count Register (PWMA_SM1INIT)	16	R/W	0000h	37.4.2/772
4003_3064	Control 2 Register (PWMA_SM1CTRL2)	16	R/W	0000h	37.4.3/773
4003_3066	Control Register (PWMA_SM1CTRL)	16	R/W	0400h	37.4.4/775
4003_306A	Value Register 0 (PWMA_SM1VAL0)	16	R/W	0000h	37.4.5/778
4003_306C	Fractional Value Register 1 (PWMA_SM1FRACVAL1)	16	R/W	0000h	37.4.6/778
4003_306E	Value Register 1 (PWMA_SM1VAL1)	16	R/W	0000h	37.4.7/779
4003_3070	Fractional Value Register 2 (PWMA_SM1FRACVAL2)	16	R/W	0000h	37.4.8/779
4003_3072	Value Register 2 (PWMA_SM1VAL2)	16	R/W	0000h	37.4.9/780
4003_3074	Fractional Value Register 3 (PWMA_SM1FRACVAL3)	16	R/W	0000h	37.4.10/ 780
4003_3076	Value Register 3 (PWMA_SM1VAL3)	16	R/W	0000h	37.4.11/ 781
4003_3078	Fractional Value Register 4 (PWMA_SM1FRACVAL4)	16	R/W	0000h	37.4.12/ 781
4003_307A	Value Register 4 (PWMA_SM1VAL4)	16	R/W	0000h	37.4.13/ 782

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_307C	Fractional Value Register 5 (PWMA_SM1FRACVAL5)	16	R/W	0000h	37.4.14/ 782
4003_307E	Value Register 5 (PWMA_SM1VAL5)	16	R/W	0000h	37.4.15/ 783
4003_3080	Fractional Control Register (PWMA_SM1FRCTRL)	16	R/W	0000h	37.4.16/ 783
4003_3082	Output Control Register (PWMA_SM1OCTRL)	16	R/W	0000h	37.4.17/ 785
4003_3084	Status Register (PWMA_SM1STS)	16	w1c	0000h	37.4.18/ 787
4003_3086	Interrupt Enable Register (PWMA_SM1INTEN)	16	R/W	0000h	37.4.19/ 788
4003_3088	DMA Enable Register (PWMA_SM1DMAEN)	16	R/W	0000h	37.4.20/ 790
4003_308A	Output Trigger Control Register (PWMA_SM1TCTRL)	16	R/W	0000h	37.4.21/ 791
4003_308C	Fault Disable Mapping Register 0 (PWMA_SM1DISMAP0)	16	R/W	FFFFh	37.4.22/ 792
4003_3090	Deadtime Count Register 0 (PWMA_SM1DTCNT0)	16	R/W	07FFh	37.4.23/ 793
4003_3092	Deadtime Count Register 1 (PWMA_SM1DTCNT1)	16	R/W	07FFh	37.4.24/ 794
4003_3094	Capture Control A Register (PWMA_SM1CAPTCTRLA)	16	R/W	0000h	37.4.25/ 794
4003_3096	Capture Compare A Register (PWMA_SM1CAPTCOMPA)	16	R/W	0000h	37.4.26/ 796
4003_3098	Capture Control B Register (PWMA_SM1CAPTCTRLB)	16	R/W	0000h	37.4.27/ 797
4003_309A	Capture Compare B Register (PWMA_SM1CAPTCOMPB)	16	R/W	0000h	37.4.28/ 798
4003_309C	Capture Control X Register (PWMA_SM1CAPTCTRLX)	16	R/W	0000h	37.4.29/ 799
4003_309E	Capture Compare X Register (PWMA_SM1CAPTCOMPX)	16	R/W	0000h	37.4.30/ 801
4003_30A0	Capture Value 0 Register (PWMA_SM1CVAL0)	16	R	0000h	37.4.31/ 801
4003_30A2	Capture Value 0 Cycle Register (PWMA_SM1CVAL0CYC)	16	R	0000h	37.4.32/ 801
4003_30A4	Capture Value 1 Register (PWMA_SM1CVAL1)	16	R	0000h	37.4.33/ 802
4003_30A6	Capture Value 1 Cycle Register (PWMA_SM1CVAL1CYC)	16	R	0000h	37.4.34/ 802
4003_30A8	Capture Value 2 Register (PWMA_SM1CVAL2)	16	R	0000h	37.4.35/ 803

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_30AA	Capture Value 2 Cycle Register (PWMA_SM1CVAL2CYC)	16	R	0000h	37.4.36/ 803
4003_30AC	Capture Value 3 Register (PWMA_SM1CVAL3)	16	R	0000h	37.4.37/ 803
4003_30AE	Capture Value 3 Cycle Register (PWMA_SM1CVAL3CYC)	16	R	0000h	37.4.38/ 804
4003_30B0	Capture Value 4 Register (PWMA_SM1CVAL4)	16	R	0000h	37.4.39/ 804
4003_30B2	Capture Value 4 Cycle Register (PWMA_SM1CVAL4CYC)	16	R	0000h	37.4.40/ 805
4003_30B4	Capture Value 5 Register (PWMA_SM1CVAL5)	16	R	0000h	37.4.41/ 805
4003_30B6	Capture Value 5 Cycle Register (PWMA_SM1CVAL5CYC)	16	R	0000h	37.4.42/ 805
4003_30C0	Counter Register (PWMA_SM2CNT)	16	R	0000h	37.4.1/772
4003_30C2	Initial Count Register (PWMA_SM2INIT)	16	R/W	0000h	37.4.2/772
4003_30C4	Control 2 Register (PWMA_SM2CTRL2)	16	R/W	0000h	37.4.3/773
4003_30C6	Control Register (PWMA_SM2CTRL)	16	R/W	0400h	37.4.4/775
4003_30CA	Value Register 0 (PWMA_SM2VAL0)	16	R/W	0000h	37.4.5/778
4003_30CC	Fractional Value Register 1 (PWMA_SM2FRACVAL1)	16	R/W	0000h	37.4.6/778
4003_30CE	Value Register 1 (PWMA_SM2VAL1)	16	R/W	0000h	37.4.7/779
4003_30D0	Fractional Value Register 2 (PWMA_SM2FRACVAL2)	16	R/W	0000h	37.4.8/779
4003_30D2	Value Register 2 (PWMA_SM2VAL2)	16	R/W	0000h	37.4.9/780
4003_30D4	Fractional Value Register 3 (PWMA_SM2FRACVAL3)	16	R/W	0000h	37.4.10/ 780
4003_30D6	Value Register 3 (PWMA_SM2VAL3)	16	R/W	0000h	37.4.11/ 781
4003_30D8	Fractional Value Register 4 (PWMA_SM2FRACVAL4)	16	R/W	0000h	37.4.12/ 781
4003_30DA	Value Register 4 (PWMA_SM2VAL4)	16	R/W	0000h	37.4.13/ 782
4003_30DC	Fractional Value Register 5 (PWMA_SM2FRACVAL5)	16	R/W	0000h	37.4.14/ 782
4003_30DE	Value Register 5 (PWMA_SM2VAL5)	16	R/W	0000h	37.4.15/ 783
4003_30E0	Fractional Control Register (PWMA_SM2FRCTRL)	16	R/W	0000h	37.4.16/ 783
4003_30E2	Output Control Register (PWMA_SM2OCTRL)	16	R/W	0000h	37.4.17/ 785
4003_30E4	Status Register (PWMA_SM2STS)	16	w1c	0000h	37.4.18/ 787
4003_30E6	Interrupt Enable Register (PWMA_SM2INTEN)	16	R/W	0000h	37.4.19/ 788

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_30E8	DMA Enable Register (PWMA_SM2DMAEN)	16	R/W	0000h	37.4.20/ 790
4003_30EA	Output Trigger Control Register (PWMA_SM2TCTRL)	16	R/W	0000h	37.4.21/ 791
4003_30EC	Fault Disable Mapping Register 0 (PWMA_SM2DISMAP0)	16	R/W	FFFFh	37.4.22/ 792
4003_30F0	Deadtime Count Register 0 (PWMA_SM2DTCNT0)	16	R/W	07FFh	37.4.23/ 793
4003_30F2	Deadtime Count Register 1 (PWMA_SM2DTCNT1)	16	R/W	07FFh	37.4.24/ 794
4003_30F4	Capture Control A Register (PWMA_SM2CAPTCTRLA)	16	R/W	0000h	37.4.25/ 794
4003_30F6	Capture Compare A Register (PWMA_SM2CAPTCOMPA)	16	R/W	0000h	37.4.26/ 796
4003_30F8	Capture Control B Register (PWMA_SM2CAPTCTRLB)	16	R/W	0000h	37.4.27/ 797
4003_30FA	Capture Compare B Register (PWMA_SM2CAPTCOMPB)	16	R/W	0000h	37.4.28/ 798
4003_30FC	Capture Control X Register (PWMA_SM2CAPTCTRLX)	16	R/W	0000h	37.4.29/ 799
4003_30FE	Capture Compare X Register (PWMA_SM2CAPTCOMPX)	16	R/W	0000h	37.4.30/ 801
4003_3100	Capture Value 0 Register (PWMA_SM2CVAL0)	16	R	0000h	37.4.31/ 801
4003_3102	Capture Value 0 Cycle Register (PWMA_SM2CVAL0CYC)	16	R	0000h	37.4.32/ 801
4003_3104	Capture Value 1 Register (PWMA_SM2CVAL1)	16	R	0000h	37.4.33/ 802
4003_3106	Capture Value 1 Cycle Register (PWMA_SM2CVAL1CYC)	16	R	0000h	37.4.34/ 802
4003_3108	Capture Value 2 Register (PWMA_SM2CVAL2)	16	R	0000h	37.4.35/ 803
4003_310A	Capture Value 2 Cycle Register (PWMA_SM2CVAL2CYC)	16	R	0000h	37.4.36/ 803
4003_310C	Capture Value 3 Register (PWMA_SM2CVAL3)	16	R	0000h	37.4.37/ 803
4003_310E	Capture Value 3 Cycle Register (PWMA_SM2CVAL3CYC)	16	R	0000h	37.4.38/ 804
4003_3110	Capture Value 4 Register (PWMA_SM2CVAL4)	16	R	0000h	37.4.39/ 804
4003_3112	Capture Value 4 Cycle Register (PWMA_SM2CVAL4CYC)	16	R	0000h	37.4.40/ 805
4003_3114	Capture Value 5 Register (PWMA_SM2CVAL5)	16	R	0000h	37.4.41/ 805

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_3116	Capture Value 5 Cycle Register (PWMA_SM2CVAL5CYC)	16	R	0000h	37.4.42/ 805
4003_3120	Counter Register (PWMA_SM3CNT)	16	R	0000h	37.4.1/772
4003_3122	Initial Count Register (PWMA_SM3INIT)	16	R/W	0000h	37.4.2/772
4003_3124	Control 2 Register (PWMA_SM3CTRL2)	16	R/W	0000h	37.4.3/773
4003_3126	Control Register (PWMA_SM3CTRL)	16	R/W	0400h	37.4.4/775
4003_312A	Value Register 0 (PWMA_SM3VAL0)	16	R/W	0000h	37.4.5/778
4003_312C	Fractional Value Register 1 (PWMA_SM3FRACVAL1)	16	R/W	0000h	37.4.6/778
4003_312E	Value Register 1 (PWMA_SM3VAL1)	16	R/W	0000h	37.4.7/779
4003_3130	Fractional Value Register 2 (PWMA_SM3FRACVAL2)	16	R/W	0000h	37.4.8/779
4003_3132	Value Register 2 (PWMA_SM3VAL2)	16	R/W	0000h	37.4.9/780
4003_3134	Fractional Value Register 3 (PWMA_SM3FRACVAL3)	16	R/W	0000h	37.4.10/ 780
4003_3136	Value Register 3 (PWMA_SM3VAL3)	16	R/W	0000h	37.4.11/ 781
4003_3138	Fractional Value Register 4 (PWMA_SM3FRACVAL4)	16	R/W	0000h	37.4.12/ 781
4003_313A	Value Register 4 (PWMA_SM3VAL4)	16	R/W	0000h	37.4.13/ 782
4003_313C	Fractional Value Register 5 (PWMA_SM3FRACVAL5)	16	R/W	0000h	37.4.14/ 782
4003_313E	Value Register 5 (PWMA_SM3VAL5)	16	R/W	0000h	37.4.15/ 783
4003_3140	Fractional Control Register (PWMA_SM3FRCTRL)	16	R/W	0000h	37.4.16/ 783
4003_3142	Output Control Register (PWMA_SM3OCTRL)	16	R/W	0000h	37.4.17/ 785
4003_3144	Status Register (PWMA_SM3STS)	16	w1c	0000h	37.4.18/ 787
4003_3146	Interrupt Enable Register (PWMA_SM3INTEN)	16	R/W	0000h	37.4.19/ 788
4003_3148	DMA Enable Register (PWMA_SM3DMAEN)	16	R/W	0000h	37.4.20/ 790
4003_314A	Output Trigger Control Register (PWMA_SM3TCTRL)	16	R/W	0000h	37.4.21/ 791
4003_314C	Fault Disable Mapping Register 0 (PWMA_SM3DISMAP0)	16	R/W	FFFFh	37.4.22/ 792
4003_3150	Deadtime Count Register 0 (PWMA_SM3DTCNT0)	16	R/W	07FFh	37.4.23/ 793
4003_3152	Deadtime Count Register 1 (PWMA_SM3DTCNT1)	16	R/W	07FFh	37.4.24/ 794
4003_3154	Capture Control A Register (PWMA_SM3CAPTCTRLA)	16	R/W	0000h	37.4.25/ 794

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_3156	Capture Compare A Register (PWMA_SM3CAPTCOMPA)	16	R/W	0000h	37.4.26/ 796
4003_3158	Capture Control B Register (PWMA_SM3CAPTCTRLB)	16	R/W	0000h	37.4.27/ 797
4003_315A	Capture Compare B Register (PWMA_SM3CAPTCOMPB)	16	R/W	0000h	37.4.28/ 798
4003_315C	Capture Control X Register (PWMA_SM3CAPTCTRLX)	16	R/W	0000h	37.4.29/ 799
4003_315E	Capture Compare X Register (PWMA_SM3CAPTCOMPX)	16	R/W	0000h	37.4.30/ 801
4003_3160	Capture Value 0 Register (PWMA_SM3CVAL0)	16	R	0000h	37.4.31/ 801
4003_3162	Capture Value 0 Cycle Register (PWMA_SM3CVAL0CYC)	16	R	0000h	37.4.32/ 801
4003_3164	Capture Value 1 Register (PWMA_SM3CVAL1)	16	R	0000h	37.4.33/ 802
4003_3166	Capture Value 1 Cycle Register (PWMA_SM3CVAL1CYC)	16	R	0000h	37.4.34/ 802
4003_3168	Capture Value 2 Register (PWMA_SM3CVAL2)	16	R	0000h	37.4.35/ 803
4003_316A	Capture Value 2 Cycle Register (PWMA_SM3CVAL2CYC)	16	R	0000h	37.4.36/ 803
4003_316C	Capture Value 3 Register (PWMA_SM3CVAL3)	16	R	0000h	37.4.37/ 803
4003_316E	Capture Value 3 Cycle Register (PWMA_SM3CVAL3CYC)	16	R	0000h	37.4.38/ 804
4003_3170	Capture Value 4 Register (PWMA_SM3CVAL4)	16	R	0000h	37.4.39/ 804
4003_3172	Capture Value 4 Cycle Register (PWMA_SM3CVAL4CYC)	16	R	0000h	37.4.40/ 805
4003_3174	Capture Value 5 Register (PWMA_SM3CVAL5)	16	R	0000h	37.4.41/ 805
4003_3176	Capture Value 5 Cycle Register (PWMA_SM3CVAL5CYC)	16	R	0000h	37.4.42/ 805
4003_3180	Output Enable Register (PWMA_OUTEN)	16	R/W	0000h	37.4.44/ 806
4003_3182	Mask Register (PWMA_MASK)	16	R/W	0000h	37.4.45/ 807
4003_3184	Software Controlled Output Register (PWMA_SWCOUT)	16	R/W	0000h	37.4.46/ 808
4003_3186	PWM Source Select Register (PWMA_DTSRCSEL)	16	R/W	0000h	37.4.47/ 809
4003_3188	Master Control Register (PWMA_MCTRL)	16	R/W	0000h	37.4.48/ 811

Table continues on the next page...

PWMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_318A	Master Control 2 Register (PWMA_MCTRL2)	16	R/W	0000h	37.4.49/ 812
4003_318C	Fault Control Register (PWMA_FCTRL)	16	R/W	0000h	37.4.50/ 813
4003_318E	Fault Status Register (PWMA_FSTS)	16	R/W	0000h	37.4.51/ 814
4003_3190	Fault Filter Register (PWMA_FFILT)	16	R/W	0000h	37.4.52/ 815
4003_3192	Fault Test Register (PWMA_FTST)	16	R/W	0000h	37.4.53/ 817
4003_3194	Fault Control 2 Register (PWMA_FCTRL2)	16	R/W	0000h	37.4.43/ 817

37.4.1 Counter Register (PWMA_SM*n*CNT)

This read-only register displays the state of the signed 16-bit submodule counter. This register is not byte accessible.

Address: 4003_3000h base + 0h offset + (96d × i), where i=0d to 3d

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

PWMA_SM*n*CNT field descriptions

Field	Description
CNT	Counter Register Bits

37.4.2 Initial Count Register (PWMA_SM*n*INIT)

The 16-bit signed value in this buffered, read/write register defines the initial count value for the PWM in PWM clock periods. This is the value loaded into the submodule counter when local sync, master sync, or master reload is asserted (based on the value of CTRL2[INIT_SEL]) or when CTRL2[FORCE] is asserted and force init is enabled. For PWM operation, the buffered contents of this register are loaded into the counter at the start of every PWM cycle. This register is not byte accessible.

NOTE

The INIT register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. This register cannot be written when MCTRL[LDOK] is set. Reading INIT reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Address: 4003_3000h base + 2h offset + (96d × i), where i=0d to 3d

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

PWMA_SM*n*INIT field descriptions

Field	Description
INIT	Initial Count Register Bits

37.4.3 Control 2 Register (PWMA_SM*n*CTRL2)

Address: 4003_3000h base + 4h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read Write	DBGEN	WAITEN	INDEP	PWM23_INIT	PWM45_INIT	PWMX_INIT	INIT_SEL	
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	FRCEN	0	FORCE_SEL			RELOAD_SEL	CLK_SEL	
Write		FORCE						
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*CTRL2 field descriptions

Field	Description
15 DBGEN	Debug Enable When set to one, the PWM will continue to run while the chip is in debug mode. If the device enters debug mode and this bit is zero, then the PWM outputs will be disabled until debug mode is exited. At that point the PWM pins will resume operation as programmed in the PWM registers. For certain types of motors (such as 3-phase AC), it is imperative that this bit be left in its default state (in which the PWM is disabled in debug mode). Failure to do so could result in damage to the motor or inverter. For other types of motors (example: DC motors), this bit might safely be set to one, enabling the PWM in debug mode. The key point is PWM parameter updates will not occur in debug mode. Any motors requiring such updates should be disabled during debug mode. If in doubt, leave this bit set to zero.

Table continues on the next page...

PWMA_SM*n*CTRL2 field descriptions (continued)

Field	Description
14 WAITEN	WAIT Enable When set to one, the PWM will continue to run while the chip is in WAIT mode. In this mode, the peripheral clock continues to run but the CPU clock does not. If the device enters WAIT mode and this bit is zero, then the PWM outputs will be disabled until WAIT mode is exited. At that point the PWM pins will resume operation as programmed in the PWM registers. For certain types of motors (such as 3-phase AC), it is imperative that this bit be left in its default state (in which the PWM is disabled in WAIT mode). Failure to do so could result in damage to the motor or inverter. For other types of motors (example: DC motors), this bit might safely be set to one, enabling the PWM in WAIT mode. The key point is PWM parameter updates will not occur in this mode. Any motors requiring such updates should be disabled during WAIT mode. If in doubt, leave this bit set to zero.
13 INDEP	Independent or Complementary Pair Operation This bit determines if the PWM_A and PWM_B channels will be independent PWMs or a complementary PWM pair. 0 PWM_A and PWM_B form a complementary PWM pair. 1 PWM_A and PWM_B outputs are independent PWMs.
12 PWM23_INIT	PWM23 Initial Value This read/write bit determines the initial value for PWM23 and the value to which it is forced when FORCE_INIT is asserted.
11 PWM45_INIT	PWM45 Initial Value This read/write bit determines the initial value for PWM45 and the value to which it is forced when FORCE_INIT is asserted.
10 PWMX_INIT	PWM_X Initial Value This read/write bit determines the initial value for PWM_X and the value to which it is forced when FORCE_INIT is asserted.
9–8 INIT_SEL	Initialization Control Select These read/write bits control the source of the INIT signal which goes to the counter. 00 Local sync (PWM_X) causes initialization. 01 Master reload from submodule 0 causes initialization. This setting should not be used in submodule 0 as it will force the INIT signal to logic 0. The submodule counter will only reinitialize when a master reload occurs. 10 Master sync from submodule 0 causes initialization. This setting should not be used in submodule 0 as it will force the INIT signal to logic 0. 11 EXT_SYNC causes initialization.
7 FRCEN	Force Initialization Enable This bit allows the CTRL2[FORCE] signal to initialize the counter without regard to the signal selected by CTRL2[INIT_SEL]. This is a software controlled initialization. 0 Initialization from a FORCE_OUT event is disabled. 1 Initialization from a FORCE_OUT event is enabled.
6 FORCE	Force Initialization If CTRL2[FORCE_SEL] is set to 000, writing a 1 to this bit results in a FORCE_OUT event. This causes the following actions to be taken:

Table continues on the next page...

PWMA_SM*n*CTRL2 field descriptions (continued)

Field	Description
	• The PWM_A and PWM_B output pins will assume values based on DTSRCSEL[SMxSEL23] and DTSRCSEL[SMxSEL45]. • If CTRL2[FRCEN] is set, the counter value will be initialized with the INIT register value.
5–3 FORCE_SEL	This read/write bit determines the source of the FORCE OUTPUT signal for this submodule. 000 The local force signal, CTRL2[FORCE], from this submodule is used to force updates. 001 The master force signal from submodule 0 is used to force updates. This setting should not be used in submodule 0 as it will hold the FORCE OUTPUT signal to logic 0. 010 The local reload signal from this submodule is used to force updates without regard to the state of LDOK. 011 The master reload signal from submodule0 is used to force updates if LDOK is set. This setting should not be used in submodule0 as it will hold the FORCE OUTPUT signal to logic 0. 100 The local sync signal from this submodule is used to force updates. 101 The master sync signal from submodule0 is used to force updates. This setting should not be used in submodule0 as it will hold the FORCE OUTPUT signal to logic 0. 110 The external force signal, EXT_FORCE, from outside the PWM module causes updates. 111 The external sync signal, EXT_SYNC, from outside the PWM module causes updates.
2 RELOAD_SEL	Reload Source Select This read/write bit determines the source of the RELOAD signal for this submodule. When this bit is set, MCTRL[LDOK[0]] for submodule 0 should be used since the local MCTRL[LDOK] will be ignored. 0 The local RELOAD signal is used to reload registers. 1 The master RELOAD signal (from submodule 0) is used to reload registers. This setting should not be used in submodule 0 as it will force the RELOAD signal to logic 0.
CLK_SEL	Clock Source Select These read/write bits determine the source of the clock signal for this submodule. 00 The IPBus clock is used as the clock for the local prescaler and counter. 01 EXT_CLK is used as the clock for the local prescaler and counter. 10 Submodule 0's clock (AUX_CLK) is used as the source clock for the local prescaler and counter. This setting should not be used in submodule 0 as it will force the clock to logic 0. 11 reserved

37.4.4 Control Register (PWMA_SM*n*CTRL)

Address: 4003_3000h base + 6h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	LDFQ				HALF	FULL	DT	
Write								
Reset	0	0	0	0	0	1	0	0

Bit	7	6	5	4	3	2	1	0
Read	COMPMODE	PRSC			0	LDMOD	DBLX	DBLEN
Write								
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*CTRL field descriptions

Field	Description
15–12 LDFQ	These buffered read/write bits select the PWM load frequency. Reset clears LDFQ, selecting loading every PWM opportunity. A PWM opportunity is determined by HALF and FULL. **NOTE:** LDFQ takes effect when the current load cycle is complete, regardless of the state of MCTRL[LDOK]. Reading LDFQ reads the buffered values and not necessarily the values currently in effect. 0000 Every PWM opportunity 0001 Every 2 PWM opportunities 0010 Every 3 PWM opportunities 0011 Every 4 PWM opportunities 0100 Every 5 PWM opportunities 0101 Every 6 PWM opportunities 0110 Every 7 PWM opportunities 0111 Every 8 PWM opportunities 1000 Every 9 PWM opportunities 1001 Every 10 PWM opportunities 1010 Every 11 PWM opportunities 1011 Every 12 PWM opportunities 1100 Every 13 PWM opportunities 1101 Every 14 PWM opportunities 1110 Every 15 PWM opportunities 1111 Every 16 PWM opportunities
11 HALF	Half Cycle Reload This read/write bit enables half-cycle reloads. A half cycle is defined by when the submodule counter matches the VAL0 register and does not have to be half way through the PWM cycle. 0 Half-cycle reloads disabled. 1 Half-cycle reloads enabled.
10 FULL	Full Cycle Reload This read/write bit enables full-cycle reloads. A full cycle is defined by when the submodule counter matches the VAL1 register. Either CTRL[HALF] or CTRL[FULL] must be set in order to move the buffered data into the registers used by the PWM generators or CTRL[LDMOD] must be set. If both CTRL[HALF] and CTRL[FULL] are set, then reloads can occur twice per cycle. 0 Full-cycle reloads disabled. 1 Full-cycle reloads enabled.
9–8 DT	Deadtime These read only bits reflect the sampled values of the PWM_X input at the end of each deadtime. Sampling occurs at the end of deadtime 0 for DT[0] and the end of deadtime 1 for DT[1]. Reset clears these bits.
7 COMPMODE	Compare Mode This bit controls how comparisons are made between the VAL* registers and the PWM submodule counter. This bit can only be written one time after which it requires a reset to release the bit for writing again. 0 The VAL* registers and the PWM counter are compared using an "equal to" method. This means that PWM edges are only produced when the counter is equal to one of the VAL* register values. This

Table continues on the next page...

PWMA_SM*n*CTRL field descriptions (continued)

Field	Description
	implies that a PWMA output that is high at the end of a period will maintain this state until a match with VAL3 clears the output in the following period. 1 The VAL* registers and the PWM counter are compared using an "equal to or greater than" method. This means that PWM edges are produced when the counter is equal to or greater than one of the VAL* register values. This implies that a PWMA output that is high at the end of a period could go low at the start of the next period if the starting counter value is greater than (but not necessarily equal to) the new VAL3 value.
6–4 PRSC	Prescaler These buffered read/write bits select the divide ratio of the PWM clock frequency selected by CTRL2[CLK_SEL]. **NOTE:** Reading CTRL[PRSC] reads the buffered values and not necessarily the values currently in effect. CTRL[PRSC] takes effect at the beginning of the next PWM cycle and only when the load okay bit, MCTRL[LDOK], is set or CTRL[LDMOD] is set. This field cannot be written when MCTRL[LDOK] is set. 000 PWM clock frequency = f_{clk} 001 PWM clock frequency = $f_{clk}/2$ 010 PWM clock frequency = $f_{clk}/4$ 011 PWM clock frequency = $f_{clk}/8$ 100 PWM clock frequency = $f_{clk}/16$ 101 PWM clock frequency = $f_{clk}/32$ 110 PWM clock frequency = $f_{clk}/64$ 111 PWM clock frequency = $f_{clk}/128$
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 LDMOD	Load Mode Select This read/write bit selects the timing of loading the buffered registers for this submodule. 0 Buffered registers of this submodule are loaded and take effect at the next PWM reload if MCTRL[LDOK] is set. 1 Buffered registers of this submodule are loaded and take effect immediately upon MCTRL[LDOK] being set. In this case it is not necessary to set CTRL[FULL] or CTRL[HALF].
1 DBLX	PWMX Double Switching Enable This read/write bit enables the double switching behavior on PWMX. When this bit is set, the PWMX output shall be the exclusive OR combination of PWMA and PWMB prior to polarity and masking considerations. 0 PWMX double pulse disabled. 1 PWMX double pulse enabled.
0 DBLEN	Double Switching Enable This read/write bit enables the double switching PWM behavior(See Double Switching PWMs). Double switching is not compatible with fractional PWM clock generation. Make sure this bit is clear when setting FRCTRL[FRAC23_EN], FRCTRL[FRAC45_EN], or FRCTRL[FRAC1_EN]. 0 Double switching disabled. 1 Double switching enabled.

37.4.5 Value Register 0 (PWMA_SM*n*VAL0)

Address: 4003_3000h base + Ah offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	VAL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*VAL0 field descriptions

Field	Description
VAL0	Value Register 0 The 16-bit signed value in this buffered, read/write register defines the mid-cycle reload point for the PWM in PWM clock periods. This value also defines when the PWM_X signal is set and the local sync signal is reset. This register is not byte accessible. **NOTE:** The VAL0 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL0 cannot be written when MCTRL[LDOK] is set. Reading VAL0 reads the value in a buffer. It is not necessarily the value the PWM generator is currently using.

37.4.6 Fractional Value Register 1 (PWMA_SM*n*FRACVAL1)

Address: 4003_3000h base + Ch offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	FRACVAL1					0										
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*FRACVAL1 field descriptions

Field	Description
15–11 FRACVAL1	Fractional Value 1 Register These bits act as a fractional addition to the value in the VAL1 register which controls the PWM period width. The PWM period is computed in terms of IPBus clock cycles. This fractional portion is accumulated at the end of every cycle until an additional whole IPBus cycle is reached. At this time the value being used for VAL1 is temporarily incremented and the PWM cycle is extended by one clock period to compensate for the accumulated fractional values. **NOTE:** The FRACVAL1 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL1 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL1 reads the value in a buffer and not necessarily the value the PWM generator is currently using.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

37.4.7 Value Register 1 (PWMA_SM*n*VAL1)

Address: 4003_3000h base + Eh offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	VAL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*VAL1 field descriptions

Field	Description
VAL1	Value Register 1 The 16-bit signed value written to this buffered, read/write register defines the modulo count value (maximum count) for the submodule counter. Upon reaching this count value, the counter reloads itself with the contents of the INIT register and asserts the local sync signal while resetting PWM_X. This register is not byte accessible. **NOTE:** The VAL1 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL1 cannot be written when MCTRL[LDOK] is set. Reading VAL1 reads the value in a buffer. It is not necessarily the value the PWM generator is currently using. **NOTE:** When using FRACVAL1, limit the maximum value of VAL1 to 0xFFFE for unsigned applications or to 0x7FFE for signed applications, to avoid counter rollovers caused by accumulating the fractional period defined by FRACVAL1. **NOTE:** If the VAL1 register defines the timer period (Local Sync is selected as the counter initialization signal), a 100% duty cycle cannot be achieved on the PWMX output. After the count reaches VAL1, the PWMX output is low for a minimum of one count every cycle. When the Master Sync signal (only originated by the Local Sync from sub-module 0) is used to control the timer period, the VAL1 register can be free for other functions such as PWM generation without the duty cycle limitation.

37.4.8 Fractional Value Register 2 (PWMA_SM*n*FRACVAL2)

Address: 4003_3000h base + 10h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	FRACVAL2					0										
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*FRACVAL2 field descriptions

Field	Description
15–11 FRACVAL2	Fractional Value 2 These bits act as a fractional addition to the value in the VAL2 register which controls the PWM_A turn on timing. It is also used to control the fractional addition to the turn off delay of PWM_B when MCTRL[IPOLx]=0 in complementary mode, CTRL2[INDEP]=0.

Table continues on the next page...

PWMA_SM*n*FRACVAL2 field descriptions (continued)

Field	Description
	NOTE: The FRACVAL2 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL2 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL2 reads the value in a buffer and not necessarily the value the PWM generator is currently using. **NOTE:** FRCTRL[FRAC23_EN] should be set to 0 when the values of VAL2 and VAL3 cause the high or low time of the PWM output to be 3 cycles or less.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

37.4.9 Value Register 2 (PWMA_SM*n*VAL2)

Address: 4003_3000h base + 12h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	VAL2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*VAL2 field descriptions

Field	Description
VAL2	Value Register 2 The 16-bit signed value in this buffered, read/write register defines the count value to set PWM23 high. This register is not byte accessible. **NOTE:** The VAL2 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL2 cannot be written when MCTRL[LDOK] is set. Reading VAL2 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

37.4.10 Fractional Value Register 3 (PWMA_SM*n*FRACVAL3)

Address: 4003_3000h base + 14h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	FRACVAL3					0										
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*FRACVAL3 field descriptions

Field	Description
15–11 FRACVAL3	Fractional Value 3 These bits act as a fractional addition to the value in the VAL3 register which controls the PWM_A turn off timing. It is also used to control the fractional addition to the turn on delay of PWM_B when MCTRL[IPOLx]=0 in complementary mode, CTRL2[INDEP]=0.

Table continues on the next page...

PWMA_SM*n*FRACVAL3 field descriptions (continued)

Field	Description
	NOTE: The FRACVAL3 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL3 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL3 reads the value in a buffer and not necessarily the value the PWM generator is currently using. **NOTE:** FRCTRL[FRAC23_EN] should be set to 0 when the values of VAL2 and VAL3 cause the high or low time of the PWM output to be 3 cycles or less.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

37.4.11 Value Register 3 (PWMA_SM*n*VAL3)

Address: 4003_3000h base + 16h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	VAL3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*VAL3 field descriptions

Field	Description
VAL3	Value Register 3 The 16-bit signed value in this buffered, read/write register defines the count value to set PWM23 low. This register is not byte accessible. **NOTE:** The VAL3 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL3 cannot be written when MCTRL[LDOK] is set. Reading VAL3 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

37.4.12 Fractional Value Register 4 (PWMA_SM*n*FRACVAL4)

Address: 4003_3000h base + 18h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	FRACVAL4					0										
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*FRACVAL4 field descriptions

Field	Description
15–11 FRACVAL4	Fractional Value 4 These bits act as a fractional addition to the value in the VAL4 register which controls the PWM_B turn on timing. It is also used to control the fractional addition to the turn off delay of PWM_A when MCTRL[IPOLx]=1 in complementary mode, CTRL2[INDEP]=0.

Table continues on the next page...

PWMA_SM*n*FRACVAL4 field descriptions (continued)

Field	Description
	NOTE: The FRACVAL4 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL4 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL4 reads the value in a buffer and not necessarily the value the PWM generator is currently using. **NOTE:** FRCTRL[FRAC45_EN] should be set to 0 when the values of VAL4 and VAL5 cause the high or low time of the PWM output to be 3 cycles or less.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

37.4.13 Value Register 4 (PWMA_SM*n*VAL4)

Address: 4003_3000h base + 1Ah offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	VAL4															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*VAL4 field descriptions

Field	Description
VAL4	Value Register 4 The 16-bit signed value in this buffered, read/write register defines the count value to set PWM45 high. This register is not byte accessible. **NOTE:** The VAL4 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL4 cannot be written when MCTRL[LDOK] is set. Reading VAL4 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

37.4.14 Fractional Value Register 5 (PWMA_SM*n*FRACVAL5)

Address: 4003_3000h base + 1Ch offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	FRACVAL5					0										
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*FRACVAL5 field descriptions

Field	Description
15–11 FRACVAL5	Fractional Value 5 These bits act as a fractional addition to the value in the VAL5 register which controls the PWM_B turn off timing. It is also used to control the fractional addition to the turn on delay of PWM_A when MCTRL[IPOLx]=1 in complementary mode, CTRL2[INDEP]=0.

Table continues on the next page...

PWMA_SM*n*FRACVAL5 field descriptions (continued)

Field	Description
	NOTE: The FRACVAL5 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRACVAL5 cannot be written when MCTRL[LDOK] is set. Reading FRACVAL5 reads the value in a buffer and not necessarily the value the PWM generator is currently using. **NOTE:** FRCTRL[FRAC45_EN] should be set to 0 when the values of VAL4 and VAL5 cause the high or low time of the PWM output to be 3 cycles or less.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

37.4.15 Value Register 5 (PWMA_SM*n*VAL5)

Address: 4003_3000h base + 1Eh offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	VAL5															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*VAL5 field descriptions

Field	Description
VAL5	Value Register 5 The 16-bit signed value in this buffered, read/write register defines the count value to set PWM45 low. This register is not byte accessible. **NOTE:** The VAL5 register is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. VAL5 cannot be written when MCTRL[LDOK] is set. Reading VAL5 reads the value in a buffer and not necessarily the value the PWM generator is currently using.

37.4.16 Fractional Control Register (PWMA_SM*n*FRCTRL)

Address: 4003_3000h base + 20h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	TEST	0						FRAC_PU
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0			FRAC45_EN	0	FRAC23_EN	FRAC1_EN	0
Write								
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*FRCTRL field descriptions

Field	Description
15 TEST	Test Status Bit This is a read only test bit for factory use. This bit will reset to 0 but may be either 0 or 1 during PWM operation.
14–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 FRAC_PU	Fractional Delay Circuit Power Up This bit is used to power up the fractional delay analog block. The fractional delay block takes 25 us to power up after the first FRAC_PU bit in any submodule is set. The fractional delay block only powers down when the FRAC_PU bits in all submodules are 0. The fractional delay logic can only be used when the IPBus clock is running at 100 MHz. When turned off, fractional placement is disabled. 0 Turn off fractional delay logic. 1 Power up fractional delay logic.
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 FRAC45_EN	Fractional Cycle Placement Enable for PWM_B This bit is used to enable the fractional cycle edge placement of PWM_B using the FRACVAL4 and FRACVAL5 registers. When disabled, the fractional cycle edge placement of PWM_B is bypassed. **NOTE:** The FRAC45_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC45_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC45_EN reads the value in a buffer and not necessarily the value the PWM generator is currently using. 0 Disable fractional cycle placement for PWM_B. 1 Enable fractional cycle placement for PWM_B.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 FRAC23_EN	Fractional Cycle Placement Enable for PWM_A This bit is used to enable the fractional cycle edge placement of PWM_A using the FRACVAL2 and FRACVAL3 registers. When disabled, the fractional cycle edge placement of PWM_A is bypassed. **NOTE:** The FRAC23_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC23_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC23_EN reads the value in a buffer and not necessarily the value the PWM generator is currently using. 0 Disable fractional cycle placement for PWM_A. 1 Enable fractional cycle placement for PWM_A.
1 FRAC1_EN	Fractional Cycle PWM Period Enable This bit is used to enable the fractional cycle length of the PWM period using the FRACVAL1 register. When disabled, the fractional cycle length of the PWM period is bypassed. **NOTE:** The FRAC1_EN bit is buffered. The value written does not take effect until MCTRL[LDOK] is set and the next PWM load cycle begins or CTRL[LDMOD] is set. FRAC1_EN cannot be written when MCTRL[LDOK] is set. Reading FRAC1_EN reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Table continues on the next page...

PWMA_SM*n*FRCTRL field descriptions (continued)

Field	Description
	0 Disable fractional cycle length for the PWM period. 1 Enable fractional cycle length for the PWM period.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

37.4.17 Output Control Register (PWMA_SM*n*OCTRL)

Address: 4003_3000h base + 22h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	PWMA_IN	PWMB_IN	PWMX_IN	0		POLA	POLB	POLX
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	0		PWMAFS		PWMBFS		PWMXFS	
Write								
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*OCTRL field descriptions

Field	Description
15 PWMA_IN	PWM_A Input This read only bit shows the logic value currently being driven into the PWM_A input. The bit's reset state is undefined.
14 PWMB_IN	PWM_B Input This read only bit shows the logic value currently being driven into the PWM_B input. The bit's reset state is undefined.
13 PWMX_IN	PWM_X Input This read only bit shows the logic value currently being driven into the PWM_X input. The bit's reset state is undefined.
12–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 POLA	PWM_A Output Polarity This bit inverts the PWM_A output polarity. 0 PWM_A output not inverted. A high level on the PWM_A pin represents the "on" or "active" state. 1 PWM_A output inverted. A low level on the PWM_A pin represents the "on" or "active" state.
9 POLB	PWM_B Output Polarity This bit inverts the PWM_B output polarity.

Table continues on the next page...

PWMA_SM*n*OCTRL field descriptions (continued)

Field	Description
	0 PWM_B output not inverted. A high level on the PWM_B pin represents the "on" or "active" state. 1 PWM_B output inverted. A low level on the PWM_B pin represents the "on" or "active" state.
8 POLX	PWM_X Output Polarity This bit inverts the PWM_X output polarity. 0 PWM_X output not inverted. A high level on the PWM_X pin represents the "on" or "active" state. 1 PWM_X output inverted. A low level on the PWM_X pin represents the "on" or "active" state.
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 PWMAFS	PWM_A Fault State These bits determine the fault state for the PWM_A output during fault conditions and STOP mode. It may also define the output state during WAIT and DEBUG modes depending on the settings of CTRL2[WAITEN] and CTRL2[DBGEN]. 00 Output is forced to logic 0 state prior to consideration of output polarity control. 01 Output is forced to logic 1 state prior to consideration of output polarity control. 10 Output is tristated. 11 Output is tristated.
3–2 PWMBFS	PWM_B Fault State These bits determine the fault state for the PWM_B output during fault conditions and STOP mode. It may also define the output state during WAIT and DEBUG modes depending on the settings of CTRL2[WAITEN] and CTRL2[DBGEN]. 00 Output is forced to logic 0 state prior to consideration of output polarity control. 01 Output is forced to logic 1 state prior to consideration of output polarity control. 10 Output is tristated. 11 Output is tristated.
PWMXFS	PWM_X Fault State These bits determine the fault state for the PWM_X output during fault conditions and STOP mode. It may also define the output state during WAIT and DEBUG modes depending on the settings of CTRL2[WAITEN] and CTRL2[DBGEN]. 00 Output is forced to logic 0 state prior to consideration of output polarity control. 01 Output is forced to logic 1 state prior to consideration of output polarity control. 10 Output is tristated. 11 Output is tristated.

37.4.18 Status Register (PWMA_SM*n*STS)

Address: 4003_3000h base + 24h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	0	RUF	REF	RF	CFA1	CFA0	CFB1	CFB0
Write			w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	CFX1	CFX0	CMPF					
Write	w1c	w1c	w1c					
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*STS field descriptions

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 RUF	Registers Updated Flag This read-only flag is set when one of the INIT, VALx,FRACVALx, or CTRL[PRSC] registers has been written, which indicates potentially non-coherent data in the set of double buffered registers. Clear this bit by a proper reload sequence consisting of a reload signal while MCTRL[LDOK] = 1. Reset clears this bit. 0 No register update has occurred since last reload. 1 At least one of the double buffered registers has been updated since the last reload.
13 REF	Reload Error Flag This read/write flag is set when a reload cycle occurs while MCTRL[LDOK] is 0 and the double buffered registers are in a non-coherent state (STS[RUF] = 1). Clear this bit by writing a logic one to this location. Reset clears this bit. 0 No reload error occurred. 1 Reload signal occurred with non-coherent data and MCTRL[LDOK] = 0.
12 RF	Reload Flag This read/write flag is set at the beginning of every reload cycle regardless of the state of MCTRL[LDOK]. Clear this bit by writing a logic one to this location when DMAEN[VALDE] is clear (non-DMA mode). This flag can also be cleared by the DMA done signal when DMAEN[VALDE] is set (DMA mode). Reset clears this bit. 0 No new reload cycle since last STS[RF] clearing 1 New reload cycle since last STS[RF] clearing
11 CFA1	Capture Flag A1 This bit is set when a capture event occurs on the Capture A1 circuit. This bit is cleared by writing a one to this bit position if DMAEN[CA1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CA1DE] is set (DMA mode). Reset clears this bit.
10 CFA0	Capture Flag A0

Table continues on the next page...

PWMA_SM*n*STS field descriptions (continued)

Field	Description
	This bit is set when a capture event occurs on the Capture A0 circuit. This bit is cleared by writing a one to this bit position if DMAEN[CA0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CA0DE] is set (DMA mode). Reset clears this bit.
9 CFB1	Capture Flag B1 This bit is set when a capture event occurs on the Capture B1 circuit. This bit is cleared by writing a one to this bit position if DMAEN[CB1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CB1DE] is set (DMA mode). Reset clears this bit.
8 CFB0	Capture Flag B0 This bit is set when a capture event occurs on the Capture B0 circuit. This bit is cleared by writing a one to this bit position if DMAEN[CB0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CB0DE] is set (DMA mode). Reset clears this bit.
7 CFX1	Capture Flag X1 This bit is set when a capture event occurs on the Capture X1 circuit. This bit is cleared by writing a one to this bit position if DMAEN[CX1DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CX1DE] is set (DMA mode). Reset clears this bit.
6 CFX0	Capture Flag X0 This bit is set when a capture event occurs on the Capture X0 circuit. This bit is cleared by writing a one to this bit position if DMAEN[CX0DE] is clear (non-DMA mode) or by the DMA done signal if DMAEN[CX0DE] is set (DMA mode). Reset clears this bit.
CMPF	Compare Flags These bits are set when the submodule counter value matches the value of one of the VALx registers. Clear these bits by writing a 1 to a bit position. 0 No compare event has occurred for a particular VALx value. 1 A compare event has occurred for a particular VALx value.

37.4.19 Interrupt Enable Register (PWMA_SM*n*INTEN)

Address: 4003_3000h base + 26h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	0		REIE	RIE	CA1IE	CA0IE	CB1IE	CB0IE
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	CX1IE	CX0IE	CMPIE					
Write								
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*INTEN field descriptions

Field	Description
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

PWMA_SM*n*INTEN field descriptions (continued)

Field	Description
13 REIE	Reload Error Interrupt Enable This read/write bit enables the reload error flag, STS[REF], to generate CPU interrupt requests. Reset clears this bit. 0 STS[REF] CPU interrupt requests disabled 1 STS[REF] CPU interrupt requests enabled
12 RIE	Reload Interrupt Enable This read/write bit enables the reload flag, STS[RF], to generate CPU interrupt requests. Reset clears this bit. 0 STS[RF] CPU interrupt requests disabled 1 STS[RF] CPU interrupt requests enabled
11 CA1IE	Capture A 1 Interrupt Enable This bit allows the STS[CFA1] flag to create an interrupt request to the CPU. Do not set both this bit and DMAEN[CA1DE]. 0 Interrupt request disabled for STS[CFA1]. 1 Interrupt request enabled for STS[CFA1].
10 CA0IE	Capture A 0 Interrupt Enable This bit allows the STS[CFA0] flag to create an interrupt request to the CPU. Do not set both this bit and DMAEN[CA0DE]. 0 Interrupt request disabled for STS[CFA0]. 1 Interrupt request enabled for STS[CFA0].
9 CB1IE	Capture B 1 Interrupt Enable This bit allows the STS[CFB1] flag to create an interrupt request to the CPU. Do not set both this bit and DMAEN[CB1DE]. 0 Interrupt request disabled for STS[CFB1]. 1 Interrupt request enabled for STS[CFB1].
8 CB0IE	Capture B 0 Interrupt Enable This bit allows the STS[CFB0] flag to create an interrupt request to the CPU. Do not set both this bit and DMAEN[CB0DE]. 0 Interrupt request disabled for STS[CFB0]. 1 Interrupt request enabled for STS[CFB0].
7 CX1IE	Capture X 1 Interrupt Enable This bit allows the STS[CFX1] flag to create an interrupt request to the CPU. Do not set both this bit and DMAEN[CX1DE]. 0 Interrupt request disabled for STS[CFX1]. 1 Interrupt request enabled for STS[CFX1].
6 CX0IE	Capture X 0 Interrupt Enable This bit allows the STS[CFX0] flag to create an interrupt request to the CPU. Do not set both this bit and DMAEN[CX0DE].

Table continues on the next page...

PWMA_SM*n*INTEN field descriptions (continued)

Field	Description
	0 Interrupt request disabled for STS[CFX0]. 1 Interrupt request enabled for STS[CFX0].
CMPIE	Compare Interrupt Enables These bits enable the STS[CMPF] flags to cause a compare interrupt request to the CPU. 0 The corresponding STS[CMPF] bit will not cause an interrupt request. 1 The corresponding STS[CMPF] bit will cause an interrupt request.

37.4.20 DMA Enable Register (PWMA_SM*n*DMAEN)

Address: 4003_3000h base + 28h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	0						VALDE	FAND
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	CAPTDE		CA1DE	CA0DE	CB1DE	CB0DE	CX1DE	CX0DE
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*DMAEN field descriptions

Field	Description
15–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 VALDE	Value Registers DMA Enable This read/write bit enables DMA write requests for the VALx and FRACVALx registers when STS[RF] is set. Reset clears this bit. 0 DMA write requests disabled 1 DMA write requests for the VALx and FRACVALx registers enabled
8 FAND	FIFO Watermark AND Control This read/write bit works in conjunction with the DMAEN[CAPTDE] field when it is set to watermark mode (DMAEN[CAPTDE] = 01). While DMAEN[CAxDE], DMAEN[CBxDE], and DMAEN[CXxDE] determine which FIFO watermarks the DMA read request is sensitive to, this bit determines if the selected watermarks are AND'ed together or OR'ed together in order to create the request. 0 Selected FIFO watermarks are OR'ed together. 1 Selected FIFO watermarks are AND'ed together.
7–6 CAPTDE	Capture DMA Enable Source Select These read/write bits select the source of enabling the DMA read requests for the capture FIFOs. Reset clears these bits. 00 Read DMA requests disabled.

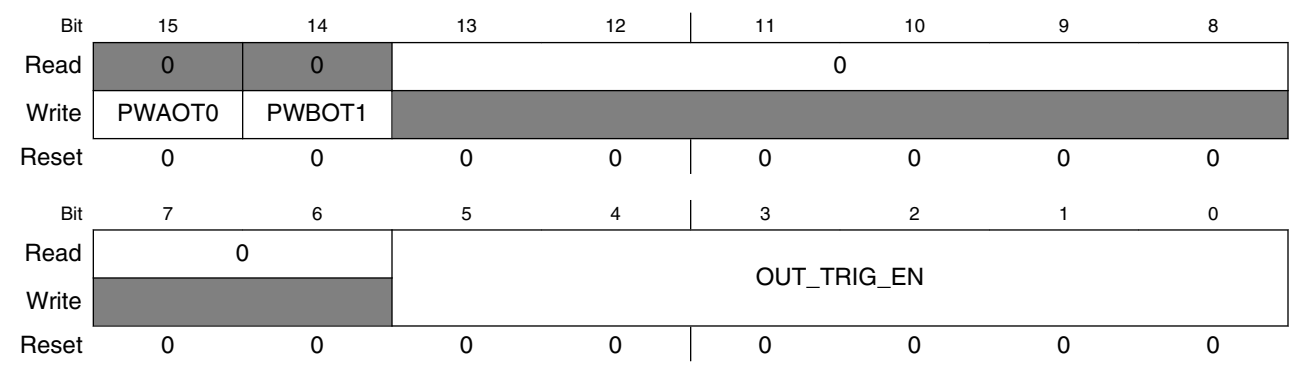
Table continues on the next page...

PWMA_SM*n*DMAEN field descriptions (continued)

Field	Description
	01 Exceeding a FIFO watermark sets the DMA read request. This requires at least one of DMAEN[CA1DE], DMAEN[CA0DE], DMAEN[CB1DE], DMAEN[CB0DE], DMAEN[CX1DE], or DMAEN[CX0DE] to also be set in order to determine to which watermark(s) the DMA request is sensitive. 10 A local sync (VAL1 matches counter) sets the read DMA request. 11 A local reload (STS[RF] being set) sets the read DMA request.
5 CA1DE	Capture A1 FIFO DMA Enable This read/write bit enables DMA read requests for the Capture A1 FIFO data when STS[CFA1] is set. Reset clears this bit. Do not set both this bit and INTEN[CA1IE].
4 CA0DE	Capture A0 FIFO DMA Enable This read/write bit enables DMA read requests for the Capture A0 FIFO data when STS[CFA0] is set. Reset clears this bit. Do not set both this bit and INTEN[CA0IE].
3 CB1DE	Capture B1 FIFO DMA Enable This read/write bit enables DMA read requests for the Capture B1 FIFO data when STS[CFB1] is set. Reset clears this bit. Do not set both this bit and INTEN[CB1IE].
2 CB0DE	Capture B0 FIFO DMA Enable This read/write bit enables DMA read requests for the Capture B0 FIFO data when STS[CFB0] is set. Reset clears this bit. Do not set both this bit and INTEN[CB0IE].
1 CX1DE	Capture X1 FIFO DMA Enable This read/write bit enables DMA read requests for the Capture X1 FIFO data when STS[CFX1] is set. Reset clears this bit. Do not set both this bit and INTEN[CX1IE].
0 CX0DE	Capture X0 FIFO DMA Enable This read/write bit enables DMA read requests for the Capture X0 FIFO data when STS[CFX0] is set. Reset clears this bit. Do not set both this bit and INTEN[CX0IE].

37.4.21 Output Trigger Control Register (PWMA_SM*n*TCTRL)

Address: 4003_3000h base + 2Ah offset + (96d × i), where i=0d to 3d



PWMA_SM*n*TCTRL field descriptions

Field	Description
15 PWAOT0	Output Trigger 0 Source Select This bit selects which signal to bring out on the PWM's PWM_OUT_TRIG0 port. The output trigger port is often connected to routing logic on the chip. This control bit allows the PWMA output signal to be driven onto the output trigger port so it can be sent to the chip routing logic. 0 Route the PWM_OUT_TRIG0 signal to PWM_OUT_TRIG0 port. 1 Route the PWMA output to the PWM_OUT_TRIG0 port.
14 PWBOT1	Output Trigger 1 Source Select This bit selects which signal to bring out on the PWM's PWM_OUT_TRIG1 port. The output trigger port is often connected to routing logic on the chip. This control bit allows the PWMB output signal to be driven onto the output trigger port so it can be sent to the chip routing logic. 0 Route the PWM_OUT_TRIG1 signal to PWM_OUT_TRIG1 port. 1 Route the PWMB output to the PWM_OUT_TRIG1 port.
13–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
OUT_TRIG_EN	Output Trigger Enables These bits enable the generation of PWM_OUT_TRIG0 and PWM_OUT_TRIG1 outputs based on the counter value matching the value in one or more of the VAL0-5 registers. VAL0, VAL2, and VAL4 are used to generate PWM_OUT_TRIG0, and VAL1, VAL3, and VAL5 are used to generate PWM_OUT_TRIG1. The PWM_OUT_TRIGx signals are only asserted as long as the counter value matches the VALx value; therefore, up to six triggers can be generated (three each on PWM_OUT_TRIG0 and PWM_OUT_TRIG1) per PWM cycle per submodule. **NOTE:** Due to delays in creating the PWM outputs, the output trigger signals will lead the PWM output edges by 2-3 clock cycles depending on the fractional cycle value being used. 0 PWM_OUT_TRIGx will not set when the counter value matches the VALx value. 1 PWM_OUT_TRIGx will set when the counter value matches the VALx value.

37.4.22 Fault Disable Mapping Register 0 (PWMA_SM*n*DISMAP0)

This register determines which PWM pins are disabled by the fault protection inputs. Reset sets all of the bits in the fault disable mapping register.

Address: 4003_3000h base + 2Ch offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	1				DIS0X				DIS0B				DIS0A			
Write																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

PWMA_SM*n*DISMAP0 field descriptions

Field	Description
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
11–8 DIS0X	PWM_X Fault Disable Mask 0 Each of the four bits of this read/write field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_X output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. A reset sets all bits in this field.
7–4 DIS0B	PWM_B Fault Disable Mask 0 Each of the four bits of this read/write field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_B output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. A reset sets all bits in this field.
DIS0A	PWM_A Fault Disable Mask 0 Each of the four bits of this read/write field is one-to-one associated with the four FAULTx inputs of fault channel 0. The PWM_A output is turned off if there is a logic 1 on a FAULTx input and a 1 in the corresponding bit of this field. A reset sets all bits in this field.

37.4.23 Deadtime Count Register 0 (PWMA_SM*n*DTCNT0)

Deadtime operation applies only to complementary channel operation. The values written to the DTCNTx registers are in terms of IPBus clock cycles regardless of the setting of CTRL[PRSC] and/or CTRL2[CLK_SEL]. Reset sets the deadtime count registers to a default value of 0x07FF, selecting a deadtime of 2047 IPBus clock cycles. The DTCNTx registers are not byte accessible.

Address: 4003_3000h base + 30h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	DTCNT0															
Reset	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

PWMA_SM*n*DTCNT0 field descriptions

Field	Description
DTCNT0	Deadtime Count Register 0 The DTCNT0 field is interpreted differently depending on whether or not the fractional delays are being used (FRCNTRL[FRAC23_EN] is set). If the fractional delays are off, then the upper 5 bits of DTCNT0 are ignored and the remaining 11 bits are used to specify the number of cycles of deadtime. In this case the maximum value is 0x07FF which indicates 2047 cycles of deadtime. If the fractional delays are being used, then the upper 11 bits of DTCNT0 represent the number of whole cycles of deadtime and the lower 5 bits of each register represent the fractional cycle added to the whole number. In this case the maximum value is 0xFFFF which represents 2047 31/32 cycles of deadtime. The DTCNT0 field is used to control the deadtime during 0 to 1 transitions of the PWM_A output (assuming normal polarity).

37.4.24 Deadtime Count Register 1 (PWMA_SM*n*DTCNT1)

Deadtime operation applies only to complementary channel operation. The values written to the DTCNTx registers are in terms of IPBus clock cycles regardless of the setting of CTRL[PRSC] and/or CTRL2[CLK_SEL]. Reset sets the deadtime count registers to a default value of 0x07FF, selecting a deadtime of 2047 IPBus clock cycles. The DTCNTx registers are not byte accessible.

Address: 4003_3000h base + 32h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	DTCNT1															
Reset	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

PWMA_SM*n*DTCNT1 field descriptions

Field	Description
DTCNT1	Deadtime Count Register 1 The DTCNT1 field is interpreted differently depending on whether or not the fractional delays are being used (FRCNTRL[FRAC45_EN] is set). If the fractional delays are off, then the upper 5 bits of DTCNT1 are ignored and the remaining 11 bits are used to specify the number of cycles of deadtime. In this case the maximum value is 0x07FF which indicates 2047 cycles of deadtime. If the fractional delays are being used, then the upper 11 bits of DTCNT1 represent the number of whole cycles of deadtime and the lower 5 bits of each register represent the fractional cycle added to the whole number. In this case the maximum value is 0xFFFF which represents 2047 31/32 cycles of deadtime. The DTCNT1 field is used to control the deadtime during 0 to 1 transitions of the complementary PWM_B output.

37.4.25 Capture Control A Register (PWMA_SM*n*CAPTCTRLA)

Address: 4003_3000h base + 34h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CA1CNT			CA0CNT			CFAWM		EDGCNTA_EN	INP_SELA	EDGA1		EDGA0		ONESHOTA	ARMA
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CAPTCTRLA field descriptions

Field	Description
15–13 CA1CNT	Capture A1 FIFO Word Count This field reflects the number of words in the Capture A1 FIFO. (FIFO depth is 1)
12–10 CA0CNT	Capture A0 FIFO Word Count This field reflects the number of words in the Capture A0 FIFO. (FIFO depth is 1)
9–8 CFAWM	Capture A FIFOs Water Mark This field represents the water mark level for capture A FIFOs. The capture flags, STS[CFA1] and STS[CFA0], are not set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1)
7 EDGCNTA_EN	Edge Counter A Enable This bit enables the edge counter which counts rising and falling edges on the PWM_A input signal. 0 Edge counter disabled and held in reset 1 Edge counter enabled
6 INP_SELA	Input Select A This bit selects between the raw PWM_A input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0 Raw PWM_A input signal selected as source. 1 Output of edge counter/compare selected as source. **NOTE:** When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLA[EDGA0] and CAPTCTRLA[EDGA1] fields are ignored. The software must still place a value other than 00 in either or both of the CAPTCTLRA[EDGA0] and/or CAPTCTRLA[EDGA1] fields in order to enable one or both of the capture registers.
5–4 EDGA1	Edge A 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00 Disabled 01 Capture falling edges 10 Capture rising edges 11 Capture any edge
3–2 EDGA0	Edge A 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00 Disabled 01 Capture falling edges 10 Capture rising edges 11 Capture any edge
1 ONESHOTA	One Shot Mode A This bit selects between free running and one shot mode for the input capture circuitry. 0 Free running mode is selected.

Table continues on the next page...

PWMA_SM*n*CAPTCTRLA field descriptions (continued)

Field	Description
	If both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1 One shot mode is selected. If both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLA[ARMA] is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed and CAPTCTRLA[ARMA] is cleared. No further captures will be performed until CAPTCTRLA[ARMA] is set again. If only one of the capture circuits is enabled, then a single capture will occur on the enabled capture circuit and CAPTCTRLA[ARMA] is then cleared.
0 ARMA	Arm A Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0 Input capture operation is disabled. 1 Input capture operation as specified by CAPTCTRLA[EDGAx] is enabled.

37.4.26 Capture Compare A Register (PWMA_SM*n*CAPTCOMPA)

Address: 4003_3000h base + 36h offset + (96d × i), where i=0d to 3d

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

PWMA_SM*n*CAPTCOMPA field descriptions

Field	Description
15–8 EDGCNTA	Edge Counter A This read-only field contains the edge counter value for the PWM_A input capture circuitry.
EDGCMPA	Edge Compare A This read/write field is the compare value associated with the edge counter for the PWM_A input capture circuitry.

37.4.27 Capture Control B Register (PWMA_SM*n*CAPTCTRLB)

Address: 4003_3000h base + 38h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	CB1CNT			CB0CNT			CFBWM	
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	EDGCNTB_EN	INP_SELB	EDGB1		EDGB0		ONESHOTB	ARMB
Write								
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*CAPTCTRLB field descriptions

Field	Description
15–13 CB1CNT	Capture B1 FIFO Word Count This field reflects the number of words in the Capture B1 FIFO. (FIFO depth is 1)
12–10 CB0CNT	Capture B0 FIFO Word Count This field reflects the number of words in the Capture B0 FIFO. (FIFO depth is 1)
9–8 CFBWM	Capture B FIFOs Water Mark This field represents the water mark level for capture B FIFOs. The capture flags, STS[CFB1] and STS[CFB0], won't be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1)
7 EDGCNTB_EN	Edge Counter B Enable This bit enables the edge counter which counts rising and falling edges on the PWM_B input signal. 0 Edge counter disabled and held in reset 1 Edge counter enabled
6 INP_SELB	Input Select B This bit selects between the raw PWM_B input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit. 0 Raw PWM_B input signal selected as source. 1 Output of edge counter/compare selected as source. **NOTE:** When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLB[EDGB0] and CAPTCTRLB[EDGB1] fields are ignored. The software must still place a value other than 00 in either or both of the CAPTCTLRB[EDGB0] and/or CAPTCTRLB[EDGB1] fields in order to enable one or both of the capture registers.
5–4 EDGB1	Edge B 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00 Disabled

Table continues on the next page...

PWMA_SM*n*CAPTCTRLB field descriptions (continued)

Field	Description
	01 Capture falling edges 10 Capture rising edges 11 Capture any edge
3–2 EDGB0	Edge B 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00 Disabled 01 Capture falling edges 10 Capture rising edges 11 Capture any edge
1 ONESHOTB	One Shot Mode B This bit selects between free running and one shot mode for the input capture circuitry. 0 Free running mode is selected. If both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1 One shot mode is selected. If both capture circuits are enabled, then capture circuit 0 is armed first after CAPTCTRLB[ARMB] is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed and CAPTCTRLB[ARMB] is cleared. No further captures will be performed until CAPTCTRLB[ARMB] is set again. If only one of the capture circuits is enabled, then a single capture will occur on the enabled capture circuit and CAPTCTRLB[ARMB] is then cleared.
0 ARMB	Arm B Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0 Input capture operation is disabled. 1 Input capture operation as specified by CAPTCTRLB[EDGBx] is enabled.

37.4.28 Capture Compare B Register (PWMA_SM*n*CAPTCOMPB)

Address: 4003_3000h base + 3Ah offset + (96d × i), where i=0d to 3d

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

PWMA_SM*n*CAPTCOMPB field descriptions

Field	Description
15–8 EDGCNTB	Edge Counter B This read-only field contains the edge counter value for the PWM_B input capture circuitry.
EDGCMPB	Edge Compare B This read/write field is the compare value associated with the edge counter for the PWM_B input capture circuitry.

37.4.29 Capture Control X Register (PWMA_SM*n*CAPTCTRLX)

Address: 4003_3000h base + 3Ch offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8
Read	CX1CNT			CX0CNT			CFXWM	
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	EDGCNTX_EN	INP_SELX	EDGX1		EDGX0		ONESHOTX	ARMX
Write								
Reset	0	0	0	0	0	0	0	0

PWMA_SM*n*CAPTCTRLX field descriptions

Field	Description
15–13 CX1CNT	Capture X1 FIFO Word Count This field reflects the number of words in the Capture X1 FIFO. (FIFO depth is 1)
12–10 CX0CNT	Capture X0 FIFO Word Count This field reflects the number of words in the Capture X0 FIFO. (FIFO depth is 1)
9–8 CFXWM	Capture X FIFOs Water Mark This field represents the water mark level for capture X FIFOs. The capture flags, STS[CFX1] and STS[CFX0], won't be set until the word count of the corresponding FIFO is greater than this water mark level. (FIFO depth is 1)
7 EDGCNTX_EN	Edge Counter X Enable This bit enables the edge counter which counts rising and falling edges on the PWM_X input signal. 0 Edge counter disabled and held in reset 1 Edge counter enabled
6 INP_SELX	Input Select X This bit selects between the raw PWM_X input signal and the output of the edge counter/compare circuitry as the source for the input capture circuit.

Table continues on the next page...

PWMA_SM*n*CAPTCTRLX field descriptions (continued)

Field	Description
	0 Raw PWM_X input signal selected as source. 1 Output of edge counter/compare selected as source. **NOTE:** When this bitfield is set to 1, the internal edge counter is enabled and the rising and/or falling edges specified by the CAPTCTRLX[EDGX0] and CAPTCTRLX[EDGX1] fields are ignored. The software must still place a value other than 00 in either or both of the CAPTCTLRX[EDGX0] and/or CAPTCTRLX[EDGX1] fields in order to enable one or both of the capture registers.
5–4 EDGX1	Edge X 1 These bits control the input capture 1 circuitry by determining which input edges cause a capture event. 00 Disabled 01 Capture falling edges 10 Capture rising edges 11 Capture any edge
3–2 EDGX0	Edge X 0 These bits control the input capture 0 circuitry by determining which input edges cause a capture event. 00 Disabled 01 Capture falling edges 10 Capture rising edges 11 Capture any edge
1 ONESHOTX	One Shot Mode Aux This bit selects between free running and one shot mode for the input capture circuitry. 0 Free running mode is selected. If both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed and capture circuit 0 is re-armed. The process continues indefinitely. If only one of the capture circuits is enabled, then captures continue indefinitely on the enabled capture circuit. 1 One shot mode is selected. If both capture circuits are enabled, then capture circuit 0 is armed first after the ARMX bit is set. Once a capture occurs, capture circuit 0 is disarmed and capture circuit 1 is armed. After capture circuit 1 performs a capture, it is disarmed and the ARMX bit is cleared. No further captures will be performed until the ARMX bit is set again. If only one of the capture circuits is enabled, then a single capture will occur on the enabled capture circuit and the ARMX bit is then cleared.
0 ARMX	Arm X Setting this bit high starts the input capture process. This bit can be cleared at any time to disable input capture operation. This bit is self-cleared when in one shot mode and one or more of the enabled capture circuits has had a capture event. 0 Input capture operation is disabled. 1 Input capture operation as specified by CAPTCTRLX[EDGXx] is enabled.

37.4.30 Capture Compare X Register (PWMA_SM*n*CAPTCOMPX)

Address: 4003_3000h base + 3Eh offset + (96d × i), where i=0d to 3d

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

PWMA_SM*n*CAPTCOMPX field descriptions

Field	Description
15–8 EDGCNTX	Edge Counter X This read-only field contains the edge counter value for the PWM_X input capture circuitry.
EDGCMPX	Edge Compare X This read/write field is the compare value associated with the edge counter for the PWM_X input capture circuitry.

37.4.31 Capture Value 0 Register (PWMA_SM*n*CVAL0)

Address: 4003_3000h base + 40h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTVAL0															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL0 field descriptions

Field	Description
CAPTVAL0	This read-only register stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX0]. Each capture will increase the value of CAPTCTRLX[CX0CNT] by 1 until the maximum value is reached. Each read of this register will decrease the value of CAPTCTRLX[CX0CNT] by 1 until 0 is reached. This register is not byte accessible.

37.4.32 Capture Value 0 Cycle Register (PWMA_SM*n*CVAL0CYC)

Address: 4003_3000h base + 42h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0												CVAL0CYC			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL0CYC field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CVAL0CYC	This read-only register stores the cycle number corresponding to the value captured in CVAL0. This register is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

37.4.33 Capture Value 1 Register (PWMA_SM*n*CVAL1)

Address: 4003_3000h base + 44h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTVAL1															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL1 field descriptions

Field	Description
CAPTVAL1	This read-only register stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLX[EDGX1]. Each capture increases the value of CAPTCTRLX[CX1CNT] by 1 until the maximum value is reached. Each read of this register decreases the value of CAPTCTRLX[CX1CNT] by 1 until 0 is reached. This register is not byte accessible.

37.4.34 Capture Value 1 Cycle Register (PWMA_SM*n*CVAL1CYC)

Address: 4003_3000h base + 46h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0												CVAL1CYC			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL1CYC field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CVAL1CYC	This read-only register stores the cycle number corresponding to the value captured in CVAL1. This register is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

37.4.35 Capture Value 2 Register (PWMA_SM*n*CVAL2)

Address: 4003_3000h base + 48h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTVAL2															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL2 field descriptions

Field	Description
CAPTVAL2	This read-only register stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLA[EDGA0]. Each capture increases the value of CAPTCTRLA[CA0CNT] by 1 until the maximum value is reached. Each read of this register decreases the value of CAPTCTRLA[CA0CNT] by 1 until 0 is reached. This register is not byte accessible.

37.4.36 Capture Value 2 Cycle Register (PWMA_SM*n*CVAL2CYC)

Address: 4003_3000h base + 4Ah offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0												CVAL2CYC			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL2CYC field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CVAL2CYC	This read-only register stores the cycle number corresponding to the value captured in CVAL2. This register is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

37.4.37 Capture Value 3 Register (PWMA_SM*n*CVAL3)

Address: 4003_3000h base + 4Ch offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTVAL3															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL3 field descriptions

Field	Description
CAPTVAL3	This read-only register stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLA[EDGA1]. Each capture increases the value of CAPTCTRLA[CA1CNT] by 1 until the maximum value is reached. Each read of this register decreases the value of CAPTCTRLA[CA1CNT] by 1 until 0 is reached. This register is not byte accessible.

37.4.38 Capture Value 3 Cycle Register (PWMA_SM*n*CVAL3CYC)

Address: 4003_3000h base + 4Eh offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0												CVAL3CYC			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL3CYC field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CVAL3CYC	This read-only register stores the cycle number corresponding to the value captured in CVAL3. This register is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

37.4.39 Capture Value 4 Register (PWMA_SM*n*CVAL4)

Address: 4003_3000h base + 50h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTVAL4															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL4 field descriptions

Field	Description
CAPTVAL4	This read-only register stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLB[EDGB0]. Each capture increases the value of CAPTCTRLB[CB0CNT] by 1 until the maximum value is reached. Each read of this register decreases the value of CAPTCTRLB[CB0CNT] by 1 until 0 is reached. This register is not byte accessible.

37.4.40 Capture Value 4 Cycle Register (PWMA_SM*n*CVAL4CYC)

Address: 4003_3000h base + 52h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0												CVAL4CYC			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL4CYC field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CVAL4CYC	This read-only register stores the cycle number corresponding to the value captured in CVAL4. This register is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

37.4.41 Capture Value 5 Register (PWMA_SM*n*CVAL5)

Address: 4003_3000h base + 54h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	CAPTVAL5															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL5 field descriptions

Field	Description
CAPTVAL5	This read-only register stores the value captured from the submodule counter. Exactly when this capture occurs is defined by CAPTCTRLB[EDGB1]. Each capture increases the value of CAPTCTRLB[CB1CNT] by 1 until the maximum value is reached. Each read of this register decreases the value of CAPTCTRLB[CB1CNT] by 1 until 0 is reached. This register is not byte accessible.

37.4.42 Capture Value 5 Cycle Register (PWMA_SM*n*CVAL5CYC)

Address: 4003_3000h base + 56h offset + (96d × i), where i=0d to 3d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0												CVAL5CYC			
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_SM*n*CVAL5CYC field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CVAL5CYC	This read-only register stores the cycle number corresponding to the value captured in CVAL5. This register is incremented each time the counter is loaded with the INIT value at the end of a PWM modulo cycle.

37.4.44 Output Enable Register (PWMA_OUTEN)

Address: 4003_3000h base + 180h offset = 4003_3180h

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

PWMA_OUTEN field descriptions

Field	Description
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 PWMA_EN	PWM_A Output Enables The four bits of this field enable the PWM_A outputs of submodules 3-0, respectively. These bits should be set to 0 (output disabled) when a PWM_A pin is being used for input capture. 0 PWM_A output disabled. 1 PWM_A output enabled.
7–4 PWMB_EN	PWM_B Output Enables The four bits of this field enable the PWM_B outputs of submodules 3-0, respectively. These bits should be set to 0 (output disabled) when a PWM_B pin is being used for input capture. 0 PWM_B output disabled. 1 PWM_B output enabled.
PWMX_EN	PWM_X Output Enables The four bits of this field enable the PWM_X outputs of submodules 3-0, respectively. These bits should be set to 0 (output disabled) when a PWM_X pin is being used for input capture or deadtime correction. 0 PWM_X output disabled. 1 PWM_X output enabled.

37.4.45 Mask Register (PWMA_MASK)

MASK is double buffered and does not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading MASK reads the buffered values and not necessarily the values currently in effect. This double buffering can be overridden by setting the UPDATE_MASK bits.

Address: 4003_3000h base + 182h offset = 4003_3182h

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

PWMA_MASK field descriptions

Field	Description
15–12 UPDATE_MASK	Update Mask Bits Immediately The four bits mask the PWM_X outputs of submodules 3-0, respectively, The four bits of this field force the MASK* bits to be immediately updated within submodules 3-0, respectively, without waiting for a FORCE_OUT event. These self-clearing bits always read as zero. Software may write to any or all of these bits and may set these bits in the same write operation that updates the MASKA, MASKB, and MASKX fields of this register. 0 Normal operation. MASK* bits within the corresponding submodule are not updated until a FORCE_OUT event occurs within the submodule. 1 Immediate operation. MASK* bits within the corresponding submodule are updated on the following clock edge after setting this bit.
11–8 MASKA	PWM_A Masks The four bits of this field mask the PWM_A outputs of submodules 3-0, respectively, forcing the output to logic 0 prior to consideration of the output polarity. 0 PWM_A output normal. 1 PWM_A output masked.
7–4 MASKB	PWM_B Masks The four bits of this field mask the PWM_B outputs of submodules 3-0, respectively, forcing the output to logic 0 prior to consideration of the output polarity. 0 PWM_B output normal. 1 PWM_B output masked.
MASKX	PWM_X Masks The four bits of this field mask the PWM_X outputs of submodules 3-0, respectively, forcing the output to logic 0 prior to consideration of the output polarity. 0 PWM_X output normal. 1 PWM_X output masked.

37.4.46 Software Controlled Output Register (PWMA_SWCOUT)

These bits are double buffered and do not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading these bits reads the buffered value and not necessarily the value currently in effect.

Address: 4003_3000h base + 184h offset = 4003_3184h

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read Write	SM3OUT23	SM3OUT45	SM2OUT23	SM2OUT45	SM1OUT23	SM1OUT45	SM0OUT23	SM0OUT45
Reset	0	0	0	0	0	0	0	0

PWMA_SWCOUT field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 SM3OUT23	Submodule 3 Software Controlled Output 23 This bit is only used when DTSRCSEL[SM3SEL23] is set to 0b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 3 instead of PWM23. 1 A logic 1 is supplied to the deadtime generator of submodule 3 instead of PWM23.
6 SM3OUT45	Submodule 3 Software Controlled Output 45 This bit is only used when DTSRCSEL[SM3SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 3 instead of PWM45. 1 A logic 1 is supplied to the deadtime generator of submodule 3 instead of PWM45.
5 SM2OUT23	Submodule 2 Software Controlled Output 23 This bit is only used when DTSRCSEL[SM2SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 2 instead of PWM23. 1 A logic 1 is supplied to the deadtime generator of submodule 2 instead of PWM23.
4 SM2OUT45	Submodule 2 Software Controlled Output 45 This bit is only used when DTSRCSEL[SM2SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 2 instead of PWM45. 1 A logic 1 is supplied to the deadtime generator of submodule 2 instead of PWM45.

Table continues on the next page...

PWMA_SWCOUT field descriptions (continued)

Field	Description
3 SM1OUT23	Submodule 1 Software Controlled Output 23 This bit is only used when DTSRCSEL[SM1SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 1 instead of PWM23. 1 A logic 1 is supplied to the deadtime generator of submodule 1 instead of PWM23.
2 SM1OUT45	Submodule 1 Software Controlled Output 45 This bit is only used when DTSRCSEL[SM1SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 1 instead of PWM45. 1 A logic 1 is supplied to the deadtime generator of submodule 1 instead of PWM45.
1 SM0OUT23	Submodule 0 Software Controlled Output 23 This bit is only used when DTSRCSEL[SM0SEL23] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 0 instead of PWM23. 1 A logic 1 is supplied to the deadtime generator of submodule 0 instead of PWM23.
0 SM0OUT45	Submodule 0 Software Controlled Output 45 This bit is only used when DTSRCSEL[SM0SEL45] is set to b10. It allows software control of which signal is supplied to the deadtime generator of that submodule. 0 A logic 0 is supplied to the deadtime generator of submodule 0 instead of PWM45. 1 A logic 1 is supplied to the deadtime generator of submodule 0 instead of PWM45.

37.4.47 PWM Source Select Register (PWMA_DTSRCSEL)

The PWM source select bits are double buffered and do not take effect until a FORCE_OUT event occurs within the appropriate submodule. Reading these bits reads the buffered value and not necessarily the value currently in effect.

Address: 4003_3000h base + 186h offset = 4003_3186h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	SM3SEL23		SM3SEL45		SM2SEL23		SM2SEL45		SM1SEL23		SM1SEL45		SM0SEL23		SM0SEL45	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_DTSRCSEL field descriptions

Field	Description
15–14 SM3SEL23	Submodule 3 PWM23 Control Select This field selects possible over-rides to the generated SM3PWM23 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule.

Table continues on the next page...

PWMA_DTSRCSEL field descriptions (continued)

Field	Description
	00 Generated SM3PWM23 signal is used by the deadtime logic. 01 Inverted generated SM3PWM23 signal is used by the deadtime logic. 10 SWCOUT[SM3OUT23] is used by the deadtime logic. 11 PWM3_EXTA signal is used by the deadtime logic.
13–12 SM3SEL45	Submodule 3 PWM45 Control Select This field selects possible over-rides to the generated SM3PWM45 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM3PWM45 signal is used by the deadtime logic. 01 Inverted generated SM3PWM45 signal is used by the deadtime logic. 10 SWCOUT[SM3OUT45] is used by the deadtime logic. 11 PWM3_EXTB signal is used by the deadtime logic.
11–10 SM2SEL23	Submodule 2 PWM23 Control Select This field selects possible over-rides to the generated SM2PWM23 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM2PWM23 signal is used by the deadtime logic. 01 Inverted generated SM2PWM23 signal is used by the deadtime logic. 10 SWCOUT[SM2OUT23] is used by the deadtime logic. 11 PWM2_EXTA signal is used by the deadtime logic.
9–8 SM2SEL45	Submodule 2 PWM45 Control Select This field selects possible over-rides to the generated SM2PWM45 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that sudeadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM2PWM45 signal is used by the deadtime logic. 01 Inverted generated SM2PWM45 signal is used by the deadtime logic. 10 SWCOUT[SM2OUT45] is used by the deadtime logic. 11 PWM2_EXTB signal is used by the deadtime logic.
7–6 SM1SEL23	Submodule 1 PWM23 Control Select This field selects possible over-rides to the generated SM1PWM23 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM1PWM23 signal is used by the deadtime logic. 01 Inverted generated SM1PWM23 signal is used by the deadtime logic. 10 SWCOUT[SM1OUT23] is used by the deadtime logic. 11 PWM1_EXTA signal is used by the deadtime logic.
5–4 SM1SEL45	Submodule 1 PWM45 Control Select This field selects possible over-rides to the generated SM1PWM45 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM1PWM45 signal is used by the deadtime logic. 01 Inverted generated SM1PWM45 signal is used by the deadtime logic. 10 SWCOUT[SM1OUT45] is used by the deadtime logic. 11 PWM1_EXTB signal is used by the deadtime logic.

Table continues on the next page...

PWMA_DTSRCSEL field descriptions (continued)

Field	Description
3–2 SM0SEL23	Submodule 0 PWM23 Control Select This field selects possible over-rides to the generated SM0PWM23 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM0PWM23 signal is used by the deadtime logic. 01 Inverted generated SM0PWM23 signal is used by the deadtime logic. 10 SWCOUT[SM0OUT23] is used by the deadtime logic. 11 PWM0_EXTA signal is used by the deadtime logic.
SM0SEL45	Submodule 0 PWM45 Control Select This field selects possible over-rides to the generated SM0PWM45 signal that will be passed to the deadtime logic upon the occurrence of a FORCE_OUT event in that submodule. 00 Generated SM0PWM45 signal is used by the deadtime logic. 01 Inverted generated SM0PWM45 signal is used by the deadtime logic. 10 SWCOUT[SM0OUT45] is used by the deadtime logic. 11 PWM0_EXTB signal is used by the deadtime logic.

37.4.48 Master Control Register (PWMA_MCTRL)

In every 4-bit field in this register, each bit acts on a separate submodule. Accordingly, the description of every bitfield refers to the effect of an individual bit.

Address: 4003_3000h base + 188h offset = 4003_3188h

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

PWMA_MCTRL field descriptions

Field	Description
15–12 IPOL	Current Polarity The four buffered read/write bits of this field correspond to submodules 3-0, respectively. Each bit selects between PWM23 and PWM45 as the source for the generation of the complementary PWM pair output for the corresponding submodule. MCTRL[IPOL] is ignored in independent mode. MCTRL[IPOL] does not take effect until a FORCE_OUT event takes place in the appropriate submodule. Reading MCTRL[IPOL] reads the buffered value and not necessarily the value currently in effect. 0 PWM23 is used to generate complementary PWM pair in the corresponding submodule. 1 PWM45 is used to generate complementary PWM pair in the corresponding submodule.

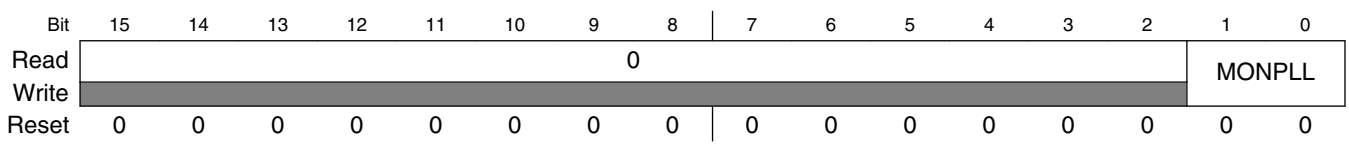
Table continues on the next page...

PWMA_MCTRL field descriptions (continued)

Field	Description
11–8 RUN	Run The four read/write bits of this field enable the clocks to the PWM generator of submodules 3-0, respectively. The corresponding MCTRL[RUN] bit must be set for each submodule that is using its input capture functions or is using the local reload as its reload source. When this bit equals zero, the submodule counter is reset. A reset clears this field. 0 PWM generator is disabled in the corresponding submodule. 1 PWM generator is enabled in the corresponding submodule.
7–4 CLDOK	Clear Load Okay The 4 bits of CLDOK field correspond to submodules 3-0, respectively. Each write-only bit is used to clear the corresponding bit of MCTRL[LDOK]. Write a 1 to CLDOK to clear the corresponding MCTRL[LDOK] bit. If a reload occurs within a submodule with the corresponding MCTRL[LDOK] bit set at the same time that MCTRL[CLDOK] is written, then the reload in that submodule will not be performed and MCTRL[LDOK] will be cleared. CLDOK bit is self-clearing and always reads as a 0.
LDOK	Load Okay The 4 bits of LDOK field correspond to submodules 3-0, respectively. Each read/set bit loads CTRL[PRSC] and the INIT, FRACVALx, and VALx registers of the corresponding submodule into a set of buffers. The buffered prescaler divisor, submodule counter modulus value, and PWM pulse width take effect at the next PWM reload if CTRL[LDMOD] is clear or immediately if CTRL[LDMOD] is set. Set the corresponding MCTRL[LDOK] bit by reading it when it is logic zero and then writing a logic one to it. The VALx, FRACVALx,INIT, and CTRL[PRSC] registers of the corresponding submodule cannot be written while the the corresponding MCTRL[LDOK] bit is set. In Master Reload Mode (CTRL2[RELOAD_SEL]=1), it is only necessary to set the LDOK bit corresponding to submodule0; however, it is recommended to also set the LDOK bit of the slave submodules, to prevent unwanted writes to the registers in the slave submodules. The MCTRL[LDOK] bit is automatically cleared after the new values are loaded, or it can be manually cleared before a reload by writing a logic 1 to the appropriate MCTRL[CLDOK] bit. LDOK bits cannot be written with a zero. MCTRL[LDOK] can be set in DMA mode when the DMA indicates that it has completed the update of all CTRL[PRSC], INIT,FRACVALx, and VALx registers in the corresponding submodule. Reset clears LDOK field. 0 Do not load new values. 1 Load prescaler, modulus, and PWM values of the corresponding submodule.

37.4.49 Master Control 2 Register (PWMA_MCTRL2)

Address: 4003_3000h base + 18Ah offset = 4003_318Ah



PWMA_MCTRL2 field descriptions

Field	Description
15–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MONPLL	Monitor PLL State These bits are used to control disabling of the fractional delay block when the chip PLL is unlocked and/or missing its input reference. The fractional delay block requires a continuous 200 MHz clock from the PLL. If this clock turns off when the fractional delay block is being used, then the output of the fractional delay block can be stuck high or low even if the PLL restarts. When this control bit is set, PLL problems cause the fractional delay block to be disabled until the PLL returns to a locked state. Once the PLL is receiving a proper reference and is locked, the fractional delay block requires a 25 µs startup time just as if the FRCTRL[FRAC*_EN] bits had been turned off and turned on again. If PLL monitoring is disabled, then software should manually clear and then set the FRCTRL[FRAC*_EN] bits when the PLL loses its reference or loses lock. This will cause the fractional delay block to be disabled and restarted. If the fractional delay block is not being used, then the value of these bits do not matter. 00 Not locked. Do not monitor PLL operation. Resetting of the fractional delay block in case of PLL losing lock will be controlled by software. 01 Not locked. Monitor PLL operation to automatically disable the fractional delay block when the PLL encounters problems. 10 Locked. Do not monitor PLL operation. Resetting of the fractional delay block in case of PLL losing lock will be controlled by software. These bits are write protected until the next reset. 11 Locked. Monitor PLL operation to automatically disable the fractional delay block when the PLL encounters problems. These bits are write protected until the next reset.

37.4.50 Fault Control Register (PWMA_FCTRL)

For every 4-bit field in this register, the bits act on the fault inputs in order. For example, FLVL bits 15-12 act on faults 3-0, respectively.

Address: 4003_3000h base + 18Ch offset = 4003_318Ch

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

PWMA_FCTRL field descriptions

Field	Description
15–12 FLVL	Fault Level The four read/write bits of this field select the active logic level of the individual fault inputs 3-0, respectively. A reset clears this field. 0 A logic 0 on the fault input indicates a fault condition. 1 A logic 1 on the fault input indicates a fault condition.

Table continues on the next page...

PWMA_FCTRL field descriptions (continued)

Field	Description
11–8 FAUTO	Automatic Fault Clearing The four read/write bits of this field select automatic or manual clearing of faults 3-0, respectively. A reset clears this field. 0 Manual fault clearing. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear at the start of a half cycle or full cycle depending the state of FSTS[FFULL]. This is further controlled by FCTRL[FSAFE]. 1 Automatic fault clearing. PWM outputs disabled by this fault are enabled when FSTS[FFPINx] is clear at the start of a half cycle or full cycle depending on the state of FSTS[FFULL] without regard to the state of FSTS[FFLAGx].
7–4 FSAFE	Fault Safety Mode These read/write bits select the safety mode during manual fault clearing. A reset clears this field. FSTS[FFPINx] may indicate a fault condition still exists even though the actual fault signal at the FAULTx pin is clear due to the fault filter latency. 0 Normal mode. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear at the start of a half cycle or full cycle depending on the state of FSTS[FFULL] without regard to the state of FSTS[FFPINx]. The PWM outputs disabled by this fault input will not be re-enabled until the actual FAULTx input signal de-asserts since the fault input will combinationally disable the PWM outputs (as programmed in DISMAPn). 1 Safe mode. PWM outputs disabled by this fault are not enabled until FSTS[FFLAGx] is clear and FSTS[FFPINx] is clear at the start of a half cycle or full cycle depending on the state of FSTS[FFULL].
FIE	Fault Interrupt Enables This read/write field enables CPU interrupt requests generated by the FAULTx pins. A reset clears this field. **NOTE:** The fault protection circuit is independent of the FIEx bit and is always active. If a fault is detected, the PWM outputs are disabled according to the disable mapping register. 0 FAULTx CPU interrupt requests disabled. 1 FAULTx CPU interrupt requests enabled.

37.4.51 Fault Status Register (PWMA_FSTS)

Address: 4003_3000h base + 18Eh offset = 4003_318Eh

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

PWMA_FSTS field descriptions

Field	Description
15–12 FHALF	Half Cycle Fault Recovery

Table continues on the next page...

PWMA_FSTS field descriptions (continued)

Field	Description
	These read/write bits are used to control the timing for re-enabling the PWM outputs after a fault condition. These bits apply to both automatic and manual clearing of a fault condition. **NOTE:** Both FHALF and FFULL can be set so that the fault recovery occurs at the start of a full cycle and at the start of a half cycle (as defined by VAL0). If neither FHALF nor FFULL is set, then no fault recovery is possible. 0 PWM outputs are not re-enabled at the start of a half cycle. 1 PWM outputs are re-enabled at the start of a half cycle (as defined by VAL0).
11–8 FFPIN	Filtered Fault Pins These read-only bits reflect the current state of the filtered FAULTx pins converted to high polarity. A logic 1 indicates a fault condition exists on the filtered FAULTx pin. A reset has no effect on this field.
7–4 FFULL	Full Cycle These read/write bits are used to control the timing for re-enabling the PWM outputs after a fault condition. These bits apply to both automatic and manual clearing of a fault condition. **NOTE:** Both FHALF and FFULL can be set so that the fault recovery occurs at the start of a full cycle and at the start of a half cycle (as defined by VAL0). If neither FHALF nor FFULL is set, then no fault recovery is possible. 0 PWM outputs are not re-enabled at the start of a full cycle 1 PWM outputs are re-enabled at the start of a full cycle
FFLAG	Fault Flags These read-only flag is set within two CPU cycles after a transition to active on the FAULTx pin. Clear this bit by writing a logic one to it. A reset clears this field. While the reset value is 0, these bits may be set to 1 by the time they can be read depending on the state of the fault input signals. 0 No fault on the FAULTx pin. 1 Fault on the FAULTx pin.

37.4.52 Fault Filter Register (PWMA_FFILT)

The settings in this register are shared among each of the fault input filters within the fault channel.

Input filter considerations include:

- The FILT_PER value should be set such that the sampling period is larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The FILT_CNT value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the FILT_CNT+3 power.
- The values of FILT_PER and FILT_CNT must also be traded off against the desire for minimal latency in recognizing input transitions. Turning on the input filter (setting FILT_PER to a non-zero value) introduces a latency of ((FILT_CNT+4) x

FILT_PER x IPBus clock period). Note that even when the filter is enabled, there is a combinational path to disable the PWM outputs. This is to ensure rapid response to fault conditions and also to ensure fault response if the PWM module loses its clock. The latency induced by the filter will be seen in the time to set FSTS[FFLAG] and FSTS[FFPIN].

Address: 4003_3000h base + 190h offset = 4003_3190h

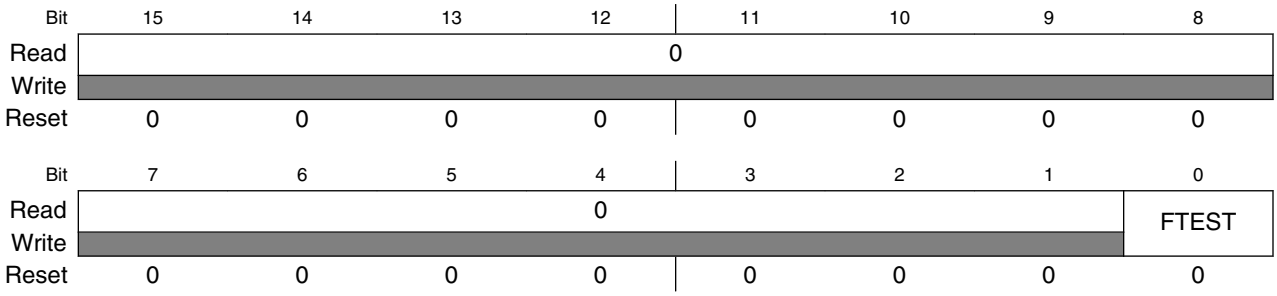
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	GSTR	0				FILT_CNT			FILT_PER							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMA_FFILT field descriptions

Field	Description
15 GSTR	Fault Glitch Stretch Enable This bit is used to enable the fault glitch stretching logic. This logic ensures that narrow fault glitches are stretched to be at least 2 IPBus clock cycles wide. In some cases a narrow fault input can cause problems due to the short PWM output shutdown/re-activation time. The stretching logic ensures that a glitch on the fault input, when the fault filter is disabled, will be registered in the fault flags. 0 Fault input glitch stretching is disabled. 1 Input fault signals will be stretched to at least 2 IPBus clock cycles.
14–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 FILT_CNT	Fault Filter Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. The number of samples is the decimal value of this field plus three: the bitfield value of 0-7 represents 3-10 samples, respectively. The value of FILT_CNT affects the input latency.
FILT_PER	Fault Filter Period This 8-bit field applies universally to all fault inputs. These bits represent the sampling period (in IPBus clock cycles) of the fault pin input filter. Each input is sampled multiple times at the rate specified by this field. If FILT_PER is 0x00 (default), then the input filter is bypassed. The value of FILT_PER affects the input latency. **NOTE:** When changing values for FILT_PER from one non-zero value to another non-zero value, first write a value of zero to clear the filter.

37.4.53 Fault Test Register (PWMA_FTST)

Address: 4003_3000h base + 192h offset = 4003_3192h

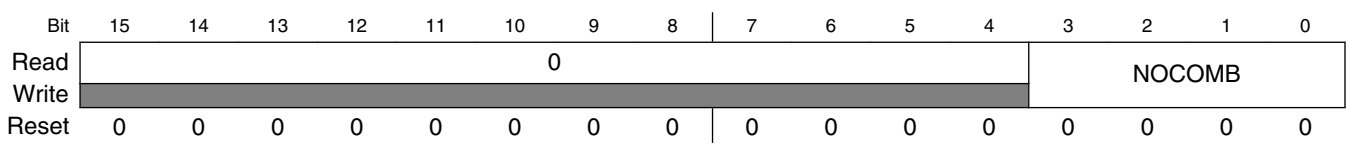


PWMA_FTST field descriptions

Field	Description
15–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 FTEST	Fault Test This read/write bit is used to simulate a fault condition. Setting this bit causes a simulated fault to be sent into all of the fault filters. The condition propagates to the fault flags and possibly the PWM outputs depending on the DISMAPn settings. Clearing this bit removes the simulated fault condition. 0 No fault 1 Cause a simulated fault

37.4.43 Fault Control 2 Register (PWMA_FCTRL2)

Address: 4003_3000h base + 194h offset = 4003_3194h



PWMA_FCTRL2 field descriptions

Field	Description
15–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
NOCOMB	No Combinational Path From Fault Input To PWM Output This read/write field is used to control the combinational path from the fault inputs to the PWM outputs. When these bits are low (default), the corresponding fault inputs have a combinational path to the PWM outputs that are sensitive to these fault inputs (as defined by DISMAP0). This combinational path is a safety feature that ensures the output is disabled even if the SOC has a failure of its clocking system. The combinational path also means that a pulse on the fault input can cause a brief disable of the PWM output even if the fault pulse is not wide enaough to get through the input filter and be latched in the fault logic.

Table continues on the next page...

PWMA_FCTRL2 field descriptions (continued)

Field	Description
	Setting these bits removes the combinational path and uses the filterred and latched fault signals as the fault source to disable the PWM outputs. This eliminates fault glitches from creating PWM output glitches but also increases the latency to respond to a real fault. 0 There is a combinational link from the fault inputs to the PWM outputs. The fault inputs are combined with the filtered and latched fault signals to disable the PWM outputs. 1 The direct combinational path from the fault inputs to the PWM outputs is disabled and the filtered and latched fault signals are used to disable the PWM outputs.

37.5 Functional Description

37.5.1 PWM Capabilities

This section describes some capabilities of the PWM module.

37.5.1.1 Center Aligned PWMs

Each submodule has its own timer that is capable of generating PWM signals on two output pins. The edges of each of these signals are controlled independently as shown in Figure 37-3.

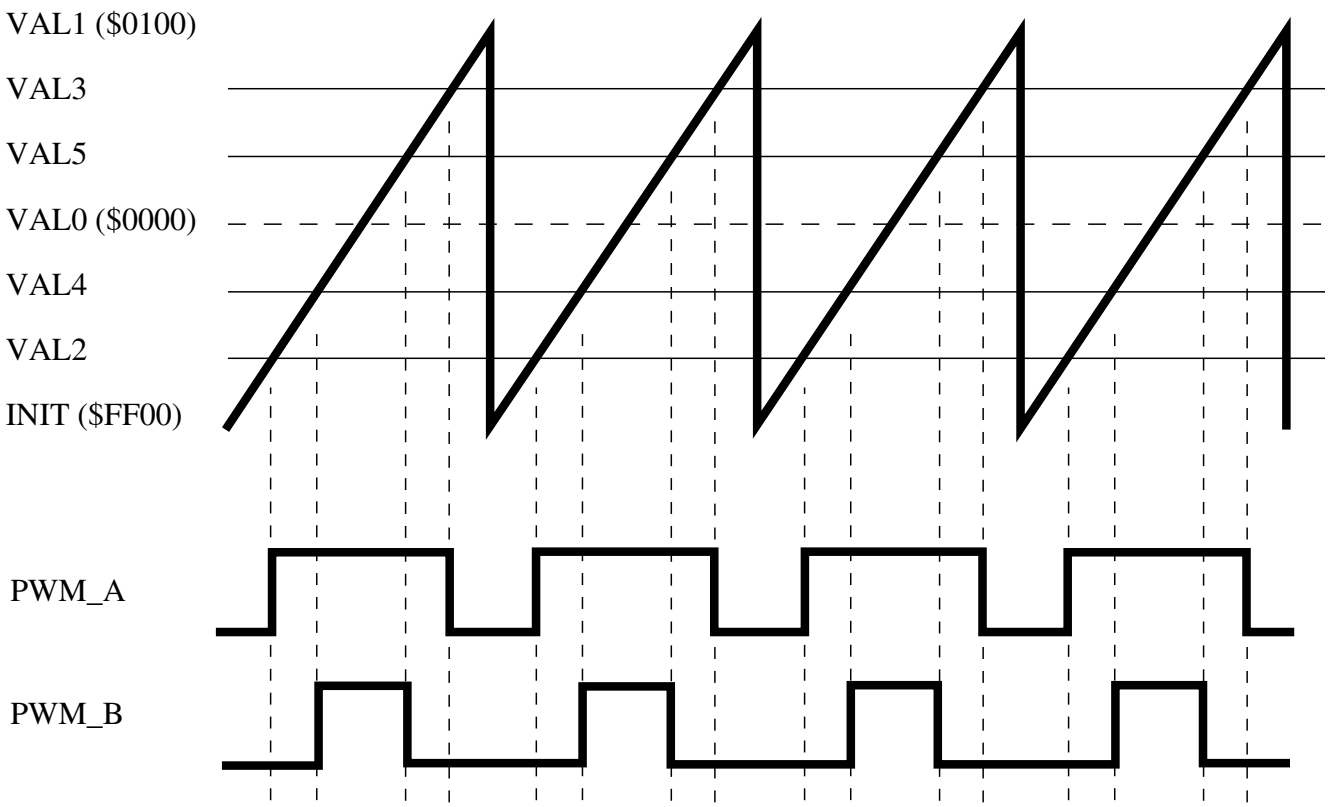


Figure 37-3. Center Aligned Example

The submodule timers only count in the up direction and then reset to the INIT value. Instead of having a single value that determines pulse width, there are two values that must be specified: the turn on edge and the turn off edge. This double action edge generation not only gives the user control over the pulse width, but over the relative alignment of the signal as well. As a result, there is no need to support separate PWM alignment modes since the PWM alignment mode is inherently a function of the turn on and turn off edge values.

Figure 37-3 also illustrates an additional enhancement to the PWM generation process. When the counter resets, it is reloaded with a user specified value, which may or may not be zero. If the value chosen happens to be the 2's complement of the modulus value, then the PWM generator operates in "signed" mode. This means that if each PWM's turn on and turn off edge values are also the same number but only different in their sign, the "on" portion of the output signal will be centered around a count value of zero. Therefore, only one PWM value needs to be calculated in software and then this value and its negative are provided to the submodule as the turn off and turn on edges respectively. This technique will result in a pulse width that always consists of an odd number of timer counts. If all PWM signal edge calculations follow this same convention, then the signals will be center aligned with respect to each other, which is the goal. Of course, center

alignment between the signals is not restricted to symmetry around the zero count value, as any other number would also work. However, centering on zero provides the greatest range in signed mode and also simplifies the calculations.

37.5.1.2 Edge Aligned PWMs

When the turn on edge for each pulse is specified to be the INIT value, then edge aligned operation results, as the following figure shows. Therefore, only the turn off edge value needs to be periodically updated to change the pulse width.

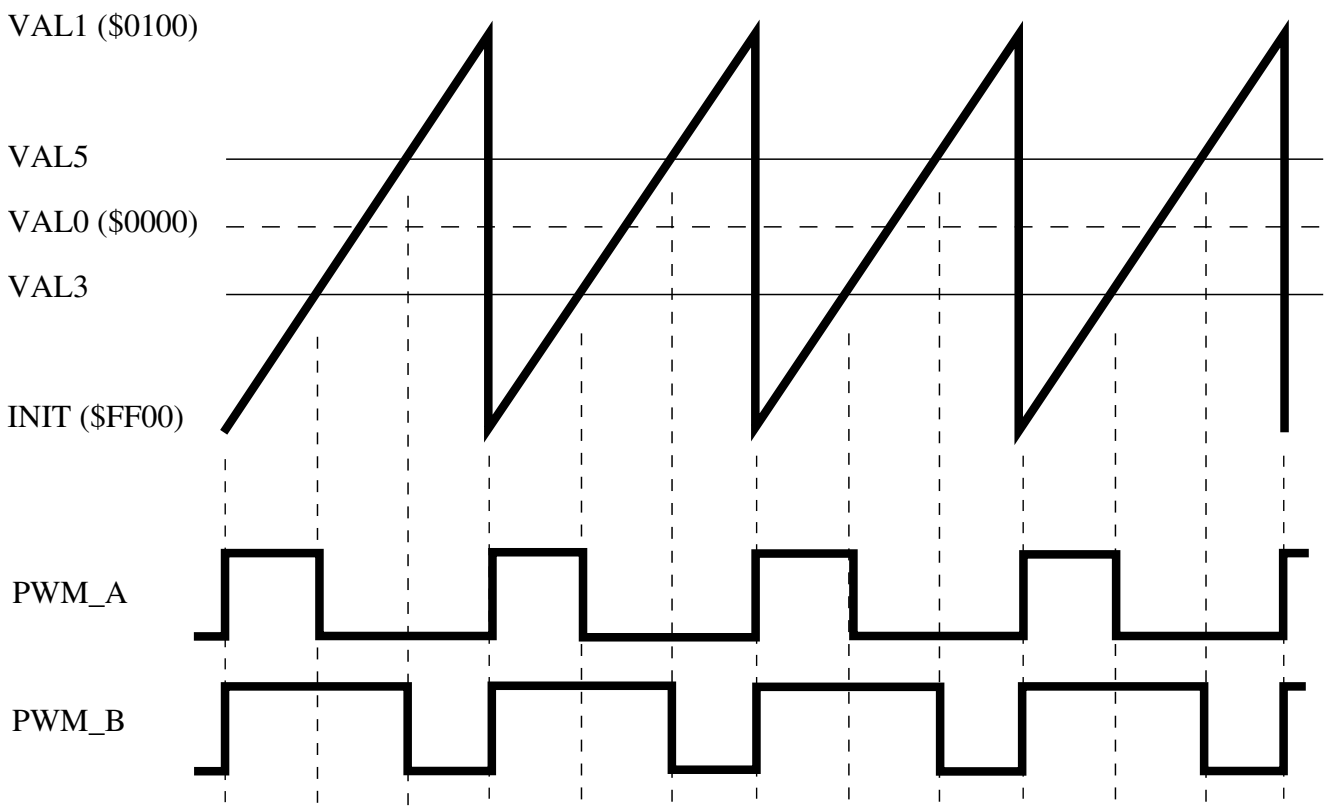


Figure 37-4. Edge Aligned Example (INIT=VAL2=VAL4)

With edge aligned PWMs, another example of the benefits of signed mode can be seen. A common way to drive an H-bridge is to use a technique called "bipolar" PWMs where a 50% duty cycle results in zero volts on the load. Duty cycles less than 50% result in negative load voltages and duty cycles greater than 50% generate positive load voltages. If the module is set to signed mode operation (the INIT and VAL1 values are the same number with opposite signs), then there is a direct proportionality between the PWM turn off edge value and the voltage, INCLUDING the sign. So once again, signed mode of operation simplifies the software interface to the PWM module since no offset calculations are required to translate the output variable control algorithm to the voltage on an H-Bridge load.

37.5.1.3 Phase Shifted PWMs

In the previous sections, the benefits of signed mode of operation were discussed in the context of simplifying the required software calculations by eliminating the requirement to bias up signed variables before applying them to the module. However, if numerical biases are applied to the turn on and turn off edges of different PWM signal, the signals will be phase shifted with respect to each other, as the following figure shows. This results in certain advantages when applied to a power stage. For example, when operating a multi-phase inverter at a low modulation index, all of the PWM switching edges from the different phases occur at nearly the same time. This can be troublesome from a noise standpoint, especially if ADC readings of the inverter must be scheduled near those times. Phase shifting the PWM signals can open up timing windows between the switching edges to allow a signal to be sampled by the ADC. However, phase shifting does NOT affect the duty cycle so average load voltage is not affected.

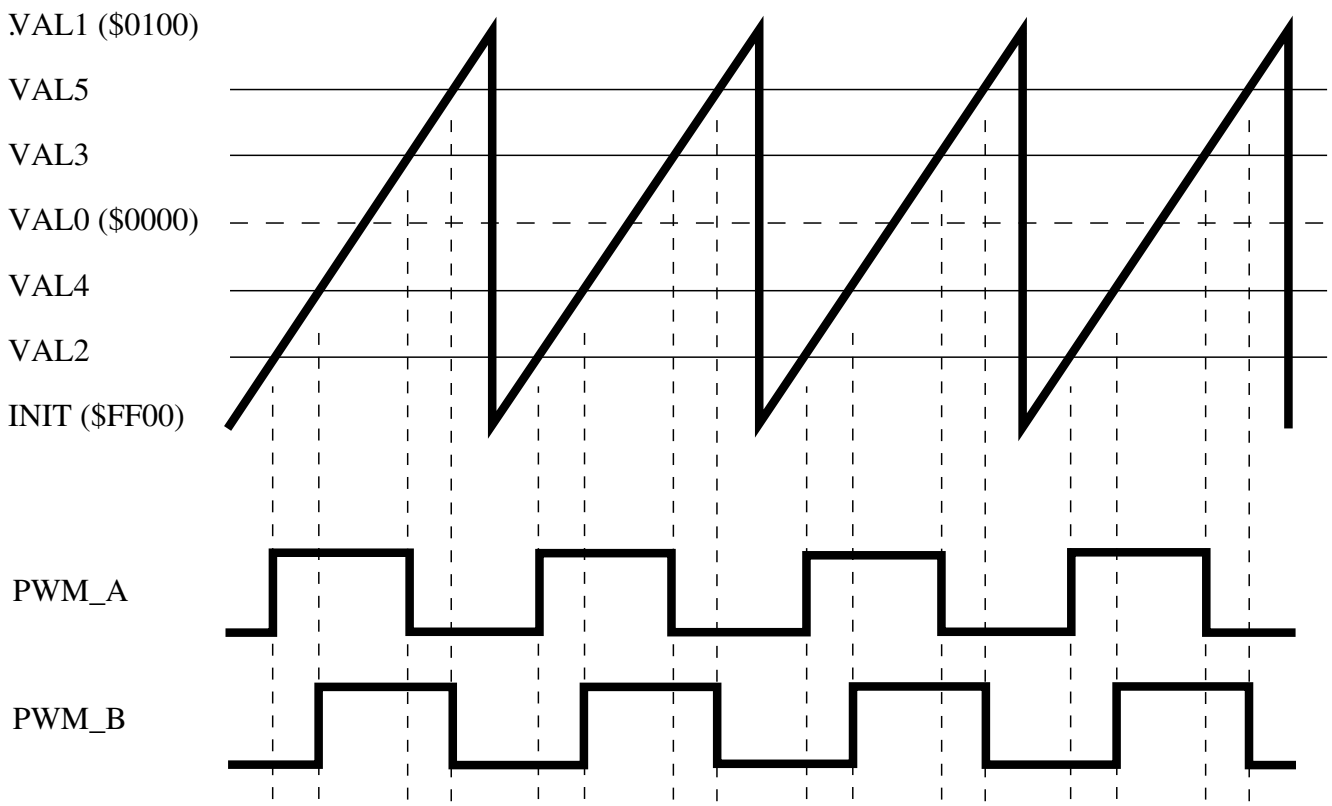


Figure 37-5. Phase Shifted Outputs Example

An additional benefit of phase shifted PWMs appears in Figure 37-6. In this case, an H-Bridge circuit is driven by 4 PWM signals to control the voltage waveform on the primary of a transformer. Both left and right side PWMs are configured to always generate a square wave with 50% duty cycle. This works out nicely for the H-Bridge since no narrow pulse widths are generated reducing the high frequency switching

requirements of the transistors. Notice that the square wave on the right side of the H-Bridge is phase shifted compared to the left side of the H-Bridge. As a result, the transformer primary sees the bottom waveform across its terminals. The RMS value of this waveform is directly controlled by the amount of phase shift of the square waves. Regardless of the phase shift, no DC component appears in the load voltage as long as the duty cycle of each square wave remains at 50% making this technique ideally suited for transformer loads. As a result, this topology is frequently used in industrial welders to adjust the amount of energy delivered to the weld arc.

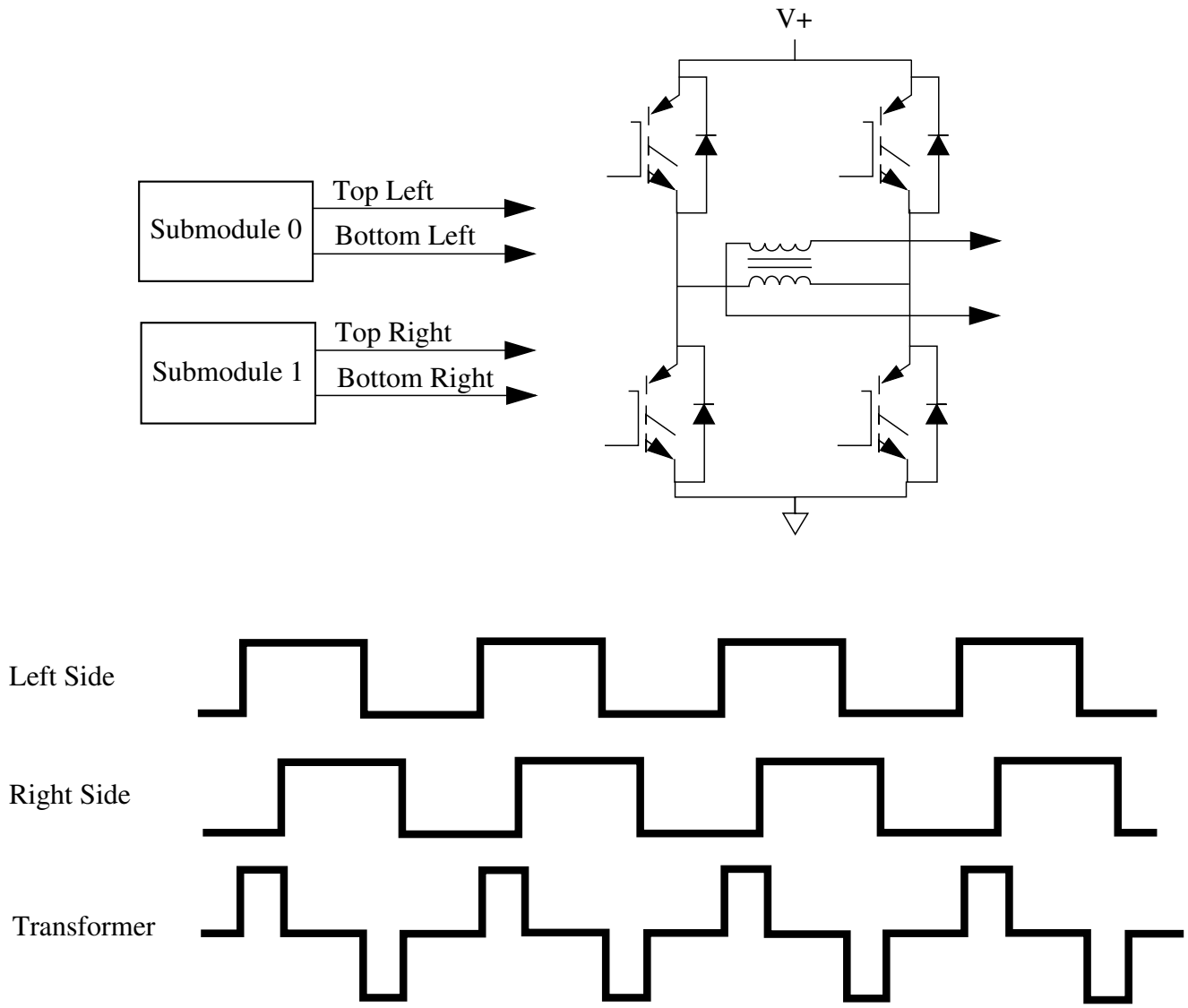


Figure 37-6. Phase Shifted PWMs Applied to a Transformer Primary

37.5.1.4 Double Switching PWMs

Double switching PWM output is supported to aid in single shunt current measurement and three phase reconstruction. This method support two independent rising edges and two independent falling edges per PWM cycle. The VAL2 and VAL3 registers are used to generate the even channel (labelled as PWM_A in the figure) while VAL4 and VAL5 are used to generate the odd channel. The two channels are combined using XOR logic (force out logic) as the following figure shows. The DBLPWM signal can be run through the deadtime insertion logic.

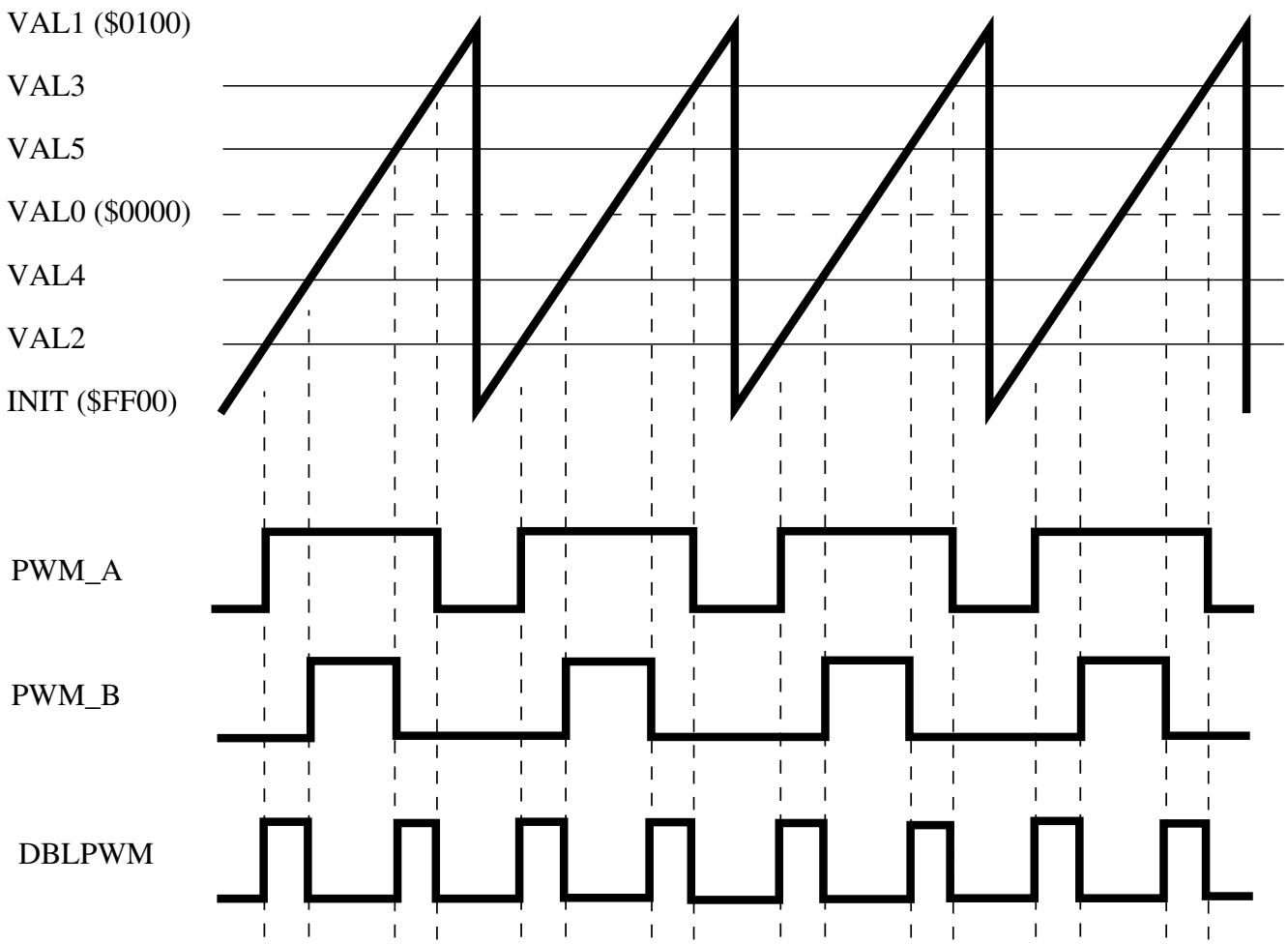


Figure 37-7. Double Switching Output Example

37.5.1.5 ADC Triggering

In cases where the timing of the ADC triggering is critical, it must be scheduled as a hardware event instead of software activated. With this PWM module, multiple ADC triggers can be generated in hardware per PWM cycle without the requirement of another timer module. Figure 37-8 shows how this is accomplished. When specifying

complementary mode of operation, only two edge comparators are required to generate the output PWM signals for a given submodule. This means that the other comparators are free to perform other functions. In this example, the software does not need to quickly respond after the first conversion to set up other conversions that must occur in the same PWM cycle.

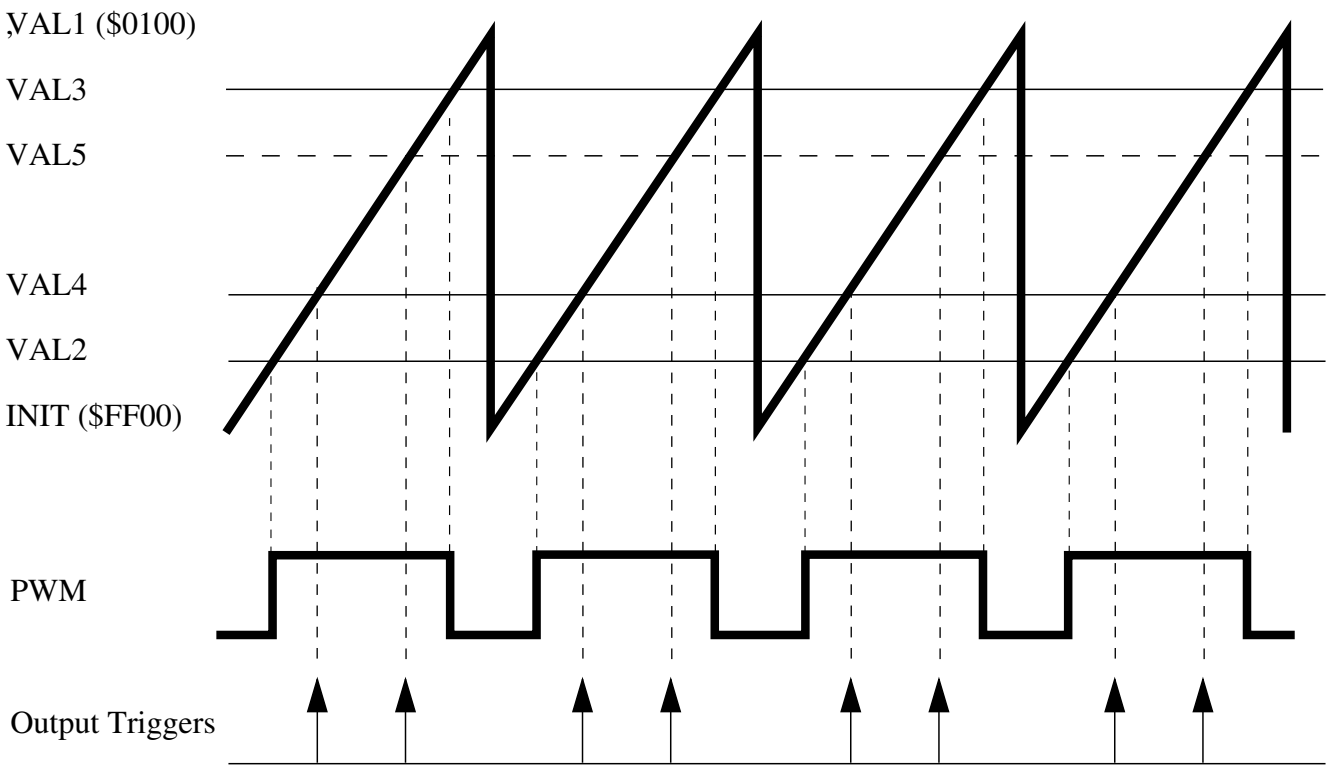


Figure 37-8. Multiple Output Trigger Generation in Hardware

Because each submodule has its own timer, it is possible for each submodule to run at a different frequency. One of the options possible with this PWM module is to have one or more submodules running at a lower frequency, but still synchronized to the timer in submodule0. Figure 37-9 shows how this feature can be used to schedule ADC triggers over multiple PWM cycles. A suggested use for this configuration would be to use the lower-frequency submodule to control the sampling frequency of the software control algorithm where multiple ADC triggers can now be scheduled over the entire sampling period. In Figure 37-9, *all* submodule comparators are shown being used for ADC trigger generation.

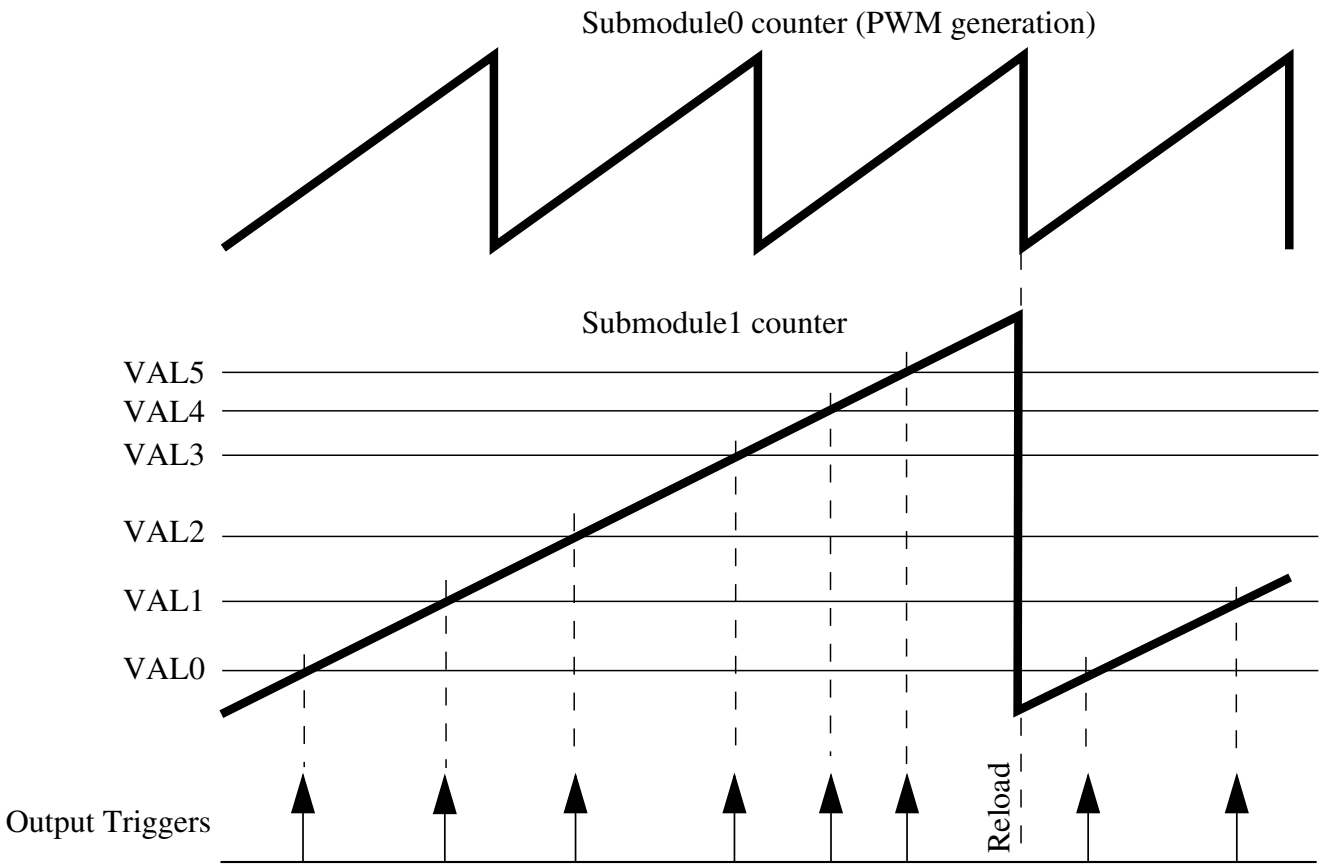


Figure 37-9. Multiple Output Triggers Over Several PWM Cycles

37.5.1.6 Enhanced Capture Capabilities (E-Capture)

When a PWM pin is not being used for PWM generation, it can be used to perform input captures. Recall that for PWM generation BOTH edges of the PWM signal are specified via separate compare register values. When programmed for input capture, both of these registers work on the same pin to capture multiple edges, toggling from one to the other in either a free running or one-shot fashion. By simply programming the desired edge of each capture circuit, period and pulse width of an input signal can easily be measured without the requirement to re-arm the circuit. In addition, each edge of the input signal can clock an 8 bit counter where the counter output is compared to a user specified value (EDGCMP). When the counter output equals EDGCMP, the value of the submodule timer is captured and the counter is automatically reset. This feature allows the module to count a specified number of edge events and then perform a capture and interrupt. The following figure illustrates some of the functionality of the E-Capture circuit.

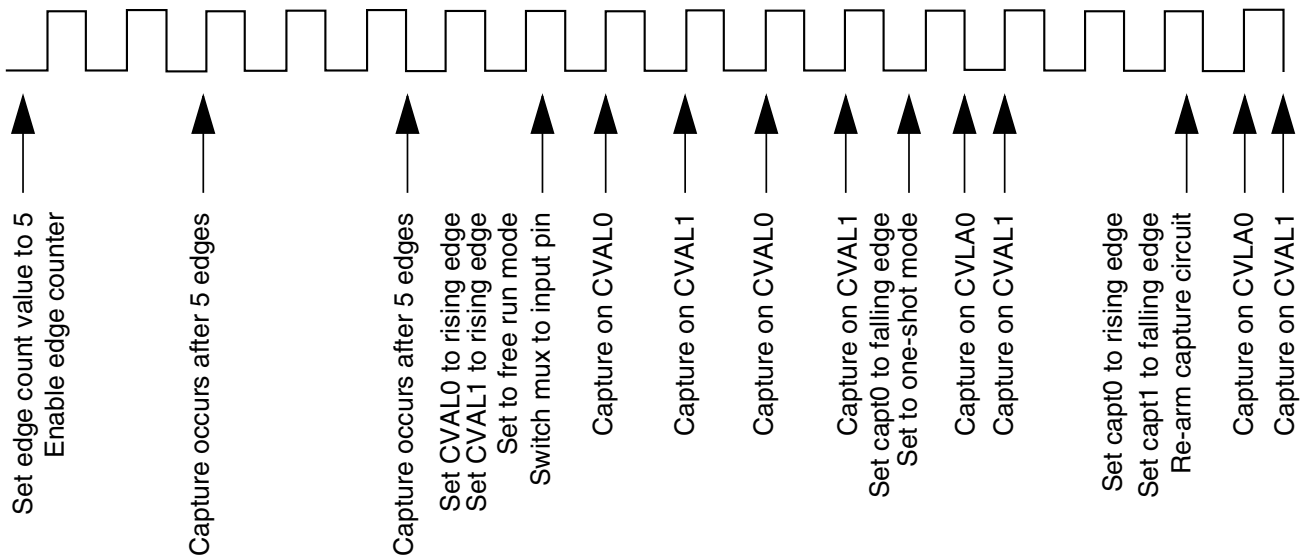


Figure 37-10. Capture Capabilities of the E-Capture Circuit

When a submodule is being used for PWM generation, its timer counts up to the modulus value used to specify the PWM frequency and then is re-initialized. Therefore, using this timer for input captures on one of the other pins (for example, PWM_X) has limited utility since it does not count through all of the numbers and the timer reset represents a discontinuity in the 16 bit number range. However, when measuring a signal that is synchronous to the PWM frequency, the timer modulus range is perfectly suited for the application. Consider the following figure as an example. In this application the output of a PWM power stage is connected to the PWM_X pin that is configured for free running input captures. Specifically, the CVAL0 capture circuitry is programmed for rising edges and the CVAL1 capture circuitry is set for falling edges. This will result in new load pulse width data being acquired every PWM cycle. To calculate the pulse width, simply subtract the CVAL0 register value from the CVAL1 register value. This measurement is extremely beneficial when performing dead-time distortion correction on a half bridge circuit driving an inductive load. Also, these values can be directly compared to the VALx registers responsible for generating the PWM outputs to obtain a measurement of system propagation delays. For details, refer to the separate discussion of deadtime distortion correction.

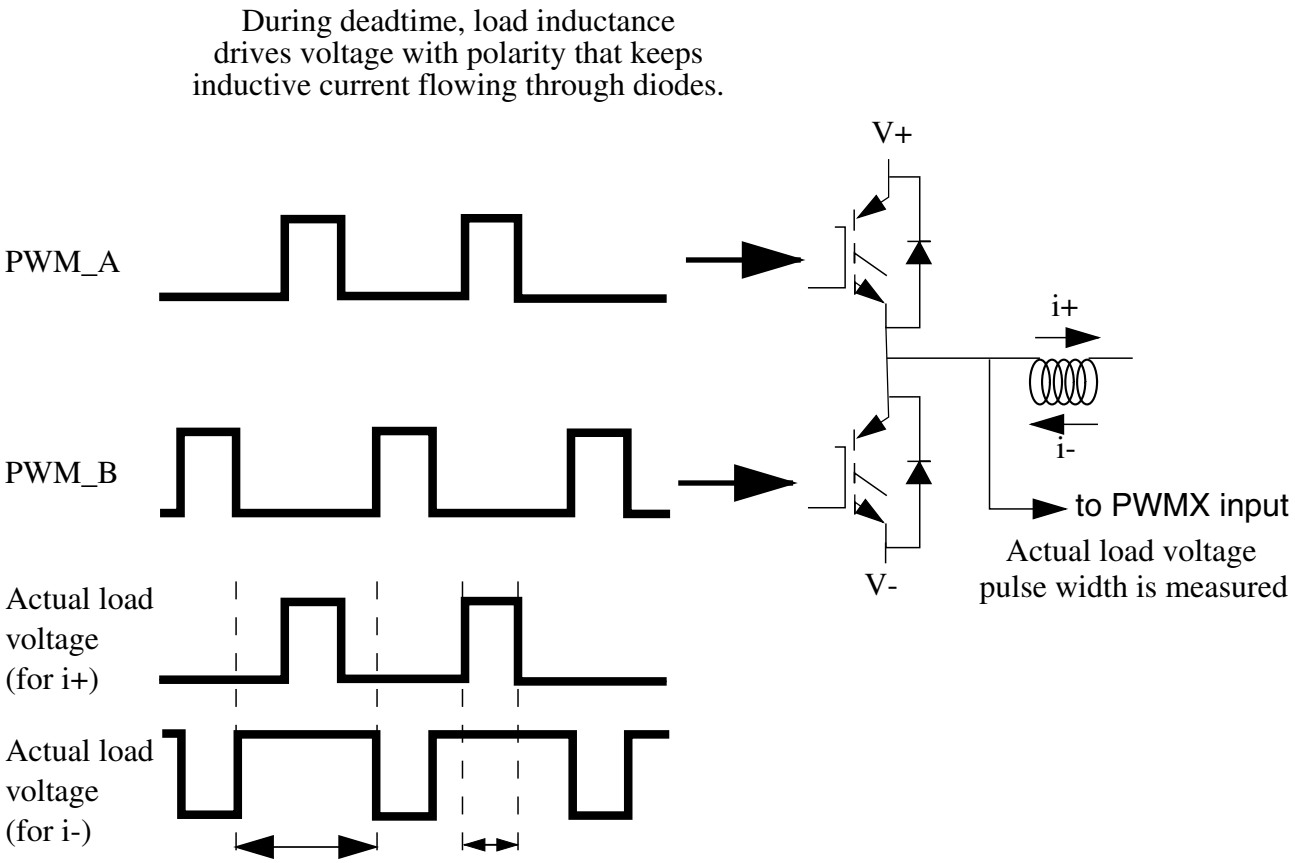


Figure 37-11. Output Pulse Width Measurement Possible with the E-Capture Circuit

37.5.1.7 Synchronous Switching of Multiple Outputs

Before the PWM signals are routed to the output pins, they are processed by a hardware block that permits all submodule outputs to be switched synchronously. Not only do all the changes occur synchronously on all submodule outputs, but they occur IMMEDIATELY after the trigger event occurs eliminating any interrupt latency.

The synchronous output switching is accomplished via a signal called FORCE_OUT. This signal originates from the local FORCE bit within the submodule, from submodule0, or from external to the PWM module and, in most cases, is supplied from an external timer channel configured for output compare. In a typical application, software sets up the desired states of the output pins in preparation for the next FORCE_OUT event. This selection lays dormant until the FORCE_OUT signal transitions and then all outputs are switched simultaneously. The signal switching is performed upstream from the deadtime generator so that any abrupt changes that might occur do not violate deadtime on the power stage when in complementary mode.

37.5.2 Functional Details

This section describes the implementation of various sections of the PWM in greater detail.

The following figure is a high-level block diagram of output PWM generation.

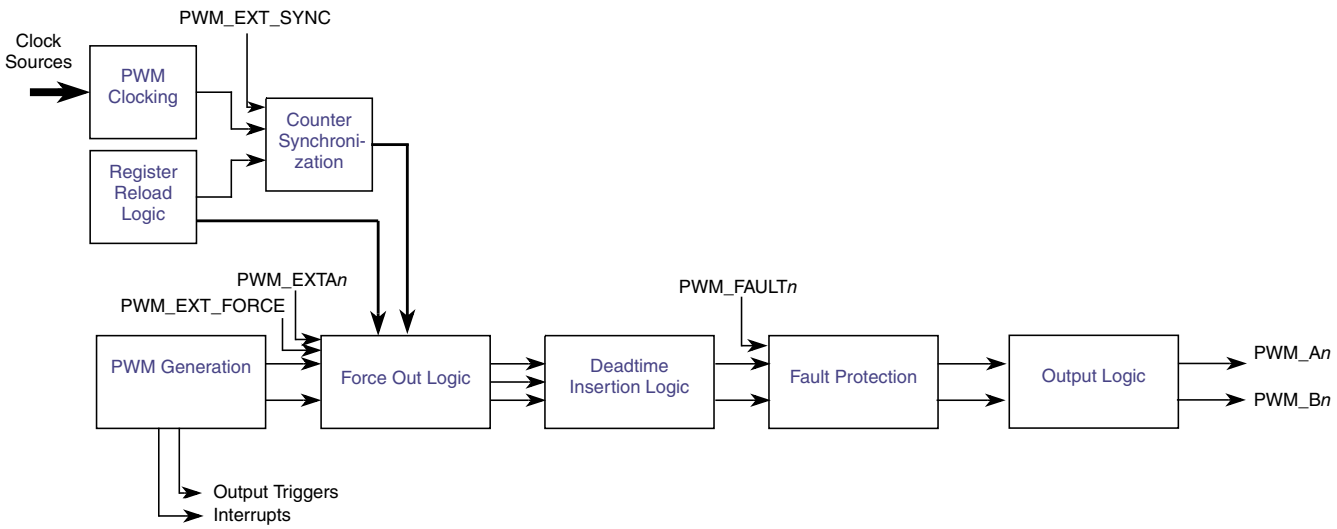


Figure 37-12. High-Level Output PWM Generation Block Diagram

37.5.2.1 PWM Clocking

Figure 37-13 shows the logic used to generate the main counter clock. Each submodule can select between three clock signals: the IPBus clock, EXT_CLK, and AUX_CLK. The EXT_CLK goes to all of the submodules. The AUX_CLK signal is broadcast from submodule0 and can be selected as the clock source by other submodules so that the 8-bit prescaler and MCTRL[RUN] from submodule0 can control all of the submodules.

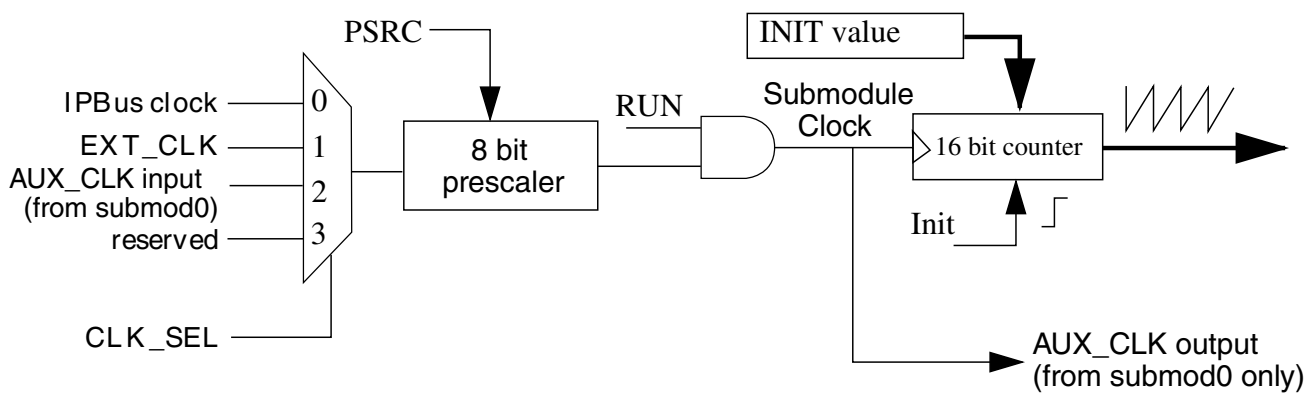


Figure 37-13. Clocking Block Diagram for Each PWM Submodule

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the IPBus clock frequency by 1-128. The prescaler bits, CTRL[PRSC], select the prescaler divisor. This prescaler is buffered and will not be used by the PWM generator until MCTRL[LDOK] is set and a new PWM reload cycle begins or CTRL[LDMOD] is set.

37.5.2.2 Register Reload Logic

The register reload logic is used to determine when the outer set of registers for all double buffered register pairs will be transferred to the inner set of registers. The register reload event can be scheduled to occur every "n" PWM cycles using CTRL[LDFQ] and CTRL[FULL]. A half cycle reload option is also supported (CTRL[HALF]) where the reload can take place in the middle of a PWM cycle. The half cycle point is defined by the VAL0 register and does not have to be exactly in the middle of the PWM cycle.

As illustrated in Figure 37-14 the reload signal from submodule0 can be broadcast as the Master Reload signal allowing the reload logic from submodule0 to control the reload of registers in other submodules.

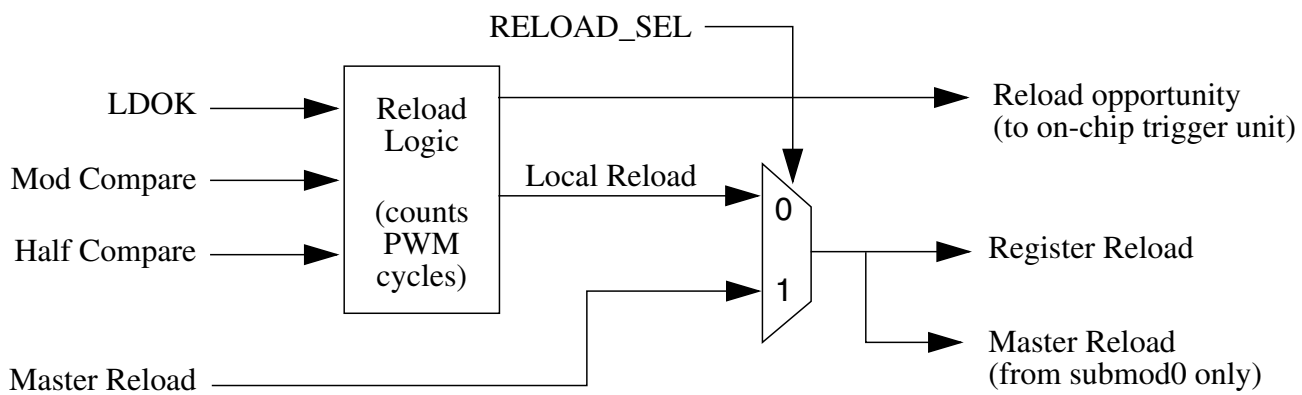


Figure 37-14. Register Reload Logic

37.5.2.3 Counter Synchronization

In the following figure, the 16 bit counter will count up until its output equals VAL1 which is used to specify the counter modulus value. The resulting compare causes a rising edge to occur on the Local Sync signal which is one of four possible sources used to cause the 16 bit counter to be initialized with INIT. If Local Sync is selected as the counter initialization signal, then VAL1 within the submodule effectively controls the timer period (and thus the PWM frequency generated by that submodule) and everything works on a local level.

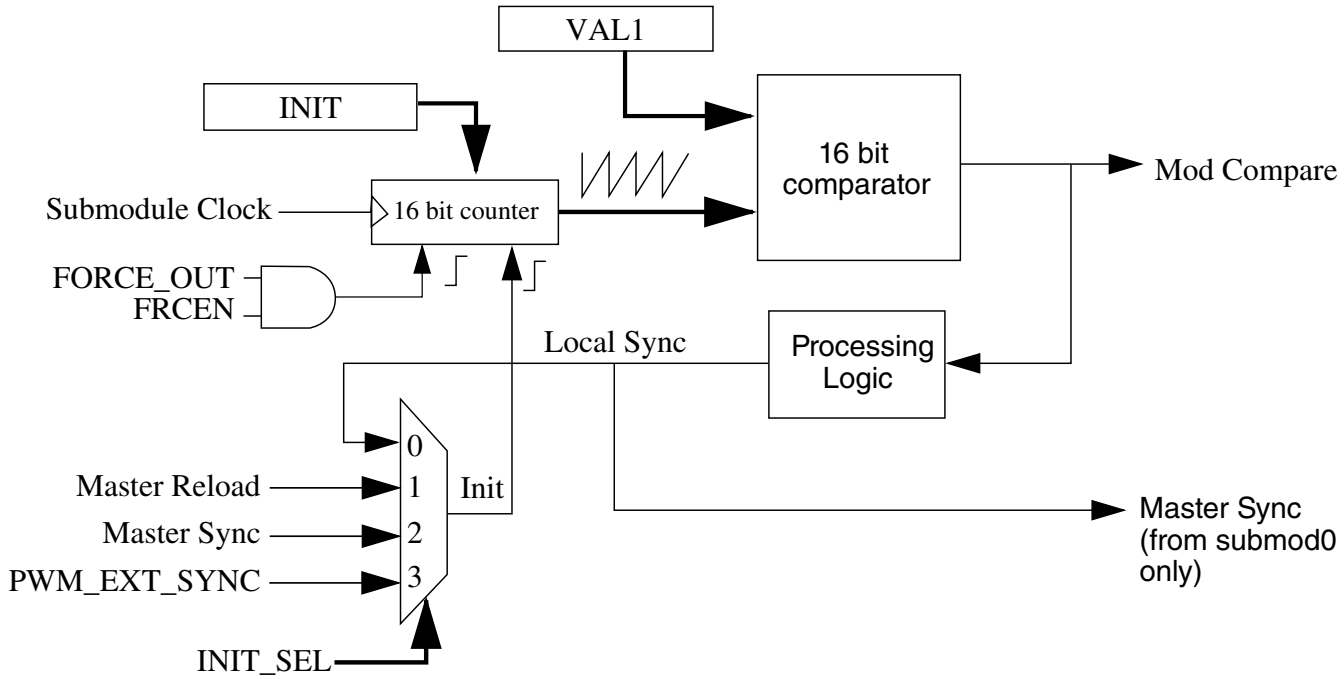


Figure 37-15. Submodule Timer Synchronization

The Master Sync signal originates as the Local Sync from submodule0. If configured to do so, the timer period of any submodule can be locked to the period of the timer in submodule0.

The PWM_EXT_SYNC signal originates on chip or off chip depending on the system architecture. This signal may be selected as the source for counter initialization so that an external source can control the period of all submodules.

If the Master Reload signal is selected as the source for counter initialization, then the period of the counter will be locked to the register reload frequency of submodule0. Since the reload frequency is usually commensurate to the sampling frequency of the software control algorithm, the submodule counter period will therefore equal the sampling period. As a result, this timer can be used to generate output compares or output triggers over the entire sampling period which may consist of several PWM cycles. The Master Reload signal can only originate from submodule0.

The counter can optionally initialize upon the assertion of the FORCE_OUT signal assuming that CTRL2[FRCEN] is set. As indicated by the preceding figure, this constitutes a second init input into the counter which will cause the counter to initialize regardless of which signal is selected as the counter init signal.

37.5.2.4 PWM Generation

Figure 37-16 illustrates how PWM generation is accomplished in each submodule. In each case, two comparators and associated VALx registers are utilized for each PWM output signal. One comparator and VALx register are used to control the turn-on edge, while a second comparator and VALx register control the turn-off edge.

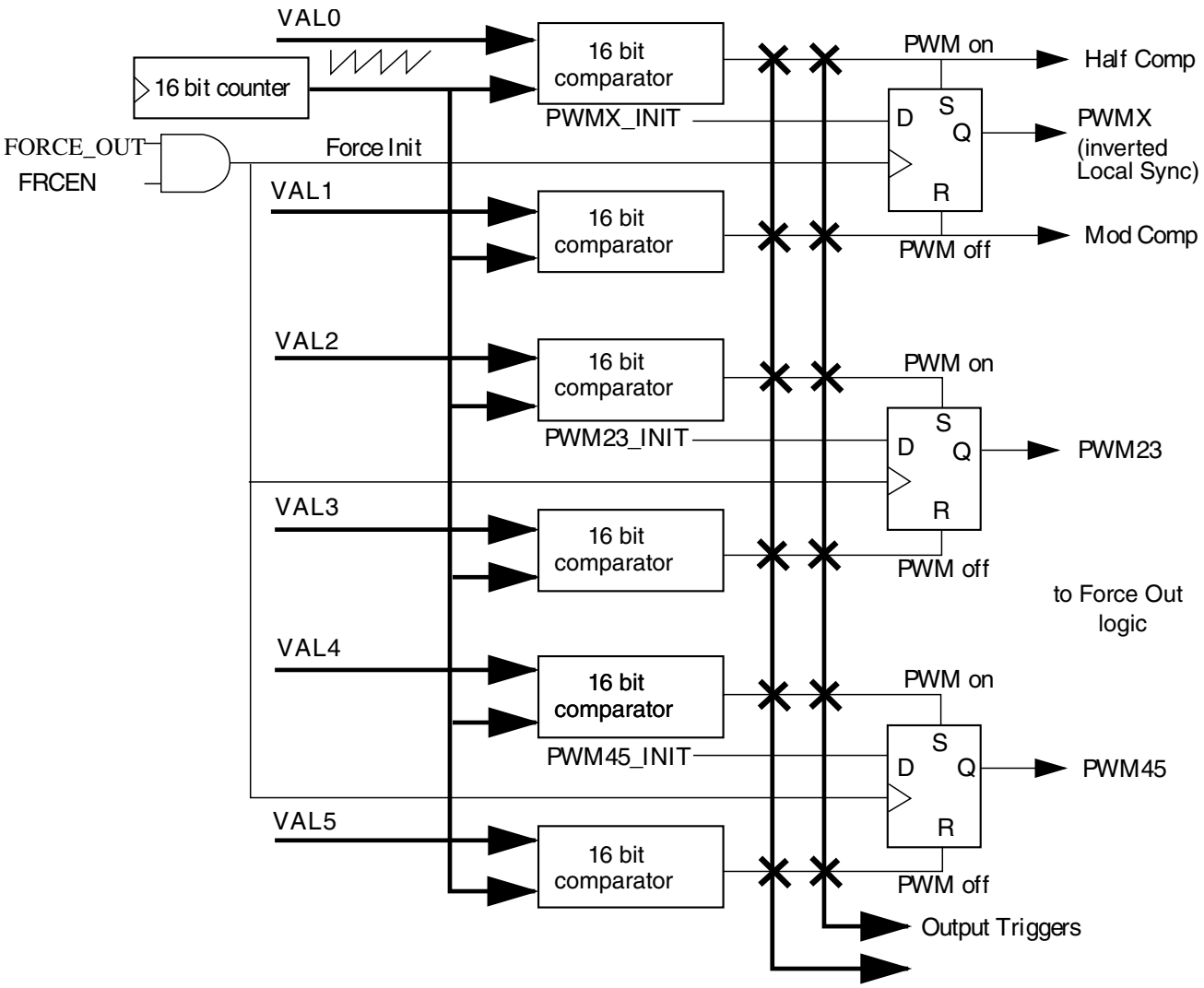


Figure 37-16. PWM Generation Hardware

The generation of the Local Sync signal is performed exactly the same way as the other PWM signals in the submodule. While comparator 0 causes a falling edge of the Local Sync signal, comparator 1 generates a rising edge. Comparator 1 is also hardwired to the reload logic to generate the full cycle reload indicator.

If VAL1 is controlling the modulus of the counter and VAL0 is half of the VAL1 register minus the INIT value, then the half cycle reload pulse will occur exactly half way through the timer count period and the Local Sync will have a 50% duty cycle. On the other hand, if the VAL1 and VAL0 registers are not required for register reloading or

counter initialization, they can be used to modulate the duty cycle of the Local Sync signal, effectively turning it into an auxiliary PWM signal (PWM_X) assuming that the PWM_X pin is not being used for another function such as input capture or deadtime distortion correction. Including the Local Sync signal, each submodule is capable of generating three PWM signals where software has complete control over each edge of each of the signals.

If the comparators and edge value registers are not required for PWM generation, they can also be used for other functions such as output compares, generating output triggers, or generating interrupts at timed intervals.

The 16-bit comparators shown in Figure 37-16 In addition, if both the set and reset of the flip-flop are asserted, then the flop output goes to 0.

37.5.2.5 Output Compare Capabilities

By using the VALx registers in conjunction with the submodule timer and 16 bit comparators, buffered output compare functionality can be achieved with no additional hardware required. Specifically, the following output compare functions are possible:

- An output compare sets the output high
- An output compare sets the output low
- An output compare generates an interrupt
- An output compare generates an output trigger

In PWM generation, an output compare is initiated by programming a VALx register for a timer compare, which in turn causes the output of the D flip-flop to either set or reset. For example, if an output compare is desired on the PWM_A signal that sets it high, VAL2 would be programmed with the counter value where the output compare should take place. However, to prevent the D flip-flop from being reset again after the compare has occurred, the VAL3 register must be programmed to a value outside of the modulus range of the counter. Therefore, a compare that would result in resetting the D flip-flop output would never occur. Conversely, if an output compare is desired on the PWM_A signal that sets it low, the VAL3 register is programmed with the appropriate count value and the VAL2 register is programmed with a value outside the counter modulus range. Regardless of whether a high compare or low compare is programmed, an interrupt or output trigger can be generated when the compare event occurs.

37.5.2.6 Force Out Logic

For each submodule, software can select between eight signal sources for the FORCE_OUT signal: local CTRL2[FORCE], the Master Force signal from submodule0, the local Reload signal, the Master Reload signal from submodule0, the Local Sync signal, the Master Sync signal from submodule0, the EXT_SYNC signal from on- or off-chip, or the EXT_FORCE signal from on- or off-chip depending on the chip architecture. The local signals are used when the user simply wants to change the signals on the output pins of the submodule without regard for synchronization with other submodules. However, if it is required that all signals on all submodule outputs change at the same time, the Master, EXT_SYNC, or EXT_FORCE signals should be selected.

Figure 37-17 illustrates the Force logic. The SEL23 and SEL45 fields each choose from one of four signals that can be supplied to the submodule outputs: the PWM signal, the inverted PWM signal, a binary level specified by software via the OUT23 and OUT45 bits, or the PWM_EXTA or PWM_EXTB alternate external control signals. The selection can be determined ahead of time and, when a FORCE_OUT event occurs, these values are presented to the signal selection mux that immediately switches the requested signal to the output of the mux for further processing downstream.

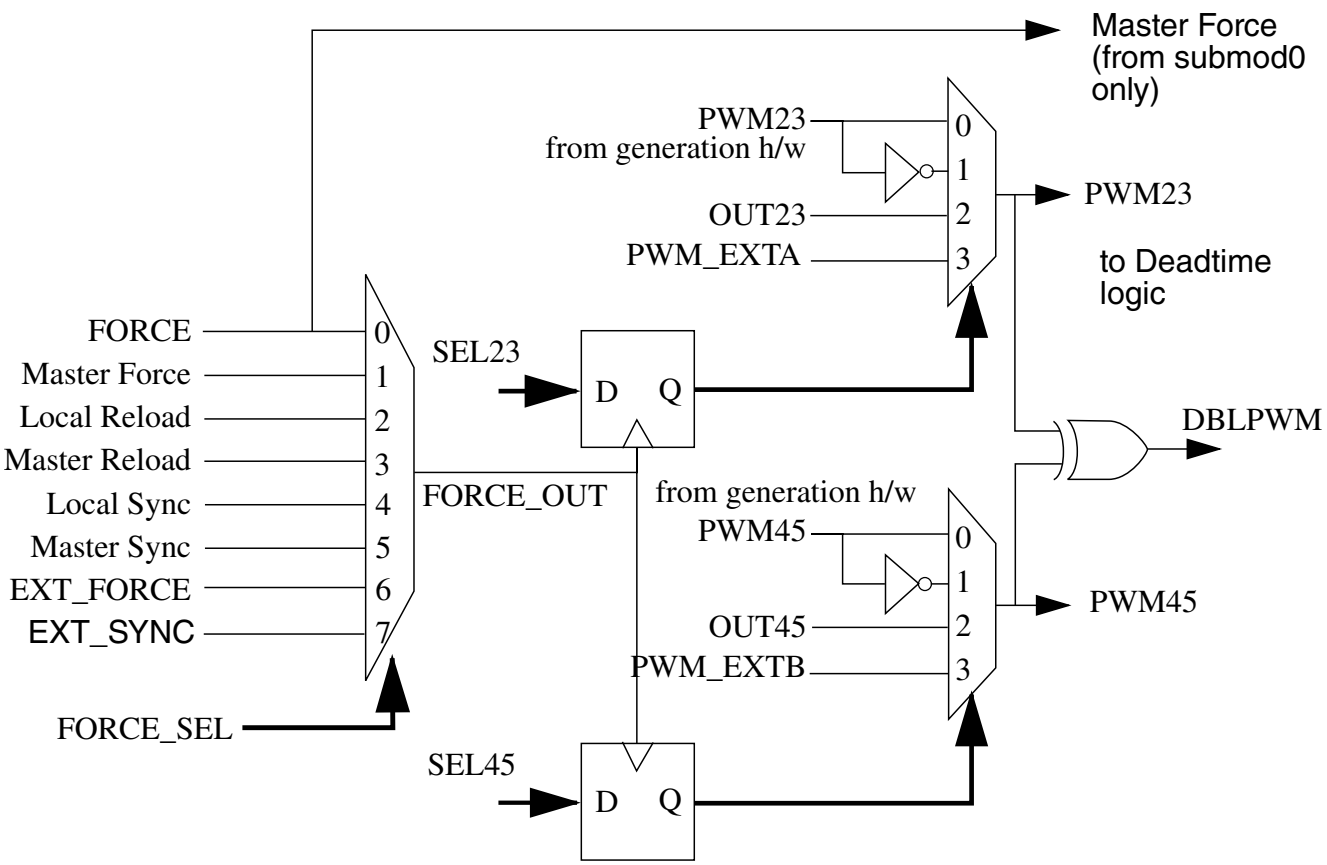


Figure 37-17. Force Out Logic

The local CTRL2[FORCE] signal of submodule0 can be broadcast as the Master Force signal to other submodules. This feature allows the CTRL2[FORCE] of submodule0 to synchronously update all of the submodule outputs at the same time. The EXT_FORCE signal originates from outside the PWM module from a source such as a timer or digital comparators in the Analog-to-Digital Converter.

37.5.2.7 Independent or Complementary Channel Operation

Writing a logic one to CTRL2[INDEP] configures the pair of PWM outputs as two independent PWM channels. Each PWM output is controlled by its own VALx pair operating independently of the other output.

Writing a logic zero to CTRL2[INDEP] configures the PWM output as a pair of complementary channels. The PWM pins are paired as shown in Figure 37-18 in complementary channel operation. Which signal is connected to the output pin (PWM23 or PWM45) is determined by MCTRL[IPOL].

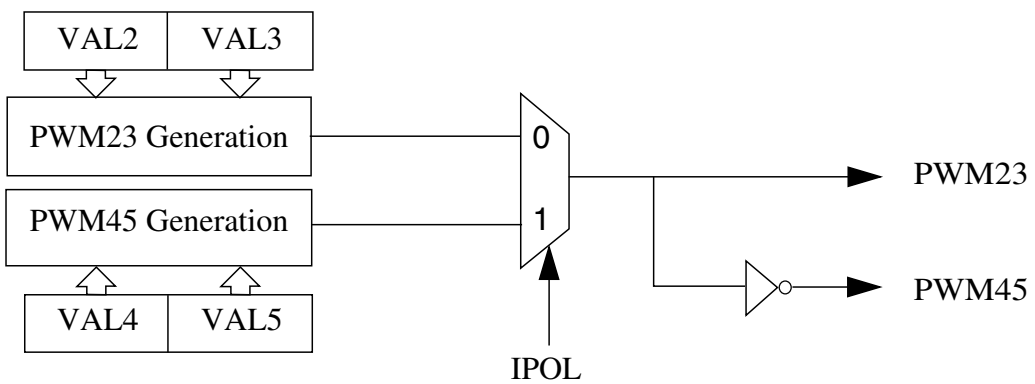


Figure 37-18. Complementary Channel Pair

Complementary operation allows the use of the deadtime insertion feature.

37.5.2.8 Deadtime Insertion Logic

The following figure shows the deadtime insertion logic of each submodule which is used to create non-overlapping complementary signals when not in independent mode.

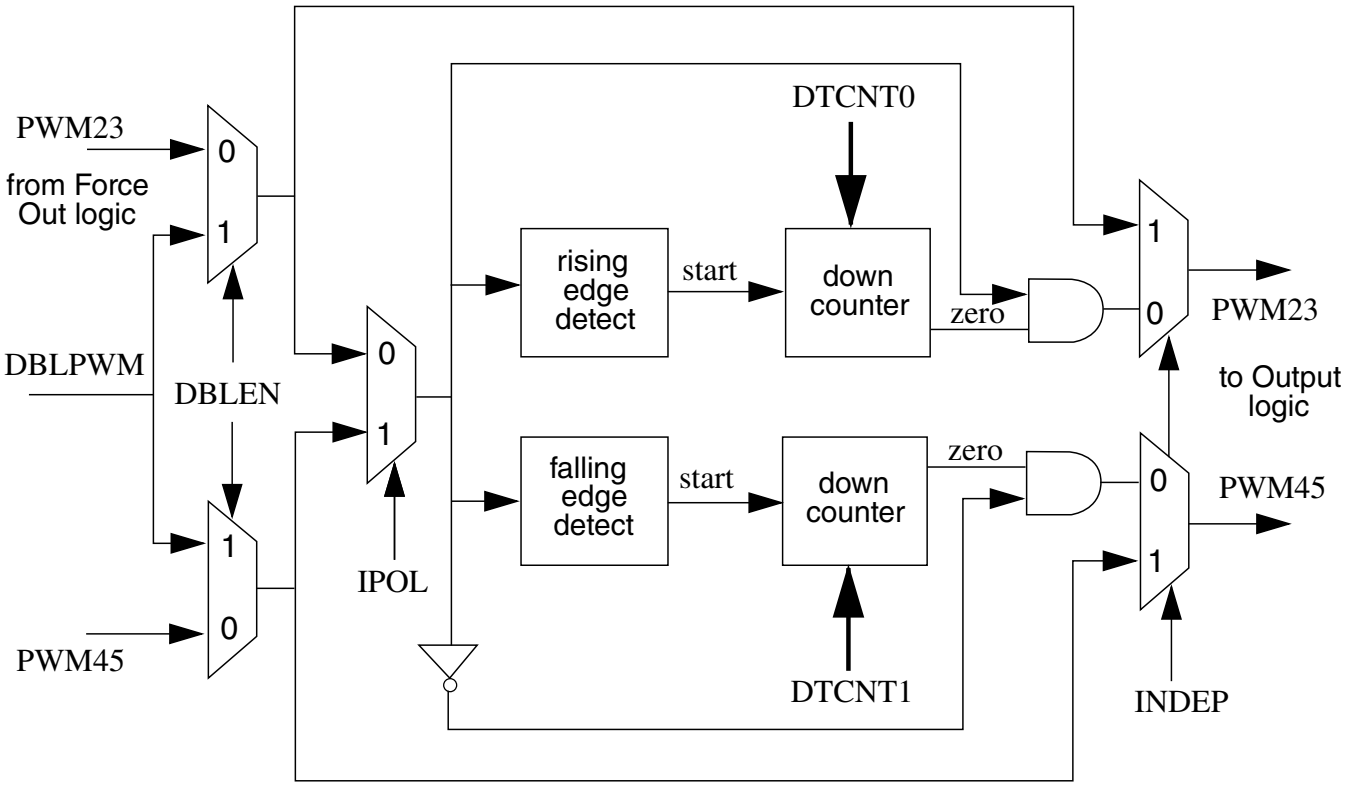


Figure 37-19. Deadtime Insertion Logic

While in the complementary mode, a PWM pair can be used to drive top/bottom transistors, as shown in the figure. When the top PWM channel is active, the bottom PWM channel is inactive, and vice versa.

Note

To avoid short circuiting the DC bus and endangering the transistor, there must be no overlap of conducting intervals between top and bottom transistor. But the transistor's characteristics may make its switching-off time longer than switching-on time. To avoid the conducting overlap of top and bottom transistors, deadtime needs to be inserted in the switching period, as illustrated in the following figure.

The deadtime generators automatically insert software-selectable activation delays into the pair of PWM outputs. The deadtime registers (DTCNT0 and DTCNT1) specify the number of IPBus clock cycles to use for deadtime delay. Every time the deadtime generator inputs change state, deadtime is inserted. Deadtime forces both PWM outputs in the pair to the inactive state.

When deadtime is inserted in complementary PWM signals connected to an inverter driving an inductive load, the PWM waveform on the inverter output will have a different duty cycle than what appears on the output pins of the PWM module. This results in a distortion in the voltage applied to the load. A method of correcting this, adding to or subtracting from the PWM value used, is discussed next.

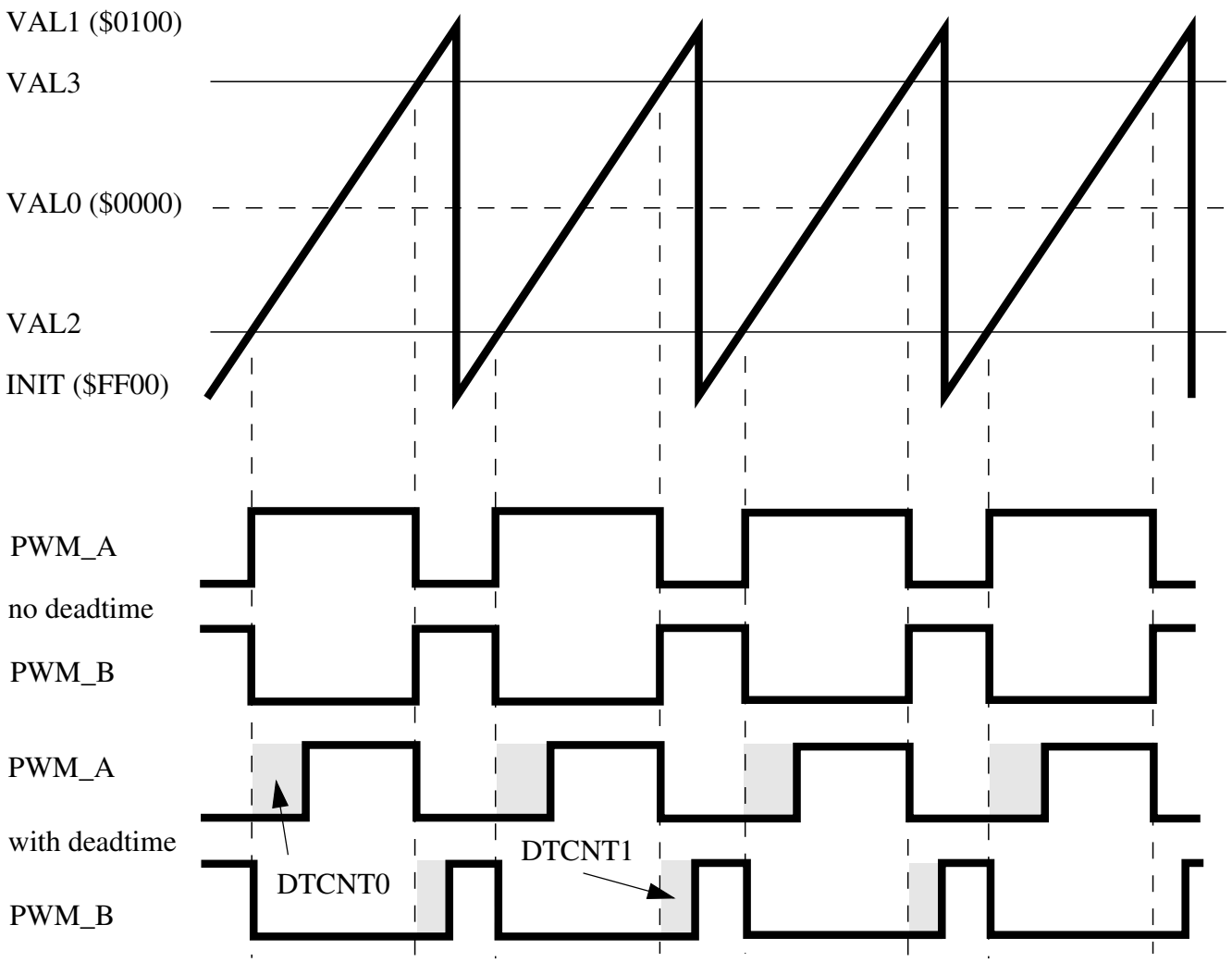


Figure 37-20. Deadtime Insertion

37.5.2.8.1 Top/Bottom Correction

In complementary mode, either the top or the bottom transistor controls the output voltage. However, deadtime has to be inserted to avoid overlap of conducting interval between the top and bottom transistor. Both transistors in complementary mode are off during deadtime, allowing the output voltage to be determined by the current status of load and introduce distortion in the output voltage. See the following figure.

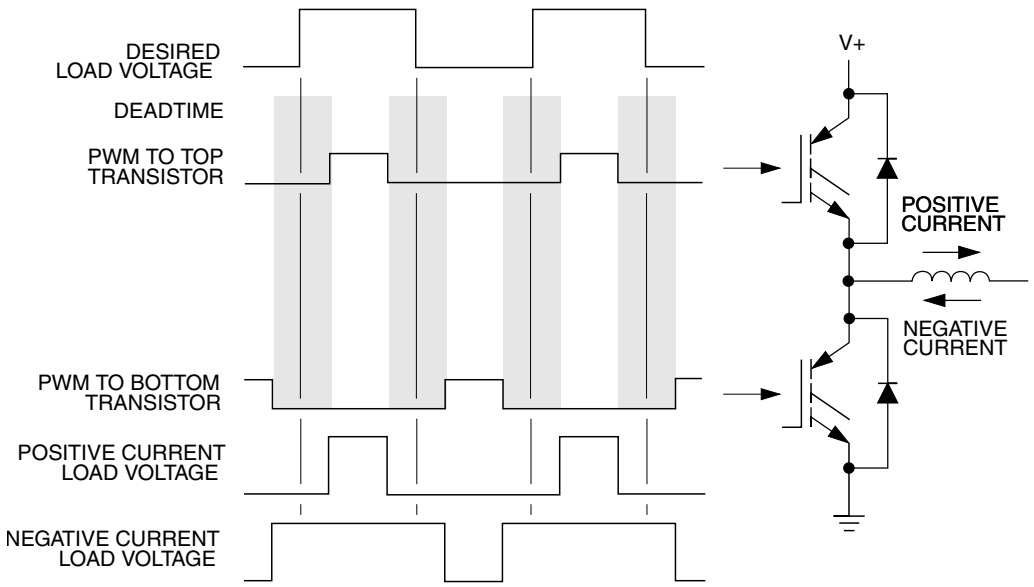


Figure 37-21. Deadtime Distortion

During deadtime, load inductance distorts output voltage by keeping current flowing through the diodes. This deadtime current flow creates a load voltage that varies with current direction. With a positive current flow, the load voltage during deadtime is equal to the bottom supply, putting the top transistor in control. With a negative current flow, the load voltage during deadtime is equal to the top supply putting the bottom transistor in control.

Remembering that the original PWM pulse widths where shortened by deadtime insertion, the averaged sinusoidal output will be less than the desired value. However, when deadtime is inserted, it creates a distortion in the . This distortion is aggravated by dissimilar turn-on and turn-off delays of each of the transistors. By giving the PWM module information on which transistor is controlling at a given time this distortion can be corrected.

For a typical circuit in complementary channel operation, only one of the transistors will be effective in controlling the output voltage at any given time. This depends on the direction of the current for that pair, as the preceding figure shows. To correct distortion one of two different factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the software is responsible for calculating both compensated PWM values prior to placing them in the VALx registers. Either the VAL2/VAL3 or the VAL4/VAL5 register pair controls the pulse width at any given time. For a given PWM pair, whether the VAL2/VAL3 or VAL4/VAL5 pair is active depends on either:

- The state of the current status pin, PWMX, for that driver
- The state of the odd/even correction bit, MCTRL[IPOL], for that driver

To correct deadtime distortion, software can decrease or increase the value in the appropriate VALx register.

- In edge-aligned operation, decreasing or increasing the PWM value by a correction value equal to the deadtime typically compensates for deadtime distortion.
- In center-aligned operation, decreasing or increasing the PWM value by a correction value equal to one-half the deadtime typically compensates for deadtime distortion.

37.5.2.8.2 Manual Correction

To detect the current status, the voltage on each PWMX pin is sampled twice in a PWM period, at the end of each deadtime. The value is stored in CTRL[DT]. CTRL[DT] is a timing marker especially indicating when to toggle between PWM value registers. Software can then set MCTRL[IPOL] to switch between VAL2/VAL3 and VAL4/VAL5 register pairs according to CTRL[DT] values.

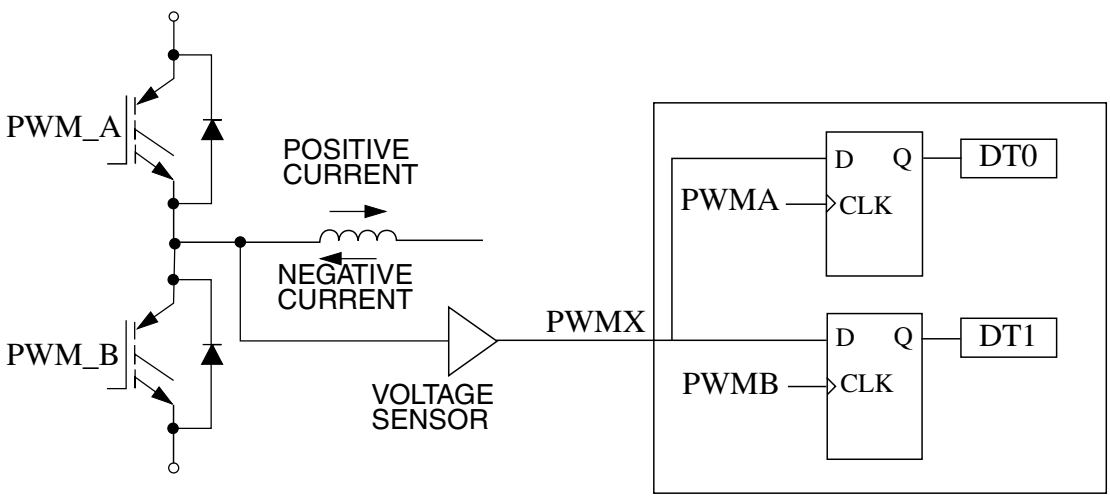


Figure 37-22. Current-status Sense Scheme for Deadtime Correction

Both D flip-flops latch low, CTRL[DT] = 00, during deadtime periods if current is large and flowing out of the complementary circuit. See the preceding figure. Both D flip-flops latch the high, CTRL[DT] = 11, during deadtime periods if current is also large and flowing into the complementary circuit.

However, under low-current, the output voltage of the complementary circuit during deadtime is somewhere between the high and low levels. The current cannot free-wheel through the opposition anti-body diode, regardless of polarity, giving additional distortion when the current crosses zero. **Sampled results will be CTRL[DT] = b10. Thus, the best time to change one PWM value register to another is just before the current zero crossing.**

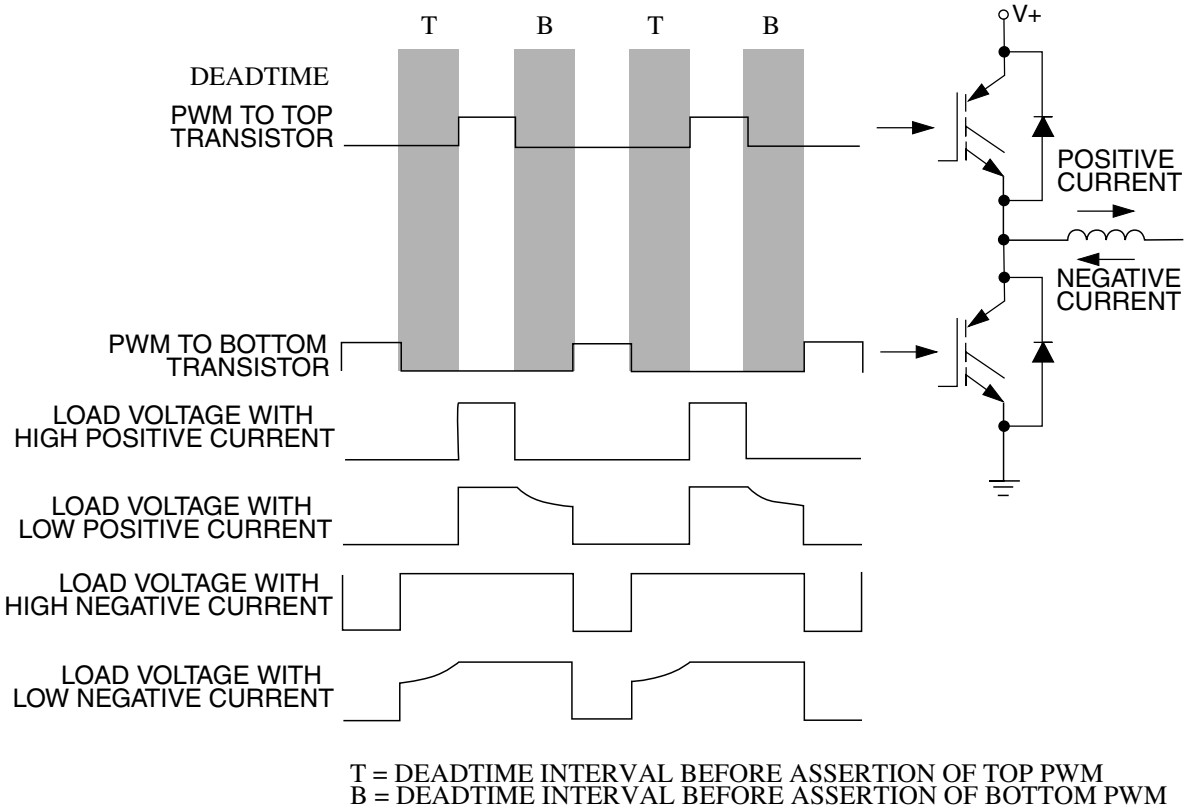


Figure 37-23. Output Voltage Waveforms

37.5.2.9 Fractional Delay Logic

For applications where more resolution than a single IPBus clock period is needed, the fractional delay logic can be used to achieve fine resolution on the rising and falling edges of the PWM_A and PWM_B outputs and fine resolution for the PWM period. Enable the use of the fractional delay logic by setting FRCTRL[FRACx_EN]. The FRACVALx registers act as a fractional clock cycle addition to the turn on and turn off count specified by the VAL2, VAL3, VAL4, or VAL5 registers. The FRACVAL1 register acts as a fractional increase in the PWM period as defined by VAL1. If FRACVAL1 is programmed to a non-zero value, then the largest value for the VAL1 register is 0xFFFE for unsigned usage or 0x7FFE for signed usage. This limit is needed in order to avoid counter rollovers when accumulating the fractional additional period.

The results of the fractional delay logic depend on whether or not the PWM submodule has an analog NanoEdge placer block available.

37.5.2.9.1 Fractional Delay Logic with NanoEdge Placement Block

Using the NanoEdge placer block requires that the IPBus clock to the PWM be set at a defined frequency. The NanoEdge placer is powered up by setting FRCTRL[FRAC_PU]. Enable fine edge control on the various PWM edges by setting FRCTRL[FRACx_EN]. The fractional values in the FRACVALx registers allow placing the PWM edge or PWM period to a granularity of 1/32 of the IPBus clock period. For example, if you desire the rising edge of the PWMA output to occur at a count of 12.25, then program VAL2 with 0x000C and FRACVAL2 with 0x4000. Using FRACVAL1 will adjust the PWM period with the same granularity of 1/32 of a clock period.

If the FRCTRL[FRAC_PU] bits in all of the submodules are clear, then the NanoEdge placer is powered down, and alternate clock frequencies can be used without the NanoEdge placement feature.

37.5.2.9.2 Fractional Delay Logic without NanoEdge Placement Block

For submodules that are not supported by the NanoEdge placer, the PWM can use dithering to simulate fine edge control. Enable this feature by setting the FRCTRL[FRAC1_EN], FRCTRL[FRAC23_EN], and FRCTRL[FRAC45_EN] bits. It is unnecessary to set FRCTRL[FRAC_PU]. The PWM period or the PWM edges will dither from the nearest whole number values to achieve an average value that is equivalent to the programmed fractional value. The added cycles are based on the accumulation of the fractional component. For example, if you want the PWM period to be 50.25 clock cycles, then program VAL1 with 0x0032 and FRACVAL1 with 0x4000. The PWM period will be 50 cycles long most of the time, but will occasionally be 51 cycles long to achieve a long-term average of 50.25 cycles.

In submodules that are not supported by a NanoEdge placer, the clock frequency is not required to be any specific value to achieve proper operation.

37.5.2.10 Output Logic

The following figure shows the output logic of each submodule including how each PWM output has individual fault disabling, polarity control, and output enable. This allows for maximum flexibility when interfacing to the external circuitry.

The PWM23 and PWM45 signals which are output from the deadtime logic (refer to the figure) are positive true signals. In other words, a high level on these signals should result in the corresponding transistor in the PWM inverter being turned ON. The voltage level required at the PWM output pin to turn the transistor ON or OFF is a function of the logic between the pin and the transistor. Therefore, it is imperative that the user program

OCTRL[POLA] and OCTRL[POLB] before enabling the output pins. A fault condition can result in the PWM output being tristated, forced to a logic 1, or forced to a logic 0 depending on the values programmed into the OCTRL[PWMxFS] fields.

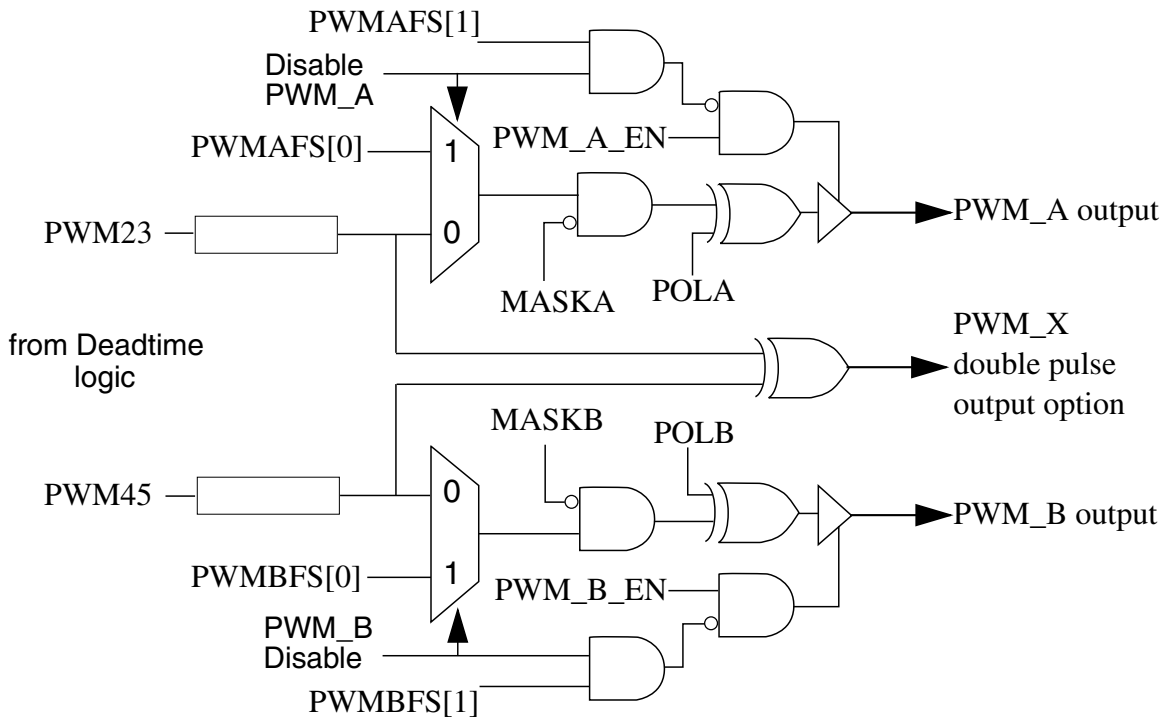


Figure 37-24. Output Logic

37.5.2.11 E-Capture

Commensurate with the idea of controlling both edges of an output signal, the Enhanced Capture (E-Capture) logic is designed to measure both edges of an input signal. As a result, when a submodule pin is configured for input capture, the CVALx registers associated with that pin are used to record the edge values.

The following figure is a block diagram of the E-Capture circuit. Upon entering the pin input, the signal is split into two paths. One goes straight to a mux input where software can select to pass the signal directly to the capture logic for processing. The other path connects the signal to an 8 bit counter which counts both the rising and falling edges of the signal. The output of this counter is compared to an 8 bit value that is specified by the user (EDGCMPx) and when the two values are equal, the comparator generates a pulse that resets the counter. This pulse is also supplied to the mux input where software can select it to be processed by the capture logic. This feature permits the E-Capture circuit to count up to 256 edge events before initiating a capture event. this feature is useful for

dividing down high frequency signals for capture processing so that capture interrupts don't overwhelm the CPU. Also, this feature can be used to simply generate an interrupt after "n" events have been counted.

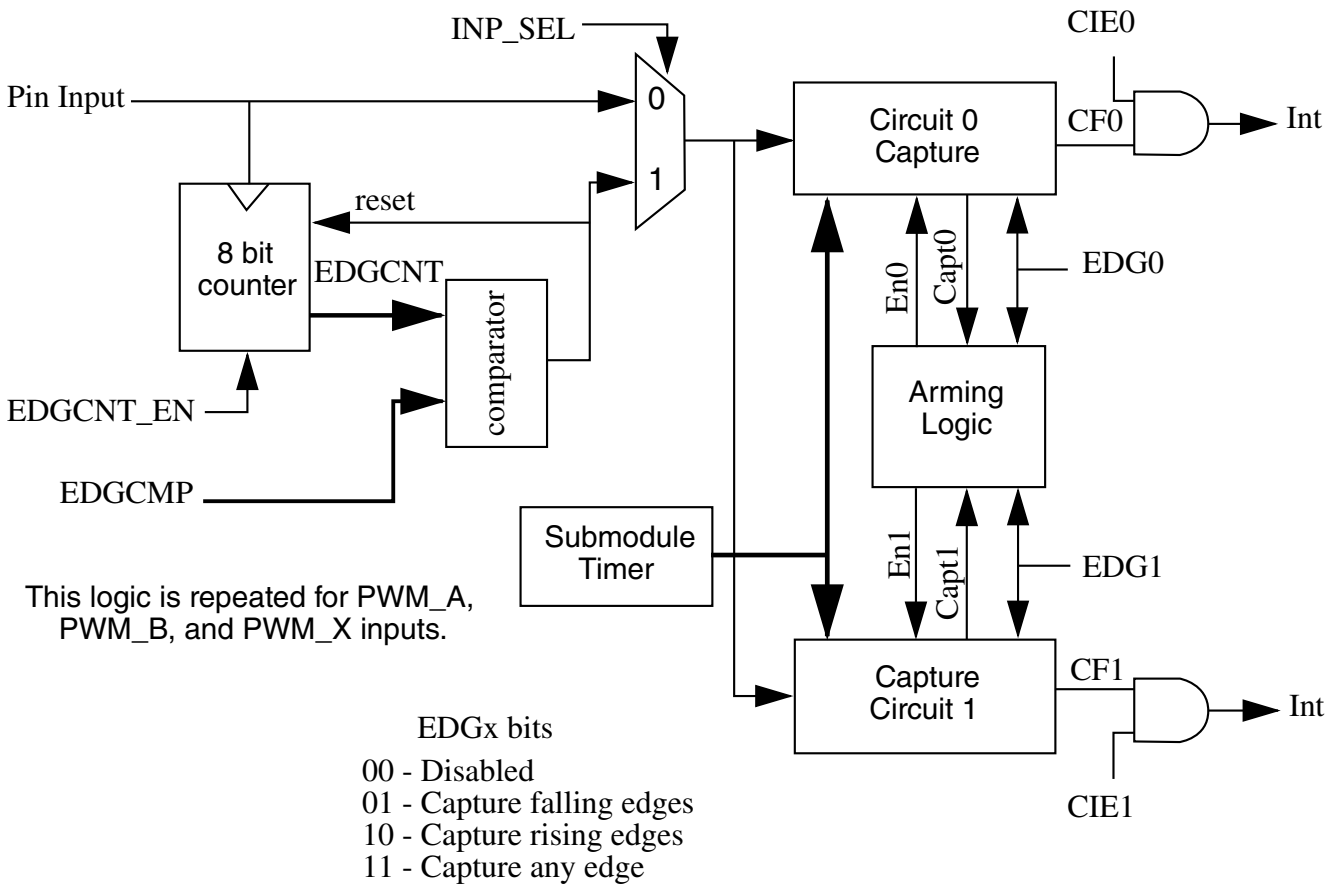


Figure 37-25. Enhanced Capture (E-Capture) Logic

Based on the mode selection, the mux selects either the pin input or the compare output from the count/compare circuit to be processed by the capture logic. The selected signal is routed to two separate capture circuits which work in tandem to capture sequential edges of the signal. The type of edge to be captured by each circuit is determined by CAPTCTRLx[EDGx1] and CAPTCTRLx[EDGx0], whose functionality is listed in the preceding figure. Also, controlling the operation of the capture circuits is the arming logic which allows captures to be performed in a free running (continuous) or one shot fashion. In free running mode, the capture sequences will be performed indefinitely. If both capture circuits are enabled, they will work together in a ping-pong style where a capture event from one circuit leads to the arming of the other and vice versa. In one shot mode, only one capture sequence will be performed. If both capture circuits are enabled, capture circuit 0 is first armed and when a capture event occurs, capture circuit 1 is armed. Once the second capture occurs, further captures are disabled until another capture sequence is initiated. Both capture circuits are also capable of generating an interrupt to the CPU.

37.5.2.12 Fault Protection

Fault protection can control any combination of PWM output pins. Faults are generated by a logic zero or one on any of the FAULTx pins. This polarity can be changed via FCTRL[FLVL]. Each FAULTx pin can be mapped arbitrarily to any of the PWM outputs. When fault protection hardware disables PWM outputs, the PWM generator continues to run, only the output pins are forced to logic 0, logic 1, or tristated depending the values of OCTRL[PWMxFS].

The fault decoder disables PWM pins selected by the fault logic and the disable mapping (DISMAPn) registers. The following figure shows an example of the fault disable logic, where a logic 1 indicates a fault condition. Each bank of bits in DISMAPn control the mapping for a single PWM pin. See the following table.

The fault protection is enabled even when the PWM module is not enabled; therefore, a fault will be latched in and must be cleared in order to prevent an interrupt when the PWM is enabled.

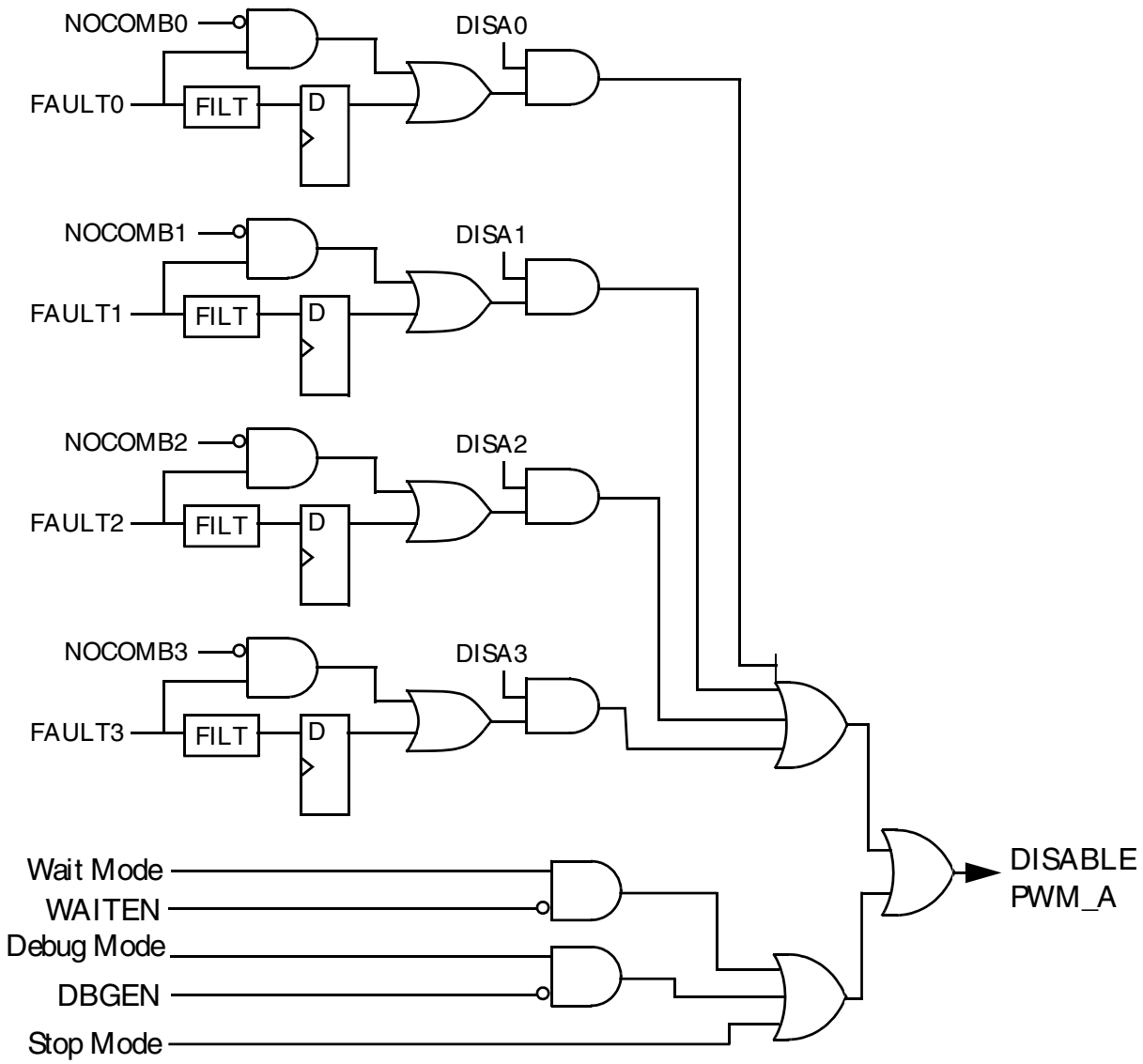


Figure 37-26. Fault Decoder for PWM_A

Table 37-2. Fault Mapping

PWM Pin	Controlling Register Bits
PWM_A	DISMAP0[DIS0A]
PWM_B	DISMAP0[DIS0B]
PWM_X	DISMAP0[DIS0X]

37.5.2.12.1 Fault Pin Filter

Each fault pin has a programmable filter that can be bypassed. The sampling period of the filter can be adjusted with FFILT[FILT_PER]. The number of consecutive samples that must agree before an input transition is recognized can be adjusted using FFILT[FILT_CNT]. Setting FFILT[FILT_PER] to all 0 disables the input filter for a given FAULTx pin.

Upon detecting an active level on the filtered FAULTx pin, the corresponding FSTS[FFPINx] and fault flag (FSTS[FFLAGx]) bits are set. FSTS[FFPINx] remains set as long as the filtered FAULTx pin is at its active level. Clear FSTS[FFLAGx] by writing a logic 1 to FSTS[FFLAGx].

If the FIEx, FAULTx pin interrupt enable bit is set, FSTS[FFLAGx] generates a CPU interrupt request. The interrupt request latch remains set until:

- Software clears FSTS[FFLAGx] by writing a logic one to the bit
- Software clears the FIEx bit by writing a logic zero to it
- A reset occurs

Even with the filter enabled, there is a combinational path from the FAULTx inputs to the PWM pins. This logic is also capable of holding a fault condition in the event of loss of clock to the PWM module.

37.5.2.12.2 Automatic Fault Clearing

Setting an automatic clearing mode bit, FCTRL[FAUTOx], configures faults from the FAULTx pin for automatic clearing.

When FCTRL[FAUTOx] is set, disabled PWM pins are enabled when the filtered FAULTx pin returns to its inactive level and a new PWM full or half cycle begins. See the following figure. If FSTS[FFULLx] is set, then the disabled PWM pins are enabled at the start of a full cycle. If FSTS[FHALFx] is set, then the disabled PWM pins are enabled at the start of a half cycle. Clearing FSTS[FFLAGx] does not affect disabled PWM pins when FCTRL[FAUTOx] is set.

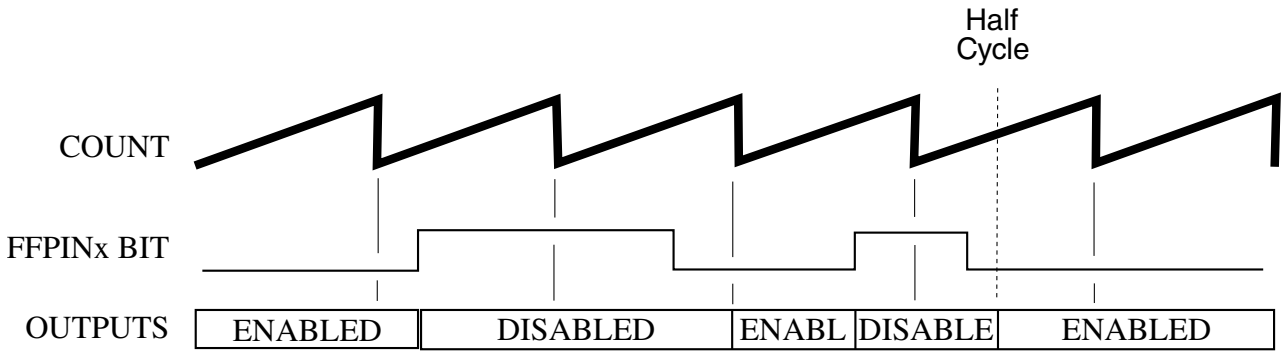


Figure 37-27. Automatic Fault Clearing

37.5.2.12.3 Manual Fault Clearing

Clearing the automatic clearing mode bit, FCTRL[FAUTOx], configures faults from the FAULTx pin for manual clearing:

- If the fault safety mode bits, FCTRL[FSAFEx], are clear (normal mode), then the series of events required to clear a fault condition are:
  - A FAULTx input goes active and the PWM outputs sensitive to this fault are combinationally disabled immediately while the fault is being filtered.
  - If the fault condition persists long enough to get through the fault filter, the FSTS[FFLAG] bit is set and an internal register is set to hold the PWM output disabled.
  - Software must clear the FSTS[FFLAG] bit since in manual mode.
  - A half cycle or full cycle boundary occurs matching the setting of FSTS[FHALF] and FSTS[FFULL]. This will clear the internal register that holds the PWM output disabled.
  - If the FAULTx input is still asserted, then the output will be combinationally disabled until the filtered version of the FAULTx input is inactive as shown in the 1st figure below. If the filtered version of the FAULTx input is already de-asserted as shown in the 2nd figure below, then this step is skipped.
  - The PWM output is re-enabled.
- If the fault safety mode bits, FCTRL[FSAFEx], are set (safe mode), then the series of events required to clear a fault condition are:
  - A FAULTx input goes active and the PWM outputs sensitive to this fault are combinationally disabled immediately while the fault is being filtered.

- If the fault condition persists long enough to get through the fault filter, the FSTS[FFLAG] bit is set and an internal register is set to hold the PWM output disabled.
- Software must clear the FSTS[FFLAG] bit since in manual mode.
- The filtered version of the FAULTx input goes inactive.
- A half cycle or full cycle boundary occurs matching the setting of FSTS[FHALF] and FSTS[FFULL]. This will clear the internal register that holds the PWM output disabled.
- The PWM output is re-enabled.

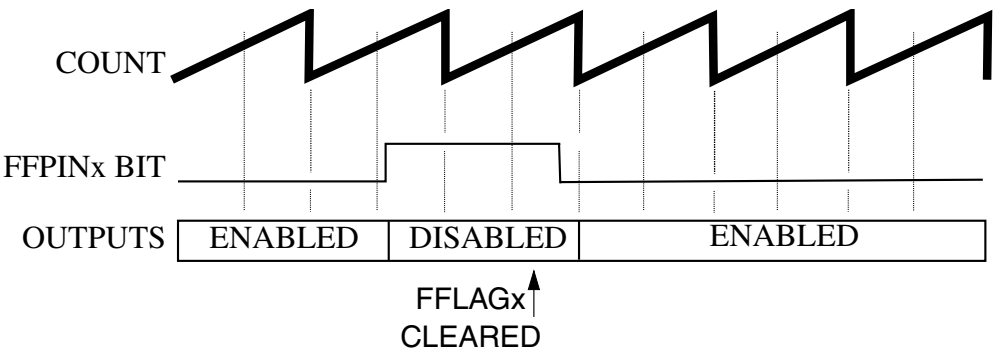


Figure 37-28. Manual Fault Clearing (FCTRL[FSAFE]=0), fault clears before reload boundary

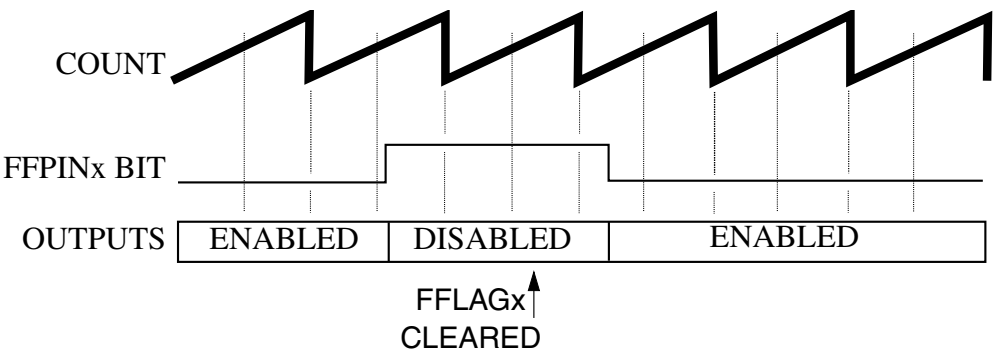


Figure 37-29. Manual Fault Clearing (FCTRL[FSAFE]=0), fault clears after reload boundary

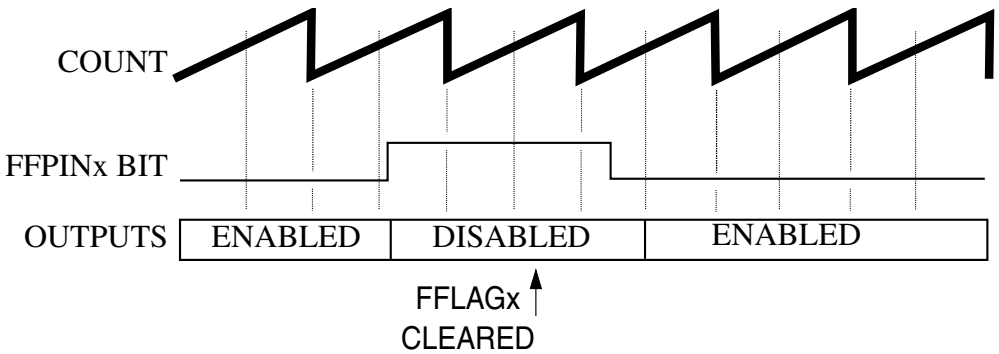


Figure 37-30. Manual Fault Clearing (FCTRL[FSAFE]=1)

Note

Fault protection also applies during software output control when the SEL23 and SEL45 fields are set to select OUT23 and OUT45 bits or PWM_EXTA and PWM_EXTB. Fault clearing still occurs at half PWM cycle boundaries while the PWM generator is engaged, MCTRL[RUN] equals one. But the OUTx bits can control the PWM pins while the PWM generator is off, MCTRL[RUN] equals zero. Thus, fault clearing occurs at IPBus cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

37.5.2.12.4 Fault Testing

FTST[FTEST] is used to simulate a fault condition on each of the fault inputs within that fault channel.

37.5.3 PWM Generator Loading

37.5.3.1 Load Enable

MCTRL[LDOK] enables loading of the following PWM generator parameters:

- The prescaler divisor—from CTRL[PRSC]
- The PWM period and pulse width—from the INIT and VALx registers

MCTRL[LDOK] allows software to finish calculating all of these PWM parameters so they can be synchronously updated. The CTRL[PRSC], INIT, and VALx registers are loaded by software into a set of outer buffers. When MCTRL[LDOK] is set, these values are transferred to an inner set of registers at the beginning of the next PWM reload cycle

to be used by the PWM generator. These values can be transfered to the inner set of registers immediately upon setting MCTRL[LDOK] if CTRL[LDMOD] is set. Set MCTRL[LDOK] by reading it when it is a logic zero and then writing a logic one to it. After loading, MCTRL[LDOK] is automatically cleared.

37.5.3.2 Load Frequency

CTRL[LDFQ] selects an integral loading frequency of one to 16 PWM reload opportunities. CTRL[LDFQ] takes effect at every PWM reload opportunity, regardless the state of MCTRL[LDOK]. CTRL[HALF] and CTRL[FULL] control reload timing. If CTRL[FULL] is set, a reload opportunity occurs at the end of every PWM cycle when the count equals VAL1. If CTRL[HALF] is set, a reload opportunity occurs at the half cycle when the count equals VAL0. If both CTRL[HALF] and CTRL[FULL] are set, a reload opportunity occurs twice per PWM cycle when the count equals VAL1 and when it equals VAL0.

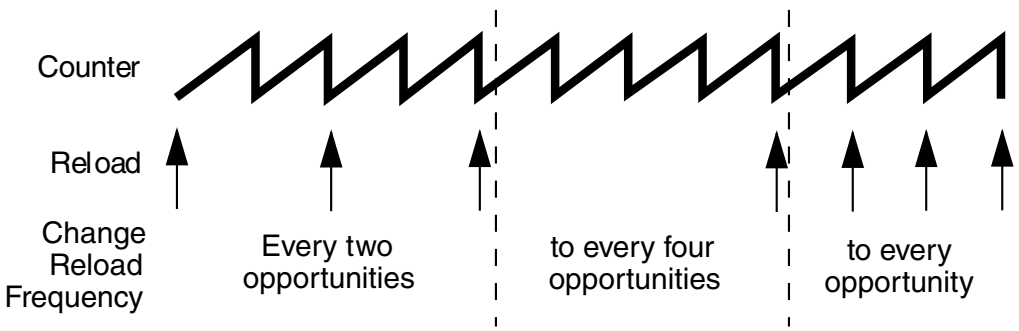


Figure 37-31. Full Cycle Reload Frequency Change

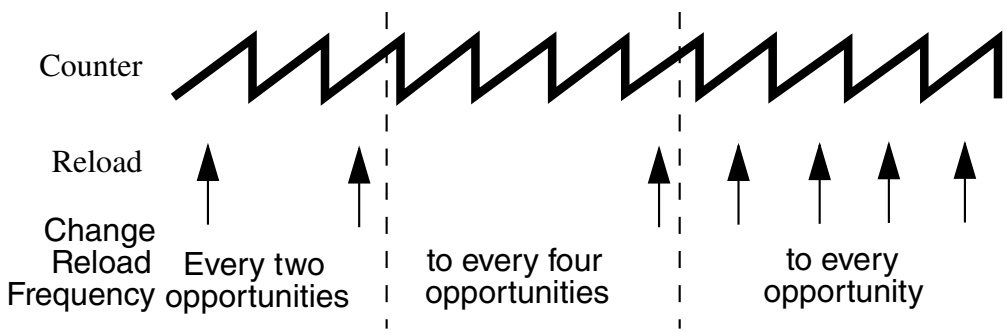


Figure 37-32. Half Cycle Reload Frequency Change

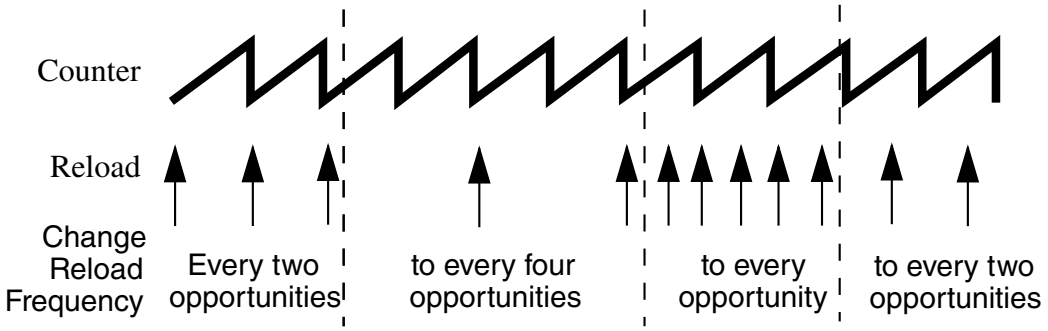


Figure 37-33. Full and Half Cycle Reload Frequency Change

37.5.3.3 Reload Flag

At every reload opportunity the PWM Reload Flag (STS[RF]) is set. Setting STS[RF] happens even if an actual reload is prevented by MCTRL[LDOK]. If the PWM reload interrupt enable bit, INTEN[RIE], is set, the STS[RF] flag generates CPU interrupt requests allowing software to calculate new PWM parameters in real time. When INTEN[RIE] is not set, reloads still occur at the selected reload rate without generating CPU interrupt requests.

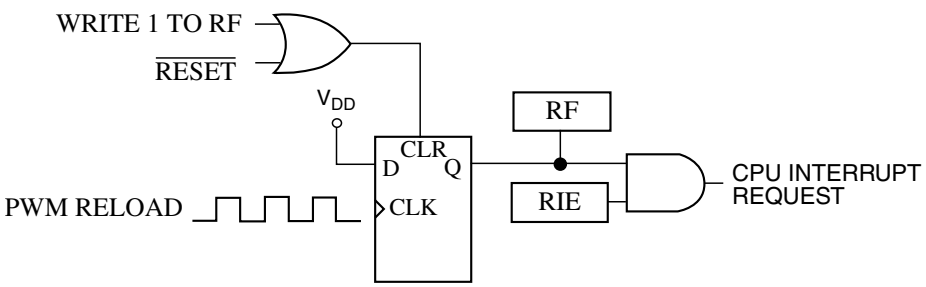


Figure 37-34. PWMF Reload Interrupt Request

37.5.3.4 Reload Errors

Whenever one of the VALx, FRACVALx, or CTRL[PRSC] registers is updated, the STS[RUF] flag is set to indicate that the data is not coherent. STS[RUF] will be cleared by a successful reload which consists of the reload signal while MCTRL[LDOK] is set. If STS[RUF] is set and MCTRL[LDOK] is clear when the reload signal occurs, a reload error has taken place and STS[REF] is set. If STS[RUF] is clear when a reload signal asserts, then the data is coherent and no error will be flagged.

37.5.3.5 Initialization

Initialize all registers and set MCTRL[LDOK] before setting MCTRL[RUN].

Note

Even if MCTRL[LDOK] is not set, setting MCTRL[RUN] also sets the STS[RF] flag. To prevent a CPU interrupt request, clear INTEN[RIE] before setting MCTRL[RUN].

The PWM generator uses the last values loaded if MCTRL[RUN] is cleared and then set while MCTRL[LDOK] equals zero.

When MCTRL[RUN] is cleared:

- The STS[RF] flag and pending CPU interrupt requests are not cleared
- All fault circuitry remains active
- Software/external output control remains active
- Deadtime insertion continues during software/external output control

37.6 Resets

All PWM registers are reset to their default values upon any system reset.

The reset forces all registers to their reset states and tri-states the PWM outputs.

37.7 Interrupts

Each of the submodules within the PWM module can generate an interrupt from several sources. The fault logic can also generate interrupts. The interrupt service routine (ISR) must check the related interrupt enables and interrupt flags to determine the actual cause of the interrupt.

Table 37-3. Interrupt Summary

Core Interrupt	Interrupt Flag	Interrupt Enable	Name	Description
PWM_CMP0	SM0STS[CMPF]	SM0INTEN[CMPIE]	Submodule 0 compare interrupt	Compare event has occurred
PWM_CAP0	SM0STS[CFA1], SM0STS[CFA0], SM0STS[CFB1],	SM0INTEN[CFA1IE], SM0INTEN[CFA0IE], SM0INTEN[CFB1IE],	Submodule 0 input capture interrupt	Input capture event has occurred

Table continues on the next page...

Table 37-3. Interrupt Summary (continued)

Core Interrupt	Interrupt Flag	Interrupt Enable	Name	Description
	SM0STS[CFB0], SM0STS[CFX1], SM0STS[CFX0]	SM0INTEN[CFB0IE], SM0INTEN[CFX1IE], SM0INTEN[CFX0IE]		
PWM_RELOAD0	SM0STS[RF]	SM0INTEN[RIE]	Submodule 0 reload interrupt	Reload event has occurred
PWM_CMP1	SM1STS[CMPF]	SM1INTEN[CMPIE]	Submodule 1 compare interrupt	Compare event has occurred
PWM_CAP1	SM1STS[CFA1], SM1STS[CFA0], SM1STS[CFB1], SM1STS[CFB0], SM1STS[CFX1], SM1STS[CFX0]	SM1INTEN[CFA1IE], SM1INTEN[CFA0IE], SM1INTEN[CFB1IE], SM1INTEN[CFB0IE], SM1INTEN[CFX1IE], SM1INTEN[CFX0IE]	Submodule 1 input capture interrupt	Input capture event has occurred
PWM_RELOAD1	SM1STS[RF]	SM1INTEN[RIE]	Submodule 1 reload interrupt	Reload event has occurred
PWM_CMP2	SM2STS[CMPF]	SM2INTEN[CMPIE]	Submodule 2 compare interrupt	Compare event has occurred
PWM_CAP2	SM2STS[CFA1], SM2STS[CFA0], SM2STS[CFB1], SM2STS[CFB0], SM2STS[CFX1], SM2STS[CFX0]	SM2INTEN[CFA1IE], SM2INTEN[CFA0IE], SM2INTEN[CFB1IE], SM2INTEN[CFB0IE], SM2INTEN[CFX1IE], SM2INTEN[CFX0IE	Submodule 2 input capture interrupt	Input capture event has occurred
PWM_RELOAD2	SM2STS[RF]	SM2INTEN[RIE]	Submodule 2 reload interrupt	Reload event has occurred
PWM_CMP3	SM3STS[CMPF]	SM3INTEN[CMPIE]	Submodule 3 compare interrupt	Compare event has occurred
PWM_CAP3	SM3STS[CFA1], SM3STS[CFA0], SM3STS[CFB1], SM3STS[CFB0], SM3STS[CFX1], SM3STS[CFX0]	SM3INTEN[CFA1IE], SM3INTEN[CFA0IE], SM3INTEN[CFB1IE], SM3INTEN[CFB0IE], SM3INTEN[CFX1IE], SM3INTEN[CFX0IE	Submodule 3 input capture interrupt	Input capture event has occurred
PWM_RELOAD3	SM3STS[RF]	SM3INTEN[RIE]	Submodule 3 reload interrupt	Reload event has occurred
PWM_RERR	SM0STS[REF]	SM0INTEN[REIE]	Submodule 0 reload error interrupt	Reload error has occurred
	SM1STS[REF]	SM1INTEN[REIE]	Submodule 1 reload error interrupt	
	SM2STS[REF]	SM2INTEN[REIE]	Submodule 2 reload error interrupt	

Table continues on the next page...

Table 37-3. Interrupt Summary (continued)

Core Interrupt	Interrupt Flag	Interrupt Enable	Name	Description
	SM3STS[REF]	SM3INTEN[REIE]	Submodule 3 reload error interrupt	
PWM_FAULT	FSTS[FFLAG]	FCTRL[FIE]	Fault input interrupt	Fault condition has been detected

37.8 DMA

Each submodule can request a DMA read access for its capture FIFOs and a DMA write request for its double buffered VALx registers.

Table 37-4. DMA Summary

DMA Request	DMA Enable	Name	Description
Submodule 0 read request	SM0DMAEN[CX0DE]	SM0 Capture FIFO X0 read request	SM0CVAL0 contains a value to be read
	SM0DMAEN[CX1DE]	SM0 Capture FIFO X1 read request	SM0CVAL1 contains a value to be read
	SM0DMAEN[CA0DE]	SM0 Capture FIFO A0 read request	SM0CVAL2 contains a value to be read
	SM0DMAEN[CA1DE]	SM0 Capture FIFO A1 read request	SM0CVAL3 contains a value to be read
	SM0DMAEN[CB0DE]	SM0 Capture FIFO B0 read request	SM0CVAL4 contains a value to be read
	SM0DMAEN[CB1DE]	SM0 Capture FIFO B1 read request	SM0CVAL5 contains a value to be read
	SM0DMAEN[CAPTDE]	SM0 Capture FIFO read request source select	Selects source of submodule0 read DMA request
Submodule 0 write request	SM0DMAEN[VALDE]	SM0VALx write request	SM0VALx registers need to be updated
Submodule 1 read request	SM1DMAEN[CX0DE]	SM1 Capture FIFO X0 read request	SM1CVAL0 contains a value to be read
	SM1DMAEN[CX1DE]	SM1 Capture FIFO X! read request	SM1CVAL1 contains a value to be read
	SM1DMAEN[CA0DE]	SM1Capture FIFO A0 read request	SM1CVAL2 contains a value to be read
	SM1DMAEN[CA1DE]	SM1 Capture FIFO A1 read request	SM1CVAL3 contains a value to be read
	SM1DMAEN[CB0DE]	SM1 Capture FIFO B0 read request	SM1CVAL4 contains a value to be read
	SM1DMAEN[CB1DE]	SM1 Capture FIFO B1 read request	SM!CVAL5 contains a value to be read

Table continues on the next page...

Table 37-4. DMA Summary (continued)

DMA Request	DMA Enable	Name	Description
	SM1DMAEN[CAPTDE]	SM1 Capture FIFO read request source select	Selects source of submodule1 read DMA request
Submodule 1 write request	SM1DMAEN[VALDE]	SM1VALx write request	SM1VALx registers need to be updated
Submodule 2 read request	SM2DMAEN[CX0DE]	SM2 Capture FIFO X0 read request	SM2CVAL0 contains a value to be read
	SM2DMAEN[CX1DE]	SM2 Capture FIFO X! read request	SM2CVAL1 contains a value to be read
	SM2DMAEN[CA0DE]	SM2 Capture FIFO A0 read request	SM2CVAL2 contains a value to be read
	SM2DMAEN[CA1DE]	SM2 Capture FIFO A1 read request	SM2CVAL3 contains a value to be read
	SM2DMAEN[CB0DE]	SM2 Capture FIFO B0 read request	SM2CVAL4 contains a value to be read
	SM2DMAEN[CB1DE]	SM2 Capture FIFO B1 read request	SM2CVAL5 contains a value to be read
	SM2DMAEN[CAPTDE]	SM2 Capture FIFO read request source select	Selects source of submodule2 read DMA request
Submodule 2 write request	SM2DMAEN[VALDE]	SM2VALx write request	SM2VALx registers need to be updated
Submodule 3 read request	SM3DMAEN[CX0DE]	SM3 Capture FIFO X0 read request	SM3CVAL0 contains a value to be read
	SM3DMAEN[CX1DE]	SM3 Capture FIFO X! read request	SM3CVAL1 contains a value to be read
	SM3DMAEN[CA0DE]	SM3 Capture FIFO A0 read request	SM3CVAL2 contains a value to be read
	SM3DMAEN[CA1DE]	SM3 Capture FIFO A1 read request	SM3CVAL3 contains a value to be read
	SM3DMAEN[CB0DE]	SM3 Capture FIFO B0 read request	SM3CVAL4 contains a value to be read
	SM3DMAEN[CB1DE]	SM3 Capture FIFO B1 read request	SM3CVAL5 contains a value to be read
	SM3DMAEN[CAPTDE]	SM3 Capture FIFO read request source select	Selects source of submodule3 read DMA request
Submodule 3 write request	SM3DMAEN[VALDE]	SM3VALx write request	SM3VALx registers need to be updated

Chapter 38
Programmable Delay Block (PDB)

38.1 Chip-specific PDB information

38.1.1 PDB Instantiation

This chip has two PDBs that primarily provide delayed triggering from the FTMs to the ADCs. Each PDB has one trigger output with four pre-trigger channels, four pulse output and one DAC trigger. The input mux capability has been increased by having an XBARA output trigger the PDBs.

38.1.1.1 PDB0 Output Triggers

Table 38-1. PDB0 output triggers

Number of PDB channels	1
Number of pre-triggers per PDB channel	4
PDB_ch0_out	ADCA sync0 , DMA_MUX source 48, FTM0_TRIG1, XBARA_IN29, XBARB_IN12
DAC trigger	DAC0_trigger
PulseOut	Window control of CMP0, CMP1,CMP2,CMP3

38.1.1.2 PDB0 Input Trigger Connections

Table 38-2. PDB0 Input Trigger Options

PDB Trigger	PDB Input
0000	External Trigger (PDB0_EXTRG)

Table continues on the next page...

Table 38-2. PDB0 Input Trigger Options (continued)

PDB Trigger	PDB Input
0001	CMP0_out
0010	CMP1_out
0011	CMP2_out
0100	PIT Ch 0 Output
0101	PIT Ch 1 Output
0110	PIT Ch 2 Output
0111	PIT Ch 3 Output
1000	FTM0 initialization trigger and channel triggers, as programmed in the FTM external trigger register (EXTTRIG)
1001	FTM1 initialization trigger and channel triggers, as programmed in the FTM external trigger register (EXTTRIG)
1010	Reserved
1011	FTM3 initialization trigger and channel triggers, as programmed in the FTM external trigger register (EXTTRIG)
1100	XBAR_OUT 38
1101	Reserved
1110	LPTMR Output
1111	Software Trigger

38.1.1.3 PDB1 Output Triggers

Table 38-3. PDB1 output triggers

Number of PDB channels	1
Number of pre-triggers per PDB channel	4
PDB_ch0_out	ADCB sync1, DMA_MUX source 47, FTM3_TRIG1, XBARIN31, XBARB_IN26
DAC trigger	DAC0_trigger
PulseOut	Window control of CMP0, CMP1,CMP2,CMP3

38.1.1.4 PDB1 Input Trigger Connections

Table 38-4. PDB1 Input Trigger Options

PDB Trigger	PDB Input
0000	Reserved
0001	CMP0_out

Table continues on the next page...

Table 38-4. PDB1 Input Trigger Options (continued)

PDB Trigger	PDB Input
0010	CMP1_out
0011	CMP2_out
0100	PIT Ch 0 Output
0101	PIT Ch 1 Output
0110	PIT Ch 2 Output
0111	PIT Ch 3 Output
1000	FTM0 initialization trigger and channel triggers, as programmed in the FTM external trigger register (EXTTRIG)
1001	FTM1 initialization trigger and channel triggers, as programmed in the FTM external trigger register (EXTTRIG)
1010	Reserved
1011	FTM3 initialization trigger and channel triggers, as programmed in the FTM external trigger register (EXTTRIG)
1100	XBAR_OUT 41
1101	Reserved
1110	LPTMR Output
1111	Software Trigger

38.1.2 PDB's DAC External Trigger Input Connections

In this MCU, the following PDB's DAC external trigger inputs are implemented.

- PDB0's DAC external trigger input 0: ADCA_scan_complete
- PDB1's DAC external trigger input 0: ADCB_scan_complete

38.1.3 Pulse-Out Connection

Individual PDB Pulse-Out signals are connected to each CMP block and used for sample window.

38.1.4 Pulse-Out Enable Register Implementation

The following table shows the comparison of pulse-out enable register at the module and chip level.

Table 38-5. PDB pulse-out enable register

Register	Module implementation	Chip implementation
POnEN	7:0 - POEN 31:8 - Reserved	0 - POEN[0] for CMP0 1 - POEN[1] for CMP1 2 - POEN[2] for CMP2 3 - POEN[3] for CMP3 31:4 - Reserved

38.1.5 Pulse-Out Enable Register Implementation

The PDB0 trigger is connected to ADC0 and PDB1 trigger is connected to ADC1.

It is not recommended to use PDB1 to trigger ADC0 or PDB0 to trigger ADC1. It is also not recommended to use XBARA to route PDB0 to ADC1 or PDB1 to ADC0.

Each of PDB has 4 pre-triggers which can be used to trigger ADC conversion 4 times in sequence.

User can use the back to back feature of this chip to configure the two PDBs as a signal chain. The ADC0 scan complete signal can be used as PDB1's Pre-trigger0 acknowledge and ADC1 scan complete signal can be used as PDB0's Pre-trigger0 acknowledge. See Figure 38-1

38.2 Introduction

The Programmable Delay Block (PDB) provides controllable delays from either an internal or an external trigger, or a programmable interval tick, to the hardware trigger inputs of ADCs and/or generates the interval triggers to DACs, so that the precise timing between ADC conversions and/or DAC updates can be achieved. The PDB can optionally provide pulse outputs (Pulse-Out's) that are used as the sample window in the CMP block.

38.2.1 Features

-
- Up to 8 configurable PDB channels for ADC hardware trigger
  - One PDB channel is associated with one ADC
  - One trigger output for ADC hardware trigger and up to 8 pre-trigger outputs for ADC trigger select per PDB channel
  - Trigger outputs can be enabled or disabled independently
  - One 16-bit delay register per pre-trigger output
  - Optional bypass of the delay registers of the pre-trigger outputs
  - Operation in One-Shot or Continuous modes
  - Optional back-to-back mode operation, which enables the ADC conversions complete to trigger the next PDB channel
  - One programmable delay interrupt
  - One sequence error interrupt
  - One channel flag and one sequence error flag per pre-trigger
  - DMA support
- Up to 8 DAC interval triggers
  - One interval trigger output per DAC
  - One 16-bit delay interval register per DAC trigger output
  - Optional bypass of the delay interval trigger registers
  - Optional external triggers
- Up to 8 pulse outputs (pulse-out's)
  - Pulse-out's can be enabled or disabled independently
  - Programmable pulse width

NOTE

The number of PDB input and output triggers are chip-specific. See the chip-specific PDB information for details.

38.2.2 Implementation

In this section, the following letters refer to the number of output triggers:

- N—Total available number of PDB channels.
- n—PDB channel number, valid from 0 to *N*-1.
- M—Total available pre-trigger per PDB channel.
- m—Pre-trigger number, valid from 0 to *M*-1.
- X—Total number of DAC interval triggers.
- x—DAC interval trigger output number, valid from 0 to *X*-1.
- Y—Total number of Pulse-Out's.
- y—Pulse-Out number, valid value is from 0 to *Y*-1.

NOTE

The number of module output triggers to core is chip-specific. For module to core output triggers implementation, see the chip configuration information.

38.2.3 Back-to-back acknowledgment connections

PDB back-to-back operation acknowledgment connections are chip-specific. For implementation, see the chip configuration information.

38.2.4 DAC External Trigger Input Connections

The implementation of DAC external trigger inputs is chip-specific. See the chip configuration information for details.

38.2.5 Block diagram

This diagram illustrates the major components of the PDB.

Figure 38-1. PDB block diagram

In this diagram, only one PDB channel *n*, one DAC interval trigger *x*, and one Pulse-Out *y* are shown. The PDB-enabled control logic and the sequence error interrupt logic are not shown.

38.2.6 Modes of operation

PDB ADC trigger operates in the following modes:

- Disabled—Counter is off, all pre-trigger and trigger outputs are low if PDB is not in back-to-back operation of Bypass mode.
- Debug—Counter is paused when processor is in Debug mode, and the counter for the DAC trigger is also paused in Debug mode.
- Enabled One-Shot—Counter is enabled and restarted at count zero upon receiving a positive edge on the selected trigger input source or software trigger is selected and SC[SWTRIG] is written with 1. In each PDB channel, an enabled pre-trigger asserts once per trigger input event. The trigger output asserts whenever any of the pre-triggers is asserted.
- Enabled Continuous—Counter is enabled and restarted at count zero. The counter is rolled over to zero again when the count reaches the value specified in the modulus register, and the counting is restarted. This enables a continuous stream of pre-triggers/trigger outputs as a result of a single trigger input event.
- Enabled Bypassed—The pre-trigger and trigger outputs assert immediately after a positive edge on the selected trigger input source or software trigger is selected and SC[SWTRIG] is written with 1, that is the delay registers are bypassed. It is possible to bypass any one or more of the delay registers; therefore, this mode can be used in conjunction with One-Shot or Continuous mode.

38.3 Memory map and register definition

PDB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_1000	Status and Control register (PDB1_SC)	32	R/W	0000_0000h	38.3.1/863
4003_1004	Modulus register (PDB1_MOD)	32	R/W	0000_FFFFh	38.3.2/866
4003_1008	Counter register (PDB1_CNT)	32	R	0000_0000h	38.3.3/866
4003_100C	Interrupt Delay register (PDB1_IDLY)	32	R/W	0000_FFFFh	38.3.4/867
4003_1010	Channel n Control register 1 (PDB1_CH0C1)	32	R/W	0000_0000h	38.3.5/867

Table continues on the next page...

PDB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_1014	Channel n Status register (PDB1_CH0S)	32	R/W	0000_0000h	38.3.6/868
4003_1018	Channel n Delay 0 register (PDB1_CH0DLY0)	32	R/W	0000_0000h	38.3.7/869
4003_101C	Channel n Delay 1 register (PDB1_CH0DLY1)	32	R/W	0000_0000h	38.3.8/870
4003_1020	Channel n Delay 2 register (PDB1_CH0DLY2)	32	R/W	0000_0000h	38.3.9/870
4003_1024	Channel n Delay 3 register (PDB1_CH0DLY3)	32	R/W	0000_0000h	38.3.10/ 871
4003_1150	DAC Interval Trigger n Control register (PDB1_DACINTC0)	32	R/W	0000_0000h	38.3.11/ 872
4003_1154	DAC Interval n register (PDB1_DACINT0)	32	R/W	0000_0000h	38.3.12/ 872
4003_1190	Pulse-Out n Enable register (PDB1_POEN)	32	R/W	0000_0000h	38.3.13/ 873
4003_1194	Pulse-Out n Delay register (PDB1_PO0DLY)	32	R/W	0000_0000h	38.3.14/ 873
4003_1198	Pulse-Out n Delay register (PDB1_PO1DLY)	32	R/W	0000_0000h	38.3.14/ 873
4003_119C	Pulse-Out n Delay register (PDB1_PO2DLY)	32	R/W	0000_0000h	38.3.14/ 873
4003_11A0	Pulse-Out n Delay register (PDB1_PO3DLY)	32	R/W	0000_0000h	38.3.14/ 873
4003_6000	Status and Control register (PDB0_SC)	32	R/W	0000_0000h	38.3.1/863
4003_6004	Modulus register (PDB0_MOD)	32	R/W	0000_FFFFh	38.3.2/866
4003_6008	Counter register (PDB0_CNT)	32	R	0000_0000h	38.3.3/866
4003_600C	Interrupt Delay register (PDB0_IDLY)	32	R/W	0000_FFFFh	38.3.4/867
4003_6010	Channel n Control register 1 (PDB0_CH0C1)	32	R/W	0000_0000h	38.3.5/867
4003_6014	Channel n Status register (PDB0_CH0S)	32	R/W	0000_0000h	38.3.6/868
4003_6018	Channel n Delay 0 register (PDB0_CH0DLY0)	32	R/W	0000_0000h	38.3.7/869
4003_601C	Channel n Delay 1 register (PDB0_CH0DLY1)	32	R/W	0000_0000h	38.3.8/870
4003_6020	Channel n Delay 2 register (PDB0_CH0DLY2)	32	R/W	0000_0000h	38.3.9/870
4003_6024	Channel n Delay 3 register (PDB0_CH0DLY3)	32	R/W	0000_0000h	38.3.10/ 871
4003_6150	DAC Interval Trigger n Control register (PDB0_DACINTC0)	32	R/W	0000_0000h	38.3.11/ 872
4003_6154	DAC Interval n register (PDB0_DACINT0)	32	R/W	0000_0000h	38.3.12/ 872
4003_6190	Pulse-Out n Enable register (PDB0_POEN)	32	R/W	0000_0000h	38.3.13/ 873
4003_6194	Pulse-Out n Delay register (PDB0_PO0DLY)	32	R/W	0000_0000h	38.3.14/ 873
4003_6198	Pulse-Out n Delay register (PDB0_PO1DLY)	32	R/W	0000_0000h	38.3.14/ 873

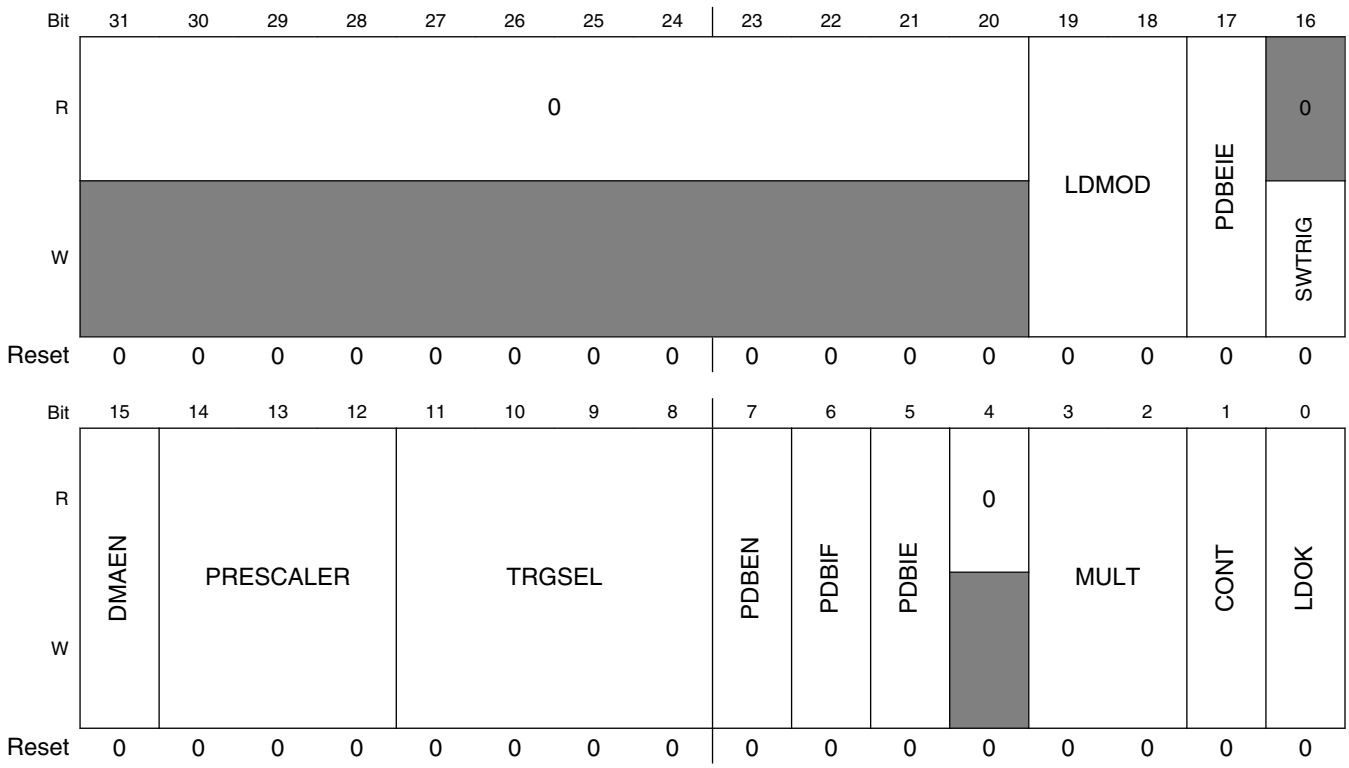
Table continues on the next page...

PDB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_619C	Pulse-Out n Delay register (PDB0_PO2DLY)	32	R/W	0000_0000h	38.3.14/ 873
4003_61A0	Pulse-Out n Delay register (PDB0_PO3DLY)	32	R/W	0000_0000h	38.3.14/ 873

38.3.1 Status and Control register (PDBx_SC)

Address: Base address + 0h offset



PDBx_SC field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–18 LDMOD	Load Mode Select Selects the mode to load the MOD, IDLY, CH*n*DLY*m*, INT*x*, and PO*y*DLY registers, after 1 is written to LDOK. 00 The internal registers are loaded with the values from their buffers, immediately after 1 is written to LDOK. 01 The internal registers are loaded with the values from their buffers when the PDB counter (CNT) = MOD + 1 CNT delay elapsed, after 1 is written to LDOK.

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
	10 The internal registers are loaded with the values from their buffers when a trigger input event is detected, after 1 is written to LDOK. 11 The internal registers are loaded with the values from their buffers when either the PDB counter (CNT) = MOD + 1 CNT delay elapsed, or a trigger input event is detected, after 1 is written to LDOK.
17 PDBEIE	PDB Sequence Error Interrupt Enable Enables the PDB sequence error interrupt. When PDBEIE is set, any of the PDB channel sequence error flags generates a PDB sequence error interrupt. 0 PDB sequence error interrupt disabled. 1 PDB sequence error interrupt enabled.
16 SWTRIG	Software Trigger When PDB is enabled and the software trigger is selected as the trigger input source, writing 1 to SWTRIG resets and restarts the counter. Writing 0 to SWTRIG has no effect. Reading SWTRIG yields 0.
15 DMAEN	DMA Enable When DMA is enabled, the PDBIF flag generates a DMA request instead of an interrupt. 0 DMA disabled. 1 DMA enabled.
14–12 PRESCALER	Prescaler Divider Select Counting uses the peripheral clock divided by the product of a factor (selected by MULT field) and an integer factor (set by PRESCALAR field), or in other words, (peripheral clock)/(MULT x PRESCALAR). 000 Counting uses the peripheral clock divided by MULT (the multiplication factor). 001 Counting uses the peripheral clock divided by 2 x MULT (the multiplication factor). 010 Counting uses the peripheral clock divided by 4 x MULT (the multiplication factor). 011 Counting uses the peripheral clock divided by 8 x MULT (the multiplication factor). 100 Counting uses the peripheral clock divided by 16 x MULT (the multiplication factor). 101 Counting uses the peripheral clock divided by 32 x MULT (the multiplication factor). 110 Counting uses the peripheral clock divided by 64 x MULT (the multiplication factor). 111 Counting uses the peripheral clock divided by 128 x MULT (the multiplication factor).
11–8 TRGSEL	Trigger Input Source Select Selects the trigger input source for the PDB. The trigger input source can be internal or external (EXTRG pin), or the software trigger. Refer to chip configuration details for the actual PDB input trigger connections. 0000 Trigger-In 0 is selected. 0001 Trigger-In 1 is selected. 0010 Trigger-In 2 is selected. 0011 Trigger-In 3 is selected. 0100 Trigger-In 4 is selected. 0101 Trigger-In 5 is selected. 0110 Trigger-In 6 is selected. 0111 Trigger-In 7 is selected. 1000 Trigger-In 8 is selected. 1001 Trigger-In 9 is selected.

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
	1010 Trigger-In 10 is selected. 1011 Trigger-In 11 is selected. 1100 Trigger-In 12 is selected. 1101 Trigger-In 13 is selected. 1110 Trigger-In 14 is selected. 1111 Software trigger is selected.
7 PDBEN	PDB Enable 0 PDB disabled. Counter is off. 1 PDB enabled.
6 PDBIF	PDB Interrupt Flag PDBIF is set when the counter value is equal to the IDLY register + 1. • Write zero to clear PDBIF. • Writing 1 to PDBIF has no effect.
5 PDBIE	PDB Interrupt Enable Enables the PDB interrupt. When PDBIE is set and DMAEN is cleared, PDBIF generates a PDB interrupt. 0 PDB interrupt disabled. 1 PDB interrupt enabled.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–2 MULT	Multiplication Factor Select for Prescaler Selects the multiplication factor of the prescaler divider for the counter clock. 00 Multiplication factor is 1. 01 Multiplication factor is 10. 10 Multiplication factor is 20. 11 Multiplication factor is 40.
1 CONT	Continuous Mode Enable Enables the PDB operation in Continuous mode. 0 PDB operation in One-Shot mode 1 PDB operation in Continuous mode
0 LDOK	Load OK Writing 1 to LDOK bit updates the MOD, IDLY, CHnDLYm, DACINTx, and POyDLY registers with the values previously written to their internal buffers (and stored there). The new values of MOD, IDLY, CHnDLYm, DACINTx, and POyDLY registers will take effect according to the setting of the LDMOD field (Load Mode Select). Before 1 is written to the LDOK field, the values in the internal buffers of these registers are not effective, and new values cannot be written to the internal buffers until the existing values in the internal buffers are loaded into their corresponding registers. • LDOK can be written only when PDBEN is set, or LDOK can be written at the same time when PDBEN is written to 1. • LDOK is automatically cleared when the values in the internal buffers are loaded into the registers or when PDBEN bit (PDB Enable) is cleared. • Writing 0 to LDOK has no effect.

38.3.2 Modulus register (PDBx_MOD)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 4h offset

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

PDBx_MOD field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MOD	PDB Modulus Specifies the period of the counter. When the counter reaches this value, it will be reset back to zero. If the PDB is in Continuous mode, the count begins anew. Reading this field returns the value of the internal register that is effective for the current cycle of PDB.

38.3.3 Counter register (PDBx_CNT)

Address: Base address + 8h offset

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

PDBx_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CNT	PDB Counter Contains the current value of the counter.

38.3.4 Interrupt Delay register (PDBx_IDLY)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + Ch offset

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

PDBx_IDLY field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
IDLY	PDB Interrupt Delay Specifies the delay value to schedule the PDB interrupt. It can be used to schedule an independent interrupt at some point in the PDB cycle. If enabled, a PDB interrupt is generated, when the counter is equal to the IDLY. Reading this field returns the value of internal register that is effective for the current cycle of the PDB.

38.3.5 Channel n Control register 1 (PDBx_CHnC1)

Each PDB channel has one control register, CHnC1. The fields in this register control the functionality of each PDB channel operation.

Address: Base address + 10h offset + (40d × i), where i=0d to 0d

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

PDBx_CHnC1 field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 BB	PDB Channel Pre-Trigger Back-to-Back Operation Enable

Table continues on the next page...

PDBx_CHnC1 field descriptions (continued)

Field	Description
	Enables the PDB ADC pre-trigger operation as back-to-back mode. Only lower M pre-trigger bits are implemented in this MCU. Back-to-back operation enables the ADC conversions complete to trigger the next PDB channel pre-trigger and trigger output, so that the ADC conversions can be triggered on the next set of configuration and results registers. Application code must enable only the back-to-back operation of the PDB pre-triggers at the leading of the back-to-back connection chain. 0 PDB channel's corresponding pre-trigger back-to-back operation disabled. 1 PDB channel's corresponding pre-trigger back-to-back operation enabled.
15–8 TOS	PDB Channel Pre-Trigger Output Select These bits select the PDB ADC pre-trigger outputs. Only lower M pre-trigger bits are implemented in this MCU. 0 PDB channel's corresponding pre-trigger is in bypassed mode. The pre-trigger asserts one peripheral clock cycle after a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1. 1 PDB channel's corresponding pre-trigger asserts when the counter reaches the channel delay register plus one peripheral clock cycle after a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1.
EN	PDB Channel Pre-Trigger Enable Enables the PDB ADC pre-trigger outputs. Only lower M pre-trigger fields are implemented in this MCU. 0 PDB channel's corresponding pre-trigger disabled. 1 PDB channel's corresponding pre-trigger enabled.

38.3.6 Channel n Status register (PDBx_CHnS)

Address: Base address + 14h offset + (40d × i), where i=0d to 0d

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

PDBx_CHnS field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 CF	PDB Channel Flags The CF[*m*] field is set when the PDB counter (PDB_CNT) matches the value CHnDLY*m* + 1. Write 0 to clear CF.
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ERR	PDB Channel Sequence Error Flags Only the lower M bits are implemented in this MCU.

Table continues on the next page...

PDBx_CHnS field descriptions (continued)

Field	Description
	0 Sequence error not detected on PDB channel's corresponding pre-trigger. 1 Sequence error detected on PDB channel's corresponding pre-trigger. ADCn block can be triggered for a conversion by one pre-trigger from PDB channel *n*. When one conversion, which is triggered by one of the pre-triggers from PDB channel *n*, is in progress, new trigger from PDB channel's corresponding pre-trigger m cannot be accepted by ADCn, and ERR[m] is set. Writing 0's to clear the sequence error flags.

38.3.7 Channel n Delay 0 register (PDBx_CHnDLY0)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 18h offset + (40d × i), where i=0d to 0d

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

PDBx_CHnDLY0 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay Specifies the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading this field returns the value of internal register that is effective for the current PDB cycle.

38.3.8 Channel n Delay 1 register (PDB*x*_CH*n*DLY1)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 1Ch offset + (40d × i), where i=0d to 0d

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

PDB*x*_CH*n*DLY1 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

38.3.9 Channel n Delay 2 register (PDB*x*_CH*n*DLY2)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 20h offset + (40d × i), where i=0d to 0d

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

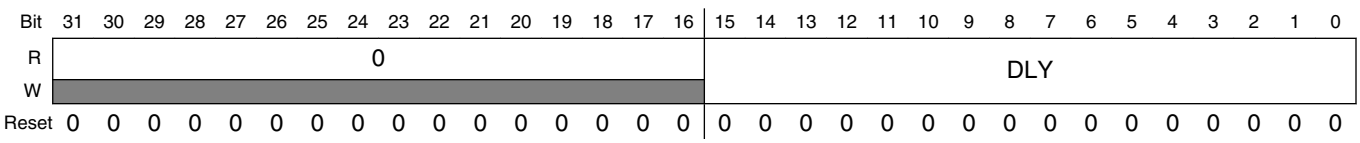
PDBx_CHnDLY2 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

38.3.10 Channel n Delay 3 register (PDBx_CHnDLY3)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 24h offset + (40d × i), where i=0d to 0d



PDBx_CHnDLY3 field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

38.3.11 DAC Interval Trigger n Control register (PDBx_DACINTC*n*)

Address: Base address + 150h offset + (8d × i), where i=0d to 0d

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

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

PDBx_DACINTC*n* field descriptions

Field	Description
31–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 EXT	DAC External Trigger Input Enable This bit enables the external trigger for DAC interval counter. 0 DAC external trigger input disabled. DAC interval counter is reset and started counting when a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1. 1 DAC external trigger input enabled. DAC interval counter is bypassed and DAC external trigger input triggers the DAC interval trigger.
0 TOE	DAC Interval Trigger Enable Enables the DAC interval trigger. 0 DAC interval trigger disabled. 1 DAC interval trigger enabled.

38.3.12 DAC Interval n register (PDBx_DACINT*n*)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 154h offset + (8d × i), where i=0d to 0d

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

PDBx_DACINTn field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
INT	DAC Interval These bits specify the interval value for DAC interval trigger. DAC interval trigger triggers DAC[1:0] update when the DAC interval counter is equal to the DACINT. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

38.3.13 Pulse-Out n Enable register (PDBx_POEN)

Address: Base address + 190h offset

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

PDBx_POEN field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
POEN	PDB Pulse-Out Enable Enables the pulse output. Only lower 8 bits are implemented in this MCU. 0 PDB Pulse-Out disabled 1 PDB Pulse-Out enabled

38.3.14 Pulse-Out n Delay register (PDBx_POnDLY)

Note: This register is internally buffered, and any values written to the register are written to its internal buffer instead; in other words, the internal device bus does not write directly to this register. The value in this register's internal buffer is loaded into this register only after "1" is written to the SC[LDOK] bit.

Address: Base address + 194h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY1																DLY2															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PDBx_PO*n*DLY field descriptions

Field	Description
31–16 DLY1	PDB Pulse-Out Delay 1 These bits specify the delay 1 value for the PDB Pulse-Out. Pulse-Out goes high when the PDB counter is equal to the DLY1. Reading these bits returns the value of internal register that is effective for the current PDB cycle.
DLY2	PDB Pulse-Out Delay 2 These bits specify the delay 2 value for the PDB Pulse-Out. Pulse-Out goes low when the PDB counter is equal to the DLY2. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

38.4 Functional description

38.4.1 PDB pre-trigger and trigger outputs

The PDB contains a counter whose output is compared to several different digital values. If the PDB is enabled, then a trigger input event will reset the counter and make it start to count. A trigger input event is defined as a rising edge being detected on a selected trigger input source, or if a software trigger is selected and the Software Trigger bit (SC[SWTRIG]) is written with 1. For each channel, a delay *m* determines the time between assertion of the trigger input event to the time at which changes in the pre-trigger *m* output signal are started. The time is defined as:

- Trigger input event to pre-trigger *m* = (prescaler × multiplication factor × delay *m*) + 2 peripheral clock cycles
- Add 1 additional peripheral clock cycle to determine the time when the channel trigger output changes.

Each channel is associated with 1 ADC block. PDB channel n pre-trigger outputs 0 to *M*; each pre-trigger output is connected to ADC hardware trigger select and hardware trigger inputs. The pre-triggers are used to precondition the ADC block before the actual trigger occurs. When the ADC receives the rising edge of the trigger, the ADC will start the conversion according to the precondition determined by the pre-triggers. The ADC contains *M* sets of configuration and result registers, allowing it to alternate conversions between *M* different analog sources (like a ping-pong game). The pre-trigger outputs are used to specify which signal will be sampled next. When a pre-trigger *m* is asserted, the ADC conversion is triggered with set *m* of the configuration and result registers.

The waveforms shown in the following diagram show the pre-trigger and trigger outputs of PDB channel *n*. The delays can be independently set using the channel delay registers (CH*n*DLY*m*), and the pre-triggers can be enabled or disabled using the PDB Channel Pre-Trigger Enables (CH*n*C1[EN[*m*]]).

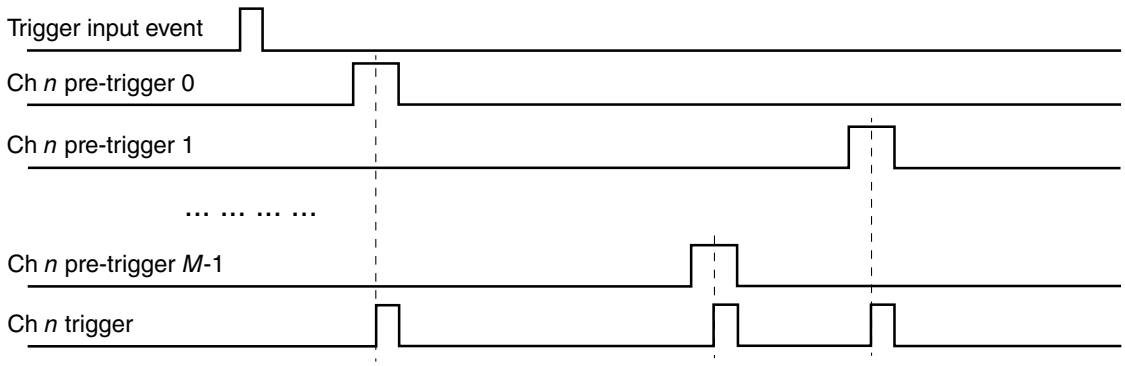


Figure 38-2. Pre-trigger and trigger outputs

The delay in the channel delay register (CH*n*DLY*m*) can be optionally bypassed, if the PDB Channel Pre-Trigger Output Select (CH*n*C1[TOS[*m*]]) is cleared. In this case, when the trigger input event occurs, the pre-trigger *m* is asserted after 2 peripheral clock cycles.

The PDB can be configured for back-to-back operation. Back-to-back operation enables the ADC conversion completions to trigger the next PDB channel pre-trigger and trigger outputs, so that the ADC conversions can be triggered on the next set of configuration and results registers. When back-to-back operation is enabled by setting the PDB Channel Pre-Trigger Back-to-Back Operation Enable (CH*n*C1[BB[*m*]]), then the delay *m* is ignored and the pre-trigger *m* is asserted 2 peripheral cycles after the acknowledgment *m* is received. The acknowledgment connections in this MCU are described in Back-to-back acknowledgment connections.

When a pre-trigger from a PDB channel *n* is asserted, the associated lock of the pre-trigger becomes active. The associated lock is released by the rising edge of the corresponding ADC_STATE[EOSI1/EOSI0]; the ADC_STATE[EOSI1/EOSI0] should be cleared after the conversion result is read, so that the next rising edge of ADC_STATE[EOSI1/EOSI0] can be generated to clear the lock later. The lock becomes inactive when:

- the rising edge of corresponding ADC_STATE[EOSI1/EOSI0] occurs,
- or the corresponding PDB pre-trigger is disabled,
- or the PDB is disabled

The channel *n* trigger output is suppressed when any of the locks of the pre-triggers in channel *n* is active.

- If a new pre-trigger *m* asserts when there is active lock in the PDB channel *n*, then a PDB Channel Sequence Error Flag (CH*n*S[ERR[*m*]], associated with the pre-trigger *m*) is set. If the PDB Sequence Error Interrupt Enable (SC[PDBEIE]) is set, then the sequence error interrupt is generated. A sequence error typically happens because the delay *m* is set too short and the pre-trigger *m* asserts before the previously triggered ADC conversion finishes.
- If the pre-trigger delay is 0 cycles, then both channels will flag a PDB channel sequence error and ADC will not perform a conversion.

The PDB reports PDB channel sequence errors only for pre-triggers in same PDB channel. For situations when PDB triggering is done through different PDB channels, software must ensure sufficient delays in between the pre-triggers.

When the PDB counter reaches the value (CNT + 1), the PDB Interrupt Flag (SC[PDBIF]) is set. A PDB interrupt can be generated if the PDB Sequence Error Interrupt Enable (SC[PDBEIE]) is set and the DMA Enable (SC[DMAEN]) is cleared. If the DMA Enable (SC[DMAEN]) is set, then the PDB requests a DMA transfer when the PDB Interrupt Flag (SC[PDBIF]) is set.

The modulus value in the Modulus register (MOD) is used to reset the counter back to zero at the end of the count. If the Continuous Mode Enable (SC[CONT]) is set, then the counter will then resume a new count; otherwise, the counter operation will stop until the next trigger input event occurs.

38.4.2 PDB trigger input source selection

They are connected to on-chip or off-chip event sources. The PDB can be triggered by software through SC[SWTRIG].

For the trigger input sources implemented in this MCU, see chip configuration information.

38.4.3 DAC interval trigger outputs

PDB can generate the interval triggers for DACs to update their outputs periodically. DAC interval counter *x* is reset and started when a trigger input event occurs if DACINTC*x*[EXT] is cleared. When the interval counter *x* is equal to the value set in DACINT*x* register, the DAC interval trigger *x* output generates a pulse of one peripheral clock cycle width to update the DAC*x*. If DACINTC*x*[EXT] is set, the DAC interval

counter is bypassed and the interval trigger output *x* generates a pulse following the detection of a rising edge on the DAC external trigger input. The counter and interval trigger can be disabled by clearing the DACINTC*x*[TOE].

DAC interval counters are also reset when the PDB counter reaches the MOD register value; therefore, when the PDB counter rolls over to zero, the DAC interval counters starts anew.

The DAC interval trigger pulse and the ADC pre-trigger/trigger pulses together allow precise timing of DAC updates and ADC measurements. This is outlined in the typical use case described in the following diagram.

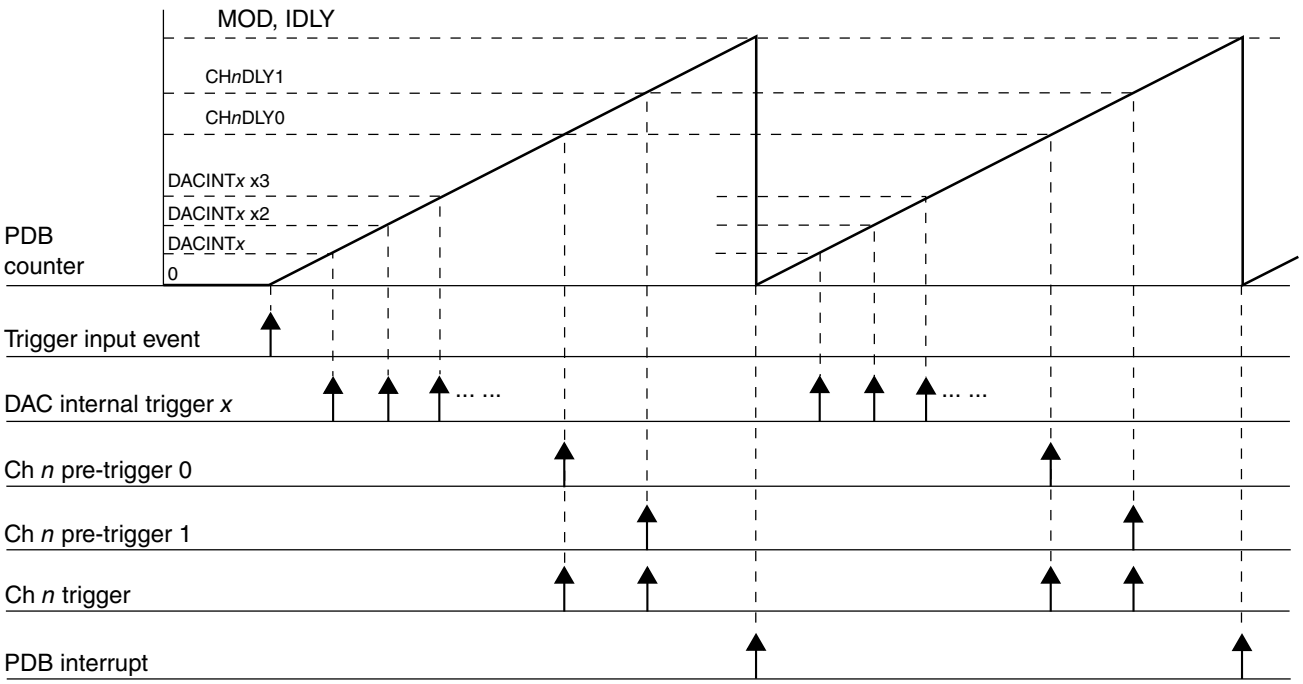


Figure 38-3. PDB ADC triggers and DAC interval triggers use case

NOTE

Because the DAC interval counters share the prescaler with PDB counter, PDB must be enabled if the DAC interval trigger outputs are used in the applications.

38.4.4 Pulse-Out's

PDB can generate pulse outputs of configurable width.

- When the PDB counter reaches the value set in PO*y*DLY[DLY1], then the Pulse-Out goes high.
- When the PDB counter reaches PO*y*DLY[DLY2], then it goes low.

POyDLY[DLY2] can be set either greater or less than POyDLY[DLY1].

ADC pre-trigger/trigger outputs and Pulse-Out generation have the same time base, because they both share the PDB counter. The pulse-out connections implemented in this MCU are described in the device's chip configuration details.

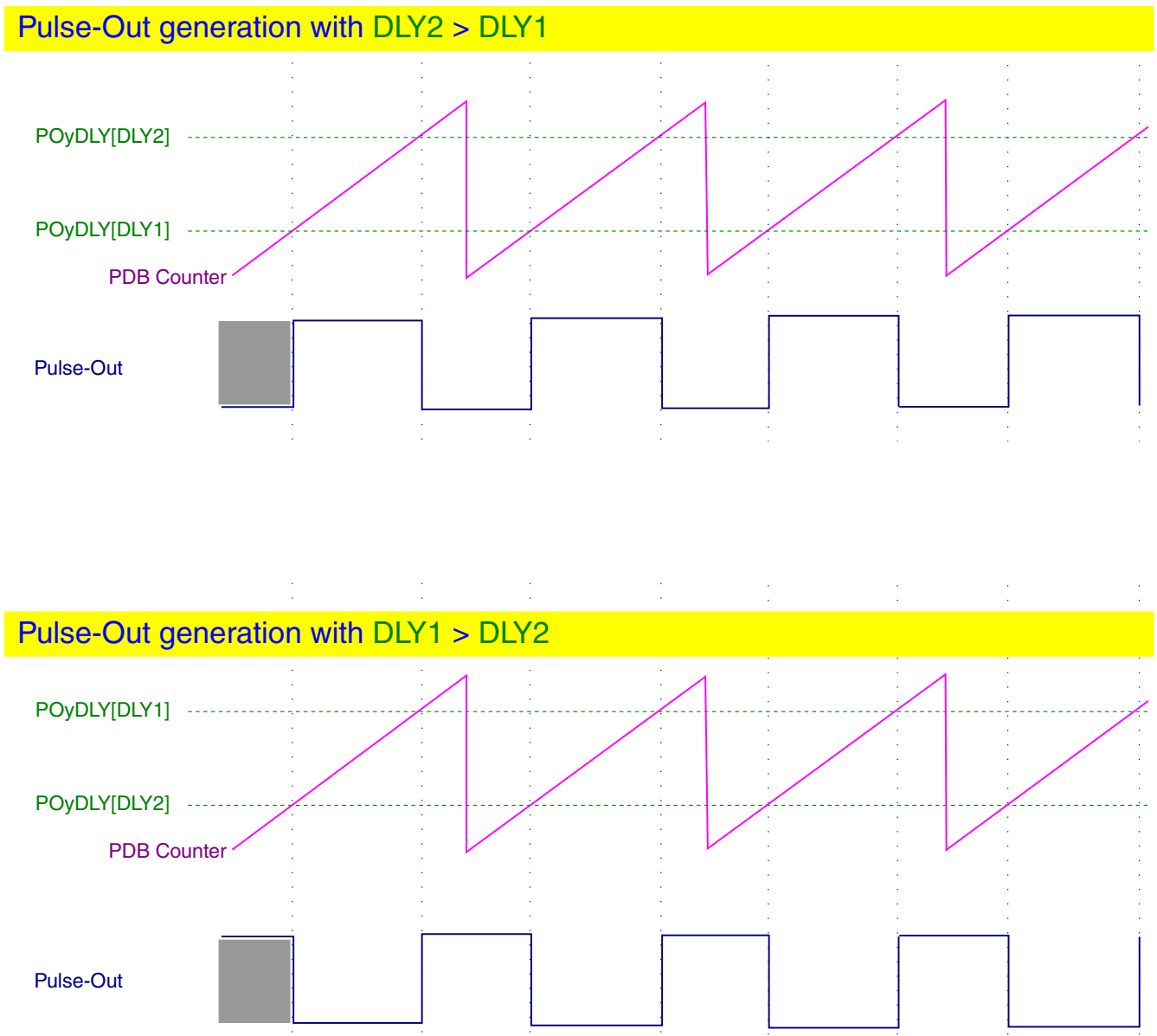


Figure 38-4. How Pulse Out is generated

38.4.5 Updating the delay registers

The following registers control the timing of the PDB operation; and in some of the applications, they may need to become effective at the same time.

- PDB Modulus register (MOD)
- PDB Interrupt Delay register (IDLY)
- PDB Channel *n* Delay *m* register (CH*n*DLY*m*)
- DAC Interval *x* register (DACINT*x*)
- PDB Pulse-Out *y* Delay register (PO*y*DLY)

The internal registers of them are buffered and any values written to them are written first to their buffers. The circumstances that cause their internal registers to be updated with the values from the buffers are summarized in the next table.

Table 38-6. When delay registers are updated

SC[LDMOD]	Update to the delay registers
00	The internal registers are loaded with the values from their buffers immediately after 1 is written to SC[LDOK].
01	The PDB counter reaches PDB_MOD[MOD] + 1 value, after 1 is written to SC[LDOK].
10	A trigger input event is detected after 1 is written to SC[LDOK].
11	Either the PDB counter reaches PDB_MOD[MOD] + 1 value or a trigger input event is detected, after 1 is written to SC[LDOK].

After 1 is written to SC[LDOK], the buffers cannot be written until the values in buffers are loaded into their internal registers. SC[LDOK] is self-cleared when the internal registers are loaded, so the application code can read it to determine the updates to the internal registers.

The following diagrams show the cases of the internal registers being updated with SC[LDMOD] is 00 and x1.

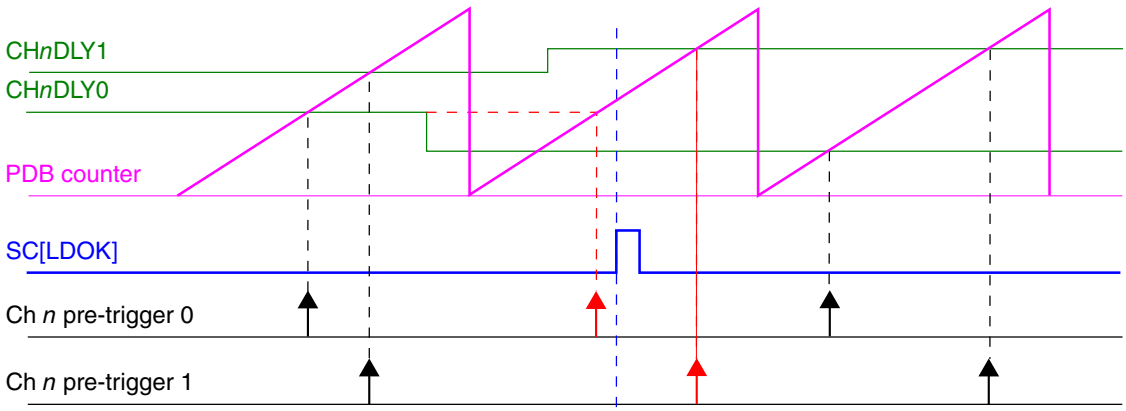


Figure 38-5. Registers update with SC[LDMOD] = 00

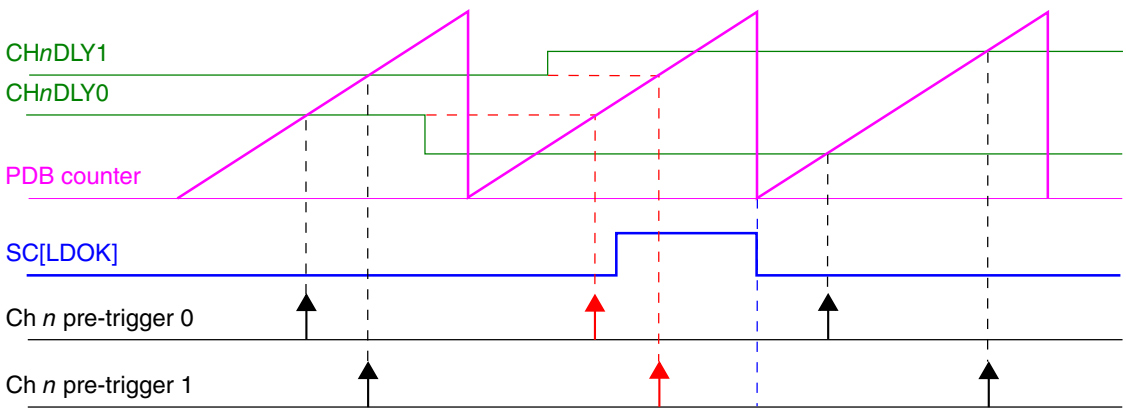


Figure 38-6. Registers update with SC[LDMOD] = x1

38.4.6 Interrupts

PDB can generate two interrupts: PDB interrupt and PDB sequence error interrupt. The following table summarizes the interrupts.

Table 38-7. PDB interrupt summary

Interrupt	Flags	Enable bit
PDB Interrupt	SC[PDBIF]	SC[PDBIE] = 1 and SC[DMAEN] = 0
PDB Sequence Error Interrupt	CH*n*S[ERR*m*]	SC[PDBEIE] = 1

38.4.7 DMA

If SC[DMAEN] is set, PDB can generate a DMA transfer request when SC[PDBIF] is set. When DMA is enabled, the PDB interrupt is not issued.

38.5 Application information

38.5.1 Impact of using the prescaler and multiplication factor on timing resolution

Use of prescaler and multiplication factor greater than 1 limits the count/delay accuracy in terms of peripheral clock cycles (to the modulus of the prescaler X multiplication factor). If the multiplication factor is set to 1 and the prescaler is set to 2 then the only values of total peripheral clocks that can be detected are even values; if prescaler is set to 4 then the only values of total peripheral clocks that can be decoded as detected are mod(4) and so forth. If the applications need a really long delay value and use a prescaler set to 128, then the resolution would be limited to 128 peripheral clock cycles.

Therefore, use the lowest possible prescaler and multiplication factor for a given application.

Chapter 39
FlexTimer Module (FTM)

39.1 Chip-specific FTM information

39.1.1 Instantiation Information

This device contains three FlexTimer modules.

The following table shows how these modules are configured. This adheres to Kinetis K series FTM instantiations.

Table 39-1. FTM Instantiations

FTM instance	Number of channels	Features/usage
FTM0	8	3-phase motor control
FTM1	2[1]	Quadrature decoder or general purpose
FTM3	8	3-phase motor control

1. These registers are not available in FTM1: FTM1_C2SC, FTM1_C3SC, FTM1_C4SC, FTM1_C5SC, FTM1_C6SC, FTM1_C7SC, FTM1_C2V, FTM1_C3V, FTM1_C4V, FTM1_C5V, FTM1_C6V, and FTM1_C7V

NOTE

FTM0 and FTM3 has Quadrature Decoder registers but doesn't route its pins out, so QE function can not be used on FTM0 and FTM3.

NOTE

CnSC[ICRST] is available only on FTM1, which has Hall sensor decoder mode. FTM0 and FTM3 do not have this field.

39.1.2 External Clock Options

By default each FTM is clocked by the internal fast peripheral clock (the FTM refers to it as system clock). Each module contains a register setting that allows the module to be clocked from an external clock instead. There are three external FTM_CLKINx pins that can be selected by any FTM module via the SIM_SOPT4 register as an external clock source. The pinmux options may provide optional external inputs.

- FTM_CLKIN0 can provide an external input clock to all or either of FTM0,FTM1 and FTM3
- FTM_CLKIN1 can provide an external input clock to all or either of FTM0,FTM1 and FTM3
- FTM_CLKIN2 can provide an external input clock to all or either of FTM0,FTM1 and FTM3

39.1.3 Fixed frequency clock

The fixed frequency clock for each FTM is MCGFFCLK.

39.1.4 FTM Interrupts

The FlexTimer has multiple sources of interrupt. However, these sources are OR'd together to generate a single interrupt request to the interrupt controller. When an FTM interrupt occurs, read the FTM status registers (FMS, SC, and STATUS) to determine the exact interrupt source.

39.1.5 FTM Fault Detection Inputs

The following fault detection input options for the FTM modules are selected via the SIM_SOPT4 register. The external pin option is selected by default.

- FTM0 FAULT0 = FTM0_FLT0 pin or CMP0 output via SIM Option register control
- FTM0 FAULT1 = FTM0_FLT1 pin or CMP1 output via SIM Option register control
- FTM0 FAULT2 = FTM0_FLT2 pin or CMP2 output via SIM Option regsiter control
- FTM0_FAULT3 = XBARA_OUT49 or FTM0_FLT3 pin selected by bit in SIM_SOPT4 register

- FTM1 FAULT0 = FTM1_FLT0 pin or CMP0 output via SIM option register control
- FTM1FAULT1 = XBAR_OUT50

- FTM3FAULT0 = FTM3_FLT0 pin or CMP0 output via SIM Option register control
- FTM3FAULT1 = reserved
- FTM3FAULT2 = reserved
- FTM3FAULT3 = XBARA_OUT52

39.1.6 FTM Hardware Triggers

The FTM input synchronization hardware triggers are connected in the chip as follows:

- FTM0 hardware trigger 0 = FTM0_SYNCBIT of SIM_OPTx register, CMP0 Output or FTM1 Match output.
- FTM0 hardware trigger 1 = PDB0 channel 1 trigger Output or FTM1 Match output
- FTM0 hardware trigger 2 = XBARA_OUT34

- FTM1 hardware trigger 0 = FTM1_SYNCBIT of SIM_OPTx register, CMP0 Output or FTM0 Match output
- FTM1 hardware trigger 1 = CMP1 Output
- FTM1 hardware trigger 2 = XBARA_OUT35

- FTM3 hardware trigger 0 = FTM3_SYNCBIT of SIM_OPTx register, CMP0 Output or FTM1 Match output
- FTM3 hardware trigger 1 = PDB1 channel 1 Trigger Output or FTM1 Match output
- FTM3 hardware trigger 2 = XBARA_OUT37

39.1.7 Input capture options for FTM module instances

The following channel 0 input capture source options are selected via SIM_SOPTx. The external pin option is selected by default.

- FTM1 channel 0 input capture = FTM1_CH0 pin or CMP0 output or CMP1 output via FTM1CH0SRC bit in SIM_OPTx regsiter
- FTM1 channel 1 input capture = FTM1_CH1 pin or exclusive OR of FTM1_CH0, FTM1_CH1, and XBARA_OUT42 via FTM1CH1SRC bit in SIM_SOPT9 register.

39.1.8 FTM output triggers for other modules

The FTM output triggers can be selected as input triggers for the PDB and ADC modules.

FTM0 has 8 channels of trigger capability which are ORed together to force an output trigger. This EXTTRG0 signal is connected to PDB0 channel 1000 input, PDB1 channel 1000 input, XBARA_IN16 and XBARB_IN4.

Individual FTM0 channels trigger signal outputs are also assigned to the DMAMUX

FTM0_CH0 event output to DMAMUX source 24

FTM0_CH1 event output to DMAMUX source 25

FTM0_CH2 event output to DMAMUX source 26

FTM0_CH3 event output to DMAMUX source 27

FTM0_CH4 event output to DMAMUX source 28

FTM0_CH5 event output to DMAMUX source 29

FTM0_CH6 event output to DMAMUX source 30

FTM0_CH7 event output to DMAMUX source 31

FTM1 has 2 channels of trigger capability which are ORed together to force an output trigger. This EXTRRG1 signal is connected to PDB0 channel 1001 input, PDB1 channel 1001 input, XBARA_IN36 and XBARB_IN16.

FTM1_CH0 event output to DMAMUX source 32, UART0TXSRC and UART1TXSRC of SIM_OPTx register.

FTM1_CH1 event output to DMAMUX source 33

FTM3 has 8 channels of trigger capability which are ORed together to force an output trigger. This EXTTRG3 signal is connected to PDB0 channel 1011 input, PDB1 channel 1011 input, XBARA_IN18 and XBARB_IN6.

FTM3_CH0 event output to DMAMUX source 36

FTM3_CH1 event output to DMAMUX source 37

FTM3_CH2 event output to DMAMUX source 38

FTM3_CH3 event output to DMAMUX source 39

FTM3_CH4 event output to DMAMUX source 54

FTM3_CH5 event output to DMAMUX source 55

FTM3_CH6 event output to DMAMUX source 56

FTM3_CH7 event output to DMAMUX source 57

39.1.9 FTM Global Time Base

This chip provides the optional FTM global time base feature (see Global time base (GTB)).

FTM0 provides the only source for the FTM global time base. The other FTM modules can share the time base as shown in the following figure:

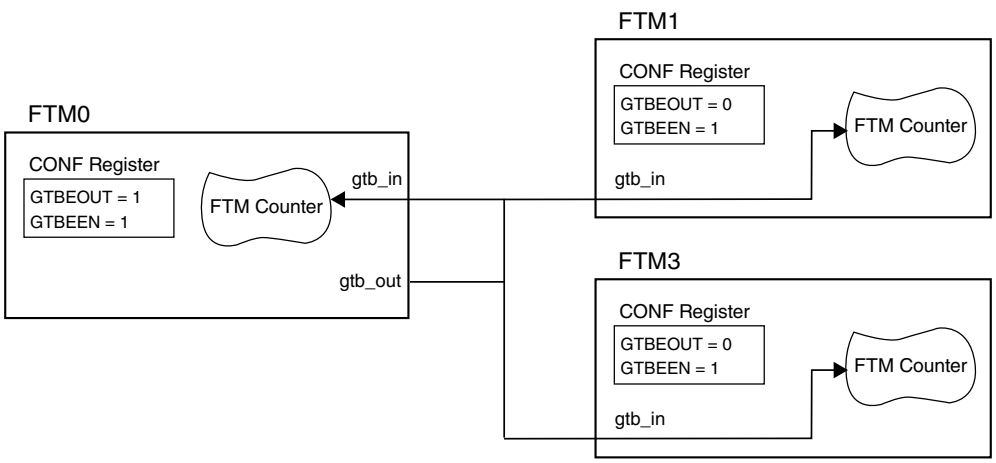


Figure 39-1. FTM Global Time Base Configuration

39.1.10 FTM BDM and debug halt mode

In the FTM chapter, references to the chip being in "BDM" are the same as the chip being in “debug halt mode".

39.2 Introduction

The FlexTimer module (FTM) is a two-to-eight channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports

only six channels, references to channel numbers 6 and 7 do not apply for that instance.

39.2.1 FlexTimer philosophy

The FlexTimer is built upon a simple timer, the Timer PWM Module – TPM, used for many years on our HCS08 family of 8-bit microcontrollers. The FlexTimer extends the functionality to meet the demands of motor control, digital lighting solutions, and power conversion, while providing low cost and backwards compatibility with the TPM module.

Several key enhancements are made:

- Signed up counter
- Deadtime insertion hardware
- Fault control inputs
- Enhanced triggering functionality
- Initialization and polarity control

All of the features common with the TPM have fully backwards compatible register assignments. The FlexTimer can also use code on the same core platform without change to perform the same functions.

Motor control and power conversion features have been added through a dedicated set of registers and defaults turn off all new features. The new features, such as hardware deadtime insertion, polarity, fault control, and output forcing and masking, greatly reduce loading on the execution software and are usually each controlled by a group of registers.

FlexTimer input triggers can be from comparators, ADC, or other submodules to initiate timer functions automatically. These triggers can be linked in a variety of ways during integration of the sub modules so please note the options available for used FlexTimer configuration.

More than one FlexTimers may be synchronized to provide a larger timer with their counters incrementing in unison, assuming the initialization, the input clocks, the initial and final counting values are the same in each FlexTimer.

All main user access registers are buffered to ease the load on the executing software. A number of trigger options exist to determine which registers are updated with this user defined data.

39.2.2 Features

The FTM features include:

- FTM source clock is selectable.
  - The source clock can be the system clock, or an external clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit counter
  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In Input Capture mode:
  - The capture can occur on rising edges, falling edges or both edges
  - An input filter can be selected for some channels
- In Output Compare mode the output signal can be set, cleared, or toggled on match
- All channels can be configured for center-aligned PWM mode
- Each pair of channels can be combined to generate a PWM signal with independent control of both edges of PWM signal
- The FTM channels can operate as pairs with equal outputs, pairs with complementary outputs, or independent channels with independent outputs
- The deadtime insertion is available for each complementary pair
- Generation of match triggers
- Initialization trigger
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- The polarity of each channel is configurable
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when the fault condition is detected
- Synchronized loading of write buffered FTM registers

- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input captures for a stuck at zero and one conditions
- Dual edge capture for pulse and period width measurement
- Quadrature decoder with input filters, relative position counting, and interrupt on position count or capture of position count on external event

39.2.3 Modes of operation

When the chip is in an active BDM mode, the FTM temporarily suspends all counting until the chip returns to normal user operating mode. During Stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During Wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the chip from Wait mode, the power can then be saved by disabling FTM functions before entering Wait mode.

39.2.4 Block diagram

The FTM uses one input/output (I/O) pin per channel, CHn (FTM channel (n)) where n is the channel number (0–7).

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

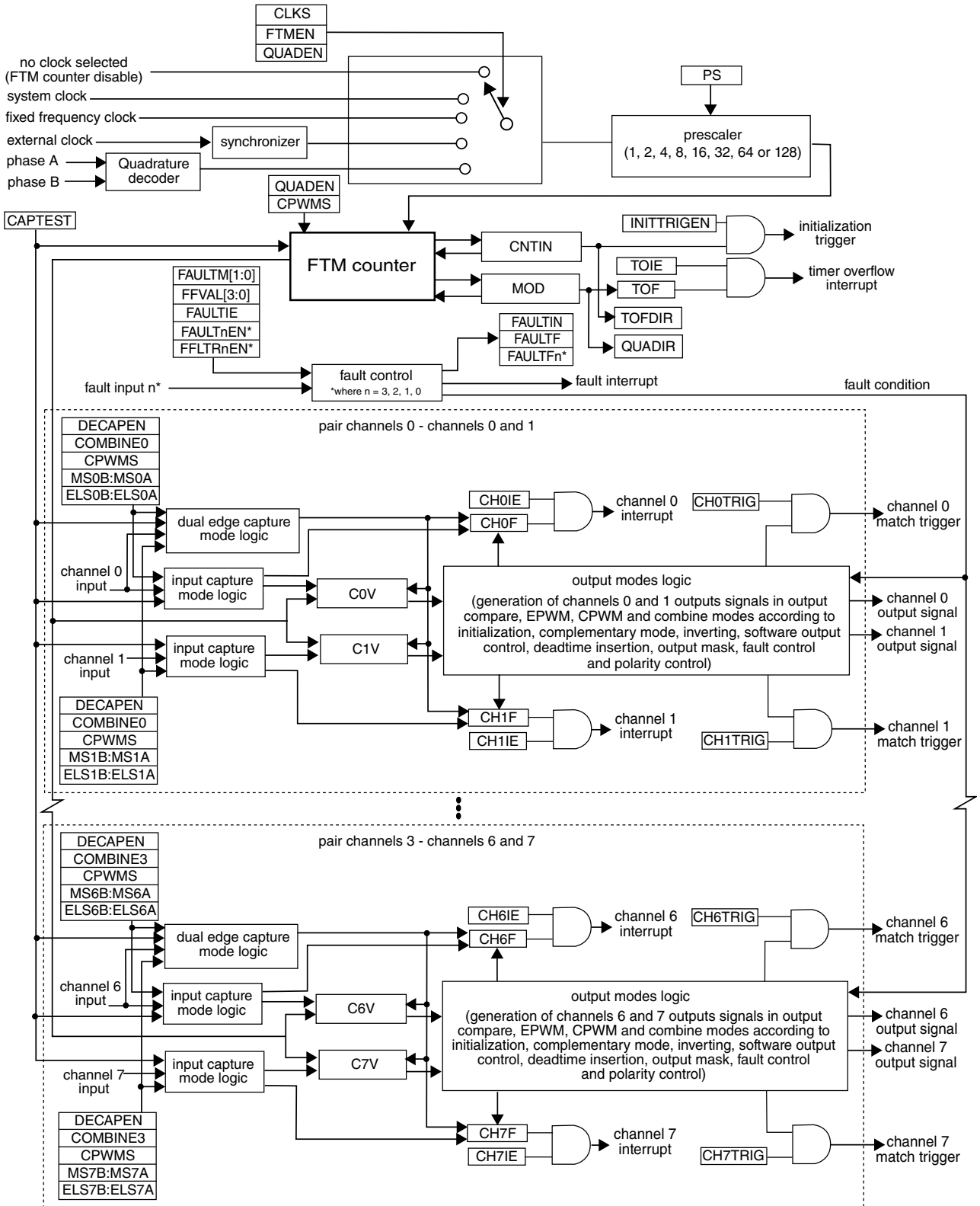


Figure 39-2. FTM block diagram

39.3 FTM signal descriptions

Table 39-2 shows the user-accessible signals for the FTM.

Table 39-2. FTM signal descriptions

Signal	Description	I/O	Function
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I	The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of system clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.
CHn	FTM channel (n), where n can be 7-0	I/O	Each FTM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.
FAULTj	Fault input (j), where j can be 3-0	I	The fault input signals are used to control the CHn channel output state. If a fault is detected, the FAULTj signal is asserted and the channel output is put in a safe state. The behavior of the fault logic is defined by the FAULTM[1:0] control bits in the MODE register and FAULTEN bit in the COMBINEm register. Note that each FAULTj input may affect all channels selectively since FAULTM[1:0] and FAULTEN control bits are defined for each pair of channels. Because there are several FAULTj inputs, maximum of 4 for the FTM module, each one of these inputs is activated by the FAULTjEN bit in the FLTCTRL register.
PHA	Quadrature decoder phase A input. Input pin associated with quadrature decoder phase A.	I	The quadrature decoder phase A input is used as the Quadrature Decoder mode is selected. The phase A input signal is one of the signals that control the FTM counter increment or decrement in the Quadrature Decoder mode.
PHB	Quadrature decoder phase B input. Input pin associated with quadrature decoder phase B.	I	The quadrature decoder phase B input is used as the Quadrature Decoder mode is selected. The phase B input signal is one of the signals that control the FTM counter increment or decrement in the Quadrature Decoder mode.

39.4 Memory map and register definition

39.4.1 Memory map

This section presents a high-level summary of the FTM registers and how they are mapped.

The registers and bits of an unavailable function in the FTM remain in the memory map and in the reset value, but they have no active function.

Note

Do not write in the region from the CNTIN register through the PWMLOAD register when FTMEN = 0.

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

39.4.2 Register descriptions

Accesses to reserved addresses result in transfer errors. Registers for absent channels are considered reserved.

FTM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_6000	Status And Control (FTM3_SC)	32	R/W	0000_0000h	39.4.3/898
4002_6004	Counter (FTM3_CNT)	32	R/W	0000_0000h	39.4.4/899
4002_6008	Modulo (FTM3_MOD)	32	R/W	0000_0000h	39.4.5/900
4002_600C	Channel (n) Status And Control (FTM3_C0SC)	32	R/W	0000_0000h	39.4.6/901
4002_6010	Channel (n) Value (FTM3_C0V)	32	R/W	0000_0000h	39.4.7/904
4002_6014	Channel (n) Status And Control (FTM3_C1SC)	32	R/W	0000_0000h	39.4.6/901
4002_6018	Channel (n) Value (FTM3_C1V)	32	R/W	0000_0000h	39.4.7/904
4002_601C	Channel (n) Status And Control (FTM3_C2SC)	32	R/W	0000_0000h	39.4.6/901
4002_6020	Channel (n) Value (FTM3_C2V)	32	R/W	0000_0000h	39.4.7/904
4002_6024	Channel (n) Status And Control (FTM3_C3SC)	32	R/W	0000_0000h	39.4.6/901
4002_6028	Channel (n) Value (FTM3_C3V)	32	R/W	0000_0000h	39.4.7/904
4002_602C	Channel (n) Status And Control (FTM3_C4SC)	32	R/W	0000_0000h	39.4.6/901
4002_6030	Channel (n) Value (FTM3_C4V)	32	R/W	0000_0000h	39.4.7/904
4002_6034	Channel (n) Status And Control (FTM3_C5SC)	32	R/W	0000_0000h	39.4.6/901
4002_6038	Channel (n) Value (FTM3_C5V)	32	R/W	0000_0000h	39.4.7/904
4002_603C	Channel (n) Status And Control (FTM3_C6SC)	32	R/W	0000_0000h	39.4.6/901
4002_6040	Channel (n) Value (FTM3_C6V)	32	R/W	0000_0000h	39.4.7/904
4002_6044	Channel (n) Status And Control (FTM3_C7SC)	32	R/W	0000_0000h	39.4.6/901
4002_6048	Channel (n) Value (FTM3_C7V)	32	R/W	0000_0000h	39.4.7/904
4002_604C	Counter Initial Value (FTM3_CNTIN)	32	R/W	0000_0000h	39.4.8/904

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_6050	Capture And Compare Status (FTM3_STATUS)	32	R/W	0000_0000h	39.4.9/905
4002_6054	Features Mode Selection (FTM3_MODE)	32	R/W	0000_0004h	39.4.10/ 907
4002_6058	Synchronization (FTM3_SYNC)	32	R/W	0000_0000h	39.4.11/ 909
4002_605C	Initial State For Channels Output (FTM3_OUTINIT)	32	R/W	0000_0000h	39.4.12/ 911
4002_6060	Output Mask (FTM3_OUTMASK)	32	R/W	0000_0000h	39.4.13/ 912
4002_6064	Function For Linked Channels (FTM3_COMBINE)	32	R/W	0000_0000h	39.4.14/ 914
4002_6068	Deadtime Insertion Control (FTM3_DEADTIME)	32	R/W	0000_0000h	39.4.15/ 919
4002_606C	FTM External Trigger (FTM3_EXTTRIG)	32	R/W	0000_0000h	39.4.16/ 920
4002_6070	Channels Polarity (FTM3_POL)	32	R/W	0000_0000h	39.4.17/ 922
4002_6074	Fault Mode Status (FTM3_FMS)	32	R/W	0000_0000h	39.4.18/ 924
4002_6078	Input Capture Filter Control (FTM3_FILTER)	32	R/W	0000_0000h	39.4.19/ 926
4002_607C	Fault Control (FTM3_FLTCTRL)	32	R/W	0000_0000h	39.4.20/ 927
4002_6080	Quadrature Decoder Control And Status (FTM3_QDCTRL)	32	R/W	0000_0000h	39.4.21/ 930
4002_6084	Configuration (FTM3_CONF)	32	R/W	0000_0000h	39.4.22/ 932
4002_6088	FTM Fault Input Polarity (FTM3_FLTPOL)	32	R/W	0000_0000h	39.4.23/ 933
4002_608C	Synchronization Configuration (FTM3_SYNCONF)	32	R/W	0000_0000h	39.4.24/ 934
4002_6090	FTM Inverting Control (FTM3_INVCTRL)	32	R/W	0000_0000h	39.4.25/ 936
4002_6094	FTM Software Output Control (FTM3_SWOCTRL)	32	R/W	0000_0000h	39.4.26/ 937
4002_6098	FTM PWM Load (FTM3_PWMLOAD)	32	R/W	0000_0000h	39.4.27/ 940
4003_8000	Status And Control (FTM0_SC)	32	R/W	0000_0000h	39.4.3/898
4003_8004	Counter (FTM0_CNT)	32	R/W	0000_0000h	39.4.4/899
4003_8008	Modulo (FTM0_MOD)	32	R/W	0000_0000h	39.4.5/900
4003_800C	Channel (n) Status And Control (FTM0_C0SC)	32	R/W	0000_0000h	39.4.6/901
4003_8010	Channel (n) Value (FTM0_C0V)	32	R/W	0000_0000h	39.4.7/904
4003_8014	Channel (n) Status And Control (FTM0_C1SC)	32	R/W	0000_0000h	39.4.6/901

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8018	Channel (n) Value (FTM0_C1V)	32	R/W	0000_0000h	39.4.7/904
4003_801C	Channel (n) Status And Control (FTM0_C2SC)	32	R/W	0000_0000h	39.4.6/901
4003_8020	Channel (n) Value (FTM0_C2V)	32	R/W	0000_0000h	39.4.7/904
4003_8024	Channel (n) Status And Control (FTM0_C3SC)	32	R/W	0000_0000h	39.4.6/901
4003_8028	Channel (n) Value (FTM0_C3V)	32	R/W	0000_0000h	39.4.7/904
4003_802C	Channel (n) Status And Control (FTM0_C4SC)	32	R/W	0000_0000h	39.4.6/901
4003_8030	Channel (n) Value (FTM0_C4V)	32	R/W	0000_0000h	39.4.7/904
4003_8034	Channel (n) Status And Control (FTM0_C5SC)	32	R/W	0000_0000h	39.4.6/901
4003_8038	Channel (n) Value (FTM0_C5V)	32	R/W	0000_0000h	39.4.7/904
4003_803C	Channel (n) Status And Control (FTM0_C6SC)	32	R/W	0000_0000h	39.4.6/901
4003_8040	Channel (n) Value (FTM0_C6V)	32	R/W	0000_0000h	39.4.7/904
4003_8044	Channel (n) Status And Control (FTM0_C7SC)	32	R/W	0000_0000h	39.4.6/901
4003_8048	Channel (n) Value (FTM0_C7V)	32	R/W	0000_0000h	39.4.7/904
4003_804C	Counter Initial Value (FTM0_CNTIN)	32	R/W	0000_0000h	39.4.8/904
4003_8050	Capture And Compare Status (FTM0_STATUS)	32	R/W	0000_0000h	39.4.9/905
4003_8054	Features Mode Selection (FTM0_MODE)	32	R/W	0000_0004h	39.4.10/ 907
4003_8058	Synchronization (FTM0_SYNC)	32	R/W	0000_0000h	39.4.11/ 909
4003_805C	Initial State For Channels Output (FTM0_OUTINIT)	32	R/W	0000_0000h	39.4.12/ 911
4003_8060	Output Mask (FTM0_OUTMASK)	32	R/W	0000_0000h	39.4.13/ 912
4003_8064	Function For Linked Channels (FTM0_COMBINE)	32	R/W	0000_0000h	39.4.14/ 914
4003_8068	Deadtime Insertion Control (FTM0_DEADTIME)	32	R/W	0000_0000h	39.4.15/ 919
4003_806C	FTM External Trigger (FTM0_EXTTRIG)	32	R/W	0000_0000h	39.4.16/ 920
4003_8070	Channels Polarity (FTM0_POL)	32	R/W	0000_0000h	39.4.17/ 922
4003_8074	Fault Mode Status (FTM0_FMS)	32	R/W	0000_0000h	39.4.18/ 924
4003_8078	Input Capture Filter Control (FTM0_FILTER)	32	R/W	0000_0000h	39.4.19/ 926
4003_807C	Fault Control (FTM0_FLTCTRL)	32	R/W	0000_0000h	39.4.20/ 927
4003_8080	Quadrature Decoder Control And Status (FTM0_QDCTRL)	32	R/W	0000_0000h	39.4.21/ 930
4003_8084	Configuration (FTM0_CONF)	32	R/W	0000_0000h	39.4.22/ 932

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8088	FTM Fault Input Polarity (FTM0_FLTPOL)	32	R/W	0000_0000h	39.4.23/ 933
4003_808C	Synchronization Configuration (FTM0_SYNCONF)	32	R/W	0000_0000h	39.4.24/ 934
4003_8090	FTM Inverting Control (FTM0_INVCTRL)	32	R/W	0000_0000h	39.4.25/ 936
4003_8094	FTM Software Output Control (FTM0_SWOCTRL)	32	R/W	0000_0000h	39.4.26/ 937
4003_8098	FTM PWM Load (FTM0_PWMLOAD)	32	R/W	0000_0000h	39.4.27/ 940
4003_9000	Status And Control (FTM1_SC)	32	R/W	0000_0000h	39.4.3/898
4003_9004	Counter (FTM1_CNT)	32	R/W	0000_0000h	39.4.4/899
4003_9008	Modulo (FTM1_MOD)	32	R/W	0000_0000h	39.4.5/900
4003_900C	Channel (n) Status And Control (FTM1_C0SC)	32	R/W	0000_0000h	39.4.6/901
4003_9010	Channel (n) Value (FTM1_C0V)	32	R/W	0000_0000h	39.4.7/904
4003_9014	Channel (n) Status And Control (FTM1_C1SC)	32	R/W	0000_0000h	39.4.6/901
4003_9018	Channel (n) Value (FTM1_C1V)	32	R/W	0000_0000h	39.4.7/904
4003_901C	Channel (n) Status And Control (FTM1_C2SC)	32	R/W	0000_0000h	39.4.6/901
4003_9020	Channel (n) Value (FTM1_C2V)	32	R/W	0000_0000h	39.4.7/904
4003_9024	Channel (n) Status And Control (FTM1_C3SC)	32	R/W	0000_0000h	39.4.6/901
4003_9028	Channel (n) Value (FTM1_C3V)	32	R/W	0000_0000h	39.4.7/904
4003_902C	Channel (n) Status And Control (FTM1_C4SC)	32	R/W	0000_0000h	39.4.6/901
4003_9030	Channel (n) Value (FTM1_C4V)	32	R/W	0000_0000h	39.4.7/904
4003_9034	Channel (n) Status And Control (FTM1_C5SC)	32	R/W	0000_0000h	39.4.6/901
4003_9038	Channel (n) Value (FTM1_C5V)	32	R/W	0000_0000h	39.4.7/904
4003_903C	Channel (n) Status And Control (FTM1_C6SC)	32	R/W	0000_0000h	39.4.6/901
4003_9040	Channel (n) Value (FTM1_C6V)	32	R/W	0000_0000h	39.4.7/904
4003_9044	Channel (n) Status And Control (FTM1_C7SC)	32	R/W	0000_0000h	39.4.6/901
4003_9048	Channel (n) Value (FTM1_C7V)	32	R/W	0000_0000h	39.4.7/904
4003_904C	Counter Initial Value (FTM1_CNTIN)	32	R/W	0000_0000h	39.4.8/904
4003_9050	Capture And Compare Status (FTM1_STATUS)	32	R/W	0000_0000h	39.4.9/905
4003_9054	Features Mode Selection (FTM1_MODE)	32	R/W	0000_0004h	39.4.10/ 907
4003_9058	Synchronization (FTM1_SYNC)	32	R/W	0000_0000h	39.4.11/ 909
4003_905C	Initial State For Channels Output (FTM1_OUTINIT)	32	R/W	0000_0000h	39.4.12/ 911
4003_9060	Output Mask (FTM1_OUTMASK)	32	R/W	0000_0000h	39.4.13/ 912
4003_9064	Function For Linked Channels (FTM1_COMBINE)	32	R/W	0000_0000h	39.4.14/ 914

Table continues on the next page...

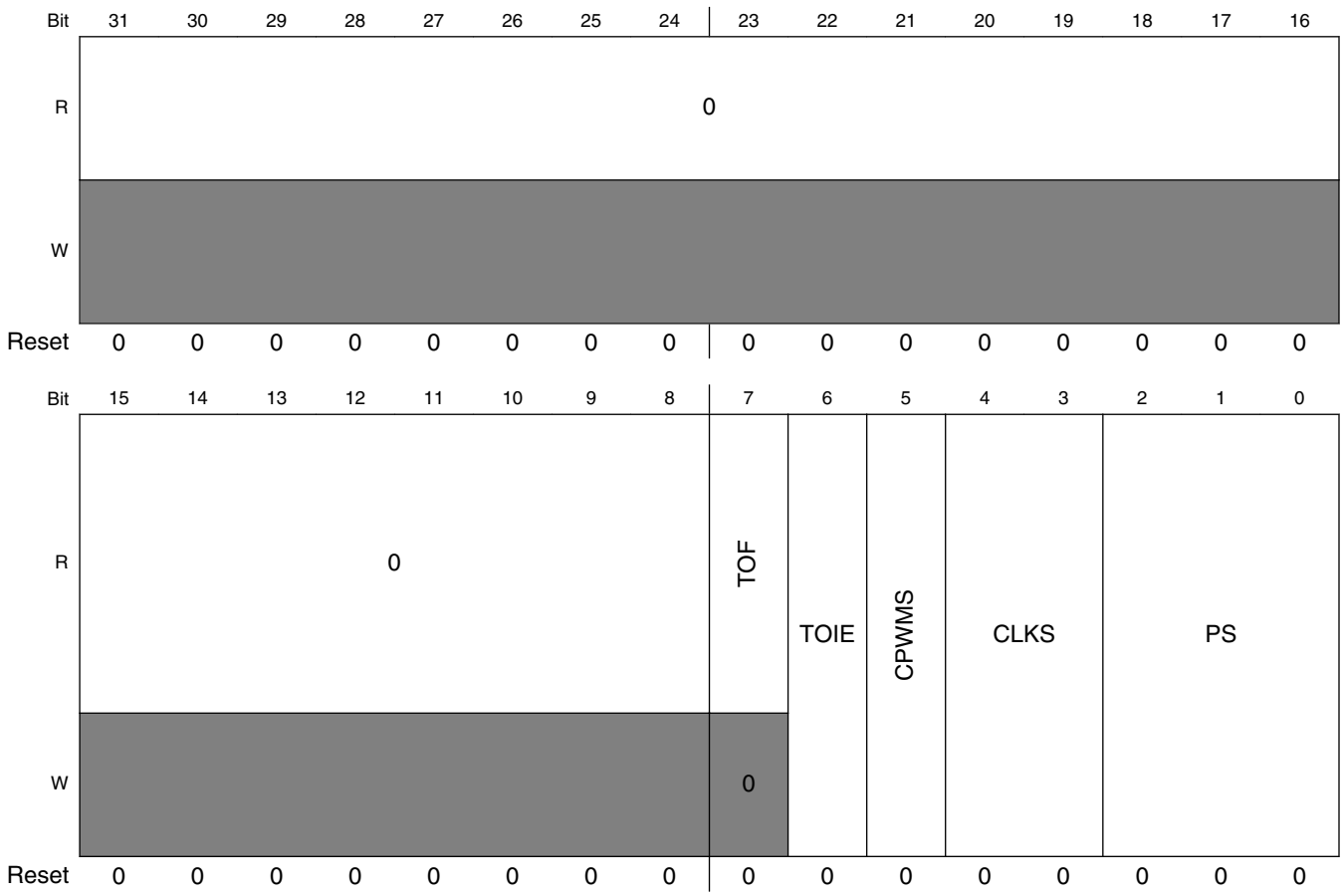
FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_9068	Deadtime Insertion Control (FTM1_DEADTIME)	32	R/W	0000_0000h	39.4.15/ 919
4003_906C	FTM External Trigger (FTM1_EXTTRIG)	32	R/W	0000_0000h	39.4.16/ 920
4003_9070	Channels Polarity (FTM1_POL)	32	R/W	0000_0000h	39.4.17/ 922
4003_9074	Fault Mode Status (FTM1_FMS)	32	R/W	0000_0000h	39.4.18/ 924
4003_9078	Input Capture Filter Control (FTM1_FILTER)	32	R/W	0000_0000h	39.4.19/ 926
4003_907C	Fault Control (FTM1_FLTCTRL)	32	R/W	0000_0000h	39.4.20/ 927
4003_9080	Quadrature Decoder Control And Status (FTM1_QDCTRL)	32	R/W	0000_0000h	39.4.21/ 930
4003_9084	Configuration (FTM1_CONF)	32	R/W	0000_0000h	39.4.22/ 932
4003_9088	FTM Fault Input Polarity (FTM1_FLTPOL)	32	R/W	0000_0000h	39.4.23/ 933
4003_908C	Synchronization Configuration (FTM1_SYNCONF)	32	R/W	0000_0000h	39.4.24/ 934
4003_9090	FTM Inverting Control (FTM1_INVCTRL)	32	R/W	0000_0000h	39.4.25/ 936
4003_9094	FTM Software Output Control (FTM1_SWOCTRL)	32	R/W	0000_0000h	39.4.26/ 937
4003_9098	FTM PWM Load (FTM1_PWMLOAD)	32	R/W	0000_0000h	39.4.27/ 940

39.4.3 Status And Control (FTMx_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, and prescaler factor. These controls relate to all channels within this module.

Address: Base address + 0h offset



FTMx_SC field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 TOF	Timer Overflow Flag Set by hardware when the FTM counter passes the value in the MOD register. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect. If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case, a TOF interrupt request is not lost due to the clearing sequence for a previous TOF. 0 FTM counter has not overflowed. 1 FTM counter has overflowed.

Table continues on the next page...

FTMx_SC field descriptions (continued)

Field	Description
6 TOIE	Timer Overflow Interrupt Enable Enables FTM overflow interrupts. 0 Disable TOF interrupts. Use software polling. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the FTM to operate in Up-Down Counting mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 FTM counter operates in Up Counting mode. 1 FTM counter operates in Up-Down Counting mode.
4–3 CLKS	Clock Source Selection Selects FTM counter clock sources. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 No clock selected. This in effect disables the FTM counter. 01 System clock 10 Reserved 11 External clock
PS	Prescale Factor Selection Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next system clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128

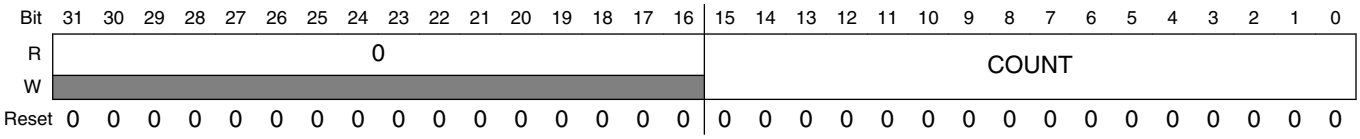
39.4.4 Counter (FTMx_CNT)

The CNT register contains the FTM counter value.

Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value, CNTIN.

When BDM is active, the FTM counter is frozen. This is the value that you may read.

Address: Base address + 4h offset



FTMx_CNT field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COUNT	Counter Value

39.4.5 Modulo (FTMx_MOD)

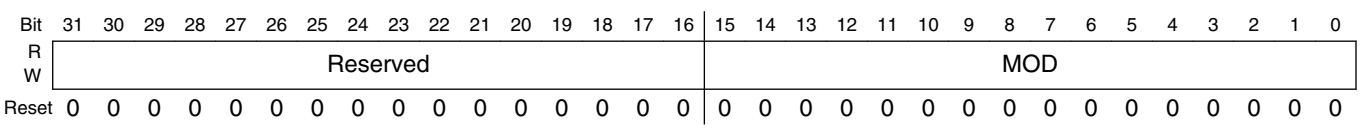
The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock, and the next value of FTM counter depends on the selected counting method; see Counter.

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to Registers updated from write buffers.

If FTMEN = 0, this write coherency mechanism may be manually reset by writing to the SC register whether BDM is active or not.

Initialize the FTM counter, by writing to CNT, before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Address: Base address + 8h offset



FTMx_MOD field descriptions

Field	Description
31–16 Reserved	This field is reserved.
MOD	Modulo Value

39.4.6 Channel (n) Status And Control (FTM*x*_C*n*SC)

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

Table 39-3. Mode, edge, and level selection

DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	X	X	XX	00	Pin not used for FTM—revert the channel pin to general purpose I/O or other peripheral control	
0	0	0	00	01	Input Capture	Capture on Rising Edge Only
				10		Capture on Falling Edge Only
				11		Capture on Rising or Falling Edge
			01	01	Output Compare	Toggle Output on match
				10		Clear Output on match
				11		Set Output on match
			1X	10	Edge-Aligned PWM	High-true pulses (clear Output on match)
				X1		Low-true pulses (set Output on match)
		1	XX	10	Center-Aligned PWM	High-true pulses (clear Output on match-up)
				X1		Low-true pulses (set Output on match-up)
	1	0	XX	10	Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
				X1		Low-true pulses (clear on channel (n) match, and set on channel (n +1) match)

Table continues on the next page...

Table 39-3. Mode, edge, and level selection (continued)

DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
1	0	0	X0	See the following table (Table 39-4).	Dual Edge Capture	One-Shot Capture mode
			X1			Continuous Capture mode

Table 39-4. Dual Edge Capture mode — edge polarity selection

ELSnB	ELSnA	Channel Port Enable	Detected Edges
0	0	Disabled	No edge
0	1	Enabled	Rising edge
1	0	Enabled	Falling edge
1	1	Enabled	Rising and falling edges

Address: Base address + Ch offset + (8d × i), where i=0d to 7d

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CHF	CHIE	MSB	MSA	ELSB	ELSA	ICRST	DMA
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_C*n*SC field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CHF	Channel Flag Set by hardware when an event occurs on the channel. CHF is cleared by reading the CSC register while CHnF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect. If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF. 0 No channel event has occurred. 1 A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0 Disable channel interrupts. Use software polling. 1 Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See Table 39-3 . This field is write protected. It can be written only when MODE[WPDIS] = 1.
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See Table 39-3 . This field is write protected. It can be written only when MODE[WPDIS] = 1.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. See Table 39-3 . This field is write protected. It can be written only when MODE[WPDIS] = 1.
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. See Table 39-3 . This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 ICRST	FTM counter reset by the selected input capture event. FTM counter reset is driven by the selected event of the channel (n) in the Input Capture mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 FTM counter is not reset when the selected channel (n) input event is detected. 1 FTM counter is reset when the selected channel (n) input event is detected.
0 DMA	DMA Enable Enables DMA transfers for the channel. 0 Disable DMA transfers. 1 Enable DMA transfers.

39.4.7 Channel (n) Value (FTMx_CnV)

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.

In Input Capture, Capture Test, and Dual Edge Capture modes, any write to a CnV register is ignored.

In output modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to Registers updated from write buffers.

If FTMEN = 0, this write coherency mechanism may be manually reset by writing to the CnSC register whether BDM mode is active or not.

Address: Base address + 10h offset + (8d × i), where i=0d to 7d

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

FTMx_CnV field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
VAL	Channel Value Captured FTM counter value of the input modes or the match value for the output modes

39.4.8 Counter Initial Value (FTMx_CNTIN)

The Counter Initial Value register contains the initial value for the FTM counter.

Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to Registers updated from write buffers.

When the FTM clock is initially selected, by writing a non-zero value to the CLKS bits, the FTM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the FTM clock, write the new value to the the CNTIN register and then initialize the FTM counter by writing any value to the CNT register.

Address: Base address + 4Ch offset

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

FTMx_CNTIN field descriptions

Field	Description
31–16 Reserved	This field is reserved.
INIT	Initial Value Of The FTM Counter

39.4.9 Capture And Compare Status (FTMx_STATUS)

The STATUS register contains a copy of the status flag CHnF bit in CnSC for each FTM channel for software convenience.

Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by reading STATUS followed by writing 0x00 to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. CHnF is cleared by reading STATUS while CHnF is set and then writing a 0 to the CHnF bit. Writing a 1 to CHnF has no effect.

If another event occurs between the read and write operations, the write operation has no effect; therefore, CHnF remains set indicating an event has occurred. In this case, a CHnF interrupt request is not lost due to the clearing sequence for a previous CHnF.

Address: Base address + 50h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_STATUS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CH7F	Channel 7 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
6 CH6F	Channel 6 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
5 CH5F	Channel 5 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
4 CH4F	Channel 4 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
3 CH3F	Channel 3 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
2 CH2F	Channel 2 Flag See the register description.

Table continues on the next page...

FTMx_STATUS field descriptions (continued)

Field	Description
	0 No channel event has occurred. 1 A channel event has occurred.
1 CH1F	Channel 1 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.

39.4.10 Features Mode Selection (FTMx_MODE)

This register contains the global enable bit for FTM-specific features and the control bits used to configure:

- Fault control mode and interrupt
- Capture Test mode
- PWM synchronization
- Write protection
- Channel output initialization

These controls relate to all channels within this module.

Address: Base address + 54h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FAULTIE	FAULTM		CAPTEST	PWMSYNC	WPDIS	INIT	FTMEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

FTMx_MODE field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 FAULTIE	Fault Interrupt Enable Enables the generation of an interrupt when a fault is detected by FTM and the FTM fault control is enabled. 0 Fault control interrupt is disabled. 1 Fault control interrupt is enabled.
6–5 FAULTM	Fault Control Mode Defines the FTM fault control mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 Fault control is disabled for all channels. 01 Fault control is enabled for even channels only (channels 0, 2, 4, and 6), and the selected mode is the manual fault clearing. 10 Fault control is enabled for all channels, and the selected mode is the manual fault clearing. 11 Fault control is enabled for all channels, and the selected mode is the automatic fault clearing.
4 CAPTEST	Capture Test Mode Enable Enables the capture test mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Capture test mode is disabled. 1 Capture test mode is enabled.
3 PWMSYNC	PWM Synchronization Mode Selects which triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. See PWM synchronization. The PWMSYNC bit configures the synchronization when SYNCMODE is 0. 0 No restrictions. Software and hardware triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. 1 Software trigger can only be used by MOD and CnV synchronization, and hardware triggers can only be used by OUTMASK and FTM counter synchronization.
2 WPDIS	Write Protection Disable When write protection is enabled (WPDIS = 0), write protected bits cannot be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN. WPDIS is set when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPDIS has no effect. 0 Write protection is enabled. 1 Write protection is disabled.
1 INIT	Initialize The Channels Output When a 1 is written to INIT bit the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect. The INIT bit is always read as 0.
0 FTMEN	FTM Enable This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_MODE field descriptions (continued)

Field	Description
	0 TPM compatibility. Free running counter and synchronization compatible with TPM. 1 Free running counter and synchronization are different from TPM behavior.

39.4.11 Synchronization (FTMx_SYNC)

This register configures the PWM synchronization.

A synchronization event can perform the synchronized update of MOD, CV, and OUTMASK registers with the value of their write buffer and the FTM counter initialization.

NOTE

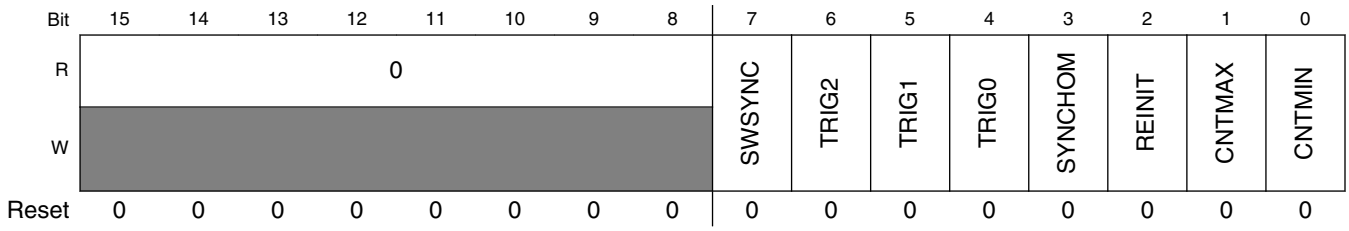
The software trigger, SWSYNC bit, and hardware triggers TRIG0, TRIG1, and TRIG2 bits have a potential conflict if used together when SYNCMODE = 0. Use only hardware or software triggers but not both at the same time, otherwise unpredictable behavior is likely to happen.

The selection of the loading point, CNTMAX and CNTMIN bits, is intended to provide the update of MOD, CNTIN, and CnV registers across all enabled channels simultaneously. The use of the loading point selection together with SYNCMODE = 0 and hardware trigger selection, TRIG0, TRIG1, or TRIG2 bits, is likely to result in unpredictable behavior.

The synchronization event selection also depends on the PWMSYNC (MODE register) and SYNCMODE (SYNCONF register) bits. See PWM synchronization.

Address: Base address + 58h offset

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



FTMx_SYNC field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 SWSYNC	PWM Synchronization Software Trigger Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit. 0 Software trigger is not selected. 1 Software trigger is selected.
6 TRIG2	PWM Synchronization Hardware Trigger 2 Enables hardware trigger 2 to the PWM synchronization. Hardware trigger 2 happens when a rising edge is detected at the trigger 2 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
5 TRIG1	PWM Synchronization Hardware Trigger 1 Enables hardware trigger 1 to the PWM synchronization. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
4 TRIG0	PWM Synchronization Hardware Trigger 0 Enables hardware trigger 0 to the PWM synchronization. Hardware trigger 0 occurs when a rising edge is detected at the trigger 0 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
3 SYNCHOM	Output Mask Synchronization Selects when the OUTMASK register is updated with the value of its buffer. 0 OUTMASK register is updated with the value of its buffer in all rising edges of the system clock. 1 OUTMASK register is updated with the value of its buffer only by the PWM synchronization.
2 REINIT	FTM Counter Reinitialization By Synchronization (FTM counter synchronization) Determines if the FTM counter is reinitialized when the selected trigger for the synchronization is detected. The REINIT bit configures the synchronization when SYNCMODE is zero. 0 FTM counter continues to count normally. 1 FTM counter is updated with its initial value when the selected trigger is detected.

Table continues on the next page...

FTMx_SYNC field descriptions (continued)

Field	Description
1 CNTMAX	Maximum Loading Point Enable Selects the maximum loading point to PWM synchronization. See Boundary cycle and loading points. If CNTMAX is 1, the selected loading point is when the FTM counter reaches its maximum value (MOD register). 0 The maximum loading point is disabled. 1 The maximum loading point is enabled.
0 CNTMIN	Minimum Loading Point Enable Selects the minimum loading point to PWM synchronization. See Boundary cycle and loading points. If CNTMIN is one, the selected loading point is when the FTM counter reaches its minimum value (CNTIN register). 0 The minimum loading point is disabled. 1 The minimum loading point is enabled.

39.4.12 Initial State For Channels Output (FTMx_OUTINIT)

Address: Base address + 5Ch offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CH7OI	CH6OI	CH5OI	CH4OI	CH3OI	CH2OI	CH1OI	CH0OI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_OUTINIT field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CH7OI	Channel 7 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
6 CH6OI	Channel 6 Output Initialization Value

Table continues on the next page...

FTMx_OUTINIT field descriptions (continued)

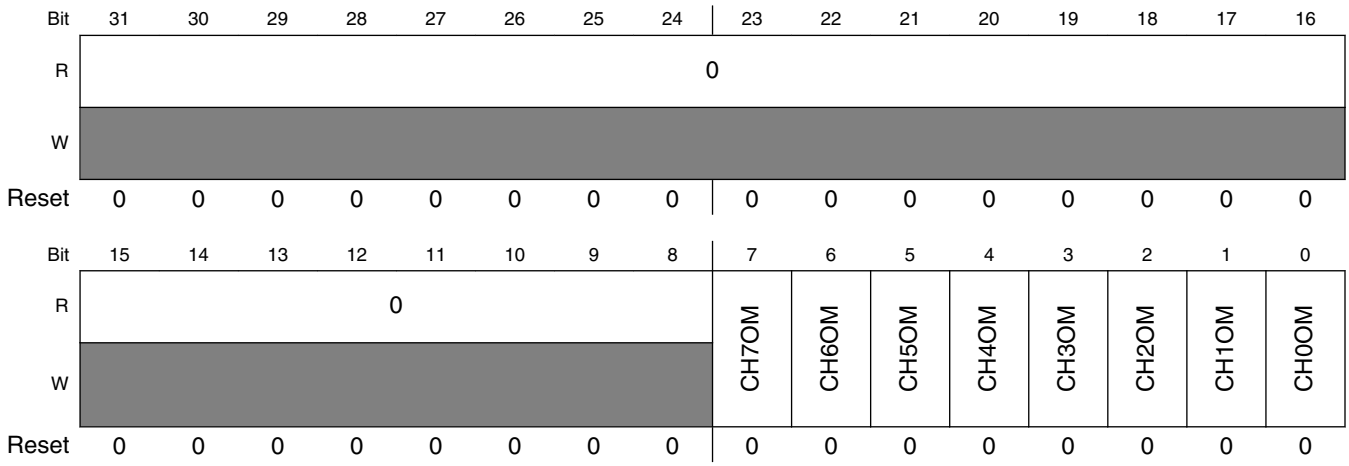
Field	Description
	Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
5 CH5OI	Channel 5 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
4 CH4OI	Channel 4 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
3 CH3OI	Channel 3 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
2 CH2OI	Channel 2 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
1 CH1OI	Channel 1 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
0 CH0OI	Channel 0 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.

39.4.13 Output Mask (FTMx_OUTMASK)

This register provides a mask for each FTM channel. The mask of a channel determines if its output responds, that is, it is masked or not, when a match occurs. This feature is used for BLDC control where the PWM signal is presented to an electric motor at specific times to provide electronic commutation.

Any write to the OUTMASK register, stores the value in its write buffer. The register is updated with the value of its write buffer according to PWM synchronization.

Address: Base address + 60h offset



FTMx_OUTMASK field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CH7OM	Channel 7 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
6 CH6OM	Channel 6 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
5 CH5OM	Channel 5 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
4 CH4OM	Channel 4 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
3 CH3OM	Channel 3 Output Mask Defines if the channel output is masked or unmasked.

Table continues on the next page...

FTMx_OUTMASK field descriptions (continued)

Field	Description
	0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
2 CH2OM	Channel 2 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
1 CH1OM	Channel 1 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
0 CH0OM	Channel 0 Output Mask Defines if the channel output is masked or unmasked. 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.

39.4.14 Function For Linked Channels (FTMx_COMBINE)

This register contains the control bits used to configure the fault control, synchronization, deadtime insertion, Dual Edge Capture mode, Complementary, and Combine mode for each pair of channels (n) and (n+1), where n equals 0, 2, 4, and 6.

Address: Base address + 64h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	FAULTEN3	SYNCEN3	DTEN3	DECAP3	DECAPEN3	COMP3	COMBINE3	0	FAULTEN2	SYNCEN2	DTEN2	DECAP2	DECAPEN2	COMP2	COMBINE2
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	FAULTEN1	SYNCEN1	DTEN1	DECAP1	DECAPEN1	COMP1	COMBINE1	0	FAULTEN0	SYNCEN0	DTEN0	DECAP0	DECAPEN0	COMP0	COMBINE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_COMBINE field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 FAULTEN3	Fault Control Enable For n = 6 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
29 SYNCEN3	Synchronization Enable For n = 6 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
28 DTEN3	Deadtime Enable For n = 6 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
27 DECAP3	Dual Edge Capture Mode Captures For n = 6 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
26 DECAPEN3	Dual Edge Capture Mode Enable For n = 6 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to Table 39-3. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.
25 COMP3	Complement Of Channel (n) for n = 6 Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
24 COMBINE3	Combine Channels For n = 6 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 FAULTEN2	Fault Control Enable For n = 4 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
21 SYNCEN2	Synchronization Enable For n = 4 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
20 DTEN2	Deadtime Enable For n = 4 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
19 DECAP2	Dual Edge Capture Mode Captures For n = 4 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
18 DECAPEN2	Dual Edge Capture Mode Enable For n = 4 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to Table 39-3. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.
17 COMP2	Complement Of Channel (n) For n = 4

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
16 COMBINE2	Combine Channels For n = 4 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 FAULTEN1	Fault Control Enable For n = 2 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
13 SYNCEN1	Synchronization Enable For n = 2 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
12 DTEN1	Deadtime Enable For n = 2 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
11 DECAP1	Dual Edge Capture Mode Captures For n = 2 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if Dual Edge Capture – One-Shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
10 DECAPEN1	Dual Edge Capture Mode Enable For n = 2 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to Table 39-3.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.
9 COMP1	Complement Of Channel (n) For n = 2 Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
8 COMBINE1	Combine Channels For n = 2 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FAULTEN0	Fault Control Enable For n = 0 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
5 SYNCEN0	Synchronization Enable For n = 0 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
4 DTEN0	Deadtime Enable For n = 0 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
3 DECAP0	Dual Edge Capture Mode Captures For n = 0 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
2 DECAPEN0	Dual Edge Capture Mode Enable For n = 0 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to Table 39-3. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.
1 COMP0	Complement Of Channel (n) For n = 0 Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
0 COMBINE0	Combine Channels For n = 0 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.

39.4.15 Deadtime Insertion Control (FTMx_DEADTIME)

This register selects the deadtime prescaler factor and deadtime value. All FTM channels use this clock prescaler and this deadtime value for the deadtime insertion.

Address: Base address + 68h offset

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

FTMx_DEADTIME field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 DTPS	Deadtime Prescaler Value Selects the division factor of the system clock. This prescaled clock is used by the deadtime counter. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0x Divide the system clock by 1.

Table continues on the next page...

FTMx_DEADTIME field descriptions (continued)

Field	Description
	10 Divide the system clock by 4. 11 Divide the system clock by 16.
DTVAL	Deadtime Value Selects the deadtime insertion value for the deadtime counter. The deadtime counter is clocked by a scaled version of the system clock. See the description of DTPS. Deadtime insert value = (DTPS × DTVAL). DTVAL selects the number of deadtime counts inserted as follows: When DTVAL is 0, no counts are inserted. When DTVAL is 1, 1 count is inserted. When DTVAL is 2, 2 counts are inserted. This pattern continues up to a possible 63 counts. This field is write protected. It can be written only when MODE[WPDIS] = 1.

39.4.16 FTM External Trigger (FTMx_EXTTRIG)

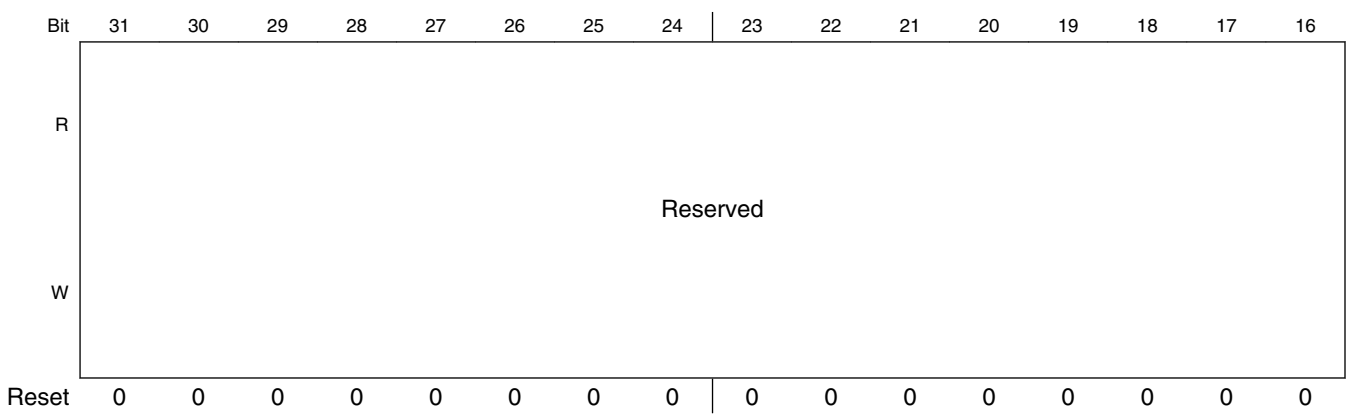
This register:

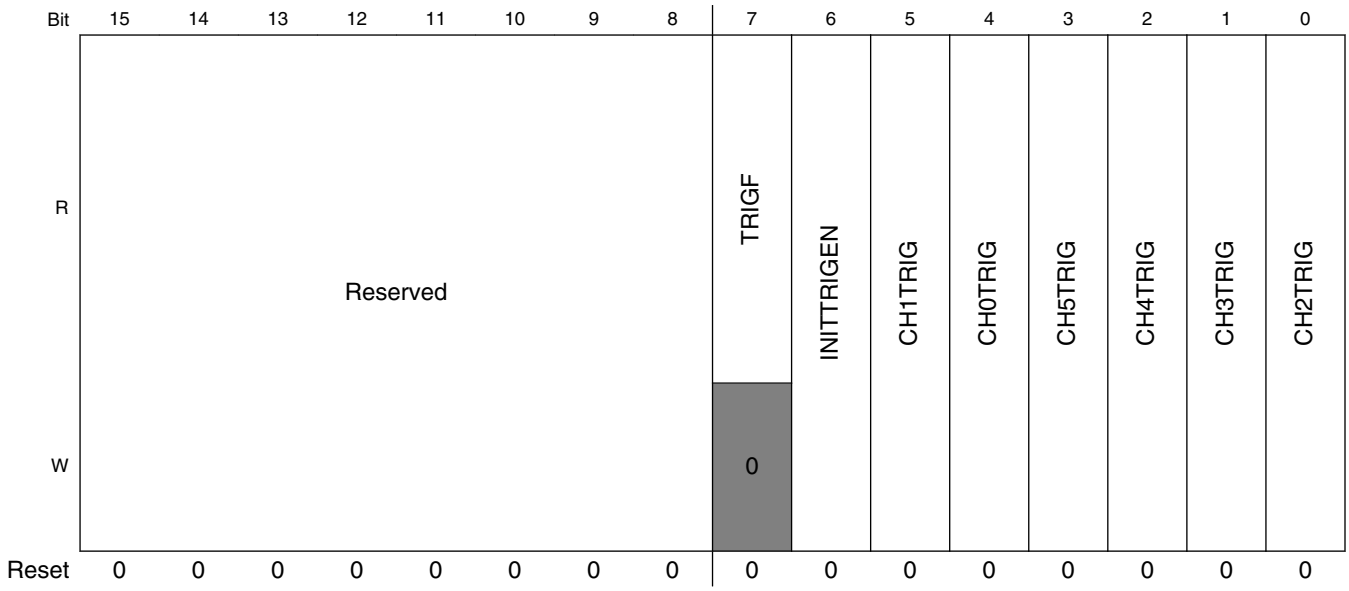
- Indicates when a channel trigger was generated
- Enables the generation of a trigger when the FTM counter is equal to its initial value
- Selects which channels are used in the generation of the channel triggers

Several channels can be selected to generate multiple triggers in one PWM period. See Channel trigger output and Initialization trigger.

Channels 6 and 7 are not used to generate channel triggers.

Address: Base address + 6Ch offset





FTMx_EXTTRIG field descriptions

Field	Description
31–8 Reserved	This field is reserved.
7 TRIGF	Channel Trigger Flag Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. Writing a 1 to TRIGF has no effect. If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF. 0 No channel trigger was generated. 1 A channel trigger was generated.
6 INITTRIGEN	Initialization Trigger Enable Enables the generation of the trigger when the FTM counter is equal to the CNTIN register. 0 The generation of initialization trigger is disabled. 1 The generation of initialization trigger is enabled.
5 CH1TRIG	Channel 1 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
4 CH0TRIG	Channel 0 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
3 CH5TRIG	Channel 5 Trigger Enable

Table continues on the next page...

FTMx_EXTTRIG field descriptions (continued)

Field	Description
	Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
2 CH4TRIG	Channel 4 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
1 CH3TRIG	Channel 3 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
0 CH2TRIG	Channel 2 Trigger Enable Enables the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.

39.4.17 Channels Polarity (FTMx_POL)

This register defines the output polarity of the FTM channels.

NOTE

The safe value that is driven in a channel output when the fault control is enabled and a fault condition is detected is the inactive state of the channel. That is, the safe value of a channel is the value of its POL bit.

Address: Base address + 70h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_POL field descriptions

Field	Description
31–8 Reserved	This field is reserved.
7 POL7	Channel 7 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
6 POL6	Channel 6 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
5 POL5	Channel 5 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
4 POL4	Channel 4 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
3 POL3	Channel 3 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
2 POL2	Channel 2 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
1 POL1	Channel 1 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_POL field descriptions (continued)

Field	Description
	0 The channel polarity is active high. 1 The channel polarity is active low.
0 POL0	Channel 0 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.

39.4.18 Fault Mode Status (FTMx_FMS)

This register contains the fault detection flags, write protection enable bit, and the logic OR of the enabled fault inputs.

Address: Base address + 74h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FAULTF	WPEN	FAULTIN	0	FAULTF3	FAULTF2	FAULTF1	FAULTF0
W									0				0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FMS field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 FAULTF	Fault Detection Flag Represents the logic OR of the individual FAULTFj bits where j = 3, 2, 1, 0. Clear FAULTF by reading the FMS register while FAULTF is set and then writing a 0 to FAULTF while there is no existing fault condition at the enabled fault inputs. Writing a 1 to FAULTF has no effect. If another fault condition is detected in an enabled fault input before the clearing sequence is completed, the sequence is reset so FAULTF remains set after the clearing sequence is completed for the earlier fault condition. FAULTF is also cleared when FAULTFj bits are cleared individually. 0 No fault condition was detected. 1 A fault condition was detected.
6 WPEN	Write Protection Enable The WPEN bit is the negation of the WPDIS bit. WPEN is set when 1 is written to it. WPEN is cleared when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPEN has no effect. 0 Write protection is disabled. Write protected bits can be written. 1 Write protection is enabled. Write protected bits cannot be written.
5 FAULTIN	Fault Inputs Represents the logic OR of the enabled fault inputs after their filter (if their filter is enabled) when fault control is enabled. 0 The logic OR of the enabled fault inputs is 0. 1 The logic OR of the enabled fault inputs is 1.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FAULTF3	Fault Detection Flag 3 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF3 by reading the FMS register while FAULTF3 is set and then writing a 0 to FAULTF3 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF3 has no effect. FAULTF3 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF3 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
2 FAULTF2	Fault Detection Flag 2 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF2 by reading the FMS register while FAULTF2 is set and then writing a 0 to FAULTF2 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF2 has no effect. FAULTF2 bit is also cleared when FAULTF bit is cleared.

Table continues on the next page...

FTMx_FMS field descriptions (continued)

Field	Description
	If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF2 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
1 FAULTF1	Fault Detection Flag 1 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF1 by reading the FMS register while FAULTF1 is set and then writing a 0 to FAULTF1 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF1 has no effect. FAULTF1 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF1 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
0 FAULTF0	Fault Detection Flag 0 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF0 by reading the FMS register while FAULTF0 is set and then writing a 0 to FAULTF0 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF0 has no effect. FAULTF0 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF0 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.

39.4.19 Input Capture Filter Control (FTMx_FILTER)

This register selects the filter value for the inputs of channels.

Channels 4, 5, 6 and 7 do not have an input filter.

NOTE

Writing to the FILTER register has immediate effect and must be done only when the channels 0, 1, 2, and 3 are not in input modes. Failure to do this could result in a missing valid signal.

Address: Base address + 78h offset

Bit	31–16	15–12	11–8	7–4	3–0
R W	Reserved	CH3FVAL	CH2FVAL	CH1FVAL	CH0FVAL
Reset	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0

FTMx_FILTER field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15–12 CH3FVAL	Channel 3 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
11–8 CH2FVAL	Channel 2 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
7–4 CH1FVAL	Channel 1 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
CH0FVAL	Channel 0 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

39.4.20 Fault Control (FTMx_FLTCTRL)

This register selects the filter value for the fault inputs, enables the fault inputs and the fault inputs filter.

Address: Base address + 7Ch offset

Bit	31–24	23–16
R	0	
W		
Reset	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0

Bit	15–12	11–8	7	6	5	4	3	2	1	0
R	0	FFVAL	FFLTR3EN	FFLTR2EN	FFLTR1EN	FFLTR0EN	FAULT3EN	FAULT2EN	FAULT1EN	FAULT0EN
W										
Reset	0 0 0 0	0 0 0 0	0	0	0	0	0	0	0	0

FTMx_FLTCTRL field descriptions

Field	Description
31–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 FFVAL	Fault Input Filter Selects the filter value for the fault inputs. The fault filter is disabled when the value is zero. **NOTE:** Writing to this field has immediate effect and must be done only when the fault control or all fault inputs are disabled. Failure to do this could result in a missing fault detection.
7 FFLTR3EN	Fault Input 3 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
6 FFLTR2EN	Fault Input 2 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
5 FFLTR1EN	Fault Input 1 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
4 FFLTR0EN	Fault Input 0 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
3 FAULT3EN	Fault Input 3 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
2 FAULT2EN	Fault Input 2 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.

Table continues on the next page...

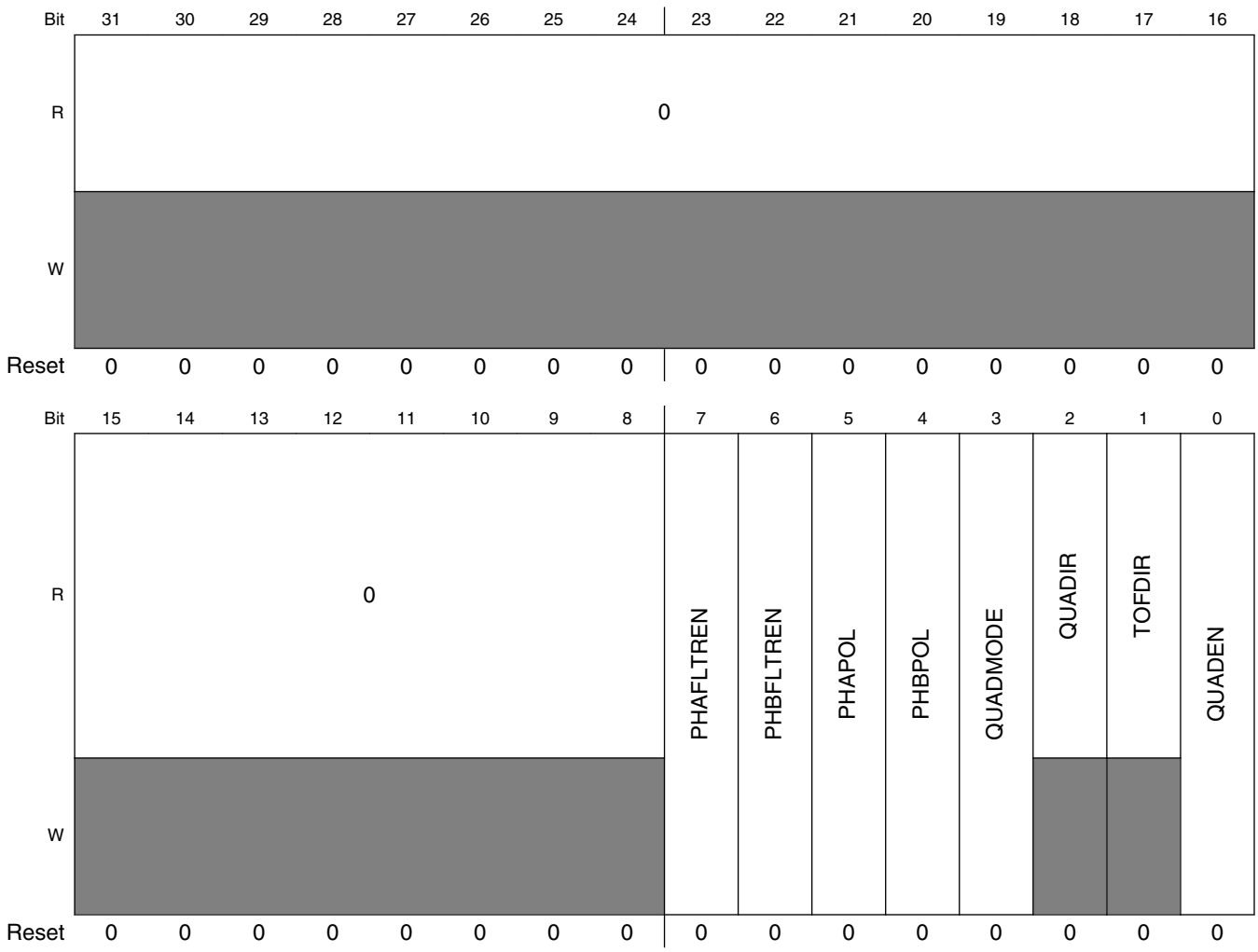
FTMx_FLTCTRL field descriptions (continued)

Field	Description
1 FAULT1EN	Fault Input 1 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
0 FAULT0EN	Fault Input 0 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.

39.4.21 Quadrature Decoder Control And Status (FTMx_QDCTRL)

This register has the control and status bits for the Quadrature Decoder mode.

Address: Base address + 80h offset



FTMx_QDCTRL field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 PHAFLTREN	Phase A Input Filter Enable Enables the filter for the quadrature decoder phase A input. The filter value for the phase A input is defined by the CH0FVAL field of FILTER. The phase A filter is also disabled when CH0FVAL is zero. 0 Phase A input filter is disabled. 1 Phase A input filter is enabled.

Table continues on the next page...

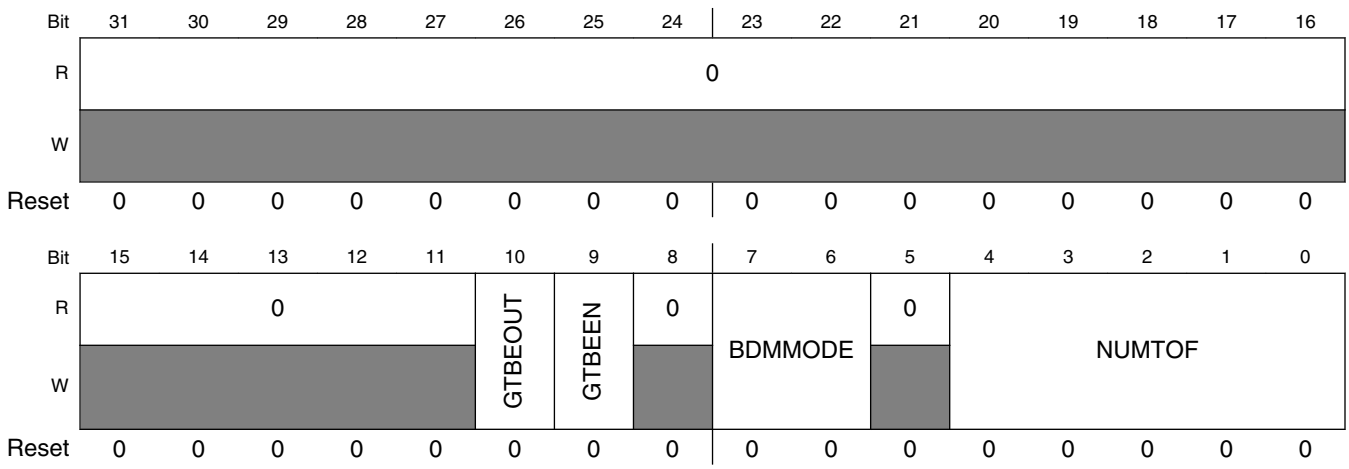
FTMx_QDCTRL field descriptions (continued)

Field	Description
6 PHBFLTREN	Phase B Input Filter Enable Enables the filter for the quadrature decoder phase B input. The filter value for the phase B input is defined by the CH1FVAL field of FILTER. The phase B filter is also disabled when CH1FVAL is zero. 0 Phase B input filter is disabled. 1 Phase B input filter is enabled.
5 PHAPOL	Phase A Input Polarity Selects the polarity for the quadrature decoder phase A input. 0 Normal polarity. Phase A input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase A input signal is inverted before identifying the rising and falling edges of this signal.
4 PHBPOL	Phase B Input Polarity Selects the polarity for the quadrature decoder phase B input. 0 Normal polarity. Phase B input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase B input signal is inverted before identifying the rising and falling edges of this signal.
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the Quadrature Decoder mode. 0 Phase A and phase B encoding mode. 1 Count and direction encoding mode.
2 QUADIR	FTM Counter Direction In Quadrature Decoder Mode Indicates the counting direction. 0 Counting direction is decreasing (FTM counter decrement). 1 Counting direction is increasing (FTM counter increment).
1 TOFDIR	Timer Overflow Direction In Quadrature Decoder Mode Indicates if the TOF bit was set on the top or the bottom of counting. 0 TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (CNTIN register) to its maximum value (MOD register). 1 TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (CNTIN register).
0 QUADEN	Quadrature Decoder Mode Enable Enables the Quadrature Decoder mode. In this mode, the phase A and B input signals control the FTM counter direction. The Quadrature Decoder mode has precedence over the other modes. See Table 39-3. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Quadrature Decoder mode is disabled. 1 Quadrature Decoder mode is enabled.

39.4.22 Configuration (FTMx_CONF)

This register selects the number of times that the FTM counter overflow should occur before the TOF bit to be set, the FTM behavior in BDM modes, the use of an external global time base, and the global time base signal generation.

Address: Base address + 84h offset



FTMx_CONF field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 GTBEOUT	Global Time Base Output Enables the global time base signal generation to other FTMs. 0 A global time base signal generation is disabled. 1 A global time base signal generation is enabled.
9 GTBEEN	Global Time Base Enable Configures the FTM to use an external global time base signal that is generated by another FTM. 0 Use of an external global time base is disabled. 1 Use of an external global time base is enabled.
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–6 BDMMODE	BDM Mode Selects the FTM behavior in BDM mode. See BDM mode.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
NUMTOF	TOF Frequency

Table continues on the next page...

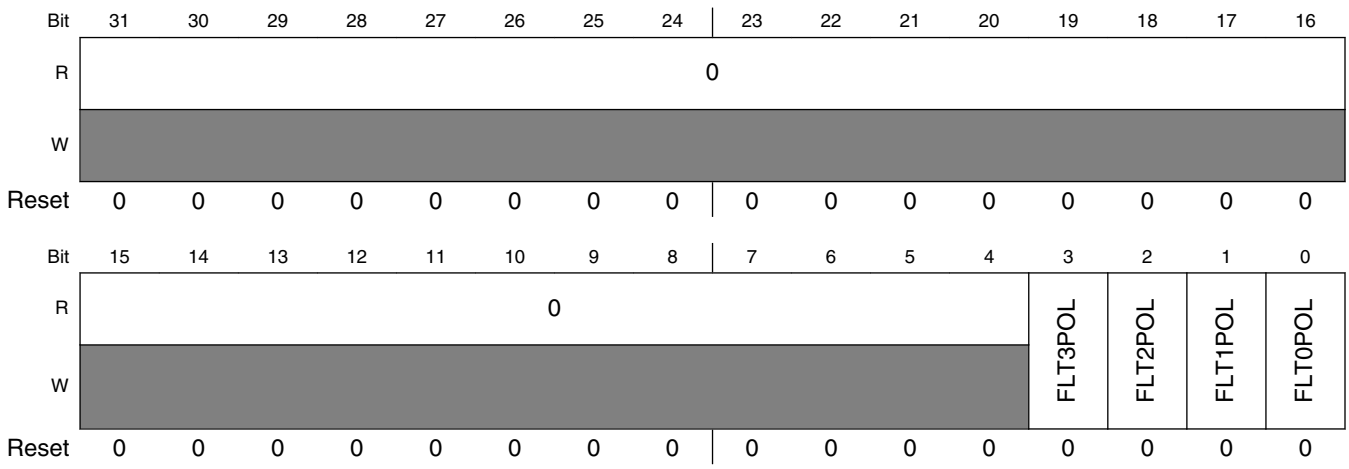
FTMx_CONF field descriptions (continued)

Field	Description
	Selects the ratio between the number of counter overflows to the number of times the TOF bit is set. NUMTOF = 0: The TOF bit is set for each counter overflow. NUMTOF = 1: The TOF bit is set for the first counter overflow but not for the next overflow. NUMTOF = 2: The TOF bit is set for the first counter overflow but not for the next 2 overflows. NUMTOF = 3: The TOF bit is set for the first counter overflow but not for the next 3 overflows. This pattern continues up to a maximum of 31.

39.4.23 FTM Fault Input Polarity (FTMx_FLTPOL)

This register defines the fault inputs polarity.

Address: Base address + 88h offset



FTMx_FLTPOL field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FLT3POL	Fault Input 3 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.
2 FLT2POL	Fault Input 2 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_FLTPOL field descriptions (continued)

Field	Description
	0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.
1 FLT1POL	Fault Input 1 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.
0 FLT0POL	Fault Input 0 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.

39.4.24 Synchronization Configuration (FTMx_SYNCONF)

This register selects the PWM synchronization configuration, SWOCTRL, INVCTRL and CNTIN registers synchronization, if FTM clears the TRIGj bit, where j = 0, 1, 2, when the hardware trigger j is detected.

Address: Base address + 8Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											HWSOC	HWINVC	HWOM	HWWRBUF	HWRSTCNT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SWSOC	SWINVC	SWOM	SWWRBUF	SWRSTCNT	SYNCMODE	0	SWOC	INVC	0	CNTINC	0	HWTRIGMODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SYNCONF field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

FTMx_SYNCONF field descriptions (continued)

Field	Description
20 HWSOC	Software output control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the SWOCTRL register synchronization. 1 A hardware trigger activates the SWOCTRL register synchronization.
19 HWINVC	Inverting control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the INVCTRL register synchronization. 1 A hardware trigger activates the INVCTRL register synchronization.
18 HWOM	Output mask synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the OUTMASK register synchronization. 1 A hardware trigger activates the OUTMASK register synchronization.
17 HWWRBUF	MOD, CNTIN, and CV registers synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate MOD, CNTIN, and CV registers synchronization. 1 A hardware trigger activates MOD, CNTIN, and CV registers synchronization.
16 HWRSTCNT	FTM counter synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the FTM counter synchronization. 1 A hardware trigger activates the FTM counter synchronization.
15–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 SWSOC	Software output control synchronization is activated by the software trigger. 0 The software trigger does not activate the SWOCTRL register synchronization. 1 The software trigger activates the SWOCTRL register synchronization.
11 SWINVC	Inverting control synchronization is activated by the software trigger. 0 The software trigger does not activate the INVCTRL register synchronization. 1 The software trigger activates the INVCTRL register synchronization.
10 SWOM	Output mask synchronization is activated by the software trigger. 0 The software trigger does not activate the OUTMASK register synchronization. 1 The software trigger activates the OUTMASK register synchronization.
9 SWWRBUF	MOD, CNTIN, and CV registers synchronization is activated by the software trigger. 0 The software trigger does not activate MOD, CNTIN, and CV registers synchronization. 1 The software trigger activates MOD, CNTIN, and CV registers synchronization.
8 SWRSTCNT	FTM counter synchronization is activated by the software trigger. 0 The software trigger does not activate the FTM counter synchronization. 1 The software trigger activates the FTM counter synchronization.
7 SYNCMODE	Synchronization Mode Selects the PWM Synchronization mode. 0 Legacy PWM synchronization is selected. 1 Enhanced PWM synchronization is selected.

Table continues on the next page...

FTMx_SYNCONF field descriptions (continued)

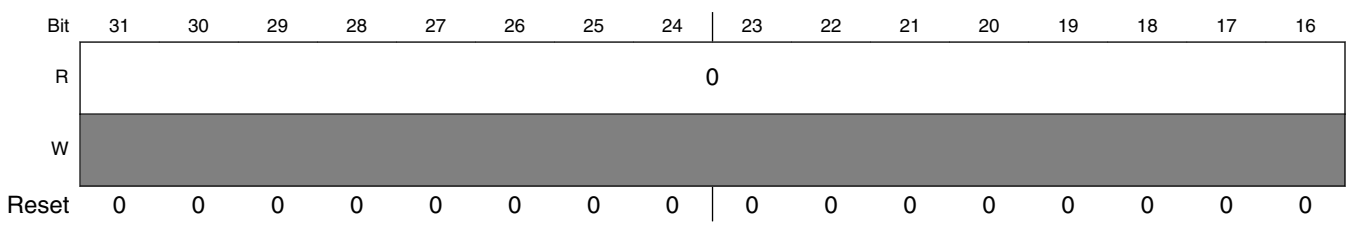
Field	Description
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 SWOC	SWOCTRL Register Synchronization 0 SWOCTRL register is updated with its buffer value at all rising edges of system clock. 1 SWOCTRL register is updated with its buffer value by the PWM synchronization.
4 INVC	INVCTRL Register Synchronization 0 INVCTRL register is updated with its buffer value at all rising edges of system clock. 1 INVCTRL register is updated with its buffer value by the PWM synchronization.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CNTINC	CNTIN Register Synchronization 0 CNTIN register is updated with its buffer value at all rising edges of system clock. 1 CNTIN register is updated with its buffer value by the PWM synchronization.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HWTRIGMODE	Hardware Trigger Mode 0 FTM clears the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2. 1 FTM does not clear the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2.

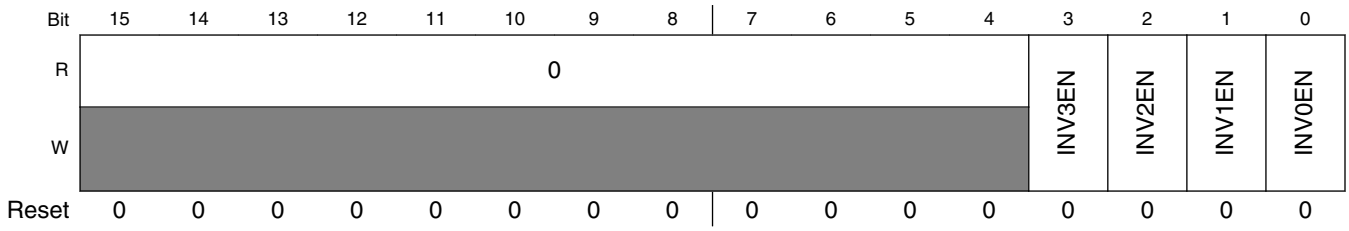
39.4.25 FTM Inverting Control (FTMx_INVCTRL)

This register controls when the channel (n) output becomes the channel (n+1) output, and channel (n+1) output becomes the channel (n) output. Each INVmEN bit enables the inverting operation for the corresponding pair channels m.

This register has a write buffer. The INVmEN bit is updated by the INVCTRL register synchronization.

Address: Base address + 90h offset





FTMx_INVCTRL field descriptions

Field	Description
31–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 INV3EN	Pair Channels 3 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
2 INV2EN	Pair Channels 2 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
1 INV1EN	Pair Channels 1 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
0 INV0EN	Pair Channels 0 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.

39.4.26 FTM Software Output Control (FTMx_SWOCTRL)

This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CHnOC bits enable the control of the corresponding channel (n) output by software.
- The CHnOCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

Address: Base address + 94h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH7OCV	CH6OCV	CH5OCV	CH4OCV	CH3OCV	CH2OCV	CH1OCV	CH0OCV	CH7OC	CH6OC	CH5OC	CH4OC	CH3OC	CH2OC	CH1OC	CH0OC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SWOCTRL field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 CH7OCV	Channel 7 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
14 CH6OCV	Channel 6 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
13 CH5OCV	Channel 5 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
12 CH4OCV	Channel 4 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
11 CH3OCV	Channel 3 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
10 CH2OCV	Channel 2 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
9 CH1OCV	Channel 1 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
8 CH0OCV	Channel 0 Software Output Control Value

Table continues on the next page...

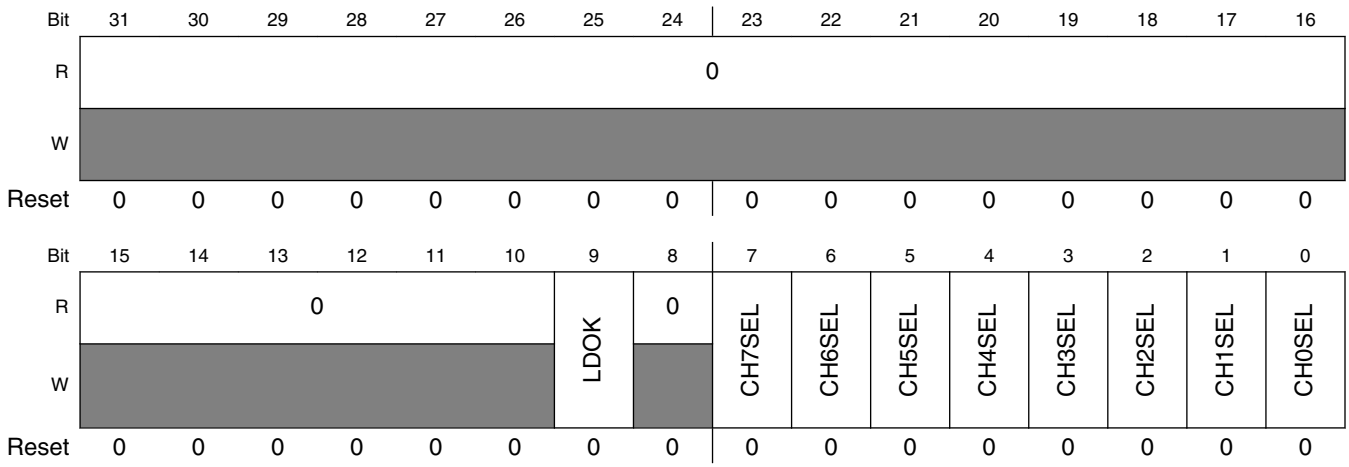
FTMx_SWOCTRL field descriptions (continued)

Field	Description
	0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
7 CH7OC	Channel 7 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
6 CH6OC	Channel 6 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
5 CH5OC	Channel 5 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
4 CH4OC	Channel 4 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
3 CH3OC	Channel 3 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
2 CH2OC	Channel 2 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
1 CH1OC	Channel 1 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
0 CH0OC	Channel 0 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.

39.4.27 FTM PWM Load (FTMx_PWMLOAD)

Enables the loading of the MOD, CNTIN, C(n)V, and C(n+1)V registers with the values of their write buffers when the FTM counter changes from the MOD register value to its next value or when a channel (j) match occurs. A match occurs for the channel (j) when FTM counter = C(j)V.

Address: Base address + 98h offset



FTMx_PWMLOAD field descriptions

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 LDOK	Load Enable Enables the loading of the MOD, CNTIN, and CV registers with the values of their write buffers. 0 Loading updated values is disabled. 1 Loading updated values is enabled.
8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 CH7SEL	Channel 7 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
6 CH6SEL	Channel 6 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
5 CH5SEL	Channel 5 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.

Table continues on the next page...

FTMx_PWMLOAD field descriptions (continued)

Field	Description
4 CH4SEL	Channel 4 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
3 CH3SEL	Channel 3 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
2 CH2SEL	Channel 2 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
1 CH1SEL	Channel 1 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
0 CH0SEL	Channel 0 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.

39.5 Functional description

The notation used in this document to represent the counters and the generation of the signals is shown in the following figure.

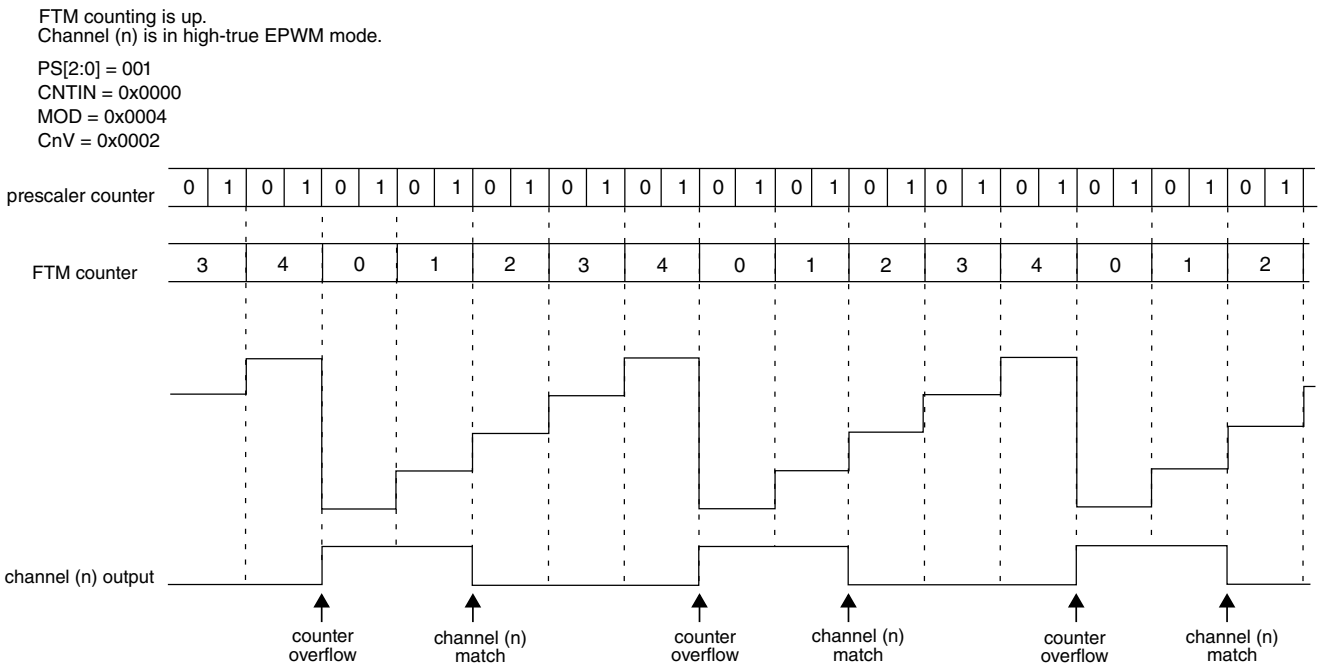


Figure 39-3. Notation used

39.5.1 Clock source

The FTM has only one clock domain: the system clock.

39.5.1.1 Counter clock source

The CLKS[1:0] bits in the SC register selects clock sources for the FTM counter or disables the FTM counter. After any chip reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The external clock passes through a synchronizer clocked by the system clock to assure that counter transitions are properly aligned to system clock transitions.Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

39.5.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and FTM counter.

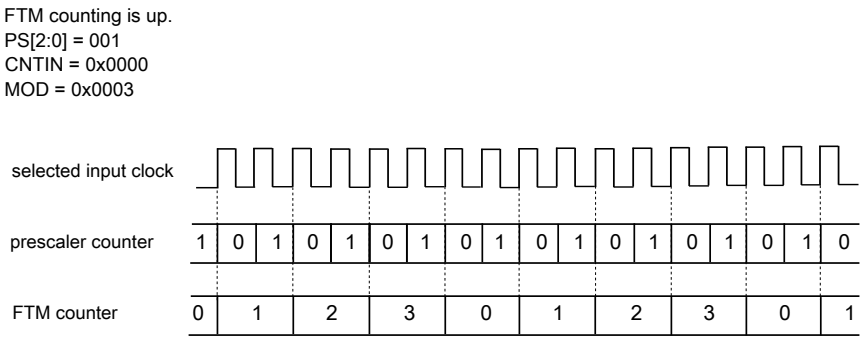


Figure 39-4. Example of the prescaler counter

39.5.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler.

The FTM counter has these modes of operation:

- Up counting
- Up-down counting
- Quadrature Decoder mode

39.5.3.1 Up counting

Up counting is selected when:

- QUADEN = 0, and
- CPWMS = 0

CNTIN defines the starting value of the count and MOD defines the final value of the count, see the following figure. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.

The FTM period when using up counting is (MOD – CNTIN + 0x0001) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to CNTIN.

FTM counting is up.
CNTIN = 0xFFFC (in two's complement is equal to -4)
MOD = 0x0004

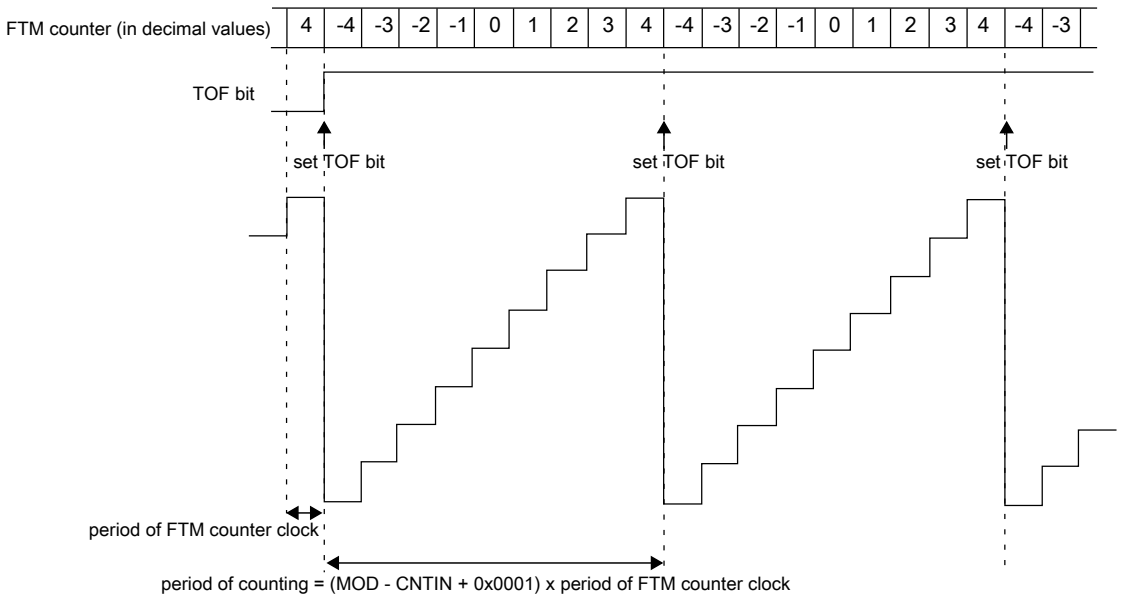


Figure 39-5. Example of FTM up and signed counting

Table 39-5. FTM counting based on CNTIN value

When	Then
CNTIN = 0x0000	The FTM counting is equivalent to TPM up counting, that is, up and unsigned counting. See the following figure.
CNTIN[15] = 1	The initial value of the FTM counter is a negative number in two's complement, so the FTM counting is up and signed.
CNTIN[15] = 0 and CNTIN ≠ 0x0000	The initial value of the FTM counter is a positive number, so the FTM counting is up and unsigned.

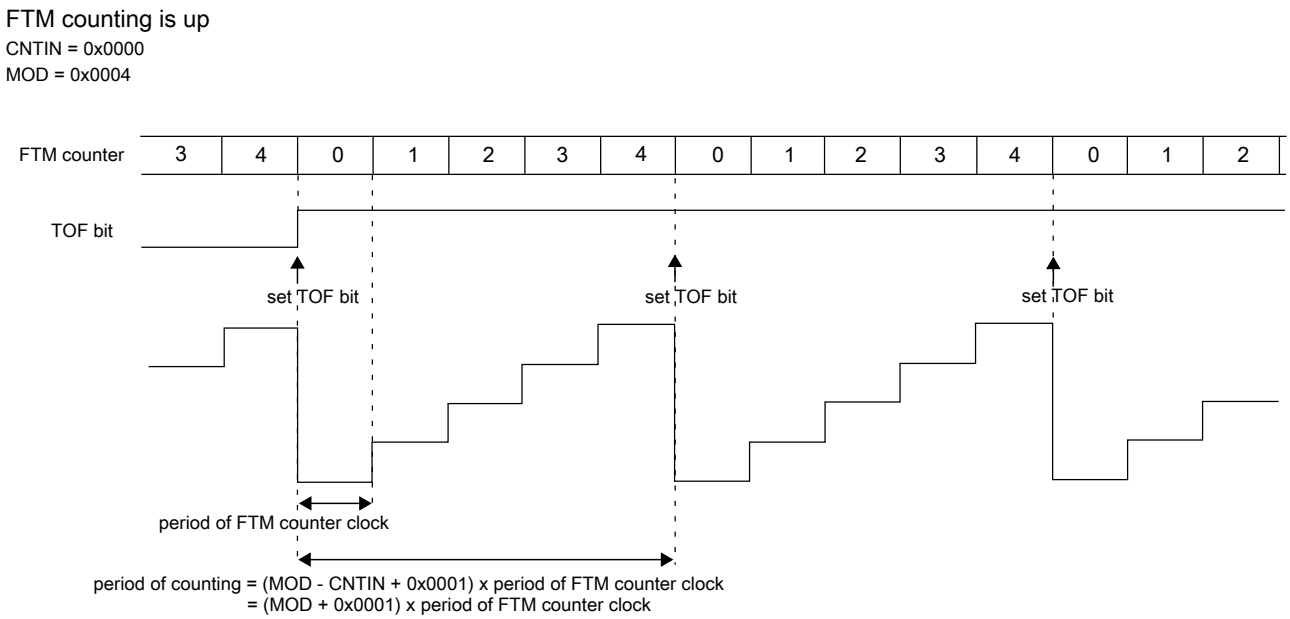


Figure 39-6. Example of FTM up counting with CNTIN = 0x0000

Note

- FTM operation is only valid when the value of the CNTIN register is less than the value of the MOD register, either in the unsigned counting or signed counting. It is the responsibility of the software to ensure that the values in the CNTIN and MOD registers meet this requirement. Any values of CNTIN and MOD that do not satisfy this criteria can result in unpredictable behavior.
- MOD = CNTIN is a redundant condition. In this case, the FTM counter is always equal to MOD and the TOF bit is set in each rising edge of the FTM counter clock.
- When MOD = 0x0000, CNTIN = 0x0000, for example after reset, and FTMEN = 1, the FTM counter remains stopped at 0x0000 until a non-zero value is written into the MOD or CNTIN registers.
- Setting CNTIN to be greater than the value of MOD is not recommended as this unusual setting may make the FTM operation difficult to comprehend. However, there is no restriction on this configuration, and an example is shown in the following figure.

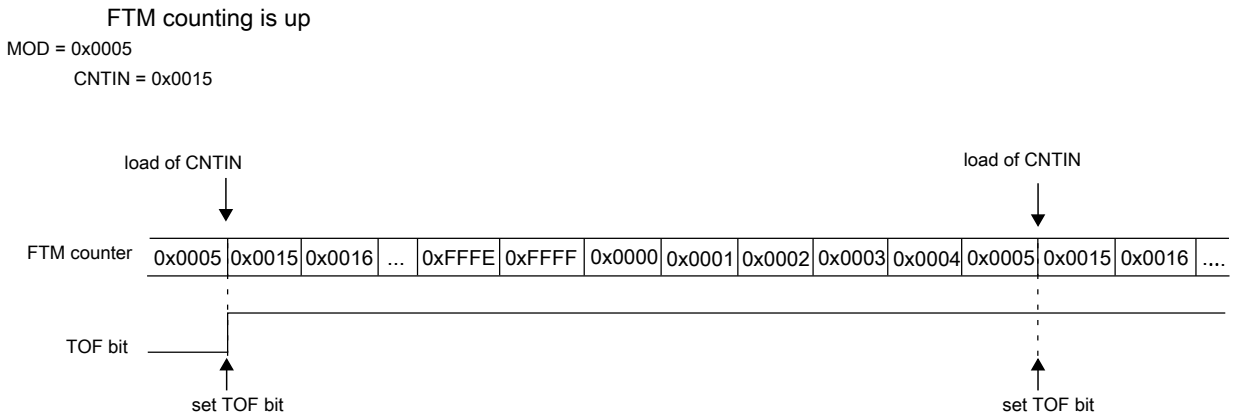


Figure 39-7. Example of up counting when the value of CNTIN is greater than the value of MOD

39.5.3.2 Up-down counting

Up-down counting is selected when:

- QUADEN = 0, and
- CPWMS = 1

CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The FTM period when using up-down counting is 2 × (MOD – CNTIN) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to (MOD – 1).

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up-down counting, that is, up-down and unsigned counting. See the following figure.

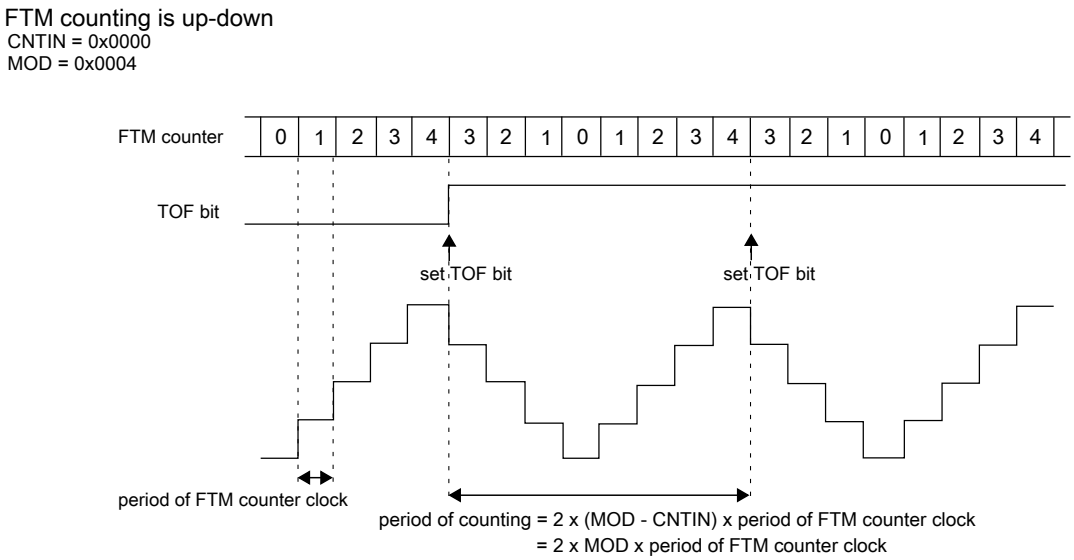


Figure 39-8. Example of up-down counting when CNTIN = 0x0000

Note

When CNTIN is different from zero in the up-down counting, a valid CPWM signal is generated:

- if CnV > CNTIN, or
- if CnV = 0 or if CnV[15] = 1. In this case, 0% CPWM is generated.

39.5.3.3 Free running counter

If (FTMEN = 0) and (MOD = 0x0000 or MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000. See the following figure.

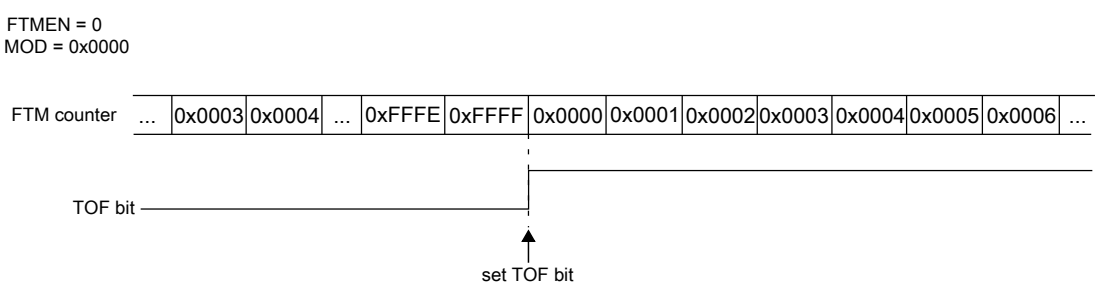


Figure 39-9. Example when the FTM counter is free running

The FTM counter is also a free running counter when:

- FTMEN = 1
- QUADEN = 0
- CPWMS = 0
- CNTIN = 0x0000, and
- MOD = 0xFFFF

39.5.3.4 Counter reset

Any one of the following cases resets the FTM counter to the value in the CNTIN register and the channels output to its initial value, except for channels in Output Compare mode.

- Any write to CNT.
- FTM counter synchronization.
- A channel in Input Capture mode with ICRST = 1 (FTM Counter Reset in Input Capture Mode).

39.5.3.5 When the TOF bit is set

The NUMTOF[4:0] bits define the number of times that the FTM counter overflow should occur before the TOF bit to be set. If NUMTOF[4:0] = 0x00, then the TOF bit is set at each FTM counter overflow.

Initialize the FTM counter, by writing to CNT, after writing to the NUMTOF[4:0] bits to avoid confusion about when the first counter overflow will occur.

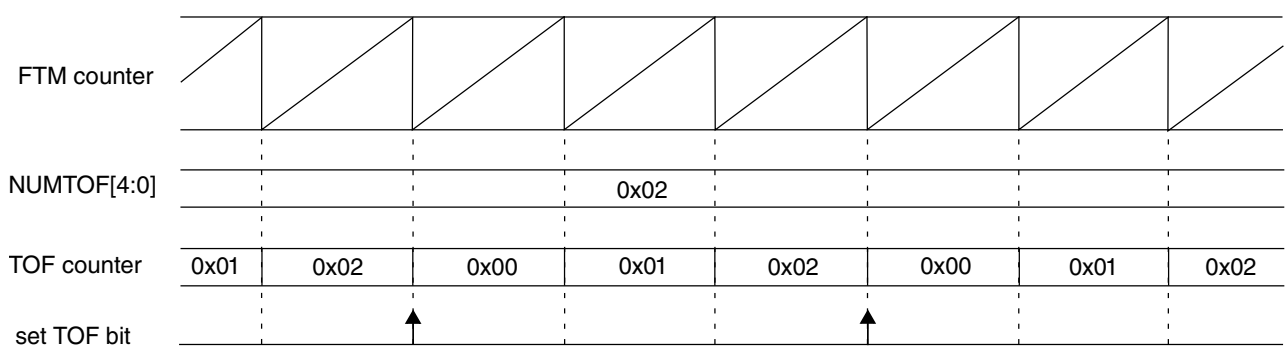


Figure 39-10. Periodic TOF when NUMTOF = 0x02

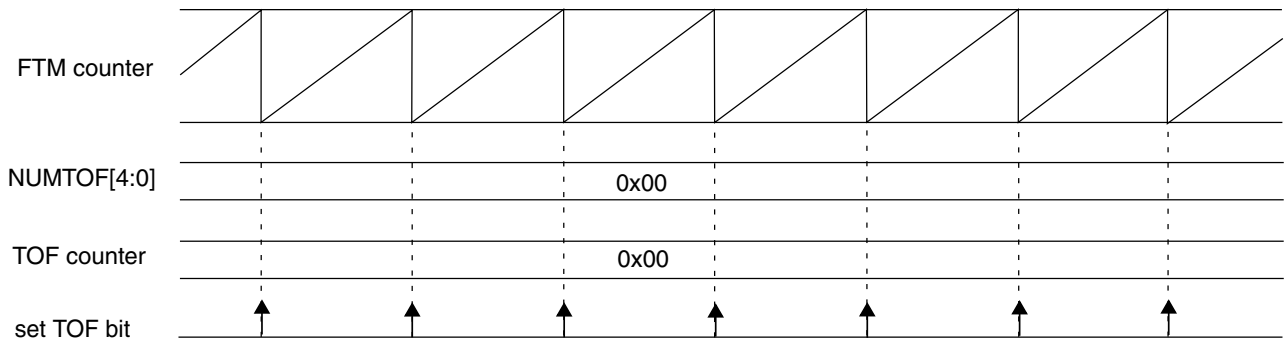


Figure 39-11. Periodic TOF when NUMTOF = 0x00

39.5.4 Input Capture mode

The Input Capture mode is selected when:

- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0
- MSnB:MSnA = 0:0, and
- ELSnB:ELSnA ≠ 0:0

When a selected edge occurs on the channel input, the current value of the FTM counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1. See the following figure.

When a channel is configured for input capture, the FTMxCHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is system clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register is ignored in Input Capture mode.

While in BDM, the input capture function works as configured. When a selected edge event occurs, the FTM counter value, which is frozen because of BDM, is captured into the CnV register and the CHnF bit is set.

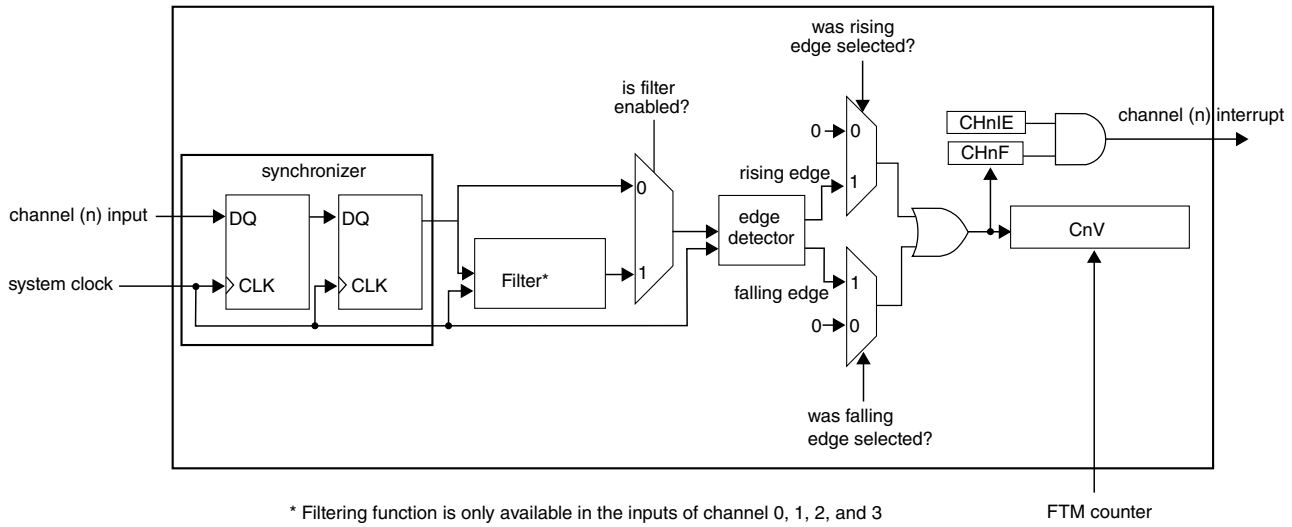


Figure 39-12. Input Capture mode

If the channel input does not have a filter enabled, then the input signal is always delayed 3 rising edges of the system clock, that is, two rising edges to the synchronizer plus one more rising edge to the edge detector. In other words, the CHnF bit is set on the third rising edge of the system clock after a valid edge occurs on the channel input.

39.5.4.1 Filter for Input Capture mode

The filter function is only available on channels 0, 1, 2, and 3.

First, the input signal is synchronized by the system clock. Following synchronization, the input signal enters the filter block. See the following figure.

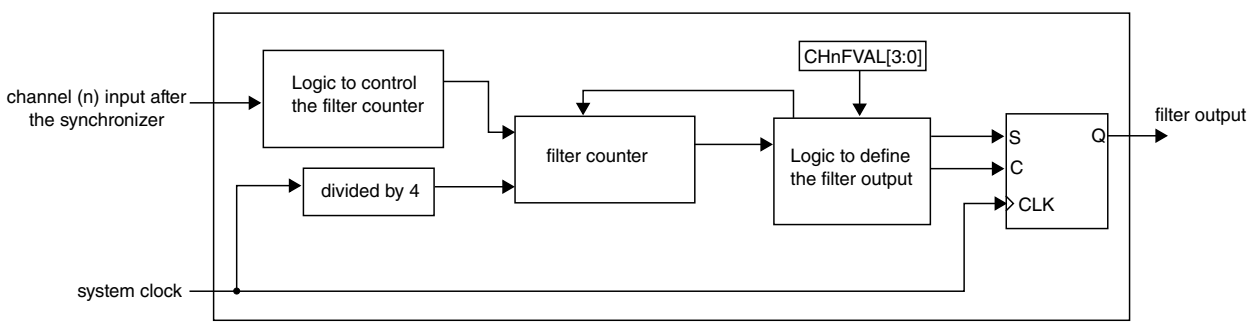


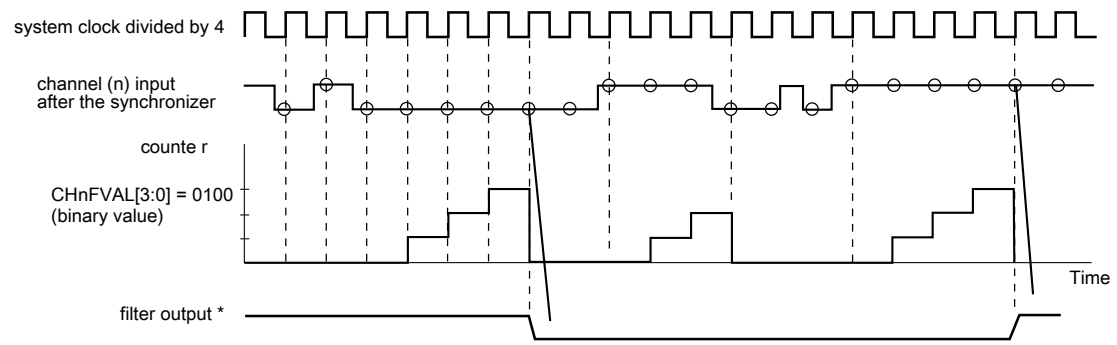
Figure 39-13. Channel input filter

When there is a state change in the input signal, the counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to CHnFVAL[3:0], the state change of the input signal is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by CHnFVAL[3:0] (× 4 system clocks) is regarded as a glitch and is not passed on to the edge detector. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed 3 rising edges of the system clock. If (CHnFVAL[3:0] ≠ 0000), then the input signal is delayed by the minimum pulse width (CHnFVAL[3:0] × 4 system clocks) plus a further 4 rising edges of the system clock: two rising edges to the synchronizer, one rising edge to the filter output, plus one more to the edge detector. In other words, CHnF is set (4 + 4 × CHnFVAL[3:0]) system clock periods after a valid edge occurs on the channel input.

The clock for the counter in the channel input filter is the system clock divided by 4.



* Note: Filter output is delayed one system clock of filter counter logic output.

Figure 39-14. Channel input filter example

The figure below shows an example of input capture with filter enabled and the delay added by each part of the input capture logic. Note that the input signal is delayed only by the synchronizer and edge dector logic if the filter is disabled.

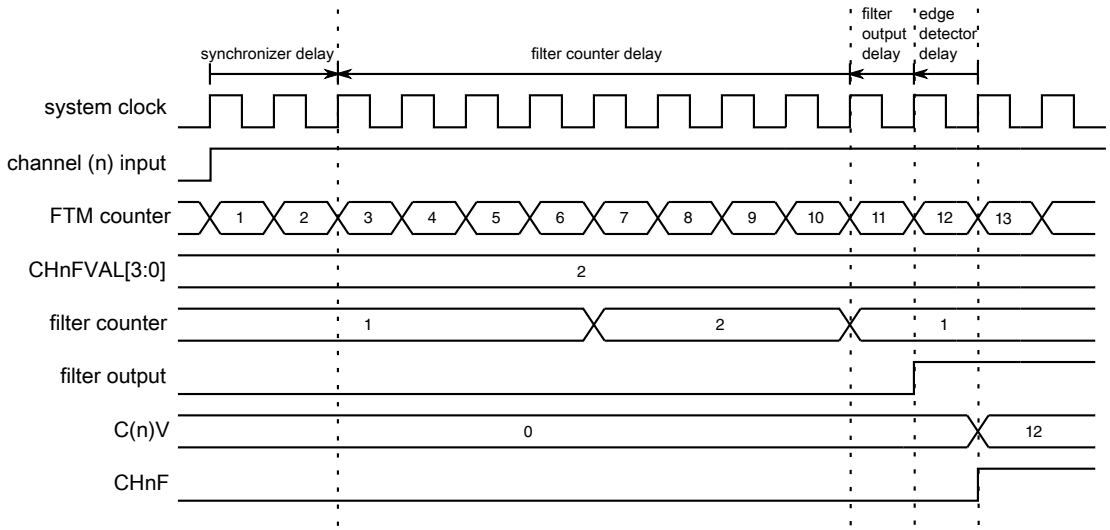


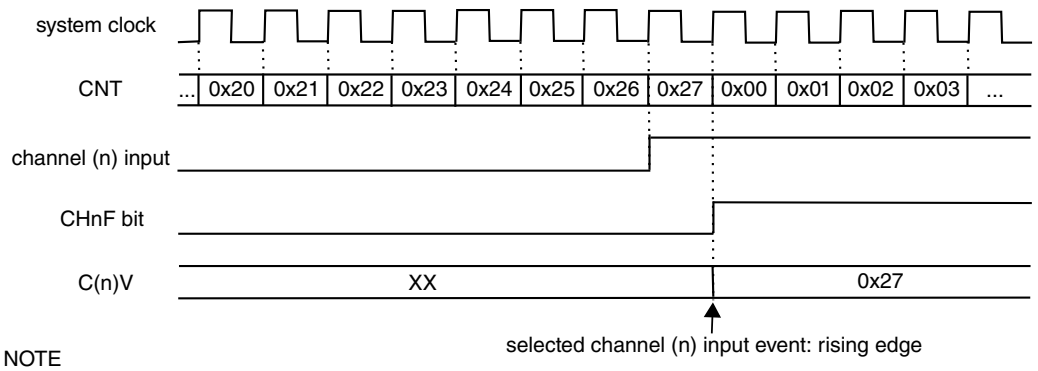
Figure 39-15. Input capture example

39.5.4.2 FTM Counter Reset in Input Capture Mode

If the channel (n) is in input capture mode and FTMx_CnSC [ICRST = 1], then when the selected input capture event occurs in the channel (n) input signal, the current value of the FTM counter is captured into the CnV register, the CHnF bit is set, the channel (n) interrupt is generated (if CHnIE = 1) and the FTM counter is reset to the CNTIN register value.

This allows the FTM to measure a period/pulse being applied to FTM_CHn (counts of the FTM clock input) without having to implement a subtraction calculation in software subsequent to the event occurring.

The figure below shows the FTM counter reset when the selected input capture event is detected in a channel in input capture mode with ICRST = 1.



NOTE
Channel (n) input after its synchronizer and filter
MOD = 0xFFFF
CNTIN = 0x0000
PS[2:0] = 3'b000
ICRST = 1'b1

Figure 39-16. Example of the Input Capture mode with ICRST = 1

NOTE

- It is expected that the ICRST bit be set only when the channel is in input capture mode.
- In this case, if the FTM counter is reset, then the prescaler counter (Prescaler) and the TOF counter (When the TOF bit is set) also are reset.

39.5.5 Output Compare mode

The Output Compare mode is selected when:

- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0, and
- MSnB:MSnA = 0:1

In Output Compare mode, the FTM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled.

When a channel is initially configured to Toggle mode, the previous value of the channel output is held until the first output compare event occurs.

The CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1 at the channel (n) match (FTM counter = CnV).

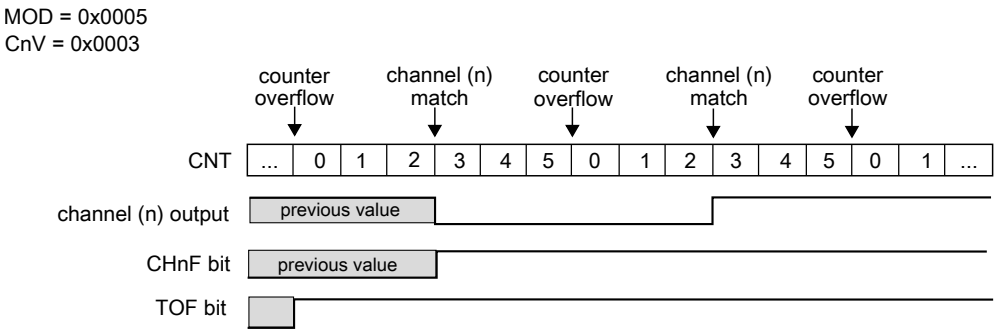


Figure 39-17. Example of the Output Compare mode when the match toggles the channel output

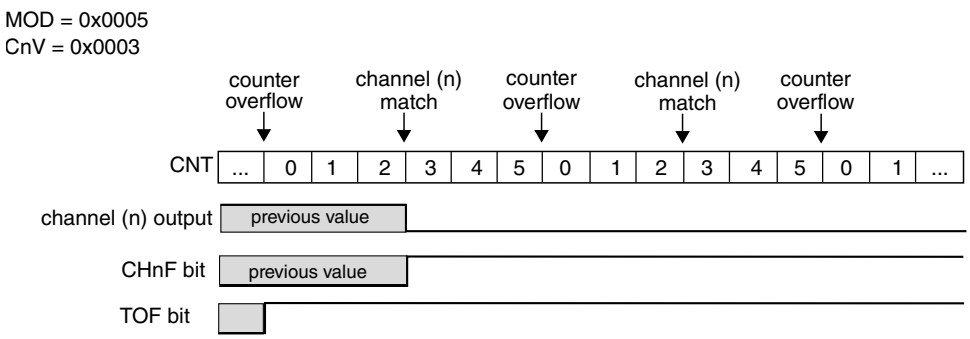


Figure 39-18. Example of the Output Compare mode when the match clears the channel output

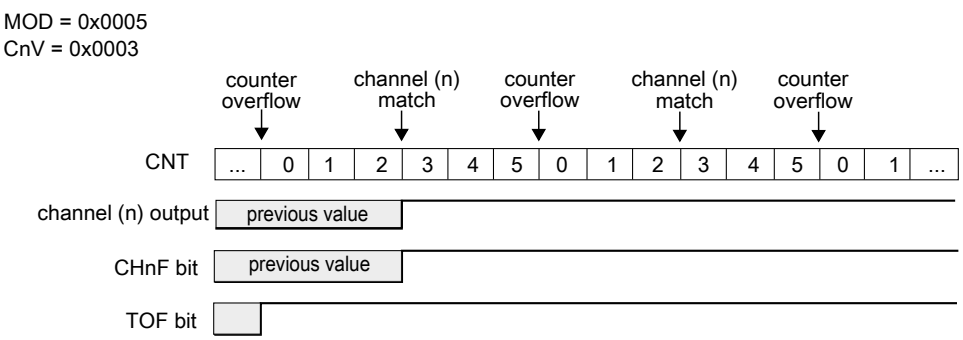


Figure 39-19. Example of the Output Compare mode when the match sets the channel output

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1, however the channel (n) output is not modified and controlled by FTM.

39.5.6 Edge-Aligned PWM (EPWM) mode

The Edge-Aligned mode is selected when:

- QUADEN = 0

- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0, and
- MSnB = 1

The EPWM period is determined by (MOD − CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV − CNTIN).

The CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1 at the channel (n) match (FTM counter = CnV), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.

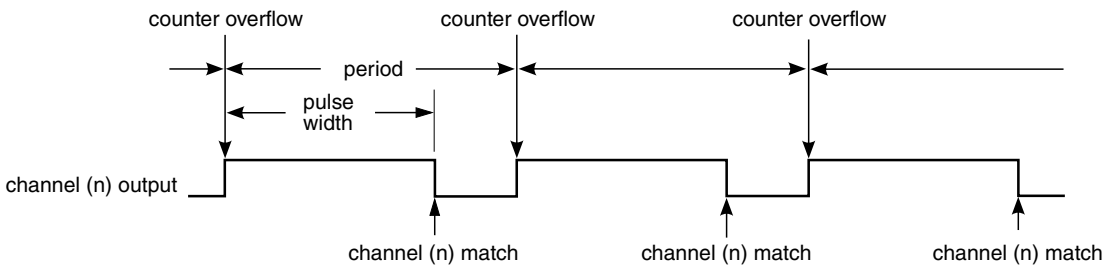


Figure 39-20. EPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1, however the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced low at the channel (n) match (FTM counter = CnV). See the following figure.

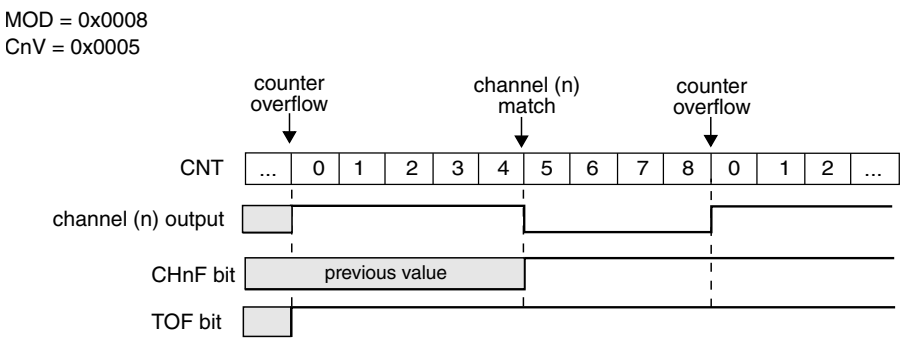


Figure 39-21. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced high at the channel (n) match (FTM counter = CnV). See the following figure.

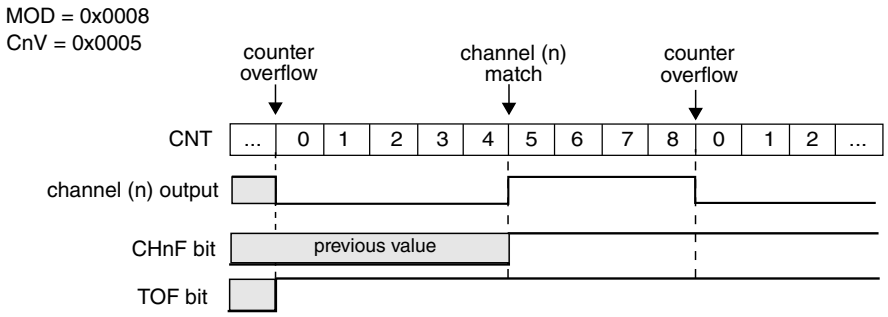


Figure 39-22. EPWM signal with ELSnB:ELSnA = X:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHnF bit is not set even when there is the channel (n) match.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set.Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

Note

When CNTIN is different from zero the following EPWM signals can be generated:

- 0% EPWM signal if CnV = CNTIN,
- EPWM signal between 0% and 100% if CNTIN < CnV <= MOD,
- 100% EPWM signal when CNTIN > CnV or CnV > MOD.

39.5.7 Center-Aligned PWM (CPWM) mode

The Center-Aligned mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 0, and
- CPWMS = 1

The CPWM pulse width (duty cycle) is determined by 2 × (CnV − CNTIN) and the period is determined by 2 × (MOD − CNTIN). See the following figure. MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the FTM counter counts up until it reaches MOD and then counts down until it reaches CNTIN.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnV) when the FTM counting is down (at the begin of the pulse width) and when the FTM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are aligned with the value of CNTIN.

The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).

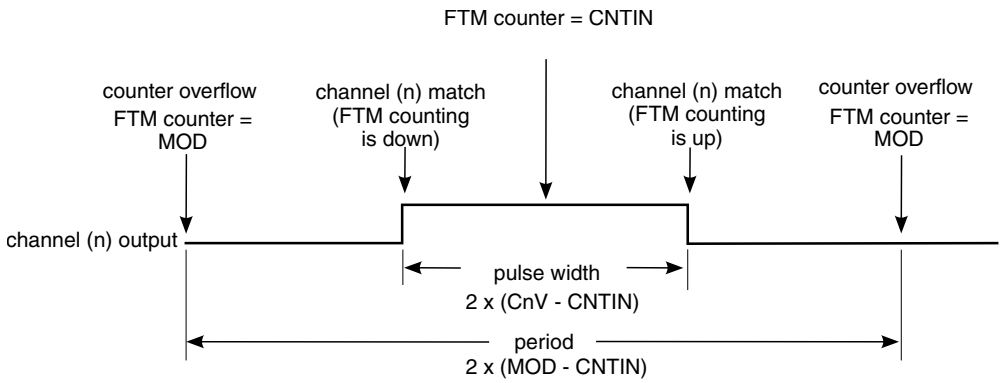


Figure 39-23. CPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the FTM counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. See the following figure.

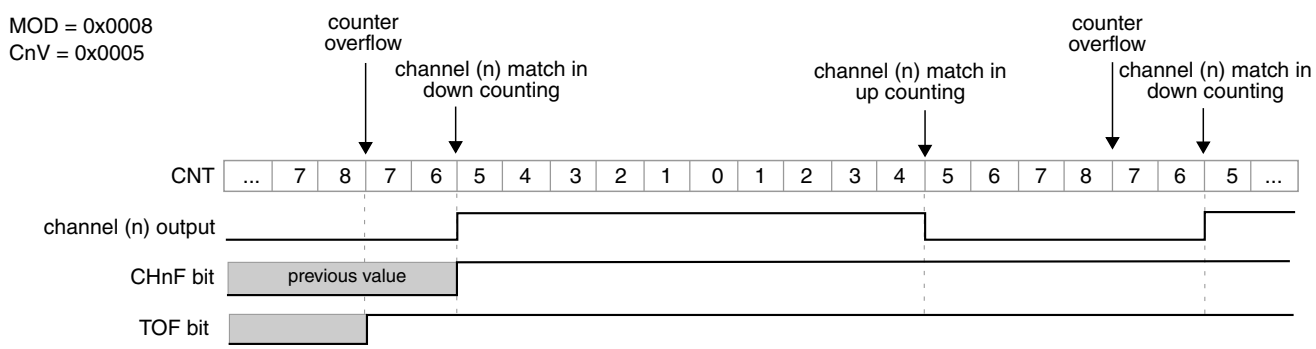


Figure 39-24. CPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. See the following figure.

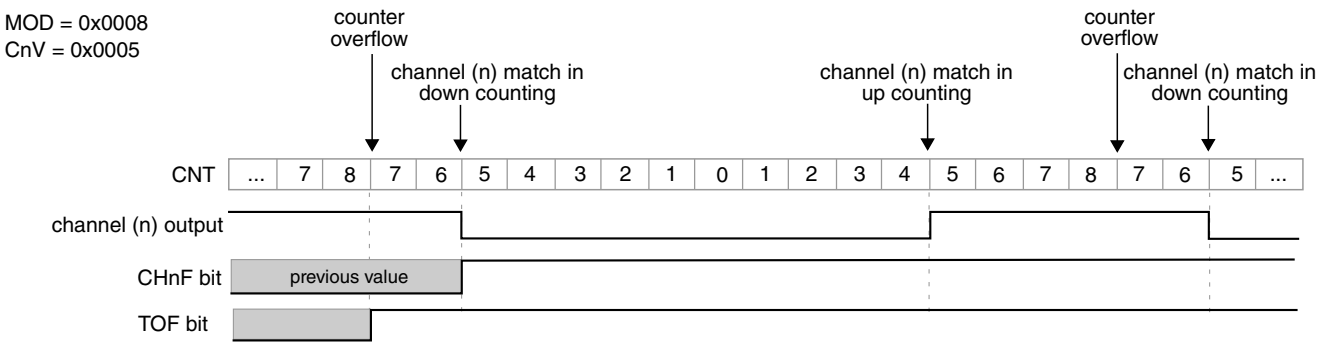


Figure 39-25. CPWM signal with ELSnB:ELSnA = X:1

If (CnV = 0x0000) or CnV is a negative value, that is (CnV[15] = 1), then the channel (n) output is a 0% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match.

If CnV is a positive value, that is (CnV[15] = 0), (CnV ≥ MOD), and (MOD ≠ 0x0000), then the channel (n) output is a 100% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MOD is 0x0001 through 0x7FFE, 0x7FFF if you do not need to generate a 100% duty cycle CPWM signal. This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

39.5.8 Combine mode

The Combine mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 1, and
- CPWMS = 0

In Combine mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

In the Combine mode, the PWM period is determined by (MOD − CNTIN + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = C(n)V). The CH(n+1)F bit is set and the channel (n +1) interrupt is generated, if CH(n+1)IE = 1, at the channel (n+1) match (FTM counter = C(n+1)V).

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n +1)V). It is forced high at the channel (n) match (FTM counter = C(n)V). See the following figure.

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced high at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n +1)V). It is forced low at the channel (n) match (FTM counter = C(n)V). See the following figure.

In Combine mode, the ELS(n+1)B and ELS(n+1)A bits are not used in the generation of the channels (n) and (n+1) output. However, if (ELSnB:ELSnA = 0:0) then the channel (n) output is not controlled by FTM, and if (ELS(n+1)B:ELS(n+1)A = 0:0) then the channel (n+1) output is not controlled by FTM.

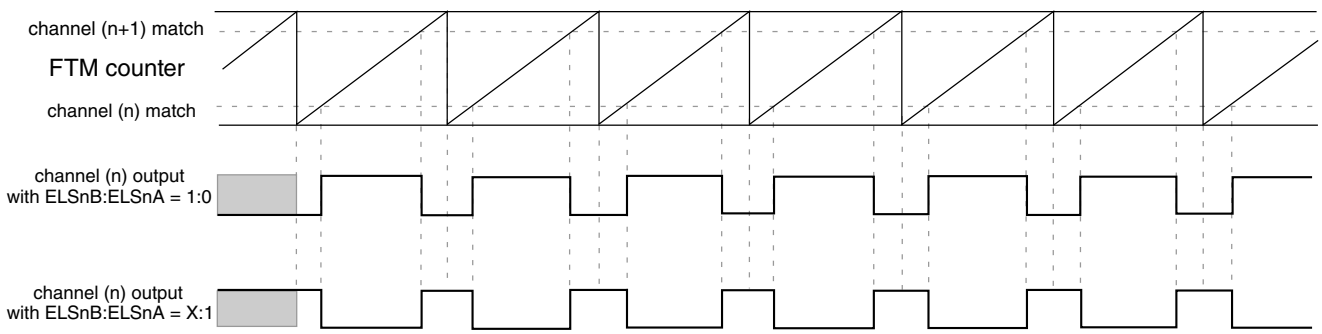


Figure 39-26. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

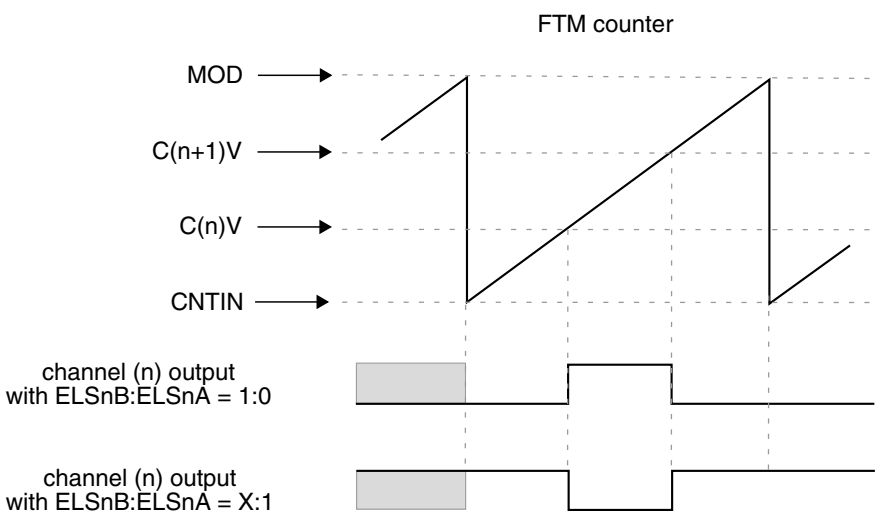


Figure 39-27. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V < C(n+1)V)

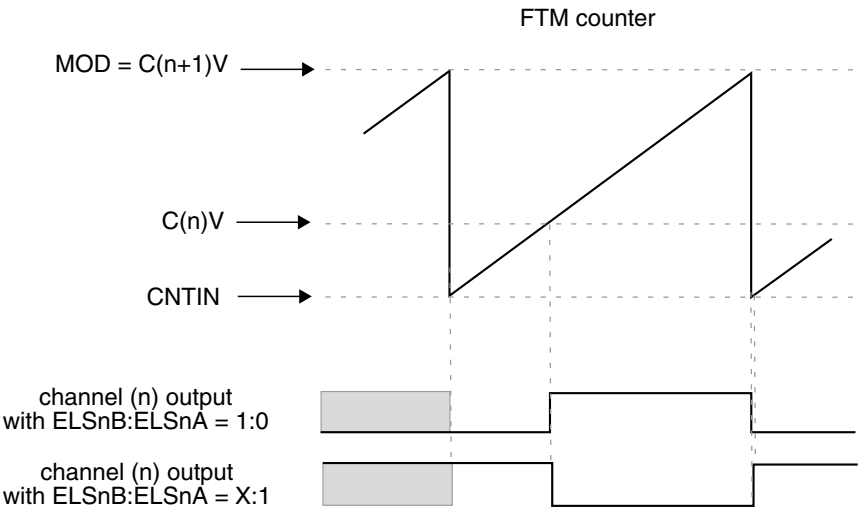


Figure 39-28. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n+1)V = MOD)

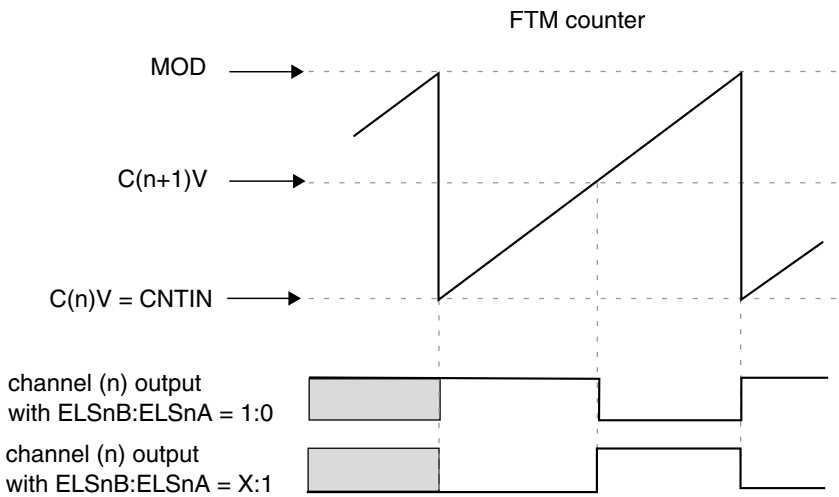


Figure 39-29. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD)

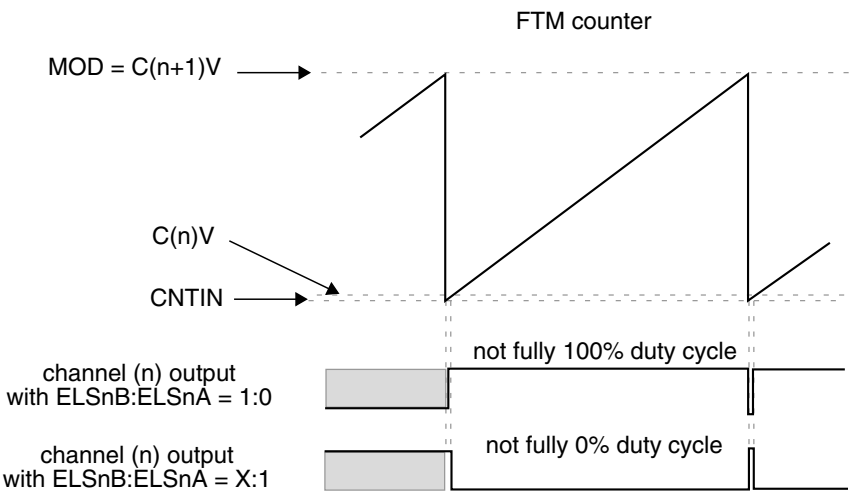


Figure 39-30. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n)V is Almost Equal to CNTIN) and (C(n+1)V = MOD)

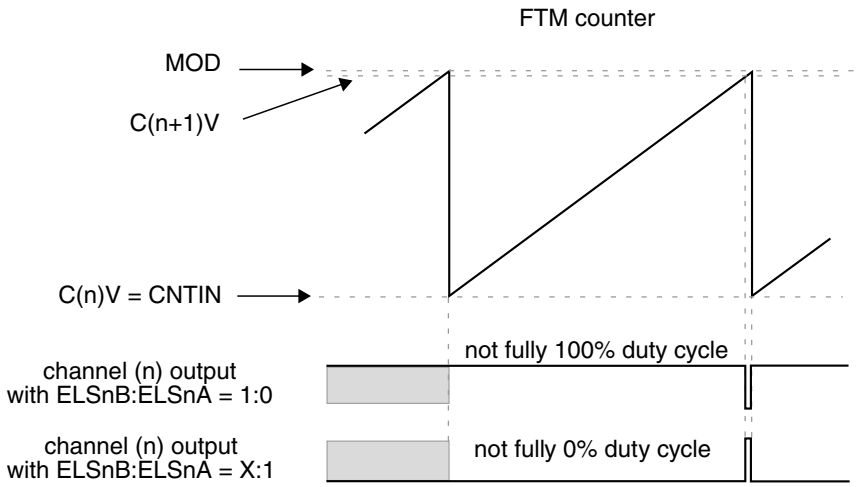


Figure 39-31. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD) and (C(n+1)V is Almost Equal to MOD)

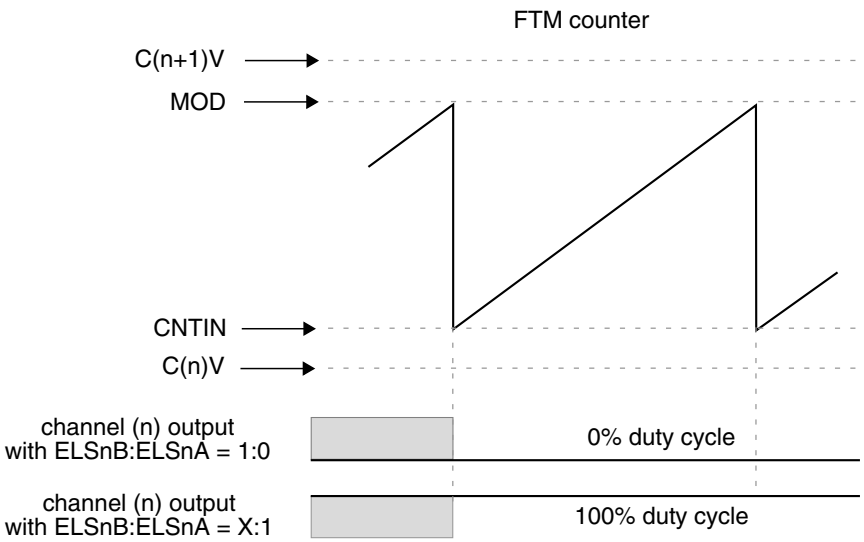


Figure 39-32. Channel (n) output if C(n)V and C(n+1)V are not between CNTIN and MOD

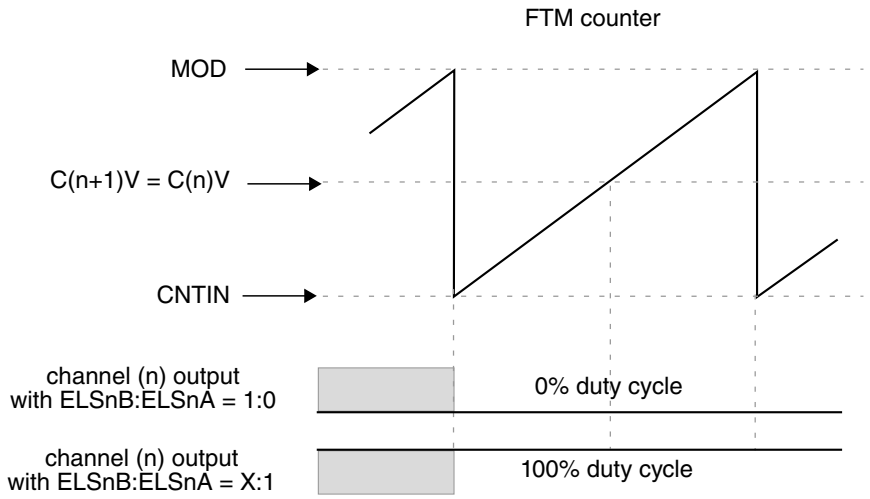


Figure 39-33. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V = C(n+1)V)

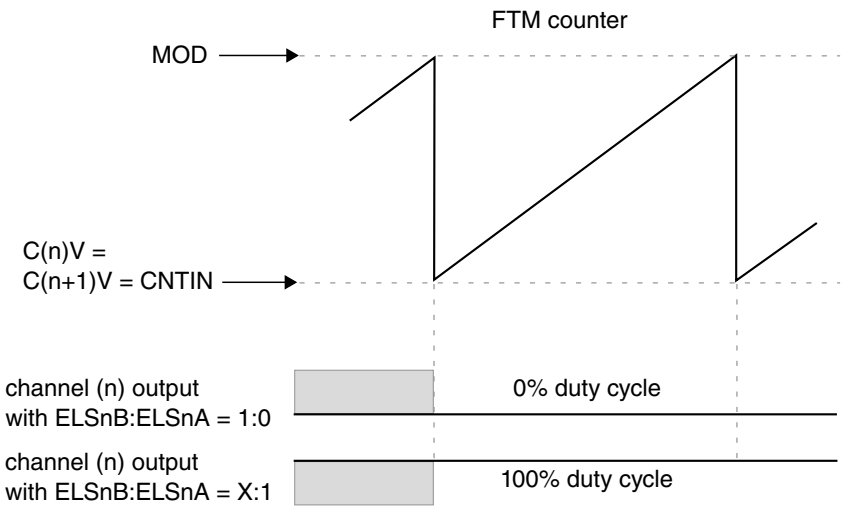


Figure 39-34. Channel (n) output if (C(n)V = C(n+1)V = CNTIN)

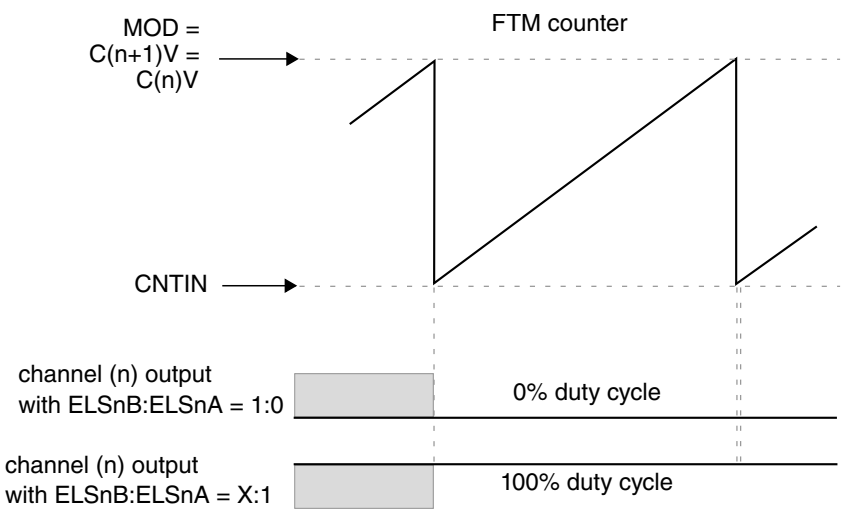


Figure 39-35. Channel (n) output if (C(n)V = C(n+1)V = MOD)

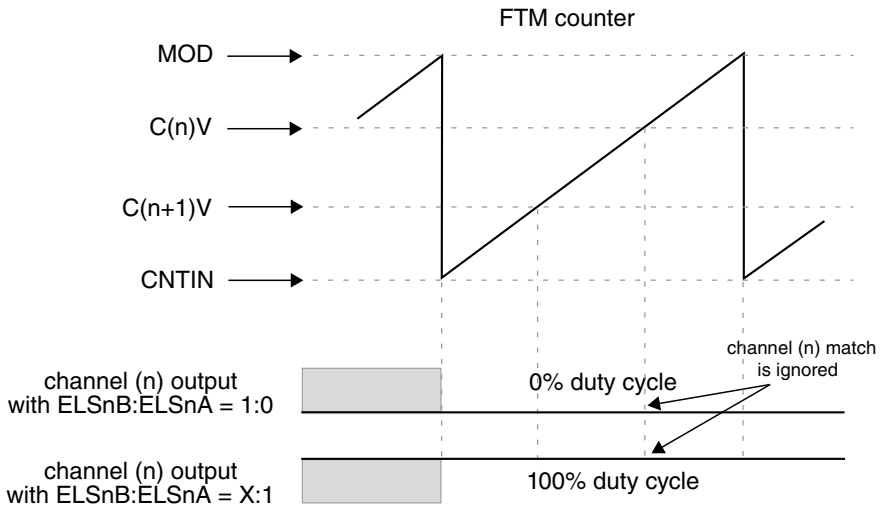


Figure 39-36. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V > C(n+1)V)

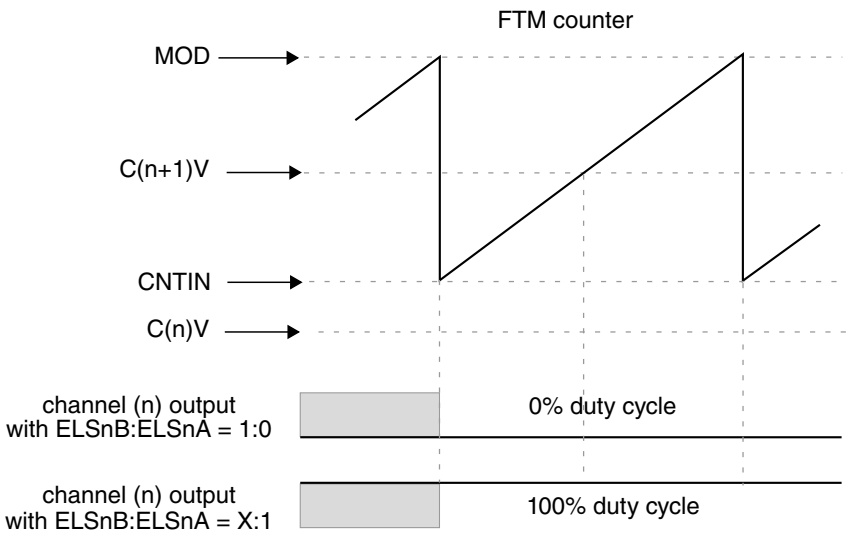


Figure 39-37. Channel (n) output if (C(n)V < CNTIN) and (CNTIN < C(n+1)V < MOD)

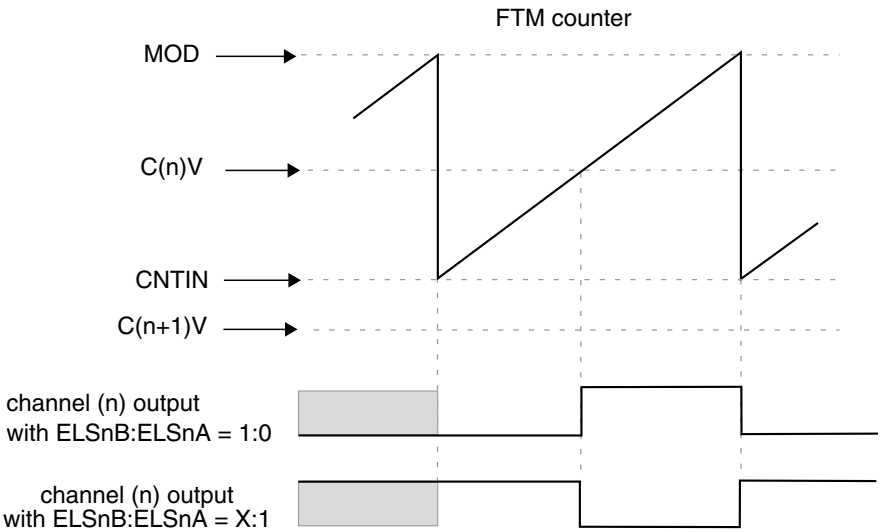


Figure 39-38. Channel (n) output if (C(n+1)V < CNTIN) and (CNTIN < C(n)V < MOD)

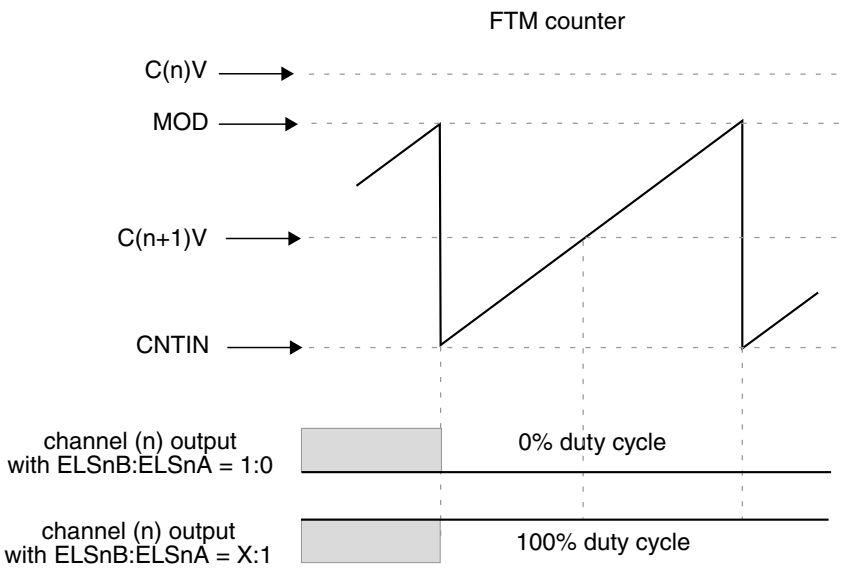


Figure 39-39. Channel (n) output if (C(n)V > MOD) and (CNTIN < C(n+1)V < MOD)

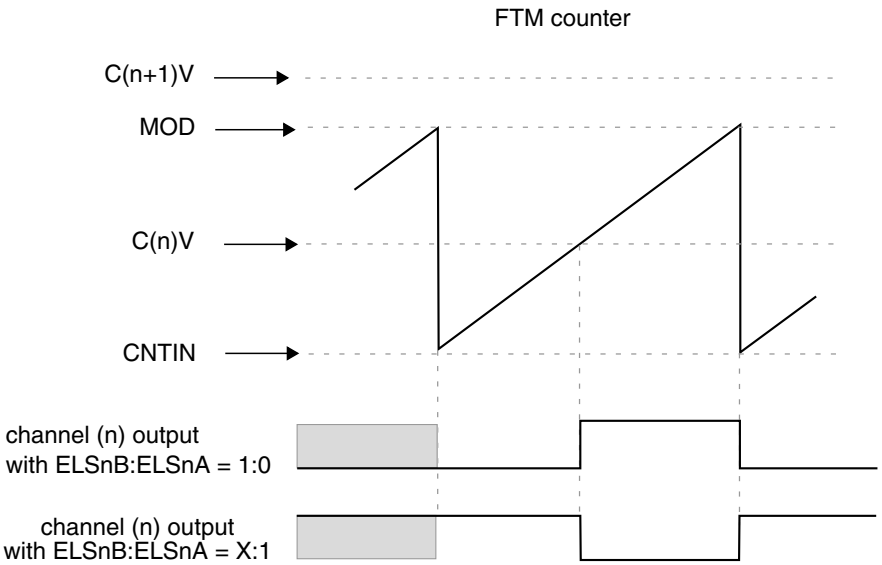


Figure 39-40. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V < MOD)

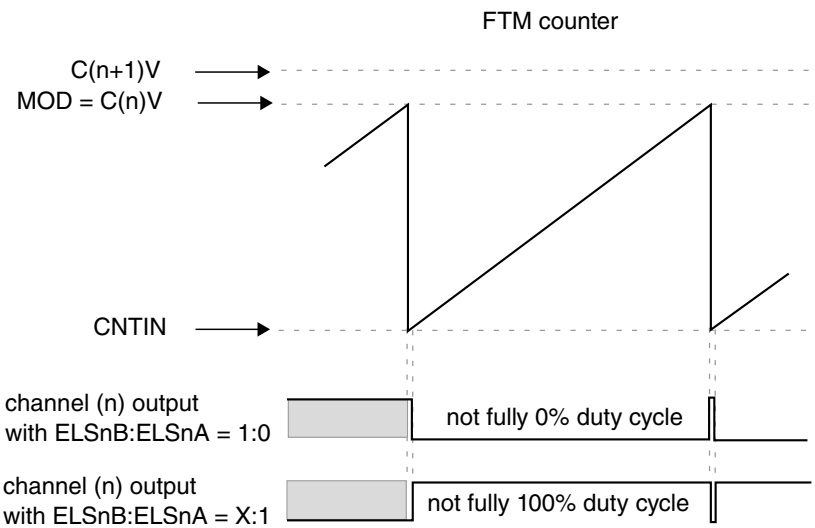


Figure 39-41. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V = MOD)

39.5.8.1 Asymmetrical PWM

In Combine mode, the control of the PWM signal first edge, when the channel (n) match occurs, that is, FTM counter = C(n)V, is independent of the control of the PWM signal second edge, when the channel (n+1) match occurs, that is, FTM counter = C(n+1)V. So, Combine mode allows the generation of asymmetrical PWM signals.

39.5.9 Complementary mode

The Complementary mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1

In Complementary mode, the channel (n+1) output is the inverse of the channel (n) output.

So, the channel (n+1) output is the same as the channel (n) output when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 0

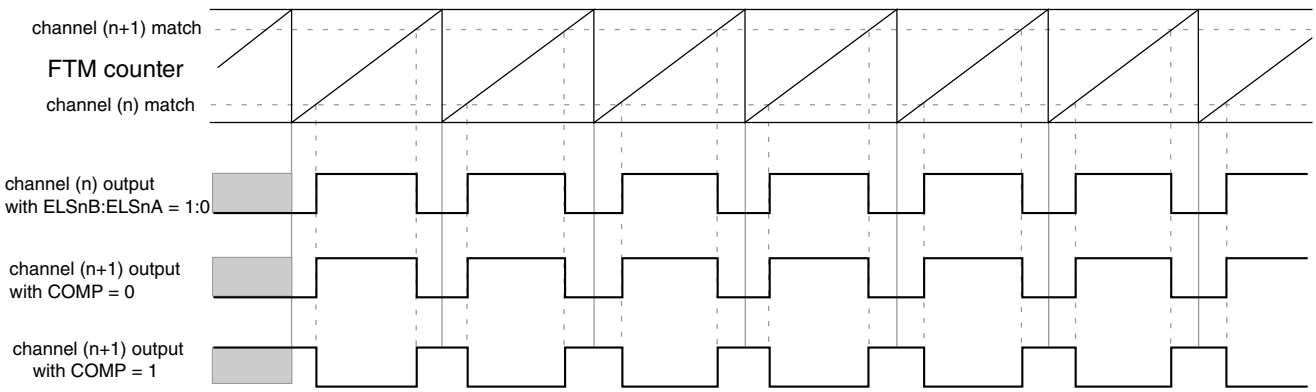


Figure 39-42. Channel (n+1) output in Complementary mode with (ELSnB:ELSnA = 1:0)

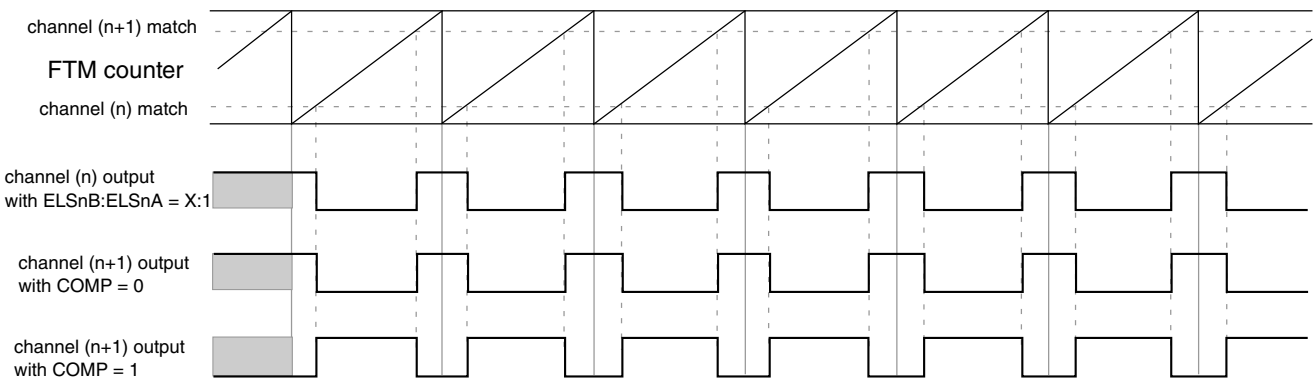


Figure 39-43. Channel (n+1) output in Complementary mode with (ELSnB:ELSnA = X:1)

NOTE

The complementary mode is not available in Output Compare mode.

39.5.10 Registers updated from write buffers

39.5.10.1 CNTIN register update

The following table describes when CNTIN register is updated:

Table 39-6. CNTIN register update

When	Then CNTIN register is updated
CLKS[1:0] = 0:0	When CNTIN register is written, independent of FTMEN bit.
• FTMEN = 0, or • CNTINC = 0	At the next system clock after CNTIN was written.
• FTMEN = 1, • SYNCMODE = 1, and • CNTINC = 1	By the CNTIN register synchronization.

39.5.10.2 MOD register update

The following table describes when MOD register is updated:

Table 39-7. MOD register update

When	Then MOD register is updated
CLKS[1:0] = 0:0	When MOD register is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the CPWMS bit, that is: • If the selected mode is not CPWM then MOD register is updated after MOD register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM then MOD register is updated after MOD register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	By the MOD register synchronization.

39.5.10.3 CnV register update

The following table describes when CnV register is updated:

Table 39-8. CnV register update

When	Then CnV register is updated
CLKS[1:0] = 0:0	When CnV register is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the selected mode, that is:

Table continues on the next page...

Table 39-8. CnV register update (continued)

When	Then CnV register is updated
	• If the selected mode is Output Compare, then CnV register is updated on the next FTM counter change, end of the prescaler counting, after CnV register was written. • If the selected mode is EPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	According to the selected mode, that is: • If the selected mode is output compare then CnV register is updated according to the SYNCEN bit. If (SYNCEN = 0) then CnV register is updated after CnV register was written at the next change of the FTM counter, the end of the prescaler counting. If (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization. • If the selected mode is not output compare and (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization.

39.5.11 PWM synchronization

The PWM synchronization provides an opportunity to update the MOD, CNTIN, CnV, OUTMASK, INVCTRL and SWOCTRL registers with their buffered value and force the FTM counter to the CNTIN register value.

Note

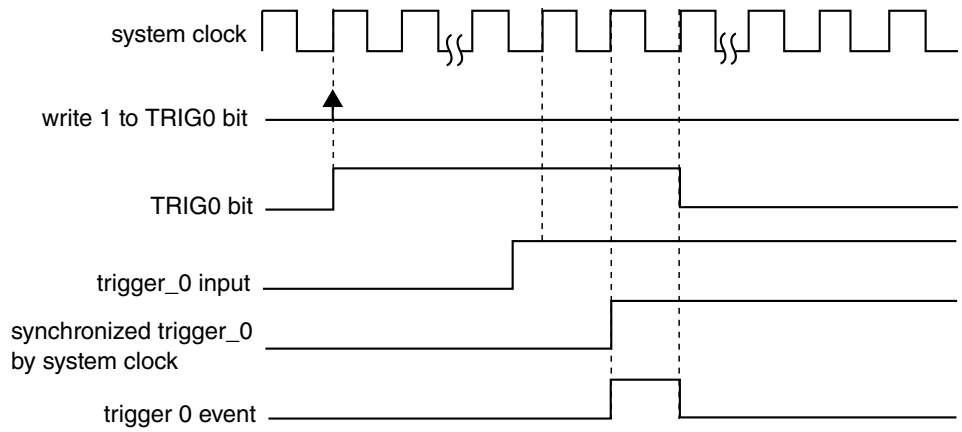
The legacy PWM synchronization (SYNCMODE = 0) is a subset of the enhanced PWM synchronization (SYNCMODE = 1). Thus, only the enhanced PWM synchronization must be used.

39.5.11.1 Hardware trigger

Three hardware trigger signal inputs of the FTM module are enabled when TRIGn = 1, where n = 0, 1 or 2 corresponding to each one of the input signals, respectively. The hardware trigger input n is synchronized by the system clock. The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs.

If (HWTRIGMODE = 0) then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case, if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs, then only TRIG1 bit is cleared. If a trigger n event occurs together with a write setting TRIGn bit, then the synchronization is initiated, but TRIGn bit remains set due to the write operation.



Note
All hardware trigger inputs have the same behavior.

Figure 39-44. Hardware trigger event with HWTRIGMODE = 0

If HWTRIGMODE = 1, then the TRIGn bit is only cleared when 0 is written to it.

NOTE

The HWTRIGMODE bit must be 1 only with enhanced PWM synchronization (SYNCMODE = 1).

39.5.11.2 Software trigger

A software trigger event occurs when 1 is written to the SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization, initiated by the software event, is completed.

If another software trigger event occurs (by writing another 1 to the SWSYNC bit) at the same time the PWM synchronization initiated by the previous software trigger event is ending, a new PWM synchronization is started and the SWSYNC bit remains equal to 1.

If SYNCMODE = 0 then the SWSYNC bit is also cleared by FTM according to PWMSYNC and REINIT bits. In this case if (PWMSYNC = 1) or (PWMSYNC = 0 and REINIT = 0) then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see Boundary cycle and loading points and the following figure. If (PWMSYNC = 0) and (REINIT = 1) then SWSYNC bit is cleared when the software trigger event occurs.

If SYNCMODE = 1 then the SWSYNC bit is also cleared by FTM according to the SWRSTCNT bit. If SWRSTCNT = 0 then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see the following figure. If SWRSTCNT = 1 then SWSYNC bit is cleared when the software trigger event occurs.

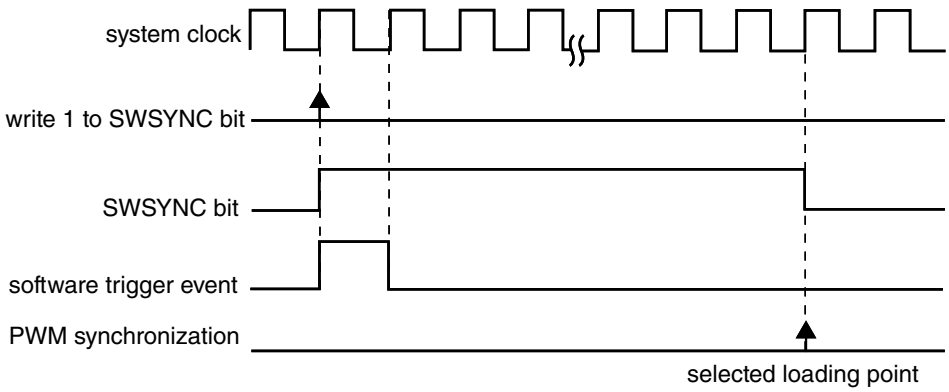


Figure 39-45. Software trigger event

39.5.11.3 Boundary cycle and loading points

The boundary cycle definition is important for the loading points for the registers MOD, CNTIN, and C(n)V.

In Up counting mode, the boundary cycle is defined as when the counter wraps to its initial value (CNTIN). If in Up-down counting mode, then the boundary cycle is defined as when the counter turns from down to up counting and when from up to down counting.

The following figure shows the boundary cycles and the loading points for the registers. In the Up Counting mode, the loading points are enabled if one of CNTMIN or CTMAX bits are 1. In the Up-Down Counting mode, the loading points are selected by CNTMIN and CNTMAX bits, as indicated in the figure. These loading points are safe places for register updates thus allowing a smooth transitions in PWM waveform generation.

For both counting modes, if neither CNTMIN nor CNTMAX are 1, then the boundary cycles are not used as loading points for registers updates. See the register synchronization descriptions in the following sections for details.

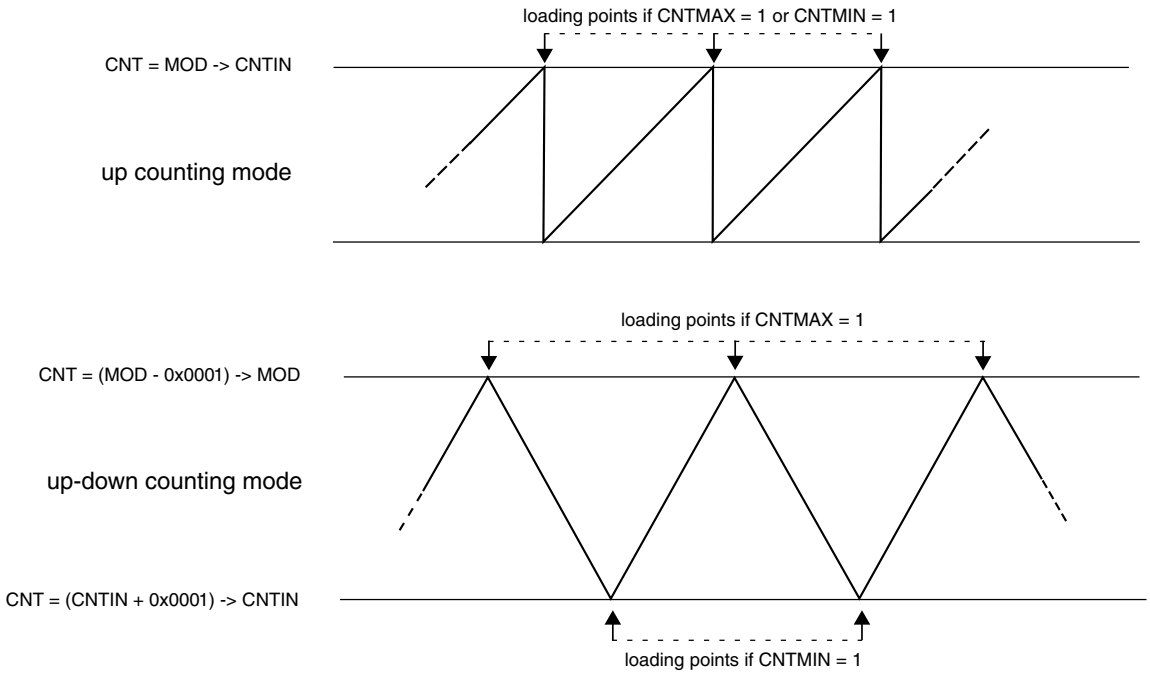


Figure 39-46. Boundary cycles and loading points

39.5.11.4 MOD register synchronization

The MOD register synchronization updates the MOD register with its buffer value. This synchronization is enabled if (FTMEN = 1).

The MOD register synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the MOD register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the MOD register synchronization depends on SWWRBUF, SWRSTCNT, HWWRBUF, and HWRSTCNT bits according to this flowchart:

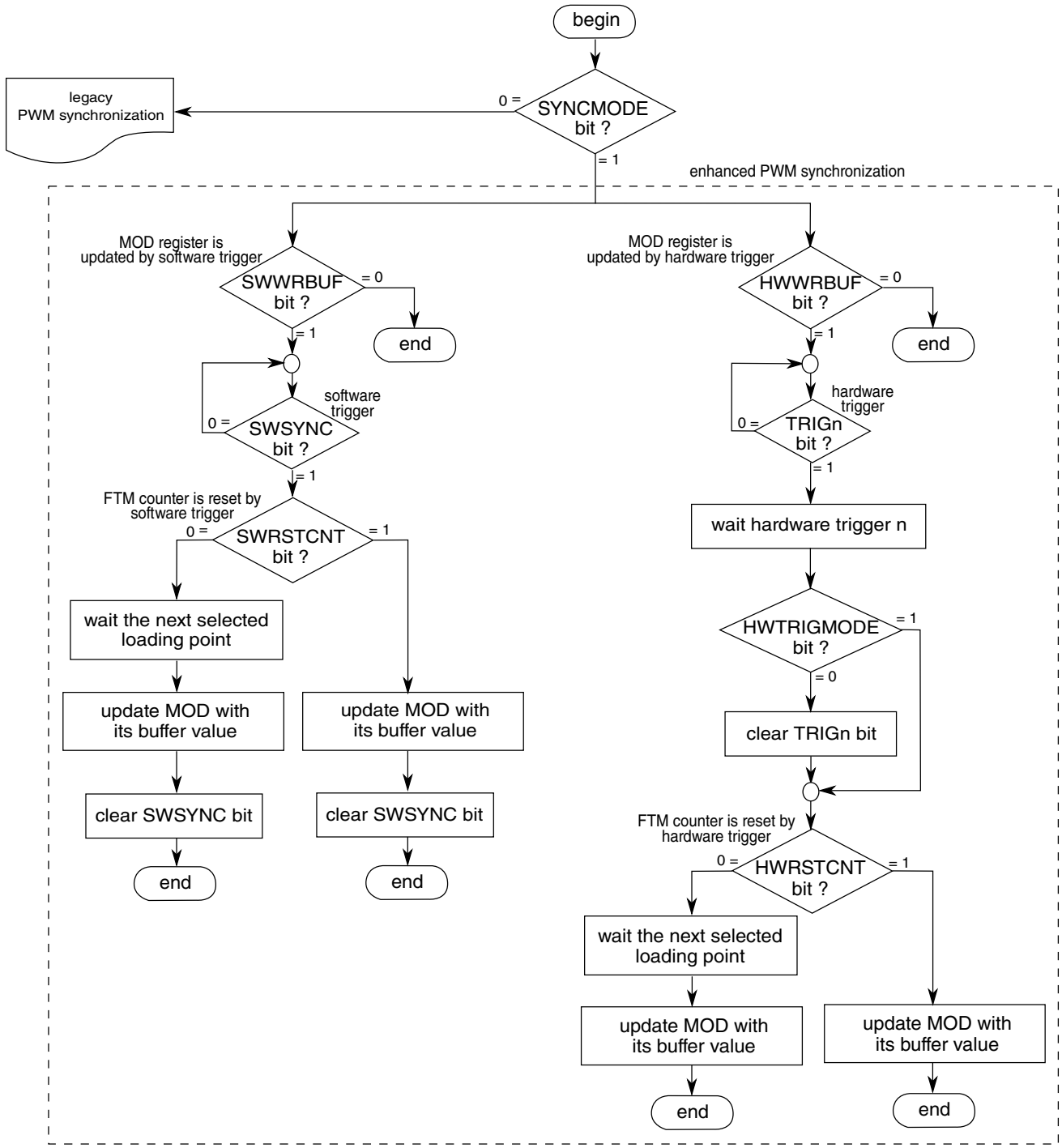


Figure 39-47. MOD register synchronization flowchart

In the case of legacy PWM synchronization, the MOD register synchronization depends on PWMSYNC and REINIT bits according to the following description.

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 0), then this synchronization is made on the next selected loading point after an enabled trigger event takes place. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected

loading point. If the trigger event was a hardware trigger, then the trigger enable bit (TRIGn) is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

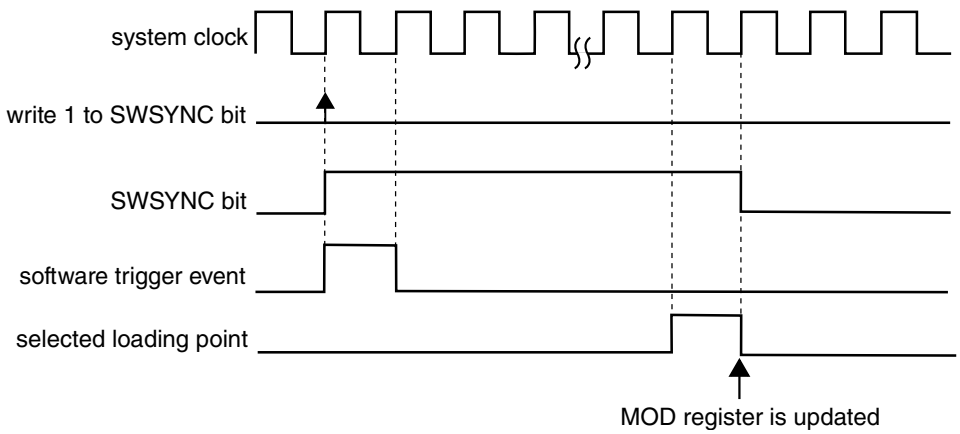


Figure 39-48. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 0), and software trigger was used

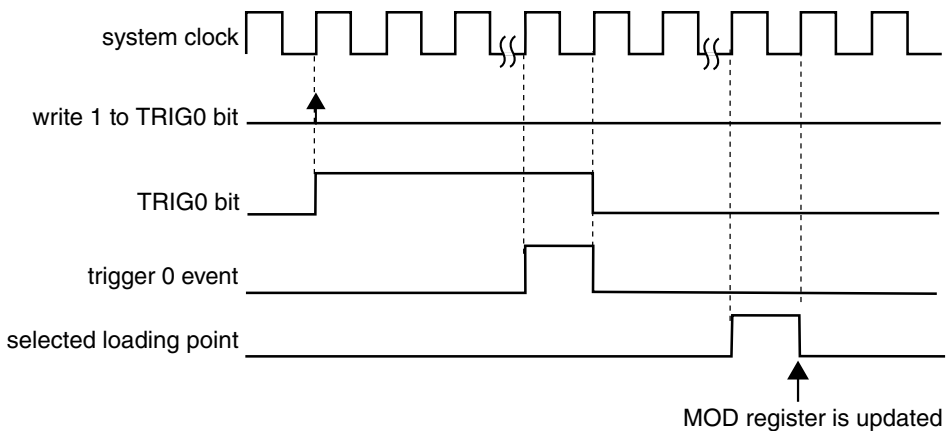


Figure 39-49. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 0), and a hardware trigger was used

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 1), then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

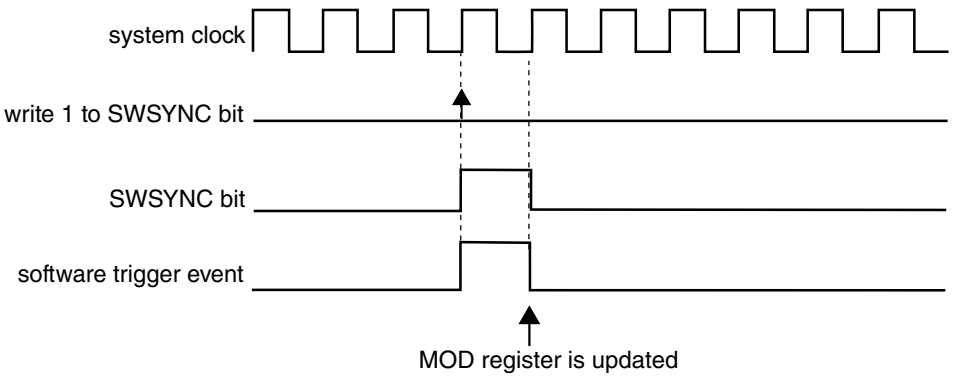


Figure 39-50. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 1), and software trigger was used

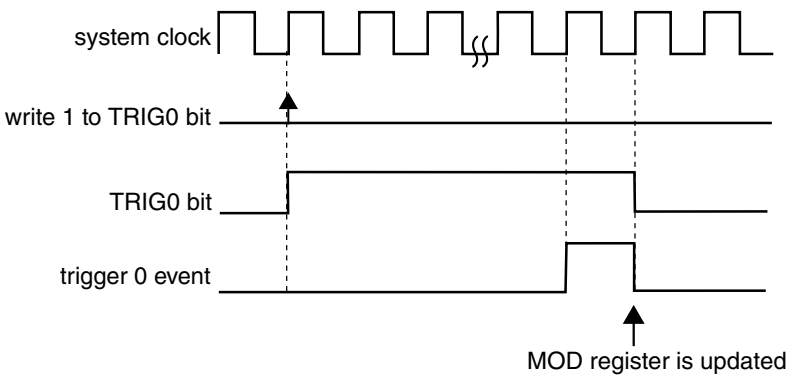


Figure 39-51. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 1), and a hardware trigger was used

If (SYNCMODE = 0) and (PWMSYNC = 1), then this synchronization is made on the next selected loading point after the software trigger event takes place. The SWSYNC bit is cleared on the next selected loading point:

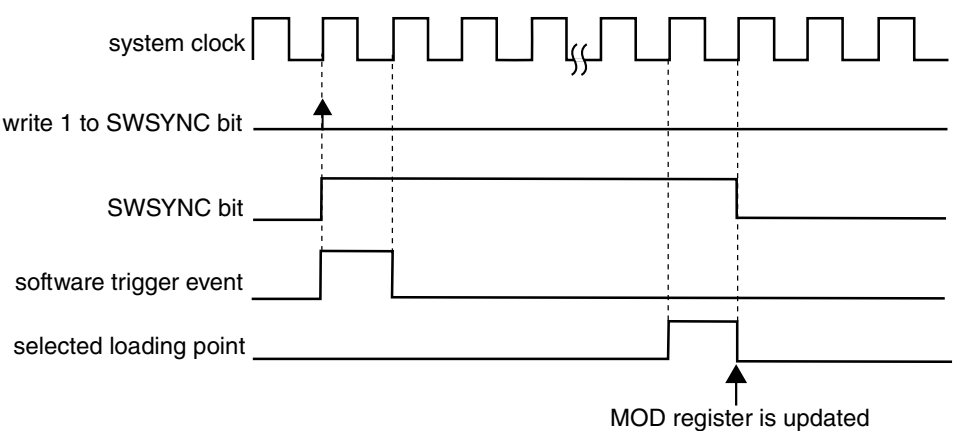


Figure 39-52. MOD synchronization with (SYNCMODE = 0) and (PWMSYNC = 1)

39.5.11.5 CNTIN register synchronization

The CNTIN register synchronization updates the CNTIN register with its buffer value.

This synchronization is enabled if (FTMEN = 1), (SYNCMODE = 1), and (CNTINC = 1). The CNTIN register synchronization can be done only by the enhanced PWM synchronization (SYNCMODE = 1). The synchronization mechanism is the same as the MOD register synchronization done by the enhanced PWM synchronization; see MOD register synchronization.

39.5.11.6 C(n)V and C(n+1)V register synchronization

The C(n)V and C(n+1)V registers synchronization updates the C(n)V and C(n+1)V registers with their buffer values.

This synchronization is enabled if (FTMEN = 1) and (SYNCEN = 1). The synchronization mechanism is the same as the MOD register synchronization. However, it is expected that the C(n)V and C(n+1)V registers be synchronized only by the enhanced PWM synchronization (SYNCMODE = 1).

39.5.11.7 OUTMASK register synchronization

The OUTMASK register synchronization updates the OUTMASK register with its buffer value.

The OUTMASK register can be updated at each rising edge of system clock (SYNCHOM = 0), by the enhanced PWM synchronization (SYNCHOM = 1 and SYNCMODE = 1) or by the legacy PWM synchronization (SYNCHOM = 1 and SYNCMODE = 0). However, it is expected that the OUTMASK register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the OUTMASK register synchronization depends on SWOM and HWOM bits. See the following flowchart:

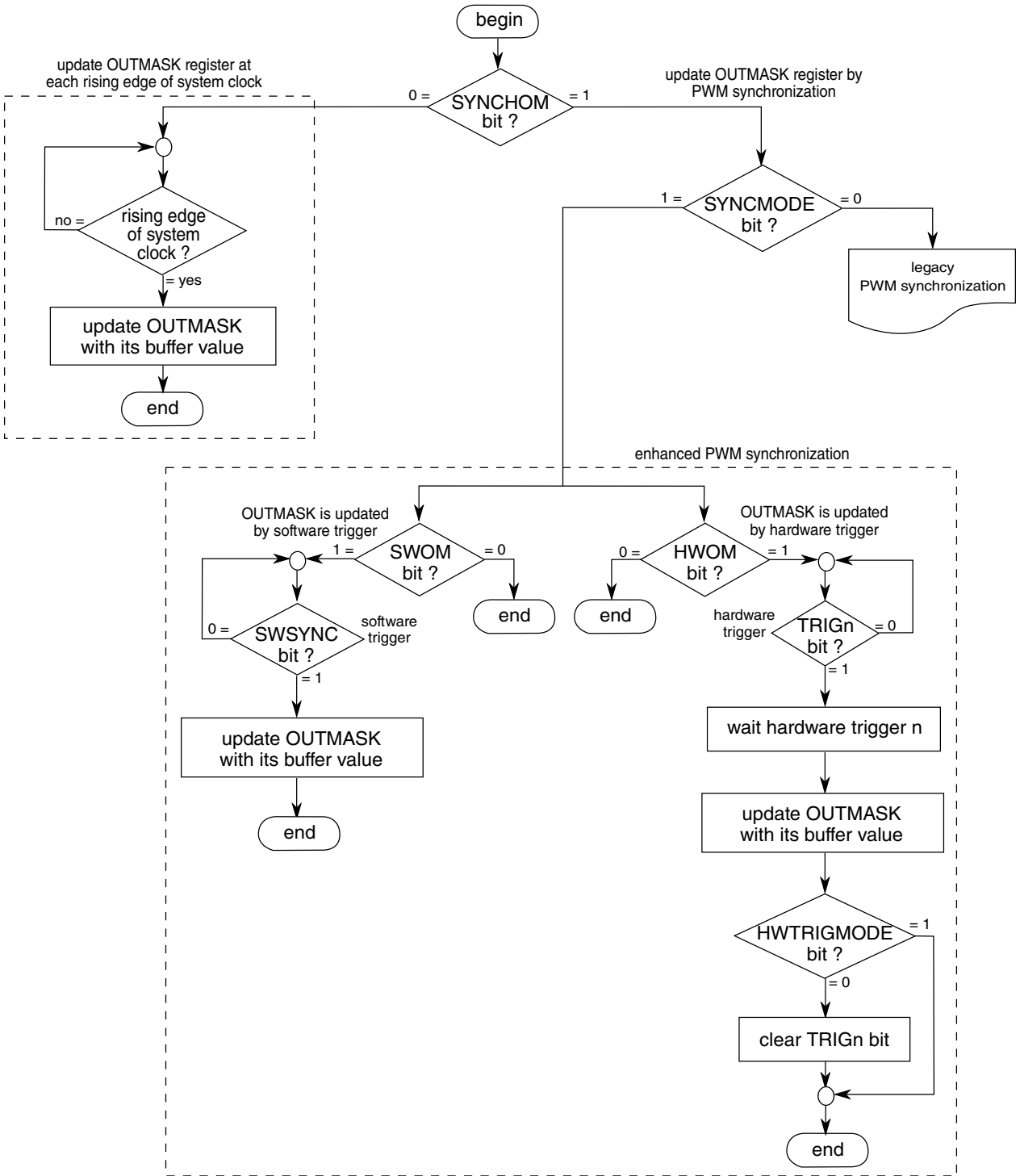


Figure 39-53. OUTMASK register synchronization flowchart

In the case of legacy PWM synchronization, the OUTMASK register synchronization depends on PWMSYNC bit according to the following description.

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 0), then this synchronization is done on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

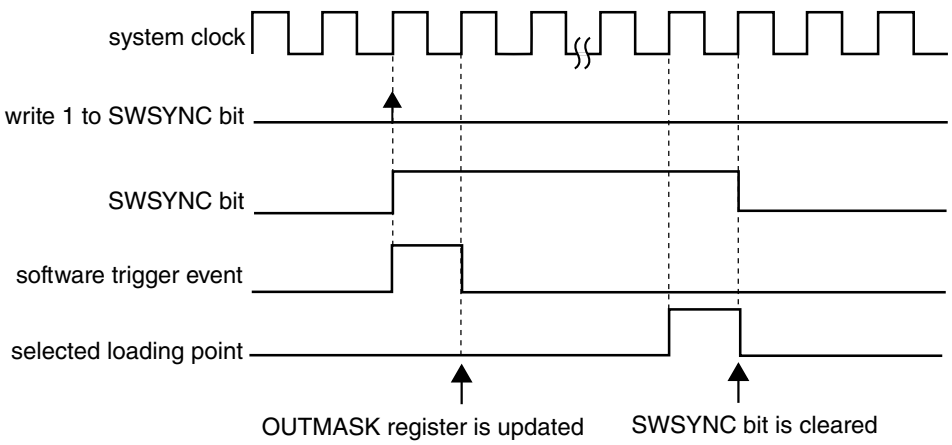


Figure 39-54. OUTMASK synchronization with (SYNCMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0) and software trigger was used

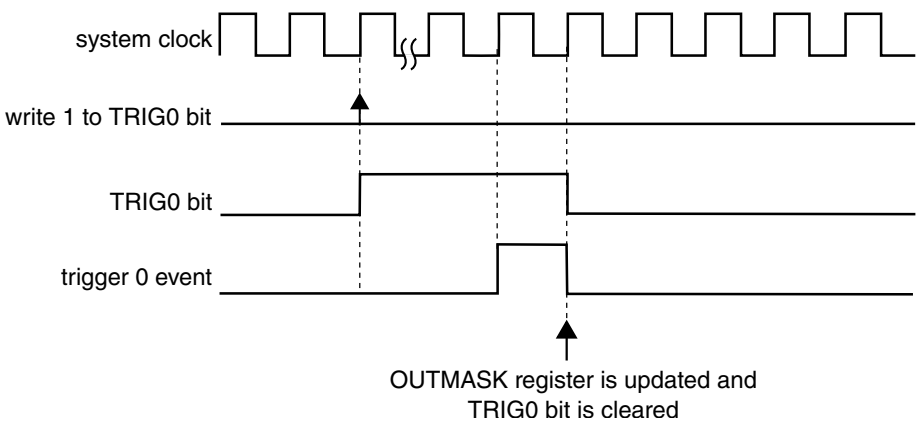


Figure 39-55. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 1), then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger. An example with a hardware trigger follows.

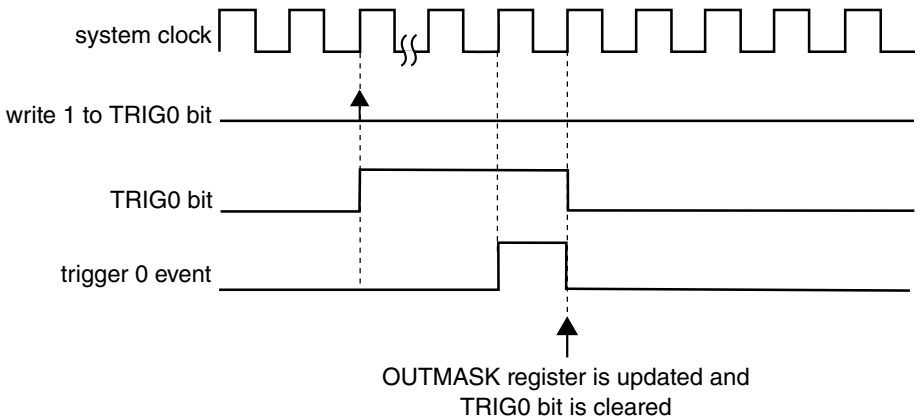


Figure 39-56. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 1), and a hardware trigger was used

39.5.11.8 INVCTRL register synchronization

The INVCTRL register synchronization updates the INVCTRL register with its buffer value.

The INVCTRL register can be updated at each rising edge of system clock (INVC = 0) or by the enhanced PWM synchronization (INVC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the INVCTRL register synchronization depends on SWINVC and HWINVC bits.

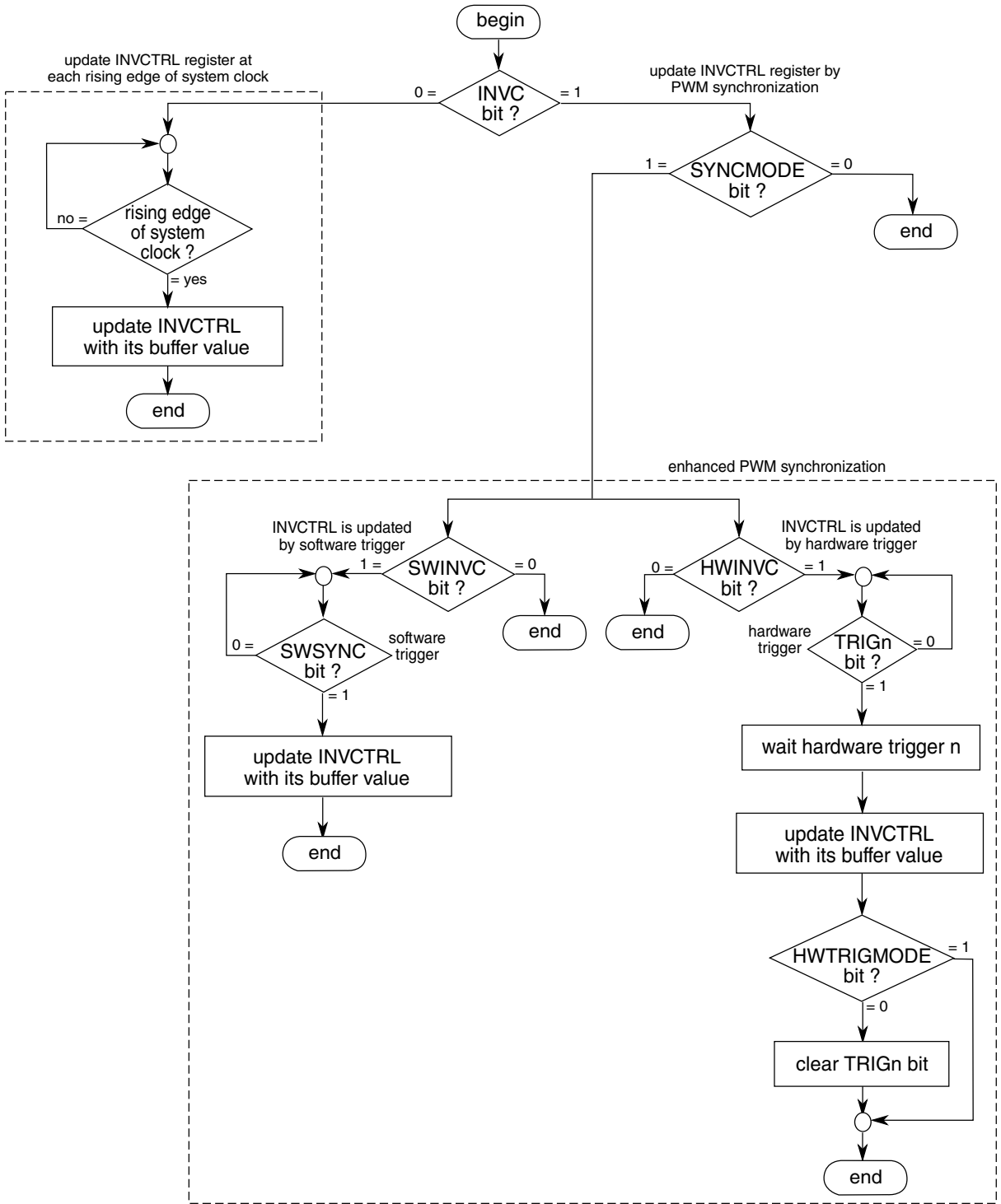


Figure 39-57. INVCTRL register synchronization flowchart

39.5.11.9 SWOCTRL register synchronization

The SWOCTRL register synchronization updates the SWOCTRL register with its buffer value.

The SWOCTRL register can be updated at each rising edge of system clock (SWOC = 0) or by the enhanced PWM synchronization (SWOC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the SWOCTRL register synchronization depends on SWSOC and HWSOC bits.

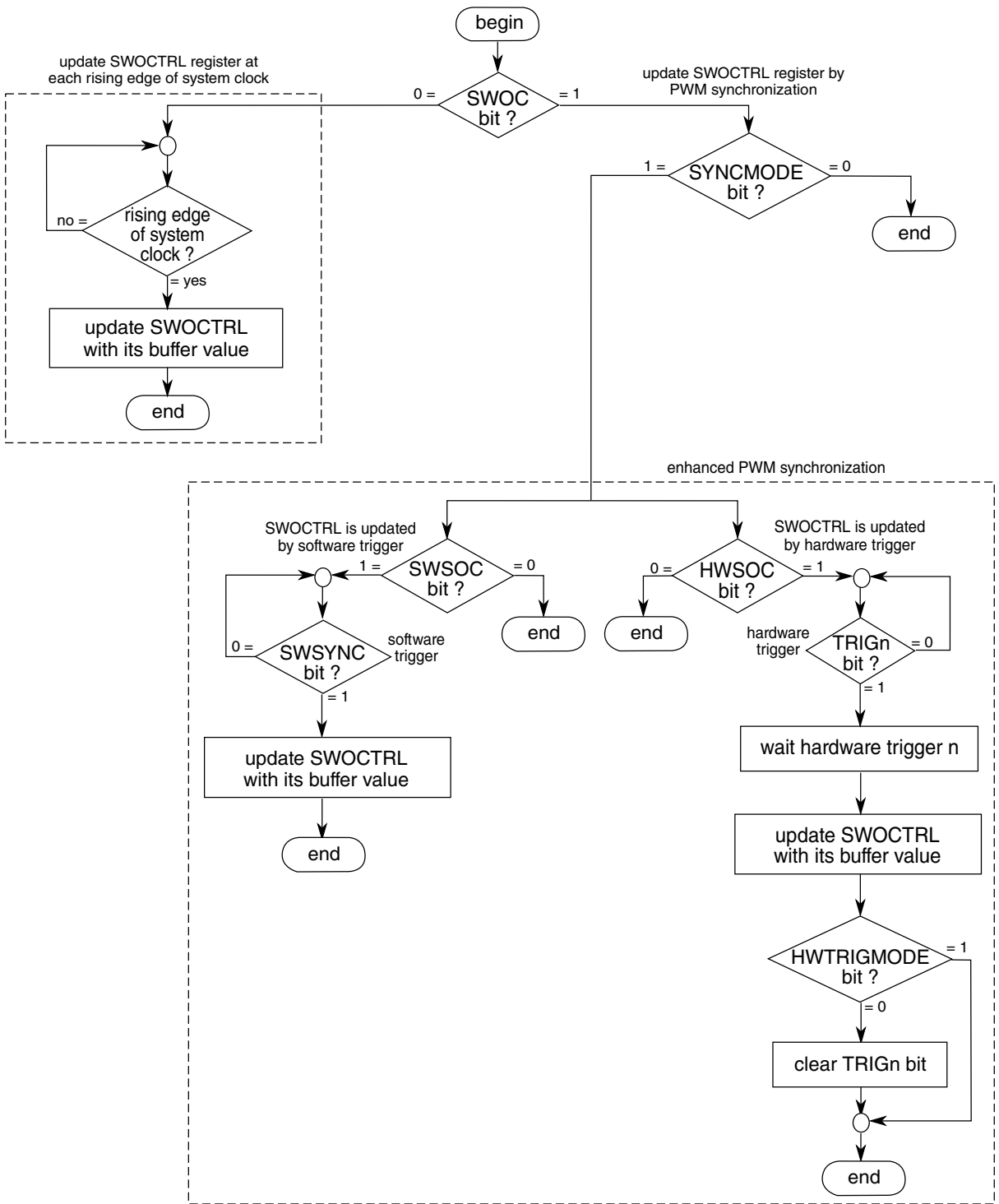


Figure 39-58. SWOCTRL register synchronization flowchart

39.5.11.10 FTM counter synchronization

The FTM counter synchronization is a mechanism that allows the FTM to restart the PWM generation at a certain point in the PWM period. The channels outputs are forced to their initial value, except for channels in Output Compare mode, and the FTM counter is forced to its initial counting value defined by CNTIN register.

The following figure shows the FTM counter synchronization. Note that after the synchronization event occurs, the channel (n) is set to its initial value and the channel (n +1) is not set to its initial value due to a specific timing of this figure in which the deadtime insertion prevents this channel output from transitioning to 1. If no deadtime insertion is selected, then the channel (n+1) transitions to logical value 1 immediately after the synchronization event occurs.

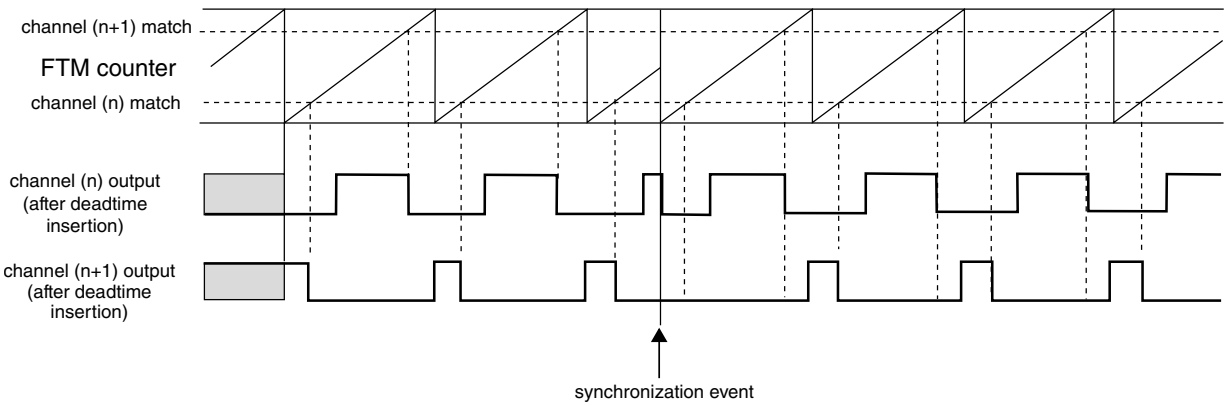


Figure 39-59. FTM counter synchronization

The FTM counter synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, the FTM counter must be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the FTM counter synchronization depends on SWRSTCNT and HWRSTCNT bits according to the following flowchart.

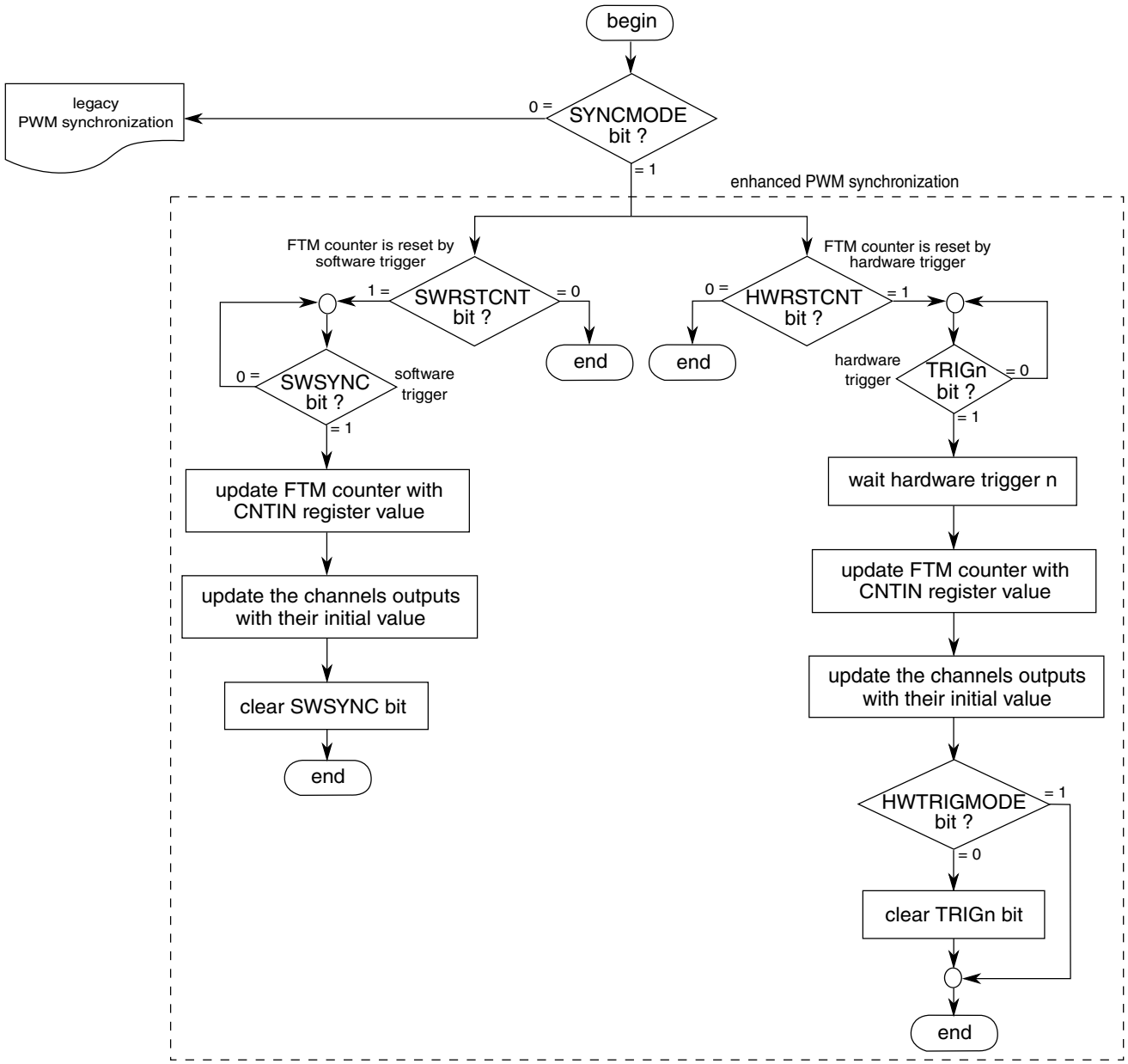


Figure 39-60. FTM counter synchronization flowchart

In the case of legacy PWM synchronization, the FTM counter synchronization depends on REINIT and PWMSYNC bits according to the following description.

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 0) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

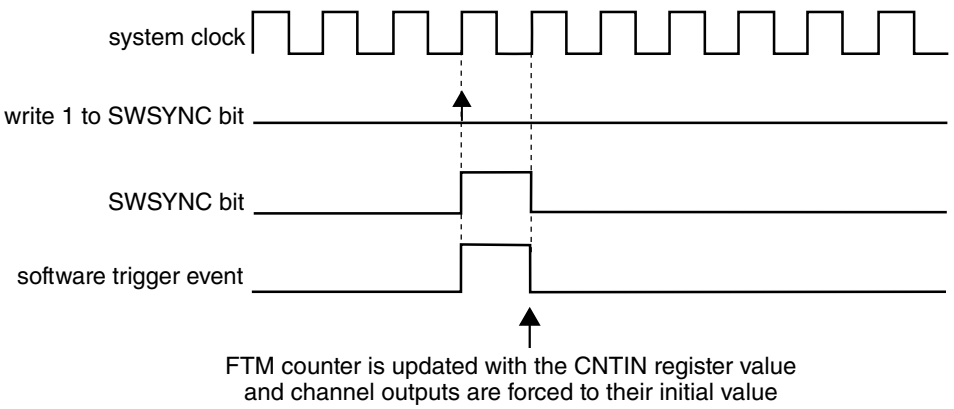


Figure 39-61. FTM counter synchronization with (SYNCMODE = 0), (REINIT = 1), (PWMSYNC = 0), and software trigger was used

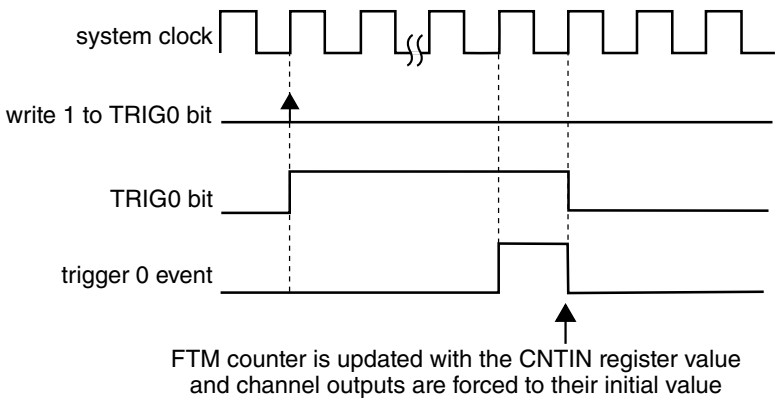


Figure 39-62. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger.

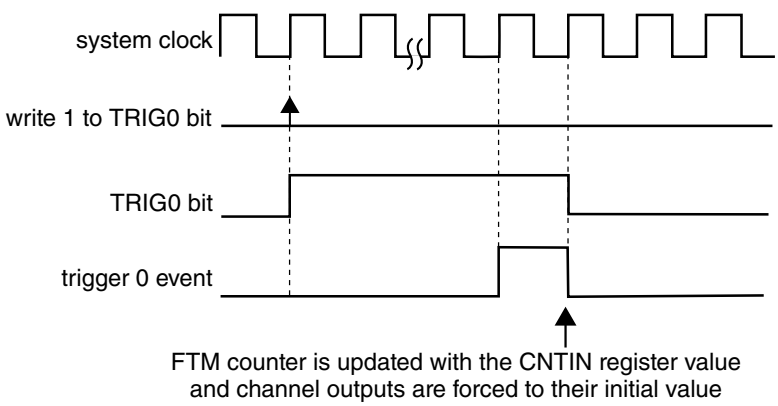


Figure 39-63. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 1), and a hardware trigger was used

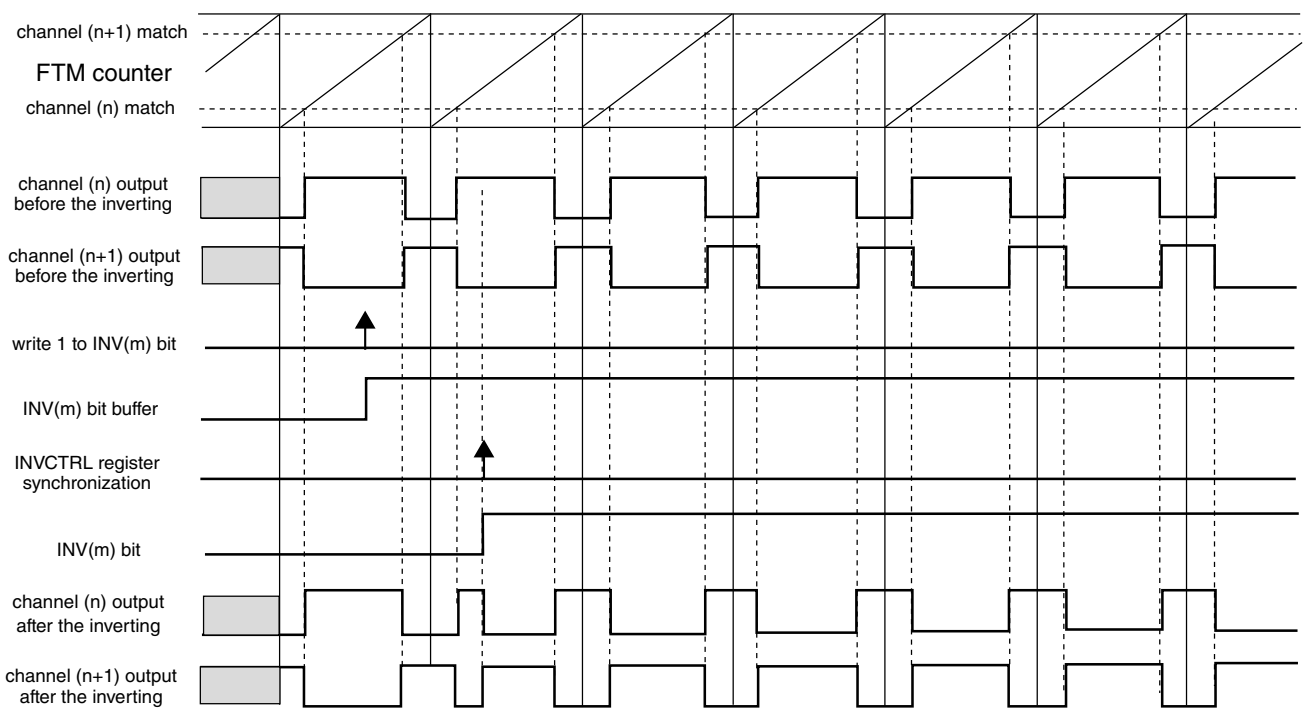
39.5.12 Inverting

The invert functionality swaps the signals between channel (n) and channel (n+1) outputs. The inverting operation is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1, and
- INVm = 1 (where m represents a channel pair)

The INVm bit in INVCTRL register is updated with its buffer value according to INVCTRL register synchronization

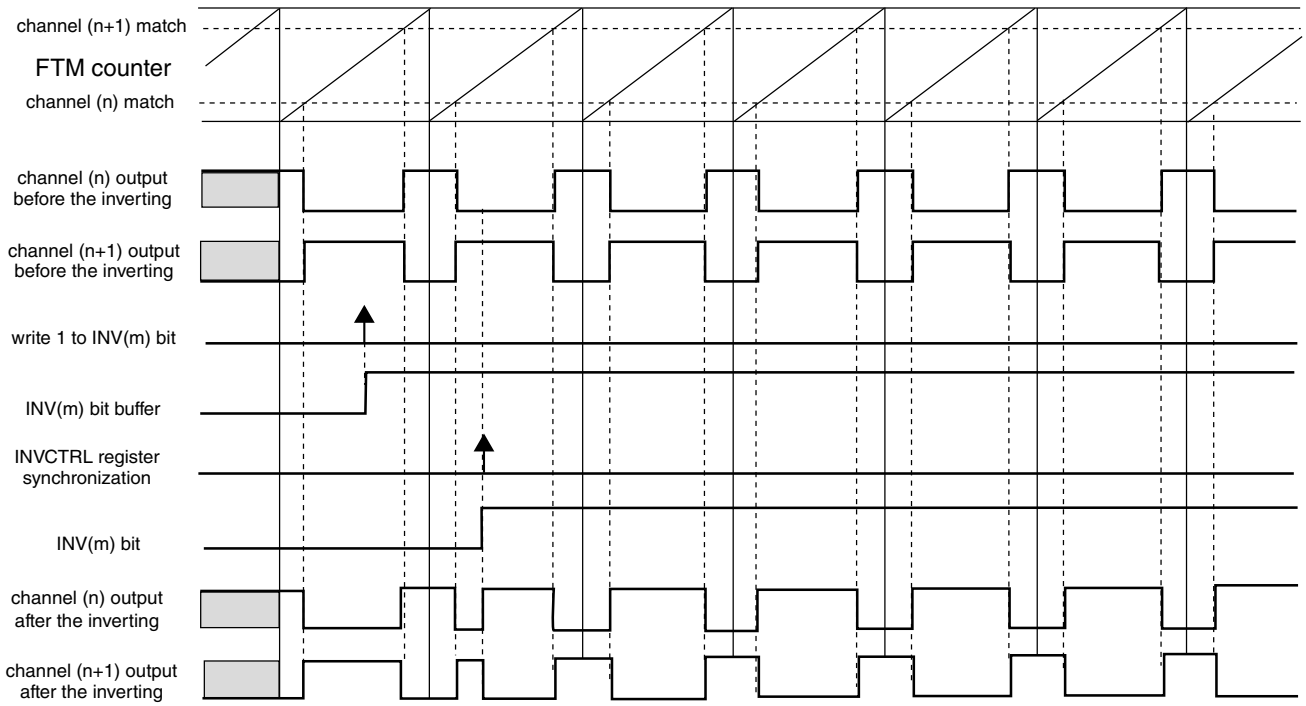
In High-True (ELSnB:ELSnA = 1:0) Combine mode, the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN), forced high at the channel (n) match and forced low at the channel (n+1) match. If the inverting is selected, the channel (n) output behavior is changed to force high at the beginning of the PWM period, force low at the channel (n) match and force high at the channel (n+1) match. See the following figure.



NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 39-64. Channels (n) and (n+1) outputs after the inverting in High-True (ELSnB:ELSnA = 1:0) Combine mode

Note that the ELSnB:ELSnA bits value should be considered because they define the active state of the channels outputs. In Low-True (ELSnB:ELSnA = X:1) Combine mode, the channel (n) output is forced high at the beginning of the period, forced low at the channel (n) match and forced high at the channel (n+1) match. When inverting is selected, the channels (n) and (n+1) present waveforms as shown in the following figure.



NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 39-65. Channels (n) and (n+1) outputs after the inverting in Low-True (ELSnB:ELSnA = X:1) Combine mode

Note

The inverting feature is not available in Output Compare mode.

39.5.13 Software output control

The software output control forces the channel output according to software defined values at a specific time in the PWM generation.

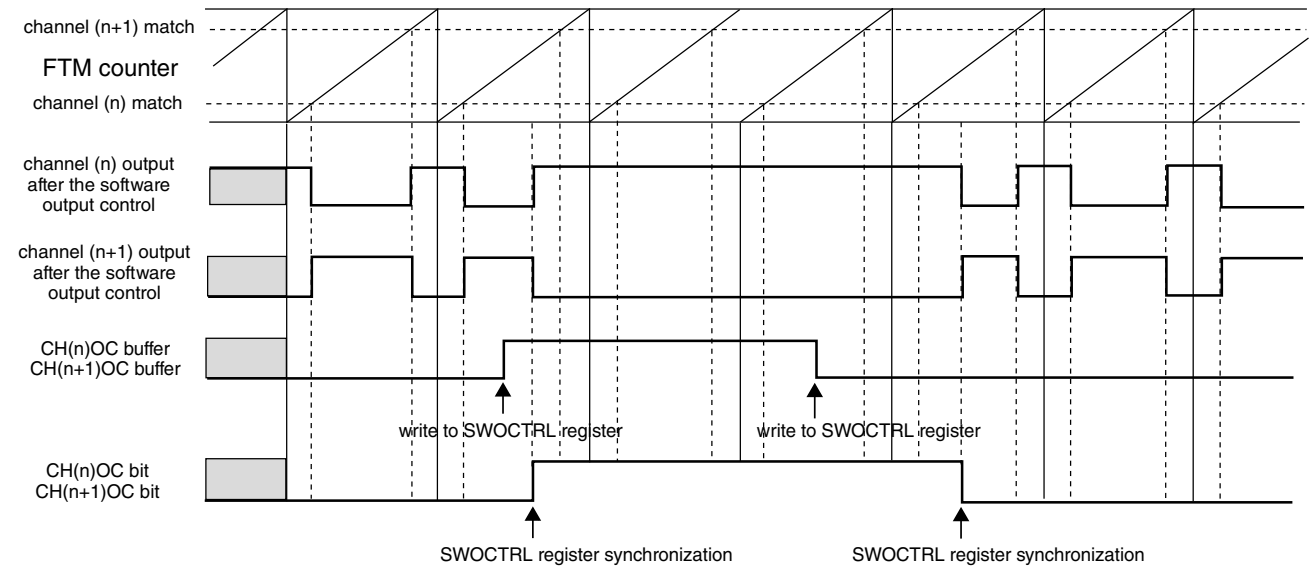
The software output control is selected when:

- QUADEN = 0
- DECAPEN = 0, and
- CHnOC = 1

The CHnOC bit enables the software output control for a specific channel output and the CHnOCV selects the value that is forced to this channel output.

Both CHnOC and CHnOCV bits in SWOCTRL register are buffered and updated with their buffer value according to SWOCTRL register synchronization.

The following figure shows the channels (n) and (n+1) outputs signals when the software output control is used. In this case the channels (n) and (n+1) are set to Combine and Complementary mode.



NOTE
CH(n)OCV = 1 and CH(n+1)OCV = 0.

Figure 39-66. Example of software output control in Combine and Complementary mode

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is zero.

Table 39-9. Software ouput control behavior when (COMP = 0)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to one

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is one.

Table 39-10. Software ouput control behavior when (COMP = 1)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to zero

Note

- The CH(n)OC and CH(n+1)OC bits should be equal.
- The COMP bit must not be modified when software output control is enabled, that is, CH(n)OC = 1 and/or CH(n +1)OC = 1.
- Software output control has the same behavior with disabled or enabled FTM counter (see the CLKS field description in the Status and Control register).

39.5.14 Deadtime insertion

The deadtime insertion is enabled when (DTEN = 1) and (DTVAL[5:0] is non- zero).

DEADTIME register defines the deadtime delay that can be used for all FTM channels. The DTPS[1:0] bits define the prescaler for the system clock and the DTVAL[5:0] bits define the deadtime modulo, that is, the number of the deadtime prescaler clocks.

The deadtime delay insertion ensures that no two complementary signals (channels (n) and (n+1)) drive the active state at the same time.

If POL(n) = 0, POL(n+1) = 0, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If POL(n) = 1, POL(n+1) = 1, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the high value until the end of the deadtime delay when the channel (n) output is cleared. Similarly,

when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the high value until the end of the deadtime delay when the channel (n +1) output is cleared.

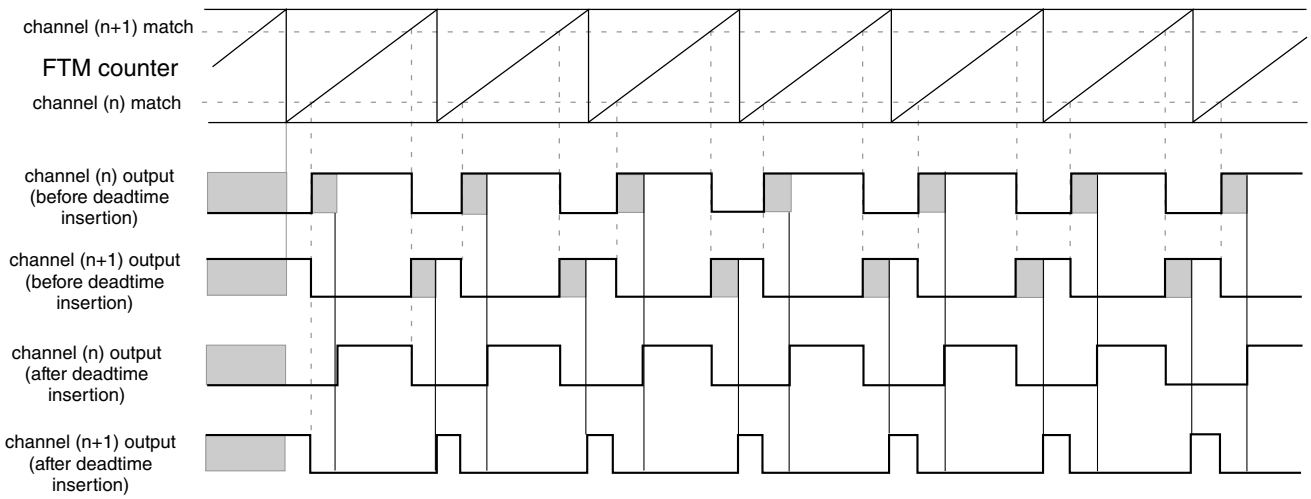


Figure 39-67. Deadtime insertion with ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0

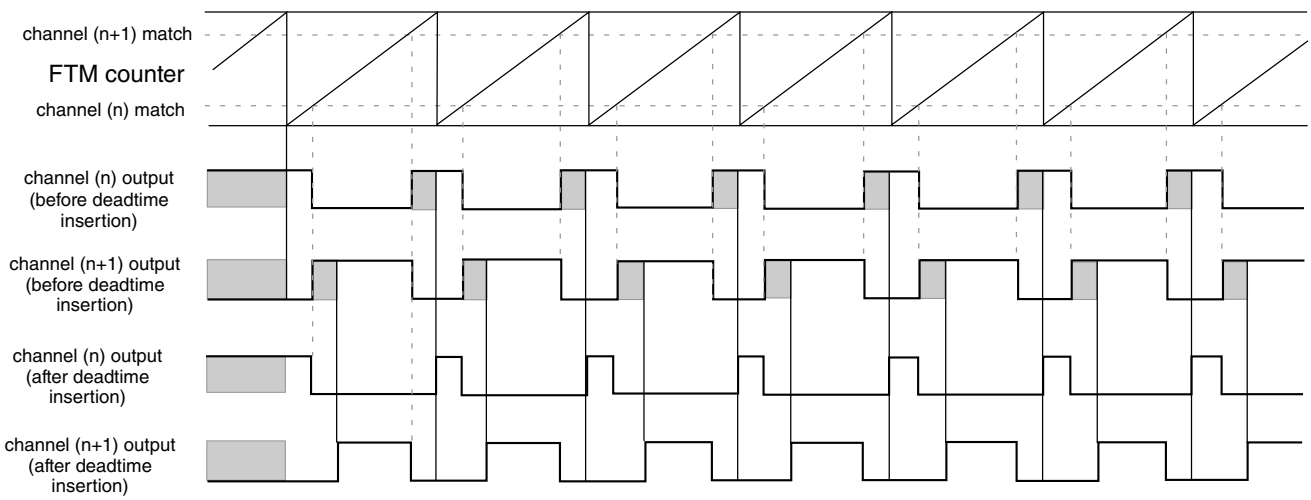


Figure 39-68. Deadtime insertion with ELSnB:ELSnA = X:1, POL(n) = 0, and POL(n+1) = 0

NOTE

- The deadtime feature must be used only in Complementary mode.
- The deadtime feature is not available in Output Compare mode.

39.5.14.1 Deadtime insertion corner cases

If (PS[2:0] is cleared), (DTPS[1:0] = 0:0 or DTPS[1:0] = 0:1):

- and the deadtime delay is greater than or equal to the channel (n) duty cycle ((C(n +1)V – C(n)V) × system clock), then the channel (n) output is always the inactive value (POL(n) bit value).
- and the deadtime delay is greater than or equal to the channel (n+1) duty cycle ((MOD – CNTIN + 1 – (C(n+1)V – C(n)V)) × system clock), then the channel (n+1) output is always the inactive value (POL(n+1) bit value).

Although, in most cases the deadtime delay is not comparable to channels (n) and (n+1) duty cycle, the following figures show examples where the deadtime delay is comparable to the duty cycle.

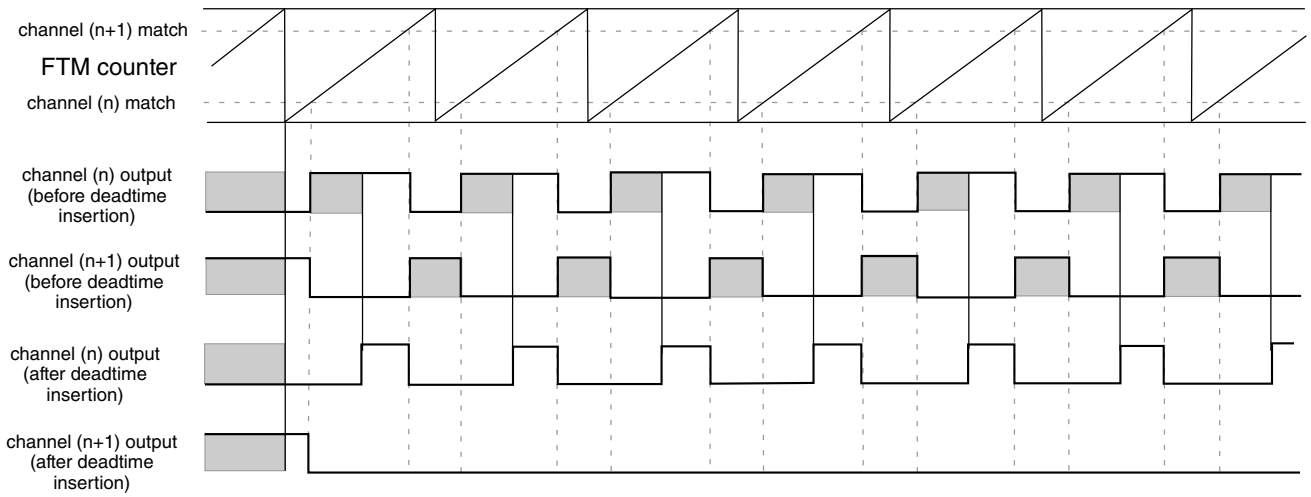


Figure 39-69. Example of the deadtime insertion (ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channel (n+1) duty cycle

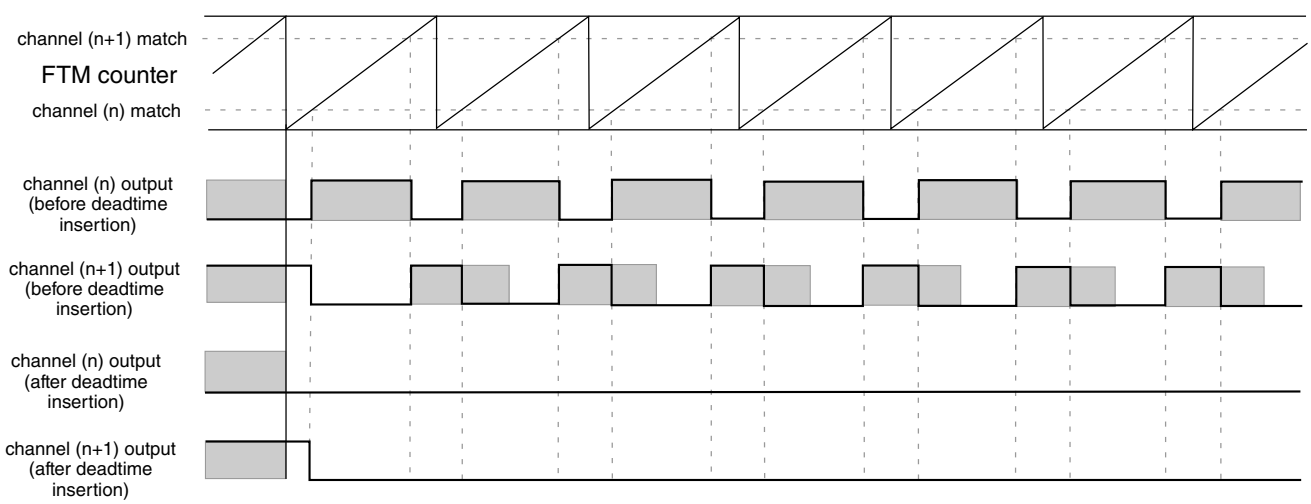


Figure 39-70. Example of the deadtime insertion (ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channels (n) and (n+1) duty cycle

39.5.15 Output mask

The output mask can be used to force channels output to their inactive state through software. For example: to control a BLDC motor.

Any write to the OUTMASK register updates its write buffer. The OUTMASK register is updated with its buffer value by PWM synchronization; see OUTMASK register synchronization.

If CHnOM = 1, then the channel (n) output is forced to its inactive state (POLn bit value). If CHnOM = 0, then the channel (n) output is unaffected by the output mask. See the following figure.

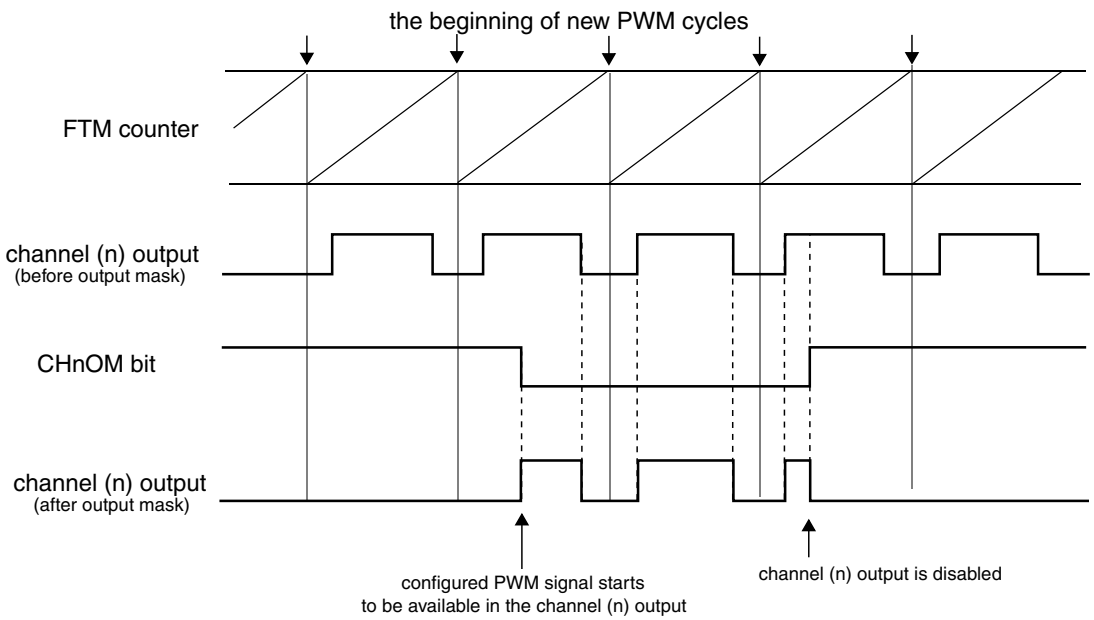


Figure 39-71. Output mask with POLn = 0

The following table shows the output mask result before the polarity control.

Table 39-11. Output mask result for channel (n) before the polarity control

CHnOM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

39.5.16 Fault control

The fault control is enabled if (FAULTM[1:0] ≠ 0:0).

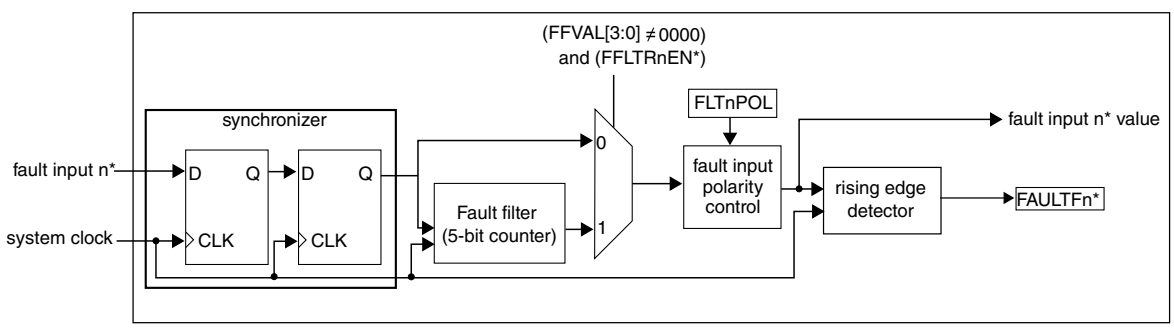
FTM can have up to four fault inputs. FAULTnEN bit (where n = 0, 1, 2, 3) enables the fault input n and FFLTRnEN bit enables the fault input n filter. FFVAL[3:0] bits select the value of the enabled filter in each enabled fault input.

First, each fault input signal is synchronized by the system clock; see the synchronizer block in the following figure. Following synchronization, the fault input n signal enters the filter block. When there is a state change in the fault input n signal, the 5-bit counter is reset and starts counting up. As long as the new state is stable on the fault input n, the counter continues to increment. If the 5-bit counter overflows, that is, the counter exceeds the value of the FFVAL[3:0] bits, the new fault input n value is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the fault input n signal before validation (counter overflow), the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by FFVAL[3:0] bits (× system clock) is regarded as a glitch and is not passed on to the edge detector.

The fault input n filter is disabled when the FFVAL[3:0] bits are zero or when FAULTnEN = 0. In this case, the fault input n signal is delayed 2 rising edges of the system clock and the FAULTFn bit is set on 3th rising edge of the system clock after a rising edge occurs on the fault input n.

If FFVAL[3:0] ≠ 0000 and FAULTnEN = 1, then the fault input n signal is delayed (3 + FFVAL[3:0]) rising edges of the system clock, that is, the FAULTFn bit is set (4 + FFVAL[3:0]) rising edges of the system clock after a rising edge occurs on the fault input n.



* where n = 3, 2, 1, 0

Figure 39-72. Fault input n control block diagram

If the fault control and fault input n are enabled and a rising edge at the fault input n signal is detected, a fault condition has occurred and the FAULTFn bit is set. The FAULTF bit is the logic OR of FAULTFn[3:0] bits. See the following figure.

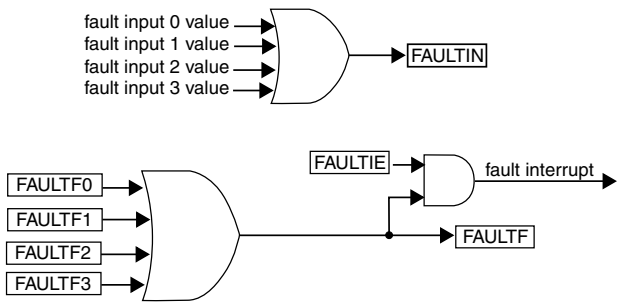


Figure 39-73. FAULTF and FAULTIN bits and fault interrupt

If the fault control is enabled (FAULTM[1:0] ≠ 0:0), a fault condition has occurred and (FAULTEN = 1), then outputs are forced to their safe values:

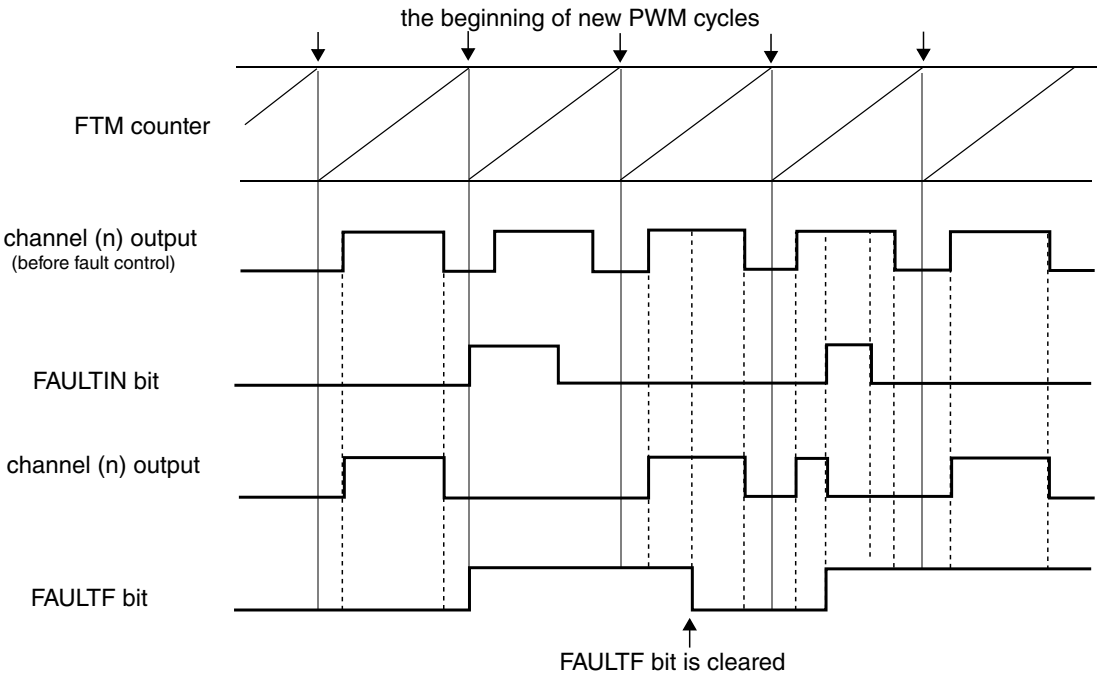
- Channel (n) output takes the value of POL(n)
- Channel (n+1) takes the value of POL(n+1)

The fault interrupt is generated when (FAULTF = 1) and (FAULTIE = 1). This interrupt request remains set until:

- Software clears the FAULTF bit by reading FAULTF bit as 1 and writing 0 to it
- Software clears the FAULTIE bit
- A reset occurs

39.5.16.1 Automatic fault clearing

If the automatic fault clearing is selected (FAULTM[1:0] = 1:1), then the channels output disabled by fault control is again enabled when the fault input signal (FAULTIN) returns to zero and a new PWM cycle begins. See the following figure.

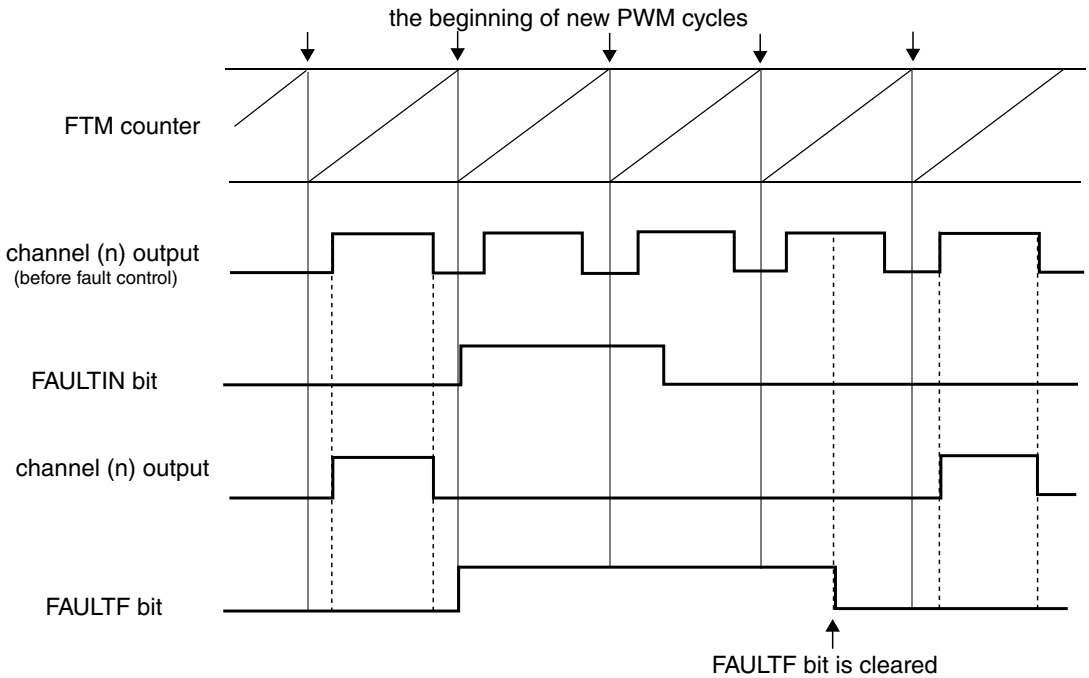


NOTE
The channel (n) output is after the fault control with automatic fault clearing and POLn = 0.

Figure 39-74. Fault control with automatic fault clearing

39.5.16.2 Manual fault clearing

If the manual fault clearing is selected (FAULTM[1:0] = 0:1 or 1:0), then the channels output disabled by fault control is again enabled when the FAULTF bit is cleared and a new PWM cycle begins. See the following figure.



NOTE
The channel (n) output is after the fault control with manual fault clearing and POLn = 0.

Figure 39-75. Fault control with manual fault clearing

39.5.16.3 Fault inputs polarity control

The FLTjPOL bit selects the fault input j polarity, where j = 0, 1, 2, 3:

- If FLTjPOL = 0, the fault j input polarity is high, so the logical one at the fault input j indicates a fault.
- If FLTjPOL = 1, the fault j input polarity is low, so the logical zero at the fault input j indicates a fault.

39.5.17 Polarity control

The POLn bit selects the channel (n) output polarity:

- If POLn = 0, the channel (n) output polarity is high, so the logical one is the active state and the logical zero is the inactive state.
- If POLn = 1, the channel (n) output polarity is low, so the logical zero is the active state and the logical one is the inactive state.

39.5.18 Initialization

The initialization forces the CHnOI bit value to the channel (n) output when a one is written to the INIT bit.

The initialization depends on COMP and DTEN bits. The following table shows the values that channels (n) and (n+1) are forced by initialization when the COMP and DTEN bits are zero.

Table 39-12. Initialization behavior when (COMP = 0 and DTEN = 0)

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	0	is forced to zero	is forced to zero
0	1	is forced to zero	is forced to one
1	0	is forced to one	is forced to zero
1	1	is forced to one	is forced to one

The following table shows the values that channels (n) and (n+1) are forced by initialization when (COMP = 1) or (DTEN = 1).

Table 39-13. Initialization behavior when (COMP = 1 or DTEN = 1)

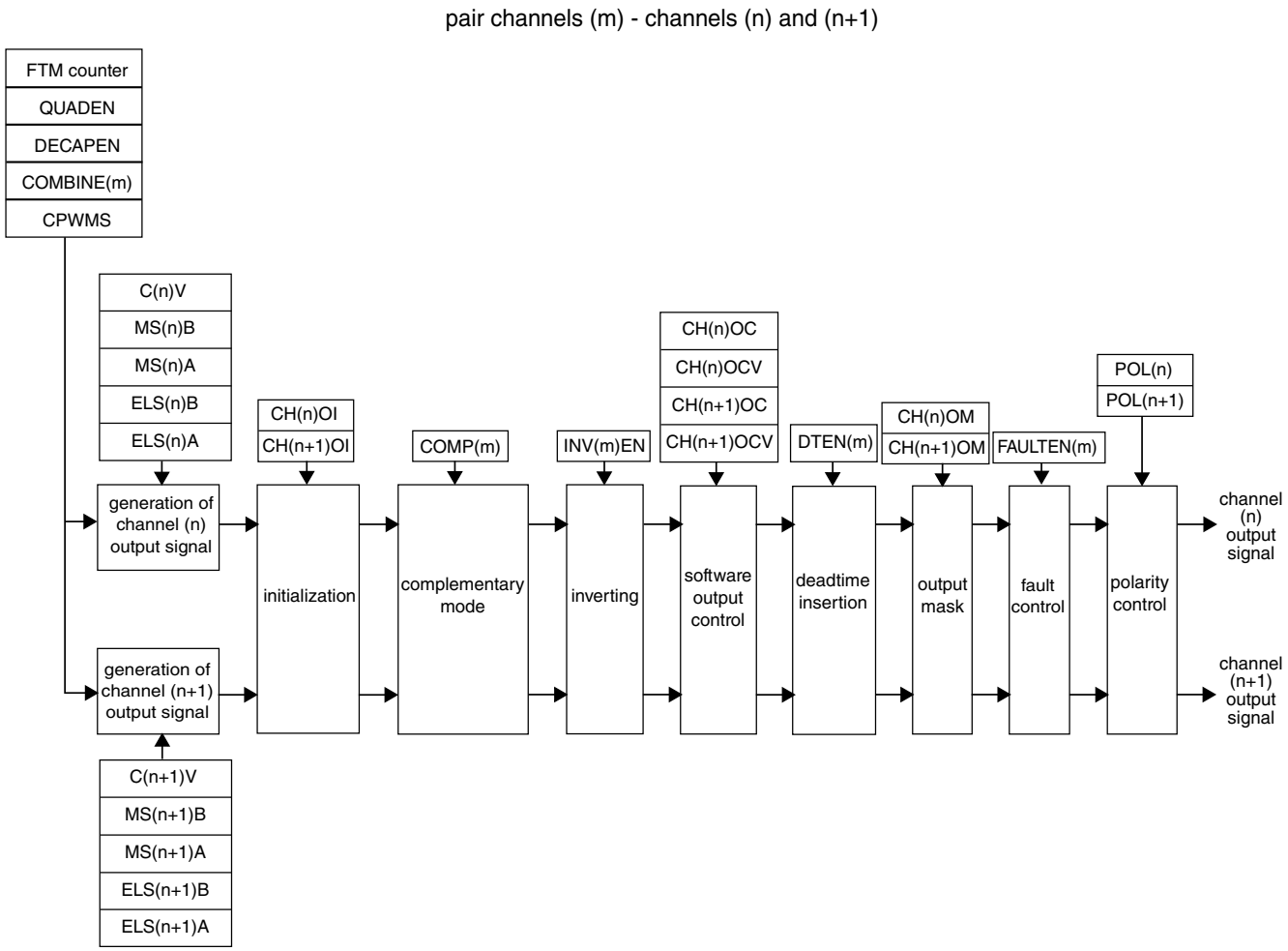
CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	X	is forced to zero	is forced to one
1	X	is forced to one	is forced to zero

Note

The initialization feature must be used only with disabled FTM counter. See the description of the CLKS field in the Status and Control register.

39.5.19 Features priority

The following figure shows the priority of the features used at the generation of channels (n) and (n+1) outputs signals.



NOTE

The channels (n) and (n+1) are in output compare, EPWM, CPWM or combine modes.

Figure 39-76. Priority of the features used at the generation of channels (n) and (n+1) outputs signals

Note

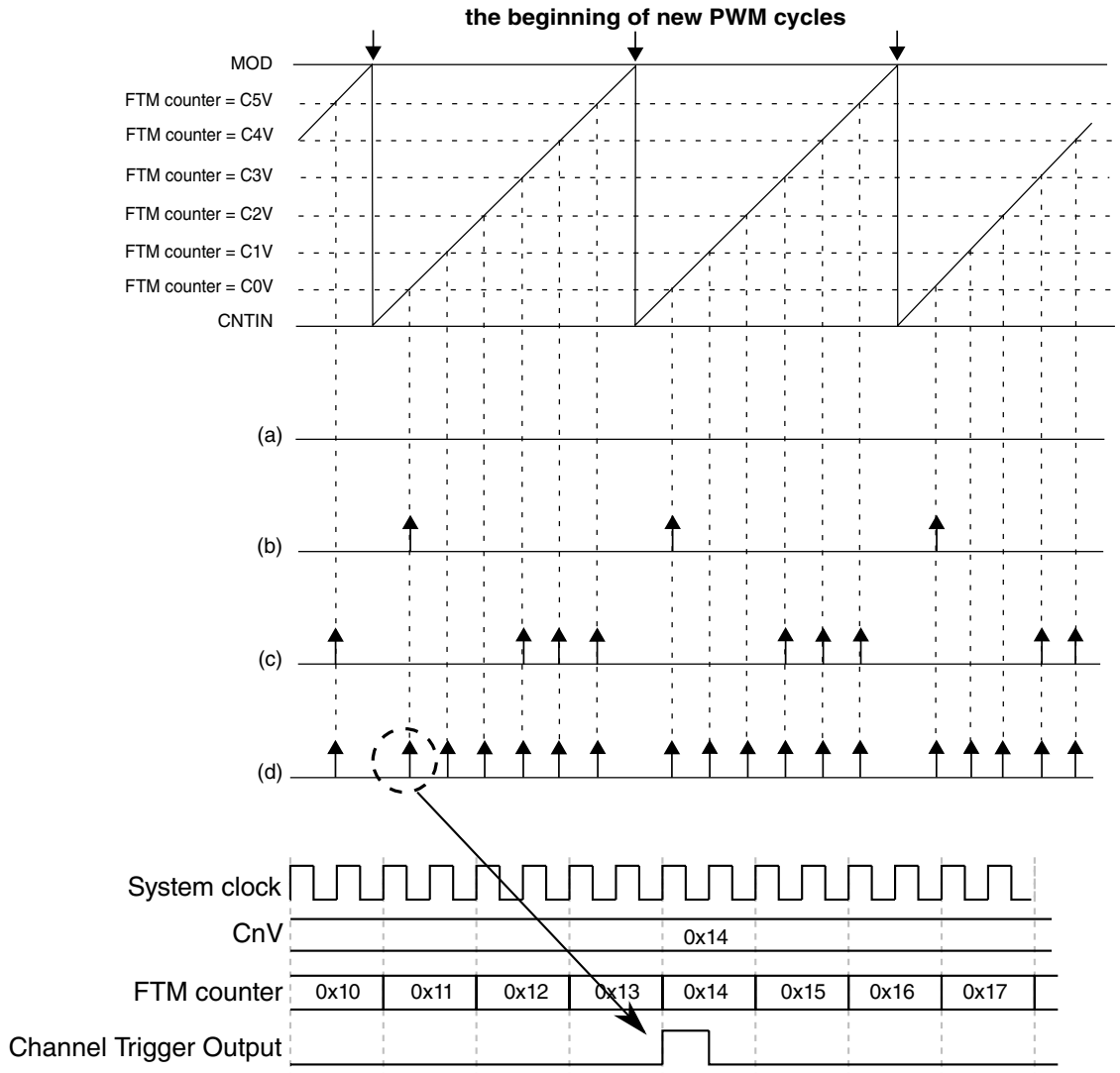
The Initialization feature must not be used with Inverting and Software output control features.

39.5.20 Channel trigger output

If CH(j)TRIG bit of the FTM External Trigger (FTM_EXTTRIG) register is set, where j = 0, 1, 2, 3, 4, or 5, then the FTM generates a trigger when the channel (j) match occurs (FTM counter = C(j)V).

The channel trigger output provides a trigger signal which has one FTM clock period width and is used for on-chip modules.

The FTM is able to generate multiple triggers in one PWM period. Because each trigger is generated for a specific channel, several channels are required to implement this functionality. This behavior is described in the following figure.



NOTE

(a) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0
(b) CH0TRIG = 1, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0
(c) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1
(d) CH0TRIG = 1, CH1TRIG = 1, CH2TRIG = 1, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1

Figure 39-77. Channel match trigger

39.5.21 Initialization trigger

If INITTRIGEN = 1, then the FTM generates a trigger when the FTM counter is updated with the CNTIN register value in the following cases.

- The FTM counter is automatically updated with the CNTIN register value by the selected counting mode.
- When there is a write to CNT register.
- When there is the FTM counter synchronization.
- If (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits.
- If the channel (n) is in Input Capture mode, (ICRST = 1) and the selected input capture event occurs in the channel (n) input.

The following figures show these cases.

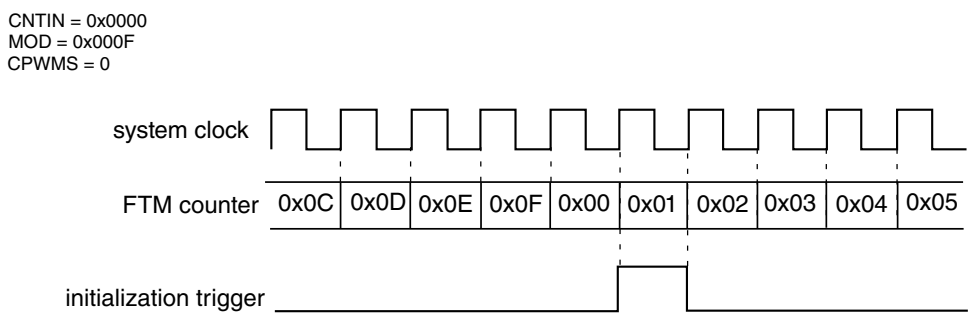


Figure 39-78. Initialization trigger is generated when the FTM counting achieves the CNTIN register value

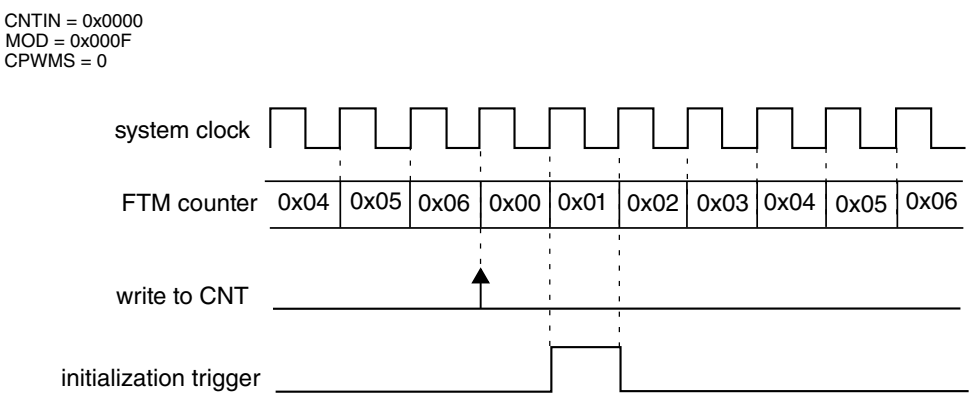


Figure 39-79. Initialization trigger is generated when there is a write to CNT register

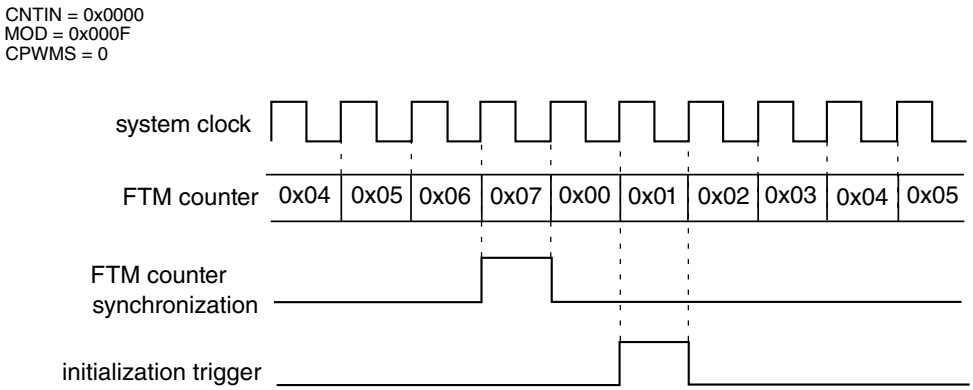


Figure 39-80. Initialization trigger is generated when there is the FTM counter synchronization

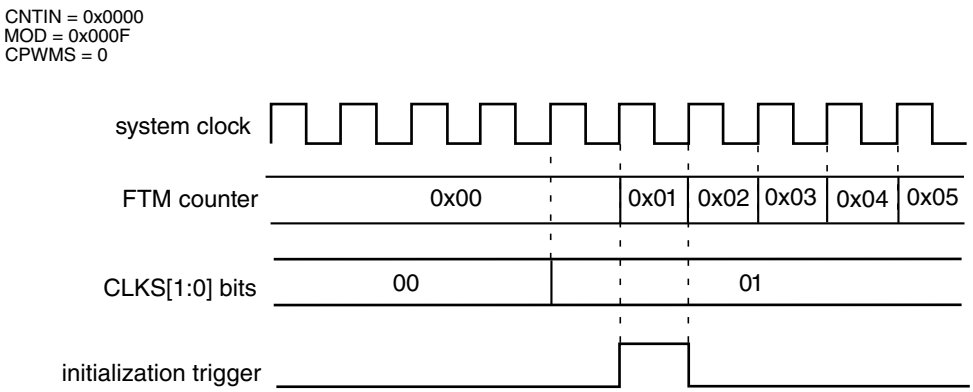
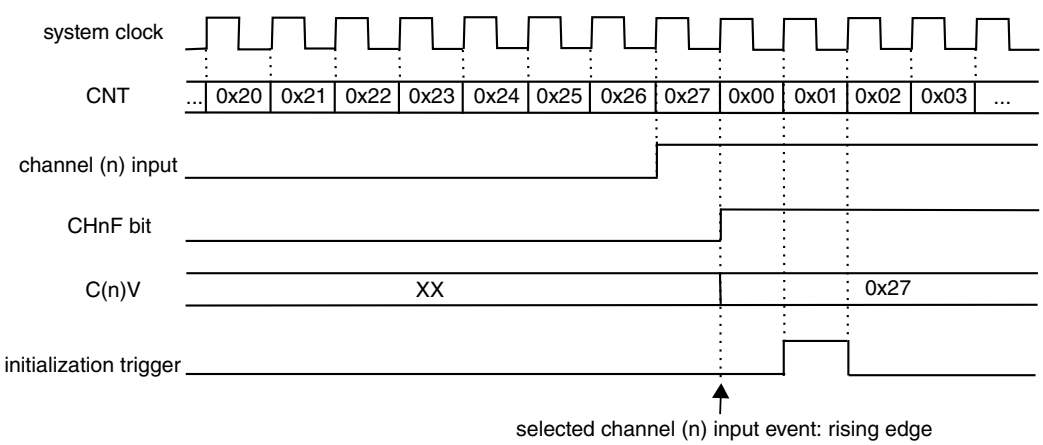


Figure 39-81. Initialization trigger is generated if (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits



NOTE
Channel (n) input after its synchronizer and filter
MOD = 0xFFFF
CNTIN = 0x0000
PS[2:0] = 3'b000
ICRST = 1'b1

Figure 39-82. Initialization trigger is generated if the channel (n) is in Input Capture mode, ICRST = 1 and the selected input capture event occurs in the channel (n) input

The initialization trigger output provides a trigger signal that is used for on-chip modules.

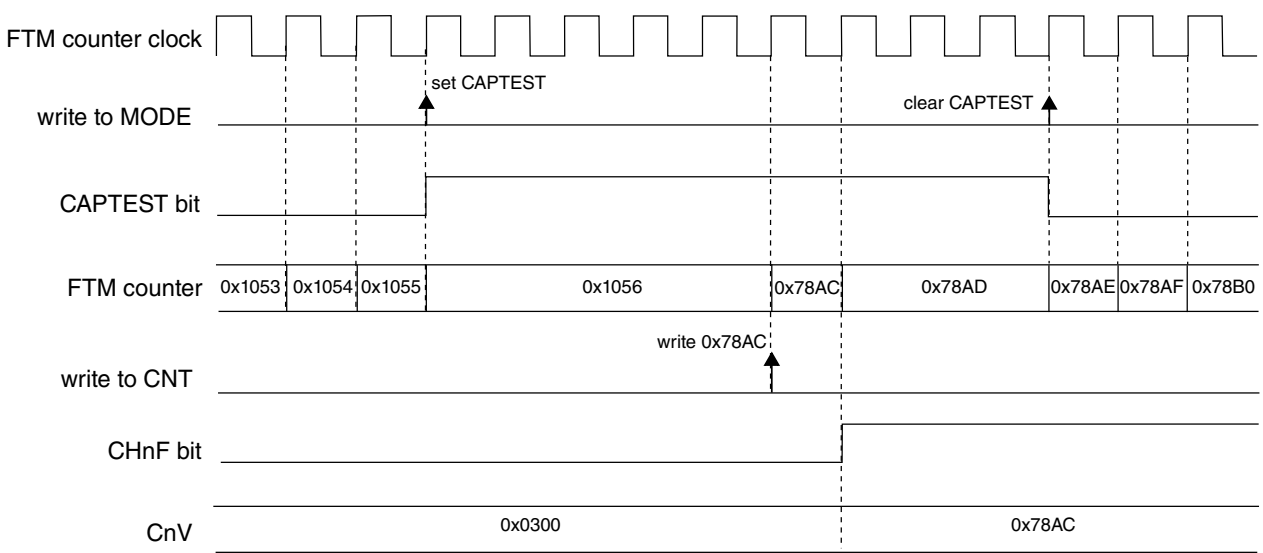
39.5.22 Capture Test mode

The Capture Test mode allows to test the CnV registers, the FTM counter and the interconnection logic between the FTM counter and CnV registers.

In this test mode, all channels must be configured for Input Capture mode and FTM counter must be configured to the Up counting.

When the Capture Test mode is enabled (CAPTEST = 1), the FTM counter is frozen and any write to CNT register updates directly the FTM counter; see the following figure. After it was written, all CnV registers are updated with the written value to CNT register and CHnF bits are set. Therefore, the FTM counter is updated with its next value according to its configuration. Its next value depends on CNTIN, MOD, and the written value to FTM counter.

The next reads of CnV registers return the written value to the FTM counter and the next reads of CNT register return FTM counter next value.



NOTE
- FTM counter configuration: (FTMEN = 1), (QUADEN = 0), (CAPTEST = 1), (CPWMS = 0), (CNTIN = 0x0000), and (MOD = 0xFFFF)
- FTM channel n configuration: input capture mode - (DECAPEN = 0), (COMBINE = 0), and (MSnB:MSnA = 0:0)

Figure 39-83. Capture Test mode

39.5.23 DMA

The channel generates a DMA transfer request according to DMA and CHnIE bits. See the following table.

Table 39-14. Channel DMA transfer request

DMA	CHnIE	Channel DMA Transfer Request	Channel Interrupt
0	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
0	1	The channel DMA transfer request is not generated.	The channel interrupt is generated if (CHnF = 1).
1	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
1	1	The channel DMA transfer request is generated if (CHnF = 1).	The channel interrupt is not generated.

If DMA = 1, the CHnF bit is cleared either by channel DMA transfer done or reading CnSC while CHnF is set and then writing a zero to CHnF bit according to CHnIE bit. See the following table.

Table 39-15. Clear CHnF bit when DMA = 1

CHnIE	How CHnF Bit Can Be Cleared
0	CHnF bit is cleared either when the channel DMA transfer is done or by reading CnSC while CHnF is set and then writing a 0 to CHnF bit.
1	CHnF bit is cleared when the channel DMA transfer is done.

39.5.24 Dual Edge Capture mode

The Dual Edge Capture mode is selected if DECAPEN = 1. This mode allows to measure a pulse width or period of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode when n is 0 or 2.

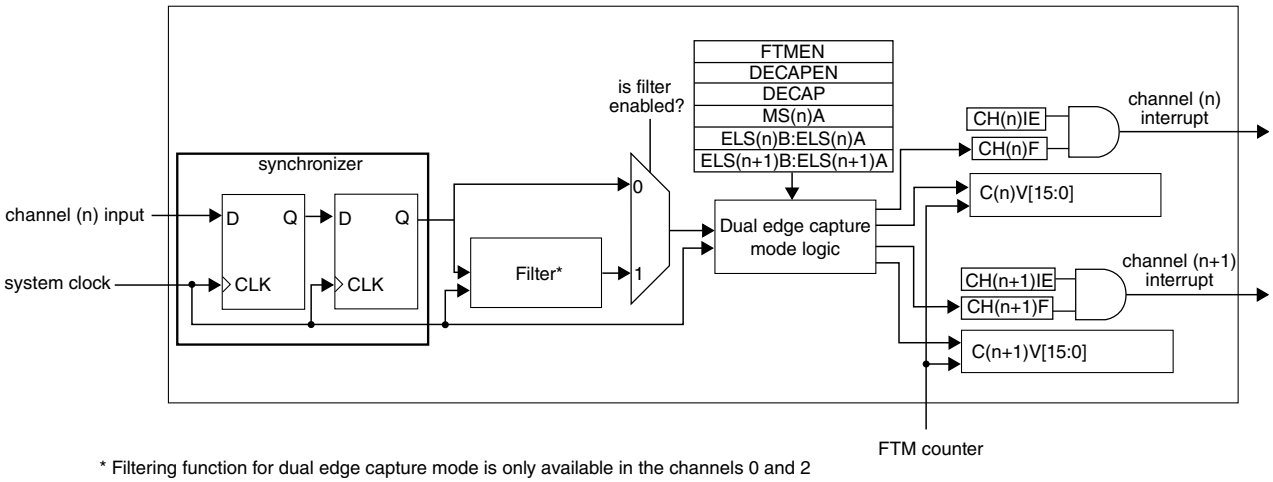


Figure 39-84. Dual Edge Capture mode block diagram

The MS(n)A bit defines if the Dual Edge Capture mode is one-shot or continuous.

The ELS(n)B:ELS(n)A bits select the edge that is captured by channel (n), and ELS(n +1)B:ELS(n+1)A bits select the edge that is captured by channel (n+1). If both ELS(n)B:ELS(n)A and ELS(n+1)B:ELS(n+1)A bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

In the Dual Edge Capture mode, only channel (n) input is used and channel (n+1) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then CH(n)F bit is set and the channel (n) interrupt is generated (if CH(n)IE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input and (CH(n)F = 1), then CH(n+1)F bit is set and the channel (n+1) interrupt is generated (if CH(n+1)IE = 1).

The C(n)V register stores the value of FTM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of FTM counter when the selected edge by channel (n+1) is detected at channel (n) input.

In this mode, a coherency mechanism ensures coherent data when the C(n)V and C(n +1)V registers are read. The only requirement is that C(n)V must be read before C(n +1)V.

Note

- The CH(n)F, CH(n)IE, MS(n)A, ELS(n)B, and ELS(n)A bits are channel (n) bits.

- The CH(n+1)F, CH(n+1)IE, MS(n+1)A, ELS(n+1)B, and ELS(n+1)A bits are channel (n+1) bits.
- The Dual Edge Capture mode must be used with ELS(n)B:ELS(n)A = 0:1 or 1:0, ELS(n+1)B:ELS(n+1)A = 0:1 or 1:0 and the FTM counter in Free running counter.

39.5.24.1 One-Shot Capture mode

The One-Shot Capture mode is selected when (DECAPEN = 1), and (MS(n)A = 0). In this capture mode, only one pair of edges at the channel (n) input is captured. The ELS(n)B:ELS(n)A bits select the first edge to be captured, and ELS(n+1)B:ELS(n+1)A bits select the second edge to be captured.

The edge captures are enabled while DECAP bit is set. For each new measurement in One-Shot Capture mode, first the CH(n)F and CH(n+1) bits must be cleared, and then the DECAP bit must be set.

In this mode, the DECAP bit is automatically cleared by FTM when the edge selected by channel (n+1) is captured. Therefore, while DECAP bit is set, the one-shot capture is in process. When this bit is cleared, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

Similarly, when the CH(n+1)F bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

39.5.24.2 Continuous Capture mode

The Continuous Capture mode is selected when (DECAPEN = 1), and (MS(n)A = 1). In this capture mode, the edges at the channel (n) input are captured continuously. The ELS(n)B:ELS(n)A bits select the initial edge to be captured, and ELS(n+1)B:ELS(n+1)A bits select the final edge to be captured.

The edge captures are enabled while DECAP bit is set. For the initial use, first the CH(n)F and CH(n+1)F bits must be cleared, and then DECAP bit must be set to start the continuous measurements.

When the CH(n+1)F bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers. The latest captured values are always available in these registers even after the DECAP bit is cleared.

In this mode, it is possible to clear only the CH(n+1)F bit. Therefore, when the CH(n+1)F bit is set again, the latest captured values are available in C(n)V and C(n+1)V registers.

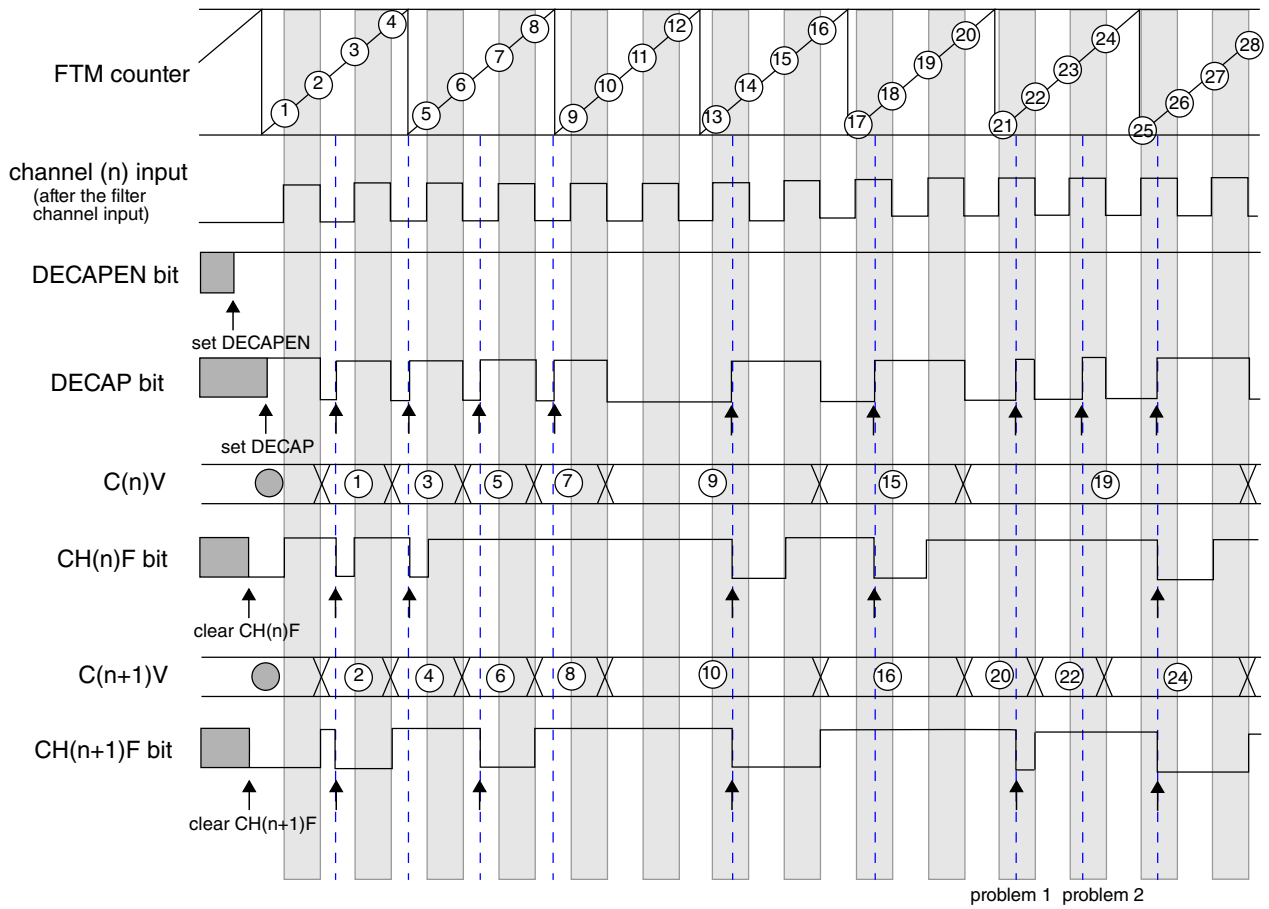
For a new sequence of the measurements in the Dual Edge Capture – Continuous mode, clear the CH(n)F and CH(n+1)F bits to start new measurements.

39.5.24.3 Pulse width measurement

If the channel (n) is configured to capture rising edges (ELS(n)B:ELS(n)A = 0:1) and the channel (n+1) to capture falling edges (ELS(n+1)B:ELS(n+1)A = 1:0), then the positive polarity pulse width is measured. If the channel (n) is configured to capture falling edges (ELS(n)B:ELS(n)A = 1:0) and the channel (n+1) to capture rising edges (ELS(n +1)B:ELS(n+1)A = 0:1), then the negative polarity pulse width is measured.

The pulse width measurement can be made in One-Shot Capture mode or Continuous Capture mode.

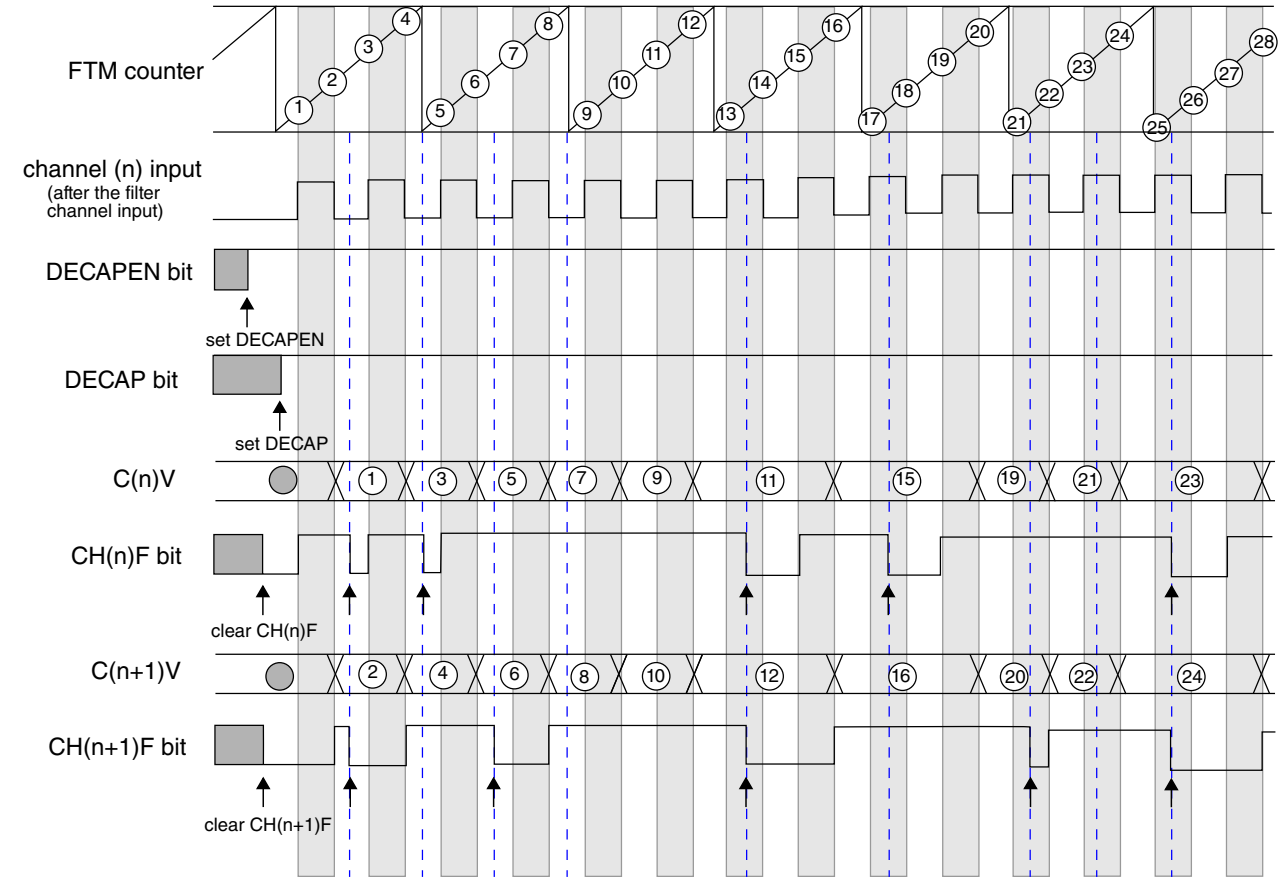
The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next positive polarity pulse width. The CH(n)F bit is set when the first edge of this pulse is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set and DECAP bit is cleared when the second edge of this pulse is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. Both DECAP and CH(n+1)F bits indicate when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.
- Problem 1: channel (n) input = 1, set DECAP, not clear CH(n)F, and clear CH(n+1)F.
- Problem 2: channel (n) input = 1, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.

Figure 39-85. Dual Edge Capture – One-Shot mode for positive polarity pulse width measurement

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The CH(n)F bit is set when the first edge of the positive polarity pulse is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set when the second edge of this pulse is detected, that is, the edge selected by ELS(n +1)B:ELS(n+1)A bits. The CH(n+1)F bit indicates when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.

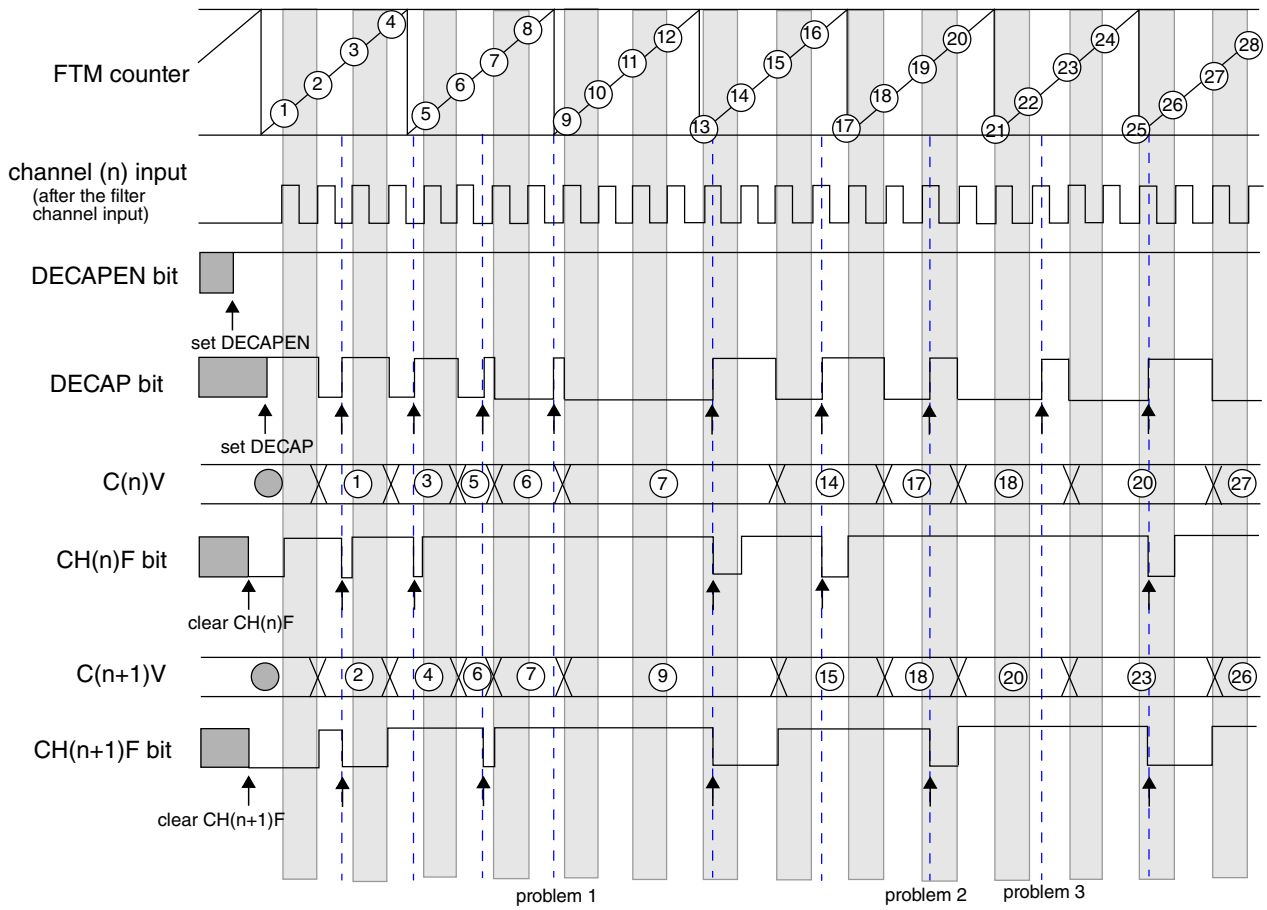
Figure 39-86. Dual Edge Capture – Continuous mode for positive polarity pulse width measurement

39.5.24.4 Period measurement

If the channels (n) and (n+1) are configured to capture consecutive edges of the same polarity, then the period of the channel (n) input signal is measured. If both channels (n) and (n+1) are configured to capture rising edges (ELS(n)B:ELS(n)A = 0:1 and ELS(n +1)B:ELS(n+1)A = 0:1), then the period between two consecutive rising edges is measured. If both channels (n) and (n+1) are configured to capture falling edges (ELS(n)B:ELS(n)A = 1:0 and ELS(n+1)B:ELS(n+1)A = 1:0), then the period between two consecutive falling edges is measured.

The period measurement can be made in One-Shot Capture mode or Continuous Capture mode.

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next period. The CH(n)F bit is set when the first rising edge is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set and DECAP bit is cleared when the second rising edge is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. Both DECAP and CH(n+1)F bits indicate when two selected edges were captured and the C(n)V and C(n+1)V registers are ready for reading.

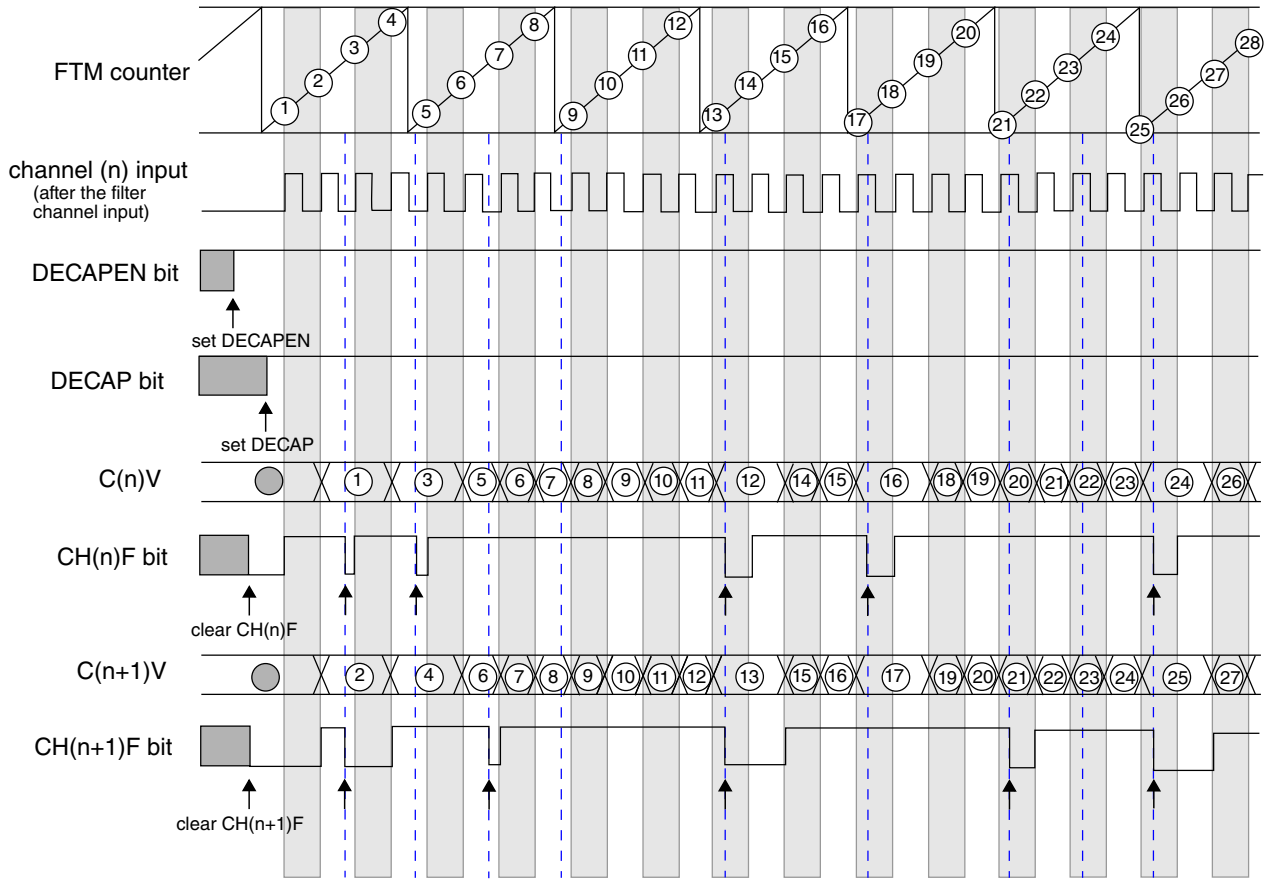


Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.
- Problem 1: channel (n) input = 0, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.
- Problem 2: channel (n) input = 1, set DECAP, not clear CH(n)F, and clear CH(n+1)F.
- Problem 3: channel (n) input = 1, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.

Figure 39-87. Dual Edge Capture – One-Shot mode to measure of the period between two consecutive rising edges

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The CH(n)F bit is set when the first rising edge is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set

when the second rising edge is detected, that is, the edge selected by ELS(n+1)B:ELS(n +1)A bits. The CH(n+1)F bit indicates when two edges of the period were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.

Figure 39-88. Dual Edge Capture – Continuous mode to measure of the period between two consecutive rising edges

39.5.24.5 Read coherency mechanism

The Dual Edge Capture mode implements a read coherency mechanism between the FTM counter value captured in C(n)V and C(n+1)V registers. The read coherency mechanism is illustrated in the following figure. In this example, the channels (n) and (n +1) are in Dual Edge Capture – Continuous mode for positive polarity pulse width measurement. Thus, the channel (n) is configured to capture the FTM counter value when there is a rising edge at channel (n) input signal, and channel (n+1) to capture the FTM counter value when there is a falling edge at channel (n) input signal.

When a rising edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n) capture buffer. The channel (n) capture buffer value is transferred to C(n)V register when a falling edge occurs in the channel (n) input signal. C(n)V register has the FTM counter value when the previous rising edge occurred, and the channel (n) capture buffer has the FTM counter value when the last rising edge occurred.

When a falling edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n+1) capture buffer. The channel (n+1) capture buffer value is transferred to C(n+1)V register when the C(n)V register is read.

In the following figure, the read of C(n)V returns the FTM counter value when the event 1 occurred and the read of C(n+1)V returns the FTM counter value when the event 2 occurred.

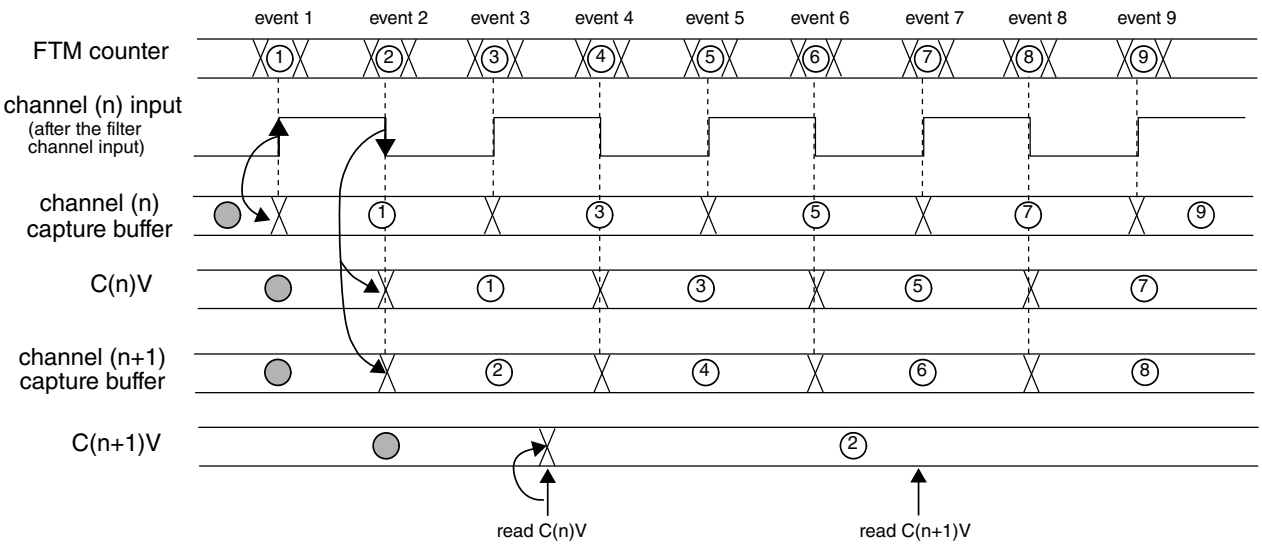


Figure 39-89. Dual Edge Capture mode read coherency mechanism

C(n)V register must be read prior to C(n+1)V register in dual edge capture one-shot and continuous modes for the read coherency mechanism works properly.

39.5.25 Quadrature Decoder mode

The Quadrature Decoder mode is selected if (QUADEN = 1). The Quadrature Decoder mode uses the input signals phase A and B to control the FTM counter increment and decrement. The following figure shows the quadrature decoder block diagram.

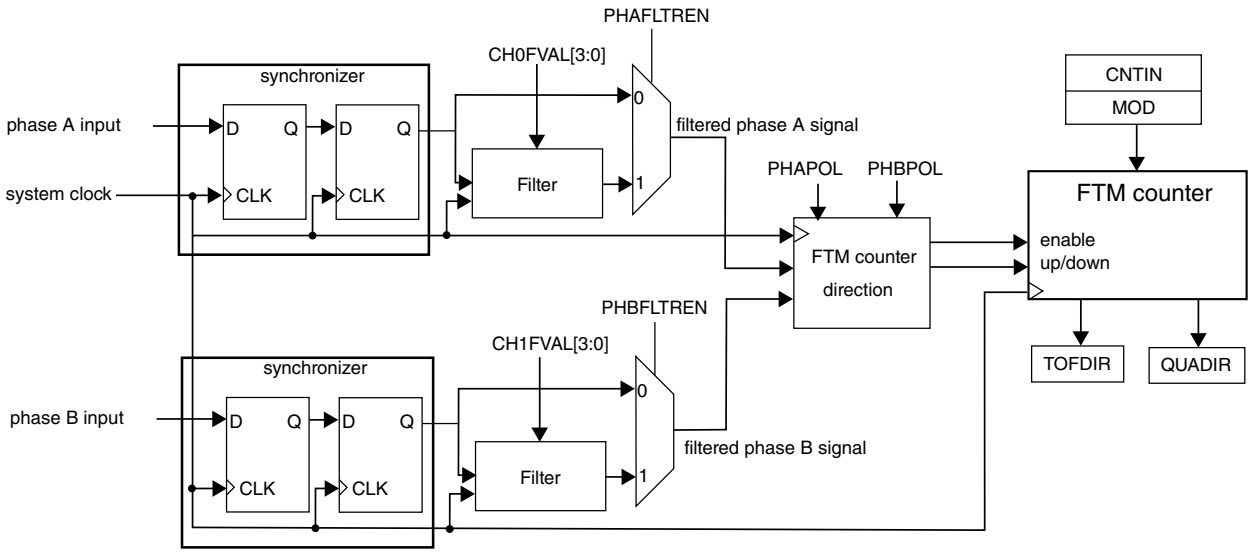


Figure 39-90. Quadrature Decoder block diagram

Each one of input signals phase A and B has a filter that is equivalent to the filter used in the channels input; Filter for Input Capture mode. The phase A input filter is enabled by PHAFLTREN bit and this filter's value is defined by CH0FVAL[3:0] bits (CH(n)FVAL[3:0] bits in FILTER0 register). The phase B input filter is enabled by PHBFLTREN bit and this filter's value is defined by CH1FVAL[3:0] bits (CH(n +1)FVAL[3:0] bits in FILTER0 register).

Except for CH0FVAL[3:0] and CH1FVAL[3:0] bits, no channel logic is used in Quadrature Decoder mode.

Note

Notice that the FTM counter is clocked by the phase A and B input signals when quadrature decoder mode is selected. Therefore it is expected that the Quadrature Decoder be used only with the FTM channels in input capture or output compare modes.

Note

An edge at phase A must not occur together an edge at phase B and vice-versa.

The PHAPOL bit selects the polarity of the phase A input, and the PHBPOL bit selects the polarity of the phase B input.

The QUADMODE selects the encoding mode used in the Quadrature Decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the phase B input value indicates the counting direction, and the phase A input defines the counting rate. The FTM counter is updated when there is a rising edge at phase A input signal.

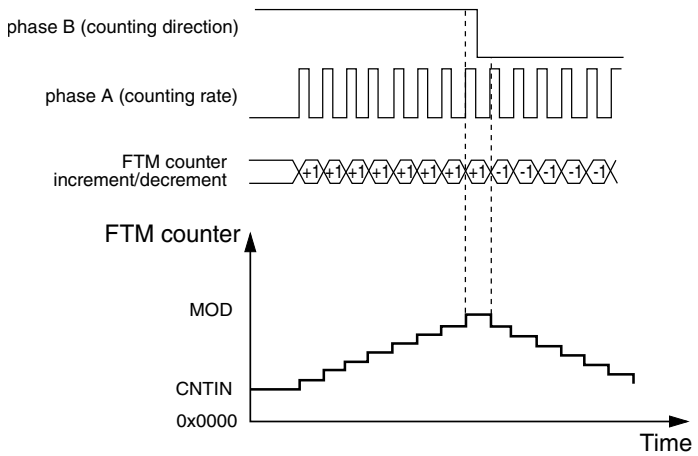


Figure 39-91. Quadrature Decoder – Count and Direction Encoding mode

If QUADMODE = 0, then the Phase A and Phase B Encoding mode is enabled; see the following figure. In this mode, the relationship between phase A and B signals indicates the counting direction, and phase A and B signals define the counting rate. The FTM counter is updated when there is an edge either at the phase A or phase B signals.

If PHAPOL = 0 and PHBPOL = 0, then the FTM counter increment happens when:

- there is a rising edge at phase A signal and phase B signal is at logic zero;
- there is a rising edge at phase B signal and phase A signal is at logic one;
- there is a falling edge at phase B signal and phase A signal is at logic zero;
- there is a falling edge at phase A signal and phase B signal is at logic one;

and the FTM counter decrement happens when:

- there is a falling edge at phase A signal and phase B signal is at logic zero;
- there is a falling edge at phase B signal and phase A signal is at logic one;
- there is a rising edge at phase B signal and phase A signal is at logic zero;
- there is a rising edge at phase A signal and phase B signal is at logic one.

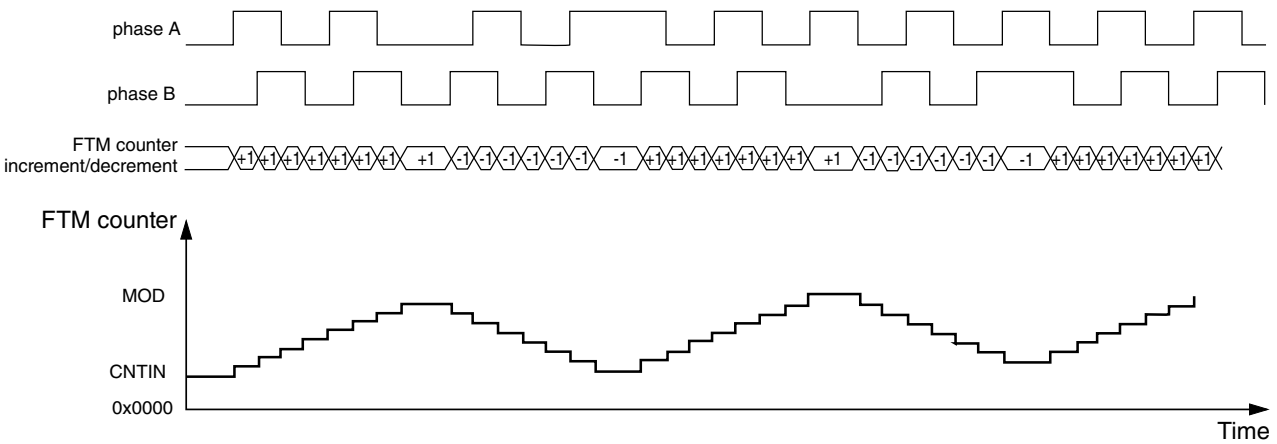


Figure 39-92. Quadrature Decoder – Phase A and Phase B Encoding mode

The following figure shows the FTM counter overflow in up counting. In this case, when the FTM counter changes from MOD to CNTIN, TOF and TOFDIR bits are set. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was up when the FTM counter overflow occurred.

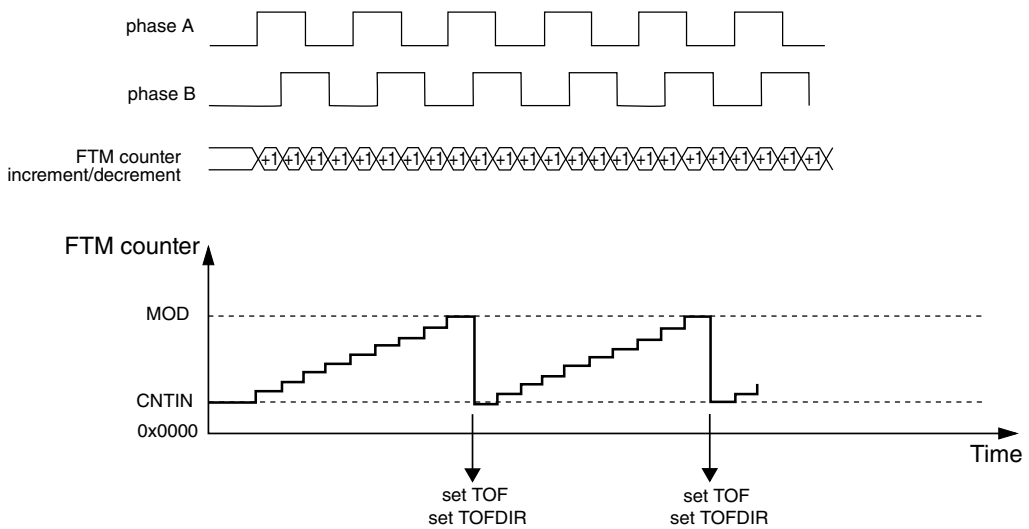


Figure 39-93. FTM Counter overflow in up counting for Quadrature Decoder mode

The following figure shows the FTM counter overflow in down counting. In this case, when the FTM counter changes from CNTIN to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was down when the FTM counter overflow occurred.

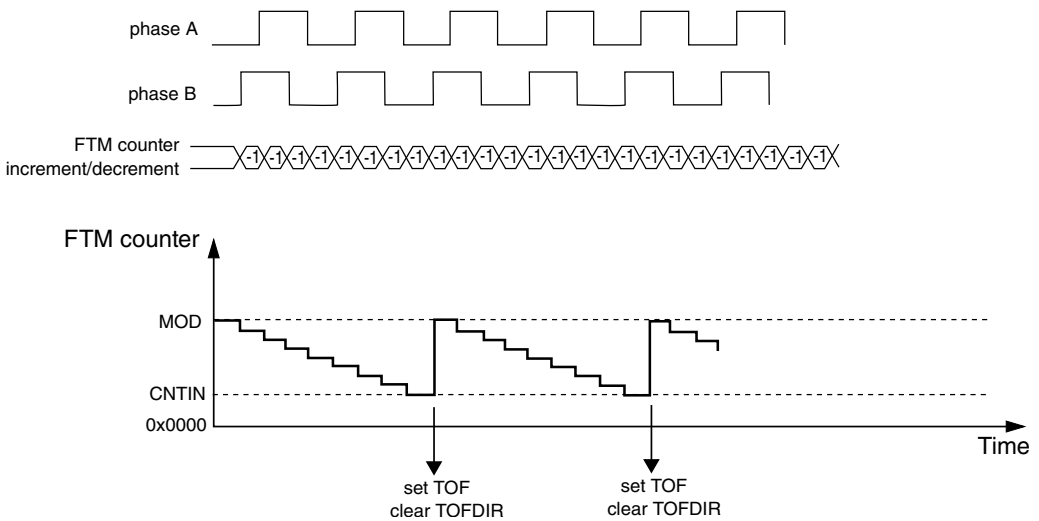


Figure 39-94. FTM counter overflow in down counting for Quadrature Decoder mode

39.5.25.1 Quadrature Decoder boundary conditions

The following figures show the FTM counter responding to motor jittering typical in motor position control applications.

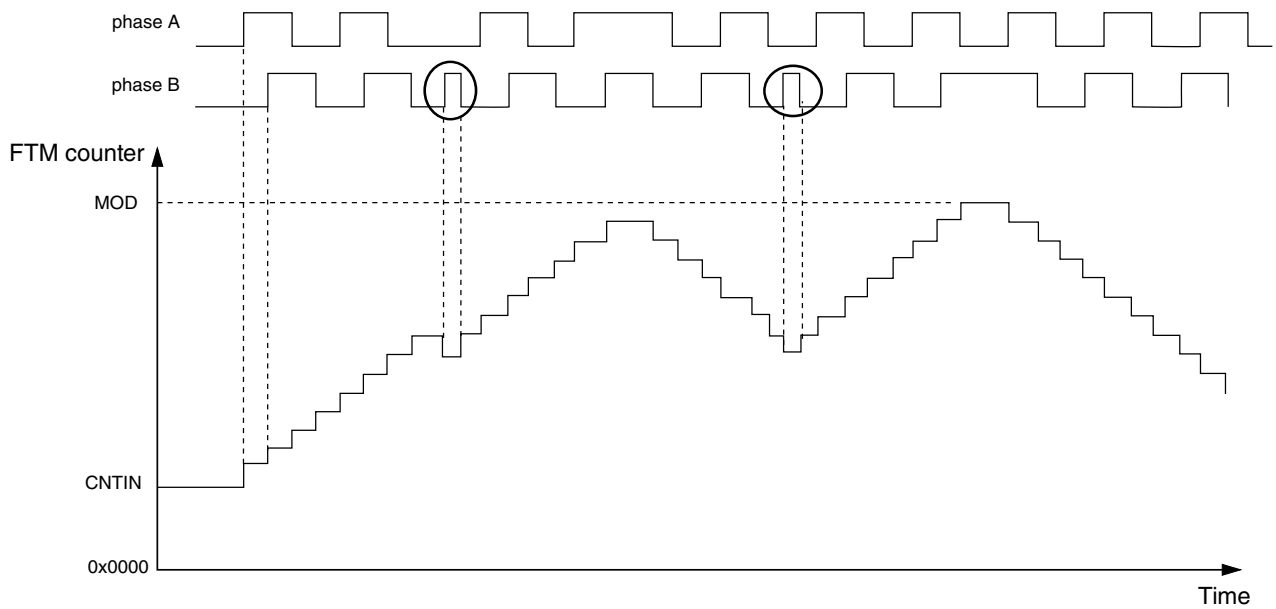


Figure 39-95. Motor position jittering in a mid count value

The following figure shows motor jittering produced by the phase B and A pulses respectively:

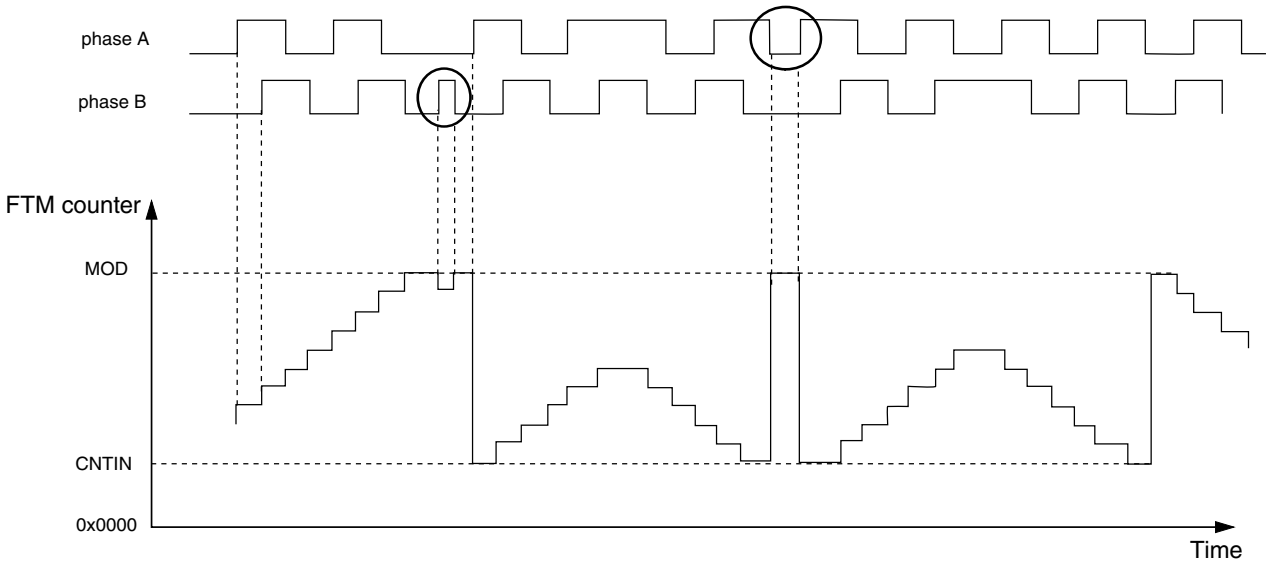


Figure 39-96. Motor position jittering near maximum and minimum count value

The first highlighted transition causes a jitter on the FTM counter value near the maximum count value (MOD). The second indicated transition occurs on phase A and causes the FTM counter transition between the maximum and minimum count values which are defined by MOD and CNTIN registers.

The appropriate settings of the phase A and phase B input filters are important to avoid glitches that may cause oscillation on the FTM counter value. The preceding figures show examples of oscillations that can be caused by poor input filter setup. Thus, it is important to guarantee a minimum pulse width to avoid these oscillations.

39.5.26 BDM mode

When the chip is in BDM mode, the BDMMODE[1:0] bits select the behavior of the FTM counter, the CH(n)F bit, the channels output, and the writes to the MOD, CNTIN, and C(n)V registers according to the following table.

Table 39-16. FTM behavior when the chip Is in BDM mode

BDMMODE	FTM Counter	CH(n)F Bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
00	Stopped	can be set	Functional mode	Writes to these registers bypass the registers buffers
01	Stopped	is not set	The channels outputs are forced to their safe value according to POLn bit	Writes to these registers bypass the registers buffers
10	Stopped	is not set	The channels outputs are frozen when the chip enters in BDM mode	Writes to these registers bypass the registers buffers

Table continues on the next page...

Table 39-16. FTM behavior when the chip Is in BDM mode (continued)

BDMMODE	FTM Counter	CH(n)F Bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
11	Functional mode	can be set	Functional mode	Functional mode

Note that if BDMMODE[1:0] = 2’b00 then the channels outputs remain at the value when the chip enters in BDM mode, because the FTM counter is stopped. However, the following situations modify the channels outputs in this BDM mode.

- Write any value to CNT register; see Counter reset. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for those channels set to Output Compare mode.
- FTM counter is reset by PWM Synchronization mode; see FTM counter synchronization. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for channels in Output Compare mode.
- In the channels outputs initialization, the channel (n) output is forced to the CH(n)OI bit value when the value 1 is written to INIT bit. See Initialization.

Note

The BDMMODE[1:0] = 2’b00 must not be used with the Fault control. Even if the fault control is enabled and a fault condition exists, the channels outputs values are updated as above.

Note

If CLKS[1:0] = 2'b00 in BDM, a non-zero value is written to CLKS in BDM, and CnV = CNTIN when the BDM is disabled, then the CHnF bit is set (since if the channel is a 0% EPWM signal) when the BDM is disabled.

39.5.27 Intermediate load

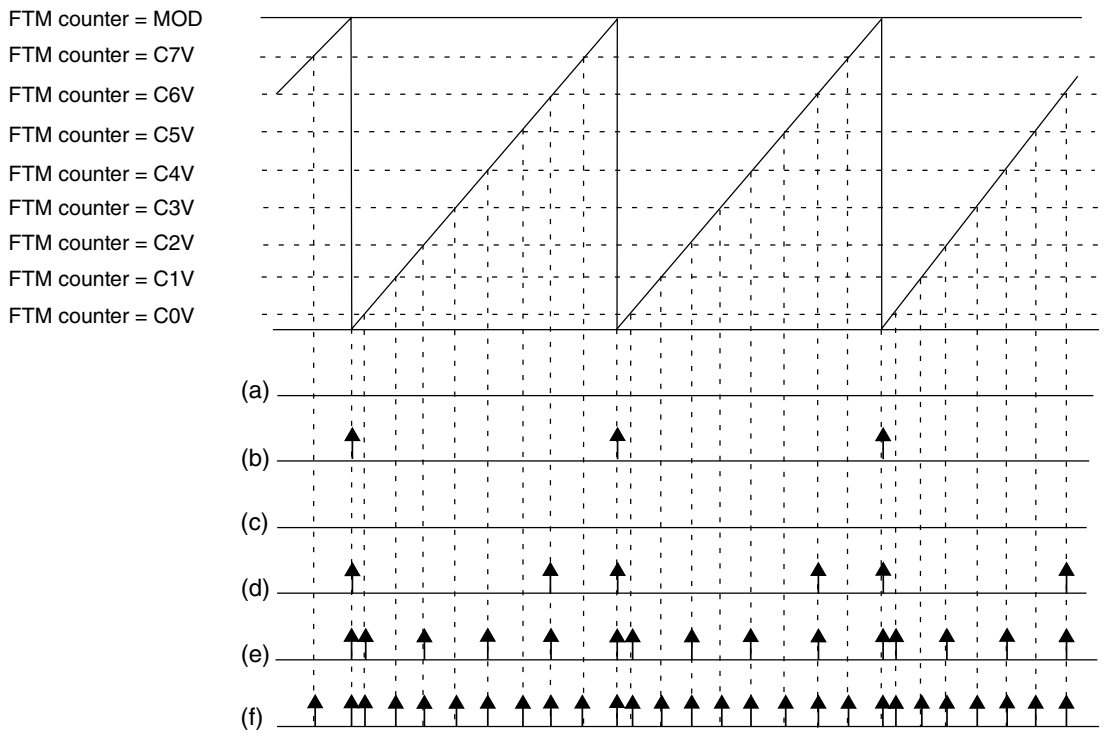
The PWMLOAD register allows to update the MOD, CNTIN, and C(n)V registers with the content of the register buffer at a defined load point. In this case, it is not required to use the PWM synchronization.

There are multiple possible loading points for intermediate load:

Table 39-17. When possible loading points are enabled

Loading point	Enabled
When the FTM counter wraps from MOD value to CNTIN value	Always
At the channel (j) match (FTM counter = C(j)V)	When CHjSEL = 1

The following figure shows some examples of enabled loading points.



NOTE

(a) LDOK = 0, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
(b) LDOK = 1, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
(c) LDOK = 0, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 1, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
(d) LDOK = 1, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 1, CH7SEL = 0
(e) LDOK = 1, CH0SEL = 1, CH1SEL = 0, CH2SEL = 1, CH3SEL = 0, CH4SEL = 1, CH5SEL = 0, CH6SEL = 1, CH7SEL = 0
(f) LDOK = 1, CH0SEL = 1, CH1SEL = 1, CH2SEL = 1, CH3SEL = 1, CH4SEL = 1, CH5SEL = 1, CH6SEL = 1, CH7SEL = 1

Figure 39-97. Loading points for intermediate load

After enabling the loading points, the LDOK bit must be set for the load to occur. In this case, the load occurs at the next enabled loading point according to the following conditions:

Table 39-18. Conditions for loads occurring at the next enabled loading point

When a new value was written	Then
To the MOD register	The MOD register is updated with its write buffer value.

Table continues on the next page...

Table 39-18. Conditions for loads occurring at the next enabled loading point (continued)

When a new value was written	Then
To the CNTIN register and CNTINC = 1	The CNTIN register is updated with its write buffer value.
To the C(n)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1)	The C(n)V register is updated with its write buffer value.
To the C(n+1)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1)	The C(n+1)V register is updated with its write buffer value.

NOTE

- If ELSjB and ELSjA bits are different from zero, then the channel (j) output signal is generated according to the configured output mode. If ELSjB and ELSjA bits are zero, then the generated signal is not available on channel (j) output.
- If CHjIE = 1, then the channel (j) interrupt is generated when the channel (j) match occurs.
- At the intermediate load neither the channels outputs nor the FTM counter are changed. Software must set the intermediate load at a safe point in time.

39.5.28 Global time base (GTB)

The global time base (GTB) is a FTM function that allows the synchronization of multiple FTM modules on a chip. The following figure shows an example of the GTB feature used to synchronize two FTM modules. In this case, the FTM A and B channels can behave as if just one FTM module was used, that is, a global time base.

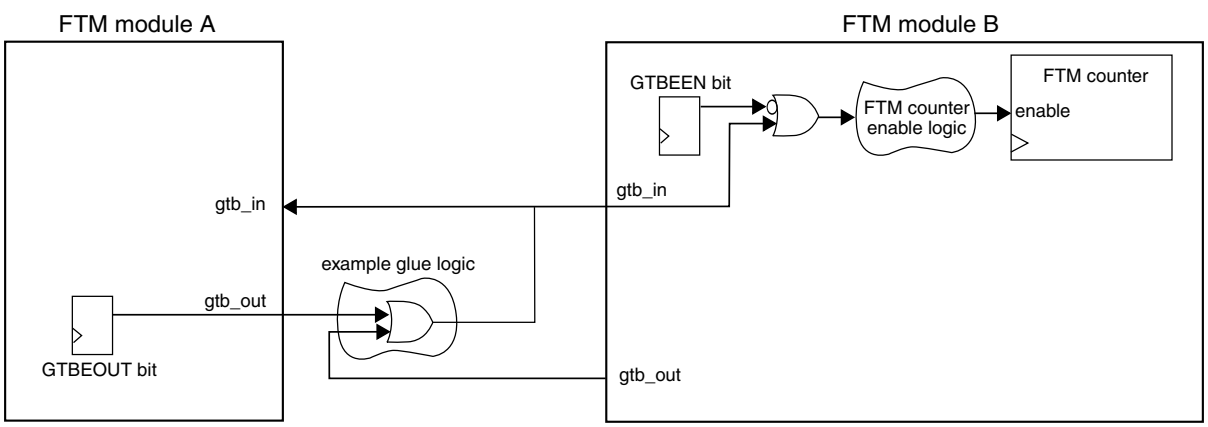


Figure 39-98. Global time base (GTB) block diagram

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the CONF register, the input signal *gtb_in*, and the output signal *gtb_out*. The GTBEEN bit enables gtb_in to control the FTM counter enable signal:

- If GTBEEN = 0, each one of FTM modules works independently according to their configured mode.
- If GTBEEN = 1, the FTM counter update is enabled only when gtb_in is 1.

In the configuration described in the preceding figure, FTM modules A and B have their FTM counters enabled if at least one of the gtb_out signals from one of the FTM modules is 1. There are several possible configurations for the interconnection of the gtb_in and gtb_out signals, represented by the example glue logic shown in the figure. Note that these configurations are chip-dependent and implemented outside of the FTM modules. See the chip-specific FTM information for the chip's specific implementation.

NOTE

- In order to use the GTB signals to synchronize the FTM counter of different FTM modules, the configuration of each FTM module should guarantee that its FTM counter starts counting as soon as the gtb_in signal is 1.
- The GTB feature does not provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during FTM operation. The GTB feature only allows the FTM counters to *start* their operation synchronously.

39.5.28.1 Enabling the global time base (GTB)

To enable the GTB feature, follow these steps for each participating FTM module:

1. Stop the FTM counter: Write 00b to SC[CLKS].
2. Program the FTM to the intended configuration. The FTM counter mode needs to be consistent across all participating modules.
3. Write 1 to CONF[GTBEEN] and write 0 to CONF[GTBEOUT] at the same time.
4. Select the intended FTM counter clock source in SC[CLKS]. The clock source needs to be consistent across all participating modules.
5. Reset the FTM counter: Write any value to the CNT register.

To initiate the GTB feature in the configuration described in the preceding figure, write 1 to CONF[GTBEOUT] in the FTM module used as the time base.

39.6 Reset overview

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

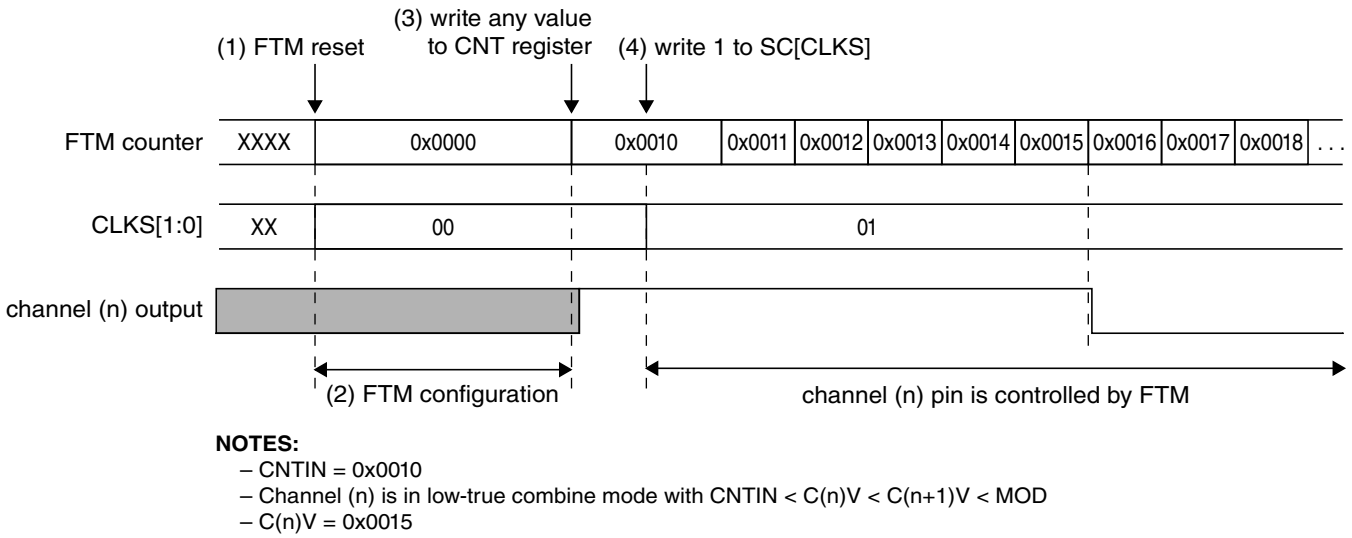
- the FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 00b);
- the timer overflow interrupt is zero, see Timer Overflow Interrupt;
- the channels interrupts are zero, see Channel (n) Interrupt;
- the fault interrupt is zero, see Fault Interrupt;
- the channels are in input capture mode, see Input Capture mode;
- the channels outputs are zero;
- the channels pins are not controlled by FTM (ELS(n)B:ELS(n)A = 0:0) (See the table in the description of CnSC register).

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (see the description of the CLKS field in the Status and Control register), its value is updated to zero and the pins are not controlled by FTM (See the table in the description of CnSC register).

After the reset, the FTM should be configurated (item 2). It is necessary to define the FTM counter mode, the FTM counting limits (MOD and CNTIN registers value), the channels mode and CnV registers value according to the channels mode.

Thus, it is recommended to write any value to CNT register (item 3). This write updates the FTM counter with the CNTIN register value and the channels output with its initial value (except for channels in output compare mode) (Counter reset).

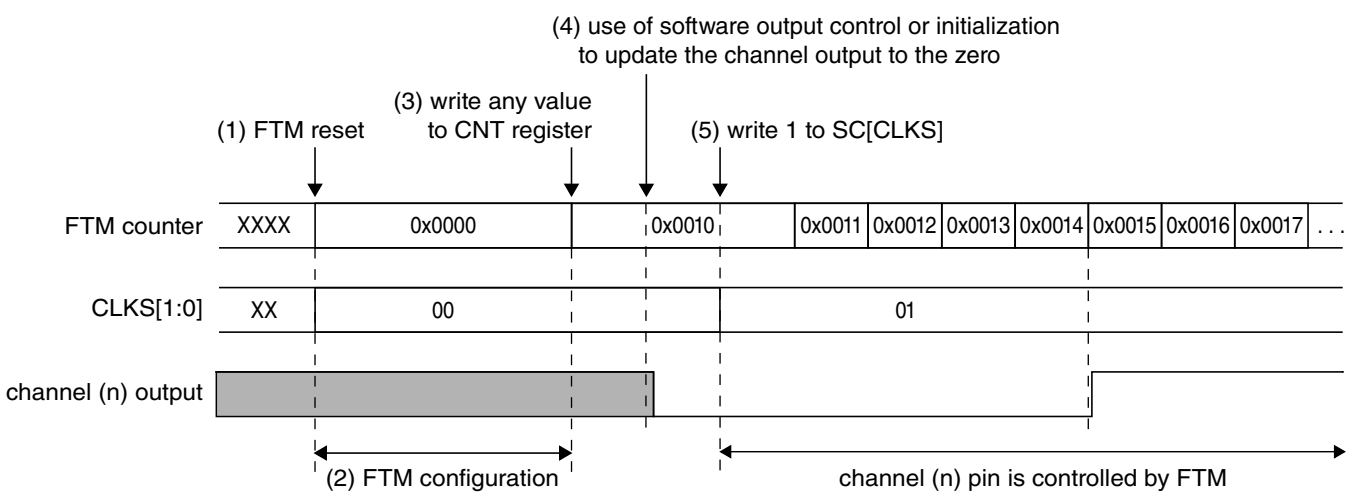
The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are only controlled by FTM when CLKS[1:0] bits are different from zero (See the table in the description of CnSC register).



NOTES:
– CNTIN = 0x0010
– Channel (n) is in low-true combine mode with CNTIN < C(n)V < C(n+1)V < MOD
– C(n)V = 0x0015

Figure 39-99. FTM behavior after reset when the channel (n) is in Combine mode

The following figure shows an example when the channel (n) is in Output Compare mode and the channel (n) output is toggled when there is a match. In the Output Compare mode, the channel output is not updated to its initial value when there is a write to CNT register (item 3). In this case, use the software output control (Software output control) or the initialization (Initialization) to update the channel output to the selected value (item 4).



NOTES:
– CNTIN = 0x0010
– Channel (n) is in output compare and the channel (n) output is toggled when there is a match
– C(n)V = 0x0014

Figure 39-100. FTM behavior after reset when the channel (n) is in Output Compare mode

39.7 FTM Interrupts

39.7.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

39.7.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

39.7.3 Fault Interrupt

The fault interrupt is generated when (FAULTIE = 1) and (FAULTF = 1).

39.8 Initialization Procedure

The following initialization procedure is recommended to configure the FlexTimer operation. This procedure can also be used to do a new configuration of the FlexTimer operation.

- Define the POL bits.
- Mask the channels outputs using SYNCHOM = 0. Two clocks after the write to OUTMASK, the channels output are in the safe value.
- (Re)Configuration FTM counter and channels to generation of periodic signals - Disable the clock. If the selected mode is Quadrature Decoder, then disable this mode. Examples of the (re)configuration:
  - Write to MOD.
  - Write to CNTIN.
  - Select OC, EPWM, CPWM, Combine, Complement modes for all channels that will be used
  - Select the high-true and low-true channels modes.
  - Write to CnV for all channels that will be used .
  - (Re)Configure deadtime and fault control.
  - Do not use the SWOC without SW synchronization (see item 6).
  - Do not use the Inverting without SW synchronization (see item 6).
  - Do not use the Initialization.
  - Do not change the polarity control.
  - Do not configure the HW synchronization

- Write any value to CNT. The FTM Counter is reset and the channels output are updated according to new configuration.
- Enable the clock. Write to CLKS[1:0] bits a value different from zero. If in the Quadrature Decoder mode, enable this mode.
- Configure the SW synchronization for SWOC (if it is necessary), Inverting (if it is necessary) and Output Mask (always)
  - Select synchronization for Output Mask Write to SYNC (SWSYNC = 0, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)
  - Write to SYNCONF.
    - HW Synchronization can not be enabled (HWSOC = 0, HWINVC = 0, HWOM = 0, HWWRBUF = 0, HWRSTCNT = 0, HWTRIGMODE = 0).
    - SW Synchronization for SWOC (if it is necessary): SWSOC = [0/1] and SWOC = [0/1].
    - SW Synchronization for Inverting (if it is necessary): SWINVC = [0/1] and INVC = [0/1].
    - SW Synchronization for SWOM (always): SWOM = 1. No enable the SW Synchronization for write buffers (because the writes to registers with write buffer are done using CLKS[1:0] = 2'b00): SWWRBUF = 0 and CNTINC = 0 .
    - SW Synchronization for counter reset (always): SWRSTCNT = 1.
    - Enhanced synchronization (always): SYNCMODE = 1
  - If the SWOC is used (SWSOC = 1 and SWOC = 1), then write to SWOCTRL register.
  - If the Inverting is used (SWINVC = 1 and INVC = 1), then write to INVCTRL register.
  - Write to OUTMASK to enable the masked channels.
- Generate the Software Trigger Write to SYNC (SWSYNC = 1, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)

Chapter 40
Periodic Interrupt Timer (PIT)

40.1 Chip-specific PIT information

40.1.1 PIT/DMA Periodic Trigger Assignments

The PIT generates periodic trigger events to the DMA Mux as shown in the table below.

Table 40-1. PIT channel assignments for periodic DMA triggering

DMA Channel Number	PIT Channel
DMA Channel 0	PIT Channel 0
DMA Channel 1	PIT Channel 1
DMA Channel 2	PIT Channel 2
DMA Channel 3	PIT Channel 3

40.1.2 PIT Trigger Output Assignments

The PIT timer channels are used to trigger other peripheral events such as ADC acquisitions or DMA transfers. On this device the following assignments have been made for each of the PIT channels.

PIT channel 0 output - XBARA_IN42, XBARB_IN24

PIT channel 1 output - XBARA_IN43 , XBARB_IN25

PIT channel 2 output - XBARA_IN50

PIT channel 3 output - XBARA_IN51

Routing the PIT channel outputs via XBARA/B provide options to periodically trigger events on other peripherals such as ADC, CMPs, Timers.

40.2 Introduction

The PIT module is an array of timers that can be used to raise interrupts and trigger DMA channels.

40.2.1 Block diagram

The following figure shows the block diagram of the PIT module.

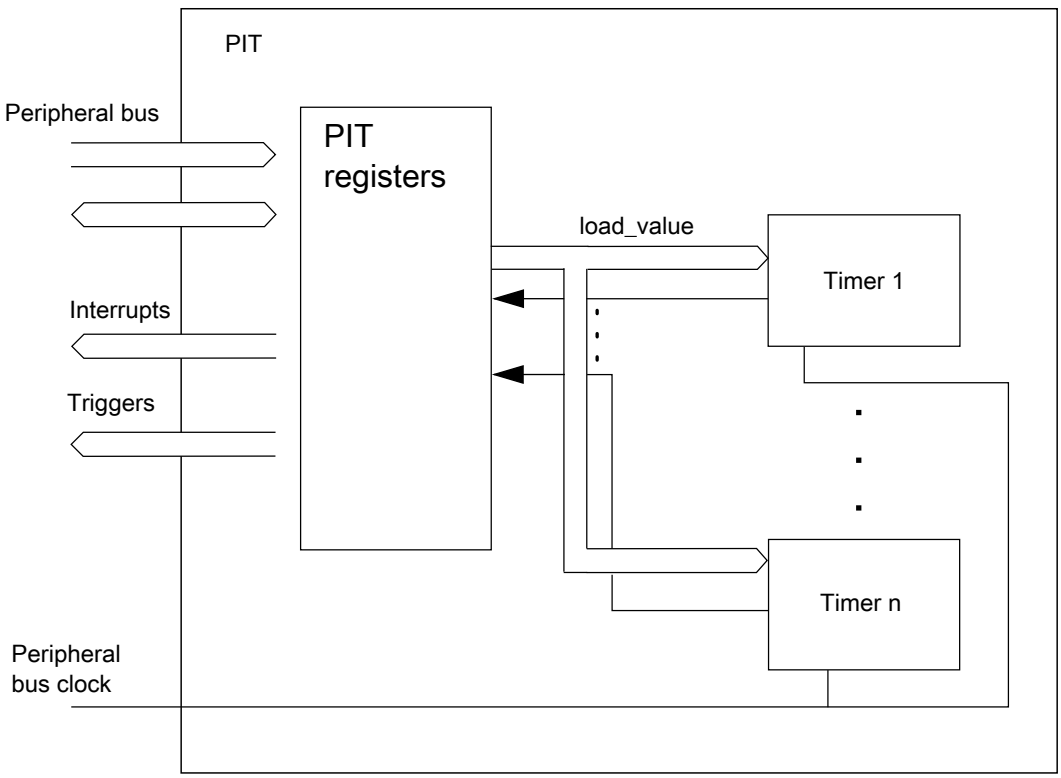


Figure 40-1. Block diagram of the PIT

NOTE

See the chip-specific PIT information for the number of PIT channels used in this MCU.

40.2.2 Features

The main features of this block are:

- Ability of timers to generate DMA trigger pulses

- Ability of timers to generate interrupts
- Maskable interrupts
- Independent timeout periods for each timer

40.3 Signal description

The PIT module has no external pins.

40.4 Memory map/register description

This section provides a detailed description of all registers accessible in the PIT module.

- See the chip-specific PIT information for the number of PIT channels used in this MCU.

NOTE

Write request to a register must have the byte enable asserted for the bytes user wants to update.

PIT memory map

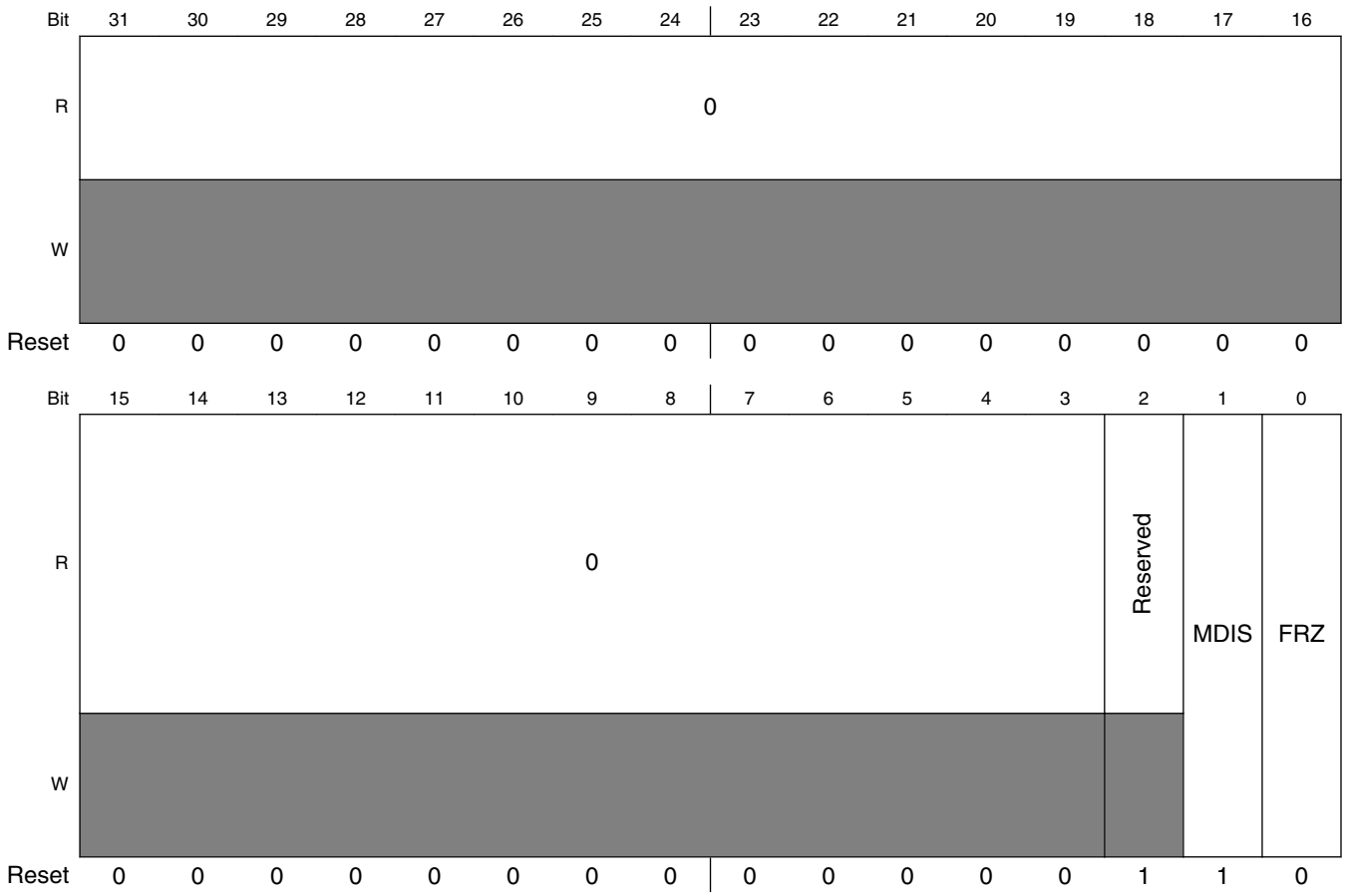
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_7000	PIT Module Control Register (PIT_MCR)	32	R/W	0000_0006h	40.4.1/1026
4003_7100	Timer Load Value Register (PIT_LDVAL0)	32	R/W	0000_0000h	40.4.2/1027
4003_7104	Current Timer Value Register (PIT_CVAL0)	32	R	0000_0000h	40.4.3/1027
4003_7108	Timer Control Register (PIT_TCTRL0)	32	R/W	0000_0000h	40.4.4/1028
4003_710C	Timer Flag Register (PIT_TFLG0)	32	R/W	0000_0000h	40.4.5/1029
4003_7110	Timer Load Value Register (PIT_LDVAL1)	32	R/W	0000_0000h	40.4.2/1027
4003_7114	Current Timer Value Register (PIT_CVAL1)	32	R	0000_0000h	40.4.3/1027
4003_7118	Timer Control Register (PIT_TCTRL1)	32	R/W	0000_0000h	40.4.4/1028
4003_711C	Timer Flag Register (PIT_TFLG1)	32	R/W	0000_0000h	40.4.5/1029
4003_7120	Timer Load Value Register (PIT_LDVAL2)	32	R/W	0000_0000h	40.4.2/1027
4003_7124	Current Timer Value Register (PIT_CVAL2)	32	R	0000_0000h	40.4.3/1027
4003_7128	Timer Control Register (PIT_TCTRL2)	32	R/W	0000_0000h	40.4.4/1028
4003_712C	Timer Flag Register (PIT_TFLG2)	32	R/W	0000_0000h	40.4.5/1029
4003_7130	Timer Load Value Register (PIT_LDVAL3)	32	R/W	0000_0000h	40.4.2/1027
4003_7134	Current Timer Value Register (PIT_CVAL3)	32	R	0000_0000h	40.4.3/1027
4003_7138	Timer Control Register (PIT_TCTRL3)	32	R/W	0000_0000h	40.4.4/1028
4003_713C	Timer Flag Register (PIT_TFLG3)	32	R/W	0000_0000h	40.4.5/1029

40.4.1 PIT Module Control Register (PIT_MCR)

This register enables or disables the PIT timer clocks and controls the timers when the PIT enters the Debug mode.

Access: User read/write

Address: 4003_7000h base + 0h offset = 4003_7000h



PIT_MCR field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved.
1 MDIS	Module Disable - (PIT section) Disables the standard timers. This field must be enabled before any other setup is done. 0 Clock for standard PIT timers is enabled. 1 Clock for standard PIT timers is disabled.

Table continues on the next page...

PIT_MCR field descriptions (continued)

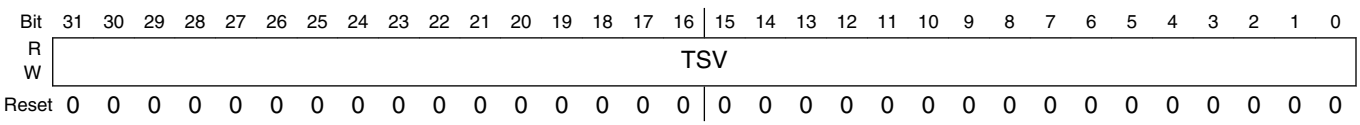
Field	Description
0 FRZ	Freeze Allows the timers to be stopped when the device enters the Debug mode. 0 Timers continue to run in Debug mode. 1 Timers are stopped in Debug mode.

40.4.2 Timer Load Value Register (PIT_LDVAL*n*)

These registers select the timeout period for the timer interrupts.

Access: User read/write

Address: 4003_7000h base + 100h offset + (16d × i), where i=0d to 3d



PIT_LDVAL*n* field descriptions

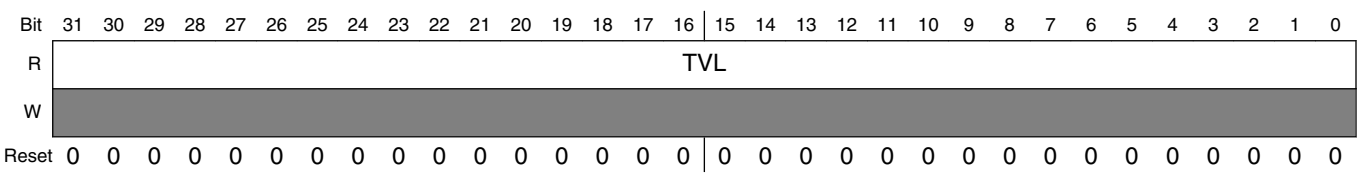
Field	Description
TSV	Timer Start Value Sets the timer start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer; instead the value will be loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.

40.4.3 Current Timer Value Register (PIT_CVAL*n*)

These registers indicate the current timer position.

Access: User read only

Address: 4003_7000h base + 104h offset + (16d × i), where i=0d to 3d



PIT_CVAL*n* field descriptions

Field	Description
TVL	Current Timer Value

PIT_CVAL*n* field descriptions (continued)

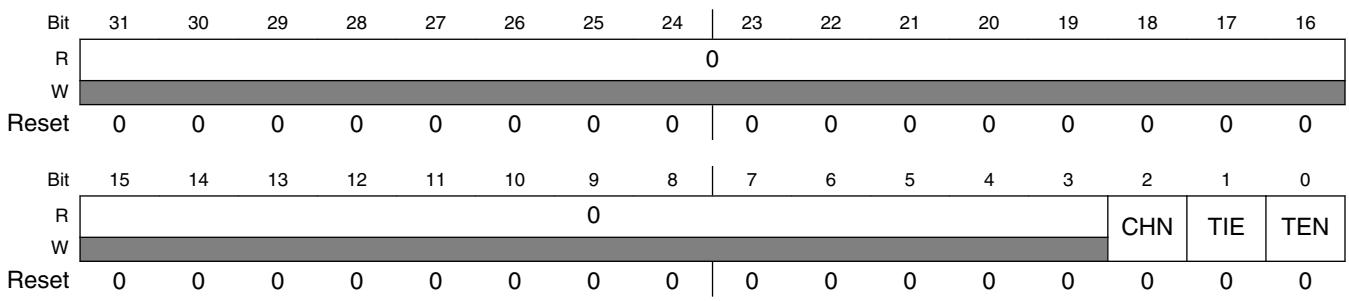
Field	Description
	Represents the current timer value, if the timer is enabled. **NOTE:** • If the timer is disabled, do not use this field as its value is unreliable. • The timer uses a downcounter. The timer values are frozen in Debug mode if MCR[FRZ] is set.

40.4.4 Timer Control Register (PIT_TCTRL*n*)

These registers contain the control bits for each timer.

Access: User read/write

Address: 4003_7000h base + 108h offset + (16d × i), where i=0d to 3d



PIT_TCTRL*n* field descriptions

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CHN	Chain Mode When activated, Timer n-1 needs to expire before timer n can decrement by 1. Timer 0 cannot be chained. 0 Timer is not chained. 1 Timer is chained to previous timer. For example, for Channel 2, if this field is set, Timer 2 is chained to Timer 1.
1 TIE	Timer Interrupt Enable When an interrupt is pending, or, TFLGn[TIF] is set, enabling the interrupt will immediately cause an interrupt event. To avoid this, the associated TFLGn[TIF] must be cleared first. 0 Interrupt requests from Timer n are disabled. 1 Interrupt will be requested whenever TIF is set.
0 TEN	Timer Enable Enables or disables the timer. 0 Timer n is disabled. 1 Timer n is enabled.

40.4.5 Timer Flag Register (PIT_TFLG*n*)

These registers hold the PIT interrupt flags.

Access: User read/write

Address: 4003_7000h base + 10Ch offset + (16d × i), where i=0d to 3d

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TIF
W																w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PIT_TFLG*n* field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 TIF	Timer Interrupt Flag Sets to 1 at the end of the timer period. Writing 1 to this flag clears it. Writing 0 has no effect. If enabled, or, when TCTRLn[TIE] = 1, TIF causes an interrupt request. 0 Timeout has not yet occurred. 1 Timeout has occurred.

40.5 Functional description

This section provides the functional description of the module.

40.5.1 General operation

This section gives detailed information on the internal operation of the module. Each timer can be used to generate trigger pulses and interrupts. Each interrupt is available on a separate interrupt line.

40.5.1.1 Timers

The timers generate triggers at periodic intervals, when enabled. The timers load the start values as specified in their LDVAL registers, count down to 0 and then load the respective start value again. Each time a timer reaches 0, it will generate a trigger pulse and set the interrupt flag.

All interrupts can be enabled or masked by setting TCTRLn[TIE]. A new interrupt can be generated only after the previous one is cleared.

If desired, the current counter value of the timer can be read via the CVAL registers.

The counter period can be restarted, by first disabling, and then enabling the timer with TCTRLn[TEN]. See the following figure.

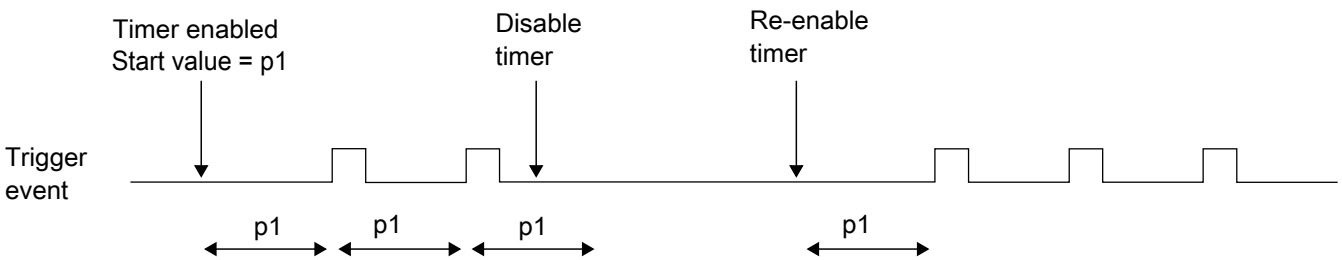


Figure 40-2. Stopping and starting a timer

The counter period of a running timer can be modified, by first disabling the timer, setting a new load value, and then enabling the timer again. See the following figure.

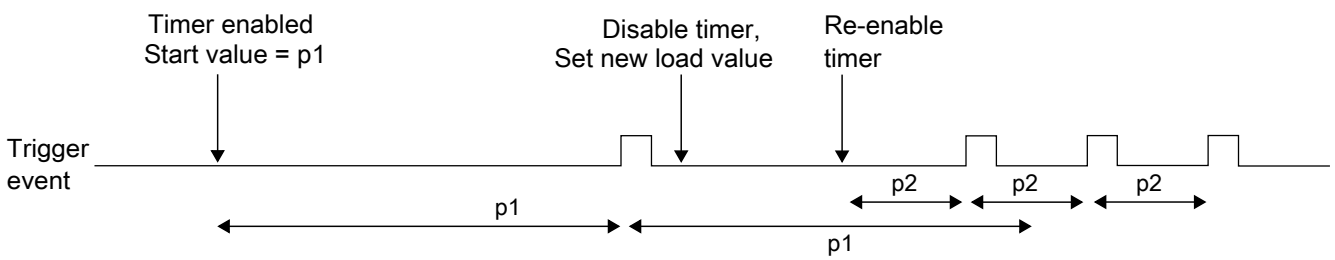


Figure 40-3. Modifying running timer period

It is also possible to change the counter period without restarting the timer by writing LDVAL with the new load value. This value will then be loaded after the next trigger event. See the following figure.

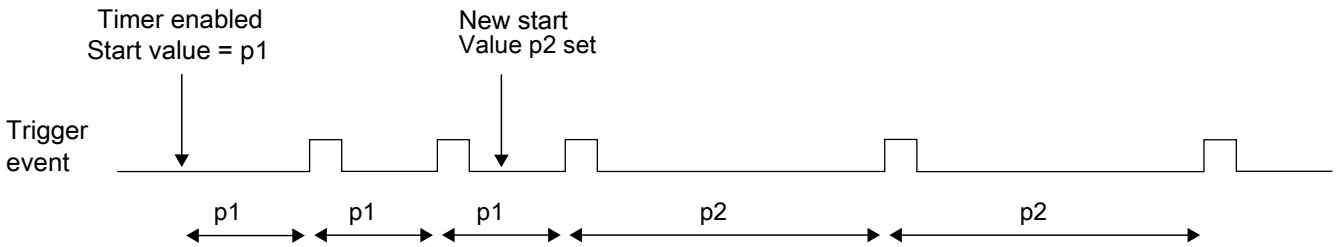


Figure 40-4. Dynamically setting a new load value

NOTE

- If the pause is initiated, when the timer is nearing 0 (CVALn =0x0), the pause command may not make it to the IP before the timer expires and generates a trigger.
- Pause will be ignored if (CVALn =0x0), but the trigger will remain asserted until the pause is removed. The user is recommended to remove Pause and then clear the interrupt TFLGn[TIF].
- If the timers are to be paused, sufficient time must be ensured for the IP to react.

40.5.1.2 Debug mode

In Debug mode, the timers will be frozen based on MCR[FRZ]. This is intended to aid software development, allowing the developer to halt the processor, investigate the current state of the system, for example, the timer values, and then continue the operation.

40.5.2 Interrupts

All the timers support interrupt generation. See the MCU specification for related vector addresses and priorities.

Timer interrupts can be enabled by setting TCTRLn[TIE]. TFLGn[TIF] are set to 1 when a timeout occurs on the associated timer, and are cleared to 0 by writing a 1 to the corresponding TFLGn[TIF].

40.5.3 Chained timers

When a timer has chain mode enabled, it will only count after the previous timer has expired. So if timer n-1 has counted down to 0, counter n will decrement the value by one. This allows to chain some of the timers together to form a longer timer. The first timer (timer 0) cannot be chained to any other timer.

40.6 Initialization and application information

In the example configuration:

- The PIT clock has a frequency of 50 MHz.
- Timer 1 creates an interrupt every 5.12 ms.
- Timer 3 creates a trigger event every 30 ms.

The PIT module must be activated by writing a 0 to MCR[MDIS].

The 50 MHz clock frequency equates to a clock period of 20 ns. Timer 1 needs to trigger every 5.12 ms/20 ns = 256,000 cycles and Timer 3 every 30 ms/20 ns = 1,500,000 cycles. The value for the LDVAL register trigger is calculated as:

LDVAL trigger = (period / clock period) -1

This means LDVAL1 and LDVAL3 must be written with 0x0003E7FF and 0x0016E35F respectively.

The interrupt for Timer 1 is enabled by setting TCTRL1[TIE]. The timer is started by writing 1 to TCTRL1[TEN].

Timer 3 shall be used only for triggering. Therefore, Timer 3 is started by writing a 1 to TCTRL3[TEN]. TCTRL3[TIE] stays at 0.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;


// Timer 1
PIT_LDVAL1 = 0x0003E7FF; // setup timer 1 for 256000 cycles
PIT_TCTRL1 = TIE; // enable Timer 1 interrupts
PIT_TCTRL1 |= TEN; // start Timer 1


// Timer 3
PIT_LDVAL3 = 0x0016E35F; // setup timer 3 for 1500000 cycles
PIT_TCTRL3 |= TEN; // start Timer 3
```

40.7 Example configuration for chained timers

In the example configuration:

- The PIT clock has a frequency of 100 MHz.
- Timers 1 and 2 are available.
- An interrupt shall be raised every 1 minute.

The PIT module needs to be activated by writing a 0 to MCR[MDIS].

The 100 MHz clock frequency equates to a clock period of 10 ns, so the PIT needs to count for 6000 million cycles, which is more than a single timer can do. So, Timer 1 is set up to trigger every 6 s (600 million cycles). Timer 2 is chained to Timer 1 and programmed to trigger 10 times.

The value for the LDVAL register trigger is calculated as number of cycles-1, so LDVAL1 receives the value 0x23C345FF and LDVAL2 receives the value 0x00000009.

The interrupt for Timer 2 is enabled by setting TCTRL2[TIE], the Chain mode is activated by setting TCTRL2[CHN], and the timer is started by writing a 1 to TCTRL2[TEN]. TCTRL1[TEN] needs to be set, and TCTRL1[CHN] and TCTRL1[TIE] are cleared.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;


// Timer 2
PIT_LDVAL2 = 0x00000009; // setup Timer 2 for 10 counts
PIT_TCTRL2 = TIE; // enable Timer 2 interrupt
PIT_TCTRL2 |= CHN; // chain Timer 2 to Timer 1
PIT_TCTRL2 |= TEN; // start Timer 2


// Timer 1
PIT_LDVAL1 = 0x23C345FF; // setup Timer 1 for 600 000 000 cycles
PIT_TCTRL1 = TEN; // start Timer 1
```

Chapter 41
Quadrature Encoder/Decoder (ENC)

41.1 Chip-specific ENC information

41.1.1 ENC Instantiation Information and signal assignment

This device contains one ENC module. The ENC module input signals and output signal are connected to the XBARA module which provides the capability to drive the ENC module from external pins or from a Timer. Using timer functions to drive the ENC inputs can provide a mechanism for customer;s to test the ENC module is functioning without connected in the intended "incremenatal position sensor" for motor control.

The following table shows how these modules are configured. This adheres to Kinetis K series FTM instantiations.

Table 41-1. ENC Signal Assignment

ENC Signal	Assignment	Features/usage
ENC_PHA	XBARA_OUT44	quadrature waveform input
ENC_PHB	XBARA_OUT45	quadrature waveform input
ENC_INDEX	XBARA_OUT46	refresh/reload input
ENC_HOME	XBARA_OUT47	home position input
ENC_CAP/Trigger	XBARA_OUT48	clear/snapshot input
ENC_POSMATCH	XBARA_IN45	compare trigger output

41.1.2 Clocks

The IPBus clock is the only clock required by this module in normal operation.

41.2 Introduction

The enhanced quadrature encoder/decoder module interfaces to position/speed sensors that are used in industrial motor control applications. Using 5 input signals (PHASEA, PHASEB, INDEX, TRIGGER, and HOME) from those position/speed sensors, the quadrature decoder module decodes shaft position, revolution count and speed.

41.2.1 Features

- Includes logic to decode quadrature signals
- Inputs can be connected to a general purpose timer to make low speed velocity measurements
- Configurable digital filter for inputs
- Quadrature decoder filter can be bypassed
- 32-bit position counter capable of modulo counting
- Position counter can be initialized by software or external events
- 16-bit position difference register
- Compare function can indicate when shaft has reached a defined position
- A watchdog timer can detect a non-rotating shaft condition
- Preloadable 16-bit revolution counter
- Maximum count frequency equals the peripheral clock rate
- Optional interrupt when both PHASEA and PHASEB inputs change in the same cycle

41.2.2 System Block Diagram

The next figure shows the block diagram of the quadrature decoder module integrated into an SoC.

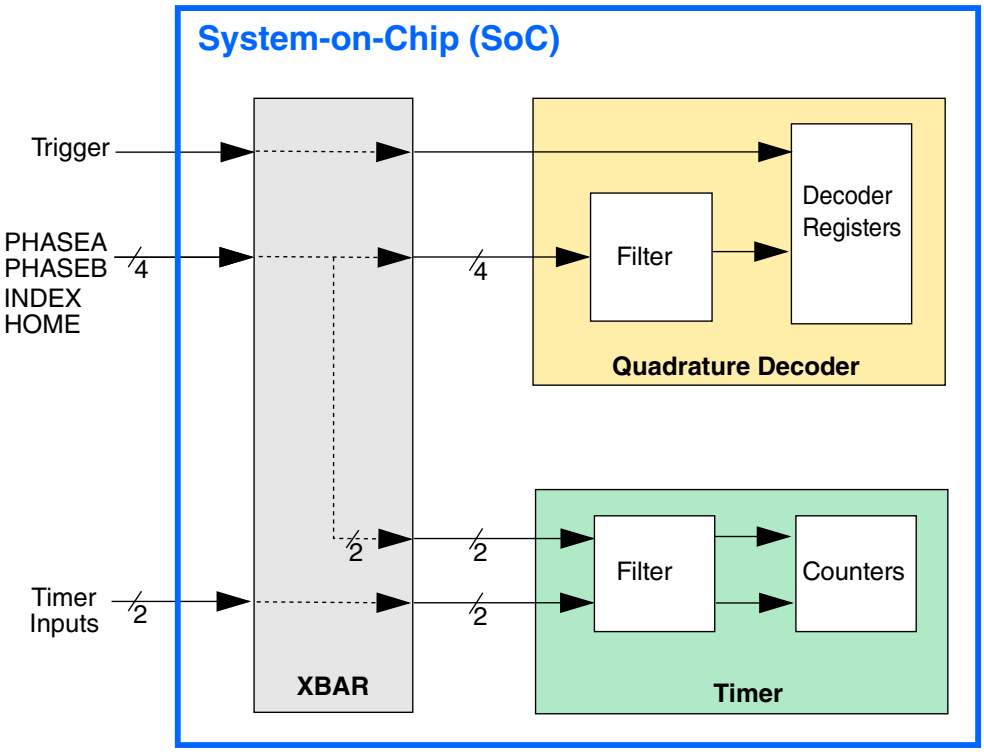


Figure 41-1. System Block Diagram

41.2.3 Decoder Block Diagram

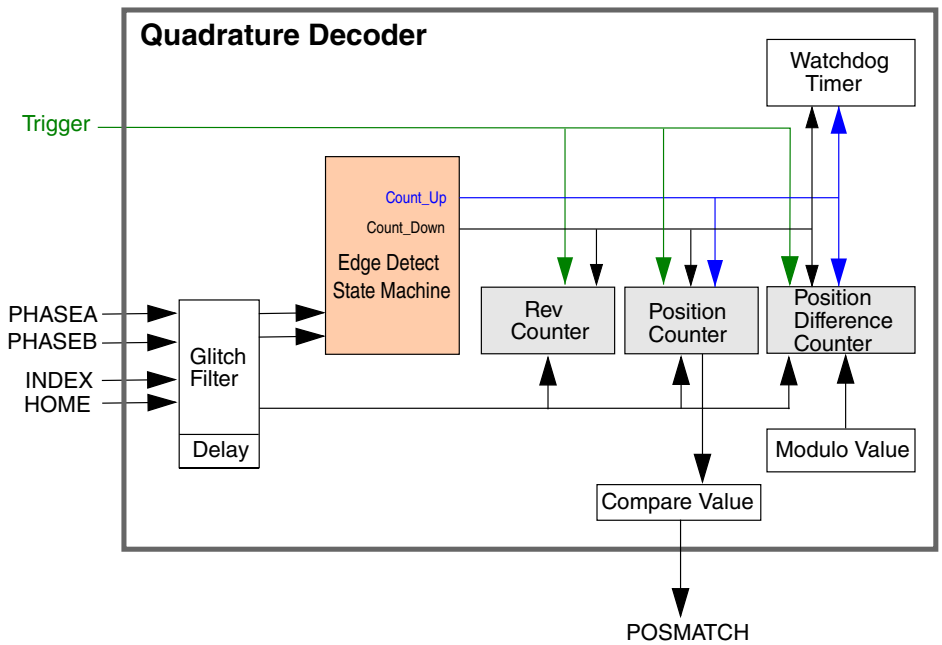


Figure 41-2. Quadrature Decoder Block Diagram

41.2.3.1 Glitch Filter

Because the quadrature decoder logic must sense signal transitions, the signal inputs are first run through a glitch filter. This glitch filter has a digital delay line, which samples multiple time points on the signal and verifies a stable new signal state, before outputting this new signal state to the internal quadrature decoder logic. To adapt to a variety of signal bandwidths, the sample rate of this delay line is programmable.

41.2.3.2 Edge Detect State Machine

The edge detect state machine looks for changes in the 4 possible states of the filtered PHASEA and PHASEB inputs, which are used to calculate the direction of motion. This information is formatted as Count_Up and Count_Down signals. These signals are routed into up to 3 up/down counters:

- Position counter
- Revolution counter
- Position difference counter

41.2.3.3 Position Counter

The 32-bit position counter calculates up or down on every count pulse, generated by the difference of PHASEA and PHASEB. The position counter acts as an integrator, whose count value is proportional to position. The direction of the count is determined by the Count_Up and Count_Down signals. Position counters may be initialized to a predetermined value by one of 3 different methods:

- Software-triggered event
- INDEX signal transition
- HOME signal transition

The INDEX and HOME signals can be programmed to interrupt the processor. Whenever the position counters (UPOS or LPOS) are read, a snapshot of the

- position counter
- position difference counter
- revolution counter

are each placed into their respective hold registers.

41.2.3.4 Position Difference Counter

The 16-bit position difference counter contains the position difference value occurring between each read of the position register. The position register counts up or down on every count pulse. The position difference counter acts as a differentiator, whose count value is proportional to the change in position since the last time the position counter was read.

When the position register, the position difference counter, or the revolution counter is read, the position difference of the counter's contents is copied into the position difference hold register (POSDH), and the position difference counter is cleared.

41.2.3.5 Position Difference Counter Hold

The Position Difference Counter Hold register stores a copy of the position difference counter at the time that the position register was read. When the position register is read, the position difference of the counter's contents is copied into the position difference hold register (POSDH), and the position difference counter is cleared.

41.2.3.6 Revolution Counter

The 16-bit up/down revolution counter is intended to count or integrate revolutions by counting index pulses. The direction of the count is determined by the Count_Up and Count_Down signals, determined by the Phase A and B inputs. A different count direction on the rising and falling edges of the index pulse indicates that the quadrature encoder changed direction on the index pulse.

41.2.3.7 Watchdog Timer

The watchdog timer ensures that the algorithm is indicating motion in the shaft; 2 successive counts indicate proper operation and will reset the timer.

- The timeout value is programmable.
- When a timeout occurs, an interrupt to the processor can be generated.

41.2.3.8 Pulse Accumulator Functionality

The logic can be programmed to integrate only selected transitions of the PHASEA signal. In this way, the position counter can be used as a pulse accumulator.

- The count direction is up.
- The pulse accumulator can also optionally be initialized by the INDEX input.

41.3 Signal Descriptions

Four external signals are multiplexed to the four pins (PHASEA, PHASEB, INDEX, HOME) of a quad timer module. Two internal signals (TRIGGER, POSMATCH) can be connected to other SoC resources.

Table 41-2. Signals

Signal		Description
PHASEA	Phase A Input	The PHASEA input can be connected to one of the phases from a two-phase shaft quadrature encoder output. PHASEA and PHASEB are used by the quadrature decoder module to indicate a decoder increment has passed, and to calculate its direction. • When the quadrature decoder module is used as a single phase pulse accumulator, PHASEA can also be used as the single input. • The PHASEA input can be an input capture channel for one of the timer modules (in the SoC), which can be connected to using a crossbar switch. Direction for two-phase shaft quadrature encoder output: • PHASEA is the leading phase for a shaft rotating in the positive direction. • PHASEA is the trailing phase for a shaft rotating in the negative direction.

Table continues on the next page...

Table 41-2. Signals (continued)

Signal		Description
PHASEB	Phase B Input	The PHASEB input can be connected to the other phase from a two-phase shaft quadrature encoder output. • The PHASEB input can be an input capture channel for one of the timer modules (in the SoC), which can be connected to using a crossbar switch. Direction for two-phase shaft quadrature encoder output: • PHASEB is the trailing phase for a shaft rotating in the positive direction. • PHASEB is the leading phase for a shaft rotating in the negative direction.
INDEX	Index Input	• Normally connected to the index pulse output of a quadrature encoder, the INDEX pulse can optionally reset the position counter and the pulse accumulator of the quadrature decoder module. The INDEX pulse also causes a change of state on the revolution counter. The direction of this state change (increment or decrement) is calculated from the PHASEA and PHASEB inputs. • The INDEX input can be an input capture channel for one of the timer modules (in the SoC), which can be connected to using a crossbar switch.
HOME	Home Switch Input	The HOME input can be used by the quadrature decoder and the timer module. • The HOME input can be used to trigger the initialization of the position counters (UPOS and LPOS). Often the HOME signal is connected to a sensor on the motor or machine, sending notification that it has reached a defined home position. • The HOME signal can be connected to the timer module (in the SoC) using a crossbar switch.
TRIGGER	Trigger Input	The TRIGGER input can be used • to clear the position counters (UPOS and LPOS) • to take a snapshot of the POS, REV, and POSD registers • to indicate an elapsed time period, by connecting the TRIGGER input signal to a periodic pulse generator or timer
POSMATCH	Position Match Output	• To record the time at which the position of the shaft matches a user-defined compare value (COMP), the POSMATCH output can be used to trigger a timer channel. • Alternatively, to record the time at which the position information was read, (when the position counters (UPOS and LPOS), revolution counter (REV), or position difference counter (POSD) registers are read) the POSMATCH output can be used to trigger a timer channel.

41.4 Memory Map and Registers

The address of a register is the sum of a base address and an address offset. The base address is defined at the MCU level, while the address offset is defined at the module level. Refer to the specific chip documentation for the definition of the base address. All memory locations base and offsets are given in hex.

ENC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_5000	Control Register (ENC0_CTRL)	16	R/W	0000h	41.4.1/1043
4005_5002	Input Filter Register (ENC0_FILT)	16	R/W	0000h	41.4.2/1045
4005_5004	Watchdog Timeout Register (ENC0_WTR)	16	R/W	0000h	41.4.3/1046
4005_5006	Position Difference Counter Register (ENC0_POSD)	16	R/W	0000h	41.4.4/1047
4005_5008	Position Difference Hold Register (ENC0_POSDH)	16	R	0000h	41.4.5/1047
4005_500A	Revolution Counter Register (ENC0_REV)	16	R/W	0000h	41.4.6/1048
4005_500C	Revolution Hold Register (ENC0_REVH)	16	R	0000h	41.4.7/1048
4005_500E	Upper Position Counter Register (ENC0_UPOS)	16	R/W	0000h	41.4.8/1048
4005_5010	Lower Position Counter Register (ENC0_LPOS)	16	R/W	0000h	41.4.9/1049
4005_5012	Upper Position Hold Register (ENC0_UPOSH)	16	R	0000h	41.4.10/ 1049
4005_5014	Lower Position Hold Register (ENC0_LPOSH)	16	R	0000h	41.4.11/ 1049
4005_5016	Upper Initialization Register (ENC0_UINIT)	16	R/W	0000h	41.4.12/ 1050
4005_5018	Lower Initialization Register (ENC0_LINIT)	16	R/W	0000h	41.4.13/ 1050
4005_501A	Input Monitor Register (ENC0_IMR)	16	R	0000h	41.4.14/ 1051
4005_501C	Test Register (ENC0_TST)	16	R/W	0000h	41.4.15/ 1052
4005_501E	Control 2 Register (ENC0_CTRL2)	16	R/W	0000h	41.4.16/ 1053
4005_5020	Upper Modulus Register (ENC0_UMOD)	16	R/W	0000h	41.4.17/ 1055
4005_5022	Lower Modulus Register (ENC0_LMOD)	16	R/W	0000h	41.4.18/ 1055
4005_5024	Upper Position Compare Register (ENC0_UCOMP)	16	R/W	FFFFh	41.4.19/ 1056
4005_5026	Lower Position Compare Register (ENC0_LCOMP)	16	R/W	FFFFh	41.4.20/ 1056

41.4.1 Control Register (ENCx_CTRL)

Address: 4005_5000h base + 0h offset = 4005_5000h

Bit	15	14	13	12	11	10	9	8
Read	HIRQ	HIE	HIP	HNE	0	REV	PH1	XIRQ
Write	w1c				SWIP			w1c
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	XIE	XIP	XNE	DIRQ	DIE	WDE	CMPIRQ	CMPIE
Write				w1c			w1c	
Reset	0	0	0	0	0	0	0	0

ENCx_CTRL field descriptions

Field	Description
15 HIRQ	HOME Signal Transition Interrupt Request This bit is set when a transition on the HOME signal occurs according to the CTRL[HNE] bit. If this bit is set and CTRL[HIE] is set, a HOME interrupt occurs. This bit will remain set until it is cleared by software. Write a one to this bit to clear. 0 No interrupt 1 HOME signal transition interrupt request
14 HIE	HOME Interrupt Enable This read/write bit enables HOME signal interrupts. 0 Disable HOME interrupts 1 Enable HOME interrupts
13 HIP	Enable HOME to Initialize Position Counters UPOS and LPOS This read/write bit allows the position counter to be initialized by the HOME signal. 0 No action 1 HOME signal initializes the position counter
12 HNE	Use Negative Edge of HOME Input This read/write bit determines whether to use the positive or negative edge of the HOME input. 0 Use positive going edge-to-trigger initialization of position counters UPOS and LPOS 1 Use negative going edge-to-trigger initialization of position counters UPOS and LPOS
11 SWIP	Software Triggered Initialization of Position Counters UPOS and LPOS Writing a one to this bit will transfer the UINIT and LINIT contents to UPOS and LPOS. This bit is always read as a zero. 0 No action 1 Initialize position counter

Table continues on the next page...

ENCx_CTRL field descriptions (continued)

Field	Description
10 REV	Enable Reverse Direction Counting This read/write bit reverses the interpretation of the quadrature signal, changing the direction of count. 0 Count normally 1 Count in the reverse direction
9 PH1	Enable Signal Phase Count Mode This read/write bit bypasses the quadrature decoder logic. 0 Use standard quadrature decoder where PHASEA and PHASEB represent a two phase quadrature signal. 1 Bypass the quadrature decoder. A positive transition of the PHASEA input generates a count signal. The PHASEB input and the REV bit control the counter direction. • If CTRL[REV] = 0, PHASEB = 0, then count up • If CTRL[REV] = 0, PHASEB = 1, then count down • If CTRL[REV] = 1, PHASEB = 0, then count down • If CTRL[REV] = 1, PHASEB = 1, then count up
8 XIRQ	INDEX Pulse Interrupt Request This bit is set when an INDEX interrupt occurs. It will remain set until cleared by software. The clearing procedure is to write a one to this bit. 0 No interrupt has occurred 1 INDEX pulse interrupt has occurred
7 XIE	INDEX Pulse Interrupt Enable This read/write bit enables index interrupts. 0 INDEX pulse interrupt is disabled 1 INDEX pulse interrupt is enabled
6 XIP	INDEX Triggered Initialization of Position Counters UPOS and LPOS This read/write bit enables the position counter to be initialized by the INDEX pulse. 0 No action 1 INDEX pulse initializes the position counter
5 XNE	Use Negative Edge of INDEX Pulse This read/write bit determines the edge of the INDEX pulse used to initialize the position counter. 0 Use positive transition edge of INDEX pulse 1 Use negative transition edge of INDEX pulse
4 DIRQ	Watchdog Timeout Interrupt Request This bit is set when a watchdog timeout interrupt occurs. It will remain set until cleared by software. Write a one to this bit to clear. This bit is also cleared when CTRL[WDE] is 0. 0 No interrupt has occurred 1 Watchdog timeout interrupt has occurred
3 DIE	Watchdog Timeout Interrupt Enable This read/write bit enables watchdog timeout interrupts.

Table continues on the next page...

ENCx_CTRL field descriptions (continued)

Field	Description
	0 Watchdog timer interrupt is disabled 1 Watchdog timer interrupt is enabled
2 WDE	Watchdog Enable This bit allows operation of the watchdog timer monitoring the PHASEA and PHASEB inputs for motor movement. 0 Watchdog timer is disabled 1 Watchdog timer is enabled
1 CMPIRQ	Compare Interrupt Request This bit is set when a match occurs between the counter and the COMP value. It will remain set until cleared by software. Write a one to this bit to clear. 0 No match has occurred 1 COMP match has occurred
0 CMPIE	Compare Interrupt Enable This read/write bit enables compare interrupts. 0 Compare interrupt is disabled 1 Compare interrupt is enabled

41.4.2 Input Filter Register (ENCx_FILT)

This register sets the values of the input filter sample period (FILT_PER) and the input filter sample count (FILT_CNT).

The FILT_PER value should be set such that the sampling period is larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The FILT_CNT value should be chosen to reduce the probablility of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of FILT_CNT+3.

The values of FILT_PER and FILT_CNT must also be traded off against the desire for minimal latency in recognizing valid input transitions. Turning on the input filter (setting FILT_PER to a non-zero value) introduces a latency of ((FILT_CNT+3)*FILT_PER+2) IPBus clock periods.

The filter latency can be measured as follows: drive the quadrature decoder inputs, PHASEA, PHASEB, INDEX, and HOME monitoring the filtered output in the input monitor register (IMR). Determine how many IPBus clock cycles it takes before the output shows up, by using the following equations, where f is FILT_PER and s is FILT_CNT.

1. DELAY (IPBus clock cycles) = f * (s+3) +1 (to read the filtered output)

2. DELAY (IPBus clock cycles) = f * (s+3) +2 (to monitor the output in the IMR)

One more additional IPBus clock cycle is needed to read the filtered output, and two more IPBus clock cycles are needed to monitor the filtered output in the IMR. The sample rate is set when it reaches the number f. The following examples employ the preceding equations:

- Example: when f = 0, the filter is bypassed. Therefore, DELAY = 1 or 2 clock cycles.
- Example: when f = 5 and s = 2, DELAY = 5 * (2+3) + (1 or 2) = 26 or 27 clock cycles.

Address: 4005_5000h base + 2h offset = 4005_5002h

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

ENCx_FILT field descriptions

Field	Description
15–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10–8 FILT_CNT	Input Filter Sample Count These bits represent the number of consecutive samples that must agree prior to the input filter accepting an input transition. A value of 0x0 represents 3 samples. A value of 0x7 represents 10 samples. The value of FILT_CNT affects the input latency.
FILT_PER	Input Filter Sample Period These bits represent the sampling period (in IPBus clock cycles) of the decoder input signals. Each input is sampled multiple times at the rate specified by this field. If FILT_PER is 0x00 (default), then the input filter is bypassed. Bypassing the digital filter enables the position/position difference counters to operate with count rates up to the IPBus frequency. The value of FILT_PER affects the input latency. When changing FILT_PER from one non-zero value to another non-zero value, write a value of 0 first in order to clear the filter.

41.4.3 Watchdog Timeout Register (ENCx_WTR)

This read/write register stores the timeout count for the quadrature decoder module watchdog timer. This timer is separate from the watchdog timer in the COP module.

Address: 4005_5000h base + 4h offset = 4005_5004h

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

ENCx_WTR field descriptions

Field	Description
WDOG	WDOG[15:0] is a binary representation of the number of clock cycles plus one that the watchdog timer counts before timing out and optionally generating an interrupt.

41.4.4 Position Difference Counter Register (ENCx_POSD)

Address: 4005_5000h base + 6h offset = 4005_5006h

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

ENCx_POSD field descriptions

Field	Description
POSD	This read/write register contains the position change in value occurring between each read of the position register. The value of the position difference counter register (POSD) can be used to calculate velocity. The 16-bit position difference counter computes up or down on every count pulse. This counter acts as a differentiator whose count value is proportional to the change in position since the last time the position counter was read. When the position register, the position difference counter, or the revolution counter is read, the position difference counter's contents are copied into the position difference hold register (POSDH) and the position difference counter is cleared.

41.4.5 Position Difference Hold Register (ENCx_POSDH)

Address: 4005_5000h base + 8h offset = 4005_5008h

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

ENCx_POSDH field descriptions

Field	Description
POSDH	This read-only register contains a snapshot of the value of the POSD register. The value of the position difference hold register (POSDH) can be used to calculate velocity.

41.4.6 Revolution Counter Register (ENCx_REV)

Address: 4005_5000h base + Ah offset = 4005_500Ah

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

ENCx_REV field descriptions

Field	Description
REV	This read/write register contains the current value of the revolution counter.

41.4.7 Revolution Hold Register (ENCx_REVH)

Address: 4005_5000h base + Ch offset = 4005_500Ch

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

ENCx_REVH field descriptions

Field	Description
REVH	This read-only register contains a snapshot of the value of the REV register.

41.4.8 Upper Position Counter Register (ENCx_UPOS)

Address: 4005_5000h base + Eh offset = 4005_500Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	POS[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENCx_UPOS field descriptions

Field	Description
POS[31:16]	This read/write register contains the upper (most significant) half of the position counter. This is the binary count from the position counter.

41.4.9 Lower Position Counter Register (ENCx_LPOS)

Address: 4005_5000h base + 10h offset = 4005_5010h

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

ENCx_LPOS field descriptions

Field	Description
POS[15:0]	This read/write register contains the lower (least significant) half of the position counter. This is the binary count from the position counter.

41.4.10 Upper Position Hold Register (ENCx_UPOSH)

Address: 4005_5000h base + 12h offset = 4005_5012h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	POSH[31:16]															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENCx_UPOSH field descriptions

Field	Description
POSH[31:16]	This read-only register contains a snapshot of the UPOS register.

41.4.11 Lower Position Hold Register (ENCx_LPOSH)

Address: 4005_5000h base + 14h offset = 4005_5014h

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

ENCx_LPOSH field descriptions

Field	Description
POSH[15:0]	This read-only register contains a snapshot of the LPOS register.

41.4.12 Upper Initialization Register (ENCx_UINIT)

Address: 4005_5000h base + 16h offset = 4005_5016h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	INIT[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENCx_UINIT field descriptions

Field	Description
INIT[31:16]	This read/write register contains the value to be used to initialize the upper half of the position counter (UPOS).

41.4.13 Lower Initialization Register (ENCx_LINIT)

Address: 4005_5000h base + 18h offset = 4005_5018h

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

ENCx_LINIT field descriptions

Field	Description
INIT[15:0]	This read/write register contains the value to be used to initialize the lower half of the position counter (LPOS).

41.4.14 Input Monitor Register (ENCx_IMR)

This read-only register contains the values of the raw and filtered PHASEA, PHASEB, INDEX, and HOME input signals. The reset value depends on the values of the raw and filtered values of PHASEA, PHASEB, INDEX, and HOME. If these input pins are connected to a pull-up, bits 0–7 of the IMR are all ones. If these input pins are connected to a pull-down device, bits 0–7 are all zeros.

Address: 4005_5000h base + 1Ah offset = 4005_501Ah

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	FPHA	FPHB	FIND	FHOM	PHA	PHB	INDEX	HOME
Write								
Reset	0	0	0	0	0	0	0	0

ENCx_IMR field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 FPHA	This is the filtered version of PHASEA input.
6 FPHB	This is the filtered version of PHASEB input.
5 FIND	This is the filtered version of INDEX input.
4 FHOM	This is the filtered version of HOME input.
3 PHA	This is the raw PHASEA input.
2 PHB	This is the raw PHASEB input.
1 INDEX	This is the raw INDEX input.
0 HOME	This is the raw HOME input.

41.4.15 Test Register (ENCx_TST)

This read/write register controls and sets the frequency of a quadrature signal generator. It provides a quadrature test signal to the inputs of the quadrature decoder module. The TEST_COUNT value is counted down to zero when the test module is enabled (TEN = 1) and the count is enabled (TCE = 1). Each count value of one represents a single quadrature cycle interpreted as a count of one by the position counter (UPOS and LPOS) if it is so enabled. Repeated writing of new values to TEST_COUNT can cause an extra phase transition and therefore an extra count by the position counter. The period field determines in IPBus clock cycles the length of each quadrature cycle phase. This register is a factory test feature; however, it may be useful to customers' software development and testing.

Address: 4005_5000h base + 1Ch offset = 4005_501Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	TEN	TCE	QDN	TEST_PERIOD					TEST_COUNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENCx_TST field descriptions

Field	Description
15 TEN	Test Mode Enable This bit connects the test module to inputs of the quadrature decoder module. 0 Test module is not enabled 1 Test module is enabled
14 TCE	Test Counter Enable This bit connects the test counter to inputs of the quadrature decoder module. 0 Test count is not enabled 1 Test count is enabled
13 QDN	Quadrature Decoder Negative Signal When set, this bit generates a negative quadrature signal. Otherwise the signal is in a positive direction. 0 Leaves quadrature decoder signal in a positive direction 1 Generates a negative quadrature decoder signal
12–8 TEST_PERIOD	These bits hold the period of quadrature phase in IPBus clock cycles.
TEST_COUNT	These bits hold the number of quadrature advances to generate.

41.4.16 Control 2 Register (ENCx_CTRL2)

Address: 4005_5000h base + 1Eh offset = 4005_501Eh

Bit	15	14	13	12	11	10	9	8
Read	0				SABIRQ	SABIE	OUTCTL	REVMOD
Write					w1c			
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	ROIRQ	ROIE	RUIRQ	RUIE	DIR	MOD	UPDPOS	UPDHLD
Write	w1c		w1c					
Reset	0	0	0	0	0	0	0	0

ENCx_CTRL2 field descriptions

Field	Description
15–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 SABIRQ	Simultaneous PHASEA and PHASEB Change Interrupt Request This bit indicates that the PHASEA and PHASEB inputs changed simultaneously (within a single clock period). This event typically indicates an error condition because quadrature coding requires only one of these inputs to change at a time. The bit remains set until it is cleared by software or a reset. Write 1 to this bit to clear it. 0 No simultaneous change of PHASEA and PHASEB has occurred. 1 A simultaneous change of PHASEA and PHASEB has occurred.
10 SABIE	Simultaneous PHASEA and PHASEB Change Interrupt Enable This read/write bit enables simultaneous PHASEA and PHASEB change interrupts based on SABIRQ being set. 0 Simultaneous PHASEA and PHASEB change interrupt disabled. 1 Simultaneous PHASEA and PHASEB change interrupt enabled.
9 OUTCTL	Output Control This bit is used to control the behavior of the POSMATCH output signal. This can control when a timer channel captures a time stamp. 0 POSMATCH pulses when a match occurs between the position counters (POS) and the compare value (COMP). 1 POSMATCH pulses when the UPOS, LPOS, REV, or POSD registers are read.
8 REVMOD	Revolution Counter Modulus Enable This bit is used to determine how the revolution counter (REV) is incremented/decremented. By default REV is controlled based on the count direction and the INDEX pulse. As an option, REV can be controlled using the roll-over/under detection during modulo counting. 0 Use INDEX pulse to increment/decrement revolution counter (REV). 1 Use modulus counting roll-over/under to increment/decrement revolution counter (REV).

Table continues on the next page...

ENCx_CTRL2 field descriptions (continued)

Field	Description
7 ROIRQ	Roll-over Interrupt Request This bit is set when the position counter (POS) rolls over from the MOD value to the INIT value or from 0xffffffff to 0x00000000. It will remain set until cleared by software. Write a one to this bit to clear. 0 No roll-over has occurred 1 Roll-over has occurred
6 ROIE	Roll-over Interrupt Enable This read/write bit enables roll-over interrupts based on CTRL2[ROIRQ] being set. This interrupt is combined with the index interrupt signal. 0 Roll-over interrupt is disabled 1 Roll-over interrupt is enabled
5 RUIRQ	Roll-under Interrupt Request This bit is set when the position counter (POS) rolls under from the INIT value to the MOD value or from 0x00000000 to 0xffffffff. It will remain set until cleared by software. Write a one to this bit to clear. 0 No roll-under has occurred 1 Roll-under has occurred
4 RUIE	Roll-under Interrupt Enable This read/write bit enables roll-under interrupts based on CTRL2[RUIRQ] being set. This interrupt is combined with the index interrupt signal. 0 Roll-under interrupt is disabled 1 Roll-under interrupt is enabled
3 DIR	Count Direction Flag This read-only flag is used to indicate the direction of the last count. 0 Last count was in the down direction 1 Last count was in the up direction
2 MOD	Enable Modulo Counting When set, this bit allows the position counters (UPOS and LPOS) to count in a modulo fashion using MOD and INIT as the upper and lower bounds of the counting range. During modulo counting when a count up is indicated and the position counter is equal to MOD, then the postion counter will be reloaded with the value of INIT. When a count down is indicated and the position counter is equal to INIT, then the position counter will be reloaded with the value of MOD. When clear, then the values of MOD and INIT are ignored and the position counter wraps to zero when counting up from 0xffffffff and wraps to 0xffffffff when counting down from 0. 0 Disable modulo counting 1 Enable modulo counting
1 UPDPOS	Update Position Registers When set, this bit allows the TRIGGER input to clear the POSD, REV, UPOS and LPOS registers. When clear, the POSD, REV, UPOS and LPOS registers ignore the TRIGGER input. 0 No action for POSD, REV, UPOS and LPOS on rising edge of TRIGGER 1 Clear POSD, REV, UPOS and LPOS on rising edge of TRIGGER

Table continues on the next page...

ENCx_CTRL2 field descriptions (continued)

Field	Description
0 UPDHLD	Update Hold Registers When set, this bit allows the TRIGGER input to cause an update of the POSDH, REVH, UPOSH, and LPOSH registers. When clear, the hold registers are not updated by the TRIGGER input. Updating the POSDH register will also cause the POSD register to be cleared. 0 Disable updates of hold registers on rising edge of TRIGGER 1 Enable updates of hold registers on rising edge of TRIGGER

41.4.17 Upper Modulus Register (ENCx_UMOD)

Address: 4005_5000h base + 20h offset = 4005_5020h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	MOD[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENCx_UMOD field descriptions

Field	Description
MOD[31:16]	This read/write register contains the upper (most significant) half of the modulus register. MOD acts as the upper bound during modulo counting and as the upper reload value when rolling over from the lower bound.

41.4.18 Lower Modulus Register (ENCx_LMOD)

Address: 4005_5000h base + 22h offset = 4005_5022h

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

ENCx_LMOD field descriptions

Field	Description
MOD[15:0]	This read/write register contains the lower (least significant) half of the modulus register. MOD acts as the upper bound during modulo counting and as the upper reload value when rolling over from the lower bound.

41.4.19 Upper Position Compare Register (ENCx_UCOMP)

Address: 4005_5000h base + 24h offset = 4005_5024h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	COMP[31:16]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

ENCx_UCOMP field descriptions

Field	Description
COMP[31:16]	This read/write register contains the upper (most significant) half of the position compare register. When the value of POS matches the value of COMP, the CTRL[CMPIRQ] flag is set and the POSMATCH output is asserted.

41.4.20 Lower Position Compare Register (ENCx_LCOMP)

Address: 4005_5000h base + 26h offset = 4005_5026h

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

ENCx_LCOMP field descriptions

Field	Description
COMP[15:0]	This read/write register contains the lower (least significant) half of the position compare register. When the value of POS matches the value of COMP, the CTRL[CMPIRQ] flag is set and the POSMATCH output is asserted.

41.5 Functional Description

The next timing diagram shows the basic operation of a quadrature incremental position quadrature decoder.

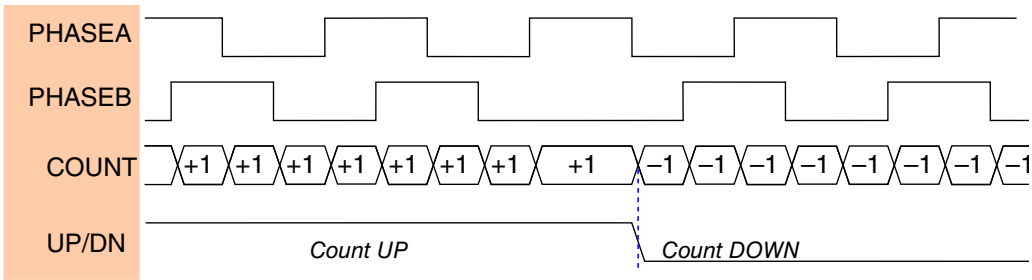


Figure 41-3. Quadrature Decoder Signals

41.5.1 Positive versus Negative Direction

A typical quadrature encoder has 3 outputs: PHASEA signal, PHASEB signal, INDEX pulse (not shown).

- If PHASEA leads PHASEB, then motion is in the positive direction.
- If PHASEA trails PHASEB, then motion is in the negative direction.

Transitions on these phases can be integrated to yield position or differentiated to yield velocity. The quadrature decoder is designed to perform these functions in hardware.

41.5.2 Prescaler for Slow or Fast Speed Measurement

- For applications with a fast moving shaft encoder, the speed can be computed by calculating the change in the position counter per unit time, or by reading the position difference counter register (POSD) and calculating speed.
- For applications with slow motor speeds and low line count quadrature encoders, the timer module enables high resolution velocity measurements by measuring the time period between quadrature phases.
  - The timer module uses a 16-bit free running counter operated from a prescaled version of the IPBus clock.
  - The prescaler divides the IPBus clock by values ranging from 1 to 128. A 60 MHz IPBus clock frequency would yield a resolution of from 17 ns to 2.1 µs and a maximum count period of from 1.09 ms to 140 ms. For example, with a 1000-tooth decoder, speeds could be calculated down to 0.11 rpm using a prescaler.

41.5.3 Holding Registers and Initializing Registers

Hold registers are associated with 3 types of counters:

- Position
- Position difference
- Revolution

When any of the counter registers is read, the contents of each counter register is written to the corresponding hold register. Taking a snapshot of the counters' values provides a consistent view of a system's position and the velocity to be attained. to capture a time stamp of when these registers are read, use the POSMATCH output with a timer channel.

The position counter is 32 bits wide. To ensure that the position counter can be reliably initialized with two 16-bit accesses, two registers (an upper and a lower initialization register) are provided. The upper initialization register (UINIT) and lower initialization register (LINIT) should be loaded with the desired value. Next, the position counter can be loaded by writing 1 to the Software-Triggered Initialization of Position Counters UPOS and LPOS bit (CTRL[SWIP]). Alternatively, either the INDEX-Triggered Initialization of Position Counters UPOS and LPOS bit (CTRL[XIP]) or the Enable HOME to Initialize Position Counters UPOS and LPOS bit (CTRL[HIP]) can enable the position counter to be initialized in response to a HOME or INDEX signal transition.

41.6 Resets

There are no special requirements. This module is reset by any system reset.

41.7 Clocks

The IPBus clock is the only clock required by this module in normal operation.

41.8 Interrupts

The following table lists the module interrupts.

Table 41-3. Interrupt Summary

Core Interrupt	Interrupt Flag	Interrupt Enable	Description
ipi_int_home	CTRL[HIRQ]	CTRL[HIE]	HOME signal transition interrupt
ipi_int_index	CTRL[XIRQ]	CTRL[XIE]	INDEX signal transition interrupt or roll-over/ under interrupt

Table continues on the next page...

Table 41-3. Interrupt Summary (continued)

Core Interrupt	Interrupt Flag	Interrupt Enable	Description
	CTRL2[ROIRQ]	CTRL2[ROIE]	
	CTRL2[RUIRQ]	CTRL2[RUIE]	
ipi_int_wdog	CTRL[DIRQ]	CTRL[DIE]	Watchdog timeout interrupt
ipi_int_comp	CTRL[CMPIRQ]	CTRL[CMPIE]	Compare match interrupt
ipi_int_sab	CTRL2[SABIRQ]	CTRL2[SABIE]	Simultaneous PHASEA and PHASEB change interrupt

Chapter 42
Low-Power Timer (LPTMR)

42.1 Chip-specific LPTMR information

42.1.1 LPTMR prescaler/glitch filter clocking options

The prescaler and glitch filter of the LPTMR module can be clocked from one of four sources determined by the LPTMR0_PSR[PCS] bitfield. The following table shows the chip-specific clock assignments for this bitfield.

NOTE

The chosen clock must remain enabled if the LPTMR is to continue operating in all required low-power modes.

LPTMR0_PSR[PCS]	Prescaler/glitch filter clock number	Chip clock
00	0	MCGIRCLK — internal reference clock (not available in VLPS/VLLS modes)
01	1	LPO — 1 kHz clock (not available in VLLS0 mode)
10	2	ERCLK32K — secondary external reference clock
11	3	OSCERCLK_UNDIV — Undivided external reference clock (not available in VLLS0 mode)

See Clock Distribution for more details on these clocks.

42.1.2 LPTMR pulse counter input options

The LPTMR_CSR[TPS] bitfield configures the input source used in pulse counter mode. The following table shows the chip-specific input assignments for this bitfield.

LPTMR_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0 output
01	1	LPTMR_ALT1 pin
10	2	LPTMR_ALT2 pin
11	3	LPTMR_ALT3 pin

42.2 Introduction

The low-power timer (LPTMR) can be configured to operate as a time counter with optional prescaler, or as a pulse counter with optional glitch filter, across all power modes, including the low-leakage modes. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

42.2.1 Features

The features of the LPTMR module include:

- 16-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wakeup from any low-power mode
  - Hardware trigger output
  - Counter supports free-running mode or reset on compare
- Configurable clock source for prescaler/glitch filter
- Configurable input source for pulse counter
  - Rising-edge or falling-edge

42.2.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

Table 42-1. Modes of operation

Modes	Description
Run	The LPTMR operates normally.
Wait	The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request.
Stop	The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request.

Table continues on the next page...

Table 42-1. Modes of operation (continued)

Modes	Description
Low-Leakage	The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request.
Debug	The LPTMR operates normally in Pulse Counter mode, but counter does not increment in Time Counter mode.

42.3 LPTMR signal descriptions

Table 42-2. LPTMR signal descriptions

Signal	I/O	Description
LPTMR0_ALT*n*	I	Pulse Counter Input pin

42.3.1 Detailed signal descriptions

Table 42-3. LPTMR interface—detailed signal descriptions

Signal	I/O	Description	
LPTMR_ALT*n*	I	Pulse Counter Input The LPTMR can select one of the input pins to be used in Pulse Counter mode.	
		State meaning	Assertion—If configured for pulse counter mode with active-high input, then assertion causes the CNR to increment. Deassertion—If configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment.
		Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.

42.4 Memory map and register definition

LPTMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_0000	Low Power Timer Control Status Register (LPTMR0_CSR)	32	R/W	0000_0000h	42.4.1/1064

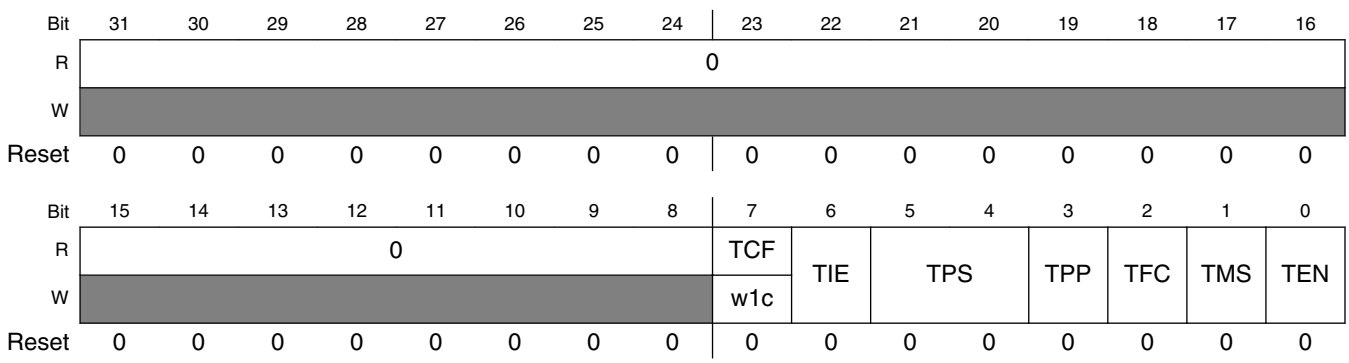
Table continues on the next page...

LPTMR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_0004	Low Power Timer Prescale Register (LPTMR0_PSR)	32	R/W	0000_0000h	42.4.2/1065
4004_0008	Low Power Timer Compare Register (LPTMR0_CMR)	32	R/W	0000_0000h	42.4.3/1067
4004_000C	Low Power Timer Counter Register (LPTMR0_CNR)	32	R/W	0000_0000h	42.4.4/1067

42.4.1 Low Power Timer Control Status Register (LPTMRx_CSR)

Address: 4004_0000h base + 0h offset = 4004_0000h



LPTMRx_CSR field descriptions

Field	Description
31–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7 TCF	Timer Compare Flag TCF is set when the LPTMR is enabled and the CNR equals the CMR and increments. TCF is cleared when the LPTMR is disabled or a logic 1 is written to it. 0 The value of CNR is not equal to CMR and increments. 1 The value of CNR is equal to CMR and increments.
6 TIE	Timer Interrupt Enable When TIE is set, the LPTMR Interrupt is generated whenever TCF is also set. 0 Timer interrupt disabled. 1 Timer interrupt enabled.
5–4 TPS	Timer Pin Select Configures the input source to be used in Pulse Counter mode. TPS must be altered only when the LPTMR is disabled. The input connections vary by device. See the for information on the connections to these inputs. 00 Pulse counter input 0 is selected. 01 Pulse counter input 1 is selected.

Table continues on the next page...

LPTMRx_CSR field descriptions (continued)

Field	Description
	10 Pulse counter input 2 is selected. 11 Pulse counter input 3 is selected.
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. TPP must be changed only when the LPTMR is disabled. 0 Pulse Counter input source is active-high, and the CNR will increment on the rising-edge. 1 Pulse Counter input source is active-low, and the CNR will increment on the falling-edge.
2 TFC	Timer Free-Running Counter When clear, TFC configures the CNR to reset whenever TCF is set. When set, TFC configures the CNR to reset on overflow. TFC must be altered only when the LPTMR is disabled. 0 CNR is reset whenever TCF is set. 1 CNR is reset on overflow.
1 TMS	Timer Mode Select Configures the mode of the LPTMR. TMS must be altered only when the LPTMR is disabled. 0 Time Counter mode. 1 Pulse Counter mode.
0 TEN	Timer Enable When TEN is clear, it resets the LPTMR internal logic, including the CNR and TCF. When TEN is set, the LPTMR is enabled. While writing 1 to this field, CSR[5:1] must not be altered. 0 LPTMR is disabled and internal logic is reset. 1 LPTMR is enabled.

42.4.2 Low Power Timer Prescale Register (LPTMRx_PSR)

Address: 4004_0000h base + 4h offset = 4004_0004h

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

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

LPTMRx_PSR field descriptions

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–3 PRESCALE	Prescale Value Configures the size of the Prescaler in Time Counter mode or width of the glitch filter in Pulse Counter mode. PRESCALE must be altered only when the LPTMR is disabled. 0000 Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration. 0001 Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges. 0010 Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges. 0011 Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges. 0100 Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges. 0101 Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges. 0110 Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges. 0111 Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges. 1000 Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges. 1001 Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges. 1010 Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges. 1011 Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges. 1100 Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges. 1101 Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges. 1110 Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges. 1111 Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges.
2 PBYP	Prescaler Bypass When PBYP is set, the selected prescaler clock in Time Counter mode or selected input source in Pulse Counter mode directly clocks the CNR. When PBYP is clear, the CNR is clocked by the output of the prescaler/glitch filter. PBYP must be altered only when the LPTMR is disabled. 0 Prescaler/glitch filter is enabled. 1 Prescaler/glitch filter is bypassed.
PCS	Prescaler Clock Select Selects the clock to be used by the LPTMR prescaler/glitch filter. PCS must be altered only when the LPTMR is disabled. The clock connections vary by device. **NOTE:** See the chip configuration details for information on the connections to these inputs.

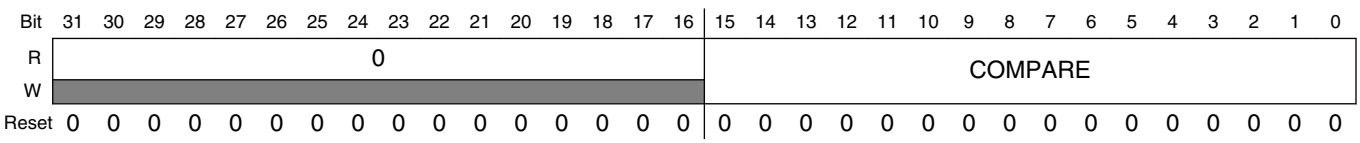
Table continues on the next page...

LPTMRx_PSR field descriptions (continued)

Field	Description
	00 Prescaler/glitch filter clock 0 selected. 01 Prescaler/glitch filter clock 1 selected. 10 Prescaler/glitch filter clock 2 selected. 11 Prescaler/glitch filter clock 3 selected.

42.4.3 Low Power Timer Compare Register (LPTMRx_CMR)

Address: 4004_0000h base + 8h offset = 4004_0008h



LPTMRx_CMR field descriptions

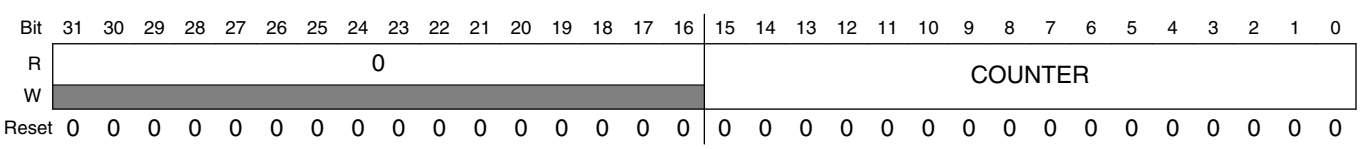
Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
COMPARE	Compare Value When the LPTMR is enabled and the CNR equals the value in the CMR and increments, TCF is set and the hardware trigger asserts until the next time the CNR increments. If the CMR is 0, the hardware trigger will remain asserted until the LPTMR is disabled. If the LPTMR is enabled, the CMR must be altered only when TCF is set.

42.4.4 Low Power Timer Counter Register (LPTMRx_CNR)

NOTE

See LPTMR counter for details on how to read counter value.

Address: 4004_0000h base + Ch offset = 4004_000Ch



LPTMRx_CNR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

LPTMRx_CNR field descriptions (continued)

Field	Description
COUNTER	Counter Value

42.5 Functional description

42.5.1 LPTMR power and reset

The LPTMR remains powered in all power modes, including low-leakage modes. If the LPTMR is not required to remain operating during a low-power mode, then it must be disabled before entering the mode.

The LPTMR is reset only on global Power On Reset (POR) or Low Voltage Detect (LVD). When configuring the LPTMR registers, the CSR must be initially written with the timer disabled, before configuring the PSR and CMR. Then, CSR[TIE] must be set as the last step in the initialization. This ensures the LPTMR is configured correctly and the LPTMR counter is reset to zero following a warm reset.

42.5.2 LPTMR clocking

The LPTMR prescaler/glitch filter can be clocked by one of the four clocks. The clock source must be enabled before the LPTMR is enabled.

NOTE

The clock source selected need to be configured to remain enabled in low-power modes, otherwise the LPTMR will not operate during low-power modes.

In Pulse Counter mode with the prescaler/glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the prescaler clock source for a clock that is not toggling.

NOTE

The clock source or pulse input source selected for the LPTMR should not exceed the frequency f_{LPTMR} defined in the device datasheet.

42.5.3 LPTMR prescaler/glitch filter

The LPTMR prescaler and glitch filter share the same logic which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

NOTE

The prescaler/glitch filter configuration must not be altered when the LPTMR is enabled.

42.5.3.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks the CNR. When the LPTMR is enabled, the CNR will increment every 2^2 to 2^{16} prescaler clock cycles. After the LPTMR is enabled, the first increment of the CNR will take an additional one or two prescaler clock cycles due to synchronization logic.

42.5.3.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments the CNR on every clock cycle. When the LPTMR is enabled, the first increment will take an additional one or two prescaler clock cycles due to synchronization logic.

42.5.3.3 Glitch filter

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks the CNR. When the LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

If	Then
The selected input source remains deasserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output will also deassert.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising-edges	The glitch filter output will also assert.

NOTE

The input is only sampled on the rising clock edge.

The CNR will increment each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which the CNR can increment is once every 2^2 to 2^{16} prescaler clock edges. When first enabled, the glitch filter will wait an additional one or two prescaler clock edges due to synchronization logic.

42.5.3.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the CNR every time it asserts. Before the LPTMR is first enabled, the selected input source is forced to be asserted. This prevents the CNR from incrementing if the selected input source is already asserted when the LPTMR is first enabled.

42.5.4 LPTMR compare

When the CNR equals the value of the CMR and increments, the following events occur:

- CSR[TCF] is set.
- LPTMR interrupt is generated if CSR[TIE] is also set.
- LPTMR hardware trigger is generated.
- CNR is reset if CSR[TFC] is clear.

When the LPTMR is enabled, the CMR can be altered only when CSR[TCF] is set. When updating the CMR, the CMR must be written and CSR[TCF] must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

42.5.5 LPTMR counter

The CNR increments by one on every:

- Prescaler clock in Time Counter mode with prescaler bypassed
- Prescaler output in Time Counter mode with prescaler enabled
- Input source assertion in Pulse Counter mode with glitch filter bypassed
- Glitch filter output in Pulse Counter mode with glitch filter enabled

The CNR is reset when the LPTMR is disabled or if the counter register overflows. If CSR[TFC] is cleared, then the CNR is also reset whenever CSR[TCF] is set.

The CNR continues incrementing when the core is halted in Debug mode when configured for Pulse Counter mode, the CNR will stop incrementing when the core is halted in Debug mode when configured for Time Counter mode.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This will synchronize and register the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

42.5.6 LPTMR hardware trigger

The LPTMR hardware trigger asserts at the same time the CSR[TCF] is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

When	Then
The CMR is set to 0 with CSR[TFC] clear	The LPTMR hardware trigger will assert on the first compare and does not deassert.
The CMR is set to a nonzero value, or, if CSR[TFC] is set	The LPTMR hardware trigger will assert on each compare and deassert on the following increment of the CNR.

42.5.7 LPTMR interrupt

The LPTMR interrupt is generated whenever CSR[TIE] and CSR[TCF] are set. CSR[TCF] is cleared by disabling the LPTMR or by writing a logic 1 to it.

CSR[TIE] can be altered and CSR[TCF] can be cleared while the LPTMR is enabled.

The LPTMR interrupt is generated asynchronously to the system clock and can be used to generate a wakeup from any low-power mode, including the low-leakage modes, provided the LPTMR is enabled as a wakeup source.

Chapter 43
Flex Controller Area Network (FlexCAN)

43.1 Chip-specific FlexCAN information

43.1.1 FlexCAN instantiation

This device contains two FlexCAN modules.

43.1.2 FlexCAN3 glitch filter

This chip supports wakeup from the FlexCAN3 module's Stop and Doze mode through a CAN wakeup interrupt. Any recessive to dominant transition on the CAN bus (CAN_RX) can wake the chip from Stop or Doze mode. An optional glitch filter is connected on CAN_RX to the interrupt generation logic path.

The glitch filter provides the following functionality:

- Filtering out of unwanted noise on the CAN bus
- Selection of the wakeup source, either from the filtered or unfiltered CAN bus
- Routing of the wakeup source to either the synchronous (Doze) or asynchronous (Stop) wakeup path within the FlexCAN module

The reference clock for the glitch filter is a 4 MHz clock derived from the MCGIRCLK. The MCGIRCLK must be configured to be 4 MHz and must remain on if the user wants a low power wakeup through the glitch filter. The glitch filter counts 11 cycles of the 4 MHz clock before recognizing it as a valid recessive to dominant transition.

43.1.3 FlexCAN3 Supervisor Mode

The module's MCR[SUPV] field configures the FlexCAN to be in either Supervisor or User Mode. On this chip:

- MCR[SUPV] is always 1: the FlexCAN is in Supervisor Mode.
- Writes to MCR[SUPV] have no effect.

43.1.4 FlexCAN signals

Signal	I/O	Connected to
FkexCAN0		
Receive Mailbox full	Output	DMA_MUX source 14
FlexCAN1		
Receive Mailbox full	Output	DMA_MUX source 15

43.2 Introduction

The FlexCAN module is a communication controller implementing the CAN protocol according to the ISO 11898-1 standard and CAN 2.0 B protocol specifications. A general block diagram is shown in the following figure, which describes the main subblocks implemented in the FlexCAN module, including one associated memory for storing message buffers, Receive Global Mask registers, Receive Individual Mask registers, Receive FIFO filters, and Receive FIFO ID filters. The functions of the submodules are described in subsequent sections.

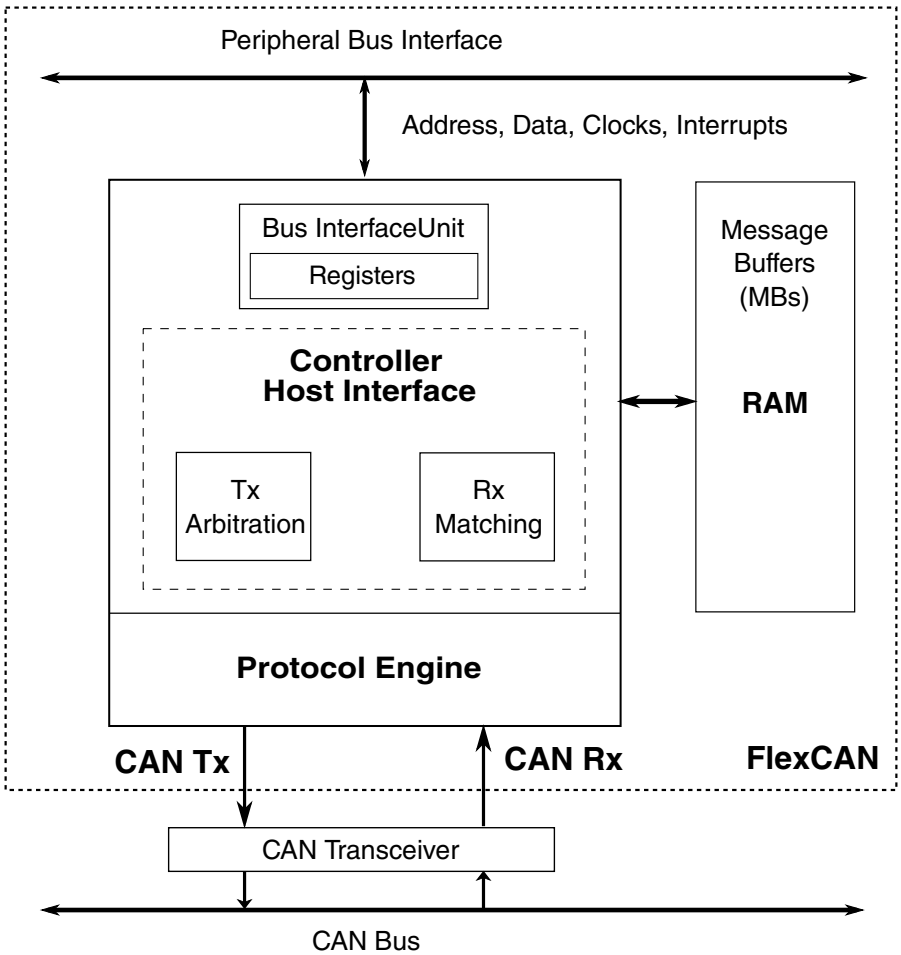


Figure 43-1. FlexCAN block diagram

43.2.1 Overview

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific requirements of this field:

- Real-time processing
- Reliable operation in the EMI environment of a vehicle
- Cost-effectiveness
- Required bandwidth

The FlexCAN module is a full implementation of the CAN protocol specification, the CAN 2.0 version B protocol, which supports both standard and extended message frames. The message buffers are stored in an embedded RAM dedicated to the FlexCAN module. See the chip configuration details for the actual number of message buffers configured in the chip.

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- Requesting RAM access for receiving and transmitting message frames
- Validating received messages
- Performing error handling

The Controller Host Interface (CHI) sub-module handles message buffer selection for reception and transmission, taking care of arbitration and ID matching algorithms.

The Bus Interface Unit (BIU) sub-module controls the access to and from the internal interface bus, in order to establish connection to the CPU and to other blocks. Clocks, address and data buses, interrupt outputs, DMA and test signals are accessed through the BIU.

43.2.2 FlexCAN module features

The FlexCAN module includes these distinctive features:

- Full implementation of the CAN protocol specification, Version 2.0 B
  - Standard data frames
  - Extended data frames
  - Zero to eight bytes data length
  - Programmable bit rate up to 1 Mb/sec
  - Content-related addressing
- Compliant with the ISO 11898-1 standard
- Flexible mailboxes of zero to eight bytes data length
- Each mailbox configurable as receive or transmit, all supporting standard and extended messages
- Individual Rx Mask registers per mailbox
- Full-featured Rx FIFO with storage capacity for up to six frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Flexible message buffers (MBs), totaling 16 message buffers of 8 bytes data length each, configurable as Rx or Tx

- Programmable clock source to the CAN Protocol Interface, either peripheral clock or oscillator clock
- RAM not used by reception or transmission structures can be used as general purpose RAM space
- Listen-Only mode capability
- Programmable Loop-Back mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Time stamp based on 16-bit free-running timer
- Global network time, synchronized by a specific message
- Maskable interrupts
- Independence from the transmission medium (an external transceiver is assumed)
- Short latency time due to an arbitration scheme for high-priority messages
- Low power modes, with programmable wake up on bus activity
- Remote request frames may be handled automatically or by software
- CAN bit time settings and configuration bits can only be written in Freeze mode
- Tx mailbox status (Lowest priority buffer or empty buffer)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- SYNCH bit available in Error in Status 1 register to inform that the module is synchronous with CAN bus
- CRC status for transmitted message
- Rx FIFO Global Mask register
- Selectable priority between mailboxes and Rx FIFO during matching process
- Powerful Rx FIFO ID filtering, capable of matching incoming IDs against either 128 extended, 256 standard, or 512 partial (8 bit) IDs, with up to 32 individual masking capability
- 100% backward compatibility with previous FlexCAN version

43.2.3 Modes of operation

The FlexCAN module has these functional modes:

- Normal mode (User or Supervisor):

  In Normal mode, the module operates receiving and/or transmitting message frames, errors are handled normally, and all CAN Protocol functions are enabled. User and Supervisor Modes differ in the access to some restricted control registers.

- Freeze mode:

  Freeze mode is enabled when the FRZ bit in MCR is asserted. If enabled, Freeze mode is entered when MCR[HALT] is set or when Debug mode is requested at chip level and MCR[FRZ_ACK] is asserted by the FlexCAN. In this mode, no transmission or reception of frames is done and synchronicity to the CAN bus is lost. See Freeze mode for more information.

- Loop-Back mode:

  The module enters this mode when the LPB field in the Control 1 Register is asserted. In this mode, FlexCAN performs an internal loop back that can be used for self-test operation. The bit stream output of the transmitter is internally fed back to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic '1'). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

- Listen-Only mode:

  The module enters this mode when the LOM field in the Control 1 Register is asserted. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.

For low-power operation, the FlexCAN module has:

- Module Disable mode:

This low-power mode is entered when the MDIS bit in the MCR Register is asserted by the CPU and the LPM_ACK is asserted by the FlexCAN. When disabled, the module requests to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Exit from this mode is done by negating the MDIS bit in the MCR register. See Module Disable mode for more information.

- Doze mode:

  This low power mode is entered when the DOZE bit in MCR is asserted and Doze mode is requested at chip level and the LPM_ACK bit in the MCR Register is asserted by the FlexCAN. When in Doze mode, the module requests to disable the clocks to the CAN Protocol Engine and the CAN Controller-Host Interface sub-modules. Exit from this mode happens when the DOZE bit in MCR is negated, when the chip is removed from Doze mode, or when activity is detected on the CAN bus and the Self Wake Up mechanism is enabled. See Doze mode for more information.

- Stop mode:

  This low power mode is entered when Stop mode is requested at chip level and the LPM_ACK bit in the MCR Register is asserted by the FlexCAN. When in Stop Mode, the module puts itself in an inactive state and then informs the CPU that the clocks can be shut down globally. Exit from this mode happens when the Stop mode request is removed, or when activity is detected on the CAN bus and the Self Wake Up mechanism is enabled. See Stop mode for more information.

43.3 FlexCAN signal descriptions

The FlexCAN module has two I/O signals connected to the external chip pins. These signals are summarized in the following table and described in more detail in the next subsections.

Table 43-1. FlexCAN signal descriptions

Signal	Description	I/O
CAN Rx	CAN Receive Pin	Input
CAN Tx	CAN Transmit Pin	Output

43.3.1 CAN Rx

This pin is the receive pin from the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

43.3.2 CAN Tx

This pin is the transmit pin to the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

43.4 Memory map/register definition

This section describes the registers and data structures in the FlexCAN module. The base address of the module depends on the particular memory map of the chip.

43.4.1 FlexCAN memory mapping

The memory map for the FlexCAN module is shown in the following table.

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0x0080.

Each individual register is identified by its complete name and the corresponding mnemonic. The access type can be Supervisor (S) or Unrestricted (U). Most of the registers can be configured to have either Supervisor or Unrestricted access by programming the SUPV field in the MCR register. These registers are identified as S/U in the Access column of Table 43-2.

Table 43-2. Register access and reset information

Register	Access type	Affected by hard reset	Affected by soft reset
Module Configuration Register (CAN_MCR)	S	Yes	Yes
Control 1 register (CAN_CTRL1)	S/U	Yes	No
Free Running Timer register (CAN_TIMER)	S/U	Yes	Yes
Rx Mailboxes Global Mask register (CAN_RXMGMASK)	S/U	No	No
Rx Buffer 14 Mask register (CAN_RX14MASK)	S/U	No	No
Rx Buffer 15 Mask register (CAN_RX15MASK)	S/U	No	No
Error Counter Register (CAN_ECR)	S/U	Yes	Yes
Error and Status 1 Register (CAN_ESR1)	S/U	Yes	Yes
Interrupt Masks 1 register (CAN_IMASK1)	S/U	Yes	Yes
Interrupt Flags 1 register (CAN_IFLAG1)	S/U	Yes	Yes
Control 2 Register (CAN_CTRL2)	S/U	Yes	No
Error and Status 2 Register (CAN_ESR2)	S/U	Yes	Yes
CRC Register (CAN_CRCR)	S/U	Yes	Yes

Table continues on the next page...

Table 43-2. Register access and reset information (continued)

Register	Access type	Affected by hard reset	Affected by soft reset
Rx FIFO Global Mask register (CAN_RXFGMASK)	S/U	No	No
Rx FIFO Information Register (CAN_RXFIR)	S/U	No	No
CAN Bit Timing Register (CAN_CBT)	S/U	Yes	No
Message buffers	S/U	No	No
Rx Individual Mask Registers	S/U	No	No

The FlexCAN module can store CAN messages for transmission and reception using mailboxes and Rx FIFO structures.

The table below shows the FlexCAN memory map.

The address range from offset 0x80 to 0x17F allocates the sixteen 128-bit Message Buffers (MBs).

CAN memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4000	Module Configuration Register (CAN0_MCR)	32	R/W	See section	43.4.2/1084
4002_4004	Control 1 register (CAN0_CTRL1)	32	R/W	0000_0000h	43.4.3/1089
4002_4008	Free Running Timer (CAN0_TIMER)	32	R/W	0000_0000h	43.4.4/1093
4002_4010	Rx Mailboxes Global Mask Register (CAN0_RXMGMASK)	32	R/W	Undefined	43.4.5/1094
4002_4014	Rx 14 Mask register (CAN0_RX14MASK)	32	R/W	Undefined	43.4.6/1095
4002_4018	Rx 15 Mask register (CAN0_RX15MASK)	32	R/W	Undefined	43.4.7/1096
4002_401C	Error Counter (CAN0_ECR)	32	R/W	0000_0000h	43.4.8/1096
4002_4020	Error and Status 1 register (CAN0_ESR1)	32	R/W	See section	43.4.9/1098
4002_4028	Interrupt Masks 1 register (CAN0_IMASK1)	32	R/W	0000_0000h	43.4.10/ 1104
4002_4030	Interrupt Flags 1 register (CAN0_IFLAG1)	32	R/W	0000_0000h	43.4.11/ 1104
4002_4034	Control 2 register (CAN0_CTRL2)	32	R/W	See section	43.4.12/ 1107
4002_4038	Error and Status 2 register (CAN0_ESR2)	32	R/W	0000_0000h	43.4.13/ 1110
4002_4044	CRC Register (CAN0_CRCR)	32	R	0000_0000h	43.4.14/ 1111
4002_4048	Rx FIFO Global Mask register (CAN0_RXFGMASK)	32	R/W	Undefined	43.4.15/ 1112
4002_404C	Rx FIFO Information Register (CAN0_RXFIR)	32	R	Undefined	43.4.16/ 1113

Table continues on the next page...

CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4050	CAN Bit Timing Register (CAN0_CBT)	32	R/W	See section	43.4.17/ 1114
4002_4880	Rx Individual Mask Registers (CAN0_RXIMR0)	32	R/W	Undefined	43.4.18/ 1116
4002_4884	Rx Individual Mask Registers (CAN0_RXIMR1)	32	R/W	Undefined	43.4.18/ 1116
4002_4888	Rx Individual Mask Registers (CAN0_RXIMR2)	32	R/W	Undefined	43.4.18/ 1116
4002_488C	Rx Individual Mask Registers (CAN0_RXIMR3)	32	R/W	Undefined	43.4.18/ 1116
4002_4890	Rx Individual Mask Registers (CAN0_RXIMR4)	32	R/W	Undefined	43.4.18/ 1116
4002_4894	Rx Individual Mask Registers (CAN0_RXIMR5)	32	R/W	Undefined	43.4.18/ 1116
4002_4898	Rx Individual Mask Registers (CAN0_RXIMR6)	32	R/W	Undefined	43.4.18/ 1116
4002_489C	Rx Individual Mask Registers (CAN0_RXIMR7)	32	R/W	Undefined	43.4.18/ 1116
4002_48A0	Rx Individual Mask Registers (CAN0_RXIMR8)	32	R/W	Undefined	43.4.18/ 1116
4002_48A4	Rx Individual Mask Registers (CAN0_RXIMR9)	32	R/W	Undefined	43.4.18/ 1116
4002_48A8	Rx Individual Mask Registers (CAN0_RXIMR10)	32	R/W	Undefined	43.4.18/ 1116
4002_48AC	Rx Individual Mask Registers (CAN0_RXIMR11)	32	R/W	Undefined	43.4.18/ 1116
4002_48B0	Rx Individual Mask Registers (CAN0_RXIMR12)	32	R/W	Undefined	43.4.18/ 1116
4002_48B4	Rx Individual Mask Registers (CAN0_RXIMR13)	32	R/W	Undefined	43.4.18/ 1116
4002_48B8	Rx Individual Mask Registers (CAN0_RXIMR14)	32	R/W	Undefined	43.4.18/ 1116
4002_48BC	Rx Individual Mask Registers (CAN0_RXIMR15)	32	R/W	Undefined	43.4.18/ 1116
4002_5000	Module Configuration Register (CAN1_MCR)	32	R/W	See section	43.4.2/1084
4002_5004	Control 1 register (CAN1_CTRL1)	32	R/W	0000_0000h	43.4.3/1089
4002_5008	Free Running Timer (CAN1_TIMER)	32	R/W	0000_0000h	43.4.4/1093
4002_5010	Rx Mailboxes Global Mask Register (CAN1_RXMGMASK)	32	R/W	Undefined	43.4.5/1094
4002_5014	Rx 14 Mask register (CAN1_RX14MASK)	32	R/W	Undefined	43.4.6/1095
4002_5018	Rx 15 Mask register (CAN1_RX15MASK)	32	R/W	Undefined	43.4.7/1096
4002_501C	Error Counter (CAN1_ECR)	32	R/W	0000_0000h	43.4.8/1096
4002_5020	Error and Status 1 register (CAN1_ESR1)	32	R/W	See section	43.4.9/1098

Table continues on the next page...

CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_5028	Interrupt Masks 1 register (CAN1_IMASK1)	32	R/W	0000_0000h	43.4.10/ 1104
4002_5030	Interrupt Flags 1 register (CAN1_IFLAG1)	32	R/W	0000_0000h	43.4.11/ 1104
4002_5034	Control 2 register (CAN1_CTRL2)	32	R/W	See section	43.4.12/ 1107
4002_5038	Error and Status 2 register (CAN1_ESR2)	32	R/W	0000_0000h	43.4.13/ 1110
4002_5044	CRC Register (CAN1_CRCR)	32	R	0000_0000h	43.4.14/ 1111
4002_5048	Rx FIFO Global Mask register (CAN1_RXFGMASK)	32	R/W	Undefined	43.4.15/ 1112
4002_504C	Rx FIFO Information Register (CAN1_RXFIR)	32	R	Undefined	43.4.16/ 1113
4002_5050	CAN Bit Timing Register (CAN1_CBT)	32	R/W	See section	43.4.17/ 1114
4002_5880	Rx Individual Mask Registers (CAN1_RXIMR0)	32	R/W	Undefined	43.4.18/ 1116
4002_5884	Rx Individual Mask Registers (CAN1_RXIMR1)	32	R/W	Undefined	43.4.18/ 1116
4002_5888	Rx Individual Mask Registers (CAN1_RXIMR2)	32	R/W	Undefined	43.4.18/ 1116
4002_588C	Rx Individual Mask Registers (CAN1_RXIMR3)	32	R/W	Undefined	43.4.18/ 1116
4002_5890	Rx Individual Mask Registers (CAN1_RXIMR4)	32	R/W	Undefined	43.4.18/ 1116
4002_5894	Rx Individual Mask Registers (CAN1_RXIMR5)	32	R/W	Undefined	43.4.18/ 1116
4002_5898	Rx Individual Mask Registers (CAN1_RXIMR6)	32	R/W	Undefined	43.4.18/ 1116
4002_589C	Rx Individual Mask Registers (CAN1_RXIMR7)	32	R/W	Undefined	43.4.18/ 1116
4002_58A0	Rx Individual Mask Registers (CAN1_RXIMR8)	32	R/W	Undefined	43.4.18/ 1116
4002_58A4	Rx Individual Mask Registers (CAN1_RXIMR9)	32	R/W	Undefined	43.4.18/ 1116
4002_58A8	Rx Individual Mask Registers (CAN1_RXIMR10)	32	R/W	Undefined	43.4.18/ 1116
4002_58AC	Rx Individual Mask Registers (CAN1_RXIMR11)	32	R/W	Undefined	43.4.18/ 1116
4002_58B0	Rx Individual Mask Registers (CAN1_RXIMR12)	32	R/W	Undefined	43.4.18/ 1116
4002_58B4	Rx Individual Mask Registers (CAN1_RXIMR13)	32	R/W	Undefined	43.4.18/ 1116

Table continues on the next page...

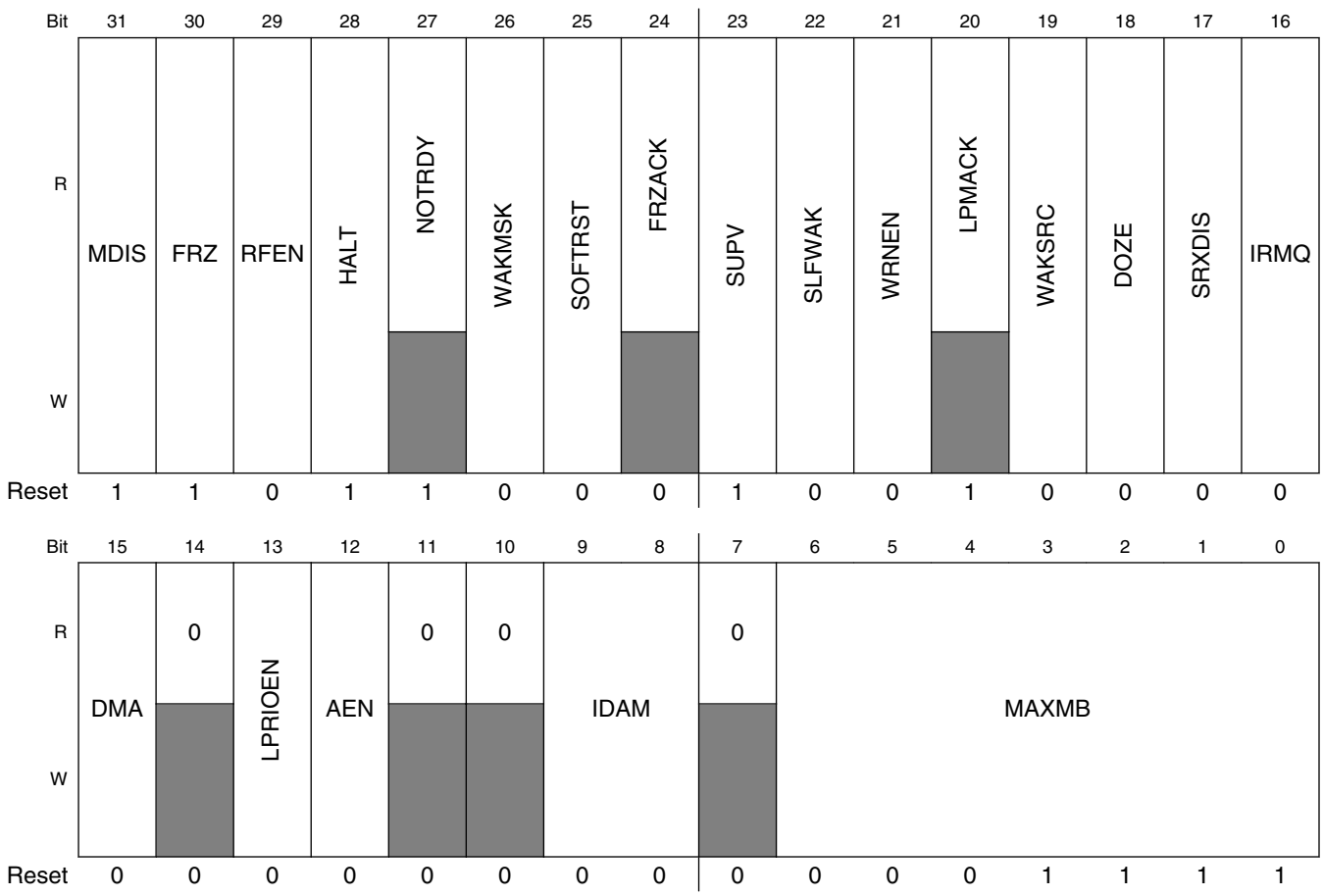
CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_58B8	Rx Individual Mask Registers (CAN1_RXIMR14)	32	R/W	Undefined	43.4.18/ 1116
4002_58BC	Rx Individual Mask Registers (CAN1_RXIMR15)	32	R/W	Undefined	43.4.18/ 1116

43.4.2 Module Configuration Register (CANx_MCR)

This register defines global system configurations, such as the module operation modes and the maximum message buffer configuration.

Address: Base address + 0h offset



CANx_MCR field descriptions

Field	Description
31 MDIS	Module Disable This bit controls whether FlexCAN is enabled or not. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface sub-modules. This bit is not affected by soft reset.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	0 Enable the FlexCAN module. 1 Disable the FlexCAN module.
30 FRZ	Freeze Enable The FRZ bit specifies the FlexCAN behavior when the HALT bit in the CAN_MCR Register is set or when Debug mode is requested at chip level. When FRZ is asserted, FlexCAN is enabled to enter Freeze mode. Negation of this bit field causes FlexCAN to exit from Freeze mode. 0 Not enabled to enter Freeze mode. 1 Enabled to enter Freeze mode.
29 RFEN	Rx FIFO Enable This bit controls whether the Rx FIFO feature is enabled or not. When RFEN is set, MBs 0 to 5 cannot be used for normal reception and transmission because the corresponding memory region (0x80-0xDC) is used by the FIFO engine as well as additional MBs (up to 32, depending on CAN_CTRL2[RFFN] setting) which are used as Rx FIFO ID Filter Table elements. RFEN also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in the table "Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate" (see Arbitration and matching timing). This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Rx FIFO not enabled. 1 Rx FIFO enabled.
28 HALT	Halt FlexCAN Assertion of this bit puts the FlexCAN module into Freeze mode. The CPU should clear it after initializing the Message Buffers and the Control Registers CAN_CTRL1 and CAN_CTRL2. No reception or transmission is performed by FlexCAN before this bit is cleared. Freeze mode cannot be entered while FlexCAN is in a low power mode. 0 No Freeze mode request. 1 Enters Freeze mode if the FRZ bit is asserted.
27 NOTRDY	FlexCAN Not Ready This read-only bit indicates that FlexCAN is either in Disable mode, Doze mode, Stop mode or Freeze mode. It is negated once FlexCAN has exited these modes. This bit is not affected by soft reset. 0 FlexCAN module is either in Normal mode, Listen-Only mode or Loop-Back mode. 1 FlexCAN module is either in Disable mode, Doze mode, Stop mode or Freeze mode.
26 WAKMSK	Wake Up Interrupt Mask This bit enables the Wake Up Interrupt generation under Self Wake Up mechanism. 0 Wake Up Interrupt is disabled. 1 Wake Up Interrupt is enabled.
25 SOFTRST	Soft Reset When this bit is asserted, FlexCAN resets its internal state machines and some of the memory mapped registers.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	The SOFTRST bit can be asserted directly by the CPU when it writes to the MCR Register, but it is also asserted when global soft reset is requested at chip level. Because soft reset is synchronous and has to follow a request/acknowledge procedure across clock domains, it may take some time to fully propagate its effect. The SOFTRST bit remains asserted while reset is pending, and is automatically negated when reset completes. Therefore, software can poll this bit to know when the soft reset has completed. Soft reset cannot be applied while clocks are shut down in a low power mode. The module should be first removed from low power mode, and then soft reset can be applied. This bit is not affected by soft reset. 0 No reset request. 1 Resets the registers affected by soft reset.
24 FRZACK	Freeze Mode Acknowledge This read-only bit indicates that FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore the software can poll the FRZACK bit to know when FlexCAN has actually entered Freeze mode. If Freeze Mode request is negated, then this bit is negated after the FlexCAN prescaler is running again. If Freeze mode is requested while FlexCAN is in a low power mode, then the FRZACK bit will be set only when the low-power mode is exited. See Section "Freeze Mode". This bit is not affected by soft reset. **NOTE:** FRZACK will be asserted within 178 CAN bits from the freeze mode request by the CPU, and negated within 2 CAN bits after the freeze mode request removal (see Protocol timing). 0 FlexCAN not in Freeze mode, prescaler running. 1 FlexCAN in Freeze mode, prescaler stopped.
23 SUPV	Supervisor Mode This bit configures the FlexCAN to be either in Supervisor or User mode. The registers affected by this bit are marked as S/U in the Access Type column of the module memory map. Reset value of this bit is 1, so the affected registers start with Supervisor access allowance only. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 FlexCAN is in User mode. Affected registers allow both Supervisor and Unrestricted accesses. 1 FlexCAN is in Supervisor mode. Affected registers allow only Supervisor access. Unrestricted access behaves as though the access was done to an unimplemented register location.
22 SLFWAK	Self Wake Up This bit enables the Self Wake Up feature when FlexCAN is in a low-power mode other than Disable mode. When this feature is enabled, the FlexCAN module monitors the bus for wake up event, that is, a recessive-to-dominant transition. If a wake up event is detected during Doze mode, FlexCAN requests to resume its clocks and, if enabled to do so, generates a Wake Up interrupt to the CPU. If a wake up event is detected during Stop mode, then FlexCAN generates, if enabled to do so, a Wake Up interrupt to the CPU so that it can exit Stop mode globally and FlexCAN can request to resume the clocks. When FlexCAN is in a low-power mode other than Disable mode, this bit cannot be written as it is blocked by hardware. 0 FlexCAN Self Wake Up feature is disabled. 1 FlexCAN Self Wake Up feature is enabled.
21 WRNEN	Warning Interrupt Enable

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	When asserted, this bit enables the generation of the TWRNINT and RWRNINT flags in the Error and Status Register 1 (ESR1). If WRNEN is negated, the TWRNINT and RWRNINT flags will always be zero, independent of the values of the error counters, and no warning interrupt will ever be generated. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 TWRNINT and RWRNINT bits are zero, independent of the values in the error counters. 1 TWRNINT and RWRNINT bits are set when the respective error counter transitions from less than 96 to greater than or equal to 96.
20 LPMACK	Low-Power Mode Acknowledge This read-only bit indicates that FlexCAN is in a low-power mode (Disable mode, Doze mode, Stop mode). A low-power mode cannot be entered until all current transmission or reception processes have finished, so the CPU can poll the LPMACK bit to know when FlexCAN has actually entered low power mode. This bit is not affected by soft reset. **NOTE:** LPMACK will be asserted within 180 CAN bits from the low-power mode request by the CPU, and negated within 2 CAN bits after the low-power mode request removal (see Protocol timing). 0 FlexCAN is not in a low-power mode. 1 FlexCAN is in a low-power mode.
19 WAKSRC	Wake Up Source This bit defines whether the integrated low-pass filter is applied to protect the Rx CAN input from spurious wake up. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 FlexCAN uses the unfiltered Rx input to detect recessive to dominant edges on the CAN bus. 1 FlexCAN uses the filtered Rx input to detect recessive to dominant edges on the CAN bus.
18 DOZE	Doze Mode Enable This bit defines whether FlexCAN is allowed to enter low-power mode when Doze mode is requested at chip level . This bit is automatically reset when FlexCAN wakes up from Doze mode upon detecting activity on the CAN bus (self wake-up enabled). 0 FlexCAN is not enabled to enter low-power mode when Doze mode is requested. 1 FlexCAN is enabled to enter low-power mode when Doze mode is requested.
17 SRXDIS	Self Reception Disable This bit defines whether FlexCAN is allowed to receive frames transmitted by itself. If this bit is asserted, frames transmitted by the module will not be stored in any MB, regardless if the MB is programmed with an ID that matches the transmitted frame, and no interrupt flag or interrupt signal will be generated due to the frame reception. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 Self reception enabled. 1 Self reception disabled.
16 IRMQ	Individual Rx Masking And Queue Enable This bit indicates whether Rx matching process will be based either on individual masking and queue or on masking scheme with CAN_RXMGMASK, CAN_RX14MASK, CAN_RX15MASK and CAN_RXFGMASK. This bit can be written in Freeze mode only because it is blocked by hardware in other modes.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	0 Individual Rx masking and queue feature are disabled. For backward compatibility with legacy applications, the reading of C/S word locks the MB even if it is EMPTY. 1 Individual Rx masking and queue feature are enabled.
15 DMA	DMA Enable The DMA Enable bit controls whether the DMA feature is enabled or not. The DMA feature can only be used in Rx FIFO, consequently the bit CAN_MCR[RFEN] must be asserted. When DMA and RFEN are set, the CAN_IFLAG1[BUF5I] generates the DMA request and no RX FIFO interrupt is generated. This bit can be written in Freeze mode only as it is blocked by hardware in other modes. 0 DMA feature for RX FIFO disabled. 1 DMA feature for RX FIFO enabled.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 LPRIOEN	Local Priority Enable This bit is provided for backwards compatibility with legacy applications. It controls whether the local priority feature is enabled or not. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word, but the actual transmitted ID still has 11-bit for standard frames and 29-bit for extended frames. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0 Local Priority disabled. 1 Local Priority enabled.
12 AEN	Abort Enable When asserted, this bit enables the Tx abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE:** When CAN_MCR[AEN] is asserted, only the abort mechanism (see Transmission abort mechanism) must be used for updating Mailboxes configured for transmission. **CAUTION:** Writing the Abort code into Rx Mailboxes can cause unpredictable results when the CAN_MCR[AEN] is asserted. 0 Abort disabled. 1 Abort enabled.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9–8 IDAM	ID Acceptance Mode This 2-bit field identifies the format of the Rx FIFO ID Filter Table elements. Note that all elements of the table are configured at the same time by this field (they are all the same format). See Section "Rx FIFO Structure". This field can be written only in Freeze mode because it is blocked by hardware in other modes. 00 Format A: One full ID (standard and extended) per ID Filter Table element. 01 Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID Filter Table element.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	10 Format C: Four partial 8-bit Standard IDs per ID Filter Table element. 11 Format D: All frames rejected.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MAXMB	Number Of The Last Message Buffer This 7-bit field defines the number of the last Message Buffers that will take part in the matching and arbitration processes. The reset value (0x0F) is equivalent to a 16 MB configuration. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Number of the last MB = MAXMB **NOTE:** MAXMB must be programmed with a value smaller than or equal to the number of available Message Buffers. Additionally, the definition of MAXMB value must take into account the region of MBs occupied by Rx FIFO and its ID filters table space defined by RFFN bit in CAN_CTRL2 register. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table "Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate" (see Arbitration and matching timing).

43.4.3 Control 1 register (CANx_CTRL1)

This register is defined for specific FlexCAN control features related to the CAN bus, such as bit-rate, programmable sampling point within an Rx bit, Loop Back mode, Listen-Only mode, Bus Off recovery behavior and interrupt enabling (Bus-Off, Error, Warning). It also determines the Division Factor for the clock prescaler.

The CAN bit timing variables (PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW) can also be configured in CAN_CBT register, which extends the range of all these variables. If CAN_CBT[BTF] is set, PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW fields of CAN_CTRL1 become read only.

The contents of this register are not affected by soft reset.

NOTE

The CAN bit variables in CAN_CTRL1 and in CAN_CBT are stored in the same register.

NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRESDIV								RJW		PSEG1			PSEG2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BOFFMSK	ERRMSK	CLKSRC	LPB	TWRNMSK	RWRNMSK	Reserved	Reserved	SMP	BOFFREC	TSYN	LBUF	LOM	PROPSEG		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_CTRL1 field descriptions

Field	Description
31–24 PRESDIV	Prescaler Division Factor This 8-bit field defines the ratio between the PE clock frequency and the Serial Clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The Maximum value of this field is 0xFF, that gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing . This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (PRESDIV + 1)
23–22 RJW	Resync Jump Width This 2-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization. One time quantum is equal to the Sclock period. The valid programmable values are 0–3. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = RJW + 1.
21–19 PSEG1	Phase Segment 1 This 3-bit field defines the length of Phase Segment 1 in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) × Time-Quanta.
18–16 PSEG2	Phase Segment 2 This 3-bit field defines the length of Phase Segment 2 in the bit time. The valid programmable values are 1–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time-Quanta.

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
15 BOFFMSK	Bus Off Interrupt Mask This bit provides a mask for the Bus Off Interrupt BOFFINT in CAN_ESR1 register. 0 Bus Off interrupt disabled. 1 Bus Off interrupt enabled.
14 ERRMSK	Error Interrupt Mask This bit provides a mask for the Error Interrupt ERRINT in the CAN_ESR1 register. 0 Error interrupt disabled. 1 Error interrupt enabled.
13 CLKSRC	CAN Engine Clock Source This bit selects the clock source to the CAN Protocol Engine (PE) to be either the peripheral clock or the oscillator clock. The selected clock is the one fed to the prescaler to generate the Serial Clock (Sclock). In order to guarantee reliable operation, this bit can be written only in Disable mode because it is blocked by hardware in other modes. See Protocol timing. **NOTE:** Please refer to the clock distribution chapter (module clocks table) to identify the proper clock source. **NOTE:** The user must ensure the protocol engine clock tolerance according to the CAN Protocol standard (ISO 11898-1). 0 The CAN engine clock source is the oscillator clock. Under this condition, the oscillator clock frequency must be lower than the bus clock. 1 The CAN engine clock source is the peripheral clock.
12 LPB	Loop Back Mode This bit configures FlexCAN to operate in Loop-Back mode. In this mode, FlexCAN performs an internal loop back that can be used for self test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field, generating an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE:** In this mode, the CAN_MCR[SRXDIS] cannot be asserted because this will impede the self reception of a transmitted message. 0 Loop Back disabled. 1 Loop Back enabled.
11 TWRNMSK	Tx Warning Interrupt Mask This bit provides a mask for the Tx Warning Interrupt associated with the TWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when CAN_MCR[WRNEN] bit is negated. This bit can be written only if CAN_MCR[WRNEN] bit is asserted. 0 Tx Warning Interrupt disabled. 1 Tx Warning Interrupt enabled.

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
10 RWRNMSK	Rx Warning Interrupt Mask This bit provides a mask for the Rx Warning Interrupt associated with the RWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when CAN_MCR[WRNEN] bit is negated. This bit can be written only if CAN_MCR[WRNEN] bit is asserted. 0 Rx Warning Interrupt disabled. 1 Rx Warning Interrupt enabled.
9 Reserved	This field is reserved.
8 Reserved	This field is reserved.
7 SMP	CAN Bit Sampling This bit defines the sampling mode of CAN bits at the Rx input. It can be written in Freeze mode only because it is blocked by hardware in other modes. **NOTE:** For proper operation, to assert SMP it is necessary to guarantee a minimum value of 2 TQs in CAN_CTRL1[PSEG1] (or CAN_CBT[EPSEG1]). 0 Just one sample is used to determine the bit value. 1 Three samples are used to determine the value of the received bit: the regular one (sample point) and 2 preceding samples; a majority rule is used.
6 BOFFREC	Bus Off Recovery This bit defines how FlexCAN recovers from Bus Off state. If this bit is negated, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If the bit is asserted, automatic recovering from Bus Off is disabled and the module remains in Bus Off state until the bit is negated by the user. If the negation occurs before 128 sequences of 11 recessive bits are detected on the CAN bus, then Bus Off recovery happens as if the BOFFREC bit had never been asserted. If the negation occurs after 128 sequences of 11 recessive bits occurred, then FlexCAN will re-synchronize to the bus by waiting for 11 recessive bits before joining the bus. After negation, the BOFFREC bit can be re-asserted again during Bus Off, but it will be effective only the next time the module enters Bus Off. If BOFFREC was negated when the module entered Bus Off, asserting it during Bus Off will not be effective for the current Bus Off recovery. **NOTE:** Refer to Bus off in the CAN Protocol standard (ISO 11898-1) for details. 0 Automatic recovering from Bus Off state enabled. 1 Automatic recovering from Bus Off state disabled.
5 TSYN	Timer Sync This bit enables a mechanism that resets the free-running timer each time a message is received in Message Buffer 0. This feature provides means to synchronize multiple FlexCAN stations with a special “SYNC” message, that is, global network time. If the RFEN bit in CAN_MCR is set (Rx FIFO enabled), the first available Mailbox, according to CAN_CTRL2[RFFN] setting, is used for timer synchronization instead of MB0. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Timer Sync feature disabled 1 Timer Sync feature enabled
4 LBUF	Lowest Buffer Transmitted First

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
	This bit defines the ordering mechanism for Message Buffer transmission. When asserted, the CAN_MCR[LPRIOEN] bit does not affect the priority arbitration. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Buffer with highest priority is transmitted first. 1 Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode This bit configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in CAN_ECR register are frozen and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error without changing the receive error counter (RXERRCNT) in CAN_ECR register, as if it was trying to acknowledge the message. Listen-Only mode is acknowledged by the state of CAN_ESR1[FLTCONF] field indicating Passive Error. There can be some delay between the Listen-Only mode request and acknowledge. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0 Listen-Only mode is deactivated. 1 FlexCAN module operates in Listen-Only mode.
PROPSEG	Propagation Segment This 3-bit field defines the length of the Propagation Segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = (PROPSEG + 1) × Time-Quanta. Time-Quantum = one Sclock period.

43.4.4 Free Running Timer (CANx_TIMER)

This register represents a 16-bit free running counter that can be read and written by the CPU. The timer starts from 0x0 after Reset, counts linearly to 0xFFFF, and wraps around.

The timer is incremented by the CAN bit clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable, Doze, Stop and Freeze modes.

The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the Time Stamp entry in a message buffer after a successful reception or transmission of a message.

If bit CAN_CTRL1[TSYN] is asserted, the Timer is reset whenever a message is received in the first available Mailbox, according to CAN_CTRL2[RFFN] setting.

The CPU can write to this register anytime. However, if the write occurs at the same time that the Timer is being reset by a reception in the first Mailbox, then the write value is discarded.

Reading this register affects the Mailbox Unlocking procedure, see Section "Mailbox Lock Mechanism".

Address: Base address + 8h offset

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

CANx_TIMER field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TIMER	Timer Value Contains the free-running counter value.

43.4.5 Rx Mailboxes Global Mask Register (CANx_RXMGMASK)

This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When the CAN_MCR[IRMQ] bit is negated, RXMGMASK is always in effect (the bits in the MG field will mask the Mailbox filter bits).
- When the CAN_MCR[IRMQ] bit is asserted, RXMGMASK has no effect (the bits in the MG field will not mask the Mailbox filter bits).

RXMGMASK is used to mask the filter fields of all Rx MBs, excluding MBs 14-15, which have individual mask registers.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Address: Base address + 10h offset

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

* Notes:
- x = Undefined at reset.

CANx_RXMGMASK field descriptions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>MG</td><td>Rx Mailboxes Global Mask Bits

These bits mask the Mailbox filter bits. Note that the alignment with the ID word of the Mailbox is not perfect as the two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the Mailbox. The following table shows in detail which MG bits mask each Mailbox filter field.

<table>
<tr><th rowspan="2">SMB[RTR] [1]</th><th rowspan="2">CAN_CTRL2[RRS]</th><th rowspan="2">CAN_CTRL2[EACEN]</th><th colspan="4">Mailbox filter fields</th></tr>
<tr><th>MB[RTR]</th><th>MB[IDE]</th><th>MB[ID]</th><th>Reserved</th></tr>
<tr><td>0</td><td>-</td><td>0</td><td>note [2]</td><td>note [3]</td><td>MG[28:0]</td><td>MG[31:29]</td></tr>
<tr><td>0</td><td>-</td><td>1</td><td>MG[31]</td><td>MG[30]</td><td>MG[28:0]</td><td>MG[29]</td></tr>
<tr><td>1</td><td>0</td><td>-</td><td>-</td><td>-</td><td>-</td><td>MG[31:0]</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>-</td><td>-</td><td>MG[28:0]</td><td>MG[31:29]</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>MG[31]</td><td>MG[30]</td><td>MG[28:0]</td><td>MG[29]</td></tr>
</table>
0 The corresponding bit in the filter is "don't care."
1 The corresponding bit in the filter is checked.</td></tr>
</table>

1. RTR bit of the Incoming Frame. It is saved into an auxiliary MB called Rx Serial Message Buffer (Rx SMB).
2. If the CTRL2[EACEN] bit is negated, the RTR bit of Mailbox is never compared with the RTR bit of the incoming frame.
3. If the CAN_CTRL2[EACEN] bit is negated, the IDE bit of Mailbox is always compared with the IDE bit of the incoming frame.

43.4.6 Rx 14 Mask register (CANx_RX14MASK)

This register is located in RAM.

RX14MASK is provided for legacy application support. When the CAN_MCR[IRMQ] bit is asserted, RX14MASK has no effect.

RX14MASK is used to mask the filter fields of Message Buffer 14.

This register can only be programmed while the module is in Freeze mode as it is blocked by hardware in other modes.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RX14M																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CANx_RX14MASK field descriptions

Field	Description
RX14M	Rx Buffer 14 Mask Bits Each mask bit masks the corresponding Mailbox 14 filter field in the same way that RXMGMASK masks other Mailboxes' filters. See the description of the CAN_RXMGMASK register. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

43.4.7 Rx 15 Mask register (CANx_RX15MASK)

This register is located in RAM.

RX15MASK is provided for legacy application support. When the CAN_MCR[IRMQ] bit is asserted, RX15MASK has no effect.

RX15MASK is used to mask the filter fields of Message Buffer 15.

This register can be programmed only while the module is in Freeze mode because it is blocked by hardware in other modes.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RX15M																															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CANx_RX15MASK field descriptions

Field	Description
RX15M	Rx Buffer 15 Mask Bits Each mask bit masks the corresponding Mailbox 15 filter field in the same way that RXMGMASK masks other Mailboxes' filters. See the description of the CAN_RXMGMASK register. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

43.4.8 Error Counter (CANx_ECR)

This register has two 8-bit fields reflecting the value of the FlexCAN error counters:

- Transmit Error Counter (TXERRCNT field)
- Receive Error Counter (RXERRCNT field)

The Fault Confinement State (FLTCONF field in Error and Status Register 1 - CAN_ESR1) is updated based on TXERRCNT and RXERRCNT counters. TXERRCNT and RXERRCNT counters can be written in Freeze mode only. The rules for increasing and decreasing these counters are described in the CAN protocol and are completely implemented in the FlexCAN module.

The following are the basic rules for FlexCAN bus state transitions:

- If the value of TXERRCNT or RXERRCNT increases to be greater than or equal to 128, the FLTCONF field in the Error and Status Register is updated to reflect "Error Passive" state.
- If the FlexCAN state is "Error Passive", and either TXERRCNT or RXERRCNT decrements to a value less than or equal to 127 while the other already satisfies this condition, the FLTCONF field in the Error and Status Register is updated to reflect "Error Active" state.
- If the value of TXERRCNT increases to be greater than 255, the FLTCONF field in the Error and Status Register is updated to reflect "Bus Off" state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in "Bus Off' state, then TXERRCNT is cascaded together with another internal counter to count the 128th occurrences of 11 consecutive recessive bits on the bus. Hence, TXERRCNT is reset to zero and counts in a manner where the internal counter counts 11 such bits and then wraps around while incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, the FLTCONF field in the Error and Status Register is updated to be "Error Active" and both error counters are reset to zero. At any instance of dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value.
- If during system start-up, only one node is operating, then its TXERRCNT increases in each message it is trying to transmit, as a result of acknowledge errors (indicated by the ACKERR bit in the Error and Status Register). After the transition to "Error Passive" state, the TXERRCNT does not increment anymore by acknowledge errors. Therefore the device never goes to the "Bus Off" state.
- If the RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected while being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to resume to "Error Active" state.

NOTE

Refer to Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Address: Base address + 1Ch offset

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

CANx_ECR field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 RXERRCNT	Receive Error Counter Receive Error Counter for all errors detected in received messages. The RXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.
TXERRCNT	Transmit Error Counter Transmit Error Counter for all errors detected in transmitted messages. The TXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.

43.4.9 Error and Status 1 register (CANx_ESR1)

This register reports various error conditions detected in the reception and transmission of a CAN frame, some general status of the device and it is the source of some interrupts to the CPU.

The reported error conditions are BIT1ERR, BIT0ERR, ACKKERR, CRCERR, FRMERR and STFERR.

An error detected in a single CAN frame may be reported by one or more error flags. Also, error reporting is cummulative in case more error events happen in the next frames while the CPU does not attempt to read this register.

TXWRN, RXWRN, IDLE, TX, FLTCONF, RX and SYNCH are status bits.

BOFFINT, BOFFDONEINT, ERRINT, WAKINT, TWRNINT and RWRNINT are interrupt bits. It is recommended that CPU use the following procedure when servicing interrupt requests generated by these bits:

- Read this register to capture all error condition and status bits. This action clear the respective bits that were set since the last read access.
- Write 1 to clear the interrupt bit that has triggered the interrupt request.
- Write 1 to clear the ERR_OVR bit if it is set.

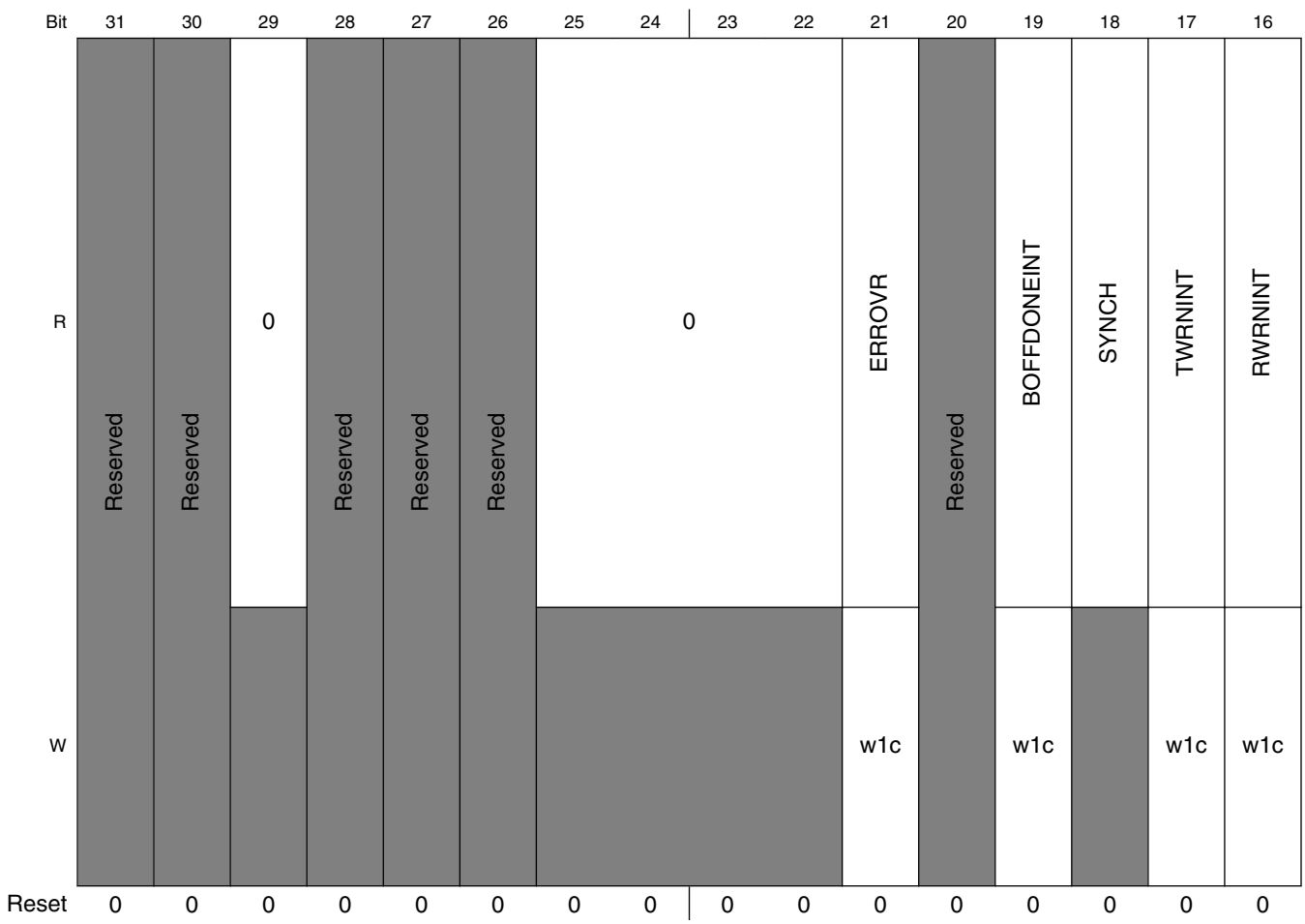
Starting from all error flags cleared, a first error event sets the ERRINT (provided the corresponding mask bit is asserted). If other error events in subsequent frames happen before the CPU to serve the interrupt request, the ERR_OVR bit is set to indicate that errors from different frames had accumulated.

SYNCH	IDLE	TX	RX	FlexCAN State
0	0	0	0	Not synchronized to CAN bus
1	1	x	x	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

Refer to Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Address: Base address + 20h offset



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	WAKINT
W														w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_ESR1 field descriptions

Field	Description
31 Reserved	This field is reserved.
30 Reserved	This field is reserved.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 Reserved	This field is reserved.
27 Reserved	This field is reserved.
26 Reserved	This field is reserved.
25–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 ERROVR	Error Overrun bit This bit indicates that an error condition occurred when any error flag is already set. This bit is cleared by writing it to 1.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	0 Overrun has not occurred. 1 Overrun has occurred.
20 Reserved	This field is reserved.
19 BOFFDONEINT	Bus Off Done Interrupt This bit is set when the Tx Error Counter (TXERRCNT) has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If the corresponding mask bit in the Control 2 Register (BOFFDONEMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0 No such occurrence. 1 FlexCAN module has completed Bus Off process.
18 SYNCH	CAN Synchronization Status This read-only flag indicates whether the FlexCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the FlexCAN. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not synchronized to the CAN bus. 1 FlexCAN is synchronized to the CAN bus.
17 TWRNINT	Tx Warning Interrupt Flag If the WRNEN bit in CAN_MCR is asserted, the TWRNINT bit is set when the TXWRN flag transitions from 0 to 1, meaning that the Tx error counter reached 96. If the corresponding mask bit in the Control 1 Register (CAN_CTRL1[TWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when the WRNEN is set again. Writing 0 has no effect. This flag is not generated during Bus Off state. This bit is not updated during Freeze mode. 0 No such occurrence. 1 The Tx error counter transitioned from less than 96 to greater than or equal to 96.
16 RWRNINT	Rx Warning Interrupt Flag If the WRNEN bit in CAN_MCR is asserted, the RWRNINT bit is set when the RXWRN flag transitions from 0 to 1, meaning that the Rx error counters reached 96. If the corresponding mask bit in the Control 1 Register (CAN_CTRL1[RWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when the WRNEN is set again. Writing 0 has no effect. This bit is not updated during Freeze mode. 0 No such occurrence. 1 The Rx error counter transitioned from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a message. **NOTE:** This bit is not set by a transmitter in case of arbitration field or ACK slot, or in case of a node sending a passive error flag that detects dominant bits. 0 No such occurrence. 1 At least one bit sent as recessive is received as dominant.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
14 BIT0ERR	Bit0 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a message. 0 No such occurrence. 1 At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error This bit indicates that an Acknowledge Error has been detected by the transmitter node, that is, a dominant bit has not been detected during the ACK SLOT. 0 No such occurrence. 1 An ACK error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error This bit indicates that a CRC Error has been detected by the receiver node, that is, the calculated CRC is different from the received. 0 No such occurrence. 1 A CRC error occurred since last read of this register.
11 FRMERR	Form Error This bit indicates that a Form Error has been detected by the receiver node, that is, a fixed-form bit field contains at least one illegal bit. 0 No such occurrence. 1 A Form Error occurred since last read of this register.
10 STFERR	Stuffing Error This bit indicates that a Stuffing Error has been detected by the receiver node. 0 No such occurrence. 1 A Stuffing Error occurred since last read of this register.
9 TXWRN	TX Error Warning This bit indicates when repetitive errors are occurring during message transmission and is affected by the value of TXERRCNT in CAN_ECR register only. This bit is not updated during Freeze mode. 0 No such occurrence. 1 TXERRCNT is greater than or equal to 96.
8 RXWRN	Rx Error Warning This bit indicates when repetitive errors are occurring during message reception and is affected by the value of RXERRCNT in CAN_ECR register only. This bit is not updated during Freeze mode. 0 No such occurrence. 1 RXERRCNT is greater than or equal to 96.
7 IDLE	This bit indicates when CAN bus is in IDLE state. See the table in the overall CAN_ESR1 register description. 0 No such occurrence. 1 CAN bus is now IDLE.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
6 TX	FlexCAN In Transmission This bit indicates if FlexCAN is transmitting a message. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not transmitting a message. 1 FlexCAN is transmitting a message.
5–4 FLTCONF	Fault Confinement State This 2-bit field indicates the Confinement State of the FlexCAN module. If the LOM bit in the Control Register 1 is asserted, after some delay that depends on the CAN bit timing the FLTCONF field will indicate “Error Passive”. Because of the very same delay, the way in which FLTCONF reflects an update to the CAN_ECR register by the CPU, also gets affected. This bit field is affected by soft reset, but if the LOM bit is asserted, its reset value lasts just one CAN bit. After this time, FLTCONF reports "Error Passive". 00 Error Active 01 Error Passive 1x Bus Off
3 RX	FlexCAN In Reception This bit indicates if FlexCAN is receiving a message. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not receiving a message. 1 FlexCAN is receiving a message.
2 BOFFINT	Bus Off Interrupt This bit is set when FlexCAN enters ‘Bus Off’ state. If the corresponding mask bit in the Control Register 1 (CAN_CTRL1[BOFFMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0 No such occurrence. 1 FlexCAN module entered Bus Off state.
1 ERRINT	Error Interrupt This bit indicates that at least one of the Error Bits (BIT1ERR, BIT0ERR, ACKERR. CRCERR, FRMERR or STFERR) is set. If the corresponding mask bit CAN_CTRL1[ERRMSK] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0 No such occurrence. 1 Indicates setting of any Error Bit in the Error and Status Register.
0 WAKINT	Wake-Up Interrupt This field applies when FlexCAN is in low-power mode under Self Wake Up mechanism: • Doze mode • Stop mode When a recessive-to-dominant transition is detected on the CAN bus and if the CAN_MCR[WAKMSK] bit is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When CAN_MCR[SLFWAK] is negated, this flag is masked. The CPU must clear this flag before disabling the bit. Otherwise it will be set when the SLFWAK is set again. Writing 0 has no effect.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	0 No such occurrence. 1 Indicates a recessive to dominant transition was received on the CAN bus.

43.4.10 Interrupt Masks 1 register (CANx_IMASK1)

This register allows any number of a range of the 16 Message Buffer Interrupts to be enabled or disabled for MB15 to MB0. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding CAN_IFLAG1 bit is set.

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																BUF15TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_IMASK1 field descriptions

Field	Description
31–16 Reserved	This field is reserved.
BUF15TO0M	Buffer MB $_i$ Mask Each bit enables or disables the corresponding FlexCAN Message Buffer Interrupt for MB15 to MB0. **NOTE:** Setting or clearing a bit in the CAN_IMASK1 Register can assert or negate an interrupt request, if the corresponding IFLAG1 bit is set. 0 The corresponding buffer Interrupt is disabled. 1 The corresponding buffer Interrupt is enabled.

43.4.11 Interrupt Flags 1 register (CANx_IFLAG1)

This register defines the flags for the 16 Message Buffer interrupts for MB15 to MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding CAN_IFLAG1 bit. If the corresponding CAN_IMASK1 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect. There is an exception when DMA for Rx FIFO is enabled, as described below.

The BUF7I to BUF5I flags are also used to represent FIFO interrupts when the Rx FIFO is enabled. When the bit CAN_MCR[RFEN] is set and the bit CAN_MCR[DMA] is negated, the function of the 8 least significant interrupt flags changes: BUF7I, BUF6I and BUF5I indicate operating conditions of the FIFO, BUF0I is used to empty FIFO, and BUF4I to BUF1I bits are reserved.

Before enabling the CAN_MCR[RFEN], the CPU must service the IFLAG bits asserted in the Rx FIFO region; see Section "Rx FIFO". Otherwise, these IFLAG bits will mistakenly show the related MBs now belonging to FIFO as having contents to be serviced. When the CAN_MCR[RFEN] bit is negated, the FIFO flags must be cleared. The same care must be taken when an CAN_CTRL2[RFFN] value is selected extending Rx FIFO filters beyond MB7. For example, when RFFN is 0x8, the MB0-23 range is occupied by Rx FIFO filters and related IFLAG bits must be cleared.

When both the CAN_MCR[RFEN] and CAN_MCR[DMA] bits are asserted (DMA feature for Rx FIFO enabled), the function of the 8 least significant interrupt flags (BUF7I - BUF0I) are changed to support the DMA operation. BUF7I and BUF6I are not used, as well as, BUF4I to BUF1I. BUF5I indicates operating condition of FIFO, and BUF0I is used to empty FIFO. Moreover, BUF5I does not generate a CPU interrupt, but generates a DMA request. IMASK1 bits in Rx FIFO region are not considered when bit CAN_MCR[DMA] is enabled. In addition the CPU must not clear the flag BUF5I when DMA is enabled. Before enabling the bit CAN_MCR[DMA], the CPU must service the IFLAGs asserted in the Rx FIFO region. When the bit CAN_MCR[DMA] is negated, the FIFO must be empty.

Before updating CAN_MCR[MAXMB] field, CPU must service the CAN_IFLAG1 bits whose MB value is greater than the CAN_MCR[MAXMB] to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Address: Base address + 30h offset

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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF15TO8I								BUF7I	BUF6I	BUF5I	BUF4TO1I				BUF0I
W	w1c								w1c	w1c	w1c	w1c				w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_IFLAG1 field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15–8 BUF15TO8I	Buffer MB_i Interrupt Each bit flags the corresponding FlexCAN Message Buffer interrupt for MB15 to MB8. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception. 1 The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt Or "Rx FIFO Overflow" When the RFEN bit in the CAN_MCR register is cleared (Rx FIFO disabled), this bit flags the interrupt for MB7. **NOTE:** This flag is cleared by the FlexCAN whenever the bit CAN_MCR[RFEN] is changed by CPU writes. The BUF7I flag represents "Rx FIFO Overflow" when CAN_MCR[RFEN] is set. In this case, the flag indicates that a message was lost because the Rx FIFO is full. Note that the flag will not be asserted when the Rx FIFO is full and the message was captured by a Mailbox. 0 No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO overflow when MCR[RFEN]=1 1 MB7 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO overflow when MCR[RFEN]=1
6 BUF6I	Buffer MB6 Interrupt Or "Rx FIFO Warning" When the RFEN bit in the CAN_MCR register is cleared (Rx FIFO disabled), this bit flags the interrupt for MB6. **NOTE:** This flag is cleared by the FlexCAN whenever the bit CAN_MCR[RFEN] is changed by CPU writes. The BUF6I flag represents "Rx FIFO Warning" when CAN_MCR[RFEN] is set. In this case, the flag indicates when the number of unread messages within the Rx FIFO is increased to 5 from 4 due to the reception of a new one, meaning that the Rx FIFO is almost full. Note that if the flag is cleared while the number of unread messages is greater than 4, it does not assert again until the number of unread messages within the Rx FIFO is decreased to be equal to or less than 4. 0 No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Rx FIFO almost full when MCR[RFEN]=1 1 MB6 completed transmission/reception when MCR[RFEN]=0, or Rx FIFO almost full when MCR[RFEN]=1

Table continues on the next page...

CANx_IFLAG1 field descriptions (continued)

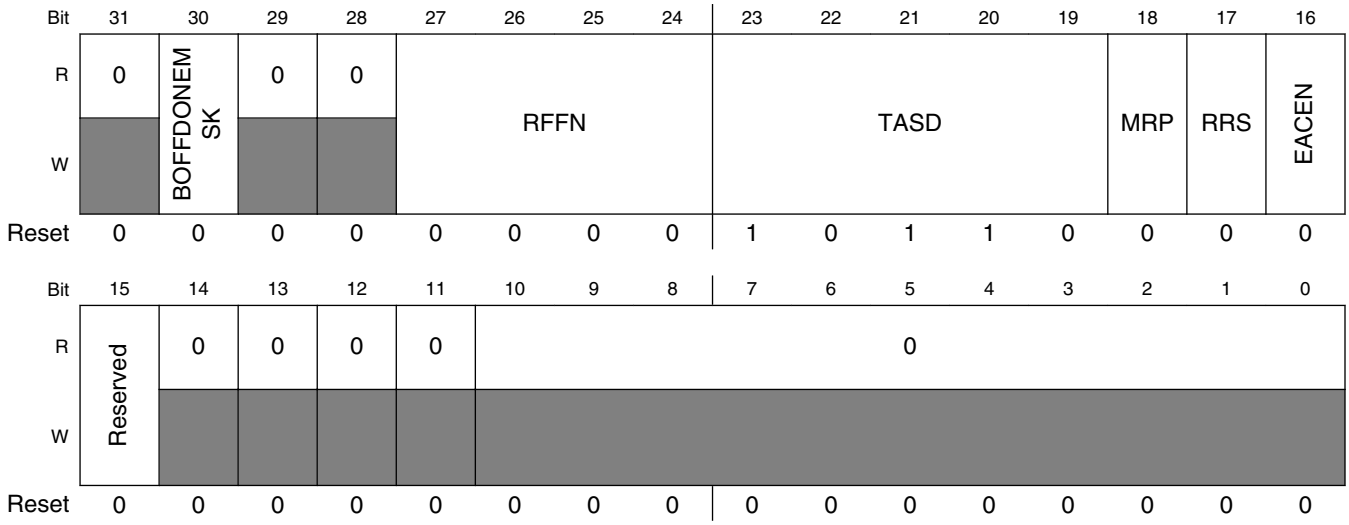
Field	Description
5 BUF5I	Buffer MB5 Interrupt Or "Frames available in Rx FIFO" When the RFEN bit in the MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB5. **NOTE:** This flag is cleared by the FlexCAN whenever the bit MCR[RFEN] is changed by CPU writes. When MCR[RFEN] is set (Rx FIFO enabled), the BUF5I flag represents "Frames available in Rx FIFO" and indicates that at least one frame is available to be read from the Rx FIFO. When the MCR[DMA] bit is enabled, this flag generates a DMA request and the CPU must not clear this bit by writing 1 in BUF5I. 0 No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the FIFO, when MCR[RFEN]=1 1 MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.
4–1 BUF4TO1I	Buffer MB $_i$ Interrupt Or "reserved" When the RFEN bit in the CAN_MCR register is cleared (Rx FIFO disabled), these bits flag the interrupts for MB4 to MB1. **NOTE:** These flags are cleared by the FlexCAN whenever the bit CAN_MCR[RFEN] is changed by CPU writes. The BUF4TO1I flags are reserved when CAN_MCR[RFEN] is set. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1 The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.
0 BUF0I	Buffer MB0 Interrupt Or Clear FIFO bit When the RFEN bit in MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB0. If the Rx FIFO is enabled, this bit is used to trigger the clear FIFO operation. This operation empties FIFO contents. Before performing this operation the CPU must service all FIFO related IFLAGs. When the bit MCR[DMA] is enabled this operation also clears the BUF5I flag and consequently abort the DMA request. The clear FIFO operation occurs when the CPU writes 1 in BUF0I. It is only allowed in Freeze Mode and is blocked by hardware in other conditions. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1 The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.

43.4.12 Control 2 register (CANx_CTRL2)

This register complements Control1 Register providing control bits for memory write access in Freeze Mode, for extending FIFO filter quantity, and for adjust the operation of internal FlexCAN processes like matching and arbitration.

The contents of this register are not affected by soft reset.

Address: Base address + 34h offset



CANx_CTRL2 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 BOFFDONEMSK	Bus Off Done Interrupt Mask This bit provides a mask for the Bus Off Done Interrupt in CAN_ESR1 register. 0 Bus Off Done interrupt disabled. 1 Bus Off Done interrupt enabled.
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–24 RFFN	Number Of Rx FIFO Filters This 4-bit field defines the number of Rx FIFO filters, as shown in the following table. The maximum selectable number of filters is determined by the chip. This field can only be written in Freeze mode as it is blocked by hardware in other modes. This field must not be programmed with values that make the number of Message Buffers occupied by Rx FIFO and ID Filter exceed the number of Mailboxes present, defined by CAN_MCR[MAXMB]. **NOTE:** Each group of eight filters occupies a memory space equivalent to two Message Buffers which means that the more filters are implemented the less Mailboxes will be available. Considering that the Rx FIFO occupies the memory space originally reserved for MB0-5, RFFN should be programmed with a value corresponding to a number of filters not greater than the number of available memory words which can be calculated as follows: (SETUP_MB - 6) × 4 where SETUP_MB is the least between the parameter NUMBER_OF_MB and CAN_MCR[MAXMB]. The number of remaining Mailboxes available will be: (SETUP_MB - 8) - (RFFN × 2)

Table continues on the next page...

CANx_CTRL2 field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>
If the Number of Rx FIFO Filters programmed through RFFN exceeds the SETUP_MB value (memory space available) the exceeding ones will not be functional.

NOTE:
• The number of the last remaining available mailboxes is defined by the least value between the NUMBER_OF_MB minus 1 and the CAN_MCR[MAXMB] field.
• If Rx Individual Mask Registers are not enabled then all Rx FIFO filters are affected by the Rx FIFO Global Mask.

<table>
<tr><th>RFFN[3:0]</th><th>Number of Rx FIFO filter elements</th><th>Message Buffers occupied by Rx FIFO and ID Filter Table</th><th>Remaining Available Mailboxes</th><th>Rx FIFO ID Filter Table Elements Affected by Rx Individual Masks</th><th>Rx FIFO ID Filter Table Elements Affected by Rx FIFO Global Mask</th></tr>
<tr><td>0x0</td><td>8</td><td>MB 0-7</td><td>MB 8-15</td><td>Elements 0-7</td><td>none</td></tr>
<tr><td>0x1</td><td>16</td><td>MB 0-9</td><td>MB 10-15</td><td>Elements 0-9</td><td>Elements 10-15</td></tr>
<tr><td>0x2</td><td>24</td><td>MB 0-11</td><td>MB 12-15</td><td>Elements 0-11</td><td>Elements 12-23</td></tr>
<tr><td>0x3</td><td>32</td><td>MB 0-13</td><td>MB 14-15</td><td>Elements 0-13</td><td>Elements 14-31</td></tr>
<tr><td>0x4</td><td>40</td><td>MB 0-15</td><td>none</td><td>Elements 0-15</td><td>Elements 16-39</td></tr>
</table>
</td></tr>
<tr><td>23–19
TASD</td><td>Tx Arbitration Start Delay

This 5-bit field indicates how many CAN bits the Tx arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See Tx Arbitration start delay for more details. This field can be written only in Freeze mode because it is blocked by hardware in other modes.</td></tr>
<tr><td>18
MRP</td><td>Mailboxes Reception Priority

If this bit is set the matching process starts from the Mailboxes and if no match occurs the matching continues on the Rx FIFO. This bit can be written only in Freeze mode because it is blocked by hardware in other modes.

0 Matching starts from Rx FIFO and continues on Mailboxes.
1 Matching starts from Mailboxes and continues on Rx FIFO.</td></tr>
<tr><td>17
RRS</td><td>Remote Request Storing

If this bit is asserted Remote Request Frame is submitted to a matching process and stored in the corresponding Message Buffer in the same fashion of a Data Frame. No automatic Remote Response Frame will be generated.

If this bit is negated the Remote Request Frame is submitted to a matching process and an automatic Remote Response Frame is generated if a Message Buffer with CODE=0b1010 is found with the same ID.

This bit can be written only in Freeze mode because it is blocked by hardware in other modes.

0 Remote Response Frame is generated.
1 Remote Request Frame is stored.</td></tr>
<tr><td>16
EACEN</td><td>Entire Frame Arbitration Field Comparison Enable For Rx Mailboxes

This bit controls the comparison of IDE and RTR bits within Rx Mailboxes filters with their corresponding bits in the incoming frame by the matching process. This bit does not affect matching for Rx FIFO. This bit can be written only in Freeze mode because it is blocked by hardware in other modes.</td></tr>
</table>

Table continues on the next page...

CANx_CTRL2 field descriptions (continued)

Field	Description
	0 Rx Mailbox filter's IDE bit is always compared and RTR is never compared despite mask bits. 1 Enables the comparison of both Rx Mailbox filter's IDE and RTR bit with their corresponding bits within the incoming frame. Mask bits do apply.
15 Reserved	This field is reserved. When writing to this field, always write the reset value.
14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.4.13 Error and Status 2 register (CANx_ESR2)

This register reports some general status information.

Address: Base address + 38h offset

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

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

CANx_ESR2 field descriptions

Field	Description
31–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22–16 LPTM	Lowest Priority Tx Mailbox If CAN_ESR2[VPS] is asserted, this field indicates the lowest number inactive Mailbox (see the CAN_ESR2[IMB] bit description). If there is no inactive Mailbox then the Mailbox indicated depends on CAN_CTRL1[LBUF] bit value. If CAN_CTRL1[LBUF] bit is negated then the Mailbox indicated is the one that has the greatest arbitration value (see the "Highest priority Mailbox first" section). If CAN_CTRL1[LBUF] bit is asserted then the Mailbox indicated is the highest number active Tx Mailbox. If a Tx Mailbox is being transmitted it is not considered in LPTM calculation. If CAN_ESR2[IMB] is not asserted and a frame is transmitted successfully, LPTM is updated with its Mailbox number.

Table continues on the next page...

CANx_ESR2 field descriptions (continued)

Field	Description
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 VPS	Valid Priority Status This bit indicates whether CAN_ESR2[IMB] and CAN_ESR2[LPTM] contents are currently valid or not. It is asserted upon every complete Tx arbitration process unless the CPU writes to Control and Status word of a Mailbox that has already been scanned, that is, it is behind Tx Arbitration Pointer, during the Tx arbitration process. If there is no inactive Mailbox and only one Tx Mailbox that is being transmitted then VPS is not asserted. This bit is negated upon the start of every Tx arbitration process or upon a write to Control and Status word of any Mailbox. **NOTE:** CAN_ESR2[VPS] is not affected by any CPU write into Control Status (C/S) of a MB that is blocked by abort mechanism. When CAN_MCR[AEN] is asserted, the abort code write in C/S of a MB that is being transmitted (pending abort), or any write attempt into a Tx MB with CAN_IFLAG set is blocked. 0 Contents of IMB and LPTM are invalid. 1 Contents of IMB and LPTM are valid.
13 IMB	Inactive Mailbox If ESR2[VPS] is asserted, this bit indicates whether there is any inactive Mailbox (CODE field is either 0b1000 or 0b0000). This bit is asserted in the following cases: • During arbitration, if an CAN_ESR2[LPTM] is found and it is inactive. • If CAN_ESR2[IMB] is not asserted and a frame is transmitted successfully. This bit is cleared in all start of arbitration (see Section "Arbitration process"). **NOTE:** CAN_ESR2[LPTM] mechanism have the following behavior: if an MB is successfully transmitted and CAN_ESR2[IMB]=0 (no inactive Mailbox), then CAN_ESR2[VPS] and CAN_ESR2[IMB] are asserted and the index related to the MB just transmitted is loaded into CAN_ESR2[LPTM]. 0 If ESR2[VPS] is asserted, the ESR2[LPTM] is not an inactive Mailbox. 1 If ESR2[VPS] is asserted, there is at least one inactive Mailbox. LPTM content is the number of the first one.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

43.4.14 CRC Register (CANx_CRCR)

This register provides information about the CRC of transmitted messages. This register is updated at the same time the Tx Interrupt Flag is asserted.

NOTE

Refer to CRC sequence calculation in the CAN Protocol standard (ISO 11898-1) for details.

Address: Base address + 44h offset

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

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

CANx_CRCR field descriptions

Field	Description
31–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22–16 MBCRC	CRC Mailbox This field indicates the number of the Mailbox corresponding to the value in CAN_CRCR[TXCRC] field.
15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXCRC	Transmitted CRC value This field indicates the CRC value of the last transmitted message.

43.4.15 Rx FIFO Global Mask register (CANx_RXFGMASK)

This register is located in RAM.

If Rx FIFO is enabled, RXFGMASK is used to mask the Rx FIFO ID Filter Table elements that do not have a corresponding RXIMR according to CAN_CTRL2[RFFN] field setting.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Address: Base address + 48h offset

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

* Notes:

- x = Undefined at reset.

CANx_RXFGMASK field descriptions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>FGM</td><td>Rx FIFO Global Mask Bits

These bits mask the ID Filter Table elements bits in a perfect alignment.

The following table shows how the FGM bits correspond to each IDAF field.

<table>
<tr><th rowspan="2">Rx FIFO ID Filter Table Elements Format (CAN_MCR[IDAM])</th><th colspan="6">Identifier Acceptance Filter Fields</th></tr>
<tr><th>RTR</th><th>IDE</th><th>RXIDA</th><th>RXIDB [1]</th><th>RXIDC [2]</th><th>Reserved</th></tr>
<tr><td>A</td><td>FGM[31]</td><td>FGM[30]</td><td>FGM[29:1]</td><td>-</td><td>-</td><td>FGM[0]</td></tr>
<tr><td>B</td><td>FGM[31], FGM[15]</td><td>FGM[30], FGM[14]</td><td rowspan="2">-</td><td>FGM[29:16], FGM[13:0]</td><td rowspan="2"></td><td rowspan="2">-</td></tr>
<tr><td>C</td><td>-</td><td>-</td><td>-</td><td>FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]</td></tr>
</table>

0 The corresponding bit in the filter is "don’t care."
1 The corresponding bit in the filter is checked.</td></tr>
</table>

1. If CAN_MCR[IDAM] field is equivalent to the format B only the fourteen most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.
2. If CAN_MCR[IDAM] field is equivalent to the format C only the eight most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.

43.4.16 Rx FIFO Information Register (CANx_RXFIR)

RXFIR provides information on Rx FIFO.

This register is the port through which the CPU accesses the output of the RXFIR FIFO located in RAM. The RXFIR FIFO is written by the FlexCAN whenever a new message is moved into the Rx FIFO as well as its output is updated whenever the output of the Rx FIFO is updated with the next message. See Section "Rx FIFO" for instructions on reading this register.

NOTE

RXFIR can be written only during memory initialization, due to the error code correction (ECC) feature. In every other case the register is read-only.

Address: Base address + 4Ch offset

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

* Notes:
- x = Undefined at reset.

CANx_RXFIR field descriptions

Field	Description
31–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
IDHIT	Identifier Acceptance Filter Hit Indicator This field indicates which Identifier Acceptance Filter was hit by the received message that is in the output of the Rx FIFO. If multiple filters match the incoming message ID then the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only while the CAN_IFLAG1[BUF5I] is asserted.

43.4.17 CAN Bit Timing Register (CANx_CBT)

This register is an alternative way to store the CAN bit timing variables described in CAN_CTRL1 register. EPRESDIV, EPROPSEG, EPSEG1, EPSEG2 and ERJW are extended versions of PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW bit fields respectively.

The BTF bit selects the use of the timing variables defined in this register.

The contents of this register are not affected by soft reset.

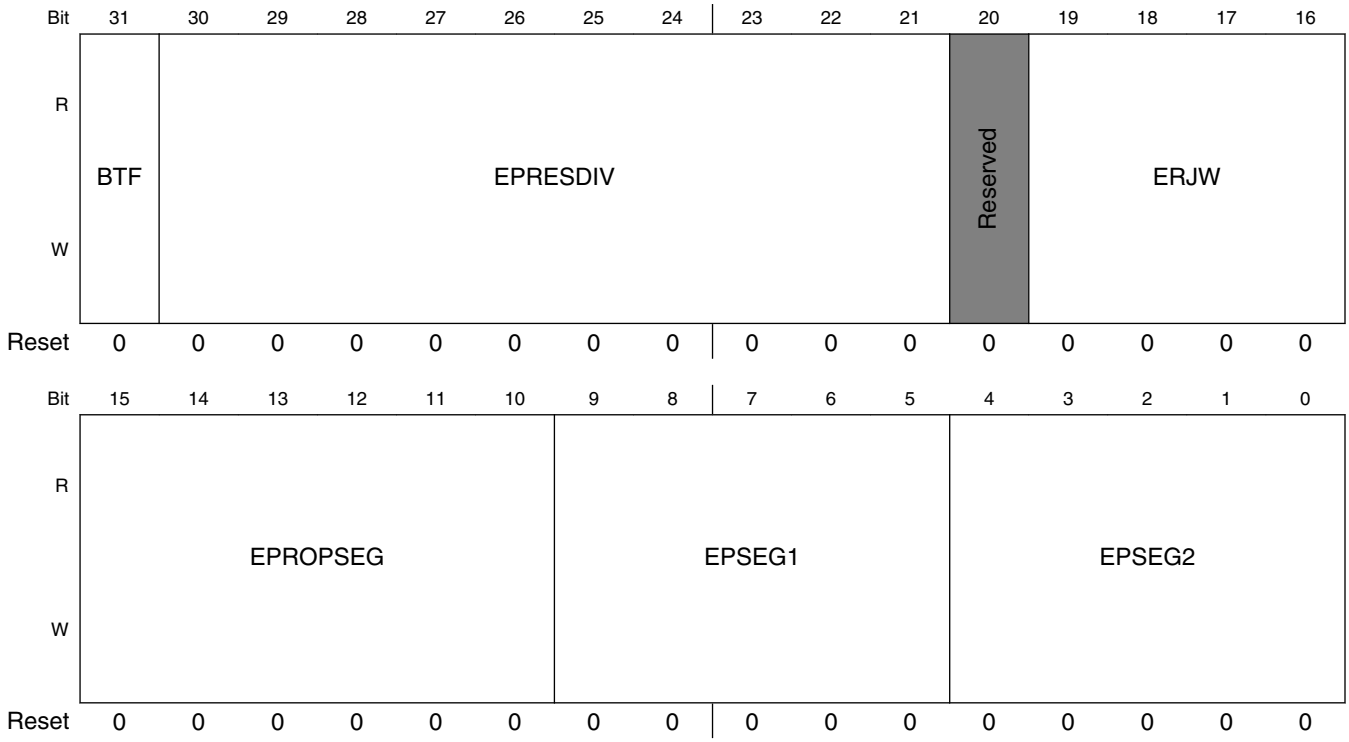
NOTE

The CAN bit variables in CAN_CTRL1 and in CAN_CBT are stored in the same register.

NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

Address: Base address + 50h offset



CANx_CBT field descriptions

Field	Description
31 BTF	Bit Timing Format Enable Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2 and ERJW replacing the CAN bit timing variables defined in CAN_CTRL1 register. This field can be written in Freeze mode only. 0 Extended bit time definitions disabled. 1 Extended bit time definitions enabled.
30–21 EPRESDIV	Extended Prescaler Division Factor This 10-bit field defines the ratio between the PE clock frequency and the Serial Clock (Sclock) frequency when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (EPRESDIV + 1)
20 Reserved	This field is reserved.
19–16 ERJW	Extended Resync Jump Width This 4-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[RJW] value range. One time quantum is equal to the Sclock period. This field can be written only in Freeze mode because it is blocked by hardware in other modes.

Table continues on the next page...

CANx_CBT field descriptions (continued)

Field	Description
	Resync Jump Width = ERJW + 1.
15–10 EPROPSEG	Extended Propagation Segment This 6-bit field defines the length of the Propagation Segment in the bit time when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PROPSEG] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time-Quanta. Time-Quantum = one Sclock period.
9–5 EPSEG1	Extended Phase Segment 1 This 5-bit field defines the length of Phase Segment 1 in the bit time when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PSEG1] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
EPSEG2	Extended Phase Segment 2 This 5-bit field defines the length of Phase Segment 2 in the bit time when CAN_CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CAN_CTRL1[PSEG2] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

43.4.18 Rx Individual Mask Registers (CANx_RXIMR*n*)

The RX Individual Mask Registers are used to store the acceptance masks for ID filtering in Rx MBs and the Rx FIFO.

When the Rx FIFO is disabled (CAN_MCR[RFEN] bit is negated), an individual mask is provided for each available Rx Mailbox on a one-to-one correspondence. When the Rx FIFO is enabled (CAN_MCR[RFEN] bit is asserted), an individual mask is provided for each Rx FIFO ID Filter Table Element on a one-to-one correspondence depending on the setting of CAN_CTRL2[RFFN] (see Rx FIFO).

CAN_RXIMR0 stores the individual mask associated to either MB0 or ID Filter Table Element 0, CAN_RXIMR1 stores the individual mask associated to either MB1 or ID Filter Table Element 1 and so on.

CAN_RXIMR registers can only be accessed by the CPU while the module is in Freeze mode, otherwise, they are blocked by hardware. These registers are not affected by reset. They are located in RAM and must be explicitly initialized prior to any reception.

It is possible for the RXIMR memory region to be accessed as general purpose memory. See Bus interface for more information.

Address: Base address + 880h offset + (4d × i), where i=0d to 15d

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

* Notes:
- x = Undefined at reset.

CANx_RXIMR*n* field descriptions

Field	Description
MI	Individual Mask Bits Each Individual Mask Bit masks the corresponding bit in both the Mailbox filter and Rx FIFO ID Filter Table element in distinct ways. For Mailbox filters, see the RXMGMASK register description. For Rx FIFO ID Filter Table elements, see the RXFGMASK register description. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

43.4.53 Message buffer structure

The message buffer structure used by the FlexCAN module is represented in the following figure. Both Extended (29-bit identifier) and Standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual MB is formed by 16 bytes.

The memory area from 0x80 to 0x17F is used by the mailboxes.

Table 43-3. Message buffer structure

	31 30 29 28	27 24	23	22	21	20	19 18 17 16	15 8	7 0
0x0		CODE		SRR	IDE	RTR	DLC	TIME STAMP	
0x4	PRIO	ID (Standard/Extended)					ID (Extended)		
0x8	Data Byte 0		Data Byte 1					Data Byte 2	Data Byte 3
0xC	Data Byte 4		Data Byte 5					Data Byte 6	Data Byte 7

= Unimplemented or Reserved

CODE - Message Buffer Code

This 4-bit field can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 43-4 and Table 43-5. See Functional description for additional information.

Table 43-4. Message buffer code for Rx buffers

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
0b0000: INACTIVE - MB is not active.	INACTIVE	-	-	-	MB does not participate in the matching process.
0b0100: EMPTY - MB is active and empty.	EMPTY	-	FULL	-	When a frame is received successfully (after the Move-in) process), the CODE field is automatically updated to FULL.
0b0010: FULL - MB is full.	FULL	Yes	FULL	-	The act of reading the C/S word followed by unlocking the MB (SRV) does not make the code return to EMPTY. It remains FULL. If a new frame is moved to the MB after the MB was serviced, the code still remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	-	If the MB is FULL and a new frame is moved to this MB before the CPU services it, the CODE field is automatically updated to OVERRUN. See Matching process for details about overrun behavior.
0b0110: OVERRUN - MB is being overwritten into a full buffer.	OVERRUN	Yes	FULL	-	If the CODE field indicates OVERRUN and CPU has serviced

Table continues on the next page...

Table 43-4. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
					the MB, when a new frame is moved to the MB then the code returns to FULL.
		No	OVERRUN	-	If the CODE field already indicates OVERRUN, and another new frame must be moved, the MB will be overwritten again, and the code will remain OVERRUN. See Matching process for details about overrun behavior.
0b1010: RANSWER[4] - A frame was configured to recognize a Remote Request Frame and transmit a Response Frame in return.	RANSWER	-	TANSWER(0b1110)	0	A Remote Answer was configured to recognize a remote request frame received. After that an MB is set to transmit a response frame. The code is automatically changed to TANSWER (0b1110). See Matching process for details. If CAN_CTRL2[RRS] is negated, transmit a response frame whenever a remote request frame with the same ID is received.
		-	-	1	This code is ignored during matching and arbitration process. See Matching process for details.
CODE[0]=1: BUSY - FlexCAN is updating the contents of the MB. The CPU must not access the MB.	BUSY[5]	-	FULL	-	Indicates that the MB is being updated. It will be negated automatically and does not interfere
		-	OVERRUN	-	

Table 43-4. Message buffer code for Rx buffers

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
					with the next CODE.

1. SRV: Serviced MB. MB was read and unlocked by reading TIMER or other MB.
2. A frame is considered a successful reception after the frame to be moved to MB (move-in process). See Move-in for details.
3. Remote Request Stored bit, see "Control 2 Register (CAN_CTRL2)" for details.
4. Code 0b1010 is not considered Tx and an MB with this code should not be aborted.
5. Note that for Tx MBs, the BUSY bit should be ignored upon read, except when AEN bit is set in the MCR register. If this bit is asserted, the corresponding MB does not participate in the matching process.

Table 43-5. Message buffer code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
0b1000: INACTIVE - MB is not active	INACTIVE	-	-	MB does not participate in arbitration process.
0b1001: ABORT - MB is aborted	ABORT	-	-	MB does not participate in arbitration process.
0b1100: DATA - MB is a Tx Data Frame (MB RTR must be 0)	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the MB automatically returns to the INACTIVE state.
0b1100: REMOTE - MB is a Tx Remote Request Frame (MB RTR must be 1)	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the MB automatically becomes an Rx Empty MB with the same ID.
0b1110: TANSWER - MB is a Tx Response Frame from an incoming Remote Request Frame	TANSWER	-	RANSWER	This is an intermediate code that is automatically written to the MB by the CHI as a result of a match to a remote request frame. The remote response frame will be transmitted unconditionally once, and then the code will automatically return to RANSWER (0b1010). The CPU can also write this code with the same effect. The remote response frame can be either a data frame or

Table 43-5. Message buffer code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
				another remote request frame depending on the RTR bit value. See Matching process and Arbitration process for details.

SRR - Substitute Remote Request

Fixed recessive bit, used only in extended format. It must be set to one by the user for transmission (Tx Buffers) and will be stored with the value received on the CAN bus for Rx receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as an arbitration loss.

1 = Recessive value is compulsory for transmission in extended format frames

0 = Dominant is not a valid value for transmission in extended format frames

IDE - ID Extended Bit

This field identifies whether the frame format is standard or extended.

1 = Frame format is extended

0 = Frame format is standard

RTR - Remote Transmission Request

This bit affects the behavior of remote frames and is part of the reception filter. See Table 43-4, Table 43-5, and the description of the RRS bit in Control 2 Register (CAN_CTRL2) for additional details.

If FlexCAN transmits this bit as '1' (recessive) and receives it as '0' (dominant), it is interpreted as an arbitration loss. If this bit is transmitted as '0' (dominant), then if it is received as '1' (recessive), the FlexCAN module treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission.

1 = Indicates the current MB may have a remote request frame to be transmitted if MB is Tx. If the MB is Rx then incoming remote request frames may be stored.

0 = Indicates the current MB has a data frame to be transmitted. In Rx MB it may be considered in matching processes.

DLC - Length of Data in Bytes

This 4-bit field is the length (in bytes) of the Rx or Tx data, which is located in offset 0x8 through 0xF of the MB space (see Table 43-3) . In reception, this field is written by the FlexCAN module, copied from the DLC (Data Length Code) field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 43-6).

TIME STAMP - Free-Running Counter Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured for Tx and Rx frames at the time when the beginning of the Identifier field appears on the CAN bus.

PRIO - Local priority

This 3-bit field is used only when LPRIO_EN bit is set in CAN_MCR, and it only makes sense for Tx mailboxes. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.

ID - Frame Identifier

In standard frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.

DATA BYTE 0 to 7 - Data Field

Up to eight bytes can be used for a data frame.

For Rx frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC as shown in the table below.

Table 43-6. DATA BYTEs validity

DLC	Valid DATA BYTEs
0	none
1	DATA BYTE 0
2	DATA BYTE 0 to 1
3	DATA BYTE 0 to 2
4	DATA BYTE 0 to 3
5	DATA BYTE 0 to 4
6	DATA BYTE 0 to 5
7	DATA BYTE 0 to 6
8 or above	DATA BYTE 0 to 7

43.4.54 Rx FIFO structure

When the CAN_MCR[RFEN] bit is set, the memory area from 0x80 to 0xDC (which is normally occupied by MBs 0–5) is used by the reception FIFO engine.

The region 0x80-0x8C contains the output of the FIFO which must be read by the CPU as a message buffer. This output contains the oldest message that has been received but not yet read. The region 0x90-0xDC is reserved for internal use of the FIFO engine.

An additional memory area, which starts at 0xE0 and may extend up to 0x17C (normally occupied by MBs 6–15) depending on the CAN_CTRL2[RFFN] field setting, contains the ID filter table (configurable from 8 to 40 table elements) that specifies filtering criteria for accepting frames into the FIFO.

Out of reset, the ID filter table flexible memory area defaults to 0xE0 and extends only to 0xFC, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Rx FIFO data structure.

Table 43-7. Rx FIFO structure

	31 28 24	23 22	21	20	19 18 17 16	15 8	7 0
0x80	IDHIT	SRR	IDE	RTR	DLC	TIME STAMP	
0x84		ID standard			ID extended		
0x88	Data byte 0	Data byte 1				Data byte 2	Data byte 3
0x8C	Data byte 4	Data byte 5				Data byte 6	Data byte 7
0x90 to 0xDC	Reserved						
0xE0	ID filter table element 0						
0xE4	ID filter table element 1						
0xE8 to 0x2D4	ID filter table elements 2 to 125						
0x2D8	ID filter table element 126						
0x2DC	ID filter table element 127						
	= Unimplemented or reserved						

Each ID filter table element occupies an entire 32-bit word and can be compounded by one, two, or four Identifier Acceptance Filters (IDAF) depending on the CAN_MCR[IDAM] field setting. The following figures show the IDAF indexation.

The following table shows the three different formats of the ID table elements. Note that all elements of the table must have the same format. See Rx FIFO for more information.

Table 43-8. ID table structure

Format	31	30	29 … 24	23 … 16	15	14	13 … 8	7 … 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 29–16)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std/ext = 31–24)			RXIDC_1 (std/ext = 23–16)	RXIDC_2 (std/ext = 15–8)			RXIDC_3 (std/ext = 7–0)	
	= Unimplemented or Reserved								

RTR — Remote Frame

This bit specifies if Remote Frames are accepted into the FIFO if they match the target ID.

1 = Remote Frames can be accepted and data frames are rejected

0 = Remote Frames are rejected and data frames can be accepted

IDE — Extended Frame

Specifies whether extended or standard frames are accepted into the FIFO if they match the target ID.

1 = Extended frames can be accepted and standard frames are rejected

0 = Extended frames are rejected and standard frames can be accepted

RXIDA — Rx Frame Identifier (Format A)

Specifies an ID to be used as acceptance criteria for the FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.

RXIDB_0, RXIDB_1 — Rx Frame Identifier (Format B)

Specifies an ID to be used as acceptance criteria for the FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.

RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3 — Rx Frame Identifier (Format C)

Specifies an ID to be used as acceptance criteria for the FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.

IDHIT — Identifier Acceptance Filter Hit Indicator

This 9-bit field indicates which Identifier Acceptance Filter was hit by the received message that is in the output of the Rx FIFO. See Rx FIFO for more information.

43.5 Functional description

The FlexCAN module is a CAN protocol engine with a very flexible mailbox system for transmitting and receiving CAN frames. The mailbox system is composed by a set of Message Buffers (MB) that store configuration and control data, time stamp, message ID and data (see Message buffer structure). The memory corresponding to the first 38 MBs can be configured to support a FIFO reception scheme with a powerful ID filtering mechanism, capable of checking incoming frames against a table of IDs (up to 128 extended IDs or 256 standard IDs or 512 8-bit ID slices), with individual mask register for up to 32 ID Filter Table elements.

Simultaneous reception through FIFO and mailbox is supported. For mailbox reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed on its ID field. A masking scheme makes it possible to match the ID programmed on the MB with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of MBs to be transmitted based on the message ID (optionally augmented by 3 local priority bits) or the MB ordering.

Before proceeding with the functional description, an important concept must be explained. A Message Buffer is said to be "active" at a given time if it can participate in both the Matching and Arbitration processes. An Rx MB with a 0b0000 code is inactive (refer to Table 43-4). Similarly, a Tx MB with a 0b1000 or 0b1001 code is also inactive (refer to Table 43-5).

43.5.1 Transmit process

To transmit a CAN frame, the CPU must prepare a Message Buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt bit is set and clear it.

2. If the MB is active (transmission pending), write the ABORT code (0b1001) to the CODE field of the Control and Status word to request an abort of the transmission.
3. Wait for the corresponding IFLAG bit to be asserted by polling the CAN_IFLAG register, or by the interrupt request if enabled by the respective IMASK bit.
4. Read back the CODE field to check if the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIO_EN]).
7. Write payload data bytes.
8. Configure the Control and Status word with the desired configuration.
   a. Set ID type via MB_CS[IDE].
   b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
   c. Set Data Length Code in bytes via MB_CS[DLC]. See Table 43-6 for detailed information.
   d. Activate the message buffer to transmit the CAN frame by setting MB_CS[CODE] to 0xC.

NOTE

It is strongly recommended that all the fields in MB_CS word be configured in only one 32-bit write operation to maximize software performance. If the fields are configured in separate writes, the MB_CS[CODE] must be the last write in the C/S word.

When the MB is activated, it participates in the arbitration process and is eventually transmitted according to its priority.

At the end of the successful transmission:

- The value of the Free Running Timer is written into the Time Stamp field.
- The CODE field in the Control and Status word is updated.
- A status flag is set in the Interrupt Flag register.
- An interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The new CODE field after transmission depends on the code that was used to activate the MB (see Table 43-4 and Table 43-5 in Message buffer structure).

When the Abort feature is enabled (CAN_MCR[AEN] is asserted), after the Interrupt Flag is asserted for a Mailbox configured as transmit buffer, the Mailbox is blocked. Therefore the CPU is not able to update it until the Interrupt Flag is negated by the CPU. This means that the CPU must clear the corresponding IFLAG bit before starting to prepare this MB for a new transmission or reception.

NOTE

If backwards compatibility is desired (CAN_MCR[AEN] bit is negated), write the INACTIVE code (0b1000) to the CODE field to inactivate the MB. However, in this case the pending frame may be transmitted without notification (see Mailbox inactivation).

43.5.2 Arbitration process

The arbitration process scans the Mailboxes searching the Tx one that holds the message to be sent in the next opportunity. This Mailbox is called the *arbitration winner*.

The scan starts from the lowest number Mailbox and runs toward the higher ones.

The arbitration process is triggered in the following events:

- From the CRC field of the CAN frame. The start point depends on the CAN_CTRL2[TASD] field value.
- During the Error Delimiter field of a CAN frame.
- During the Overload Delimiter field of a CAN frame.
- When the winner is inactivated and the CAN bus has still not reached the first bit of the Intermission field.
- When there is CPU write to the C/S word of a winner MB and the CAN bus has still not reached the first bit of the Intermission field.
- When CHI is in Idle state and the CPU writes to the C/S word of any MB.
- When FlexCAN exits Bus Off state.
- Upon leaving Freeze mode or Low Power mode.

If the arbitration process does not manage to evaluate all Mailboxes before the CAN bus has reached the first bit of the Intermission field the temporary arbitration winner is invalidated and the FlexCAN will not compete for the CAN bus in the next opportunity.

The arbitration process selects the winner among the active Tx Mailboxes at the end of the scan according to both CAN_CTRL1[LBUF] and CAN_MCR[LPRIOEN] bits settings.

43.5.2.1 Lowest-number Mailbox first

If CAN_CTRL1[LBUF] bit is asserted the first (lowest number) active Tx Mailbox found is the arbitration winner. CAN_MCR[LPRIOEN] bit has no effect when CAN_CTRL1[LBUF] is asserted.

43.5.2.2 Highest-priority Mailbox first

If CAN_CTRL1[LBUF] bit is negated, then the arbitration process searches the active Tx Mailbox with the highest priority, which means that this Mailbox's frame would have a higher probability to win the arbitration on CAN bus when multiple external nodes compete for the bus at the same time.

The sequence of bits considered for this arbitration is called the *arbitration value* of the Mailbox. The highest-priority Tx Mailbox is the one that has the lowest arbitration value among all Tx Mailboxes.

If two or more Mailboxes have equivalent arbitration values, the Mailbox with the lowest number is the arbitration winner.

The composition of the arbitration value depends on CAN_MCR[LPRIOEN] bit setting.

43.5.2.2.1 Local Priority disabled

If CAN_MCR[LPRIOEN] bit is negated the arbitration value is built in the exact sequence of bits as they would be transmitted in a CAN frame (see the following table) in such a way that the Local Priority is disabled.

Table 43-9. Composition of the arbitration value when Local Priority is disabled

Format	Mailbox Arbitration Value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

43.5.2.2.2 Local Priority enabled

If Local Priority is desired CAN_MCR[LPRIOEN] must be asserted. In this case the Mailbox PRIO field is included at the very left of the arbitration value (see the following table).

Table 43-10. Composition of the arbitration value when Local Priority is enabled

Format	Mailbox Arbitration Value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

As the PRIO field is the most significant part of the arbitration value Mailboxes with low PRIO values have higher priority than Mailboxes with high PRIO values regardless the rest of their arbitration values.

Note that the PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

43.5.2.3 Arbitration process (continued)

After the arbitration winner is found, its content is copied to a hidden auxiliary MB called Tx Serial Message Buffer (Tx SMB), which has the same structure as a normal MB but is not user accessible. This operation is called move-out and after it is done, write access to the C/S word of the corresponding MB is blocked (if the AEN bit in CAN_MCR register is asserted). Write access is restored in the following events:

- After the MB is transmitted and the corresponding IFLAG bit is cleared by the CPU
- FlexCAN enters in Freeze mode or Bus Off
- FlexCAN loses the bus arbitration or there is an error during the transmission

At the first opportunity window on the CAN bus, the message on the Tx SMB is transmitted according to the CAN protocol rules.

Arbitration process can be triggered in the following situations:

- During Rx and Tx frames from CAN CRC field to end of frame. CAN_CTRL2[TASD] bit value may be changed to optimize the arbitration start point.
- During CAN BusOff state from TX_ERR_CNT=124 to 128. CAN_CTRL2[TASD] bit value may be changed to optimize the arbitration start point.
- During C/S write by CPU in BusIdle. First C/S write starts arbitration process and a second C/S write during this same arbitration restarts the process. If other C/S writes are performed, Tx arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and BusIdle state starts then an arbitration process is triggered. In this case the first and second C/S write in

BusIdle will not restart the arbitration process. It is possible that there is not enough time to finish arbitration in WaitForBusIdle state and the next state is Idle. In this case the scan is not interrupted, and it is completed during BusIdle state. During this arbitration C/S write does not cause arbitration restart.
- Arbitration winner deactivation during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the WaitForBusIdle state). If there is a re-synchronization during WaitForBusIdle, the arbitration process is restarted.

Arbitration process stops in the following situations:

- All Mailboxes were scanned
- A Tx active Mailbox is found in case of Lowest Buffer feature enabled
- Arbitration winner inactivation or abort during any arbitration process
- There was not enough time to finish Tx arbitration process (for instance, when a deactivation was performed near the end of frame). In this case arbitration process is pending.
- Error or Overload flag in the bus
- Low Power or Freeze mode request in Idle state

Arbitration is considered pending as described below:

- It was not possible to finish arbitration process in time
- C/S write during arbitration if write is performed in a MB whose number is lower than the Tx arbitration pointer
- Any C/S write if there is no Tx Arbitration process in progress
- Rx Match has just updated a Rx Code to Tx Code
- Entering Busoff state

C/S write during arbitration has the following effect:

- If C/S write is performed in the arbitration winner, a new process is restarted immediately.
- If C/S write is performed in a MB whose number is higher than the Tx arbitration pointer, the ongoing arbitration process will scan this MB as normal.

43.5.3 Receive process

To be able to receive CAN frames into a Mailbox, the CPU must prepare it for reception by executing the following steps:

1. If the Mailbox is active (either Tx or Rx) inactivate the Mailbox (see Mailbox inactivation), preferably with a safe inactivation (see Transmission abort mechanism).

2. Write the ID word
3. Write the EMPTY code (0b0100) to the CODE field of the Control and Status word to activate the Mailbox.

After the MB is activated, it will be able to receive frames that match the programmed filter. At the end of a successful reception, the Mailbox is updated by the *move-in* process (see Move-in) as follows:

1. The received Data field (8 bytes at most for Classical CAN message format) is stored.
2. The received Identifier field is stored.
3. The value of the Free Running Timer at the time of the second bit of frame's Identifier field is written into the Mailbox's Time Stamp field.
4. The received SRR, IDE, RTR and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 43-4 and Table 43-5 in Section Message buffer structure).
6. A status flag is set in the Interrupt Flag Register and an interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The recommended way for CPU servicing (read) the frame received in an Mailbox is using the following procedure:

1. Read the Control and Status word of that Mailbox.
2. Check if the BUSY bit is deasserted, indicating that the Mailbox is locked. Repeat step 1) while it is asserted. See Mailbox lock mechanism.
3. Read the contents of the Mailbox. Once Mailbox is locked now, its contents won't be modified by FlexCAN Move-in processes. See Move-in.
4. Acknowledge the proper flag at IFLAG registers.
5. Read the free running timer to unlock the mailbox.

The CPU should poll for frame reception by the status flag bit for the specific Mailbox in one of the IFLAG Registers and not by the CODE field of that Mailbox. Polling the CODE field does not work because once a frame was received and the CPU services the Mailbox (by reading the C/S word followed by unlocking the Mailbox), the CODE field will not return to EMPTY. It will remain FULL, as explained in Table 43-4 . If the CPU

tries to workaround this behavior by writing to the C/S word to force an EMPTY code after reading the Mailbox without a prior *safe inactivation*, a newly received frame matching the filter of that Mailbox may be lost.

CAUTION

In summary: *never do polling by reading directly the C/S word of the Mailboxes. Instead, read the IFLAG registers.*

Note that the received frame's Identifier field is always stored in the matching Mailbox, thus the contents of the ID field in an Mailbox may change if the match was due to masking. When CAN_MCR[SRXDIS] bit is asserted, FlexCAN will not store frames transmitted by itself in any MB, even if it contains a matching Rx Mailbox, and no interrupt flag or interrupt signal will be generated. Otherwise, when CAN_MCR[SRXDIS] bit is deasserted, FlexCAN can receive frames transmitted by itself if there exists a matching Rx Mailbox.

To be able to receive CAN frames through the Rx FIFO, the CPU must enable and configure the Rx FIFO during Freeze mode (see Rx FIFO). Upon receiving the Frames Available in Rx FIFO interrupt (see the description of the BUF5I bit "Frames available in Rx FIFO" bit in the CAN_IFLAG1 register), the CPU should service the received frame using the following procedure:

1. Read the Control and Status word (optional: needed only if a mask was used for IDE and RTR bits)
2. Read the ID field (optional: needed only if a mask was used)
3. Read the Data field
4. Read the CAN_RXFIR register (optional)
5. Clear the Frames Available in Rx FIFO interrupt by writing 1 to CAN_IFLAG1[BUF5I] bit (mandatory: releases the MB and allows the CPU to read the next Rx FIFO entry)

When CAN_MCR[DMA] is asserted, upon receiving a frame in FIFO, CAN_IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Rx FIFO under DMA Operation). The CAN_IMASK1 bits in Rx FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 0x80 address, optional)
2. Read the ID field (read 0x84 address, optional)

3. Read all Data Bytes (start read at 0x88 address, optional)
4. Read the last Data Bytes (read 0x8C address is mandatory)

43.5.4 Matching process

The matching process scans the MB memory looking for Rx MBs programmed with the same ID as the one received from the CAN bus. If the FIFO is enabled, the priority of scanning can be selected between Mailboxes and FIFO filters. The matching starts from the lowest number Message Buffer toward the higher ones. If no match is found within the first structure then the other is scanned subsequently. In the event that the FIFO is full, the matching algorithm always looks for a matching MB outside the FIFO region.

As the frame is being received, it is stored in a hidden auxiliary MB called Rx Serial Message Buffer (Rx SMB).

The matching process start point depends on the following conditions:

- If the received frame is a remote frame, the start point is the CRC field of the frame
- If the received frame is a data frame with DLC field equal to zero, the start point is the CRC field of the frame
- If the received frame is a data frame with DLC field different than zero, the start point is the DATA field of the frame

If a matching ID is found in the FIFO table or in one of the Mailboxes, the contents of the Rx SMB are transferred to the FIFO or to the matched Mailbox by the move-in process. If any CAN protocol error is detected then no match results are transferred to the FIFO or to the matched Mailbox at the end of reception.

The matching process scans all matching elements of both Rx FIFO (if enabled) and the active Rx Mailboxes (CODE is EMPTY, FULL, OVERRUN or RANSWER) in search of a successful comparison with the matching elements of the Rx SMB that is receiving the frame on the CAN bus. The Rx SMB has the same structure of a Mailbox. The reception structures (Rx FIFO or Mailboxes) associated with the matching elements that had a successful comparison are the *matched structures*. The *matching winner* is selected at the end of the scan among those matched structures and depends on conditions described ahead. See the following table.

Table 43-11. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Mailbox	0	-	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY or FULL or OVERRUN
Mailbox	0	-	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	0	-	cmp	no_cmp	cmp	RANSWER
Mailbox	1	1	0	cmp	no_cmp	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
FIFO[5]	-	-	-	cmp_msk	cmp_msk	cmp_msk	-

1. For Mailbox structure, If SMB[IDE] is asserted, the ID is 29 bits (ID Standard + ID Extended). If SMB[IDE] is negated, the ID is only 11 bits (ID Standard). For FIFO structure, the ID depends on IDAM.
2. cmp: Compares the Rx SMB contents with the MB contents regardless the masks.
3. no_cmp: The Rx SMB contents are not compared with the MB contents.
4. cmp_msk: Compares the Rx SMB contents with MB contents taking into account the masks.
5. SMB[IDE] and SMB[RTR] are not taken into account when IDAM is type C.

A reception structure is *free-to-receive* when any of the following conditions is satisfied:

- The CODE field of the Mailbox is EMPTY
- The CODE field of the Mailbox is either FULL or OVERRUN and it has already been serviced (the C/S word was read by the CPU and unlocked as described in Mailbox lock mechanism)
- The CODE field of the Mailbox is either FULL or OVERRUN and an inactivation (see Mailbox inactivation) is performed
- The Rx FIFO is not full

The scan order for Mailboxes and Rx FIFO is from the matching element with lowest number to the higher ones.

The matching winner search for Mailboxes is affected by the CAN_MCR[IRMQ] bit. If it is negated, the matching winner is the first matched Mailbox regardless if it is free-to-receive or not. If it is asserted, the matching winner is selected according to the priority below:

1. the first free-to-receive matched Mailbox;
2. the last non free-to-receive matched Mailbox.

It is possible to select the priority of scan between Mailboxes and Rx FIFO by the CAN_CTRL2[MRP] bit.

If the selected priority is Rx FIFO first:

- If the Rx FIFO is a matched structure and is free-to-receive, then the Rx FIFO is the matching winner regardless of the scan for Mailboxes
- Otherwise (the Rx FIFO is not a matched structure or is not free-to-receive), then the matching winner is searched among Mailboxes as described above

If the selected priority is Mailboxes first:

- If a free-to-receive matched Mailbox is found, it is the matching winner regardless of the scan for Rx FIFO
- If no matched Mailbox is found, then the matching winner is searched in the scan for the Rx FIFO
- If both conditions above are not satisfied and a non free-to-receive matched Mailbox is found, then the matching winner determination is conditioned by the CAN_MCR[IRMQ] bit:
- - If CAN_MCR[IRMQ] bit is negated, the matching winner is the first matched Mailbox
  - If CAN_MCR[IRMQ] bit is asserted, the matching winner is the Rx FIFO if it is a free-to-receive matched structure; otherwise, the matching winner is the last non free-to-receive matched Mailbox

See the following table for a summary of matching possibilities.

Table 43-12. Matching possibilities and resulting reception structures

RFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
No FIFO, only MB, match is always MB first						
0	0	X[1]	None[2]	-[3]	None	Frame lost by no match
0	0	X	Free[4]	-	FirstMB	
0	1	X	None	-	None	Frame lost by no match
0	1	X	Free	-	FirstMb	
0	1	X	NotFree	-	LastMB	Overrun
FIFO enabled, no match in FIFO is as if FIFO does not exist						
1	0	X	None	None[5]	None	Frame lost by no match
1	0	X	Free	None	FirstMB	
1	1	X	None	None	None	Frame lost by no match
1	1	X	Free	None	FirstMb	
1	1	X	NotFree	None	LastMB	Overrun
FIFO enabled, Queue disabled						

Table continues on the next page...

Table 43-12. Matching possibilities and resulting reception structures (continued)

RFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception structure	Description
1	0	0	X	NotFull[6]	FIFO	
1	0	0	None	Full[7]	None	Frame lost by FIFO full (FIFO Overflow)
1	0	0	Free	Full	FirstMB	
1	0	0	NotFree	Full	FirstMB	
1	0	1	None	NotFull	FIFO	
1	0	1	None	Full	None	Frame lost by FIFO full (FIFO Overflow)
1	0	1	Free	X	FirstMB	
1	0	1	NotFree	X	FirtsMb	Overrun
FIFO enabled, Queue enabled						
1	1	0	X	NotFull	FIFO	
1	1	0	None	Full	None	Frame lost by FIFO full (FIFO Overflow)
1	1	0	Free	Full	FirstMB	
1	1	0	NotFree	Full	LastMb	Overrun
1	1	1	None	NotFull	FIFO	
1	1	1	Free	X	FirstMB	
1	1	1	NotFree	NotFull	FIFO	
1	1	1	NotFree	Full	LastMb	Overrun

1. This is a don't care condition.
2. Matched in MB "None" means that the frame has not matched any MB (free-to-receive or non-free-to-receive).
3. This is a forbidden condition.
4. Matched in MB "Free" means that the frame matched at least one MB free-to-receive regardless of whether it has matched MBs non-free-to-receive.
5. Matched in FIFO "None" means that the frame has not matched any filter in FIFO. It is as if the FIFO didn't exist (CAN_CTRL2[RFEN]=0).
6. Matched in FIFO "NotFull" means that the frame has matched a FIFO filter and has empty slots to receive it.
7. Matched in FIFO "Full" means that the frame has matched a FIFO filter but couldn't store it because it has no empty slots to receive it.

If a non-safe Mailbox inactivation (see Mailbox inactivation) occurs during matching process and the Mailbox inactivated is the temporary matching winner, then the temporary matching winner is invalidated. The matching elements scan is not stopped nor restarted, it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist, therefore a message may be lost.

Suppose, for example, that the FIFO is disabled, IRMQ is enabled and there are two MBs with the same ID, and FlexCAN starts receiving messages with that ID. Let us say that these MBs are the second and the fifth in the array. When the first message arrives, the

matching algorithm finds the first match in MB number 2. The code of this MB is EMPTY, so the message is stored there. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive", so it keeps looking, finds MB number 5 and stores the message there. If yet another message with the same ID arrives, the matching algorithm finds out that there are no matching MBs that are "free-to-receive", so it decides to overwrite the last matched MB, which is number 5. In doing so, it sets the CODE field of the MB to indicate OVERRUN.

The ability to match the same ID in more than one MB can be exploited to implement a reception queue (in addition to the full featured FIFO) to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the MBs. The CPU can examine the Time Stamp field of the MBs to determine the order in which the messages arrived.

Matching to a range of IDs is possible by using ID Acceptance Masks. FlexCAN supports individual masking per MB. See the description of the Rx Individual Mask Registers (CAN_RXIMRx). During the matching algorithm, if a mask bit is asserted, then the corresponding ID bit is compared. If the mask bit is negated, the corresponding ID bit is a "don't care". Note that the Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed while the module is in Freeze mode; otherwise, they are blocked by hardware.

FlexCAN also supports an alternate masking scheme with only four mask registers (CAN_RXFGMASK, CAN_RXMGMASK, CAN_RX14MASK and CAN_RX15MASK) for backwards compatibility with legacy applications. This alternate masking scheme is enabled when the IRMQ bit in the CAN_MCR Register is negated.

43.5.5 Move process

There are two types of move process: move-in and move-out.

43.5.5.1 Move-in

The move-in process is the copy of a message received by an Rx SMB to a Rx Mailbox or FIFO that has matched it. If the move destination is the Rx FIFO, attributes of the message are also copied to the CAN_RXFIR FIFO. Each Rx SMB has its own move-in process, but only one is performed at a given time as described ahead. The move-in starts only when the message held by the Rx SMB has a corresponding matching winner (see Matching process) and all of the following conditions are true:

- The CAN bus has reached or let past either:

    - The second bit of Intermission field next to the frame that carried the message that is in the Rx SMB
    - The first bit of an overload frame next to the frame that carried the message that is in the Rx SMB
- There is no ongoing matching process
- The destination Mailbox is not locked by the CPU
- There is no ongoing move-in process from another Rx SMB. If more than one move-in processes are to be started at the same time both are performed and the newest substitutes the oldest.

The term *pending move-in* is used throughout the documentation and stands for a move-to-be that still does not satisfy all of the aforementioned conditions.

The move-in is cancelled and the Rx SMB is able to receive another message if any of the following conditions is satisfied:

- The destination Mailbox is inactivated after the CAN bus has reached the first bit of Intermission field next to the frame that carried the message and its matching process has finished
- There is a previous pending move-in to the same destination Mailbox
- The Rx SMB is receiving a frame transmitted by the FlexCAN itself and the self-reception is disabled (CAN_MCR[SRXDIS] bit is asserted)
- Any CAN protocol error is detected

Note that the pending move-in is not cancelled if the module enters Freeze or Low-Power mode. It only stays on hold waiting for exiting Freeze and Low-Power mode and to be unlocked. If an MB is unlocked during Freeze mode, the move-in happens immediately.

The move-in process is the execution by the FlexCAN of the following steps:

1. Push IDHIT into the RXFIR FIFO if the message is destined to the Rx FIFO.
2. Read DATA0-3 and DATA4-7 words from the Rx SMB.
3. Write DATA0-3 and DATA4-7 words to the Rx Mailbox
4. Read the Control/Status and ID words from the Rx SMB.
5. Write Control/Status and ID words to the Rx Mailbox, and update the CODE field.

The move-in process is not atomic, in such a way that it is immediately cancelled by the inactivation of the destination Mailbox (see Mailbox inactivation) and in this case the Mailbox may be left partially updated, thus incoherent. The exception is if the move-in destination is an Rx FIFO Message Buffer, then the process cannot be cancelled.

The BUSY Bit (least significant bit of the CODE field) of the destination Message Buffer is asserted while the move-in is being performed to alert the CPU that the Message Buffer content is temporarily incoherent.

43.5.5.2 Move-out

The move-out process is the copy of the content from a Tx Mailbox to the Tx SMB when a message for transmission is available (see Section "Arbitration process"). The move-out occurs in the following conditions:

- The first bit of Intermission field
- During Bus Off state when TX Error Counter is in the 124 to 128 range
- During Bus Idle state
- During Wait For Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently out of Bus Idle state. In Bus Idle, the move-out has the lowest priority to the concurrent memory accesses.

43.5.6 Data coherence

In order to maintain data coherency and FlexCAN proper operation, the CPU must obey the rules described in Transmit process and Receive process.

43.5.6.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abortion of a pending transmission. A feedback mechanism is provided to inform the CPU if the transmission was aborted or if the frame could not be aborted and was transmitted instead.

Two primary conditions must be fulfilled in order to abort a transmission:

- CAN_MCR[AEN] bit must be asserted
- The first CPU action must be the writing of abort code (0b1001) into the CODE field of the Control and Status word.

Active MBs configured for transmission must be aborted first before they can be updated. If the abort code is written to a Mailbox that is currently being transmitted or to a Mailbox that was already loaded into the Tx SMB for transmission, the write operation is blocked and the transmission is not disturbed. However, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration
- There is an error during the transmission
- The module is put into Freeze mode

- The module enters the BusOff state
- There is an overload frame

If none of the conditions above are reached, the MB is transmitted correctly, the interrupt flag is set in the IFLAG register, and an interrupt to the CPU is generated (if enabled). The abort request is automatically cleared when the interrupt flag is set. On the other hand, if one of the above conditions is reached, the frame is not transmitted; therefore, the abort code is written into the CODE field, the interrupt flag is set in the IFLAG, and an interrupt is (optionally) generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, then the write operation is not blocked; therefore, the MB is updated and the interrupt flag is set. In this way the CPU just needs to read the abort code to make sure the active MB was *safely inactivated*. Although the AEN bit is asserted and the CPU wrote the abort code, in this case the MB is inactivated and not aborted, because the transmission did not start yet. One Mailbox is only aborted when the abort request is captured and kept pending until one of the previous conditions are satisfied.

43.5.6.2 Mailbox inactivation

Inactivation is a mechanism provided to protect the Mailbox against updates by the FlexCAN internal processes, thus allowing the CPU to rely on Mailbox data coherence after having updated it, even in Normal mode.

Inactivation of transmission Mailboxes must be performed just when MCR[AEN] bit is deasserted.

If a Mailbox is inactivated, it participates in neither the arbitration process nor the matching process until it is reactivated. See Transmit process and Receive process for more detailed instructions on how to inactivate and reactivate a Mailbox.

To inactivate a Mailbox, the CPU must update its CODE field to INACTIVE (either 0b0000 or 0b1000).

Because the user is not able to synchronize the CODE field update with the FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of the inactivated Rx Mailbox may be lost without notice, even if there are other Mailboxes with the same filter
- A frame containing the message within the inactivated Tx Mailbox may be transmitted without setting the respective IFLAG

In order to perform a *safe inactivation* and avoid the above consequences for Tx Mailboxes, the CPU must use the Transmission Abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the Mailbox (see Mailbox lock mechanism).

NOTE

Message Buffers that are part of the Rx FIFO cannot be inactivated. There is no write protection on the FIFO region by FlexCAN. CPU must maintain data coherency in the FIFO region when RFEN is asserted.

43.5.6.3 Mailbox lock mechanism

Other than Mailbox inactivation, FlexCAN has another data coherence mechanism for the receive process. When the CPU reads the Control and Status word of an Rx MB with codes FULL or OVERRUN, FlexCAN assumes that the CPU wants to read the whole MB in an atomic operation, and therefore it sets an internal lock flag for that MB. The lock is released when the CPU reads the Free Running Timer (global unlock operation), or when it reads the Control and Status word of another MB regardless of its code. A CPU write into the C/S word also unlocks the MB, but this procedure is not recommended for normal unlock use because it cancels a pending-move and potentially may lose a received message. The MB locking prevents a new frame from being written into the MB while the CPU is reading it.

NOTE

The locking mechanism applies only to Rx MBs that are not part of the FIFO and have a code different than INACTIVE (0b0000) or EMPTY[1] (0b0100). Also, Tx MBs can not be locked.

Suppose, for example, that the FIFO is disabled and the second and the fifth MBs of the array are programmed with the same ID, and FlexCAN has already received and stored messages into these two MBs. Suppose now that the CPU decides to read MB number 5 and at the same time another message with the same ID is arriving. When the CPU reads the Control and Status word of MB number 5, this MB is locked. The new message arrives and the matching algorithm finds out that there are no "free-to-receive" MBs, so it decides to override MB number 5. However, this MB is locked, so the new message can not be written there. It will remain in the Rx SMB waiting for the MB to be unlocked, and only then will be written to the MB.

1. In previous FlexCAN versions, reading the C/S word locked the MB even if it was EMPTY. This behavior is maintained when the IRMQ bit is negated.

If the MB is not unlocked in time and yet another new message with the same ID arrives, then the new message overwrites the one on the Rx SMB and there will be no indication of lost messages either in the CODE field of the MB or in the Error and Status Register.

While the message is being moved-in from the Rx SMB to the MB, the BUSY bit on the CODE field is asserted. If the CPU reads the Control and Status word and finds out that the BUSY bit is set, it should defer accessing the MB until the BUSY bit is negated.

Note

If the BUSY bit is asserted or if the MB is empty, then reading the Control and Status word does not lock the MB.

Inactivation takes precedence over locking. If the CPU inactivates a locked Rx MB, then its lock status is negated and the MB is marked as invalid for the current matching round. Any pending message on the Rx SMB will not be transferred anymore to the MB. An MB is unlocked when the CPU reads the Free Running Timer Register (see Section "Free Running Timer Register (CAN_TIMER)"), or the C/S word of another MB.

Lock and unlock mechanisms have the same functionality in both Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. However, the move-in is postponed if an unlock occurs during a low power mode (see Modes of operation), and it takes place only when the module resumes to Normal or Freeze modes.

43.5.7 Rx FIFO

The Rx FIFO is receive-only and is enabled by asserting the CAN_MCR[RFEN] bit. The reset value of this bit is zero to maintain software backward compatibility with previous versions of the module that did not have the FIFO feature.

The FIFO is 6-message deep. The memory region occupied by the FIFO structure (both Message Buffers and FIFO engine) is described in Rx FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a Message Buffer structure at the output of the FIFO.

The CAN_IFLAG1[BUF5I] (Frames available in Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO. An interrupt is generated if it is enabled by the corresponding mask bit. Upon receiving the interrupt, the CPU can read the message (accessing the output of the FIFO as a Message Buffer) and the CAN_RXFIR register and then clear the interrupt. If there are more messages in the FIFO the act of clearing the interrupt updates the output of the FIFO with the next message and

update the CAN_RXFIR with the attributes of that message, reissuing the interrupt to the CPU. Otherwise, the flag remains negated. The output of the FIFO is only valid whilst the CAN_IFLAG1[BUF5I] is asserted.

The CAN_IFLAG1[BUF6I] (Rx FIFO Warning) is asserted when the number of unread messages within the Rx FIFO is increased to 5 from 4 due to the reception of a new one, meaning that the Rx FIFO is almost full. The flag remains asserted until the CPU clears it.

The CAN_IFLAG1[BUF7I] (Rx FIFO Overflow) is asserted when an incoming message was lost because the Rx FIFO is full. Note that the flag will not be asserted when the Rx FIFO is full and the message was captured by a Mailbox. The flag remains asserted until the CPU clears it.

Clearing one of those three flags does not affect the state of the other two.

An interrupt is generated if an IFLAG bit is asserted and the corresponding mask bit is asserted too.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing work load. The filtering criteria is specified by programming a table of up to 128 32-bit registers, according to CAN_CTRL2[RFFN] setting, that can be configured to one of the following formats (see also Rx FIFO structure):

- Format A: 128 IDAFs (extended or standard IDs including IDE and RTR)
- Format B: 256 IDAFs (standard IDs or extended 14-bit ID slices including IDE and RTR)
- Format C: 512 IDAFs (standard or extended 8-bit ID slices)

Note

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the FIFO has a corresponding IDHIT (Identifier Acceptance Filter Hit Indicator) that can be read in the IDHIT field from C/S word, as shown in the Rx FIFO Structure description. Another way the CPU can obtain this information is by accessing the CAN_RXFIR register. The CAN_RXFIR[IDHIT] field refers to the message at the output of the FIFO and is valid while the CAN_IFLAG1[BUF5I] flag is asserted. The CAN_RXFIR register must be read only before clearing the flag, which guarantees that the information refers to the correct frame within the FIFO.

Up to 16 elements of the filter table are individually affected by the Individual Mask Registers (CAN_RXIMRx), according to the setting of CAN_CTRL2[RFFN], allowing very powerful filtering criteria to be defined. If the CAN_MCR[IRMQ] bit is negated, then the FIFO filter table is affected by CAN_RXFGMASK.

43.5.7.1 Rx FIFO under DMA Operation

The receive-only FIFO can support DMA, this feature is enabled by asserting both the CAN_MCR[RFEN] and CAN_MCR[DMA] bits. The reset value of CAN_MCR[DMA] bit is zero to maintain backward compatibility with previous versions of the module that did not have the DMA feature.

The DMA controller can read the received message by reading a Message Buffer structure at the FIFO output port at the 0x80-0x8C address range.

When CAN_MCR[DMA] is asserted the CPU must not access the FIFO output port address range. Before enabling the CAN_MCR[DMA], the CPU must service the IFLAGs asserted in the Rx FIFO region. Otherwise, these IFLAGs may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before disabling the CAN_MCR[DMA], the CPU must perform a clear FIFO operation.

The CAN_IFLAG1[BUF5I] (Frames available in Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO, consequently a DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the FIFO as a Message Buffer). The DMA reading process must end by reading address 0x8C, which clears the CAN_IFLAG1[BUF5I] and updates both the FIFO output with the next message (if FIFO is not empty) and the CAN_RXFIR register with the attributes of the new message. If there are more messages stored in the FIFO, the CAN_IFLAG1[BUF5I] will be re-asserted and another DMA request is issued. Otherwise, the flag remains negated.

NOTE

CAN_RXFIR register contents cannot be read after DMA completes the FIFO read. The IDHIT information is also available in the C/S word at address 0x080 (see Rx FIFO structure.

The CAN_IFLAG1[BUF6I] and CAN_IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Rx FIFO interruption and must not clear the related IFLAGs. In addition, the related IMASKs are not used to mask the generation of DMA requests.

43.5.7.2 Clear FIFO Operation

When CAN_MCR[RFEN] is asserted, the clear FIFO operation is a feature used to empty FIFO contents. With CAN_MCR[RFEN] asserted the Clear FIFO occurs when the CPU writes 1 in CAN_IFLAG1[BUF0I]. This operation can only be performed in Freeze Mode and is blocked by hardware in other modes. This operation does not clear the FIFO IFLAGs, consequently the CPU must service all FIFO IFLAGs before execute the clear FIFO task.

When Rx FIFO is working with DMA, the clear FIFO operation clears the CAN_IFLAG1[BUF5I] and the DMA request is canceled.

CAUTION

Clear FIFO operation does not clear IFLAGs, except when CAN_MCR[DMA] is asserted, in this case only the CAN_IFLAG1[BUF5I] is cleared.

43.5.8 CAN protocol related features

This section describes the CAN protocol related features.

43.5.8.1 Remote frames

Remote frame is a special kind of frame. The user can program a mailbox to be a Remote Request Frame by configuring the mailbox as Transmit with the RTR bit set to '1'. After the remote request frame is transmitted successfully, the mailbox becomes a Receive Message Buffer, with the same ID as before.

When a remote request frame is received by FlexCAN, it can be treated in three ways, depending on Remote Request Storing (CTRL2[RRS]) and Rx FIFO Enable (MCR[RFEN]) bits:

- If RRS is negated the frame's ID is compared to the IDs of the Transmit Message Buffers with the CODE field 0b1010. If there is a matching ID, then this mailbox frame will be transmitted. Note that if the matching mailbox has the RTR bit set, then FlexCAN will transmit a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response. The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. In the case that a remote request frame is received and matches a mailbox, this message buffer

immediately enters the internal arbitration process, but is considered as a normal Tx mailbox, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.

- If RRS is asserted the frame's ID is compared to the IDs of the receive mailboxes with the CODE field 0b0100, 0b0010 or 0b0110. If there is a matching ID, then this mailbox will store the remote frame in the same fashion of a data frame. No automatic remote response frame will be generated. The mask registers are used in the matching process.
- If RFEN is asserted FlexCAN will not generate an automatic response for remote request frames that match the FIFO filtering criteria. If the remote frame matches one of the target IDs, it will be stored in the FIFO and presented to the CPU. Note that for filtering formats A and B, it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted (if they match the ID). Remote Request Frames are considered as normal frames, and generate a FIFO overflow when a successful reception occurs and the FIFO is already full.

43.5.8.2 Overload frames

FlexCAN does transmit overload frames due to detection of following conditions on CAN bus:

- Detection of a dominant bit in the first/second bit of Intermission
- Detection of a dominant bit at the 7th bit (last) of End of Frame field (Rx frames)
- Detection of a dominant bit at the 8th bit (last) of Error Frame Delimiter or Overload Frame Delimiter

43.5.8.3 Time stamp

The value of the Free Running Timer is sampled at the beginning of the Identifier field on the CAN bus, and is stored at the end of "move-in" in the TIME STAMP field, providing network behavior with respect to time.

The Free Running Timer is clocked by the FlexCAN bit-clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The Free Running Timer is not incremented during Disable, Doze, Stop, and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See the TSYN description in Control 1 Register (CAN_CTRL1).

43.5.8.4 Protocol timing

The following figure shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule. The clock source bit CLKSRC in the CAN_CTRL1 Register defines whether the internal clock is connected to the output of a crystal oscillator (Oscillator Clock) or to the Peripheral Clock. In order to guarantee reliable operation, the clock source should be selected while the module is in Disable Mode (MDIS bit set in the Module Configuration Register).

NOTE

Please refer to the clock distribution chapter (module clocks table) to identify the proper clock source.

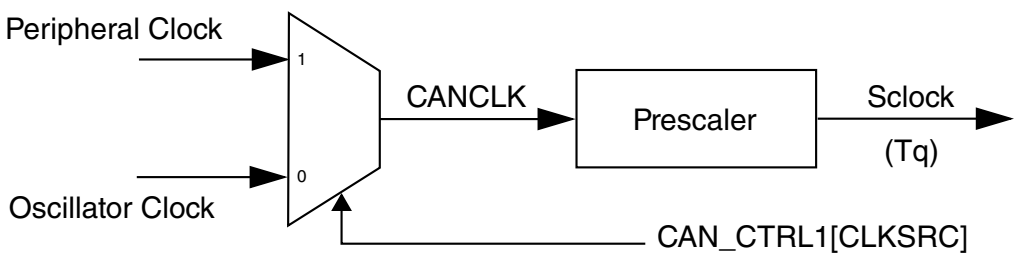


Figure 43-2. CAN engine clocking scheme

The oscillator clock should be selected whenever a tight tolerance (up to 0.1%) is required in the CAN bus timing. The crystal oscillator clock has better jitter performance than the peripheral clock.

The FlexCAN module supports a variety of means to setup bit timing parameters that are required by the CAN protocol. The Control 1 Register (CAN_CTRL1) has various fields used to control bit timing parameters: PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW.

The CAN Bit Timing register (CAN_CBT) extends the range of the CAN bit timing variables in CAN_CTRL1.

The PRESDIV field (as well as its extended range EPRESDIV) defines the Prescaler Value (see the equation below) that generates the Serial Clock (Sclock), whose period defines the 'time quantum' used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time handled by the CAN engine.

$$Tq = \frac{(PRESDIV + 1)}{f_{CANCLK}}$$

The bit rate, which defines the rate the CAN message is either received or transmitted, is given by the formula:

CAN Bit Time = (Number of Time Quanta in 1 bit time) * Tq

$$\text{Bit Rate} = \frac{1}{\text{CAN Bit Time}}$$

A bit time is subdivided into three segments[1] (see Figure 43-3 and Table 43-13):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section
- Time Segment 1: This segment includes the Propagation Segment and the Phase Segment 1 of the CAN standard. It can be programmed by setting the PROPSEG and the PSEG1 fields of the CAN_CTRL1 Register so that their sum (plus 2) is in the range of 2 to 16 time quanta. When CAN_CBT[BTF] bit is asserted, FlexCAN uses EPROPSEG and EPSEG1 fields from CAN_CBT register so that their sum (plus 2) is in the range of 2 to 96 time quanta.
- Time Segment 2: This segment represents the Phase Segment 2 of the CAN standard. It can be programmed by setting the PSEG2 field of the CAN_CTRL1 Register (plus 1) to be 2 to 8 time quanta long. When CAN_CBT[BTF] bit is asserted, FlexCAN uses EPSEG2 fields of CAN_CBT register so that its value (plus 1) is in the range of 2 to 32 time quanta. The Time Segment 2 cannot be smaller than the Information Processing Time (IPT), which value is 2 time quanta in FlexCAN.

NOTE

The bit time defined by the above time segments must not be smaller than 5 time quanta. For bit time calculations, use an Information Processing Time (IPT) of 2, which is the value implemented in the FlexCAN module.

1. For further explanation of the underlying concepts, see ISO 11898-1. See also the CAN 2.0A/B protocol specification for bit timing.

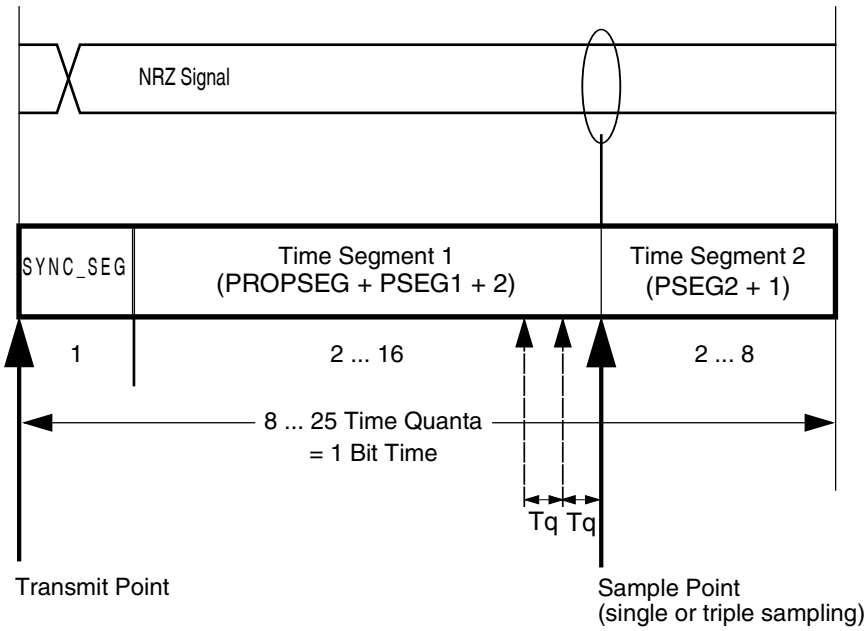


Figure 43-3. Segments within the bit time (example using CAN_CTRL1 bit timing variables for Classical CAN format)

Table 43-13. Time segment syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the bus during this period.
TSEG1	Corresponds to the sum of PROPSEG and PSEG1.
TSEG2	Corresponds to the PSEG2 value.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node samples the bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The following table gives some examples of the CAN compliant segment settings for Classical CAN format (Bosch CAN 2.0B) messages.

Table 43-14. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Re-synchronization jump width
5 .. 10	2	1 .. 2
4 .. 11	3	1 .. 3
5 .. 12	4	1 .. 4
6 .. 13	5	1 .. 4
7 .. 14	6	1 .. 4
8 .. 15	7	1 .. 4
9 .. 16	8	1 .. 4

Note

The user must ensure the bit time settings are in compliance with the CAN Protocol standard (ISO 11898-1).

Whenever CAN bit is used as a measure of time duration (e.g. estimating the occurrence of a CAN bit event in a message), the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$\text{NumClkBit} = \frac{f_{SYS}}{f_{CANCLK}} \text{ X } (\text{PRESDIV} + 1) \text{ X } (\text{PROPSEG} + \text{PSEG1} + \text{PSEG2} + 4)$$

where:

- NumClkBit is the number of peripheral clocks in one CAN bit;
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure "CAN Engine Clocking Scheme"), in Hz;
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz;
- PSEG1 is the value in CAN_CTRL1[PSEG1] field;
- PSEG2 is the value in CAN_CTRL1[PSEG2] field;
- PROPSEG is the value in CAN_CTRL1[PROPSEG] field;
- PRESDIV is the value in CAN_CTRL1[PRESDIV] field.

The formula above is also applicable to the alternative CAN bit timing variables described in the CAN Bit Timing Register (CAN_CBT).

For example, 180 CAN bits = (180 x NumClkBit) peripheral clock periods.

43.5.8.5 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in and move-out processes are executed during certain time windows inside the CAN frame, as shown in the following figures.

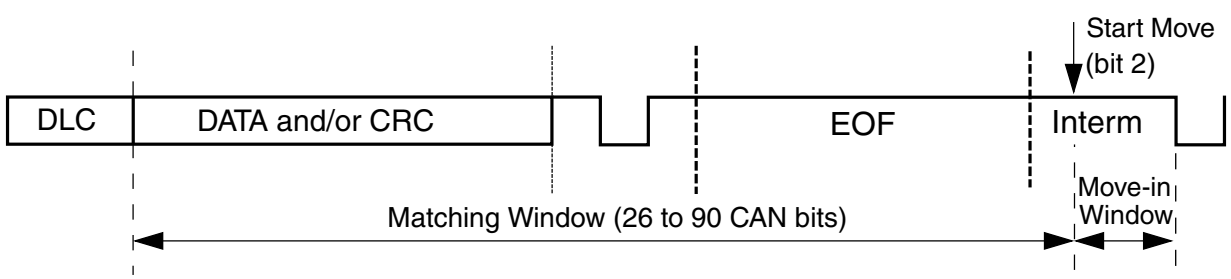


Figure 43-4. Matching and move-in time windows

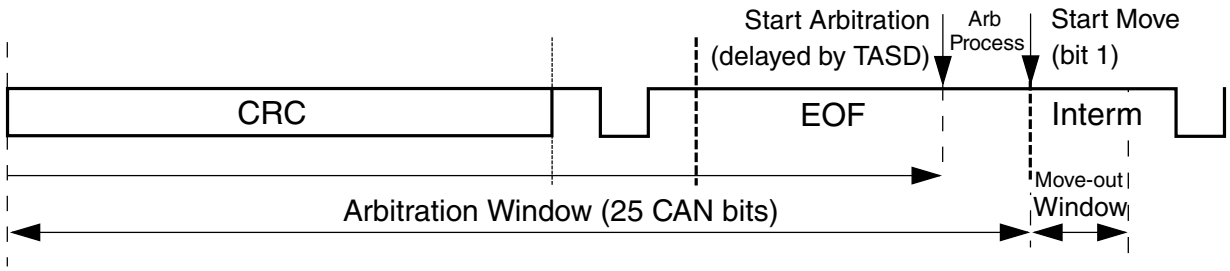


Figure 43-5. Arbitration and move-out time windows

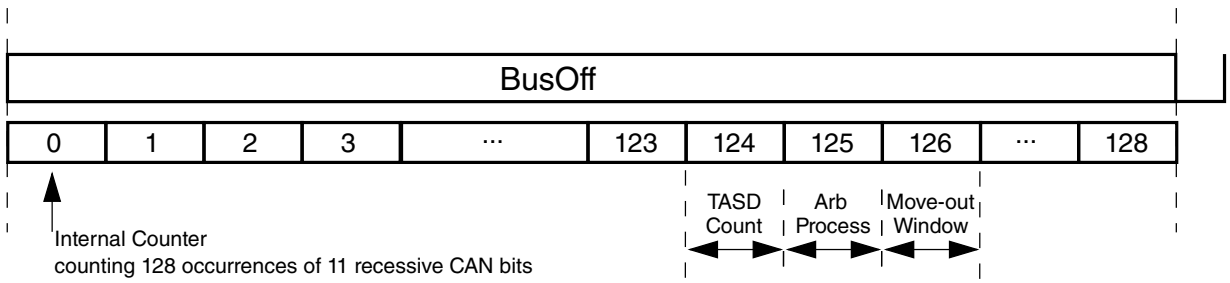


Figure 43-6. Arbitration at the end of bus off and move-out time windows

NOTE

In the preceding figures, the matching and arbitration timing does not take into account the delay caused by the concurrent memory access due to the CPU or other internal FlexCAN sub-blocks.

43.5.8.6 Tx Arbitration start delay

The Tx Arbitration Start Delay (TASD) bit field in Control 2 register (CAN_CTRL2[TASD]) is a variable that indicates the number of CAN bits used by FlexCAN to delay the Tx Arbitration process start point from the first bit of CRC field of the current frame. This variable can be written only in Freeze mode because it is blocked by hardware in other modes.

The transmission performance is impacted by the ability of the CPU to reconfigure Message Buffers (MBs) for transmission after the end of the internal Arbitration process, where FlexCAN finds the winner MB for transmission (see Arbitration process). If the Arbitration ends too early before the first bit of Intermission field, then there is a chance that the CPU reconfigures some Tx MBs and the winner MB is no longer the best candidate to be transmitted.

TASD is useful to optimize the transmission performance by defining the Arbitration start point, as shown in the next figure, based on factors such as:

- The peripheral-to-oscillator clock ratio

- CAN bit timing variables that determine the CAN bit rate
- The number of Message Buffers (MBs) in use by the Matching and Arbitration processes.

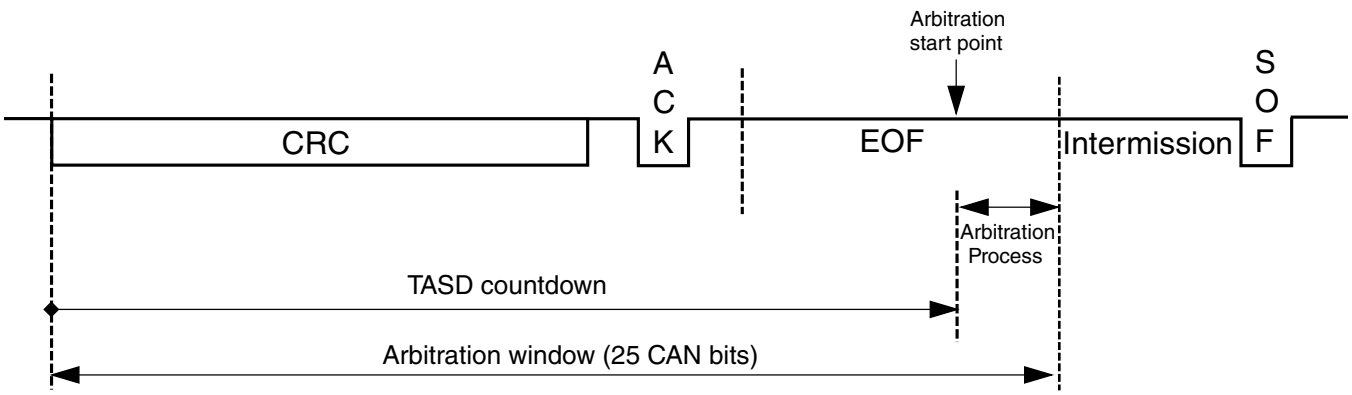


Figure 43-7. Optimal Tx Arbitration start point

The duration of an Arbitration process, in terms of CAN bits, is directly proportional to the number of available MBs and to the CAN bit rate, and inversely proportional to the peripheral clock frequency.

The optimal Arbitration timing is that in which the last MB is scanned right before the first bit of the Intermission field of a CAN frame. For instance, if there are few MBs and the peripheral/oscillator clock ratio is high and the CAN baud rate is low, then the Arbitration can be placed closer to the frame's end, adding more delay to its start point, and vice-versa.

If TASD is set to 0 then the Arbitration start is not delayed and more time is reserved for Arbitration. On the other hand, if TASD is close to 24 then the CPU can configure a Tx MB later and less time is reserved for Arbitration. If too little time is reserved for Arbitration the FlexCAN may be not be able to find a winner MB in time to be transmitted with the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

$$TASD = 25 - \left(\frac{f_{CANCLK} \times [\, MAXMB + 3 - (RFEN \times 8) - (RFEN \times RFFN \times 2)\,] \times 2}{f_{SYS} \times [1 + (PSEG1 + 1) + (PSEG2 + 1) + (PROPSEG + 1)\,] \times (PRESDIV + 1)} \right)$$

where:

- MAXMB is the value in CAN_CTRL1[MAXMB] field
- f_{CANCLK} is the oscillator clock, in Hz

- f_{SYS} is the peripheral clock, in Hz
- RFEN is the value in CAN_CTRL1[RFEN] bit
- RFFN is the value in CAN_CTRL2[RFFN] field
- PSEG1 is the value in CAN_CTRL1[PSEG1] field
- PSEG2 is the value in CAN_CTRL1[PSEG2] field
- PROPSEG is the value in CAN_CTRL1[PROPSEG] field
- PRESDIV is the value in CAN_CTRL1[PRESDIV] field

See also Protocol timing for more details.

43.5.9 Clock domains and restrictions

The FlexCAN module has two clock domains asynchronous to each other:

- The Bus Domain feeds the Control Host Interface (CHI) submodule and is derived from the peripheral clock.
- The Oscillator Domain feeds the CAN Protocol Engine (PE) submodule and is derived directly from a crystal oscillator clock, so that very low jitter performance can be achieved on the CAN bus.

When CAN_CTRL1[CLKSRC] bit is set, synchronous operation occurs because both domains are connected to the peripheral clock (creating a 1:1 ratio between the peripheral and oscillator clocks).

When the two domains are connected to clocks with different frequencies and/or phases, there are restrictions on the frequency relationship between the two clock domains. In the case of asynchronous operation, the Bus Domain clock frequency must always be greater than the Oscillator Domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When doing matching and arbitration, FlexCAN needs to scan the whole Message Buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. In order to have sufficient time to do that, the following requirements must be observed:

- The peripheral clock frequency can not be smaller than the oscillator clock frequency
- For 16 Mailboxes, the minimum number of peripheral clocks per CAN bit is 16

The minimum number of peripheral clocks per CAN bit determines the minimum peripheral clock frequency for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit, that can be defined by adjusting one or more of the bit timing values contained in either the Control 1 Register (CAN_CTRL1) or CAN Bit Time register (CAN_CBT). The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be 8; therefore, the oscillator clock frequency should be at least 8 times the CAN bit rate.

43.5.10 Modes of operation details

The FlexCAN module has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following sub-sections contain functional details on Freeze mode and the low-power modes.

CAUTION

"Permanent Dominant" failure on CAN Bus line is not supported by FlexCAN. If a Low-Power request or Freeze mode request is done during a "Permanent Dominant", the corresponding acknowledge can never be asserted.

43.5.10.1 Freeze mode

This mode is requested either by the CPU through the assertion of the HALT bit in the CAN_MCR Register or when the chip is put into Debug mode. In both cases it is also necessary that the FRZ bit is asserted in the CAN_MCR Register and the module is not in a low-power mode.

The acknowledgement is obtained through the assertion by the FlexCAN of FRZ_ACK bit in the same register. The CPU must only consider the FlexCAN in Freeze mode when both request and acknowledgement conditions are satisfied.

When Freeze mode is requested, FlexCAN does the following:

- Waits to be in either Intermission, Passive Error, Bus Off or Idle state
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in does not prevent going to Freeze mode.
- Ignores the Rx input pin and drives the Tx pin as recessive
- Stops the prescaler, thus halting all CAN protocol activities

- Grants write access to the Error Counters Register, which is read-only in other modes
- Sets the NOT_RDY and FRZ_ACK bits in CAN_MCR

After requesting Freeze mode, the user must wait for the FRZ_ACK bit to be asserted in CAN_MCR before executing any other action, otherwise FlexCAN may operate in an unpredictable way. In Freeze mode, all memory mapped registers are accessible, except for CAN_CTRL1[CLKSRC] bit that can be read but cannot be written.

Exiting Freeze mode is done in one of the following ways:

- CPU negates the FRZ bit in the CAN_MCR Register
- The chip is removed from Debug Mode and/or the HALT bit is negated

The FRZ_ACK bit is negated after the protocol engine recognizes the negation of the freeze request. When out of Freeze mode, FlexCAN tries to re-synchronize to the CAN bus by waiting for 11 consecutive recessive bits.

43.5.10.2 Module Disable mode

This low power mode is normally used to temporarily disable a complete FlexCAN block, with no power consumption. It is requested by the CPU through the assertion of the CAN_MCR[MDIS] bit, and the acknowledgement is obtained through the assertion by the FlexCAN of the CAN_MCR[LPMACK] bit. The CPU must only consider the FlexCAN in Disable mode when both request and acknowledgement conditions are satisfied.

If the module is disabled during Freeze mode, it requests to disable the clocks to the PE and CHI sub-modules, sets the LPMACK bit and negates the FRZACK bit.

If the module is disabled during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and then checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Shuts down the clocks to the PE and CHI sub-modules
- Sets the NOTRDY and LPMACK bits in CAN_MCR

The Bus Interface Unit continues to operate, enabling the CPU to access memory mapped registers, except the Rx Mailboxes Global Mask Registers, the Rx Buffer 14 Mask Register, the Rx Buffer 15 Mask Register, the Rx FIFO Global Mask Register. The Rx FIFO Information Register, the Message Buffers, the Rx Individual Mask Registers, and the reserved words within RAM may not be accessed when the module is in Disable Mode. Exiting from this mode is done by negating the MDIS bit by the CPU, which causes the FlexCAN to request to resume the clocks and negate the LPMACK bit after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

43.5.10.3 Doze mode

This is a system low power mode in which the CPU bus is kept alive and a global Doze mode request is sent to all peripherals asking them to enter low-power mode. When Doze mode is globally requested, the DOZE bit in CAN_MCR Register needs to have been asserted previously for Doze mode to be triggered. The acknowledgement is obtained through the assertion by the FlexCAN of the LPMACK bit in the same register. The CPU must only consider the FlexCAN in Doze mode when both request and acknowledgement conditions are satisfied.

If Doze mode is triggered during Freeze mode, FlexCAN requests to shut down the clocks to the PE and CHI sub-modules, sets the LPMACK bit and negates the FRZACK bit. If Doze Mode is triggered during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Shuts down the clocks to the PE and CHI sub-modules
- Sets the NOTRDY and LPMACK bits in CAN_MCR

The Bus Interface Unit continues to operate, enabling the CPU to access memory mapped registers, except the Rx Mailboxes Global Mask Registers, the Rx Buffer 14 Mask Register, the Rx Buffer 15 Mask Register, the Rx FIFO Global Mask Register. The Rx FIFO Information Register, the Message Buffers, the Rx Individual Mask Registers, and the reserved words within RAM may not be accessed when the module is in Doze Mode.

Exiting Doze mode is done in one of the following ways:

- CPU removing the Doze mode request
- CPU negating the DOZE bit of the CAN_MCR Register
- Self Wake mechanism

In the Self Wake mechanism, if the SLFWAK bit in CAN_MCR Register was set at the time FlexCAN entered Doze mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN negates the DOZE bit, requests to resume its clocks and negates the LPMACK after the CAN protocol engine recognizes the negation of the Doze mode request. It also sets the WAKINT bit in the ESR Register and, if enabled by the WAKMSK bit in CAN_MCR, generates a Wake Up interrupt to the CPU. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLFWAK and WAKMSK upon wake-up from Doze mode.

Table 43-15. Wake-up from Doze mode

SLFWAK	WAKINT	WAKMSK	FlexCAN clocks enabled	Wake-up interrupt generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	Yes	No
1	1	1	Yes	Yes

The sensitivity to CAN bus activity can be modified by applying a low-pass filter function to the Rx CAN input line while in Doze Mode. See the WAKSRC bit in the description of the Module Configuration Register (CAN_MCR). This feature can be used to protect FlexCAN from waking up due to short glitches on the CAN bus lines. Such glitches can result from electromagnetic interference within noisy environments.

43.5.10.4 Stop mode

This is a system low-power mode in which all chip clocks can be stopped for maximum power savings. The Stop mode is globally requested by the CPU and the acknowledgement is obtained through the assertion by the FlexCAN of a Stop Acknowledgement signal. The CPU must only consider the FlexCAN in Stop mode when both request and acknowledgement conditions are satisfied.

If FlexCAN receives the global Stop mode request during Freeze mode, it sets the LPMACK bit, negates the FRZACK bit and then sends the Stop Acknowledge signal to the CPU, in order to shut down the clocks globally.

If Stop mode is requested during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Sets the NOTRDY and LPMACK bits in CAN_MCR
- Sends a Stop Acknowledge signal to the CPU, so that it can shut down the clocks globally

Stop mode is exited when the CPU resumes the clocks and removes the Stop Mode request. This can be as a result of the Self Wake mechanism.

In the Self Wake mechanism, if the SLFWAK bit in CAN_MCR Register was set at the time FlexCAN entered Stop mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN sets the WAKINT bit in the CAN_ESR Register and, if enabled by the WAKMSK bit in CAN_MCR, generates a Wake Up interrupt to the CPU. Upon receiving the interrupt, the CPU should resume the clocks and remove the Stop mode request. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLFWAK and WAKMSK upon wake-up from Stop mode. Note that wake-up from Stop mode only works when both bits are asserted.

After the CAN protocol engine recognizes the negation of the Stop mode request, the FlexCAN negates the LPMACK bit. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up.

Table 43-16. Wake-up from Stop Mode

SLFWAK	WAKINT	WAKMSK	Chip clocks enabled	Wake-up interrupt generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No

Table continues on the next page...

Table 43-16. Wake-up from Stop Mode (continued)

SLFWAK	WAKINT	WAKMSK	Chip clocks enabled	Wake-up interrupt generated
1	0	1	No	No
1	1	0	No	No
1	1	1	Yes	Yes

The sensitivity to CAN bus activity can be modified by applying a low-pass filter function to the Rx CAN input line while in Stop mode. See the WAKSRC bit in in the description of the Module Configuration Register (CAN_MCR). This feature can be used to protect FlexCAN from waking up due to short glitches on the CAN bus lines. Such glitches can result from electromagnetic interference within noisy environments.

43.5.11 Interrupts

The module has many interrupt sources: interrupts due to message buffers and interrupts due to the ORed interrupts from MBs, Bus Off, Bus Off Done, Error, Wake Up, Tx Warning, and Rx Warning.

Each one of the message buffers can be an interrupt source, if its corresponding IMASK bit is set. There is no distinction between Tx and Rx interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each of the buffers has an assigned flag bit in the CAN_IFLAG registers. The bit is set when the corresponding buffer completes a successful transfer and is cleared when the CPU writes it to 1 (unless another interrupt is generated at the same time).

Note

It must be guaranteed that the CPU clears only the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.

If the Rx FIFO is enabled (CAN_MCR[RFEN] = 1) and DMA is disabled (CAN_MCR[DMA] = 0), the interrupts corresponding to MBs 0 to 7 have different meanings. Bit 7 of the CAN_IFLAG1 register becomes the "FIFO Overflow" flag; bit 6 becomes the "FIFO Warning" flag, bit 5 becomes the "Frames Available in FIFO" flag and bits 4-0 are unused. See the description of the Interrupt Flags 1 Register (CAN_IFLAG1) for more information.

If both Rx FIFO and DMA are enabled (CAN_MCR[RFEN] and CAN_MCR[DMA] = 1) the FlexCAN does not generate any FIFO interrupt. Bit 5 of the CAN_IFLAG1 register still indicates "Frames Available in FIFO" and generates a DMA request. Bits 7, 6, 4-0 are unused.

For a combined interrupt where multiple MB interrupt sources are OR'd together, the interrupt is generated when any of the associated MBs (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the CAN_IFLAG registers to determine which MB or FIFO source caused the interrupt.

The interrupt sources for Bus Off, Bus Off Done, Error, Wake Up, Tx Warning and Rx Warning generate interrupts like the MB interrupt sources, and can be read from CAN_ESR1 register. The Bus Off, Error, Tx Warning, and Rx Warning interrupt mask bits are located in the CAN_CTRL1 Register; the Wake-Up interrupt mask bit is located in the CAN_MCR.

43.5.12 Bus interface

The CPU access to FlexCAN registers are subject to the following rules:

- Unrestricted read and write access to supervisor registers (registers identified with S/U in Table "Module Memory Map" in Supervisor Mode or with S only) results in access error.
- Read and write access to implemented reserved address space results in access error.
- Write access to positions whose bits are all currently read-only results in access error. If at least one of the bits is not read-only then no access error is issued. Write permission to positions or some of their bits can change depending on the mode of operation or transitory state. Refer to register and bit descriptions for details.
- Read and write access to unimplemented address space results in access error.
- Read and write access to RAM located positions during Low Power Mode results in access error.
- It is possible for the RXIMR memory region to be considered as general purpose memory and available for access. There are two ways of doing this:
  a. If CAN_MCR[IRMQ] is cleared, the individual masks (RXIMR) are disabled. In this case the RXIMR memory region is considered as general purpose memory.
  b. If CAN_MCR[MAXMB] is programmed with a value smaller than the available number of MBs, then the unused memory space can be used as general purpose RAM space. Note that reserved words within RAM cannot be used. As an example, suppose FlexCAN's RAM can support up to 16 MBs, CAN_CTRL2[RFFN] is 0x0, and CAN_MCR[MAXMB] is programmed with zero. The maximum number of MBs in this case becomes one. The RAM starts

at 0x0080, and the space from 0x0080 to 0x008F is used by the one MB. The memory space from 0x0090 to 0x017F is available. The space between 0x0180 and 0x087F is reserved. The space from 0x0880 to 0x0883 is used by the one Individual Mask and the available memory in the Mask Registers space would be from 0x0884 to 0x08BF. From 0x08C0 through 0x09DF there are reserved words for internal use which cannot be used as general purpose RAM. As a general rule, free memory space for general purpose depends only on MAXMB.

43.6 Initialization/application information

This section provide instructions for initializing the FlexCAN module.

43.6.1 FlexCAN initialization sequence

The FlexCAN module may be reset in three ways:

- Chip level hard reset, which resets all memory mapped registers asynchronously
- SOFTRST bit in MCR, which resets some of the memory mapped registers synchronously. See Table 43-2 to see what registers are affected by soft reset.
- Chip level soft reset, which has the same effect as the SOFTRST bit in MCR

Soft reset is synchronous and has to follow an internal request/acknowledge procedure across clock domains. Therefore, it may take some time to fully propagate its effects. The CAN_MCR[SOFTRST] bit remains asserted while soft reset is pending, so software can poll this bit to know when the reset has completed. Also, soft reset can not be applied while clocks are shut down in a low power mode. The low power mode should be exited and the clocks resumed before applying soft reset.

The clock source should be selected while the module is in Disable mode (see CAN_CTRL1[CLKSRC] bit). After the clock source is selected and the module is enabled (CAN_MCR[MDIS] bit negated), FlexCAN automatically goes to Freeze mode. In Freeze mode, FlexCAN is un-synchronized to the CAN bus, the HALT and FRZ bits in CAN_MCR Register are set, the internal state machines are disabled and the FRZACK and NOTRDY bits in the CAN_MCR Register are set. The Tx pin is in recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Note that the Message Buffers and the Rx Individual Mask Registers are not affected by reset, so they are not automatically initialized.

For any configuration change/initialization it is required that FlexCAN is put into Freeze mode (see Freeze mode). The following is a generic initialization sequence applicable to the FlexCAN module:

- Initialize the Module Configuration Register (CAN_MCR)
  - Enable the individual filtering per MB and reception queue features by setting the IRMQ bit
  - Enable the warning interrupts by setting the WRNEN bit
  - If required, disable frame self reception by setting the SRXDIS bit
  - Enable the Rx FIFO by setting the RFEN bit
  - If Rx FIFO is enabled and DMA is required, set DMA bit
  - Enable the abort mechanism by setting the AEN bit
  - Enable the local priority feature by setting the LPRIOEN bit
- Initialize the Control 1 Register (CAN_CTRL1) and optionally the CAN Bit Timing Register (CAN_CBT).
  - Determine the bit timing parameters: PROPSEG, PSEG1, PSEG2, RJW
  - Optionally determine the bit timing parameters: EPROPSEG, EPSEG1, EPSEG2, ERJW
  - Determine the bit rate by programming the PRESDIV field and optionally the EPRESDIV field
  - Determine the internal arbitration mode (LBUF bit)
- Initialize the Message Buffers
  - The Control and Status word of all Message Buffers must be initialized
  - If Rx FIFO was enabled, the ID filter table must be initialized
  - Other entries in each Message Buffer should be initialized as required
- Initialize the Rx Individual Mask Registers (CAN_RXIMRn)
- Set required interrupt mask bits in the CAN_IMASK Registers (for all MB interrupts), in CAN_MCR Register for Wake-Up interrupt and in CAN_CTRL1 / CAN_CTRL2 Registers (for Bus Off and Error interrupts)
- Negate the HALT bit in CAN_MCR

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

Chapter 44
Serial Peripheral Interface (SPI)

44.1 Chip-specific SPI information

44.1.1 SPI Instantiation Information

This chip contains one SPI module. It has 4-byte RX and TX FIFO, as well as DMA support.

44.1.2 SPI signals

Signal	I/O	Connected to
Receive complete	Output	DMA_MUX source 16
Transmit complete	Output	DMA_MUX source 17

44.1.3 SPI clocking

The SPI module is clocked by the fast peripheral clock (the SPI refers to it as system clock). The module has an internal divider, with a minimum divide of two. So, the SPI can run at a maximum frequency of fast peripheral clock/2.

44.1.4 Number of CTARs

SPI CTAR registers define different transfer attribute configurations. The SPI module supports up to eight CTAR registers. This device supports two CTARs on the instances of the SPI.

In master mode, the CTAR registers define combinations of transfer attributes, such as frame size, clock phase, clock polarity, data bit ordering, baud rate, and various delays. In slave mode only CTAR0 is used, and a subset of its bitfields sets the slave transfer attributes.

44.1.5 TX FIFO size

Table 44-1. SPI transmit FIFO size

SPI Module	Transmit FIFO size
SPI0	4

44.1.6 RX FIFO Size

SPI supports up to 16-bit frame size during reception.

Table 44-2. SPI receive FIFO size

SPI Module	Receive FIFO size
SPI0	4

44.1.7 Number of PCS signals

The following table shows the number of peripheral chip select signals available per SPI module.

Table 44-3. SPI PCS signals

SPI Module	PCS Signals
SPI0	SPI_PCS[5:0]

44.1.8 SPI Operation in Low Power Modes

In VLPR and VLPW modes the SPI is functional; however, the reduced system frequency also reduces the max frequency of operation for the SPI. In VLPR and VLPW modes the max SPI_CLK frequency is 2MHz.

In stop and VLPS modes, the clocks to the SPI module are disabled. The module is not functional, but it is powered so that it retains state.

There is one way to wake from stop mode via the SPI, which is explained in the following section.

44.1.8.1 Using GPIO Interrupt to Wake from stop mode

Here are the steps to use a GPIO to create a wakeup upon reception of SPI data in slave mode:

1. Point the GPIO interrupt vector to the desired interrupt handler.
2. Enable the GPIO input to generate an interrupt on either the rising or falling edge (depending on the polarity of the chip select signal).
3. Enter Stop or VLPS mode and Wait for the GPIO interrupt.

NOTE

It is likely that in using this approach the first word of data from the SPI host might not be received correctly. This is dependent on the transfer rate used for the SPI, the delay between chip select assertion and presentation of data, and the system interrupt latency.

44.1.9 SPI Doze Mode

The Doze mode for the SPI module is the same as the Wait and VLPW modes for the chip.

44.1.10 SPI Interrupts

The SPI has multiple sources of interrupt requests. However, these sources are OR'd together to generate a single interrupt request to the interrupt controller. When an SPI interrupt occurs, read the SPI_SR to determine the exact interrupt source.

44.2 Introduction

The serial peripheral interface (SPI) module provides a synchronous serial bus for communication between a chip and an external peripheral device.

44.2.1 Block Diagram

The block diagram of this module is as follows:

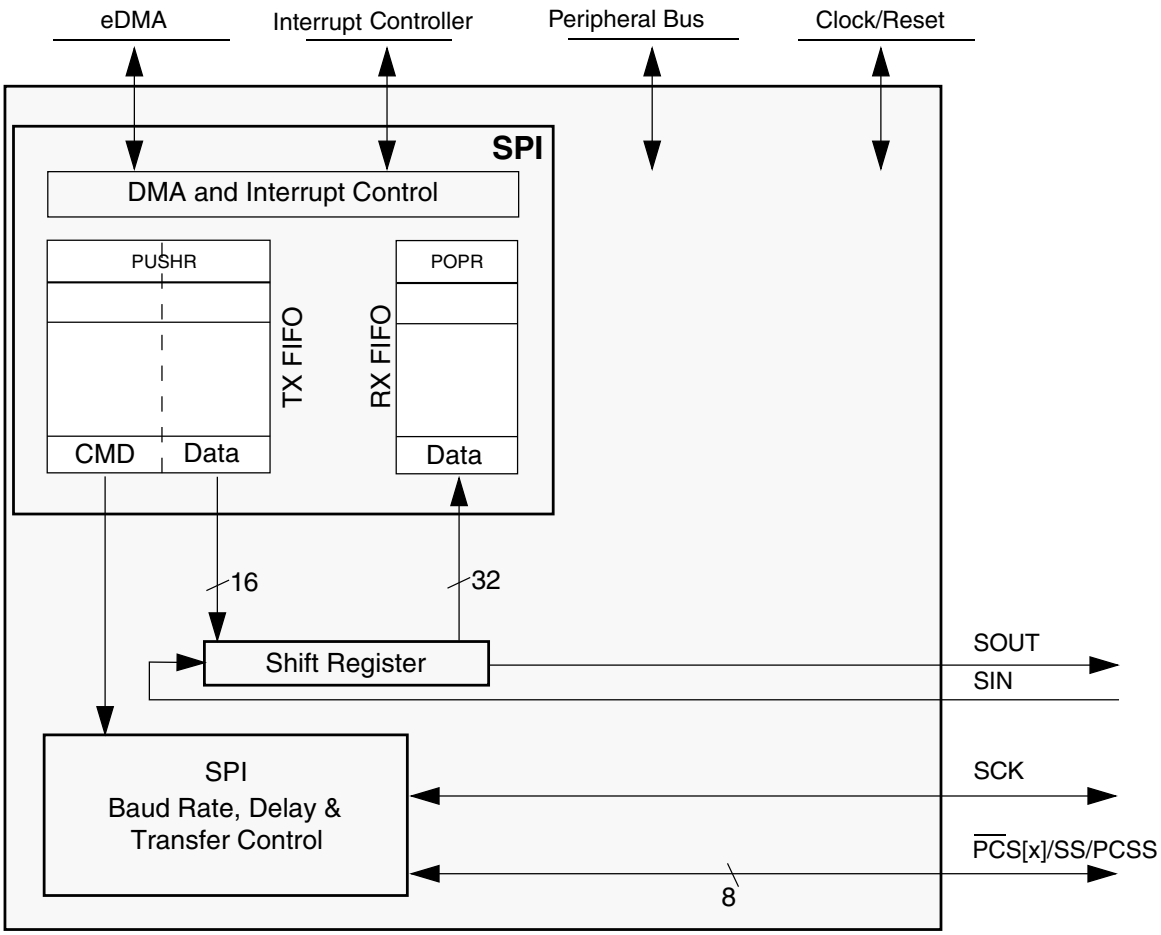


Figure 44-1. SPI Block Diagram

44.2.2 Features

The module supports the following features:

- Full-duplex, three-wire synchronous transfers
- Master mode
- Slave mode
- Data streaming operation in Slave mode with continuous slave selection
- Buffered transmit operation using the transmit first in first out (TX FIFO) with depth of 4 entries

- Buffered receive operation using the receive FIFO (RX FIFO) with depth of 4 entries
- Asynchronous clocking scheme for Register and Protocol Interfaces
- TX and RX FIFOs can be disabled individually for low-latency updates to SPI queues
- Visibility into TX and RX FIFOs for ease of debugging
- Programmable transfer attributes on a per-frame basis:
  - two transfer attribute registers
  - Serial clock (SCK) with programmable polarity and phase
  - Various programmable delays
  - Programmable serial frame size: 4 to 16
    - SPI frames longer than 16 bits can be supported using the continuous selection format.
  - Continuously held chip select capability
- 6 peripheral chip selects (PCSes), expandable to 64 with external demultiplexer
- Deglitching support for up to 32 peripheral chip selects (PCSes) with external demultiplexer
- DMA support for adding entries to TX FIFO and removing entries from RX FIFO:
  - TX FIFO is not full (TFFF)
  - RX FIFO is not empty (RFDF)
- Interrupt conditions:
  - End of Queue reached (EOQF)
  - TX FIFO is not full (TFFF)
  - Transfer of current frame complete (TCF)
  - Attempt to transmit with an empty Transmit FIFO (TFUF)
  - RX FIFO is not empty (RFDF)
  - Frame received while Receive FIFO is full (RFOF)
- Global interrupt request line
- Modified SPI transfer formats for communication with slower peripheral devices
- Power-saving architectural features:

- Support for Stop mode
- Support for Doze mode

44.2.3 Interface configurations

44.2.3.1 SPI configuration

The Serial Peripheral Interface SPI configuration allows the module to send and receive serial data. This configuration allows the module to operate as a basic SPI block with internal FIFOs supporting external queue operation. Transmitted data and received data reside in separate FIFOs. The host CPU or a DMA controller read the received data from the Receive FIFO and write transmit data to the Transmit FIFO.

For queued operations, the SPI queues can reside in system RAM, external to the module. Data transfers between the queues and the module FIFOs are accomplished by a DMA controller or host CPU. The following figure shows a system example with DMA, SPI, and external queues in system RAM.

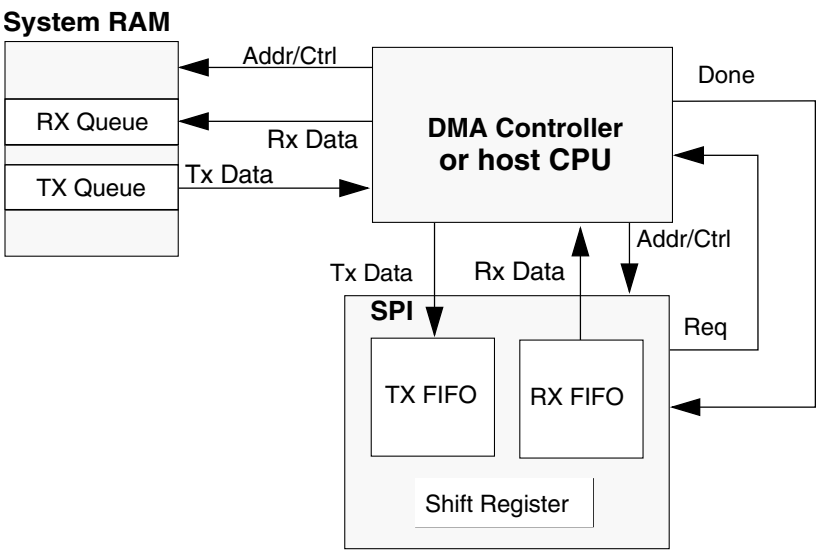


Figure 44-2. SPI with queues and DMA

44.2.4 Modes of Operation

The module supports the following modes of operation that can be divided into two categories:

- Module-specific modes:

    - Master mode
    - Slave mode
    - Module Disable mode
- Chip-specific modes:
    - External Stop mode
    - Debug mode

The module enters module-specific modes when the host writes a module register. The chip-specific modes are controlled by signals external to the module. The chip-specific modes are modes that a chip may enter in parallel to the block-specific modes.

44.2.4.1 Master Mode

Master mode allows the module to initiate and control serial communication. In this mode, these signals are controlled by the module and configured as outputs:

- SCK
- SOUT
- PCS[*x*]

44.2.4.2 Slave Mode

Slave mode allows the module to communicate with SPI bus masters. In this mode, the module responds to externally controlled serial transfers. The SCK signal and the PCS[0]/$\overline{\text{SS}}$ signals are configured as inputs and driven by an SPI bus master.

44.2.4.3 Module Disable Mode

The Module Disable mode can be used for chip power management. The clock to the non-memory mapped logic in the module can be stopped while in the Module Disable mode.

44.2.4.4 External Stop Mode

External Stop mode is used for chip power management. The module supports the Peripheral Bus Stop mode mechanism. When a request is made to enter External Stop mode, it acknowledges the request and completes the transfer that is in progress. When the module reaches the frame boundary, it signals that the protocol clock to the module may be shut off.

NOTE

In master mode, if a serial transfer is in progress, then this module waits until it reaches the frame boundary before it is ready to have its clocks shut off. In slave mode, this module waits until its chip select input signal goes high before it is ready to have its clocks shut off. It should be noted that the CS0 pad must be correctly configured to allow it to return to high when no transfer is occurring. If the pad is tied low then this could prevent the module from being correctly disabled.

44.2.4.5 Debug Mode

Debug mode is used for system development and debugging. The MCR[FRZ] bit controls module behavior in the Debug mode:

- If the bit is set, the module stops all serial transfers, when the chip is in debug mode.
- If the bit is cleared, the chip debug mode has no effect on the module.

44.3 Module signal descriptions

This table describes the signals on the boundary of the module that may connect off chip (in alphabetical order).

Table 44-4. Module signal descriptions

Signal	Master mode	Slave mode	I/O
PCS0/$\overline{SS}$	Peripheral Chip Select 0 (O)	Slave Select (I)	I/O
PCS[1:3]	Peripheral Chip Selects 1–3	(Unused)	O
PCS4	Peripheral Chip Select 4		O
PCS5/ $\overline{PCSS}$	Peripheral Chip Select 5 /Peripheral Chip Select Strobe	(Unused)	O
SCK	Serial Clock (O)	Serial Clock (I)	I/O
SIN	Serial Data In	Serial Data In	I
SOUT	Serial Data Out	Serial Data Out	O

44.3.1 PCS0/$\overline{SS}$—Peripheral Chip Select/Slave Select

Master mode: Peripheral Chip Select 0 (O)—Selects an SPI slave to receive data transmitted from the module.

Slave mode: Slave Select (I)—Selects the module to receive data transmitted from an SPI master.

NOTE

Do not tie the SPI slave select pin to ground. Otherwise, SPI cannot function properly.

44.3.2 PCS1–PCS3—Peripheral Chip Selects 1–3

Master mode: Peripheral Chip Selects 1–3 (O)—Select an SPI slave to receive data transmitted by the module.

Slave mode: Unused

44.3.3 PCS4—Peripheral Chip Select 4

Master mode: Peripheral Chip Select 4 (O)—Selects an SPI slave to receive data transmitted by the module.

Slave mode: Unused

44.3.4 PCS5/$\overline{PCSS}$—Peripheral Chip Select 5/Peripheral Chip Select Strobe

Master mode:

- Peripheral Chip Select 5 (O)—Used only when the peripheral-chip-select strobe is disabled (MCR[PCSSE]). Selects an SPI slave to receive data transmitted by the module.
- Peripheral Chip Select Strobe (O)—Used only when the peripheral-chip-select strobe is enabled (MCR[PCSSE]). Strobes an off-module peripheral-chip-select demultiplexer, which decodes the module's PCS signals other than PCS5, preventing glitches on the demultiplexer outputs.

Slave mode: Unused

44.3.5 SCK—Serial Clock

Master mode: Serial Clock (O)—Supplies a clock signal from the module to SPI slaves.

Slave mode: Serial Clock (I)—Supplies a clock signal to the module from an SPI master.

44.3.6 SIN—Serial Input

Master mode: Serial Input (I)—Receives serial data.

Slave mode: Serial Input (I)—Receives serial data.

44.3.7 SOUT—Serial Output

Master mode: Serial Output (O)—Transmits serial data.

Slave mode: Serial Output (O)—Transmits serial data.

44.4 Memory Map/Register Definition

Register accesses to memory addresses that are reserved or undefined result in a transfer error. Any Write access to the POPR and RXFRn also results in a transfer error.

NOTE

f_P and f_{sys} are used interchangeably in this chapter.

SPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_C000	Module Configuration Register (SPI0_MCR)	32	R/W	0000_4001h	44.4.1/1174
4002_C008	Transfer Count Register (SPI0_TCR)	32	R/W	0000_0000h	44.4.2/1177
4002_C00C	Clock and Transfer Attributes Register (In Master Mode) (SPI0_CTAR0)	32	R/W	7800_0000h	44.4.3/1178
4002_C00C	Clock and Transfer Attributes Register (In Slave Mode) (SPI0_CTAR0_SLAVE)	32	R/W	7800_0000h	44.4.4/1183
4002_C010	Clock and Transfer Attributes Register (In Master Mode) (SPI0_CTAR1)	32	R/W	7800_0000h	44.4.3/1178
4002_C02C	Status Register (SPI0_SR)	32	R/W	0200_0000h	44.4.5/1184

Table continues on the next page...

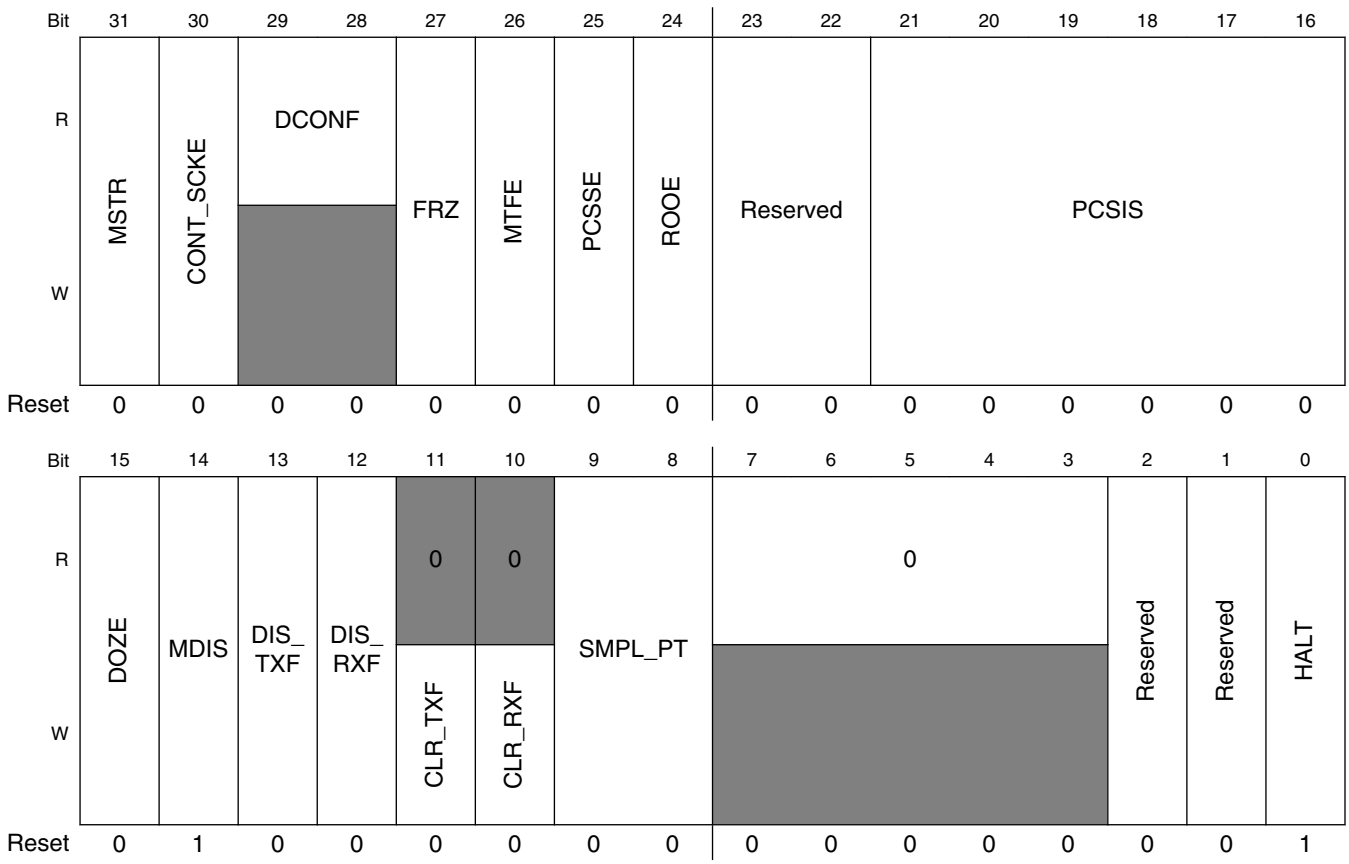
SPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_C030	DMA/Interrupt Request Select and Enable Register (SPI0_RSER)	32	R/W	0000_0000h	44.4.6/1187
4002_C034	PUSH TX FIFO Register In Master Mode (SPI0_PUSHR)	32	R/W	0000_0000h	44.4.7/1189
4002_C034	PUSH TX FIFO Register In Slave Mode (SPI0_PUSHR_SLAVE)	32	R/W	0000_0000h	44.4.8/1191
4002_C038	POP RX FIFO Register (SPI0_POPR)	32	R	0000_0000h	44.4.9/1192
4002_C03C	Transmit FIFO Registers (SPI0_TXFR0)	32	R	0000_0000h	44.4.10/ 1192
4002_C040	Transmit FIFO Registers (SPI0_TXFR1)	32	R	0000_0000h	44.4.10/ 1192
4002_C044	Transmit FIFO Registers (SPI0_TXFR2)	32	R	0000_0000h	44.4.10/ 1192
4002_C048	Transmit FIFO Registers (SPI0_TXFR3)	32	R	0000_0000h	44.4.10/ 1192
4002_C07C	Receive FIFO Registers (SPI0_RXFR0)	32	R	0000_0000h	44.4.11/ 1193
4002_C080	Receive FIFO Registers (SPI0_RXFR1)	32	R	0000_0000h	44.4.11/ 1193
4002_C084	Receive FIFO Registers (SPI0_RXFR2)	32	R	0000_0000h	44.4.11/ 1193
4002_C088	Receive FIFO Registers (SPI0_RXFR3)	32	R	0000_0000h	44.4.11/ 1193

44.4.1 Module Configuration Register (SPIx_MCR)

Contains bits to configure various attributes associated with the module operations. The HALT and MDIS bits can be changed at any time, but the effect takes place only on the next frame boundary. Only the HALT and MDIS bits in the MCR can be changed, while the module is in the Running state.

Address: 4002_C000h base + 0h offset = 4002_C000h



SPIx_MCR field descriptions

Field	Description
31 MSTR	Master/Slave Mode Select Enables either Master mode (if supported) or Slave mode (if supported) operation. 0 Enables Slave mode 1 Enables Master mode
30 CONT_SCKE	Continuous SCK Enable Enables the Serial Communication Clock (SCK) to run continuously.

Table continues on the next page...

SPIx_MCR field descriptions (continued)

Field	Description
	0 Continuous SCK disabled. 1 Continuous SCK enabled.
29–28 DCONF	SPI Configuration. Selects among the different configurations of the module. 00 SPI 01 Reserved 10 Reserved 11 Reserved
27 FRZ	Freeze Enables transfers to be stopped on the next frame boundary when the device enters Debug mode. 0 Do not halt serial transfers in Debug mode. 1 Halt serial transfers in Debug mode.
26 MTFE	Modified Transfer Format Enable Enables a modified transfer format to be used. **NOTE:** When MTFE=1 with continuous SCK enabled (MCR [CONT_SCKE] =1) in master mode, configure CTAR[LSBFE]=0 for correct operations while receiving unequal length frames. If PUSHR[CONT] is also set for back to back frame transfer, also configure the frame size of the first frame as less than or equal to the frame size of the next frame. In this scenario, make sure that for all received frames, the bits are read equal to their respective frame sizes and any extra bits during POP operation are masked. 0 Modified SPI transfer format disabled. 1 Modified SPI transfer format enabled.
25 PCSSE	Peripheral Chip Select Strobe Enable Enables the PCS5/ $\overline{\text{PCSS}}$ to operate as a PCS Strobe output signal. 0 PCS5/ $\overline{\text{PCSS}}$ is used as the Peripheral Chip Select[5] signal. 1 PCS5/ $\overline{\text{PCSS}}$ is used as an active-low PCS Strobe signal.
24 ROOE	Receive FIFO Overflow Overwrite Enable In the RX FIFO overflow condition, configures the module to ignore the incoming serial data or overwrite existing data. If the RX FIFO is full and new data is received, the data from the transfer, generating the overflow, is ignored or shifted into the shift register. 0 Incoming data is ignored. 1 Incoming data is shifted into the shift register.
23–22 Reserved	Always write the reset value to this field. This field is reserved.
21–16 PCSIS	Peripheral Chip Select x Inactive State

Table continues on the next page...

SPIx_MCR field descriptions (continued)

Field	Description
	Determines the inactive state of PCSx. Refer to the chip-specific SPI information for the number of PCS signals used in this chip. **Table 44-5. PCSIS signal and bit position mapping** Signal / Bit position: PCSIS0 16; PCSIS1 17; PCSIS2 18; PCSIS3 19; PCSIS4 20; PCSIS5 21 **NOTE:** The effect of this bit only takes place when module is enabled. Ensure that this bit is configured correctly before enabling the SPI interface. 0 The inactive state of PCSx is low. 1 The inactive state of PCSx is high.
15 DOZE	Doze Enable Provides support for an externally controlled Doze mode power-saving mechanism. 0 Doze mode has no effect on the module. 1 Doze mode disables the module.
14 MDIS	Module Disable Allows the clock to be stopped to the non-memory mapped logic in the module effectively putting it in a software-controlled power-saving state. The reset value of the MDIS bit is parameterized, with a default reset value of 1. When the module is used in Slave Mode, it is recommended to leave this bit 0, because a slave doesn't have control over master transactions. 0 Enables the module clocks. 1 Allows external logic to disable the module clocks.
13 DIS_TXF	Disable Transmit FIFO When the TX FIFO is disabled, the transmit part of the module operates as a simplified double-buffered SPI. This bit can be written only when the MDIS bit is cleared. 0 TX FIFO is enabled. 1 TX FIFO is disabled.
12 DIS_RXF	Disable Receive FIFO When the RX FIFO is disabled, the receive part of the module operates as a simplified double-buffered SPI. This bit can only be written when the MDIS bit is cleared. 0 RX FIFO is enabled. 1 RX FIFO is disabled.
11 CLR_TXF	Clear TX FIFO

Table continues on the next page...

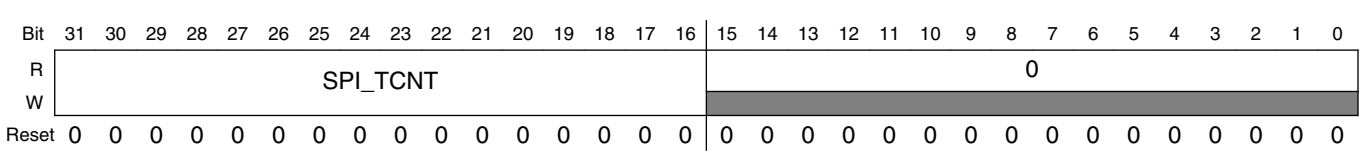
SPIx_MCR field descriptions (continued)

Field	Description
	Flushes the TX FIFO. Writing a 1 to CLR_TXF clears the TX FIFO Counter. The CLR_TXF bit is always read as zero. 0 Do not clear the TX FIFO counter. 1 Clear the TX FIFO counter.
10 CLR_RXF	Clear RX FIFO Flushes the RX FIFO. Writing a 1 to CLR_RXF clears the RX Counter. The CLR_RXF bit is always read as zero. 0 Do not clear the RX FIFO counter. 1 Clear the RX FIFO counter.
9–8 SMPL_PT	Sample Point Controls when the module master samples SIN in Modified Transfer Format. This field is valid only when CPHA bit in CTARn[CPHA] is 0. 00 0 protocol clock cycles between SCK edge and SIN sample 01 1 protocol clock cycle between SCK edge and SIN sample 10 2 protocol clock cycles between SCK edge and SIN sample 11 Reserved
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved.
1 Reserved	This field is reserved.
0 HALT	Halt The HALT bit starts and stops frame transfers. See Start and Stop of Module transfers 0 Start transfers. 1 Stop transfers.

44.4.2 Transfer Count Register (SPIx_TCR)

TCR contains a counter that indicates the number of SPI transfers made. The transfer counter is intended to assist in queue management. Do not write the TCR when the module is in the Running state.

Address: 4002_C000h base + 8h offset = 4002_C008h



SPIx_TCR field descriptions

Field	Description
31–16 SPI_TCNT	SPI Transfer Counter Counts the number of SPI transfers the module makes. The SPI_TCNT field increments every time the last bit of an SPI frame is transmitted. A value written to SPI_TCNT presets the counter to that value. SPI_TCNT is reset to zero at the beginning of the frame when the CTCNT field is set in the executing SPI command. The Transfer Counter wraps around; incrementing the counter past 65535 resets the counter to zero.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

44.4.3 Clock and Transfer Attributes Register (In Master Mode) (SPIx_CTAR*n*)

CTAR registers are used to define different transfer attributes. Do not write to the CTAR registers while the module is in the Running state.

In Master mode, the CTAR registers define combinations of transfer attributes such as frame size, clock phase and polarity, data bit ordering, baud rate, and various delays. In slave mode, a subset of the bitfields in CTAR0 are used to set the slave transfer attributes.

When the module is configured as a SPI master, the CTAS field in the command portion of the TX FIFO entry selects which of the CTAR registers is used. When the module is configured as an SPI bus slave, it uses the CTAR0 register.

Address: 4002_C000h base + Ch offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DBR	FMSZ				CPOL	CPHA	LSBFE	PCSSCK		PASC		PDT		PBR	
Reset	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSSCK				ASC				DT				BR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPIx_CTAR*n* field descriptions

Field	Description
31 DBR	Double Baud Rate

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>Doubles the effective baud rate of the Serial Communications Clock (SCK). This field is used only in master mode. It effectively halves the Baud Rate division ratio, supporting faster frequencies, and odd division ratios for the Serial Communications Clock (SCK). When the DBR bit is set, the duty cycle of the Serial Communications Clock (SCK) depends on the value in the Baud Rate Prescaler and the Clock Phase bit as listed in the following table. See the BR field description for details on how to compute the baud rate.

Table 44-6. SPI SCK Duty Cycle
<table><tr><th>DBR</th><th>CPHA</th><th>PBR</th><th>SCK Duty Cycle</th></tr><tr><td>0</td><td>any</td><td>any</td><td>50/50</td></tr><tr><td>1</td><td>0</td><td>00</td><td>50/50</td></tr><tr><td>1</td><td>0</td><td>01</td><td>33/66</td></tr><tr><td>1</td><td>0</td><td>10</td><td>40/60</td></tr><tr><td>1</td><td>0</td><td>11</td><td>43/57</td></tr><tr><td>1</td><td>1</td><td>00</td><td>50/50</td></tr><tr><td>1</td><td>1</td><td>01</td><td>66/33</td></tr><tr><td>1</td><td>1</td><td>10</td><td>60/40</td></tr><tr><td>1</td><td>1</td><td>11</td><td>57/43</td></tr></table>
0 The baud rate is computed normally with a 50/50 duty cycle.
1 The baud rate is doubled with the duty cycle depending on the Baud Rate Prescaler.</td></tr>
<tr><td>30–27
FMSZ</td><td>Frame Size

The number of bits transferred per frame is equal to the FMSZ value plus 1. Regardless of the transmission mode, the minimum valid frame size value is 4.</td></tr>
<tr><td>26
CPOL</td><td>Clock Polarity

Selects the inactive state of the Serial Communications Clock (SCK). This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock polarities. When the Continuous Selection Format is selected, switching between clock polarities without stopping the module can cause errors in the transfer due to the peripheral device interpreting the switch of clock polarity as a valid clock edge.

NOTE: In case of Continuous SCK mode, when the module goes in low power mode(disabled), inactive state of SCK is not guaranteed.

0 The inactive state value of SCK is low.
1 The inactive state value of SCK is high.</td></tr>
<tr><td>25
CPHA</td><td>Clock Phase

Selects which edge of SCK causes data to change and which edge causes data to be captured. This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the bit value is ignored and the transfers are done as if the CPHA bit is set to 1.

0 Data is captured on the leading edge of SCK and changed on the following edge.
1 Data is changed on the leading edge of SCK and captured on the following edge.</td></tr>
</table>

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

Field	Description
24 LSBFE	LSB First Specifies whether the LSB or MSB of the frame is transferred first. 0 Data is transferred MSB first. 1 Data is transferred LSB first.
23–22 PCSSCK	PCS to SCK Delay Prescaler Selects the prescaler value for the delay between assertion of PCS and the first edge of the SCK. See the CSSCK field description for information on how to compute the PCS to SCK Delay. Refer PCS to SCK Delay (t_{CSC}) for more details. 00 PCS to SCK Prescaler value is 1. 01 PCS to SCK Prescaler value is 3. 10 PCS to SCK Prescaler value is 5. 11 PCS to SCK Prescaler value is 7.
21–20 PASC	After SCK Delay Prescaler Selects the prescaler value for the delay between the last edge of SCK and the negation of PCS. See the ASC field description for information on how to compute the After SCK Delay. Refer After SCK Delay (t_{ASC}) for more details. 00 Delay after Transfer Prescaler value is 1. 01 Delay after Transfer Prescaler value is 3. 10 Delay after Transfer Prescaler value is 5. 11 Delay after Transfer Prescaler value is 7.
19–18 PDT	Delay after Transfer Prescaler Selects the prescaler value for the delay between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame. The PDT field is only used in master mode. See the DT field description for details on how to compute the Delay after Transfer. Refer Delay after Transfer (t_{DT}) for more details. 00 Delay after Transfer Prescaler value is 1. 01 Delay after Transfer Prescaler value is 3. 10 Delay after Transfer Prescaler value is 5. 11 Delay after Transfer Prescaler value is 7.
17–16 PBR	Baud Rate Prescaler Selects the prescaler value for the baud rate. This field is used only in master mode. The baud rate is the frequency of the SCK. The protocol clock is divided by the prescaler value before the baud rate selection takes place. See the BR field description for details on how to compute the baud rate. 00 Baud Rate Prescaler value is 2. 01 Baud Rate Prescaler value is 3. 10 Baud Rate Prescaler value is 5. 11 Baud Rate Prescaler value is 7.
15–12 CSSCK	PCS to SCK Delay Scaler Selects the scaler value for the PCS to SCK delay. This field is used only in master mode. The PCS to SCK Delay is the delay between the assertion of PCS and the first edge of the SCK. The delay is a multiple of the protocol clock period, and it is computed according to the following equation:

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>t $_{CSC}$ = (1/f_P) x PCSSCK x CSSCK.
The following table lists the delay scaler values.
Table 44-7. Delay Scaler Encoding<table><tr><th>Field Value</th><th>Delay Scaler Value</th></tr><tr><td>0000</td><td>2</td></tr><tr><td>0001</td><td>4</td></tr><tr><td>0010</td><td>8</td></tr><tr><td>0011</td><td>16</td></tr><tr><td>0100</td><td>32</td></tr><tr><td>0101</td><td>64</td></tr><tr><td>0110</td><td>128</td></tr><tr><td>0111</td><td>256</td></tr><tr><td>1000</td><td>512</td></tr><tr><td>1001</td><td>1024</td></tr><tr><td>1010</td><td>2048</td></tr><tr><td>1011</td><td>4096</td></tr><tr><td>1100</td><td>8192</td></tr><tr><td>1101</td><td>16384</td></tr><tr><td>1110</td><td>32768</td></tr><tr><td>1111</td><td>65536</td></tr></table>Refer PCS to SCK Delay (t_{CSC}) for more details.</td></tr>
<tr><td>11–8
ASC</td><td>After SCK Delay Scaler
Selects the scaler value for the After SCK Delay. This field is used only in master mode. The After SCK Delay is the delay between the last edge of SCK and the negation of PCS. The delay is a multiple of the protocol clock period, and it is computed according to the following equation:
t $_{ASC}$ = (1/f_P) x PASC x ASC
See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values. Refer After SCK Delay (t_{ASC}) for more details.</td></tr>
<tr><td>7–4
DT</td><td>Delay After Transfer Scaler
Selects the Delay after Transfer Scaler. This field is used only in master mode. The Delay after Transfer is the time between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame.
In the Continuous Serial Communications Clock operation, the DT value is fixed to one SCK clock period, The Delay after Transfer is a multiple of the protocol clock period, and it is computed according to the following equation:
t_{DT} = (1/f_P) x PDT x DT
See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values.</td></tr>
<tr><td>BR</td><td>Baud Rate Scaler</td></tr>
</table>

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

Field	Description
	Selects the scaler value for the baud rate. This field is used only in master mode. The prescaled protocol clock is divided by the Baud Rate Scaler to generate the frequency of the SCK. The baud rate is computed according to the following equation: SCK baud rate = (f_P /PBR) x [(1+DBR)/BR] The following table lists the baud rate scaler values.

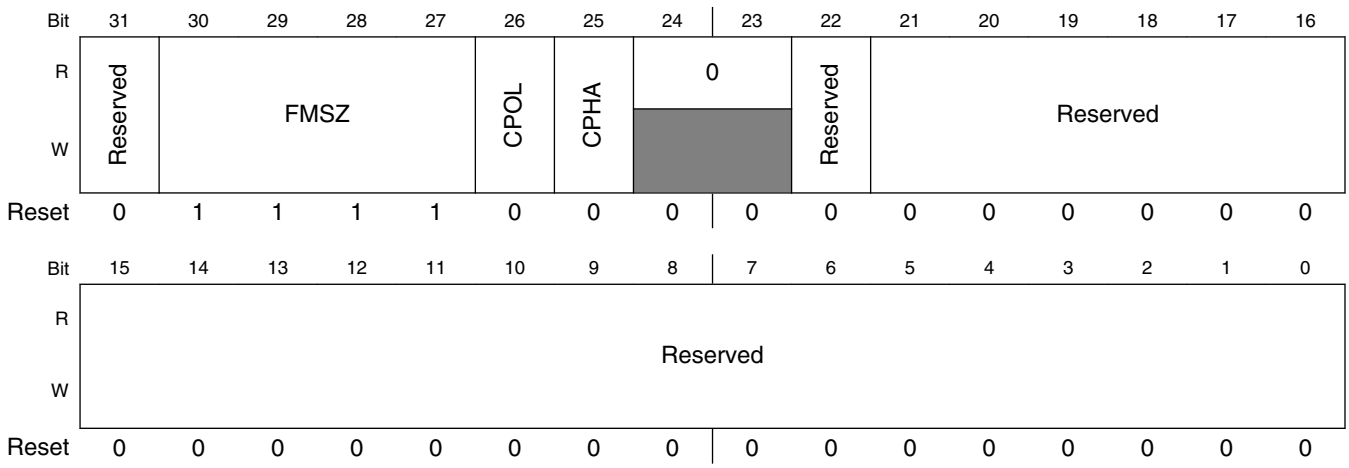
Table 44-8. Baud Rate Scaler

CTARn[BR]	Baud Rate Scaler Value
0000	2
0001	4
0010	6
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256
1001	512
1010	1024
1011	2048
1100	4096
1101	8192
1110	16384
1111	32768

44.4.4 Clock and Transfer Attributes Register (In Slave Mode) (SPI*x*_CTAR*n*_SLAVE)

When the module is configured as an SPI bus slave, the CTAR0 register is used.

Address: 4002_C000h base + Ch offset + (0d × i), where i=0d to 0d



SPI*x*_CTAR*n*_SLAVE field descriptions

Field	Description
31 Reserved	Always write the reset value to this field. This field is reserved.
30–27 FMSZ	Frame Size The number of bits transferred per frame is equal to the FMSZ field value plus 1. Note that the minimum valid value of frame size is 4.
26 CPOL	Clock Polarity Selects the inactive state of the Serial Communications Clock (SCK). **NOTE:** In case of Continuous SCK mode, when the module goes in low power mode(disabled), inactive state of SCK is not guaranteed. 0 The inactive state value of SCK is low. 1 The inactive state value of SCK is high.
25 CPHA	Clock Phase Selects which edge of SCK causes data to change and which edge causes data to be captured. This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the bit value is ignored and the transfers are done as if the CPHA bit is set to 1. 0 Data is captured on the leading edge of SCK and changed on the following edge. 1 Data is changed on the leading edge of SCK and captured on the following edge.

Table continues on the next page...

SPIx_CTAR*n*_SLAVE field descriptions (continued)

Field	Description
24–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 Reserved	This field is reserved.
Reserved	This field is reserved.

44.4.5 Status Register (SPIx_SR)

SR contains status and flag bits. The bits reflect the status of the module and indicate the occurrence of events that can generate interrupt or DMA requests. Software can clear flag bits in the SR by writing a 1 to them. Writing a 0 to a flag bit has no effect. This register may not be writable in Module Disable mode due to the use of power saving mechanisms.

Address: 4002_C000h base + 2Ch offset = 4002_C02Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCF	TXRXS	0	EOQF	TFUF	0	TFFF	0	0	0	0	0	RFOF	0	RFDF	0
W	w1c			w1c	w1c		w1c						w1c		w1c	
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXCTR				TXNXTPTR				RXCTR				POPNXTPTR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPIx_SR field descriptions

Field	Description
31 TCF	Transfer Complete Flag Indicates that all bits in a frame have been shifted out. TCF remains set until it is cleared by writing a 1 to it. 0 Transfer not complete. 1 Transfer complete.
30 TXRXS	TX and RX Status Reflects the run status of the module. 0 Transmit and receive operations are disabled (The module is in Stopped state). 1 Transmit and receive operations are enabled (The module is in Running state).
29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 EOQF	End of Queue Flag Indicates that the last entry in a queue has been transmitted when the module is in Master mode. The EOQF bit is set when the TX FIFO entry has the EOQ bit set in the command halfword and the end of the transfer is reached. The EOQF bit remains set until cleared by writing a 1 to it. When the EOQF bit is set, the TXRXS bit is automatically cleared. 0 EOQ is not set in the executing command. 1 EOQ is set in the executing SPI command.
27 TFUF	Transmit FIFO Underflow Flag Indicates an underflow condition in the TX FIFO. The transmit underflow condition is detected only for SPI blocks operating in Slave mode and SPI configuration. TFUF is set when the TX FIFO of the module operating in SPI Slave mode is empty and an external SPI master initiates a transfer. The TFUF bit remains set until cleared by writing 1 to it. 0 No TX FIFO underflow. 1 TX FIFO underflow has occurred.
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 TFFF	Transmit FIFO Fill Flag Indicates whether there is an available location to be filled in the FIFO. Either a DMA request or an interrupt indication can be used to add another entry to the FIFO. Note that this bit is set if at least one location is free in the FIFO. The TFFF bit can be cleared by writing 1 to it or by acknowledgement from the DMA controller to the TX FIFO full request, when the TX FIFO is full. If FIFO is filled manually, and not by DMA see Transmit FIFO Fill Interrupt or DMA Request **NOTE:** The reset value of this bit is 0 when the module is disabled,(MCR[MDIS]=1). 0 TX FIFO is full. 1 TX FIFO is not full.
24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SPIx_SR field descriptions (continued)

Field	Description
22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19 RFOF	Receive FIFO Overflow Flag Indicates an overflow condition in the RX FIFO. The field is set when the RX FIFO and shift register are full and a transfer is initiated. The bit remains set until it is cleared by writing a 1 to it. 0 No Rx FIFO overflow. 1 Rx FIFO overflow has occurred.
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 RFDF	Receive FIFO Drain Flag Indicates whether there is an available location to be drained from the FIFO. Either a DMA request or an interrupt indication can be used to read from the FIFO. Note that this bit is set if at least one location can be read from the FIFO. The RFDF bit can be cleared by writing 1 to it or by acknowledgement from the DMA controller when the RX FIFO is empty. 0 RX FIFO is empty. 1 RX FIFO is not empty.
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–12 TXCTR	TX FIFO Counter Indicates the number of valid entries in the TX FIFO. The TXCTR is incremented every time the PUSHR is written. The TXCTR is decremented every time an SPI command is executed and the SPI data is transferred to the shift register.
11–8 TXNXTPTR	Transmit Next Pointer Indicates which TX FIFO entry is transmitted during the next transfer. The TXNXTPTR field is updated every time SPI data is transferred from the TX FIFO to the shift register.
7–4 RXCTR	RX FIFO Counter Indicates the number of entries in the RX FIFO. The RXCTR is decremented every time the POPR is read. The RXCTR is incremented every time data is transferred from the shift register to the RX FIFO.
POPNXTPTR	Pop Next Pointer Contains a pointer to the RX FIFO entry to be returned when the POPR is read. The POPNXTPTR is updated when the POPR is read.

44.4.6 DMA/Interrupt Request Select and Enable Register (SPIx_RSER)

RSER controls DMA and interrupt requests. Do not write to the RSER while the module is in the Running state.

Address: 4002_C000h base + 30h offset = 4002_C030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCF_RE	Reserved	Reserved	EOQF_RE	TFUF_RE	Reserved	TFFF_RE	TFFF_DIRS	Reserved	Reserved	Reserved	Reserved	RFOF_RE	Reserved	RFDF_RE	RFDF_DIRS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

SPIx_RSER field descriptions

Field	Description
31 TCF_RE	Transmission Complete Request Enable Enables TCF flag in the SR to generate an interrupt request. 0 TCF interrupt requests are disabled. 1 TCF interrupt requests are enabled.
30 Reserved	Always write the reset value to this field. This field is reserved.
29 Reserved	Always write the reset value to this field. This field is reserved.
28 EOQF_RE	Finished Request Enable Enables the EOQF flag in the SR to generate an interrupt request. 0 EOQF interrupt requests are disabled. 1 EOQF interrupt requests are enabled.
27 TFUF_RE	Transmit FIFO Underflow Request Enable Enables the TFUF flag in the SR to generate an interrupt request.

Table continues on the next page...

SPIx_RSER field descriptions (continued)

Field	Description
	0 TFUF interrupt requests are disabled. 1 TFUF interrupt requests are enabled.
26 Reserved	Always write the reset value to this field. This field is reserved.
25 TFFF_RE	Transmit FIFO Fill Request Enable Enables the TFFF flag in the SR to generate a request. The TFFF_DIRS bit selects between generating an interrupt request or a DMA request. 0 TFFF interrupts or DMA requests are disabled. 1 TFFF interrupts or DMA requests are enabled.
24 TFFF_DIRS	Transmit FIFO Fill DMA or Interrupt Request Select Selects between generating a DMA request or an interrupt request. When SR[TFFF] and RSER[TFFF_RE] are set, this field selects between generating an interrupt request or a DMA request. 0 TFFF flag generates interrupt requests. 1 TFFF flag generates DMA requests.
23 Reserved	Always write the reset value to this field. This field is reserved.
22 Reserved	Always write the reset value to this field. This field is reserved.
21 Reserved	Always write the reset value to this field. This field is reserved.
20 Reserved	Always write the reset value to this field. This field is reserved.
19 RFOF_RE	Receive FIFO Overflow Request Enable Enables the RFOF flag in the SR to generate an interrupt request. 0 RFOF interrupt requests are disabled. 1 RFOF interrupt requests are enabled.
18 Reserved	Always write the reset value to this field. This field is reserved.
17 RFDF_RE	Receive FIFO Drain Request Enable Enables the RFDF flag in the SR to generate a request. The RFDF_DIRS bit selects between generating an interrupt request or a DMA request. 0 RFDF interrupt or DMA requests are disabled. 1 RFDF interrupt or DMA requests are enabled.
16 RFDF_DIRS	Receive FIFO Drain DMA or Interrupt Request Select

Table continues on the next page...

SPIx_RSER field descriptions (continued)

Field	Description
	Selects between generating a DMA request or an interrupt request. When the RFDF flag bit in the SR is set, and the RFDF_RE bit in the RSER is set, the RFDF_DIRS bit selects between generating an interrupt request or a DMA request. 0 Interrupt request. 1 DMA request.
15 Reserved	Always write the reset value to this field. This field is reserved.
14 Reserved	Always write the reset value to this field. This field is reserved.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

44.4.7 PUSH TX FIFO Register In Master Mode (SPIx_PUSHR)

Specifies data to be transferred to the TX FIFO and CMD FIFO. User must write 16-bits data into TXDATA field. An 8- or 16-bit write access to the TXDATA field transfers 16 bits of data bus to the TX FIFO. A write access to the command fields transfers the 16 bits of command information to the CMD FIFO. In Master mode, the register transfers 16 bits of data to the TX FIFO and 16 bits of command information to the CMD FIFO.

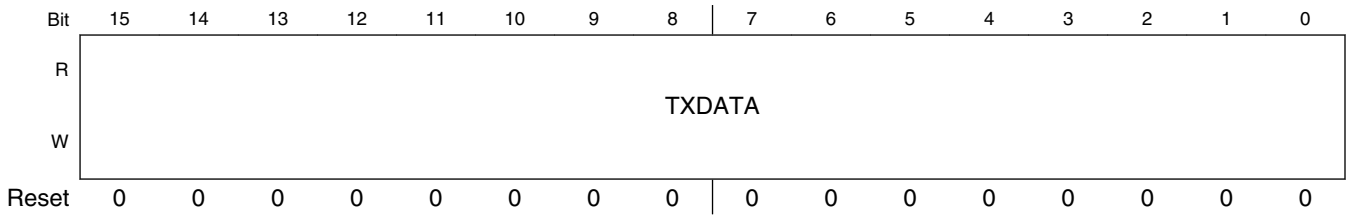
The TX FIFO and CMD FIFO must be filled simultaneously. In other words, you must perform write accesses to both the data and command fields for every PUSHR operation. Because both the TX FIFO and CMD FIFO are written to and read from simultaneously, they behave as a single 32 bit FIFO.

A read access of PUSHR returns the topmost TX FIFO and CMD FIFO entries concatenated.

When the module is disabled, writing to this register does not update the FIFO. Therefore, any reads performed while the module is disabled return the last PUSHR write performed while the module was still enabled.

Address: 4002_C000h base + 34h offset = 4002_C034h

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



SPIx_PUSHR field descriptions

Field	Description
31 CONT	Continuous Peripheral Chip Select Enable Selects a continuous selection format. The bit is used in SPI Master mode. The bit enables the selected PCS signals to remain asserted between transfers. 0 Return PCSn signals to their inactive state between transfers. 1 Keep PCSn signals asserted between transfers.
30–28 CTAS	Clock and Transfer Attributes Select Selects which CTAR to use in master mode to specify the transfer attributes for the associated SPI frame. In SPI Slave mode, CTAR0 is used. See the chip specific section for details to determine how many CTARs this device has. You should not program a value in this field for a register that is not present. 000 CTAR0 001 CTAR1 010 Reserved 011 Reserved 100 Reserved 101 Reserved 110 Reserved 111 Reserved
27 EOQ	End Of Queue Host software uses this bit to signal to the module that the current SPI transfer is the last in a queue. At the end of the transfer, the EOQF bit in the SR is set. 0 The SPI data is not the last data to transfer. 1 The SPI data is the last data to transfer.
26 CTCNT	Clear Transfer Counter Clears the TCNT field in the TCR register. The TCNT field is cleared before the module starts transmitting the current SPI frame. 0 Do not clear the TCR[TCNT] field. 1 Clear the TCR[TCNT] field.
25–24 Reserved	Always write the reset value to this field. This field is reserved.
23–22 Reserved	Always write the reset value to this field. This field is reserved.

Table continues on the next page...

SPIx_PUSHR field descriptions (continued)

Field	Description
21–16 PCS	Select which PCS signals are to be asserted for the transfer. Refer to the chip-specific SPI information for the number of PCS signals used in this chip. **Table 44-9. PCS signal and bit position mapping** Signal / Bit position: PCS0 — 16 PCS1 — 17 PCS2 — 18 PCS3 — 19 PCS4 — 20 PCS5 — 21 0 Negate the PCS[x] signal. 1 Assert the PCS[x] signal.
TXDATA	Transmit Data Holds SPI data to be transferred according to the associated SPI command.

44.4.8 PUSH TX FIFO Register In Slave Mode (SPIx_PUSHR_SLAVE)

Specifies data to be transferred to the TX FIFO in slave mode. An 8- or 16-bit write access to PUSHR transfers the 16-bit TXDATA field to the TX FIFO.

Address: 4002_C000h base + 34h offset = 4002_C034h

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

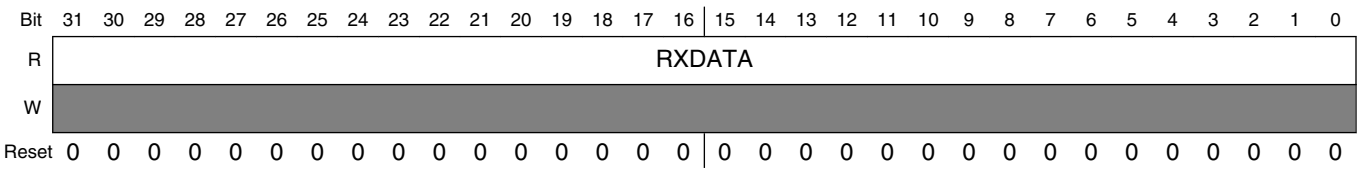
SPIx_PUSHR_SLAVE field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TXDATA	Transmit Data Holds SPI data to be transferred according to the associated SPI command.

44.4.9 POP RX FIFO Register (SPIx_POPR)

POPR is used to read the RX FIFO. Eight- or sixteen-bit read accesses to the POPR have the same effect on the RX FIFO as 32-bit read accesses. A write to this register will generate a Transfer Error.

Address: 4002_C000h base + 38h offset = 4002_C038h



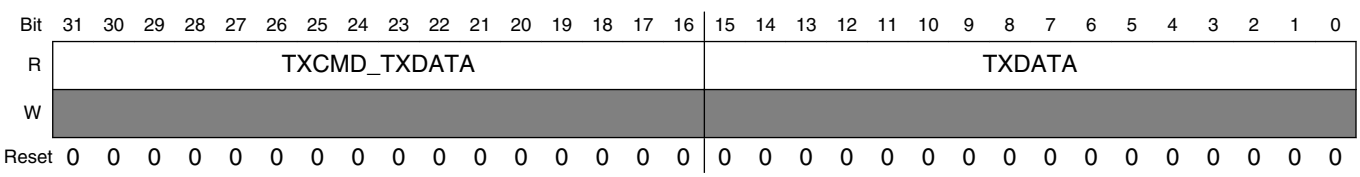
SPIx_POPR field descriptions

Field	Description
RXDATA	Received Data Contains the SPI data from the RX FIFO entry to which the Pop Next Data Pointer points.

44.4.10 Transmit FIFO Registers (SPIx_TXFR*n*)

TXFRn registers provide visibility into the TX FIFO for debugging purposes. Each register is an entry in the TX FIFO. The registers are read-only and cannot be modified. Reading the TXFRx registers does not alter the state of the TX FIFO.

Address: 4002_C000h base + 3Ch offset + (4d × i), where i=0d to 3d



SPIx_TXFR*n* field descriptions

Field	Description
31–16 TXCMD_ TXDATA	Transmit Command or Transmit Data In Master mode the TXCMD field contains the command that sets the transfer attributes for the SPI data. In Slave mode, this field is reserved.
TXDATA	Transmit Data Contains the SPI data to be shifted out.

44.4.11 Receive FIFO Registers (SPIx_RXFR*n*)

RXFRn provide visibility into the RX FIFO for debugging purposes. Each register is an entry in the RX FIFO. The RXFR registers are read-only. Reading the RXFRx registers does not alter the state of the RX FIFO.

Address: 4002_C000h base + 7Ch offset + (4d × i), where i=0d to 3d

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

SPIx_RXFR*n* field descriptions

Field	Description
RXDATA	Receive Data Contains the received SPI data.

44.5 Functional description

The module supports full-duplex, synchronous serial communications between chips and peripheral devices. The SPI configuration transfers data serially using a shift register and a selection of programmable transfer attributes.

The module has the following configuration

- The SPI Configuration in which the module operates as a basic SPI or a queued SPI.

The DCONF field in the Module Configuration Register (MCR) determines the module Configuration. SPI configuration is selected when DCONF within SPIx_MCR is 2b00.

The CTARn registers hold clock and transfer attributes. The SPI configuration allows to select which CTAR to use on a frame by frame basis by setting a field in the SPI command.

See Clock and Transfer Attributes Register (In Master Mode) (SPI_CTAR*n*) for information on the fields of CTAR registers.

Typical master to slave connections are shown in the following figure. When a data transfer operation is performed, data is serially shifted a predetermined number of bit positions. Because the modules are linked, data is exchanged between the master and the slave. The data that was in the master shift register is now in the shift register of the slave, and vice versa. At the end of a transfer, the Transfer Control Flag(TCF) bit in the Shift Register(SR) is set to indicate a completed frame transfer.

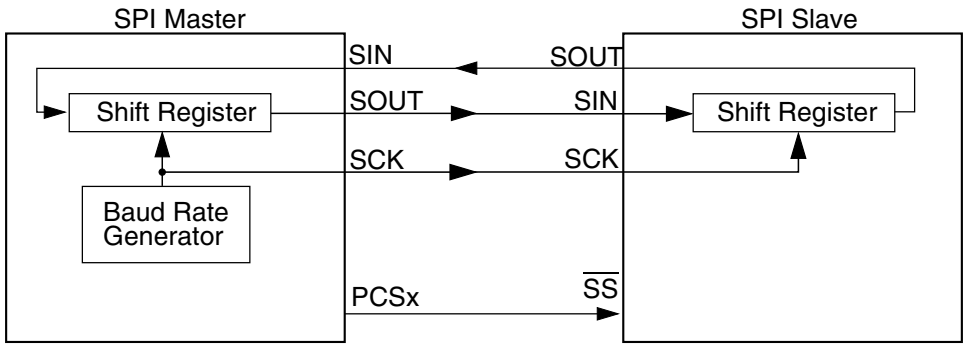


Figure 44-3. Serial protocol overview

Generally, more than one slave device can be connected to the module master. 6 Peripheral Chip Select (PCS) signals of the module masters can be used to select which of the slaves to communicate with. Refer to the chip specific section for details on the number of PCS signals used in this chip.

The SPI configuration shares transfer protocol and timing properties which are described independently of the configuration in Transfer formats. The transfer rate and delay settings are described in Module baud rate and clock delay generation.

44.5.1 Start and Stop of module transfers

The module has two operating states: Stopped and Running. Both the states are independent of it's configuration. The default state of the module is Stopped. In the Stopped state, no serial transfers are initiated in Master mode and no transfers are responded to in Slave mode. The Stopped state is also a safe state for writing the various configuration registers of the module without causing undetermined results. In the Running state serial transfers take place.

The TXRXS bit in the SR indicates the state of module. The bit is set if the module is in Running state.

The module starts or transitions to Running when all of the following conditions are true:

- SR[EOQF] bit is clear

- Chip is not in the Debug mode or the MCR[FRZ] bit is clear
- MCR[HALT] bit is clear

The module stops or transitions from Running to Stopped after the current frame when any one of the following conditions exist:

- SR[EOQF] bit is set
- Chip in the Debug mode and the MCR[FRZ] bit is set
- MCR[HALT] bit is set

State transitions from Running to Stopped occur on the next frame boundary if a transfer is in progress, or immediately if no transfers are in progress.

44.5.2 Serial Peripheral Interface SPI configuration

The SPI configuration transfers data serially using a shift register and a selection of programmable transfer attributes. The module is in SPI configuration when the DCONF field in the MCR is 2b00. The SPI frames can be 32 bits long. The host CPU or a DMA controller transfers the SPI data from the external to the module RAM queues to a TX FIFO buffer. The received data is stored in entries in the RX FIFO buffer. The host CPU or the DMA controller transfers the received data from the RX FIFO to memory external to the module. The operation of FIFO buffers is described in the following sections:

- Transmit First In First Out (TX FIFO) buffering mechanism
- Command First In First Out (CMD FIFO) Buffering Mechanism
- Receive First In First Out (RX FIFO) buffering mechanism

The interrupt and DMA request conditions are described in Interrupts/DMA requests.

The SPI configuration supports two block-specific modes—Master mode and Slave mode. In Master mode the module initiates and controls the transfer according to the fields of the executing SPI Command. In Slave mode, the module responds only to transfers initiated by a bus master external to it and the SPI command field space is reserved.

44.5.2.1 Master mode

In SPI Master mode, the module initiates the serial transfers by controlling the SCK and the PCS signals. The executing SPI Command determines which CTARs will be used to set the transfer attributes and which PCS signals to assert. The command field also

contains various bits that help with queue management and transfer protocol. See PUSH TX FIFO Register In Master Mode (SPI_PUSHR) for details on the SPI command fields. The data in the executing TX FIFO entry is loaded into the shift register and shifted out on the Serial Out (SOUT) pin. In SPI Master mode, each SPI frame to be transmitted has a command associated with it, allowing for transfer attribute control on a frame by frame basis.

44.5.2.2 Slave mode

In SPI Slave mode the module responds to transfers initiated by an SPI bus master. It does not initiate transfers. Certain transfer attributes such as clock polarity, clock phase, and frame size must be set for successful communication with an SPI master. The SPI Slave mode transfer attributes are set in the CTAR0. The data is shifted out with MSB first. Shifting out of LSB is not supported in this mode.

44.5.2.3 FIFO disable operation

The FIFO disable mechanisms allow SPI transfers without using the TX FIFO, CMD FIFO or RX FIFO. The module operates as a double-buffered simplified SPI when the FIFOs are disabled. The Transmit and Receive side of the FIFOs are disabled separately. Setting the MCR[DIS_TXF] bit disables the TX FIFO and CMD FIFO, and setting the MCR[DIS_RXF] bit disables the RX FIFO.

The FIFO disable mechanisms are transparent to the user and to host software. Transmit data and commands are written to the PUSHR and received data is read from the POPR.

When the TX FIFO and CMD FIFO are disabled:

- SR[TFFF], SR[TFUF] and SR[TXCTR] behave as if there is a one-entry FIFO
- The contents of TXFRs, SR[TXNXTPTR] are undefined

Similarly, when the RX FIFO is disabled, the RFDF, RFOF, and RXCTR fields in the SR behave as if there is a one-entry FIFO, but the contents of the RXFR registers and POPNXTPTR are undefined.

44.5.2.4 Transmit First In First Out (TX FIFO) buffering mechanism

The TX FIFO functions as a buffer of SPI data for transmission. The TX FIFO holds 4 words, each consisting of SPI data. The number of entries in the TX FIFO is device-specific. SPI data is added to the TX FIFO by writing to the Data Field of module PUSH FIFO Register (PUSHR). TX FIFO entries can only be removed from the TX FIFO by being shifted out or by flushing the TX FIFO.

The TX FIFO Counter field (TXCTR) in the module Status Register (SR) indicates the number of valid entries in the TX FIFO. The TXCTR is updated every time a 8- or 16-bit write takes place to PUSHR[TXDATA] or SPI data is transferred into the shift register from the TX FIFO.

The TXNXTPTR field indicates the TX FIFO Entry that will be transmitted during the next transfer. The TXNXTPTR field is incremented every time SPI data is transferred from the TX FIFO to the shift register. The maximum value of the field is equal to the maximum implemented TXFR number and it rolls over after reaching the maximum.

44.5.2.4.1 Filling the TX FIFO

Host software or other intelligent blocks can add (push) entries to the TX FIFO and CMD FIFO by writing to the PUSHR. When the TX FIFO is not full, the TX FIFO Fill Flag (TFFF) in the SR is set. The TFFF bit is cleared when TX FIFO is full and the DMA controller indicates that a write to PUSHR is complete. Writing a '1' to the TFFF bit also clears it. The TFFF can generate a DMA request or an interrupt request. See Transmit FIFO Fill Interrupt or DMA Request for details.

The module ignores attempts to push data to a full TX FIFO, and the state of the TX FIFO does not change and no error condition is indicated.

44.5.2.4.2 Draining the TX FIFO

The TX FIFO entries are removed (drained) by shifting SPI data out through the shift register. Entries are transferred from the TX FIFO to the shift register and shifted out as long as there are valid entries in the TX FIFO. Every time an entry is transferred from the TX FIFO to the shift register, the TX FIFO Counter decrements by one. At the end of a transfer, the TCF bit in the SR is set to indicate the completion of a transfer. The TX FIFO is flushed by writing a '1' to the CLR_TXF bit in MCR.

If an external bus master initiates a transfer with a module slave while the slave's TX FIFO is empty, the Transmit FIFO Underflow Flag (TFUF) in the slave's SR is set. See Transmit FIFO Underflow Interrupt Request for details.

44.5.2.5 Command First In First Out (CMD FIFO) Buffering Mechanism

The CMD FIFO functions as a buffer of SPI command used for SPI data transmission. The TX FIFO and CMD FIFO must be filled together, i.e. write enables should be given for both the Data and Command fields while performing a PUSHR operation.

The CMD FIFO holds 4 words, each representing SPI command fields. The number of entries in the CMD FIFO is device-specific. SPI Command is added to the CMD FIFO by writing to the command field of SPI PUSH FIFO Register (PUSHR). CMD FIFO entries can only be removed from the CMD FIFO by being shifted out (to help transmit SPI data) or by flushing the CMD FIFO.

Every CMD FIFO entry has a corresponding single TX FIFO entry attached to it because both these FIFO's are filled simultaneously.

44.5.2.6 Receive First In First Out (RX FIFO) buffering mechanism

The RX FIFO functions as a buffer for data received on the SIN pin. The RX FIFO holds 4 received SPI data frames. The number of entries in the RX FIFO is device-specific. SPI data is added to the RX FIFO at the completion of a transfer when the received data in the shift register is transferred into the RX FIFO. SPI data are removed (popped) from the RX FIFO by reading the module POP RX FIFO Register (POPR). RX FIFO entries can only be removed from the RX FIFO by reading the POPR or by flushing the RX FIFO.

The RX FIFO Counter field (RXCTR) in the module's Status Register (SR) indicates the number of valid entries in the RX FIFO. The RXCTR is updated every time the POPR is read or SPI data is copied from the shift register to the RX FIFO.

The POPNXTPTR field in the SR points to the RX FIFO entry that is returned when the POPR is read. The POPNXTPTR contains the positive offset from RXFR0 in a number of 32-bit registers. For example, POPNXTPTR equal to two means that the RXFR2 contains the received SPI data that will be returned when the POPR is read. The POPNXTPTR field is incremented every time the POPR is read. The maximum value of the field is equal to the maximum implemented RXFR number and it rolls over after reaching the maximum.

44.5.2.6.1 Filling the RX FIFO

The RX FIFO is filled with the received SPI data from the shift register. While the RX FIFO is not full, SPI frames from the shift register are transferred to the RX FIFO. Every time an SPI frame is transferred to the RX FIFO, the RX FIFO Counter is incremented by one.

If the RX FIFO and shift register are full and a transfer is initiated, the RFOF bit in the SR is set indicating an overflow condition. Depending on the state of the ROOE bit in the MCR, the data from the transfer that generated the overflow is either ignored or shifted in to the shift register. If the ROOE bit is set, the incoming data is shifted in to the shift register. If the ROOE bit is cleared, the incoming data is ignored.

44.5.2.6.2 Draining the RX FIFO

Host CPU or a DMA can remove (pop) entries from the RX FIFO by reading the module POP RX FIFO Register (POPR). A read of the POPR decrements the RX FIFO Counter by one. Attempts to pop data from an empty RX FIFO are ignored and the RX FIFO Counter remains unchanged. The data, read from the empty RX FIFO, is undetermined.

When the RX FIFO is not empty, the RX FIFO Drain Flag (RFDF) in the SR is set. The RFDF bit is cleared when the RX_FIFO is empty and the DMA controller indicates that a read from POPR is complete or by writing a 1 to it.

44.5.3 Module baud rate and clock delay generation

The SCK frequency and the delay values for serial transfer are generated by dividing the system clock frequency by a prescaler and a scaler with the option for doubling the baud rate. The following figure shows conceptually how the SCK signal is generated.

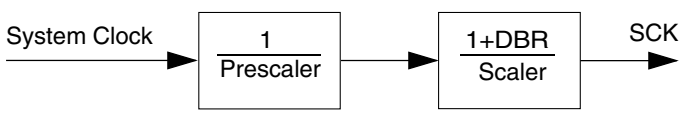


Figure 44-4. Communications clock prescalers and scalers

44.5.3.1 Baud rate generator

The baud rate is the frequency of the SCK. The protocol clock is divided by a prescaler (PBR) and scaler (BR) to produce SCK with the possibility of halving the scaler division. The DBR, PBR, and BR fields in the CTARs select the frequency of SCK by the formula in the BR field description. The following table shows an example of how to compute the baud rate.

Table 44-10. Baud rate computation example

f_P	PBR	Prescaler	BR	Scaler	DBR	Baud rate
100 MHz	0b00	2	0b0000	2	0	25 Mb/s
20 MHz	0b00	2	0b0000	2	1	10 Mb/s

NOTE

The clock frequencies mentioned in the preceding table are given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

44.5.3.2 PCS to SCK Delay (t_{CSC})

The PCS to SCK delay is the length of time from assertion of the PCS signal to the first SCK edge. See Figure 44-6 for an illustration of the PCS to SCK delay. The PCSSCK and CSSCK fields in the CTAR*x* registers select the PCS to SCK delay by the formula in the CSSCK field description. The following table shows an example of how to compute the PCS to SCK delay.

Table 44-11. PCS to SCK delay computation example

f_p	PCSSCK	Prescaler	CSSCK	Scaler	PCS to SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 µs

NOTE

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

44.5.3.3 After SCK Delay (t_{ASC})

The After SCK Delay is the length of time between the last edge of SCK and the negation of PCS. See Figure 44-6 and Figure 44-7 for illustrations of the After SCK delay. The PASC and ASC fields in the CTAR*x* registers select the After SCK Delay by the formula in the ASC field description. The following table shows an example of how to compute the After SCK delay.

Table 44-12. After SCK Delay computation example

f_P	PASC	Prescaler	ASC	Scaler	After SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 µs

NOTE

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

44.5.3.4 Delay after Transfer (t_{DT})

The Delay after Transfer is the minimum time between negation of the PCS signal for a frame and the assertion of the PCS signal for the next frame. See Figure 44-6 for an illustration of the Delay after Transfer. The PDT and DT fields in the CTAR*x* registers select the Delay after Transfer by the formula in the DT field description. The following table shows an example of how to compute the Delay after Transfer.

Table 44-13. Delay after Transfer computation example

f_P	PDT	Prescaler	DT	Scaler	Delay after Transfer
100 MHz	0b01	3	0b1110	32768	0.98 ms

NOTE

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

When in Non-Continuous Clock mode the t_{DT} delay is configured according to the equation specified in the CTAR[DT] field description. When in Continuous Clock mode, the delay is fixed at 1 SCK period.

44.5.3.5 Peripheral Chip Select Strobe Enable ($\overline{\text{PCSS}}$)

The $\overline{\text{PCSS}}$ signal provides a delay to allow the PCS signals to settle after a transition occurs thereby avoiding glitches. When the Module is in Master mode and the PCSSE bit is set in the MCR, $\overline{\text{PCSS}}$ provides a signal for an external demultiplexer to decode peripheral chip selects other than PCS5 into glitch-free PCS signals. The following figure shows the timing of the $\overline{\text{PCSS}}$ signal relative to PCS signals.

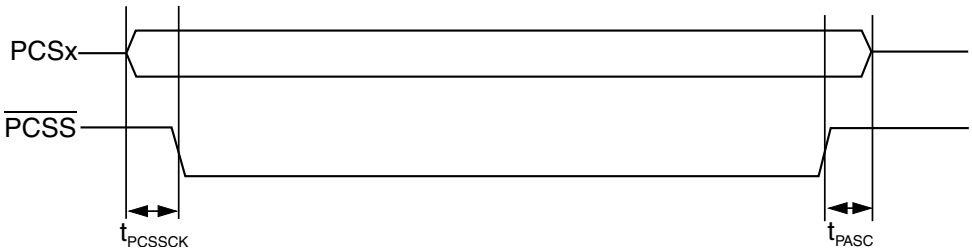


Figure 44-5. Peripheral Chip Select Strobe timing

The delay between the assertion of the PCS signals and the assertion of $\overline{\text{PCSS}}$ is selected by the PCSSCK field in the CTAR based on the following formula:

$$t_{\text{PCSSCK}} = \frac{1}{f_{\text{P}}} \times \text{PCSSCK}$$

At the end of the transfer, the delay between $\overline{\text{PCSS}}$ negation and PCS negation is selected by the PASC field in the CTAR based on the following formula:

$$t_{\text{PASC}} = \frac{1}{f_{\text{P}}} \times \text{PASC}$$

The following table shows an example of how to compute the t_{pcssck} delay.

Table 44-14. Peripheral Chip Select Strobe Assert computation example

f_P	PCSSCK	Prescaler	Delay before Transfer
100 MHz	0b11	7	70.0 ns

The following table shows an example of how to compute the t_{pasc} delay.

Table 44-15. Peripheral Chip Select Strobe Negate computation example

f_P	PASC	Prescaler	Delay after Transfer
100 MHz	0b11	7	70.0 ns

The $\overline{\text{PCSS}}$ signal is not supported when Continuous Serial Communication SCK mode is enabled.

NOTE

The clock frequency mentioned in the preceding tables is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

44.5.4 Transfer formats

The SPI serial communication is controlled by the Serial Communications Clock (SCK) signal and the PCS signals. The SCK signal provided by the master device synchronizes shifting and sampling of the data on the SIN and SOUT pins. The PCS signals serve as enable signals for the slave devices.

In Master mode, the CPOL and CPHA bits in the Clock and Transfer Attributes Registers (CTARn) select the polarity and phase of the serial clock, SCK.

- CPOL - Selects the idle state polarity of the SCK
- CPHA - Selects if the data on SOUT is valid before or on the first SCK edge

Even though the bus slave does not control the SCK signal, in Slave mode the values of CPOL and CPHA must be identical to the master device settings to ensure proper transmission. In SPI Slave mode, only CTAR0 is used.

The module supports four different transfer formats:

- Classic SPI with CPHA=0
- Classic SPI with CPHA=1
- Modified Transfer Format with CPHA = 0
- Modified Transfer Format with CPHA = 1

A modified transfer format is supported to allow for high-speed communication with peripherals that require longer setup times. The module can sample the incoming data later than halfway through the cycle to give the peripheral more setup time. The MTFE bit in the MCR selects between Classic SPI Format and Modified Transfer Format.

In the interface configurations, the module provides the option of keeping the PCS signals asserted between frames. See Continuous Selection Format for details.

44.5.4.1 Classic SPI Transfer Format (CPHA = 0)

The transfer format shown in following figure is used to communicate with peripheral SPI slave devices where the first data bit is available on the first clock edge. In this format, the master and slave sample their SIN pins on the odd-numbered SCK edges and change the data on their SOUT pins on the even-numbered SCK edges.

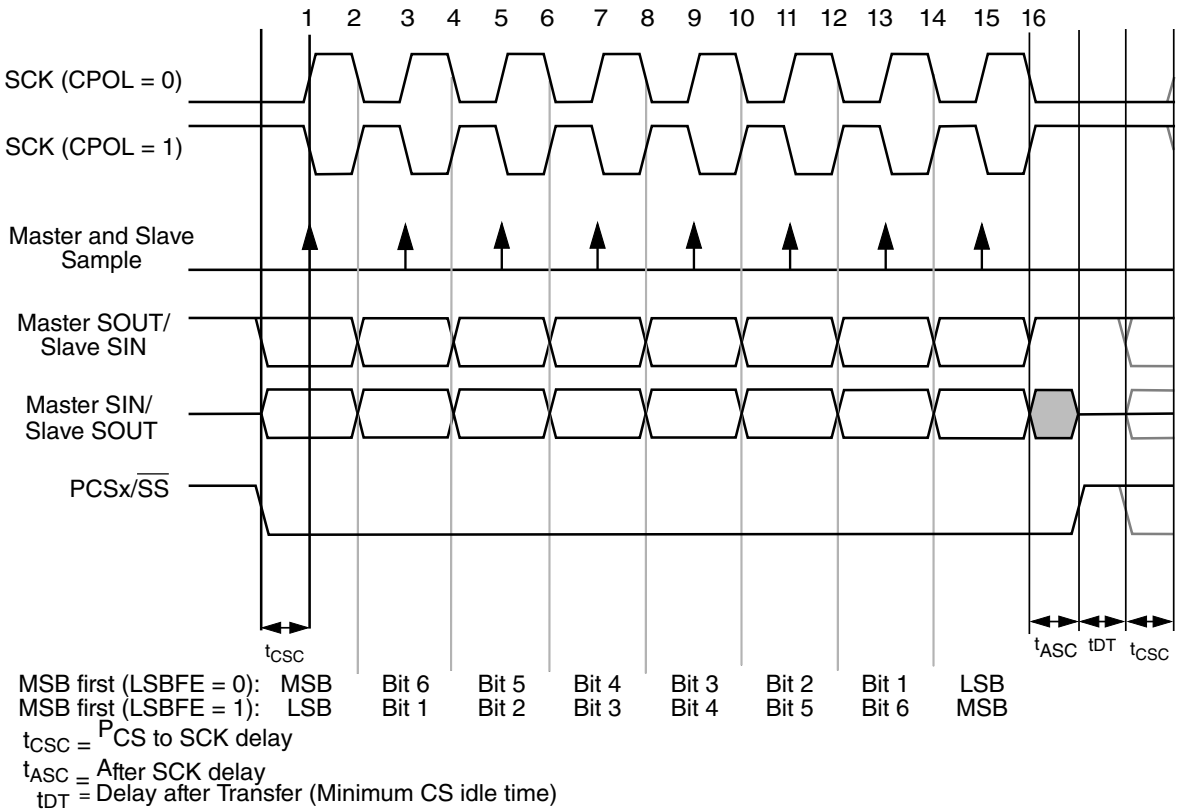


Figure 44-6. Module transfer timing diagram (MTFE=0, CPHA=0, FMSZ=8)

The master initiates the transfer by placing its first data bit on the SOUT pin and asserting the appropriate peripheral chip select signals to the slave device. The slave responds by placing its first data bit on its SOUT pin. After the t_{CSC} delay elapses, the master outputs the first edge of SCK. The master and slave devices use this edge to sample the first input data bit on their serial data input signals. At the second edge of the SCK, the master and slave devices place their second data bit on their serial data output signals. For the rest of the frame the master and the slave sample their SIN pins on the odd-numbered clock edges and changes the data on their SOUT pins on the even-numbered clock edges. After the last clock edge occurs, a delay of t_{ASC} is inserted before the master negates the PCS signals. A delay of t_{DT} is inserted before a new frame transfer can be initiated by the master.

44.5.4.2 Classic SPI Transfer Format (CPHA = 1)

This transfer format shown in the following figure is used to communicate with peripheral SPI slave devices that require the first SCK edge before the first data bit becomes available on the slave SOUT pin. In this format, the master and slave devices change the data on their SOUT pins on the odd-numbered SCK edges and sample the data on their SIN pins on the even-numbered SCK edges.

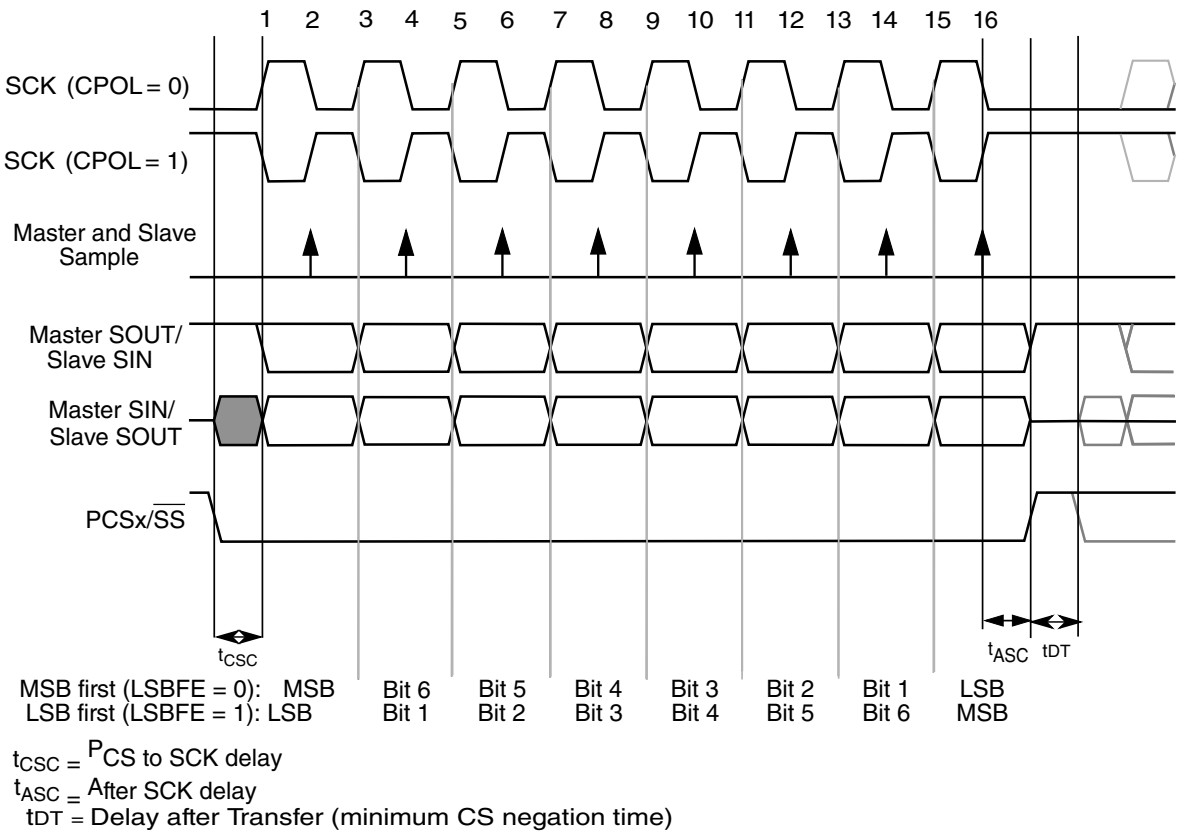


Figure 44-7. Module transfer timing diagram (MTFE=0, CPHA=1, FMSZ=8)

The master initiates the transfer by asserting the PCS signal to the slave. After the t_{CSC} delay has elapsed, the master generates the first SCK edge and at the same time places valid data on the master SOUT pin. The slave responds to the first SCK edge by placing its first data bit on its slave SOUT pin.

At the second edge of the SCK the master and slave sample their SIN pins. For the rest of the frame the master and the slave change the data on their SOUT pins on the odd-numbered clock edges and sample their SIN pins on the even-numbered clock edges. After the last clock edge occurs, a delay of t_{ASC} is inserted before the master negates the PCS signal. A delay of t_{DT} is inserted before a new frame transfer can be initiated by the master.

44.5.4.3 Modified SPI Transfer Format (MTFE = 1, CPHA = 0)

In this Modified Transfer Format both the master and the slave sample later in the SCK period than in Classic SPI mode to allow the logic to tolerate more delays in device pads and board traces. These delays become a more significant fraction of the SCK period as the SCK period decreases with increasing baud rates.

The master and the slave place data on the SOUT pins at the assertion of the PCS signal. After the PCS to SCK delay has elapsed the first SCK edge is generated. The slave samples the master SOUT signal on every odd numbered SCK edge. The SPI in the slave mode when the MTFE bit is set also places new data on the slave SOUT on every odd numbered clock edge. Regular external slave, configured with CPHA=0 format drives its SOUT output at every even numbered SCK clock edge.

The SPI master places its second data bit on the SOUT line one protocol clock after odd numbered SCK edge if the protocol clock frequency to SCK frequency ratio is higher than three. If this ratio is below four the master changes SOUT at odd numbered SCK edge. The point where the master samples the SIN is selected by the SPI_MCR[SMPL_PT] field. The master sample point can be delayed by one or two protocol clock cycles. The SMPL_PT field should be set to 0 if the protocol to SCK frequency ratio is less than 4. However if this ratio is less than 4, the actual sample point is delayed by one protocol clock cycle automatically by the design.

The following timing diagrams illustrate the SPI operation with MTFE=1. Timing delays shown are:

- T_{csc} - PCS to SCK assertion delay
- T_{acs} - After SCK PCS negation delay
- T_{su_ms} - master SIN setup time
- T_{hd_ms} - master SIN hold time
- T_{vd_sl} - slave data output valid time, time between slave data output SCK driving edge and data becomes valid.
- T_{su_sl} - data setup time on slave data input
- T_{hd_sl} - data hold time on slave data input
- T_{sys} - protocol clock period.

The following figure shows the modified transfer format for CPHA = 0 and Fsys/Fsck = 4. Only the condition where CPOL = 0 is illustrated. Solid triangles show the data sampling clock edges. The two possible slave behavior are shown.

- Signal, marked "SOUT of Ext Slave", presents regular SPI slave serial output.
- Signal, marked "SOUT of SPI Slave", presents SPI in the slave mode with MTFE bit set.

Other MTFE = 1 diagrams show SPI SIN input as being driven by a regular external SPI slave, configured according SPI master CPHA programming.

Note

In the following diagrams, f_{sys} represents the protocol clock frequency from which the Baud frequency f_{sck} is derived.

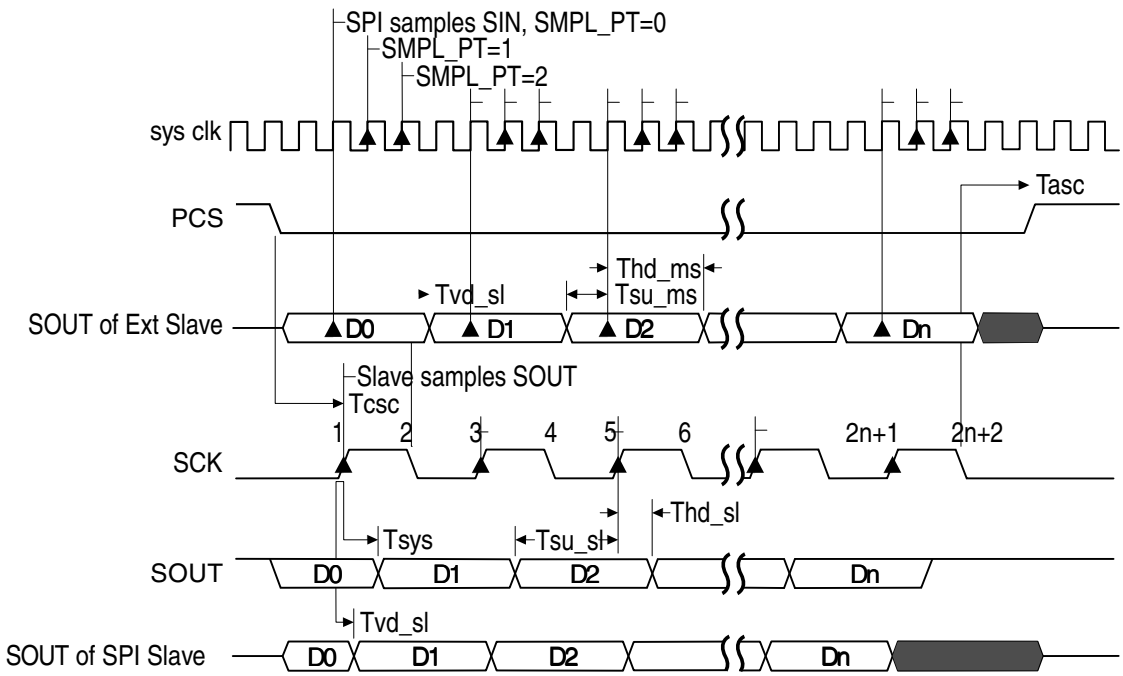


Figure 44-8. SPI Modified Transfer Format (MTFE=1, CPHA=0, $f_{sck} = f_{sys}/4$)

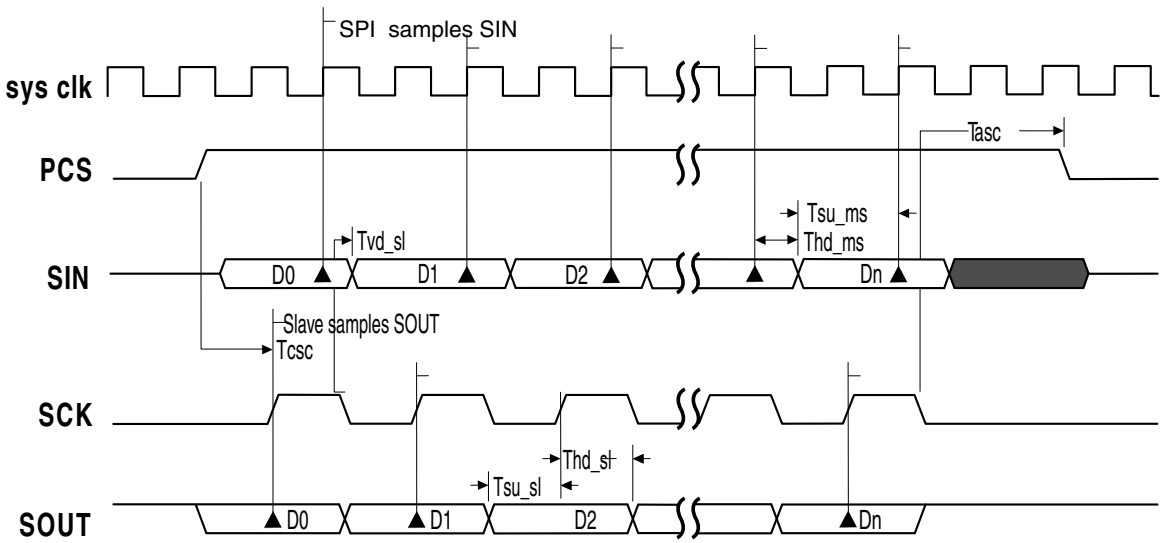


Figure 44-9. SPI Modified Transfer Format (MTFE=1, CPHA=0, $f_{sck} = f_{sys}/2$)

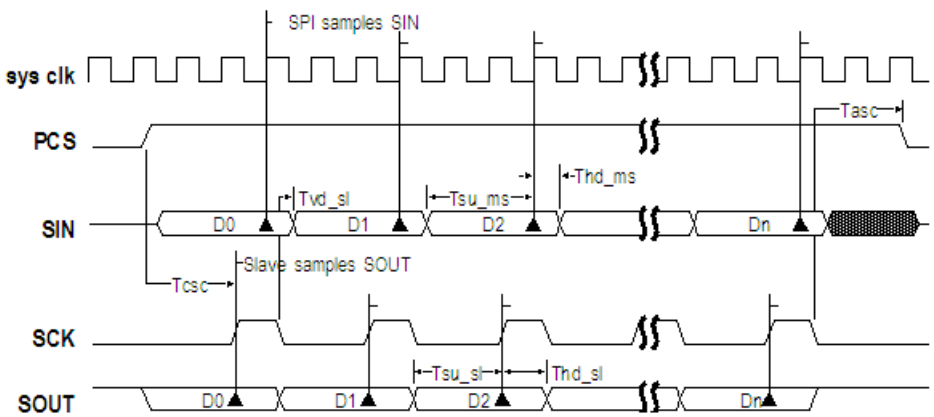


Figure 44-10. SPI Modified Transfer Format (MTFE=1, CPHA=0, $f_{sck} = f_{sys}/3$)

44.5.4.4 Modified SPI Transfer Format (MTFE = 1, CPHA = 1)

The following figures show the Modified Transfer Format for CPHA = 1. Only the condition, where CPOL = 0 is shown. At the start of a transfer the SPI asserts the PCS signal to the slave device. After the PCS to SCK delay has elapsed the master and the slave put data on their SOUT pins at the first edge of SCK . The slave samples the master SOUT signal on the even numbered edges of SCK. The master samples the slave SOUT signal on the odd numbered SCK edges starting with the third SCK edge. The slave samples the last bit on the last edge of the SCK. The master samples the last slave SOUT

bit one half SCK cycle after the last edge of SCK. No clock edge will be visible on the master SCK pin during the sampling of the last bit. **The SCK to PCS delay and the After SCK delay must be greater or equal to half of the SCK period.**

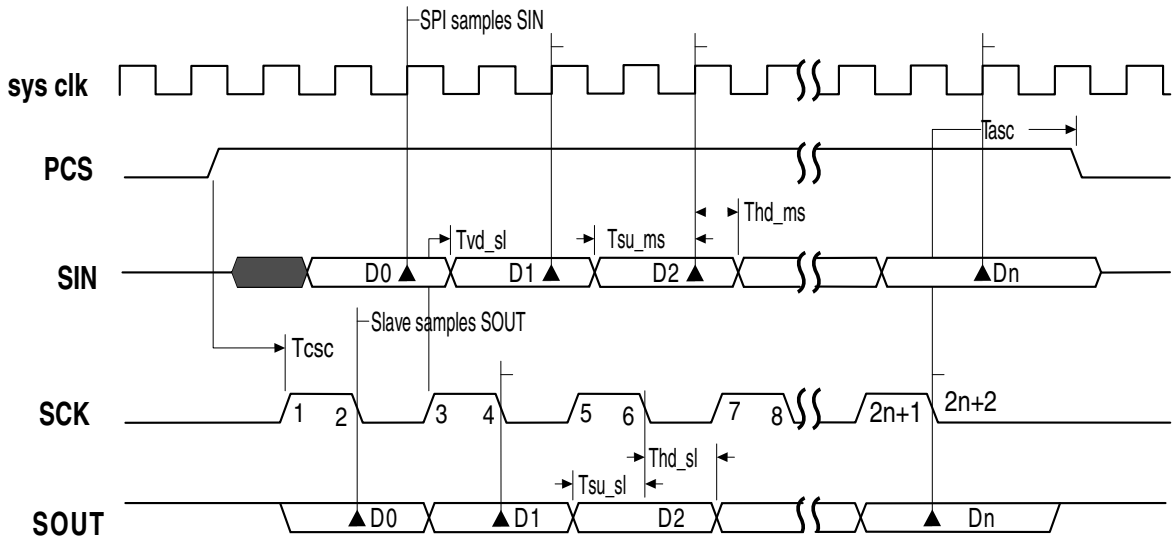


Figure 44-11. SPI Modified Transfer Format (MTFE=1, CPHA=1, $f_{sck} = f_{sys}/2$)

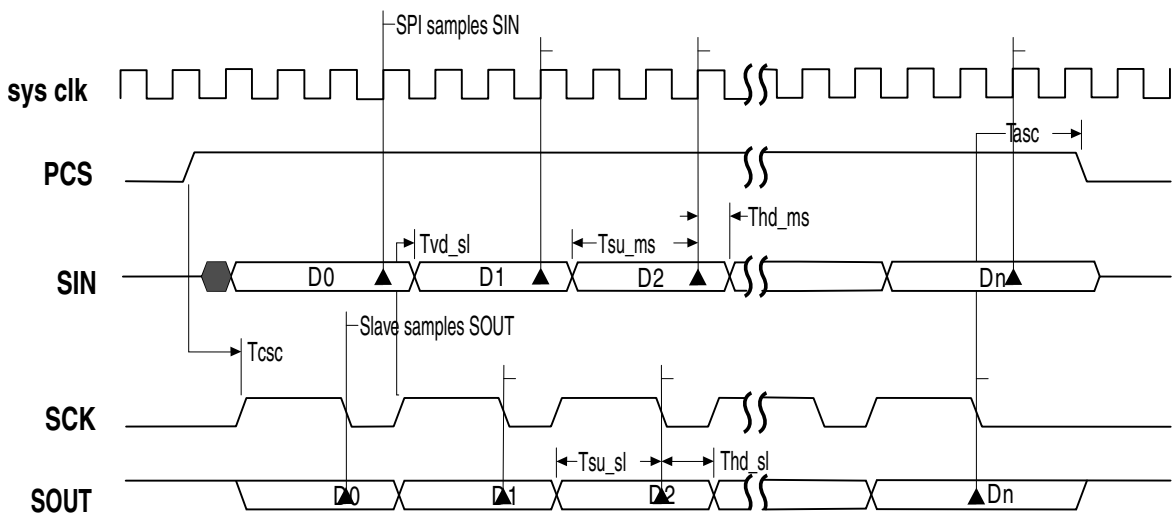


Figure 44-12. SPI Modified Transfer Format (MTFE=1, CPHA=1, $f_{sck} = f_{sys}/3$)

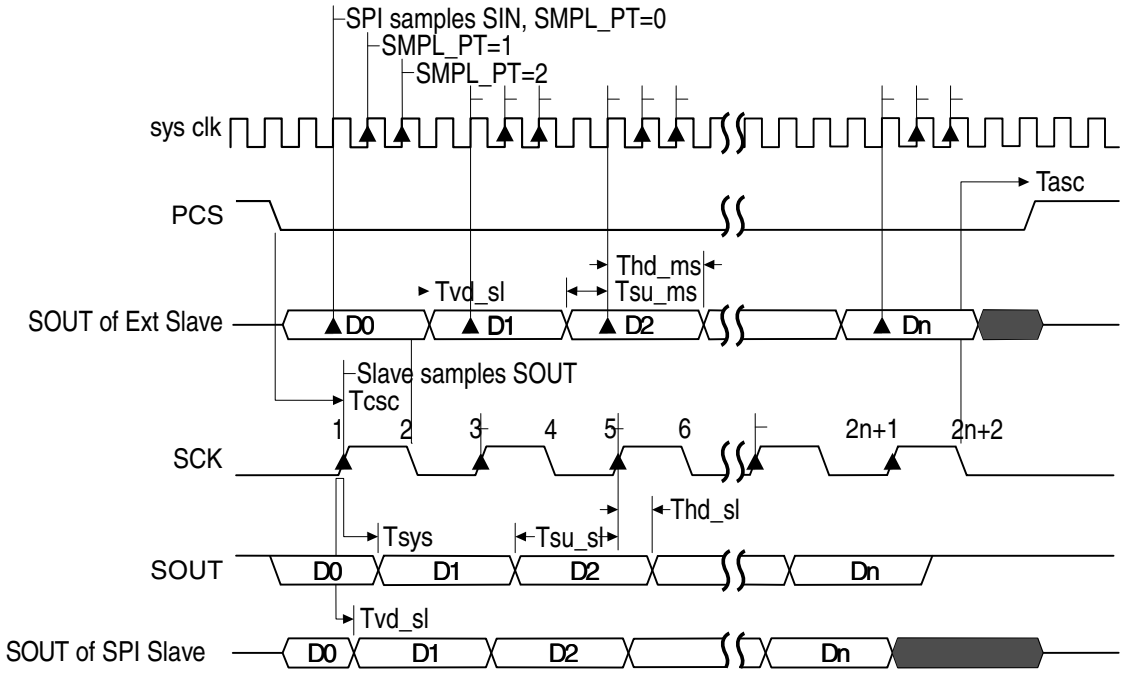


Figure 44-13. SPI Modified Transfer Format (MTFE=1, CPHA=1, $f_{sck} = f_{sys}/4$)

44.5.4.5 Continuous Selection Format

Some peripherals must be deselected between every transfer. Other peripherals must remain selected between several sequential serial transfers. The Continuous Selection Format provides the flexibility to handle the following case. The Continuous Selection Format is enabled for the SPI configuration by setting the CONT bit in the SPI command.

When the CONT bit = 0, the module drives the asserted Chip Select signals to their idle states in between frames. The idle states of the Chip Select signals are selected by the PCSISn bits in the MCR. The following timing diagram is for two four-bit transfers with CPHA = 1 and CONT = 0.

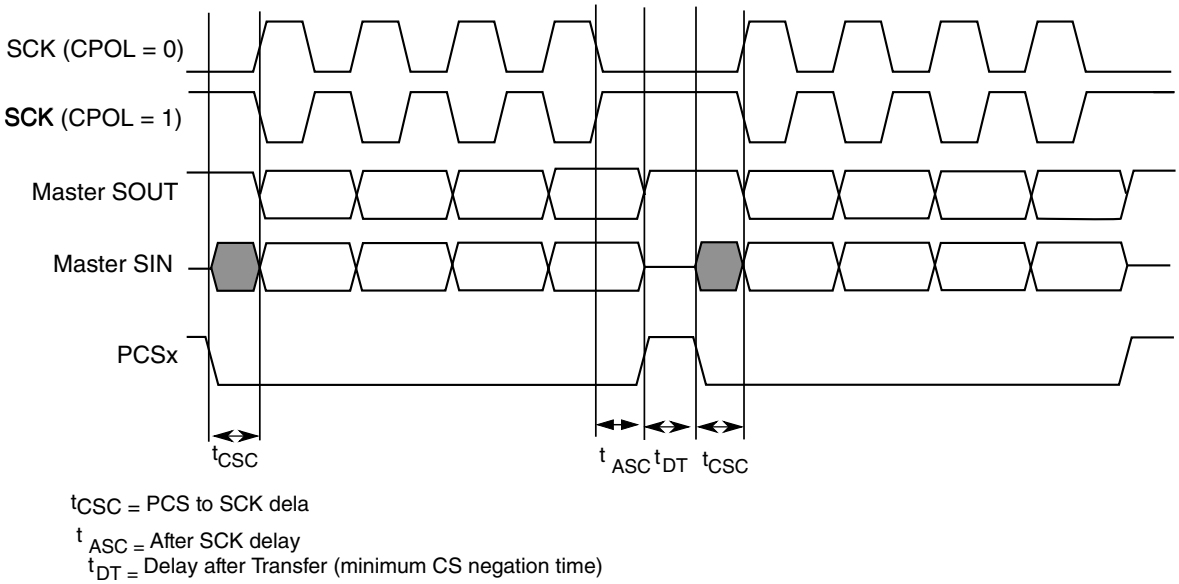


Figure 44-14. Example of non-continuous format (CPHA=1, CONT=0)

When the CONT bit = 1, the PCS signal remains asserted for the duration of the two transfers. The Delay between Transfers (t_{DT}) is not inserted between the transfers. The following figure shows the timing diagram for two four-bit transfers with CPHA = 1 and CONT = 1.

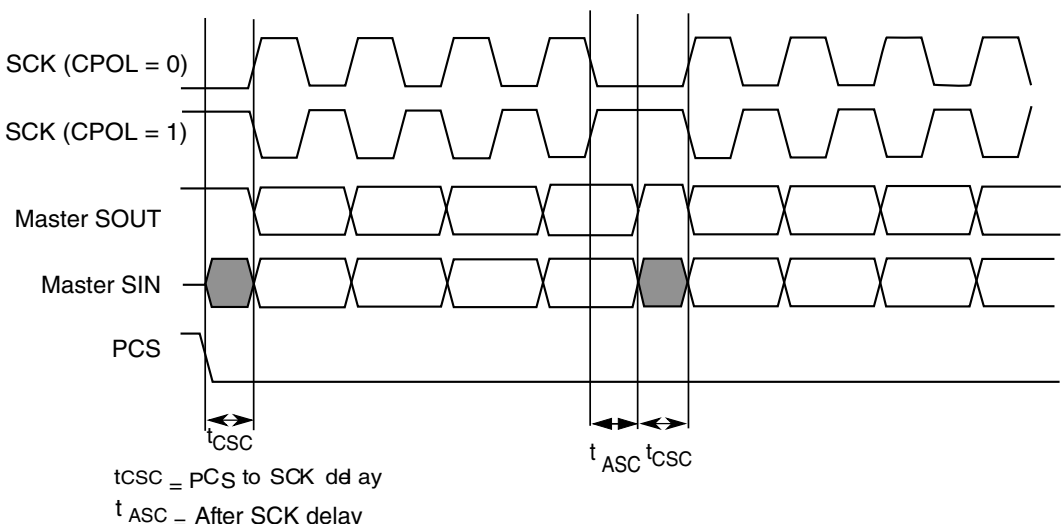


Figure 44-15. Example of continuous transfer (CPHA=1, CONT=1)

When using the module with continuous selection follow these rules:

- All transmit commands must have the same PCSn bits programming.
- The CTARs, selected by transmit commands, must be programmed with the same transfer attributes. Only FMSZ field can be programmed differently in these CTARs.

- When transmitting multiple frames in this mode, the user software must ensure that the last frame has the PUSHR[CONT] bit deasserted in Master mode and the user software must provide sufficient frames in the TX_FIFO to be sent out in Slave mode and the master deasserts the PCSn at end of transmission of the last frame.
- PUSHR[CONT] must be deasserted before asserting MCR[HALT] in master mode. This will make sure that the PCSn signals are deasserted. Asserting MCR[HALT] during continuous transfer will cause the PCSn signals to remain asserted and hence Slave Device cannot transition from Running to Stopped state.

NOTE

User must fill the TX FIFO with the number of entries that will be concatenated together under one PCS assertion for both master and slave before the TX FIFO becomes empty.

When operating in Slave mode, ensure that when the last entry in the TX FIFO is completely transmitted, that is, the corresponding TCF flag is asserted and TXFIFO is empty, the slave is deselected for any further serial communication; otherwise, an underflow error occurs.

44.5.5 Continuous Serial Communications Clock

The module provides the option of generating a Continuous SCK signal for slave peripherals that require a continuous clock.

Continuous SCK is enabled by setting the CONT_SCKE bit in the MCR. Enabling this bit generates the Continuous SCK only if MCR[HALT] bit is low. Continuous SCK is valid in all configurations.

Continuous SCK is only supported for CPHA=1. Clearing CPHA is ignored if the CONT_SCKE bit is set. Continuous SCK is supported for Modified Transfer Format.

Clock and transfer attributes for the Continuous SCK mode are set according to the following rules:

- When the module is in SPI configuration, CTAR0 is used initially. At the start of each SPI frame transfer, the CTAR specified by the CTAS for the frame is used.
- In all configurations, the currently selected CTAR remains in use until the start of a frame with a different CTAR specified, or the Continuous SCK mode is terminated.

It is recommended to keep the baud rate the same while using the Continuous SCK. Switching clock polarity between frames while using Continuous SCK can cause errors in the transfer. Continuous SCK operation is not guaranteed if the module is put into the External Stop mode or Module Disable mode.

Enabling Continuous SCK disables the PCS to SCK delay and the Delay after Transfer (t_{DT}) is fixed to one SCK cycle. The following figure is the timing diagram for Continuous SCK format with Continuous Selection disabled.

NOTE

In Continuous SCK mode, for the SPI transfer CTAR0 should always be used, and the TX FIFO must be cleared using the MCR[CLR_TXF] field before initiating transfer.

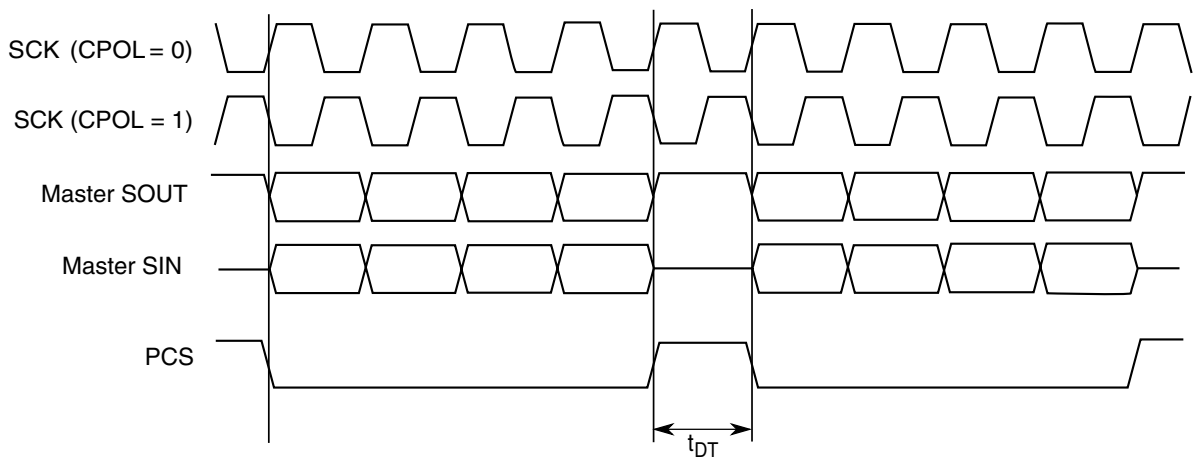


Figure 44-16. Continuous SCK Timing Diagram (CONT=0)

If the CONT bit in the TX FIFO entry is set, PCS remains asserted between the transfers. Under certain conditions, SCK can continue with PCS asserted, but with no data being shifted out of SOUT, that is, SOUT pulled high. This can cause the slave to receive incorrect data. Those conditions include:

- Continuous SCK with CONT bit set, but no data in the TX FIFO.
- Continuous SCK with CONT bit set and entering Stopped state (refer to Start and Stop of module transfers).
- Continuous SCK with CONT bit set and entering Stop mode or Module Disable mode.

The following figure shows timing diagram for Continuous SCK format with Continuous Selection enabled.

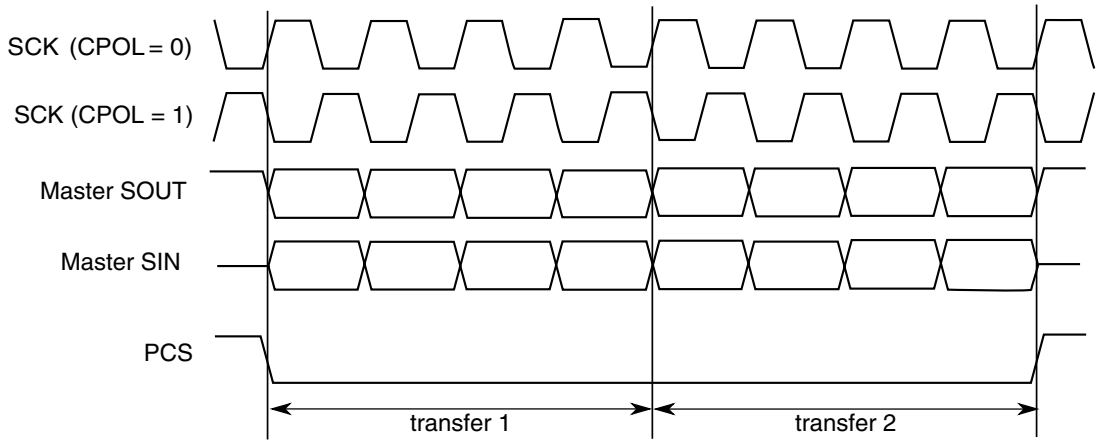


Figure 44-17. Continuous SCK timing diagram (CONT=1)

44.5.6 Slave Mode Operation Constraints

Slave mode logic shift register is buffered. This allows data streaming operation, when the module is permanently selected and data is shifted in with a constant rate.

The transmit data is transferred at second SCK clock edge of the each frame to the shift register if the $\overline{SS}$ signal is asserted and any time when transmit data is ready and $\overline{SS}$ signal is negated.

Received data is transferred to the receive buffer at last SCK edge of each frame, defined by frame size programmed to the CTAR0/1 register. Then the data from the buffer is transferred to the RXFIFO or DDR register.

If the $\overline{SS}$ negates before that last SCK edge, the data from shift register is lost.

44.5.7 Interrupts/DMA requests

The module has several conditions that can generate only interrupt requests and two conditions that can generate interrupt or DMA requests. The following table lists these conditions.

Table 44-16. Interrupt and DMA request conditions

Condition	Flag	Interrupt	DMA
End of Queue (EOQ)	EOQF	Yes	-
TX FIFO Fill	TFFF	Yes	Yes
Transfer Complete	TCF	Yes	-
TX FIFO Underflow	TFUF	Yes	-

Table continues on the next page...

Table 44-16. Interrupt and DMA request conditions (continued)

Condition	Flag	Interrupt	DMA
RX FIFO Drain	RFDF	Yes	Yes
RX FIFO Overflow	RFOF	Yes	-

Each condition has a flag bit in the module Status Register (SR) and a Request Enable bit in the DMA/Interrupt Request Select and Enable Register (RSER). Certain flags (as shown in above table) generate interrupt requests or DMA requests depending on configuration of RSER register.

The module also provides a global interrupt request line, which is asserted when any of individual interrupt requests lines is asserted.

44.5.7.1 End Of Queue interrupt request

The End Of Queue (EOQ) interrupt request indicates that the end of a transmit queue is reached. The module generates the interrupt request when EOQ interrupt requests are enabled (RSER[EOQF_RE]) and the EOQ bit in the executing SPI command is 1.

The module generates the interrupt request when the last bit of the SPI frame with EOQ bit set is transmitted.

44.5.7.2 Transmit FIFO Fill Interrupt or DMA Request

The Transmit FIFO Fill Request indicates that the TX FIFO is not full. The Transmit FIFO Fill Request is generated when the number of entries in the TX FIFO is less than the maximum number of possible entries, and the TFFF_RE bit in the RSER is set. The TFFF_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated.

NOTE

TFFF flag clears automatically when DMA is used to fill TX FIFO. Configure the DMA to fill only one FIFO location per transfer.

To clear TFFF when not using DMA, follow these steps for every PUSH performed using CPU to fill TX FIFO:

1. Wait until TFFF = 1.
2. Write data to PUSHR using CPU.

3. Clear TFFF by writing a 1 to its location. If TX FIFO is not full, this flag will not clear.

44.5.7.3 Transfer Complete Interrupt Request

The Transfer Complete Request indicates the end of the transfer of a serial frame. The Transfer Complete Request is generated at the end of each frame transfer when the TCF_RE bit is set in the RSER.

44.5.7.4 Transmit FIFO Underflow Interrupt Request

The Transmit FIFO Underflow Request indicates that an underflow condition in the TX FIFO has occurred. The transmit underflow condition is detected only for the module operating in Slave mode and SPI configuration. The TFUF bit is set when the TX FIFO of the module is empty, and a transfer is initiated from an external SPI master. If the TFUF bit is set while the TFUF_RE bit in the RSER is set, an interrupt request is generated.

44.5.7.5 Receive FIFO Drain Interrupt or DMA Request

The Receive FIFO Drain Request indicates that the RX FIFO is not empty. The Receive FIFO Drain Request is generated when the number of entries in the RX FIFO is not zero, and the RFDF_RE bit in the RSER is set. The RFDF_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated. Configure the DMA to drain only one FIFO location per transfer.

44.5.7.6 Receive FIFO Overflow Interrupt Request

The Receive FIFO Overflow Request indicates that an overflow condition in the RX FIFO has occurred. A Receive FIFO Overflow request is generated when RX FIFO and shift register are full and a transfer is initiated. The RFOF_RE bit in the RSER must be set for the interrupt request to be generated.

Depending on the state of the ROOE bit in the MCR, the data from the transfer that generated the overflow is either ignored or shifted in to the shift register. If the ROOE bit is set, the incoming data is shifted in to the shift register. If the ROOE bit is cleared, the incoming data is ignored.

44.5.8 Power saving features

The module supports following power-saving strategies:

- External Stop mode
- Module Disable mode – Clock gating of non-memory mapped logic

44.5.8.1 Stop mode (External Stop mode)

This module supports the Stop mode protocol. When a request is made to enter External Stop mode, the module acknowledges the request . If a serial transfer is in progress, then this module waits until it reaches the frame boundary before it is ready to have its clocks shut off . While the clocks are shut off, this module's memory-mapped logic is not accessible. This also puts the module in STOPPED state. The SR[TXRXS] bit is cleared to indicate STOPPED state. The states of the interrupt and DMA request signals cannot be changed while in External Stop mode.

44.5.8.2 Module Disable mode

Module Disable mode is a block-specific mode that the module can enter to save power. Host CPU can initiate the Module Disable mode by setting the MDIS bit in the MCR. The Module Disable mode can also be initiated by hardware.

When the MDIS bit is set, the module negates the Clock Enable signal at the next frame boundary. Once the Clock Enable signal is negated, it is said to have entered Module Disable Mode. This also puts the module in STOPPED state. The SR[TXRXS] bit is cleared to indicate STOPPED state. If implemented, the Clock Enable signal can stop the clock to the non-memory mapped logic. When Clock Enable is negated, the module is in a dormant state, but the memory mapped registers are still accessible. Certain read or write operations have a different effect when the module is in the Module Disable mode. Reading the RX FIFO Pop Register does not change the state of the RX FIFO. Similarly, writing to the PUSHR Register does not change the state of the TX FIFO or CMD FIFO. Clearing either of the FIFOs has no effect in the Module Disable mode. Changes to the DIS_TXF and DIS_RXF fields of the MCR have no effect in the Module Disable mode. In the Module Disable mode, all status bits and register flags in the module return the correct values when read, but writing to them has no effect. Writing to the TCR during Module Disable mode has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

44.6 Initialization/application information

This section describes how to initialize the module.

44.6.1 How to manage queues

The queues are not part of the module, but it includes features in support of queue management. Queues are primarily supported in SPI configuration.

1. When module executes last command word from a queue, the EOQ bit in the command word is set to indicate it that this is the last entry in the queue.
2. At the end of the transfer, corresponding to the command word with EOQ set is sampled, the EOQ flag (EOQF) in the SR is set.
3. The setting of the EOQF flag disables serial transmission and reception of data, putting the module in the Stopped state. The TXRXS bit is cleared to indicate the Stopped state.
4. The DMA can continue to fill TX FIFO until it is full or step 5 occurs.
5. Disable DMA transfers by disabling the DMA enable request for the DMA channel assigned to TX FIFO (and CMD FIFO) and RX FIFO. This is done by clearing the corresponding DMA enable request bits in the DMA Controller.
6. Ensure all received data in RX FIFO has been transferred to memory receive queue by reading the RXCNT in SR or by checking RFDF in the SR after each read operation of the POPR.
7. Modify DMA descriptor of TX and RX channels for new queues
8. Flush TX FIFO (and CMD FIFO) by writing a 1 to the CLR_TXF bit in the MCR. Flush RX FIFO by writing a '1' to the CLR_RXF bit in the MCR.
9. Clear transfer count either by setting CTCNT bit in the command word of the first entry in the new queue or via CPU writing directly to SPI_TCNT field in the TCR.
10. Enable DMA channel by enabling the DMA enable request for the DMA channel assigned to the module TX FIFO, (and CMD FIFO) and RX FIFO by setting the corresponding DMA set enable request bit.
11. Enable serial transmission and serial reception of data by clearing the EOQF bit.

44.6.2 Switching Master and Slave mode

When changing modes in the module, follow the steps below to guarantee proper operation.

1. Halt it by setting MCR[HALT].
2. Clear the transmit and receive FIFOs by writing a 1 to the CLR_TXF and CLR_RXF bits in MCR.
3. Set the appropriate mode in MCR[MSTR] and enable it by clearing MCR[HALT].

44.6.3 Initializing Module in Master/Slave Modes

Once the appropriate mode in MCR[MSTR] is configured, the module is enabled by clearing MCR[HALT]. It should be ensured that module Slave is enabled before enabling it's Master. This ensures the Slave is ready to be communicated with, before Master initializes communication.

44.6.4 Baud rate settings

The following table shows the baud rate that is generated based on the combination of the baud rate prescaler PBR and the baud rate scaler BR in the CTARs. The values calculated assume a 100 MHz protocol frequency and the double baud rate DBR bit is cleared.

NOTE

The clock frequency mentioned above is given as an example in this chapter. See the clocking chapter for the frequency used to drive this module in the device.

Table 44-17. Baud rate values (bps)

		Baud rate divider prescaler values			
		2	3	5	7
Baud Rate Scaler Values	2	25.0M	16.7M	10.0M	7.14M
	4	12.5M	8.33M	5.00M	3.57M
	6	8.33M	5.56M	3.33M	2.38M
	8	6.25M	4.17M	2.50M	1.79M
	16	3.12M	2.08M	1.25M	893k
	32	1.56M	1.04M	625k	446k
	64	781k	521k	312k	223k
	128	391k	260k	156k	112k
	256	195k	130k	78.1k	55.8k
	512	97.7k	65.1k	39.1k	27.9k
	1024	48.8k	32.6k	19.5k	14.0k
	2048	24.4k	16.3k	9.77k	6.98k
	4096	12.2k	8.14k	4.88k	3.49k
	8192	6.10k	4.07k	2.44k	1.74k
	16384	3.05k	2.04k	1.22k	872
	32768	1.53k	1.02k	610	436

44.6.5 Delay settings

The following table shows the values for the Delay after Transfer (t_{DT}) and CS to SCK Delay (T_{CSC}) that can be generated based on the prescaler values and the scaler values set in the CTARs. The values calculated assume a 100 MHz protocol frequency.

NOTE

The clock frequency mentioned above is given as an example in this chapter. See the clocking chapter for the frequency used to drive this module in the device.

Table 44-18. Delay values

		Delay prescaler values			
		1	3	5	7
Delay scaler values	2	20.0 ns	60.0 ns	100.0 ns	140.0 ns
	4	40.0 ns	120.0 ns	200.0 ns	280.0 ns
	8	80.0 ns	240.0 ns	400.0 ns	560.0 ns
	16	160.0 ns	480.0 ns	800.0 ns	1.1 µs
	32	320.0 ns	960.0 ns	1.6 µs	2.2 µs
	64	640.0 ns	1.9 µs	3.2 µs	4.5 µs
	128	1.3 µs	3.8 µs	6.4 µs	9.0 µs
	256	2.6 µs	7.7 µs	12.8 µs	17.9 µs
	512	5.1 µs	15.4 µs	25.6 µs	35.8 µs
	1024	10.2 µs	30.7 µs	51.2 µs	71.7 µs
	2048	20.5 µs	61.4 µs	102.4 µs	143.4 µs
	4096	41.0 µs	122.9 µs	204.8 µs	286.7 µs
	8192	81.9 µs	245.8 µs	409.6 µs	573.4 µs
	16384	163.8 µs	491.5 µs	819.2 µs	1.1 ms
	32768	327.7 µs	983.0 µs	1.6 ms	2.3 ms
	65536	655.4 µs	2.0 ms	3.3 ms	4.6 ms

44.6.6 Calculation of FIFO pointer addresses

Complete visibility of the FIFO contents is available through the FIFO registers, and valid entries can be identified through a memory-mapped pointer and counter for each FIFO. The pointer to the first-in entry in each FIFO is memory mapped. For the TX FIFO the first-in pointer is the Transmit Next Pointer (TXNXTPTR). For the CMD FIFO the first-in pointer is the Command Next Pointer (CMDNXTPTR). For the RX FIFO the first-in pointer is the Pop Next Pointer (POPNXTPTR). The following figure illustrates the concept of first-in and last-in FIFO entries along with the FIFO Counter. The TX FIFO is chosen for the illustration, but the concepts carry over. See Transmit First In First Out (TX FIFO) buffering mechanism, Command First In First Out (CMD FIFO) Buffering Mechanism and Receive First In First Out (RX FIFO) buffering mechanism for details on the FIFO operation.

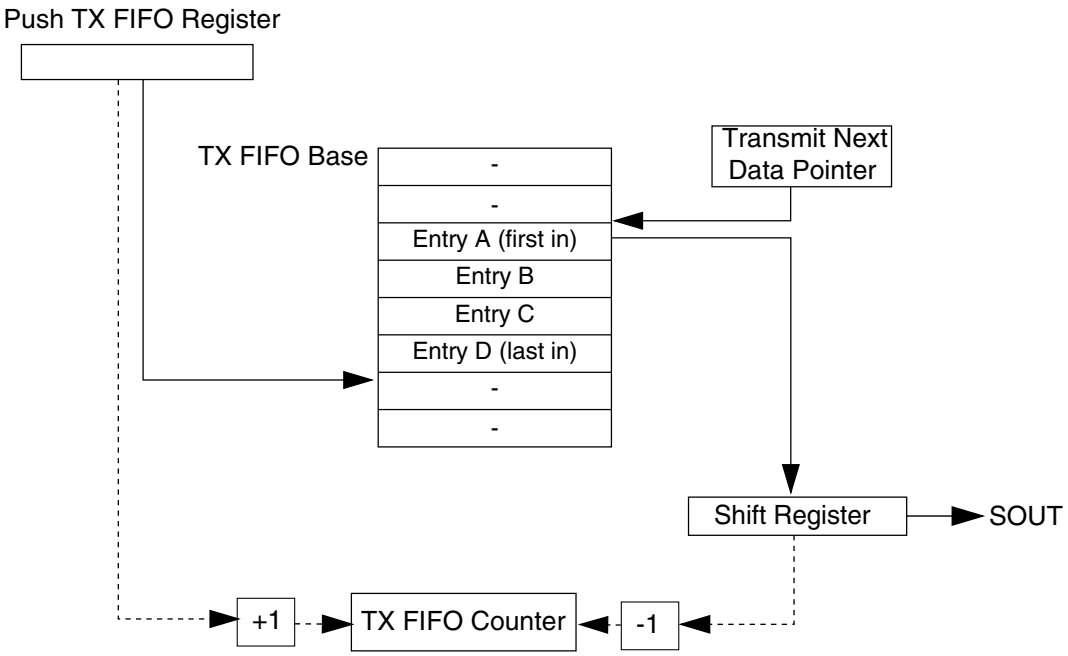


Figure 44-18. TX FIFO pointers and counter

44.6.6.1 Address Calculation for the First-in Entry and Last-in Entry in the TX FIFO

The memory address of the first-in entry in the TX FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{TXFIFOBase} + \left(4 \times \text{TXNXTPTR}\right)$$

The memory address of the last-in entry in the TX FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{TXFIFOBase} + 4 \times \left(\text{TXCTR} + \text{TXNXTPTR} - 1\right) \bmod \left(\text{TXFIFOdepth}\right)$$

TX FIFO Base - Base address of TX FIFO
TXCTR - TX FIFO Counter
TXNXTPTR - Transmit Next Pointer
TX FIFO Depth - Transmit FIFO depth, implementation specific

44.6.6.2 Address Calculation for the First-in Entry and Last-in Entry in the CMD FIFO

The memory address of the first-in entry in the CMD FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{TXFIFOBase} + (4 \times \text{TXNXTPTR})$$

The memory address of the last-in entry in the CMD FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{TXFIFOBase} + 4 \times (\text{TXCTR} + \text{TXNXTPTR} - 1) \bmod (\text{TXFIFOdepth})$$

CMD FIFO Base - Base address of CMD FIFO
CMDCTR - CMD FIFO Counter
CMDNXTPTR - Command Next Pointer
CMD FIFO Depth - Command FIFO depth, implementation specific

44.6.6.3 Address Calculation for the First-in Entry and Last-in Entry in the RX FIFO

The memory address of the first-in entry in the RX FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{RX FIFOBase} + (4 \times \text{POPNXTPTR})$$

The memory address of the last-in entry in the RX FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{RX FIFO Base} + 4 \times (\text{RXCTR} + \text{POPNXTPTR} - 1) \bmod (\text{RXFIFOdepth})$$

RX FIFO Base - Base address of RX FIFO
RXCTR - RX FIFO counter
POPNXTPTR - Pop Next Pointer
RX FIFO Depth - Receive FIFO depth, implementation specific

Chapter 45
Inter-Integrated Circuit (I2C)

45.1 Chip-specific I2C information

45.1.1 I2C signals

Signal	I/O	Connected to
Transmission complete	Output	DMA_MUX source 22

45.2 Introduction

The inter-integrated circuit (I^2C, I2C, or IIC) module provides a method of communication between a number of devices.

The interface is designed to operate up to 100 kbit/s with maximum bus loading and timing. The I2C device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF. The I2C module also complies with the *System Management Bus (SMBus) Specification, version 2*.

45.2.1 Features

The I2C module has the following features:

- Compatible with *The I^2C-Bus Specification*
- Multimaster operation
- Software programmable for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit

- Interrupt-driven byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- START and STOP signal generation and detection
- Repeated START signal generation and detection
- Acknowledge bit generation and detection
- Bus busy detection
- General call recognition
- 10-bit address extension
- Support for *System Management Bus (SMBus) Specification, version 2*
- Programmable input glitch filter
- Low power mode wakeup on slave address match
- Range slave address support
- DMA support

45.2.2 Modes of operation

The I2C module's operation in various low power modes is as follows:

- Run mode: This is the basic mode of operation. To conserve power in this mode, disable the module.
- Wait mode: The module continues to operate when the core is in Wait mode and can provide a wakeup interrupt.
- Stop mode: The module is inactive in Stop mode for reduced power consumption, except that address matching is enabled in Stop mode. The STOP instruction does not affect the I2C module's register states.

45.2.3 Block diagram

The following figure is a functional block diagram of the I2C module.

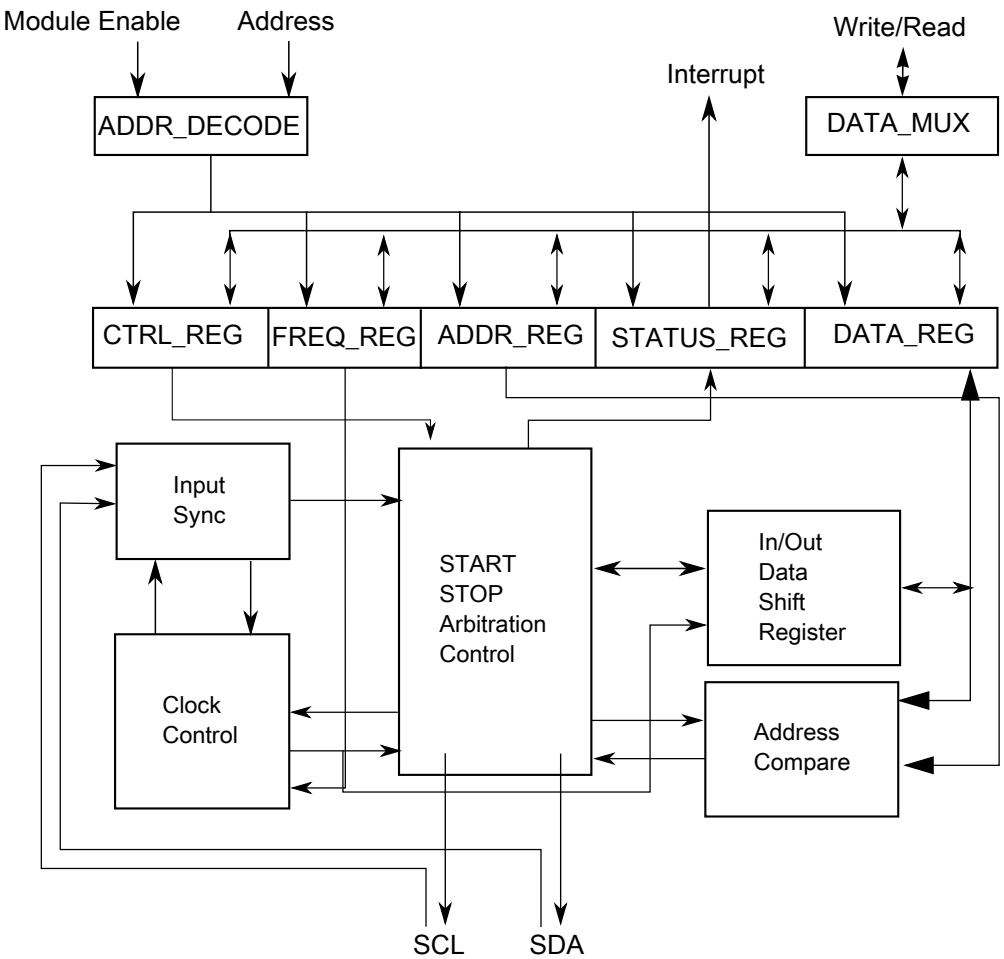


Figure 45-1. I2C Functional block diagram

45.3 I²C signal descriptions

The signal properties of I²C are shown in the table found here.

Table 45-1. I²C signal descriptions

Signal	Description	I/O
SCL	Bidirectional serial clock line of the I²C system.	I/O
SDA	Bidirectional serial data line of the I²C system.	I/O

45.4 Memory map/register definition

This section describes in detail all I2C registers accessible to the end user.

I2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6000	I2C Address Register 1 (I2C0_A1)	8	R/W	00h	45.4.1/1228
4006_6001	I2C Frequency Divider register (I2C0_F)	8	R/W	00h	45.4.2/1229
4006_6002	I2C Control Register 1 (I2C0_C1)	8	R/W	00h	45.4.3/1230
4006_6003	I2C Status register (I2C0_S)	8	R/W	80h	45.4.4/1231
4006_6004	I2C Data I/O register (I2C0_D)	8	R/W	00h	45.4.5/1233
4006_6005	I2C Control Register 2 (I2C0_C2)	8	R/W	00h	45.4.6/1234
4006_6006	I2C Programmable Input Glitch Filter Register (I2C0_FLT)	8	R/W	00h	45.4.7/1235
4006_6007	I2C Range Address register (I2C0_RA)	8	R/W	00h	45.4.8/1236
4006_6008	I2C SMBus Control and Status register (I2C0_SMB)	8	R/W	00h	45.4.9/1237
4006_6009	I2C Address Register 2 (I2C0_A2)	8	R/W	C2h	45.4.10/ 1239
4006_600A	I2C SCL Low Timeout Register High (I2C0_SLTH)	8	R/W	00h	45.4.11/ 1239
4006_600B	I2C SCL Low Timeout Register Low (I2C0_SLTL)	8	R/W	00h	45.4.12/ 1239

45.4.1 I2C Address Register 1 (I2Cx_A1)

This register contains the slave address to be used by the I2C module.

Address: 4006_6000h base + 0h offset = 4006_6000h

Bit	7	6	5	4	3	2	1	0
Read	AD[7:1]							0
Write								
Reset	0	0	0	0	0	0	0	0

I2Cx_A1 field descriptions

Field	Description
7–1 AD[7:1]	Address Contains the primary slave address used by the I2C module when it is addressed as a slave. This field is used in the 7-bit address scheme and the lower seven bits in the 10-bit address scheme.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

45.4.2 I2C Frequency Divider register (I2Cx_F)

Address: 4006_6000h base + 1h offset = 4006_6001h

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

I2Cx_F field descriptions

Field	Description
7–6 MULT	Multiplier Factor Defines the multiplier factor (mul). This factor is used along with the SCL divider to generate the I2C baud rate. 00 mul = 1 01 mul = 2 10 mul = 4 11 Reserved

ICR	ClockRate
	
	Prescales the I2C module clock for bit rate selection. This field and the MULT field determine the I2C baud rate, the SDA hold time, the SCL start hold time, and the SCL stop hold time. For a list of values corresponding to each ICR setting, see I2C divider and hold values.
	
	The SCL divider multiplied by multiplier factor (mul) determines the I2C baud rate.
	
	`I2C baud rate = I2C module clock speed (Hz)/(mul × SCL divider)`
	
	The SDA hold time is the delay from the falling edge of SCL (I2C clock) to the changing of SDA (I2C data).
	
	`SDA hold time = I2C module clock period (s) × mul × SDA hold value`
	
	The SCL start hold time is the delay from the falling edge of SDA (I2C data) while SCL is high (start condition) to the falling edge of SCL (I2C clock).
	
	`SCL start hold time = I2C module clock period (s) × mul × SCL start hold value`
	
	The SCL stop hold time is the delay from the rising edge of SCL (I2C clock) to the rising edge of SDA (I2C data) while SCL is high (stop condition).
	
	`SCL stop hold time = I2C module clock period (s) × mul × SCL stop hold value`
	
	For example, if the I2C module clock speed is 8 MHz, the following table shows the possible hold time values with different ICR and MULT selections to achieve an I^2C baud rate of 100 kbit/s.

MULT	ICR	Hold times (µs)		
		SDA	SCL Start	SCL Stop
2h	00h	3.500	3.000	5.500
1h	07h	2.500	4.000	5.250
1h	0Bh	2.250	4.000	5.250
0h	14h	2.125	4.250	5.125

Table continues on the next page...

I2Cx_F field descriptions (continued)

Field	Description				
	MULT	ICR	Hold times (µs)		
			SDA	SCL Start	SCL Stop
	0h	18h	1.125	4.750	5.125

45.4.3 I2C Control Register 1 (I2Cx_C1)

Address: 4006_6000h base + 2h offset = 4006_6002h

Bit	7	6	5	4	3	2	1	0
Read	IICEN	IICIE	MST	TX	TXAK	0	WUEN	DMAEN
Write						RSTA		
Reset	0	0	0	0	0	0	0	0

I2Cx_C1 field descriptions

Field	Description
7 IICEN	I2C Enable Enables I2C module operation. 0 Disabled 1 Enabled
6 IICIE	I2C Interrupt Enable Enables I2C interrupt requests. 0 Disabled 1 Enabled
5 MST	Master Mode Select When MST is changed from 0 to 1, a START signal is generated on the bus and master mode is selected. When this bit changes from 1 to 0, a STOP signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode
4 TX	Transmit Mode Select Selects the direction of master and slave transfers. In master mode this bit must be set according to the type of transfer required. Therefore, for address cycles, this bit is always set. When addressed as a slave this bit must be set by software according to the SRW bit in the status register. 0 Receive 1 Transmit

Table continues on the next page...

I2Cx_C1 field descriptions (continued)

Field	Description
3 TXAK	Transmit Acknowledge Enable Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers. The value of SMB[FACK] affects NACK/ACK generation. **NOTE:** SCL is held low until TXAK is written. 0 An acknowledge signal is sent to the bus on the following receiving byte (if FACK is cleared) or the current receiving byte (if FACK is set). 1 No acknowledge signal is sent to the bus on the following receiving data byte (if FACK is cleared) or the current receiving data byte (if FACK is set).
2 RSTA	Repeat START Writing 1 to this bit generates a repeated START condition provided it is the current master. This bit will always be read as 0. Attempting a repeat at the wrong time results in loss of arbitration.
1 WUEN	Wakeup Enable The I2C module can wake the MCU from low power mode with no peripheral bus running when slave address matching occurs. 0 Normal operation. No interrupt generated when address matching in low power mode. 1 Enables the wakeup function in low power mode.
0 DMAEN	DMA Enable Enables or disables the DMA function. 0 All DMA signalling disabled. 1 DMA transfer is enabled. While SMB[FACK] = 0, the following conditions trigger the DMA request: • a data byte is received, and either address or data is transmitted. (ACK/NACK is automatic) • the first byte received matches the A1 register or is a general call address. If any address matching occurs, S[IAAS] and S[TCF] are set. If the direction of transfer is known from master to slave, then it is not required to check S[SRW]. With this assumption, DMA can also be used in this case. In other cases, if the master reads data from the slave, then it is required to rewrite the C1 register operation. With this assumption, DMA cannot be used. When FACK = 1, an address or a data byte is transmitted.

45.4.4 I2C Status register (I2Cx_S)

Address: 4006_6000h base + 3h offset = 4006_6003h

Bit	7	6	5	4	3	2	1	0
Read	TCF	IAAS	BUSY	ARBL	RAM	SRW	IICIF	RXAK
Write				w1c			w1c	
Reset	1	0	0	0	0	0	0	0

I2Cx_S field descriptions

Field	Description
7 TCF	Transfer Complete Flag Acknowledges a byte transfer; TCF is set on the completion of a byte transfer. This bit is valid only during or immediately following a transfer to or from the I2C module. TCF is cleared by reading the I2C data register in receive mode or by writing to the I2C data register in transmit mode. 0 Transfer in progress 1 Transfer complete
6 IAAS	Addressed As A Slave This bit is set by one of the following conditions: • The calling address matches the programmed primary slave address in the A1 register, or matches the range address in the RA register (which must be set to a nonzero value and under the condition I2C_C2[RMEN] = 1). • C2[GCAEN] is set and a general call is received. • SMB[SIICAEN] is set and the calling address matches the second programmed slave address. • ALERTEN is set and an SMBus alert response address is received • RMEN is set and an address is received that is within the range between the values of the A1 and RA registers. IAAS sets before the ACK bit. The CPU must check the SRW bit and set TX/RX accordingly. Writing the C1 register with any value clears this bit. 0 Not addressed 1 Addressed as a slave
5 BUSY	Bus Busy Indicates the status of the bus regardless of slave or master mode. This bit is set when a START signal is detected and cleared when a STOP signal is detected. 0 Bus is idle 1 Bus is busy
4 ARBL	Arbitration Lost This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software, by writing 1 to it. 0 Standard bus operation. 1 Loss of arbitration.
3 RAM	Range Address Match This bit is set to 1 by any of the following conditions, if I2C_C2[RMEN] = 1: • Any nonzero calling address is received that matches the address in the RA register. • The calling address is within the range of values of the A1 and RA registers. **NOTE:** For the RAM bit to be set to 1 correctly, C1[IICIE] must be set to 1. Writing the C1 register with any value clears this bit to 0. 0 Not addressed 1 Addressed as a slave
2 SRW	Slave Read/Write

Table continues on the next page...

I2Cx_S field descriptions (continued)

Field	Description
	When addressed as a slave, SRW indicates the value of the R/$\overline{W}$ command bit of the calling address sent to the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IICIF	Interrupt Flag This bit sets when an interrupt is pending. This bit must be cleared by software by writing 1 to it, such as in the interrupt routine. One of the following events can set this bit: • One byte transfer, including ACK/NACK bit, completes if FACK is 0. An ACK or NACK is sent on the bus by writing 0 or 1 to TXAK after this bit is set in receive mode. • One byte transfer, excluding ACK/NACK bit, completes if FACK is 1. • Match of slave address to calling address including primary slave address, range slave address, alert response address, second slave address, or general call address. • Arbitration lost • In SMBus mode, any timeouts except SCL and SDA high timeouts • I2C bus stop or start detection if the SSIE bit in the Input Glitch Filter register is 1 **NOTE:** To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF bit in the Input Glitch Filter register by writing 1 to it, and then clear the IICIF bit. If this sequence is reversed, the IICIF bit is asserted again. 0 No interrupt pending 1 Interrupt pending
0 RXAK	Receive Acknowledge 0 Acknowledge signal was received after the completion of one byte of data transmission on the bus 1 No acknowledge signal detected

45.4.5 I2C Data I/O register (I2Cx_D)

Address: 4006_6000h base + 4h offset = 4006_6004h

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

I2Cx_D field descriptions

Field	Description
DATA	Data In master transmit mode, when data is written to this register, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data. **NOTE:** When making the transition out of master receive mode, switch the I2C mode before reading the Data register to prevent an inadvertent initiation of a master receive data transfer. In slave mode, the same functions are available after an address match occurs.

I2Cx_D field descriptions (continued)

Field	Description
	The C1[TX] bit must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For example, if the I2C module is configured for master transmit but a master receive is desired, reading the Data register does not initiate the receive. Reading the Data register returns the last byte received while the I2C module is configured in master receive or slave receive mode. The Data register does not reflect every byte that is transmitted on the I2C bus, and neither can software verify that a byte has been written to the Data register correctly by reading it back. In master transmit mode, the first byte of data written to the Data register following assertion of MST (start bit) or assertion of RSTA (repeated start bit) is used for the address transfer and must consist of the calling address (in bits 7-1) concatenated with the required R/$\overline{W}$ bit (in position bit 0).

45.4.6 I2C Control Register 2 (I2Cx_C2)

Address: 4006_6000h base + 5h offset = 4006_6005h

Bit	7	6	5	4	3	2	1	0
Read Write	GCAEN	ADEXT	HDRS	SBRC	RMEN	AD[10:8]		
Reset	0	0	0	0	0	0	0	0

I2Cx_C2 field descriptions

Field	Description
7 GCAEN	General Call Address Enable Enables general call address. 0 Disabled 1 Enabled
6 ADEXT	Address Extension Controls the number of bits used for the slave address. 0 7-bit address scheme 1 10-bit address scheme
5 HDRS	High Drive Select Controls the drive capability of the I2C pads. 0 Normal drive mode 1 High drive mode
4 SBRC	Slave Baud Rate Control Enables independent slave mode baud rate at maximum frequency, which forces clock stretching on SCL in very fast I2C modes. To a slave, an example of a "very fast" mode is when the master transfers at 40 kbit/s but the slave can capture the master's data at only 10 kbit/s. 0 The slave baud rate follows the master baud rate and clock stretching may occur 1 Slave baud rate is independent of the master baud rate

Table continues on the next page...

I2Cx_C2 field descriptions (continued)

Field	Description
3 RMEN	Range Address Matching Enable This bit controls the slave address matching for addresses between the values of the A1 and RA registers. When this bit is set, a slave address matching occurs for any address greater than the value of the A1 register and less than or equal to the value of the RA register. 0 Range mode disabled. No address matching occurs for an address within the range of values of the A1 and RA registers. 1 Range mode enabled. Address matching occurs when a slave receives an address within the range of values of the A1 and RA registers.
AD[10:8]	Slave Address Contains the upper three bits of the slave address in the 10-bit address scheme. This field is valid only while the ADEXT bit is set.

45.4.7 I2C Programmable Input Glitch Filter Register (I2Cx_FLT)

Address: 4006_6000h base + 6h offset = 4006_6006h

Bit	7	6	5	4	3	2	1	0
Read	SHEN	STOPF	SSIE	STARTF	FLT			
Write		w1c		w1c				
Reset	0	0	0	0	0	0	0	0

I2Cx_FLT field descriptions

Field	Description
7 SHEN	Stop Hold Enable Set this bit to hold off entry to stop mode when any data transmission or reception is occurring. The following scenario explains the holdoff functionality: 1. The I2C module is configured for a basic transfer, and the SHEN bit is set to 1. 2. A transfer begins. 3. The MCU signals the I2C module to enter stop mode. 4. The byte currently being transferred, including both address and data, completes its transfer. 5. The I2C slave or master acknowledges that the in-transfer byte completed its transfer and acknowledges the request to enter stop mode. 6. After receiving the I2C module's acknowledgment of the request to enter stop mode, the MCU determines whether to shut off the I2C module's clock. If the SHEN bit is set to 1 and the I2C module is in an idle or disabled state when the MCU signals to enter stop mode, the module immediately acknowledges the request to enter stop mode. If SHEN is cleared to 0 and the overall data transmission or reception that was suspended by stop mode entry was incomplete: To resume the overall transmission or reception after the MCU exits stop mode, software must reinitialize the transfer by resending the address of the slave. If the I2C Control Register 1's IICIE bit was set to 1 before the MCU entered stop mode, system software will receive the interrupt triggered by the I2C Status Register's TCF bit after the MCU wakes from the stop mode.

Table continues on the next page...

I2Cx_FLT field descriptions (continued)

Field	Description
	0 Stop holdoff is disabled. The MCU's entry to stop mode is not gated. 1 Stop holdoff is enabled.
6 STOPF	I2C Bus Stop Detect Flag Hardware sets this bit when the I2C bus's stop status is detected. The STOPF bit must be cleared by writing 1 to it. **NOTE:** The stop flag is only for the matched slave devices, therefore the master will not respond for it. 0 No stop happens on I2C bus 1 Stop detected on I2C bus
5 SSIE	I2C Bus Stop or Start Interrupt Enable This bit enables the interrupt for I2C bus stop or start detection. **NOTE:** To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF bit by writing 1 to it, and then clear the IICIF bit in the status register. If this sequence is reversed, the IICIF bit is asserted again. 0 Stop or start detection interrupt is disabled 1 Stop or start detection interrupt is enabled
4 STARTF	I2C Bus Start Detect Flag Hardware sets this bit when the I2C bus's start status is detected. The STARTF bit must be cleared by writing 1 to it. 0 No start happens on I2C bus 1 Start detected on I2C bus
FLT	I2C Programmable Filter Factor Controls the width of the glitch, in terms of I2C module clock cycles, that the filter must absorb. For any glitch whose size is less than or equal to this width setting, the filter does not allow the glitch to pass. 0h No filter/bypass 1-Fh Filter glitches up to width of *n* I2C module clock cycles, where *n*=1-15d

45.4.8 I2C Range Address register (I2Cx_RA)

Address: 4006_6000h base + 7h offset = 4006_6007h

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

I2Cx_RA field descriptions

Field	Description
7–1 RAD	Range Slave Address

Table continues on the next page...

I2Cx_RA field descriptions (continued)

Field	Description
	This field contains the slave address to be used by the I2C module. The field is used in the 7-bit address scheme. If I2C_C2[RMEN] is set to 1, any nonzero value write enables this register. This register value can be considered as a maximum boundary in the range matching mode.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

45.4.9 I2C SMBus Control and Status register (I2Cx_SMB)

NOTE

When the SCL and SDA signals are held high for a length of time greater than the high timeout period, the SHTF1 flag sets. Before reaching this threshold, while the system is detecting how long these signals are being held high, a master assumes that the bus is free. However, the SHTF1 bit is set to 1 in the bus transmission process with the idle bus state.

NOTE

When the TCKSEL bit is set, there is no need to monitor the SHTF1 bit because the bus speed is too high to match the protocol of SMBus.

Address: 4006_6000h base + 8h offset = 4006_6008h

Bit	7	6	5	4	3	2	1	0
Read	FACK	ALERTEN	SIICAEN	TCKSEL	SLTF	SHTF1	SHTF2	SHTF2IE
Write					w1c		w1c	
Reset	0	0	0	0	0	0	0	0

I2Cx_SMB field descriptions

Field	Description
7 FACK	Fast NACK/ACK Enable For SMBus packet error checking, the CPU must be able to issue an ACK or NACK according to the result of receiving data byte. 0 An ACK or NACK is sent on the following receiving data byte 1 Writing 0 to TXAK after receiving a data byte generates an ACK. Writing 1 to TXAK after receiving a data byte generates a NACK.
6 ALERTEN	SMBus Alert Response Address Enable Enables or disables SMBus alert response address matching. **NOTE:** After the host responds to a device that used the alert response address, you must use software to put the device's address on the bus. The alert protocol is described in the SMBus specification.

Table continues on the next page...

I2Cx_SMB field descriptions (continued)

Field	Description
	0 SMBus alert response address matching is disabled 1 SMBus alert response address matching is enabled
5 SIICAEN	Second I2C Address Enable Enables or disables SMBus device default address. 0 I2C address register 2 matching is disabled 1 I2C address register 2 matching is enabled
4 TCKSEL	Timeout Counter Clock Select Selects the clock source of the timeout counter. 0 Timeout counter counts at the frequency of the I2C module clock / 64 1 Timeout counter counts at the frequency of the I2C module clock
3 SLTF	SCL Low Timeout Flag This bit is set when the SLT register (consisting of the SLTH and SLTL registers) is loaded with a non-zero value (LoValue) and an SCL low timeout occurs. Software clears this bit by writing a logic 1 to it. **NOTE:** The low timeout function is disabled when the SLT register's value is 0. 0 No low timeout occurs 1 Low timeout occurs
2 SHTF1	SCL High Timeout Flag 1 This read-only bit sets when SCL and SDA are held high more than clock × LoValue / 512, which indicates the bus is free. This bit is cleared automatically. 0 No SCL high and SDA high timeout occurs 1 SCL high and SDA high timeout occurs
1 SHTF2	SCL High Timeout Flag 2 This bit sets when SCL is held high and SDA is held low more than clock × LoValue / 512. Software clears this bit by writing 1 to it. 0 No SCL high and SDA low timeout occurs 1 SCL high and SDA low timeout occurs
0 SHTF2IE	SHTF2 Interrupt Enable Enables SCL high and SDA low timeout interrupt. 0 SHTF2 interrupt is disabled 1 SHTF2 interrupt is enabled

45.4.10 I2C Address Register 2 (I2Cx_A2)

Address: 4006_6000h base + 9h offset = 4006_6009h

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

I2Cx_A2 field descriptions

Field	Description
7–1 SAD	SMBus Address Contains the slave address used by the SMBus. This field is used on the device default address or other related addresses.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

45.4.11 I2C SCL Low Timeout Register High (I2Cx_SLTH)

Address: 4006_6000h base + Ah offset = 4006_600Ah

Bit	7	6	5	4	3	2	1	0
Read	SSLT[15:8]							
Write								
Reset	0	0	0	0	0	0	0	0

I2Cx_SLTH field descriptions

Field	Description
SSLT[15:8]	SSLT[15:8] Most significant byte of SCL low timeout value that determines the timeout period of SCL low.

45.4.12 I2C SCL Low Timeout Register Low (I2Cx_SLTL)

Address: 4006_6000h base + Bh offset = 4006_600Bh

Bit	7	6	5	4	3	2	1	0
Read	SSLT[7:0]							
Write								
Reset	0	0	0	0	0	0	0	0

I2Cx_SLTL field descriptions

Field	Description
SSLT[7:0]	SSLT[7:0]

I2Cx_SLTL field descriptions (continued)

Field	Description
	Least significant byte of SCL low timeout value that determines the timeout period of SCL low.

45.5 Functional description

This section provides a comprehensive functional description of the I2C module.

45.5.1 I2C protocol

The I2C bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfers.

All devices connected to it must have open drain or open collector outputs. A logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors depends on the system.

Normally, a standard instance of communication is composed of four parts:

1. START signal
2. Slave address transmission
3. Data transfer
4. STOP signal

The STOP signal should not be confused with the CPU STOP instruction. The following figure illustrates I2C bus system communication.

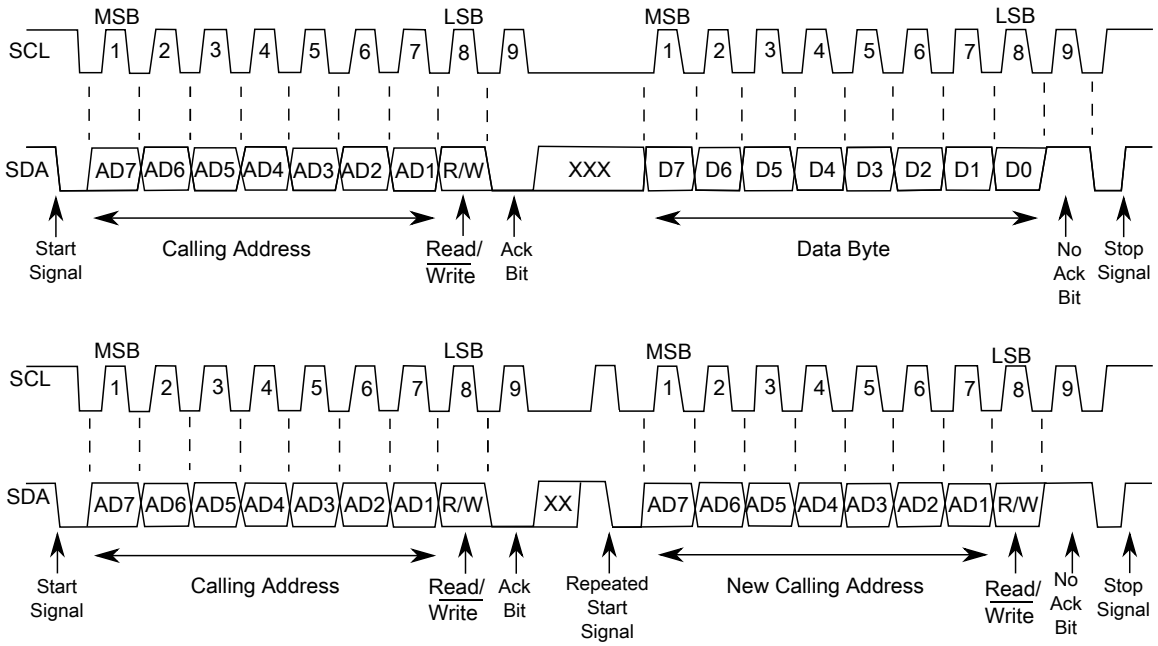


Figure 45-2. I2C bus transmission signals

45.5.1.1 START signal

The bus is free when no master device is engaging the bus (both SCL and SDA are high). When the bus is free, a master may initiate communication by sending a START signal. A START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer—each data transfer might contain several bytes of data—and brings all slaves out of their idle states.

45.5.1.2 Slave address transmission

Immediately after the START signal, the first byte of a data transfer is the slave address transmitted by the master. This address is a 7-bit calling address followed by an R/$\overline{W}$ bit. The R/$\overline{W}$ bit tells the slave the desired direction of data transfer.

- 1 = Read transfer: The slave transmits data to the master
- 0 = Write transfer: The master transmits data to the slave

Only the slave with a calling address that matches the one transmitted by the master responds by sending an acknowledge bit. The slave sends the acknowledge bit by pulling SDA low at the ninth clock.

No two slaves in the system can have the same address. If the I2C module is the master, it must not transmit an address that is equal to its own slave address. The I2C module cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the I2C module reverts to slave mode and operates correctly even if it is being addressed by another master.

45.5.1.3 Data transfers

When successful slave addressing is achieved, data transfer can proceed on a byte-by-byte basis in the direction specified by the R/$\overline{W}$ bit sent by the calling master.

All transfers that follow an address cycle are referred to as data transfers, even if they carry subaddress information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low. Data must be held stable while SCL is high. There is one clock pulse on SCL for each data bit, and the MSB is transferred first. Each data byte is followed by a ninth (acknowledge) bit, which is signaled from the receiving device by pulling SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit, the slave must leave SDA high. The master interprets the failed acknowledgement as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets it as an end to data transfer and releases the SDA line.

In the case of a failed acknowledgement by either the slave or master, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a STOP signal.
- Commences a new call by generating a repeated START signal.

45.5.1.4 STOP signal

The master can terminate the communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDA while SCL is asserted.

The master can generate a STOP signal even if the slave has generated an acknowledgement, at which point the slave must release the bus.

45.5.1.5 Repeated START signal

The master may generate a START signal followed by a calling command without generating a STOP signal first. This action is called a repeated START. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

45.5.1.6 Arbitration procedure

The I2C bus is a true multimaster bus that allows more than one master to be connected on it.

If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. The bus clock's low period is equal to the longest clock low period, and the high period is equal to the shortest one among the masters.

The relative priority of the contending masters is determined by a data arbitration procedure. A bus master loses arbitration if it transmits logic level 1 while another master transmits logic level 0. The losing masters immediately switch to slave receive mode and stop driving SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets a status bit to indicate the loss of arbitration.

45.5.1.7 Clock synchronization

Because wire AND logic is performed on SCL, a high-to-low transition on SCL affects all devices connected on the bus. The devices start counting their low period and, after a device's clock has gone low, that device holds SCL low until the clock reaches its high state. However, the change of low to high in this device clock might not change the state of SCL if another device clock is still within its low period. Therefore, the synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time; see the following diagram. When all applicable devices have counted off their low period, the synchronized clock SCL is released and pulled high. Afterward there is no difference between the device clocks and the state of SCL, and all devices start counting their high periods. The first device to complete its high period pulls SCL low again.

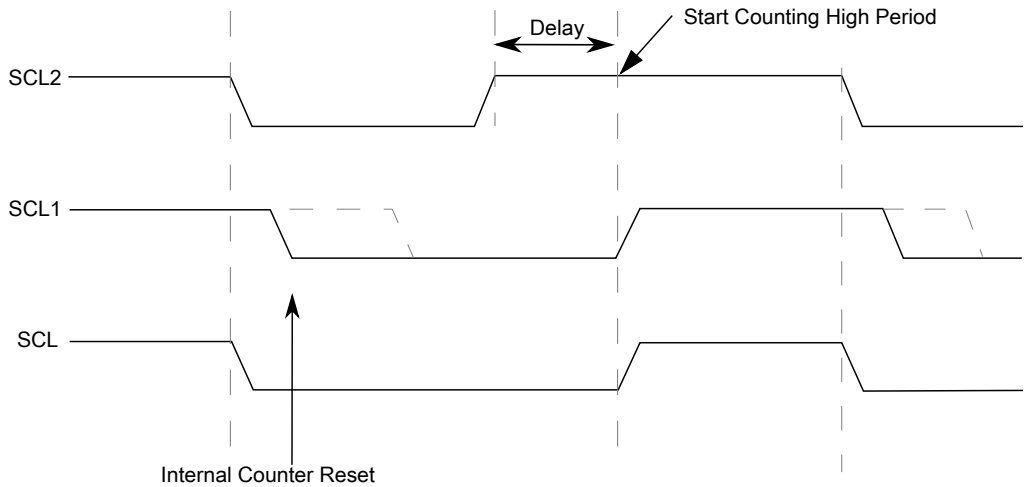


Figure 45-3. I2C clock synchronization

45.5.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. A slave device may hold SCL low after completing a single byte transfer (9 bits). In this case, it halts the bus clock and forces the master clock into wait states until the slave releases SCL.

45.5.1.9 Clock stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master drives SCL low, a slave can drive SCL low for the required period and then release it. If the slave's SCL low period is greater than the master's SCL low period, the resulting SCL bus signal's low period is stretched. In other words, the SCL bus signal's low period is increased to be the same length as the slave's SCL low period.

45.5.1.10 I2C divider and hold values

NOTE

For some cases on some devices, the SCL divider value may vary by ±2 or ±4 when ICR's value ranges from 00h to 0Fh. These potentially varying SCL divider values are highlighted in the following table.

Table 45-2. I2C divider and hold values

ICR (hex)	SCL divider	SDA hold value	SCL hold (start) value	SCL hold (stop) value	ICR (hex)	SCL divider (clocks)	SDA hold (clocks)	SCL hold (start) value	SCL hold (stop) value
00	20	7	6	11	20	160	17	78	81
01	22	7	7	12	21	192	17	94	97
02	24	8	8	13	22	224	33	110	113
03	26	8	9	14	23	256	33	126	129
04	28	9	10	15	24	288	49	142	145
05	30	9	11	16	25	320	49	158	161
06	34	10	13	18	26	384	65	190	193
07	40	10	16	21	27	480	65	238	241
08	28	7	10	15	28	320	33	158	161
09	32	7	12	17	29	384	33	190	193
0A	36	9	14	19	2A	448	65	222	225
0B	40	9	16	21	2B	512	65	254	257
0C	44	11	18	23	2C	576	97	286	289
0D	48	11	20	25	2D	640	97	318	321
0E	56	13	24	29	2E	768	129	382	385
0F	68	13	30	35	2F	960	129	478	481
10	48	9	18	25	30	640	65	318	321
11	56	9	22	29	31	768	65	382	385
12	64	13	26	33	32	896	129	446	449
13	72	13	30	37	33	1024	129	510	513
14	80	17	34	41	34	1152	193	574	577
15	88	17	38	45	35	1280	193	638	641
16	104	21	46	53	36	1536	257	766	769
17	128	21	58	65	37	1920	257	958	961
18	80	9	38	41	38	1280	129	638	641
19	96	9	46	49	39	1536	129	766	769
1A	112	17	54	57	3A	1792	257	894	897
1B	128	17	62	65	3B	2048	257	1022	1025
1C	144	25	70	73	3C	2304	385	1150	1153
1D	160	25	78	81	3D	2560	385	1278	1281
1E	192	33	94	97	3E	3072	513	1534	1537
1F	240	33	118	121	3F	3840	513	1918	1921

45.5.2 10-bit address

For 10-bit addressing, 0x11110 is used for the first 5 bits of the first address byte. Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing.

45.5.2.1 Master-transmitter addresses a slave-receiver

The transfer direction is not changed. When a 10-bit address follows a START condition, each slave compares the first 7 bits of the first byte of the slave address (11110XX) with its own address and tests whether the eighth bit (R/$\overline{W}$ direction bit) is 0. It is possible that more than one device finds a match and generates an acknowledge (A1). Each slave that finds a match compares the 8 bits of the second byte of the slave address with its own address, but only one slave finds a match and generates an acknowledge (A2). The matching slave remains addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

Table 45-3. Master-transmitter addresses slave-receiver with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Data	A	...	Data	A/A	P

After the master-transmitter has sent the first byte of the 10-bit address, the slave-receiver sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

45.5.2.2 Master-receiver addresses a slave-transmitter

The transfer direction is changed after the second R/$\overline{W}$ bit. Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks whether the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and it tests whether the eighth (R/$\overline{W}$) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

After a repeated START condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them are addressed because R/$\overline{W}$ = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

Table 45-4. Master-receiver addresses a slave-transmitter with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Sr	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 1	A3	Data	A	...	Data	A	P

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

45.5.3 Address matching

All received addresses can be requested in 7-bit or 10-bit address format.

- AD[7:1] in Address Register 1, which contains the I2C primary slave address, always participates in the address matching process. It provides a 7-bit address.
- If the ADEXT bit is set, AD[10:8] in Control Register 2 participates in the address matching process. It extends the I2C primary slave address to a 10-bit address.

Additional conditions that affect address matching include:

- If the GCAEN bit is set, general call participates the address matching process.
- If the ALERTEN bit is set, alert response participates the address matching process.
- If the SIICAEN bit is set, Address Register 2 participates in the address matching process.
- If the RMEN bit is set, when the Range Address register is programmed to a nonzero value, any address within the range of values of Address Register 1 (excluded) and the Range Address register (included) participates in the address matching process. The Range Address register must be programmed to a value greater than the value of Address Register 1.

When the I2C module responds to one of these addresses, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the Data register after the first byte transfer to determine that the address is matched.

45.5.4 System management bus specification

SMBus provides a control bus for system and power management related tasks. A system can use SMBus to pass messages to and from devices instead of tripping individual control lines.

Removing the individual control lines reduces pin count. Accepting messages ensures future expandability. With the system management bus, a device can provide manufacturer information, tell the system what its model/part number is, save its state for a suspend event, report different types of errors, accept control parameters, and return its status.

45.5.4.1 Timeouts

The $T_{TIMEOUT,MIN}$ parameter allows a master or slave to conclude that a defective device is holding the clock low indefinitely or a master is intentionally trying to drive devices off the bus. The slave device must release the bus (stop driving the bus and let SCL and SDA float high) when it detects any single clock held low longer than $T_{TIMEOUT,MIN}$. Devices that have detected this condition must reset their communication and be able to receive a new START condition within the timeframe of $T_{TIMEOUT,MAX}$.

SMBus defines a clock low timeout, $T_{TIMEOUT}$, of 35 ms, specifies $T_{LOW:SEXT}$ as the cumulative clock low extend time for a slave device, and specifies $T_{LOW:MEXT}$ as the cumulative clock low extend time for a master device.

45.5.4.1.1 SCL low timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than a timeout value condition. Devices that have detected the timeout condition must reset the communication. When the I2C module is an active master, if it detects that SMBCLK low has exceeded the value of $T_{TIMEOUT,MIN}$, it must generate a stop condition within or after the current data byte in the transfer process. When the I2C module is a slave, if it detects the $T_{TIMEOUT,MIN}$ condition, it resets its communication and is then able to receive a new START condition.

45.5.4.1.2 SCL high timeout

When the I2C module has determined that the SMBCLK and SMBDAT signals have been high for at least $T_{HIGH:MAX}$, it assumes that the bus is idle.

A HIGH timeout occurs after a START condition appears on the bus but before a STOP condition appears on the bus. Any master detecting this scenario can assume the bus is free when either of the following occurs:

- SHTF1 rises.
- The BUSY bit is high and SHTF1 is high.

When the SMBDAT signal is low and the SMBCLK signal is high for a period of time, another kind of timeout occurs. The time period must be defined in software. SHTF2 is used as the flag when the time limit is reached. This flag is also an interrupt resource, so it triggers IICIF.

45.5.4.1.3 CSMBCLK TIMEOUT MEXT and CSMBCLK TIMEOUT SEXT

The following figure illustrates the definition of the timeout intervals $T_{LOW:SEXT}$ and $T_{LOW:MEXT}$. When in master mode, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:MEXT}$ within a byte, where each byte is defined as START-to-ACK, ACK-to-ACK, or ACK-to-STOP. When CSMBCLK TIMEOUT MEXT occurs, SMBus MEXT rises and also triggers the SLTF.

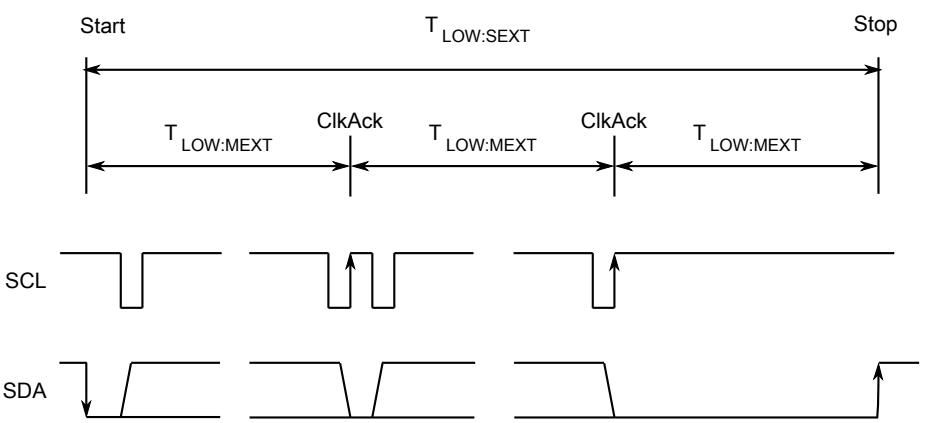


Figure 45-4. Timeout measurement intervals

A master is allowed to abort the transaction in progress to any slave that violates the $T_{LOW:SEXT}$ or $T_{TIMEOUT,MIN}$ specifications. To abort the transaction, the master issues a STOP condition at the conclusion of the byte transfer in progress. When a slave, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:SEXT}$ during any message from the initial START to the STOP. When CSMBCLK TIMEOUT SEXT occurs, SEXT rises and also triggers SLTF.

NOTE

CSMBCLK TIMEOUT SEXT and CSMBCLK TIMEOUT MEXT are optional functions that are implemented in the second step.

45.5.4.2 FAST ACK and NACK

To improve reliability and communication robustness, implementation of packet error checking (PEC) by SMBus devices is optional for SMBus devices but required for devices participating in and only during the address resolution protocol (ARP) process. The PEC is a CRC-8 error checking byte, calculated on all the message bytes. The PEC is appended to the message by the device that supplied the last data byte. If the PEC is present but not correct, a NACK is issued by the receiver. Otherwise an ACK is issued. To calculate the CRC-8 by software, this module can hold the SCL line low after receiving the eighth SCL (8th bit) if this byte is a data byte. So software can determine whether an ACK or NACK should be sent to the bus by setting or clearing the TXAK bit if the FACK (fast ACK/NACK enable) bit is enabled.

SMBus requires a device always to acknowledge its own address, as a mechanism to detect the presence of a removable device (such as a battery or docking station) on the bus. In addition to indicating a slave device busy condition, SMBus uses the NACK mechanism to indicate the reception of an invalid command or invalid data. Because such a condition may occur on the last byte of the transfer, SMBus devices are required to have the ability to generate the not acknowledge after the transfer of each byte and before the completion of the transaction. This requirement is important because SMBus does not provide any other resend signaling. This difference in the use of the NACK signaling has implications on the specific implementation of the SMBus port, especially in devices that handle critical system data such as the SMBus host and the SBS components.

NOTE

In the last byte of master receive slave transmit mode, the master must send a NACK to the bus, so FACK must be switched off before the last byte transmits.

45.5.5 Resets

The I2C module is disabled after a reset. The I2C module cannot cause a core reset.

45.5.6 Interrupts

The I2C module generates an interrupt when any of the events in the table found here occur, provided that the IICIE bit is set.

The interrupt is driven by the IICIF bit (of the I2C Status Register) and masked with the IICIE bit (of the I2C Control Register 1). The IICIF bit must be cleared (by software) by writing 1 to it in the interrupt routine. The SMBus timeouts interrupt is driven by SLTF and masked with the IICIE bit. The SLTF bit must be cleared by software by writing 1 to it in the interrupt routine. You can determine the interrupt type by reading the Status Register.

NOTE

In master receive mode, the FACK bit must be set to zero before the last byte transfer.

Table 45-5. Interrupt summary

Interrupt source	Status	Flag	Local enable
Complete 1-byte transfer	TCF	IICIF	IICIE
Match of received calling address	IAAS	IICIF	IICIE
Arbitration lost	ARBL	IICIF	IICIE
I^2C bus stop detection	STOPF	IICIF	IICIE & SSIE
I^2C bus start detection	STARTF	IICIF	IICIE & SSIE
SMBus SCL low timeout	SLTF	IICIF	IICIE
SMBus SCL high SDA low timeout	SHTF2	IICIF	IICIE & SHTF2IE
Wakeup from stop or wait mode	IAAS	IICIF	IICIE & WUEN

45.5.6.1 Byte transfer interrupt

The Transfer Complete Flag (TCF) bit is set at the falling edge of the ninth clock to indicate the completion of a byte and acknowledgement transfer. When FACK is enabled, TCF is then set at the falling edge of eighth clock to indicate the completion of byte.

45.5.6.2 Address detect interrupt

When the calling address matches the programmed slave address (I2C Address Register) or when the GCAEN bit is set and a general call is received, the IAAS bit in the Status Register is set. The CPU is interrupted, provided the IICIE bit is set. The CPU must check the SRW bit and set its Tx mode accordingly.

45.5.6.3 Stop Detect Interrupt

When the stop status is detected on the I^2C bus, the STOPF bit is set to 1. The CPU is interrupted, provided the IICIE and SSIE bits are both set to 1.

45.5.6.4 Exit from low-power/stop modes

The slave receive input detect circuit and address matching feature are still active on low power modes (wait and stop). An asynchronous input matching slave address or general call address brings the CPU out of low power/stop mode if the interrupt is not masked. Therefore, TCF and IAAS both can trigger this interrupt.

45.5.6.5 Arbitration lost interrupt

The I2C is a true multimaster bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, the relative priority of the contending masters is determined by a data arbitration procedure. The I2C module asserts the arbitration-lost interrupt when it loses the data arbitration process and the ARBL bit in the Status Register is set.

Arbitration is lost in the following circumstances:

1. SDA is sampled as low when the master drives high during an address or data transmit cycle.
2. SDA is sampled as low when the master drives high during the acknowledge bit of a data receive cycle.
3. A START cycle is attempted when the bus is busy.
4. A repeated START cycle is requested in slave mode.
5. A STOP condition is detected when the master did not request it.

The ARBL bit must be cleared (by software) by writing 1 to it.

45.5.6.6 Timeout interrupt in SMBus

When the IICIE bit is set, the I2C module asserts a timeout interrupt (outputs SLTF and SHTF2) upon detection of any of the mentioned timeout conditions, with one exception. The SCL high and SDA high TIMEOUT mechanism must not be used to influence the timeout interrupt output, because this timeout indicates an idle condition on the bus. SHTF1 rises when it matches the SCL high and SDA high TIMEOUT and falls automatically just to indicate the bus status. The SHTF2's timeout period is the same as that of SHTF1, which is short compared to that of SLTF, so another control bit, SHTF2IE, is added to enable or disable it.

45.5.7 Programmable input glitch filter

An I2C glitch filter has been added outside legacy I2C modules but within the I2C package. This filter can absorb glitches on the I2C clock and data lines for the I2C module.

The width of the glitch to absorb can be specified in terms of the number of (half) I2C module clock cycles. A single Programmable Input Glitch Filter control register is provided. Effectively, any down-up-down or up-down-up transition on the data line that occurs within the number of clock cycles programmed in this register is ignored by the I2C module. The programmer must specify the size of the glitch (in terms of I2C module clock cycles) for the filter to absorb and not pass.

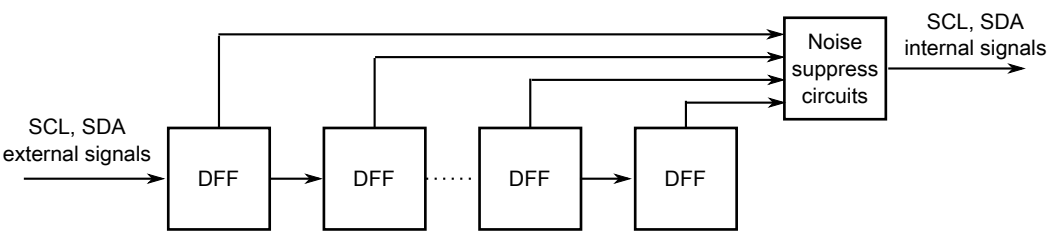


Figure 45-5. Programmable input glitch filter diagram

45.5.8 Address matching wake-up

When a primary, range, or general call address match occurs when the I2C module is in slave receive mode, the MCU wakes from a low power mode where no peripheral bus is running.

Data sent on the bus that is the same as a target device address might also wake the target MCU.

After the address matching IAAS bit is set, an interrupt is sent at the end of address matching to wake the core. The IAAS bit must be cleared after the clock recovery.

NOTE

After the system recovers and is in Run mode, restart the I2C module if it is needed to transfer packets. To avoid I2C transfer problems resulting from the situation, firmware should prevent the MCU execution of a STOP instruction when the I2C module is in the middle of a transfer unless the Stop mode holdoff feature is used during this period (set FLT[SHEN] to 1).

45.5.9 DMA support

If the DMAEN bit is cleared and the IICIE bit is set, an interrupt condition generates an interrupt request.

If the DMAEN bit is set and the IICIE bit is set, an interrupt condition generates a DMA request instead. DMA requests are generated by the transfer complete flag (TCF).

If the DMAEN bit is set, only the TCF initiates a DMA request. All other events generate CPU interrupts.

NOTE

Before the last byte of master receive mode, TXAK must be set to send a NACK after the last byte's transfer. Therefore, the DMA must be disabled before the last byte's transfer.

NOTE

In 10-bit address mode transmission, the addresses to send occupy 2–3 bytes. During this transfer period, the DMA must be disabled because the C1 register is written to send a repeat start or to change the transfer direction.

45.6 Initialization/application information

Module Initialization (Slave)

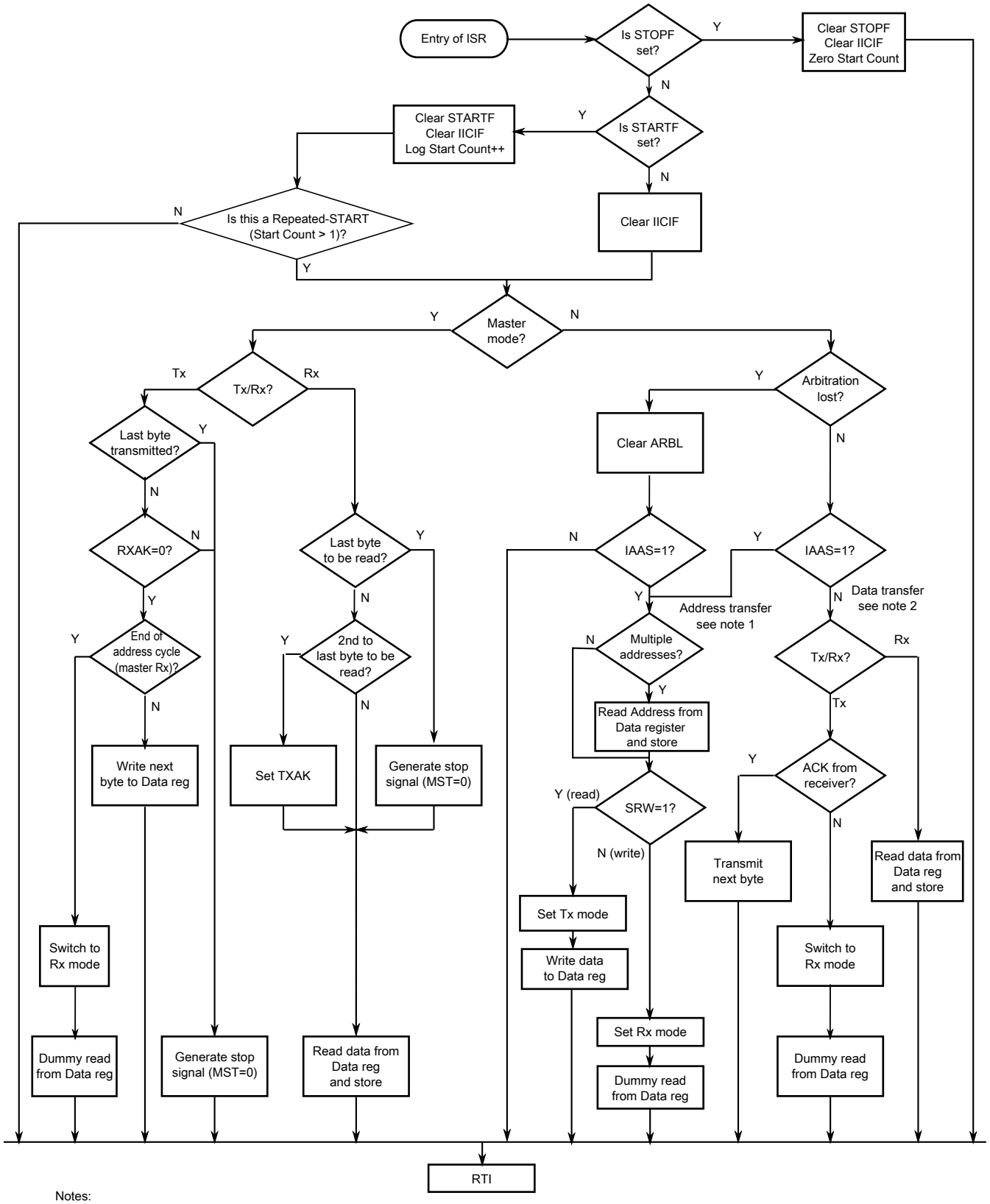
1. Write: Control Register 2
   - to enable or disable general call
   - to select 10-bit or 7-bit addressing mode
2. Write: Address Register 1 to set the slave address

3. Write: Control Register 1 to enable the I2C module and interrupts
4. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
5. Initialize RAM variables used to achieve the routine shown in the following figure

Module Initialization (Master)

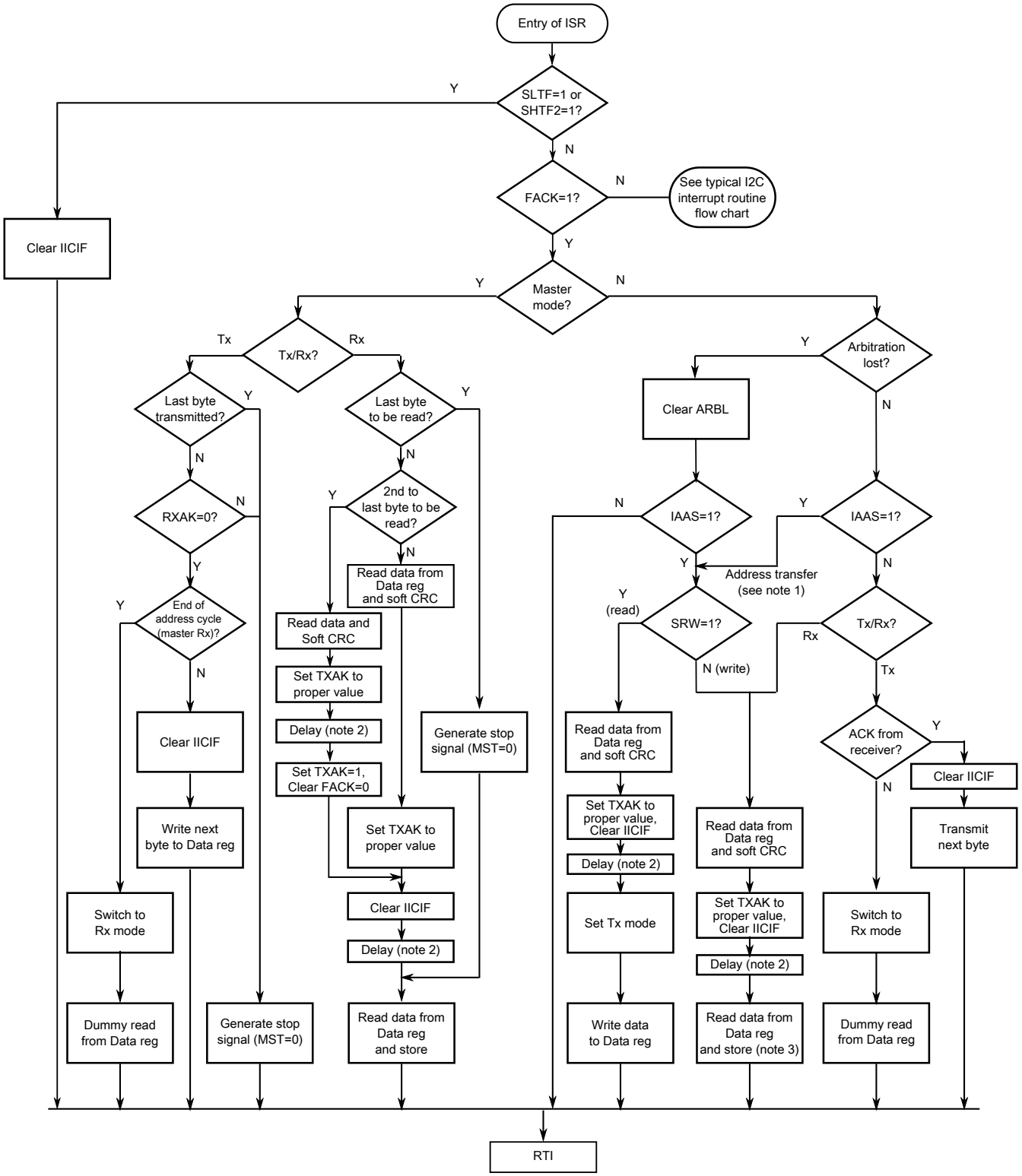
1. Write: Frequency Divider register to set the I2C baud rate (see example in description of ICR)
2. Write: Control Register 1 to enable the I2C module and interrupts
3. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
4. Initialize RAM variables used to achieve the routine shown in the following figure
5. Write: Control Register 1 to enable TX
6. Write: Control Register 1 to enable MST (master mode)
7. Write: Data register with the address of the target slave (the LSB of this byte determines whether the communication is master receive or transmit)

The routine shown in the following figure encompasses both master and slave I2C operations. For slave operation, an incoming I2C message that contains the proper address begins I2C communication. For master operation, communication must be initiated by writing the Data register. An example of an I2C driver which implements many of the steps described here is available in AN4342: Using the Inter-Integrated Circuit on ColdFire+ and Kinetis .



Notes:
1. If general call is enabled, check to determine if the received address is a general call address (0x00).
   If the received address is a general call address, the general call must be handled by user software.
2. When 10-bit addressing addresses a slave, the slave sees an interrupt following the first byte of the extended address.
   Ensure that for this interrupt, the contents of the Data register are ignored and not treated as a valid data transfer.

Figure 45-6. Typical I2C interrupt routine



Notes:
1. If general call or SIICAEN is enabled, check to determine if the received address is a general call address (0x00) or an SMBus device default address. In either case, they must be handled by user software.
2. In receive mode, one bit time delay may be needed before the first and second data reading, to wait for the possible longest time period (in worst case) of the 9th SCL cycle.
3. This read is a dummy read in order to reset the SMBus receiver state machine.

Figure 45-7. Typical I2C SMBus interrupt routine

Chapter 46
Universal Asynchronous Receiver/Transmitter (UART) / FlexSCI

46.1 Chip-specific UART information

46.1.1 UART configuration information

This chip contains two UART modules. This section describes how each module is configured on this device.

1. Standard features of all UARTs:
   - RS-485 support
   - Hardware flow control (RTS/CTS)
   - 9-bit UART to support address mark with parity
   - MSB/LSB configuration on data
2. UART0 and UART1 are clocked from the fast bus clock. The maximum baud rate is 1/16 of related source clock frequency.
3. UART0 contains 8-entry transmit and 8-entry receive FIFOs
4. All other UARTs contain a 1-entry transmit and receive FIFOs

46.1.2 UART signals

Signal	I/O	Connected to
UART0		
Receive complete	Output	DMA_MUX source 2
Transmit complete	Output	DMA_MUX source 3
UART0RXSRC	Input	UART0_RXpin or CMP0_OUT or CMP1_OUT
UART0TXSRC	Input	UART0_TX pin or UART0_TX pin modulated with FTM1 channel 0 output
UART1		
Receive complete	Output	DMA_MUX source 4

Table continues on the next page...

Signal	I/O	Connected to
Transmit complete	Output	DMA_MUX source 5
UART1RXSRC	Input	UART1_RXpin or CMP0_OUT or CMP1_OUT
UART1TXSRC	Input	UART1_TX pin or UART1_TX pin modulated with FTM1 channel 0 output

46.1.3 UART wakeup

The UART can be configured to generate an interrupt/wakeup on the first active edge that it receives.

46.1.4 UART interrupts

The UART has multiple sources of interrupt requests. However, some of these sources are OR'd together to generate a single interrupt request. See below for the mapping of the individual interrupt sources to the interrupt request:

The status interrupt combines the following interrupt sources:

Source	UART 0	UART 1
Transmit data empty	x	x
Transmit complete	x	x
Idle line	x	x
Receive data full	x	x
LIN break detect	x	x
RxD pin active edge	x	x

The error interrupt combines the following interrupt sources:

Source	UART 0	UART 1
Receiver overrun	x	x
Noise flag	x	x
Framing error	x	x
Parity error	x	x
Transmitter buffer overflow	x	x
Receiver buffer overflow	x	x
Receiver buffer underflow	x	x

46.2 Introduction

The UART allows asynchronous serial communication with peripheral devices and CPUs.

46.2.1 Features

The UART includes the following features:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 13-bit baud rate selection with /32 fractional divide, based on the module clock frequency
- Programmable 8-bit or 9-bit data format
- Programmable 1 or 2 stop bits in a data frame.
- Separately enabled transmitter and receiver
- Programmable transmitter output polarity
- Programmable receive input polarity
- Up to 16-bit break character transmission.
- 11-bit break character detection option
- Independent FIFO structure for transmit and receive
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Address match feature in the receiver to reduce address mark wakeup ISR overhead
- Ability to select MSB or LSB to be first bit on wire
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Interrupt-driven operation with flags

- Transmitter data buffer at or below watermark
- Transmission complete
- Receiver data buffer at or above watermark
- Idle receiver input
- Receiver data buffer overrun
- Receiver data buffer underflow
- Transmit data buffer overflow
- Noise error
- Framing error
- Parity error
- Active edge on receive pin
- LIN break detect

- Receiver framing error detection
- Hardware parity generation and checking
- 1/16 bit-time noise detection
- DMA interface

46.2.2 Modes of operation

The UART functions in the same way in all the normal modes.

It has the following low power modes:

- Wait mode
- Stop mode

46.2.2.1 Run mode

This is the normal mode of operation.

46.2.2.2 Wait mode

UART operation in the Wait mode depends on the state of the C1[UARTSWAI] field.

- If C1[UARTSWAI] is cleared, and the CPU is in Wait mode, the UART operates normally.
- If C1[UARTSWAI] is set, and the CPU is in Wait mode, the UART clock generation ceases and the UART module enters a power conservation state.

Setting C1[UARTSWAI] does not affect the state of the C2[RE] or C2[TE].

If C1[UARTSWAI] is set, any ongoing transmission or reception stops at the Wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of Wait mode. Bringing the CPU out of Wait mode by reset aborts any ongoing transmission or reception and resets the UART.

46.2.2.3 Stop mode

The UART is inactive during Stop mode for reduced power consumption. The STOP instruction does not affect the UART register states, but the UART module clock is disabled. The UART operation resumes after an external interrupt brings the CPU out of Stop mode. Bringing the CPU out of Stop mode by reset aborts any ongoing transmission or reception and resets the UART.

46.3 UART signal descriptions

The UART signals are shown in the following table.

Table 46-1. UART signal descriptions

Signal	Description	I/O
$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{RTS}}$	Request to send	O
RXD	Receive data	I
TXD	Transmit data	O

46.3.1 Detailed signal descriptions

The detailed signal descriptions of the UART are shown in the following table.

Table 46-2. UART—Detailed signal descriptions

Signal	I/O	Description	
$\overline{\text{CTS}}$	I	Clear to send. Indicates whether the UART can start transmitting data when flow control is enabled.	
		State meaning	Asserted—Data transmission can start. Negated—Data transmission cannot start.
		Timing	Assertion—When transmitting device's $\overline{\text{RTS}}$ asserts. Negation—When transmitting device's $\overline{\text{RTS}}$ deasserts.
$\overline{\text{RTS}}$	O	Request to send. When driven by the receiver, indicates whether the UART is ready to receive data. When driven by the transmitter, can enable an external transceiver during transmission.	
		State meaning	Asserted—When driven by the receiver, ready to receive data. When driven by the transmitter, enable the external transmitter. Negated—When driven by the receiver, not ready to receive data. When driven by the transmitter, disable the external transmitter.
		Timing	Assertion—Can occur at any time; can assert asynchronously to the other input signals. Negation—Can occur at any time; can deassert asynchronously to the other input signals.
RXD	I	Receive data. Serial data input to receiver.	
		State meaning	Whether RXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Sampled at a frequency determined by the module clock divided by the baud rate.
TXD	O	Transmit data. Serial data output from transmitter.	
		State meaning	Whether TXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Driven at the beginning or within a bit time according to the bit encoding method along with other configuration settings. Otherwise, transmissions are independent of reception timing.

46.4 Memory map and registers

This section provides a detailed description of all memory and registers.

Accessing reserved addresses within the memory map results in a transfer error. None of the contents of the implemented addresses are modified as a result of that access.

Only byte accesses are supported.

UART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A000	UART Baud Rate Registers: High (UART0_BDH)	8	R/W	00h	46.4.1/1266
4006_A001	UART Baud Rate Registers: Low (UART0_BDL)	8	R/W	04h	46.4.2/1267
4006_A002	UART Control Register 1 (UART0_C1)	8	R/W	00h	46.4.3/1268
4006_A003	UART Control Register 2 (UART0_C2)	8	R/W	00h	46.4.4/1269
4006_A004	UART Status Register 1 (UART0_S1)	8	R	C0h	46.4.5/1271
4006_A005	UART Status Register 2 (UART0_S2)	8	R/W	00h	46.4.6/1274
4006_A006	UART Control Register 3 (UART0_C3)	8	R/W	00h	46.4.7/1275
4006_A007	UART Data Register (UART0_D)	8	R/W	00h	46.4.8/1277
4006_A008	UART Match Address Registers 1 (UART0_MA1)	8	R/W	00h	46.4.9/1278
4006_A009	UART Match Address Registers 2 (UART0_MA2)	8	R/W	00h	46.4.10/ 1278
4006_A00A	UART Control Register 4 (UART0_C4)	8	R/W	00h	46.4.11/ 1278
4006_A00B	UART Control Register 5 (UART0_C5)	8	R/W	00h	46.4.12/ 1279
4006_A00C	UART Extended Data Register (UART0_ED)	8	R	00h	46.4.13/ 1280
4006_A00D	UART Modem Register (UART0_MODEM)	8	R/W	00h	46.4.14/ 1281
4006_A010	UART FIFO Parameters (UART0_PFIFO)	8	R/W	See section	46.4.15/ 1283
4006_A011	UART FIFO Control Register (UART0_CFIFO)	8	R/W	00h	46.4.16/ 1284
4006_A012	UART FIFO Status Register (UART0_SFIFO)	8	R/W	C0h	46.4.17/ 1285
4006_A013	UART FIFO Transmit Watermark (UART0_TWFIFO)	8	R/W	00h	46.4.18/ 1286
4006_A014	UART FIFO Transmit Count (UART0_TCFIFO)	8	R	00h	46.4.19/ 1287
4006_A015	UART FIFO Receive Watermark (UART0_RWFIFO)	8	R/W	01h	46.4.20/ 1287
4006_A016	UART FIFO Receive Count (UART0_RCFIFO)	8	R	00h	46.4.21/ 1288
4006_B000	UART Baud Rate Registers: High (UART1_BDH)	8	R/W	00h	46.4.1/1266
4006_B001	UART Baud Rate Registers: Low (UART1_BDL)	8	R/W	04h	46.4.2/1267
4006_B002	UART Control Register 1 (UART1_C1)	8	R/W	00h	46.4.3/1268
4006_B003	UART Control Register 2 (UART1_C2)	8	R/W	00h	46.4.4/1269
4006_B004	UART Status Register 1 (UART1_S1)	8	R	C0h	46.4.5/1271
4006_B005	UART Status Register 2 (UART1_S2)	8	R/W	00h	46.4.6/1274
4006_B006	UART Control Register 3 (UART1_C3)	8	R/W	00h	46.4.7/1275
4006_B007	UART Data Register (UART1_D)	8	R/W	00h	46.4.8/1277

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_B008	UART Match Address Registers 1 (UART1_MA1)	8	R/W	00h	46.4.9/1278
4006_B009	UART Match Address Registers 2 (UART1_MA2)	8	R/W	00h	46.4.10/ 1278
4006_B00A	UART Control Register 4 (UART1_C4)	8	R/W	00h	46.4.11/ 1278
4006_B00B	UART Control Register 5 (UART1_C5)	8	R/W	00h	46.4.12/ 1279
4006_B00C	UART Extended Data Register (UART1_ED)	8	R	00h	46.4.13/ 1280
4006_B00D	UART Modem Register (UART1_MODEM)	8	R/W	00h	46.4.14/ 1281
4006_B010	UART FIFO Parameters (UART1_PFIFO)	8	R/W	See section	46.4.15/ 1283
4006_B011	UART FIFO Control Register (UART1_CFIFO)	8	R/W	00h	46.4.16/ 1284
4006_B012	UART FIFO Status Register (UART1_SFIFO)	8	R/W	C0h	46.4.17/ 1285
4006_B013	UART FIFO Transmit Watermark (UART1_TWFIFO)	8	R/W	00h	46.4.18/ 1286
4006_B014	UART FIFO Transmit Count (UART1_TCFIFO)	8	R	00h	46.4.19/ 1287
4006_B015	UART FIFO Receive Watermark (UART1_RWFIFO)	8	R/W	01h	46.4.20/ 1287
4006_B016	UART FIFO Receive Count (UART1_RCFIFO)	8	R	00h	46.4.21/ 1288

46.4.1 UART Baud Rate Registers: High (UARTx_BDH)

This register, along with the BDL register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting (SBR[12:0]), first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written.

BDL is reset to a nonzero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] is set.

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read Write	LBKDIE	RXEDGIE	SBNS	SBR				
Reset	0	0	0	0	0	0	0	0

UARTx_BDH field descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt or DMA Request Enable Enables the LIN break detect flag, LBKDIF, to generate interrupt requests based on the state of LBKDDMAS. or DMA transfer requests, 0 LBKDIF interrupt and DMA transfer requests disabled. 1 LBKDIF interrupt or DMA transfer requests enabled.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable Enables the receive input active edge, RXEDGIF, to generate interrupt requests. 0 Hardware interrupts from RXEDGIF disabled using polling. 1 RXEDGIF interrupt request enabled.
5 SBNS	Stop Bit Number Select SBNS selects the number of stop bits present in a data frame. This field valid for all 8, 9 and 10 bit data formats available. 0 Data frame consists of a single stop bit. 1 Data frame consists of two stop bits.
SBR	UART Baud Rate Bits The baud rate for the UART is determined by the 13 SBR fields. See Baud rate generation for details. **NOTE:** • The baud rate generator is disabled until C2[TE] or C2[RE] is set for the first time after reset.The baud rate generator is disabled when SBR = 0. • Writing to BDH has no effect without writing to BDL, because writing to BDH puts the data in a temporary location until BDL is written.

46.4.2 UART Baud Rate Registers: Low (UARTx_BDL)

This register, along with the BDH register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting, SBR[12:0], first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written. BDL is reset to a nonzero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] is set.

Address: Base address + 1h offset

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

UARTx_BDL field descriptions

Field	Description
SBR	UART Baud Rate Bits

UARTx_BDL field descriptions (continued)

Field	Description
	The baud rate for the UART is determined by the 13 SBR fields. See Baud rate generation for details. **NOTE:** • The baud rate generator is disabled until C2[TE] or C2[RE] is set for the first time after reset.The baud rate generator is disabled when SBR = 0. • Writing to BDH has no effect without writing to BDL, because writing to BDH puts the data in a temporary location until BDL is written.

46.4.3 UART Control Register 1 (UARTx_C1)

This read/write register controls various optional features of the UART system.

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	LOOPS	UARTSWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

UARTx_C1 field descriptions

Field	Description
7 LOOPS	Loop Mode Select When LOOPS is set, the RxD pin is disconnected from the UART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0 Normal operation. 1 Loop mode where transmitter output is internally connected to receiver input. The receiver input is determined by RSRC.
6 UARTSWAI	UART Stops in Wait Mode 0 UART clock continues to run in Wait mode. 1 UART clock freezes while CPU is in Wait mode.
5 RSRC	Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0 Selects internal loop back mode. The receiver input is internally connected to transmitter output. 1 Single wire UART mode where the receiver input is connected to the transmit pin input signal.
4 M	9-bit or 8-bit Mode Select 0 Normal—start + 8 data bits (MSB/LSB first as determined by MSBF) + stop. 1 Use—start + 9 data bits (MSB/LSB first as determined by MSBF) + stop.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the UART: • Address mark in the most significant bit position of a received data character, or • An idle condition on the receive pin input signal.

Table continues on the next page...

UARTx_C1 field descriptions (continued)

Field	Description
	0 Idle line wakeup. 1 Address mark wakeup.
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE:** • In case the UART is programmed with ILT = 1, a logic of 1'b0 is automatically shifted after a received stop bit, therefore resetting the idle count. • In case the UART is programmed for IDLE line wakeup (RWU = 1 and WAKE = 0), ILT has no effect on when the receiver starts counting logic 1s as idle character bits. In idle line wakeup, an idle character is recognized at anytime the receiver sees 10, 11, or 12 1s depending on the M, PE, and C4[M10] fields. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables the parity function. When parity is enabled, parity function inserts a parity bit in the bit position immediately preceding the stop bit. 0 Parity function disabled. 1 Parity function enabled.
0 PT	Parity Type Determines whether the UART generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. 0 Even parity. 1 Odd parity.

46.4.4 UART Control Register 2 (UARTx_C2)

This register can be read or written at any time.

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read Write	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

UARTx_C2 field descriptions

Field	Description
7 TIE	Transmitter Interrupt or DMA Transfer Enable. Enables S1[TDRE] to generate interrupt requests or DMA transfer requests, based on the state of C5[TDMAS].

Table continues on the next page...

UARTx_C2 field descriptions (continued)

Field	Description
	NOTE: If C2[TIE] and C5[TDMAS] are both set, then TCIE must be cleared, and D[D] must not be written unless servicing a DMA request. 0 TDRE interrupt and DMA transfer requests disabled. 1 TDRE interrupt or DMA transfer requests enabled.
6 TCIE	Transmission Complete Interrupt Enable Enables the transmission complete flag, S1[TC], to generate interrupt requests . 0 TC interrupt requests disabled. 1 TC interrupt requests enabled.
5 RIE	Receiver Full Interrupt or DMA Transfer Enable Enables S1[RDRF] to generate interrupt requests or DMA transfer requests, based on the state of C5[RDMAS]. 0 RDRF interrupt and DMA transfer requests disabled. 1 RDRF interrupt or DMA transfer requests enabled.
4 ILIE	Idle Line Interrupt Enable Enables the idle line flag, S1[IDLE], to generate interrupt requests 0 IDLE interrupt requests disabled. 1 IDLE interrupt requests enabled.
3 TE	Transmitter Enable Enables the UART transmitter. TE can be used to queue an idle preamble by clearing and then setting TE. 0 Transmitter off. 1 Transmitter on.
2 RE	Receiver Enable Enables the UART receiver. 0 Receiver off. 1 Receiver on.
1 RWU	Receiver Wakeup Control This field can be set to place the UART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when C1[WAKE] is clear or an address match when C1[WAKE] is set. **NOTE:** RWU must be set only with C1[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by S2[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the UART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to reasserted. 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	Send Break

Table continues on the next page...

UARTx_C2 field descriptions (continued)

Field	Description
	Toggling SBK sends one break character from the following: See Transmitting break characters for the number of logic 0s for the different configurations. Toggling implies clearing the SBK field before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10, 11, or 12 bits, or 13 or 14 bits). Ensure that C2[TE] is asserted atleast 1 clock before assertion of this bit. • 10, 11, or 12 logic 0s if S2[BRK13] is cleared • 13 or 14 logic 0s if S2[BRK13] is set. 0 Normal transmitter operation. 1 Queue break characters to be sent.

46.4.5 UART Status Register 1 (UARTx_S1)

The S1 register provides inputs to the MCU for generation of UART interrupts or DMA requests. This register can also be polled by the MCU to check the status of its fields. To clear a flag, the status register should be read followed by a read or write to D register, depending on the interrupt flag type. Other instructions can be executed between the two steps as long the handling of I/O is not compromised, but the order of operations is important for flag clearing. When a flag is configured to trigger a DMA request, assertion of the associated DMA done signal from the DMA controller clears the flag.

NOTE

- If the condition that results in the assertion of the flag, interrupt, or DMA request is not resolved prior to clearing the flag, the flag, and interrupt/DMA request, reasserts. For example, if the DMA or interrupt service routine fails to write sufficient data to the transmit buffer to raise it above the watermark level, the flag reasserts and generates another interrupt or DMA request.
- Reading an empty data register to clear one of the flags of the S1 register causes the FIFO pointers to become misaligned. A receive FIFO flush reinitializes the pointers. A better way to prevent this situation is to always leave one byte in FIFO and this byte will be read eventually in clearing the flag bit.

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write								
Reset	1	1	0	0	0	0	0	0

UARTx_S1 field descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag TDRE will set when the number of datawords in the transmit buffer (D and C3[T8])is equal to or less than the number indicated by TWFIFO[TXWATER]. A character that is in the process of being transmitted is not included in the count. To clear TDRE, read S1 when TDRE is set and then write to the UART data register (D). For more efficient interrupt servicing, all data except the final value to be written to the buffer must be written to D/C3[T8]. Then S1 can be read before writing the final data value, resulting in the clearing of the TRDE flag. This is more efficient because the TDRE reasserts until the watermark has been exceeded. So, attempting to clear the TDRE with every write will be ineffective until sufficient data has been written. 0 The amount of data in the transmit buffer is greater than the value indicated by TWFIFO[TXWATER]. 1 The amount of data in the transmit buffer is less than or equal to the value indicated by TWFIFO[TXWATER] at some point in time since the flag has been cleared.
6 TC	Transmit Complete Flag TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by reading S1 with TC set and then doing one of the following: • Writing to D to transmit new data. • Queuing a preamble by clearing and then setting C2[TE]. • Queuing a break character by writing 1 to SBK in C2. 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
5 RDRF	Receive Data Register Full Flag RDRF is set when the number of datawords in the receive buffer is equal to or more than the number indicated by RWFIFO[RXWATER]. A dataword that is in the process of being received is not included in the count. To clear RDRF, read S1 when RDRF is set and then read D. For more efficient interrupt and DMA operation, read all data except the final value from the buffer, using D/C3[T8]/ED. Then read S1 and the final data value, resulting in the clearing of the RDRF flag. Even if RDRF is set, data will continue to be received until an overrun condition occurs.RDRF is prevented from setting while S2[LBKDE] is set. Additionally, when S2[LBKDE] is set, the received datawords are stored in the receive buffer but over-write each other. 0 The number of datawords in the receive buffer is less than the number indicated by RXWATER. 1 The number of datawords in the receive buffer is equal to or greater than the number indicated by RXWATER at some point in time since this flag was last cleared.
4 IDLE	Idle Line Flag After the IDLE flag is cleared, a frame must be received (although not necessarily stored in the data buffer, for example if C2[RWU] is set), or a LIN break character must set the S2[LBKDIF] flag before an idle condition can set the IDLE flag. To clear IDLE, read UART status S1 with IDLE set and then read D. IDLE is set when either of the following appear on the receiver input: • 10 consecutive logic 1s if C1[M] = 0 • 11 consecutive logic 1s if C1[M] = 1 and C4[M10] = 0 • 12 consecutive logic 1s if C1[M] = 1, C4[M10] = 1, and C1[PE] = 1 **NOTE:** When RWU is set and WAKE is cleared, an idle line condition sets the IDLE flag if RWUID is set, else the IDLE flag does not become set. 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared. 1 Receiver input has become idle or the flag has not been cleared since it last asserted.
3 OR	Receiver Overrun Flag

Table continues on the next page...

UARTx_S1 field descriptions (continued)

Field	Description
	OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the UART data registers is not affected. If the OR flag is set, no data is stored in the data buffer even if sufficient room exists. Additionally, while the OR flag is set, the RDRF and IDLE flags are blocked from asserting, that is, transition from an inactive to an active state. To clear OR, read S1 when OR is set and then read D. See functional description for more details regarding the operation of the OR bit.If LBKDE is enabled and a LIN Break is detected, the OR field asserts if S2[LBKDIF] is not cleared before the next data character is received. 0 No overrun has occurred since the last time the flag was cleared. 1 Overrun has occurred or the overrun flag has not been cleared since the last overrun occured.
2 NF	Noise Flag NF is set when the UART detects noise on the receiver input. NF does not become set in the case of an overrun or while the LIN break detect feature is enabled (S2[LBKDE] = 1). When NF is set, it indicates only that a dataword has been received with noise since the last time it was cleared. There is no guarantee that the first dataword read from the receive buffer has noise or that there is only one dataword in the buffer that was received with noise unless the receive buffer has a depth of one. To clear NF, read S1 and then read D. 0 No noise detected since the last time this flag was cleared. If the receive buffer has a depth greater than 1 then there may be data in the receiver buffer that was received with noise. 1 At least one dataword was received with noise detected since the last time the flag was cleared.
1 FE	Framing Error Flag FE is set when a logic 0 is accepted as the stop bit. When BDH[SBNS] is set, then FE will set when a logic 0 is accepted for either of the two stop bits. FE does not set in the case of an overrun or while the LIN break detect feature is enabled (S2[LBKDE] = 1). FE inhibits further data reception until it is cleared. To clear FE, read S1 with FE set and then read D. The last data in the receive buffer represents the data that was received with the frame error enabled. 0 No framing error detected. 1 Framing error.
0 PF	Parity Error Flag PF is set when PE is set and the parity of the received data does not match its parity bit. The PF is not set in the case of an overrun condition. When PF is set, it indicates only that a dataword was received with parity error since the last time it was cleared. There is no guarantee that the first dataword read from the receive buffer has a parity error or that there is only one dataword in the buffer that was received with a parity error, unless the receive buffer has a depth of one. To clear PF, read S1 and then read D., S2[LBKDE] is disabled,Within the receive buffer structure the received dataword is tagged if it is received with a parity error. This information is available by reading the ED register prior to reading the D register. 0 No parity error detected since the last time this flag was cleared. If the receive buffer has a depth greater than 1, then there may be data in the receive buffer what was received with a parity error. 1 At least one dataword was received with a parity error since the last time this flag was cleared.

46.4.6 UART Status Register 2 (UARTx_S2)

The S2 register provides inputs to the MCU for generation of UART interrupts or DMA requests. Also, this register can be polled by the MCU to check the status of these bits. This register can be read or written at any time, with the exception of the MSBF and RXINV bits, which should be changed by the user only between transmit and receive packets.

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF
Write	w1c	w1c						
Reset	0	0	0	0	0	0	0	0

UARTx_S2 field descriptions

Field	Description
7 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when LBKDE is set and a LIN break character is detected on the receiver input. The LIN break characters are 11 consecutive logic 0s if C1[M] = 0 or 12 consecutive logic 0s if C1[M] = 1. LBKDIF is set after receiving the last LIN break character. LBKDIF is cleared by writing a 1 to it. 0 No LIN break character detected. 1 LIN break character detected.
6 RXEDGIF	RxD Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge occurs on the RxD pin. The active edge is falling if RXINV = 0, and rising if RXINV=1. RXEDGIF is cleared by writing a 1 to it. See for additional details. RXEDGIF description **NOTE:** The active edge is detected only in two wire mode and on receiving data coming from the RxD pin. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
5 MSBF	Most Significant Bit First Setting this field reverses the order of the bits that are transmitted and received on the wire. This field does not affect the polarity of the bits, the location of the parity bit, or the location of the start or stop bits. 0 LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1 MSB (bit8, bit7 or bit6) is the first bit that is transmitted following the start bit, depending on the setting of C1[M] and C1[PE]. Further, the first bit received after the start bit is identified as bit8, bit7, or bit6, depending on the setting of C1[M] and C1[PE].
4 RXINV	Receive Data Inversion Setting this field reverses the polarity of the received data input. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. **NOTE:** Setting RXINV inverts the RxD input for data bits, start and stop bits, break, and idle.

Table continues on the next page...

UARTx_S2 field descriptions (continued)

Field	Description
	0 Receive data is not inverted. 1 Receive data is inverted.
3 RWUID	Receive Wakeup Idle Detect When RWU is set and WAKE is cleared, this field controls whether the idle character that wakes the receiver sets S1[IDLE]. 0 S1[IDLE] is not set upon detection of an idle character. 1 S1[IDLE] is set upon detection of an idle character.
2 BRK13	Break Transmit Character Length Determines whether the transmit break character is 10, 11, or 12 bits long, or 13 or 14 bits long. See for the length of the break character for the different configurations. The detection of a framing error is not affected by this field. Transmitting break characters 0 Break character is 10, 11, or 12 bits long. 1 Break character is 13 or 14 bits long.
1 LBKDE	LIN Break Detection Enable Enables the LIN Break detection feature. While LBKDE is set, S1[RDRF], S1[NF], S1[FE], and S1[PF] are prevented from setting. When LBKDE is set, see . Overrun operation 0 Break character detection is disabled. 1 Break character is detected at length of 11 bit times if C1[M] = 0 or 12 bits time if C1[M] = 1.
0 RAF	Receiver Active Flag RAF is set when the UART receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 UART receiver idle/inactive waiting for a start bit. 1 UART receiver active, RxD input not idle.

46.4.7 UART Control Register 3 (UARTx_C3)

Writing R8 does not have any effect. TXDIR and TXINV can be changed only between transmit and receive packets.

Address: Base address + 6h offset

Bit	7	6	5	4	3	2	1	0
Read	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C3 field descriptions

Field	Description
7 R8	Received Bit 8

Table continues on the next page...

UARTx_C3 field descriptions (continued)

Field	Description
	R8 is the ninth data bit received when the UART is configured for 9-bit data format, that is, if C1[M] = 1 or C4[M10] = 1. The R8 value corresponds to the current data value in the UARTx_D register. To read the 9th bit, read the value of UARTx_C3[R8], then read the UARTx_D register.
6 T8	Transmit Bit 8 T8 is the ninth data bit transmitted when the UART is configured for 9-bit data format, that is, if C1[M] = 1 or C4[M10] = 1. **NOTE:** If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten. The same value is transmitted until T8 is rewritten. To correctly transmit the 9th bit, write UARTx_C3[T8] to the desired value, then write the UARTx_D register with the remaining data.
5 TXDIR	Transmitter Pin Data Direction in Single-Wire mode Determines whether the TXD pin is used as an input or output in the single-wire mode of operation. This field is relevant only to the single wire mode. 0 TXD pin is an input in single wire mode. 1 TXD pin is an output in single wire mode.
4 TXINV	Transmit Data Inversion. Setting this field reverses the polarity of the transmitted data output. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. **NOTE:** Setting TXINV inverts all transmitted values, including idle, break, start, and stop bits. In loop mode, if TXINV is set, the receiver gets the transmit inversion bit when RXINV is disabled. 0 Transmit data is not inverted. 1 Transmit data is inverted.
3 ORIE	Overrun Error Interrupt Enable Enables the overrun error flag, S1[OR], to generate interrupt requests. 0 OR interrupts are disabled. 1 OR interrupt requests are enabled.
2 NEIE	Noise Error Interrupt Enable Enables the noise flag, S1[NF], to generate interrupt requests. 0 NF interrupt requests are disabled. 1 NF interrupt requests are enabled.
1 FEIE	Framing Error Interrupt Enable Enables the framing error flag, S1[FE], to generate interrupt requests. 0 FE interrupt requests are disabled. 1 FE interrupt requests are enabled.
0 PEIE	Parity Error Interrupt Enable Enables the parity error flag, S1[PF], to generate interrupt requests. 0 PF interrupt requests are disabled. 1 PF interrupt requests are enabled.

46.4.8 UART Data Register (UARTx_D)

This register is actually two separate registers. Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

NOTE

- In 8-bit or 9-bit data format, only UART data register (D) needs to be accessed to clear the S1[RDRF] bit (assuming receiver buffer level is less than RWFIFO[RXWATER]). The C3 register needs to be read, prior to the D register, only if the ninth bit of data needs to be captured. Similarly, the ED register needs to be read, prior to the D register, only if the additional flag data for the dataword needs to be captured.
- In the normal 8-bit mode (M bit cleared) if the parity is enabled, you get seven data bits and one parity bit. That one parity bit is loaded into the D register. So, for the data bits, mask off the parity bit from the value you read out of this register.
- When transmitting in 9-bit data format and using 8-bit write instructions, write first to transmit bit 8 in UART control register 3 (C3[T8]), then D. A write to C3[T8] stores the data in a temporary register. If D register is written first, and then the new data on data bus is stored in D, the temporary value written by the last write to C3[T8] gets stored in the C3[T8] register.

Address: Base address + 7h offset

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

UARTx_D field descriptions

Field	Description
RT	Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

46.4.9 UART Match Address Registers 1 (UARTx_MA1)

The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] field is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. These registers can be read and written at anytime.

Address: Base address + 8h offset

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

UARTx_MA1 field descriptions

Field	Description
MA	Match Address

46.4.10 UART Match Address Registers 2 (UARTx_MA2)

These registers can be read and written at anytime. The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] field is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded.

Address: Base address + 9h offset

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

UARTx_MA2 field descriptions

Field	Description
MA	Match Address

46.4.11 UART Control Register 4 (UARTx_C4)

Address: Base address + Ah offset

Bit	7	6	5	4	3	2	1	0
Read Write	MAEN1	MAEN2	M10	BRFA				
Reset	0	0	0	0	0	0	0	0

UARTx_C4 field descriptions

Field	Description
7 MAEN1	Match Address Mode Enable 1 See Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN2 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA1 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer.
6 MAEN2	Match Address Mode Enable 2 See Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN1 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA2 register. If no match occurs, the data is discarded. If a match occurs, data is transferred to the data buffer.
5 M10	10-bit Mode select Causes a tenth, non-memory mapped bit to be part of the serial transmission. This tenth bit is generated and interpreted as a parity bit. The M10 field does not affect the LIN send or detect break behavior. If M10 is set, then both C1[M] and C1[PE] must also be set. See Data format for more information. 0 The parity bit is the ninth bit in the serial transmission. 1 The parity bit is the tenth bit in the serial transmission.
BRFA	Baud Rate Fine Adjust This bit field is used to add more timing resolution to the average baud frequency, in increments of 1/32. See Baud rate generation for more information.

46.4.12 UART Control Register 5 (UARTx_C5)

Address: Base address + Bh offset

Bit	7	6	5	4	3	2	1	0
Read	TDMAS	0	RDMAS	0	LBKDDMAS	0		
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C5 field descriptions

Field	Description
7 TDMAS	Transmitter DMA Select Configures the transmit data register empty flag, S1[TDRE], to generate interrupt or DMA requests if C2[TIE] is set. **NOTE:** • If C2[TIE] is cleared, TDRE DMA and TDRE interrupt request signals are not asserted when the TDRE flag is set, regardless of the state of TDMAS. • If C2[TIE] and TDMAS are both set, then C2[TCIE] must be cleared, and D must not be written unless a DMA request is being serviced.

Table continues on the next page...

UARTx_C5 field descriptions (continued)

Field	Description
	0 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE interrupt request signal is asserted to request interrupt service. 1 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE DMA request signal is asserted to request a DMA transfer.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 RDMAS	Receiver Full DMA Select Configures the receiver data register full flag, S1[RDRF], to generate interrupt or DMA requests if C2[RIE] is set. **NOTE:** If C2[RIE] is cleared, and S1[RDRF] is set, the RDRF DMA and RDFR interrupt request signals are not asserted, regardless of the state of RDMAS. 0 If C2[RIE] and S1[RDRF] are set, the RDFR interrupt request signal is asserted to request an interrupt service. 1 If C2[RIE] and S1[RDRF] are set, the RDRF DMA request signal is asserted to request a DMA transfer.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 LBKDDMAS	LIN Break Detect DMA Select Bit Configures the LIN break detect flag, S2[LBKDIF], to generate interrupt or DMA requests if BDH[LBKDIE] is set. **NOTE:** If BDH[LBKDIE] is cleared, and S2[LBKDIF] is set, the LBKDIF DMA and LBKDIF interrupt signals are not asserted, regardless of the state of LBKDDMAS. 0 If BDH[LBKDIE] and S2[LBKDIF] are set, the LBKDIF interrupt signal is asserted to request an interrupt service. 1 If BDH[LBKDIE] and S2[LBKDIF] are set, the LBKDIF DMA request signal is asserted to request a DMA transfer.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

46.4.13 UART Extended Data Register (UARTx_ED)

This register contains additional information flags that are stored with a received dataword. This register may be read at any time but contains valid data only if there is a dataword in the receive FIFO.

NOTE

- The data contained in this register represents additional information regarding the conditions on which a dataword was received. The importance of this data varies with the application, and in some cases maybe completely optional.

These fields automatically update to reflect the conditions of the next dataword whenever D is read.

- If S1[NF] and S1[PF] have not been set since the last time the receive buffer was empty, the NOISY and PARITYE fields will be zero.

Address: Base address + Ch offset

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

UARTx_ED field descriptions

Field	Description
7 NOISY	The current received dataword contained in D and C3[R8] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise.
6 PARITYE	The current received dataword contained in D and C3[R8] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

46.4.14 UART Modem Register (UARTx_MODEM)

The MODEM register controls options for setting the modem configuration.

Address: Base address + Dh offset

Bit	7	6	5	4	3	2	1	0
Read	0				RXRTSE	TXRTSPOL	TXRTSE	TXCTSE
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_MODEM field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. **NOTE:** Do not set both RXRTSE and TXRTSE.

Table continues on the next page...

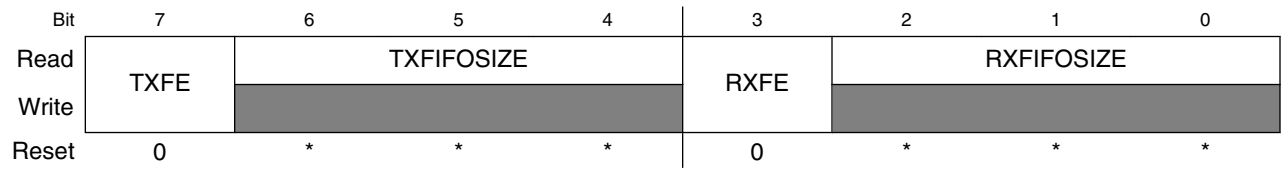
UARTx_MODEM field descriptions (continued)

Field	Description
	0 The receiver has no effect on RTS. 1 RTS is deasserted if the number of characters in the receiver data register (FIFO) is equal to or greater than RWFIFO[RXWATER]. RTS is asserted when the number of characters in the receiver data register (FIFO) is less than RWFIFO[RXWATER]. See Hardware flow control
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set. 0 Transmitter RTS is active low. 1 Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. 0 The transmitter has no effect on RTS. 1 When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. (FIFO) **NOTE:** Ensure that C2[TE] is asserted before assertion of this bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0 CTS has no effect on the transmitter. 1 Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

46.4.15 UART FIFO Parameters (UARTx_PFIFO)

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when C2[RE] and C2[TE] are cleared/not set and when the data buffer/FIFO is empty.

Address: Base address + 10h offset



* Notes:
- TXFIFOSIZE field: The reset value depends on whether the specific UART instance supports the FIFO and on the size of that FIFO. See the Chip Configuration details for more information on the FIFO size supported for each UART instance.
- RXFIFOSIZE field: The reset value depends on whether the specific UART instance supports the FIFO and on the size of that FIFO. See the Chip Configuration details for more information on the FIFO size supported for each UART instance.

UARTx_PFIFO field descriptions

Field	Description
7 TXFE	Transmit FIFO Enable When this field is set, the built in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both C2[TE] and C2[RE] must be cleared prior to changing this field. Additionally, TXFLUSH and RXFLUSH commands must be issued immediately after changing this field. 0 Transmit FIFO is not enabled. Buffer is depth 1. (Legacy support). 1 Transmit FIFO is enabled. Buffer is depth indicated by TXFIFOSIZE.
6–4 TXFIFOSIZE	Transmit FIFO. Buffer Depth The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read only. 000 Transmit FIFO/Buffer depth = 1 dataword. 001 Transmit FIFO/Buffer depth = 4 datawords. 010 Transmit FIFO/Buffer depth = 8 datawords. 011 Transmit FIFO/Buffer depth = 16 datawords. 100 Transmit FIFO/Buffer depth = 32 datawords. 101 Transmit FIFO/Buffer depth = 64 datawords. 110 Transmit FIFO/Buffer depth = 128 datawords. 111 Reserved.
3 RXFE	Receive FIFO Enable

Table continues on the next page...

UARTx_PFIFO field descriptions (continued)

Field	Description
	When this field is set, the built in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both C2[TE] and C2[RE] must be cleared prior to changing this field. Additionally, TXFLUSH and RXFLUSH commands must be issued immediately after changing this field. 0 Receive FIFO is not enabled. Buffer is depth 1. (Legacy support) 1 Receive FIFO is enabled. Buffer is depth indicted by RXFIFOSIZE.
RXFIFOSIZE	Receive FIFO. Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only. 000 Receive FIFO/Buffer depth = 1 dataword. 001 Receive FIFO/Buffer depth = 4 datawords. 010 Receive FIFO/Buffer depth = 8 datawords. 011 Receive FIFO/Buffer depth = 16 datawords. 100 Receive FIFO/Buffer depth = 32 datawords. 101 Receive FIFO/Buffer depth = 64 datawords. 110 Receive FIFO/Buffer depth = 128 datawords. 111 Reserved.

46.4.16 UART FIFO Control Register (UARTx_CFIFO)

This register provides the ability to program various control fields for FIFO operation. This register may be read or written at any time. Note that writing to TXFLUSH and RXFLUSH may result in data loss and requires careful action to prevent unintended/ unpredictable behavior. Therefore, it is recommended that TE and RE be cleared prior to flushing the corresponding FIFO.

Address: Base address + 11h offset

Bit	7	6	5	4	3	2	1	0
Read	0	0	0			RXOFE	TXOFE	RXUFE
Write	TXFLUSH	RXFLUSH						
Reset	0	0	0	0	0	0	0	0

UARTx_CFIFO field descriptions

Field	Description
7 TXFLUSH	Transmit FIFO/Buffer Flush Writing to this field causes all data that is stored in the transmit FIFO/buffer to be flushed. This does not affect data that is in the transmit shift register. 0 No flush operation occurs. 1 All data in the transmit FIFO/Buffer is cleared out.

Table continues on the next page...

UARTx_CFIFO field descriptions (continued)

Field	Description
6 RXFLUSH	Receive FIFO/Buffer Flush Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register. 0 No flush operation occurs. 1 All data in the receive FIFO/buffer is cleared out.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RXOFE	Receive FIFO Overflow Interrupt Enable When this field is set, the RXOF flag generates an interrupt to the host. 0 RXOF flag does not generate an interrupt to the host. 1 RXOF flag generates an interrupt to the host.
1 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0 TXOF flag does not generate an interrupt to the host. 1 TXOF flag generates an interrupt to the host.
0 RXUFE	Receive FIFO Underflow Interrupt Enable When this field is set, the RXUF flag generates an interrupt to the host. 0 RXUF flag does not generate an interrupt to the host. 1 RXUF flag generates an interrupt to the host.

46.4.17 UART FIFO Status Register (UARTx_SFIFO)

This register provides status information regarding the transmit and receiver buffers/ FIFOs, including interrupt information. This register may be written to or read at any time.

Address: Base address + 12h offset

Bit	7	6	5	4	3	2	1	0
Read	TXEMPT	RXEMPT	0			RXOF	TXOF	RXUF
Write						w1c	w1c	w1c
Reset	1	1	0	0	0	0	0	0

UARTx_SFIFO field descriptions

Field	Description
7 TXEMPT	Transmit Buffer/FIFO Empty Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register.

Table continues on the next page...

UARTx_SFIFO field descriptions (continued)

Field	Description
	0 Transmit buffer is not empty. 1 Transmit buffer is empty.
6 RXEMPT	Receive Buffer/FIFO Empty Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0 Receive buffer is not empty. 1 Receive buffer is empty.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 RXOF	Receiver Buffer Overflow Flag Indicates that more data has been written to the receive buffer than it can hold. This field will assert regardless of the value of CFIFO[RXOFE]. However, an interrupt will be issued to the host only if CFIFO[RXOFE] is set. This flag is cleared by writing a 1. 0 No receive buffer overflow has occurred since the last time the flag was cleared. 1 At least one receive buffer overflow has occurred since the last time the flag was cleared.
1 TXOF	Transmitter Buffer Overflow Flag Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of CFIFO[TXOFE]. However, an interrupt will be issued to the host only if CFIFO[TXOFE] is set. This flag is cleared by writing a 1. 0 No transmit buffer overflow has occurred since the last time the flag was cleared. 1 At least one transmit buffer overflow has occurred since the last time the flag was cleared.
0 RXUF	Receiver Buffer Underflow Flag Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of CFIFO[RXUFE]. However, an interrupt will be issued to the host only if CFIFO[RXUFE] is set. This flag is cleared by writing a 1. 0 No receive buffer underflow has occurred since the last time the flag was cleared. 1 At least one receive buffer underflow has occurred since the last time the flag was cleared.

46.4.18 UART FIFO Transmit Watermark (UARTx_TWFIFO)

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when C2[TE] is not set. Changing the value of the watermark will not clear the S1[TDRE] flag.

Address: Base address + 13h offset

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

UARTx_TWFIFO field descriptions

Field	Description
TXWATER	Transmit Watermark When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt via S1[TDRE] or a DMA request via C5[TDMAS] is generated as determined by C5[TDMAS] and C2[TIE]. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by PFIFO[TXFIFOSIZE] and PFIFO[TXFE].

46.4.19 UART FIFO Transmit Count (UARTx_TCFIFO)

This is a read only register that indicates how many datawords are currently in the transmit buffer/FIFO. It may be read at any time.

Address: Base address + 14h offset

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

UARTx_TCFIFO field descriptions

Field	Description
TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with PFIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.

46.4.20 UART FIFO Receive Watermark (UARTx_RWFIFO)

This register provides the ability to set a programmable threshold for notification of the need to remove data from the receiver FIFO/buffer. This register may be read at any time but must be written only when C2[RE] is not asserted. Changing the value in this register will not clear S1[RDRF].

Address: Base address + 15h offset

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

UARTx_RWFIFO field descriptions

Field	Description
RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is equal to or greater than the value in this register field, an interrupt via S1[RDRF] or a DMA request via C5[RDMAS] is generated as determined by C5[RDMAS] and C2[RIE]. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by PFIFO[RXFIFOSIZE] and PFIFO[RXFE] and must be greater than 0.

46.4.21 UART FIFO Receive Count (UARTx_RCFIFO)

This is a read only register that indicates how many datawords are currently in the receive FIFO/buffer. It may be read at any time.

Address: Base address + 16h offset

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

UARTx_RCFIFO field descriptions

Field	Description
RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with PFIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.

46.5 Functional description

This section provides a complete functional description of the UART block.

The UART allows full duplex, asynchronous, NRZ serial communication between the CPU and remote devices, including other CPUs. The UART transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the UART, writes the data to be transmitted, and processes received data.

46.5.1 Transmitter

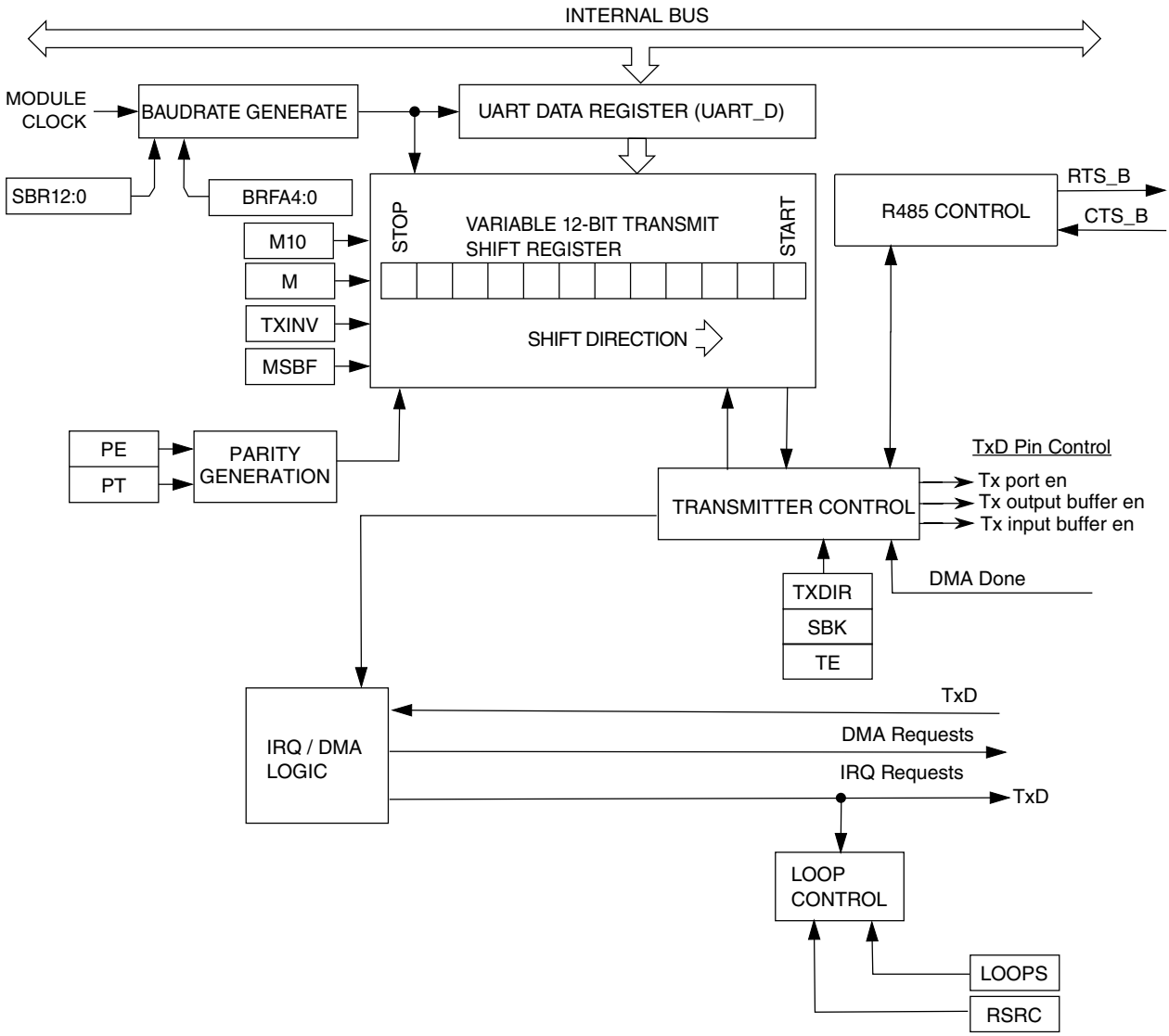


Figure 46-1. Transmitter Block Diagram

46.5.1.1 Transmitter character length

The UART transmitter can accommodate either 8, 9, or 10-bit data characters. The state of the C1[M] and C1[PE] bits and the C4[M10] bit determine the length of data characters. When transmitting 9-bit data, bit C3[T8] is the ninth bit (bit 8).

46.5.1.2 Transmission bit order

When S2[MSBF] is set, the UART automatically transmits the MSB of the data word as the first bit after the start bit. Similarly, the LSB of the data word is transmitted immediately preceding the parity bit, or the stop bit if parity is not enabled. All necessary bit ordering is handled automatically by the module. Therefore, the format of the data written to D for transmission is completely independent of the S2[MSBF] setting.

46.5.1.3 Character transmission

To transmit data, the MCU writes the data bits to the UART transmit buffer using UART data registers C3[T8] and D. Data in the transmit buffer is then transferred to the transmitter shift register as needed. The transmit shift register then shifts a frame out through the transmit data output signal after it has prefaced it with any required start and stop bits. The UART data registers, C3[T8] and D, provide access to the transmit buffer structure.

The UART also sets a flag, the transmit data register empty flag S1[TDRE], and generates an interrupt or DMA request (C5[TDMAS]) whenever the number of datawords in the transmit buffer is equal to or less than the value indicated by TWFIFO[TXWATER]. The transmit driver routine may respond to this flag by writing additional datawords to the transmit buffer using C3[T8]/D as space permits.

See Application information for specific programing sequences.

Setting C2[TE] automatically loads the transmit shift register with the following preamble:

Table 46-3. Transmit preamble length

BDH[SBNS]	C1[M]	C4[M10]	C1[PE]	Bits transmitted
0	0	—	—	10
1	0	—	—	11
0	1	0	—	11
1	1	0	—	12
0	1	1	1	12
1	1	1	1	13

After the preamble shifts out, control logic transfers the data from the D register into the transmit shift register. The transmitter automatically transmits the correct start bit and stop bit before and after the dataword. The number of stop bits transmitted after the dataword can be programmed using BDH[SBNS] field.

Hardware supports odd or even parity. When parity is enabled, the bit immediately preceding the stop bit is the parity bit.

When the transmit shift register is not transmitting a frame, the transmit data output signal goes to the idle condition, logic 1. If at any time software clears C2[TE], the transmitter enable signal goes low and the transmit signal goes idle.

If the software clears C2[TE] while a transmission is in progress, the character in the transmit shift register continues to shift out, provided S1[TC] was cleared during the data write sequence. To clear S1[TC], the S1 register must be read followed by a write to D register.

If S1[TC] is cleared during character transmission and C2[TE] is cleared, the transmission enable signal is deasserted at the completion of the current frame. Following this, the transmit data out signal enters the idle state even if there is data pending in the UART transmit data buffer. To ensure that all the data written in the FIFO is transmitted on the link before clearing C2[TE], wait for S1[TC] to set. Alternatively, the same can be achieved by setting TWFIFO[TXWATER] to 0x0 and waiting for S1[TDRE] to set.

46.5.1.4 Transmitting break characters

Setting C2[SBK] loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on C1[M], C1[PE], S2[BRK13], BDH[SBNS] and C4[M10]. See the following table.

Table 46-4. Transmit break character length

S2[BRK13]	BDH[SBNS]	C1[M]	C4[M10]	C1[PE]	Bits transmitted
0	0	0	—	—	10
0	1	0	—	—	11
0	0	1	0	—	11
0	1	1	0	—	12
0	0	1	1	1	12
0	1	1	1	1	13
1	0	0	—	—	13
1	0	1	—	—	14
1	1	0	—	—	15
1	1	1	—	—	16

As long as C2[SBK] is set, the transmitter logic continuously loads break characters into the transmit shift register. After the software clears C2[SBK], the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character.

NOTE

When queuing a break character, it will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that, if data is queued in the data buffer to be transmitted, the break character preempts that queued data. The queued data is then transmitted after the break character is complete.

46.5.1.5 Idle characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on C1[M], C1[PE], BDH[SBNS] and C4[M10]. The preamble is a synchronizing idle character that begins the first transmission initiated after setting C2[TE].

If C2[TE] is cleared during a transmission, the transmit data output signal becomes idle after completion of the transmission in progress. Clearing and then setting C2[TE] during a transmission queues an idle character to be sent after the dataword currently being transmitted.

Note

When queuing an idle character, the idle character will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that if data is queued in the data buffer to be transmitted, the idle character preempts that queued data. The queued data is then transmitted after the idle character is complete.

If C2[TE] is cleared and the transmission is completed, the UART is not the master of the TXD pin.

46.5.1.6 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS. If the clear-to-send operation is enabled, the character is transmitted when CTS is asserted. If CTS is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and TXD remains in the mark state until CTS is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS. Also, if the transmitter is forced to send a continuous low condition because it is sending a break character, the transmitter ignores the state of CTS regardless of whether the clear-to-send operation is enabled.

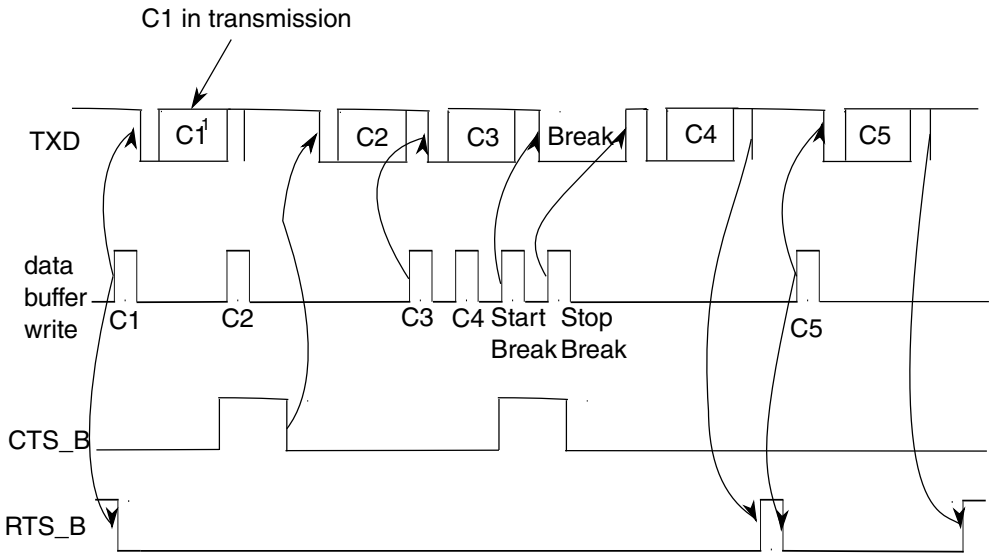
The transmitter's CTS signal can also be enabled even if the same UART receiver's RTS signal is disabled.

46.5.1.7 Transceiver driver enable

The transmitter can use RTS as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, RTS asserts one bit time before the start bit is transmitted. RTS remains asserted for the whole time that the transmitter data buffer has any characters. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts RTS, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's RTS signal asserts only when the transmitter is enabled. However, the transmitter's RTS signal is unaffected by its CTS signal. RTS will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

The following figure shows the functional timing information for the transmitter. Along with the actual character itself, TXD shows the start bit. The stop bit is also indicated, with a dashed line if necessary.



1. Cn = transmit characters

Figure 46-2. Transmitter RTS and CTS timing diagram

46.5.2 Receiver

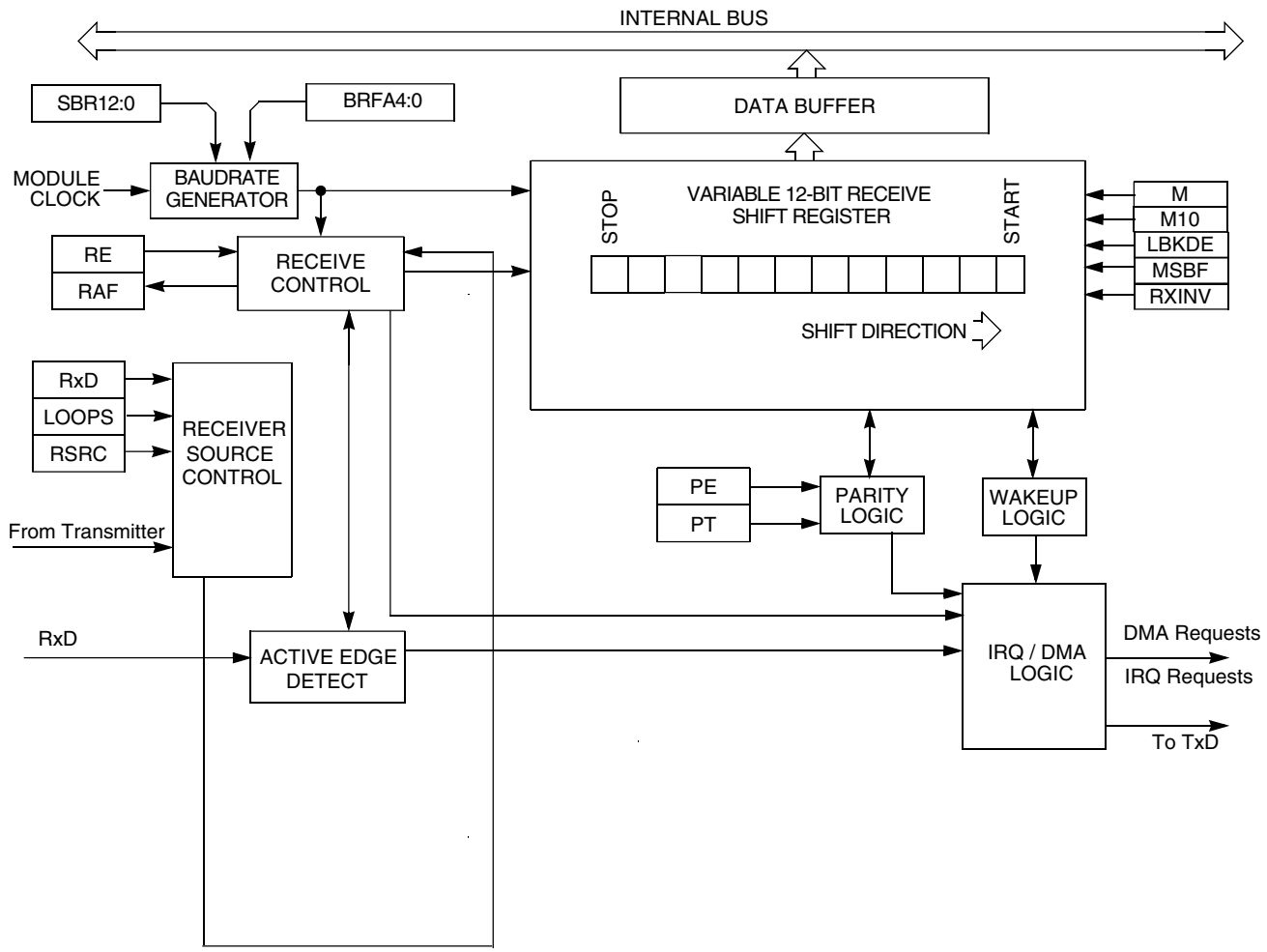


Figure 46-3. UART receiver block diagram

46.5.2.1 Receiver character length

The UART receiver can accommodate 8-, 9-, or 10-bit data characters. The states of C1[M], C1[PE], BDH[SBNS] and C4[M10] determine the length of data characters. When receiving 9 or 10-bit data, C3[R8] is the ninth bit (bit 8).

46.5.2.2 Receiver bit ordering

When S2[MSBF] is set, the receiver operates such that the first bit received after the start bit is the MSB of the dataword. Similarly, the bit received immediately preceding the parity bit, or the stop bit if parity is not enabled, is treated as the LSB for the dataword. All necessary bit ordering is handled automatically by the module. Therefore, the format of the data read from receive data buffer is completely independent of S2[MSBF].

46.5.2.3 Character reception

During UART reception, the receive shift register shifts a frame in from the unsynchronized receiver input signal. After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the UART receive buffer. Additionally, the noise and parity error flags that are calculated during the receive process are also captured in the UART receive buffer. The receive data buffer is accessible via the D and C3[T8] registers. Additional received information flags regarding the receive dataword can be read in ED register. S1[RDRF] is set if the number of resulting datawords in the receive buffer is equal to or greater than the number indicated by RWFIFO[RXWATER]. If the C2[RIE] is also set, RDRF generates an RDRF interrupt request. Alternatively, by programming C5[RDMAS], a DMA request can be generated.

46.5.2.4 Data sampling

The receiver samples the unsynchronized receiver input signal at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see the following figure) is re-synchronized:

- After every start bit.
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0).

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

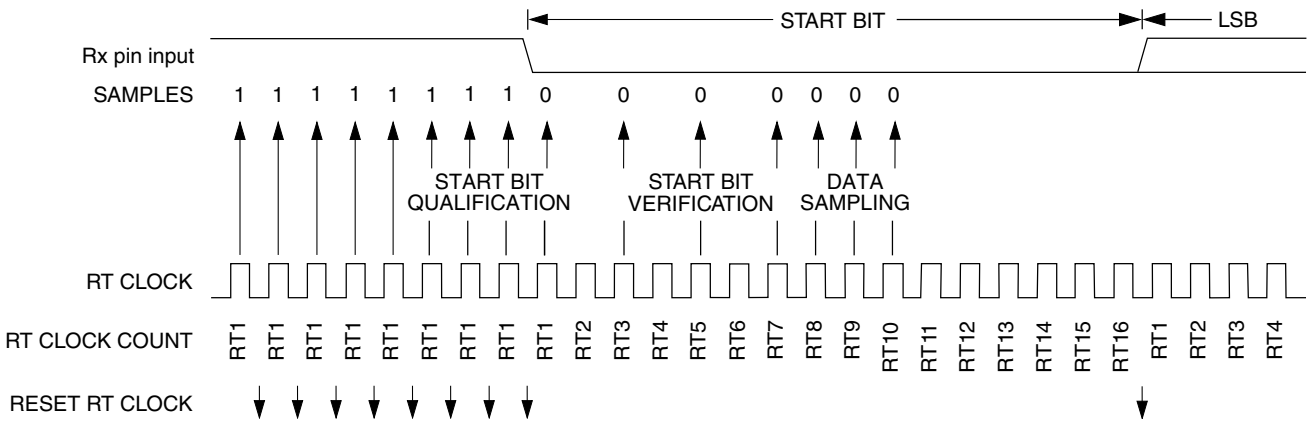


Figure 46-4. Receiver data sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7 . The following table summarizes the results of the start bit verification samples.

Table 46-5. Start bit verification

RT3, RT5, and RT7 samples	Start bit verification	Noise flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. The following table summarizes the results of the data bit samples.

Table 46-6. Data bit recovery

RT8, RT9, and RT10 samples	Data bit determination	Noise flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

Note

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (S1[NF]) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. The following table summarizes the results of the stop bit samples.

Table 46-7. Stop bit recovery

RT8, RT9, and RT10 samples	Framing error flag	Noise flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In the following figure, the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

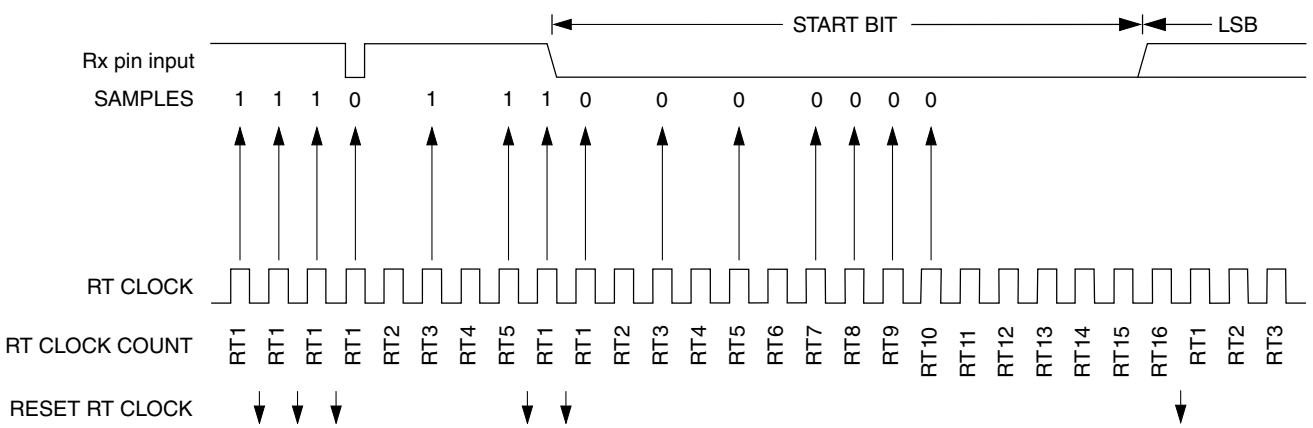


Figure 46-5. Start bit search example 1

In the following figure, verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

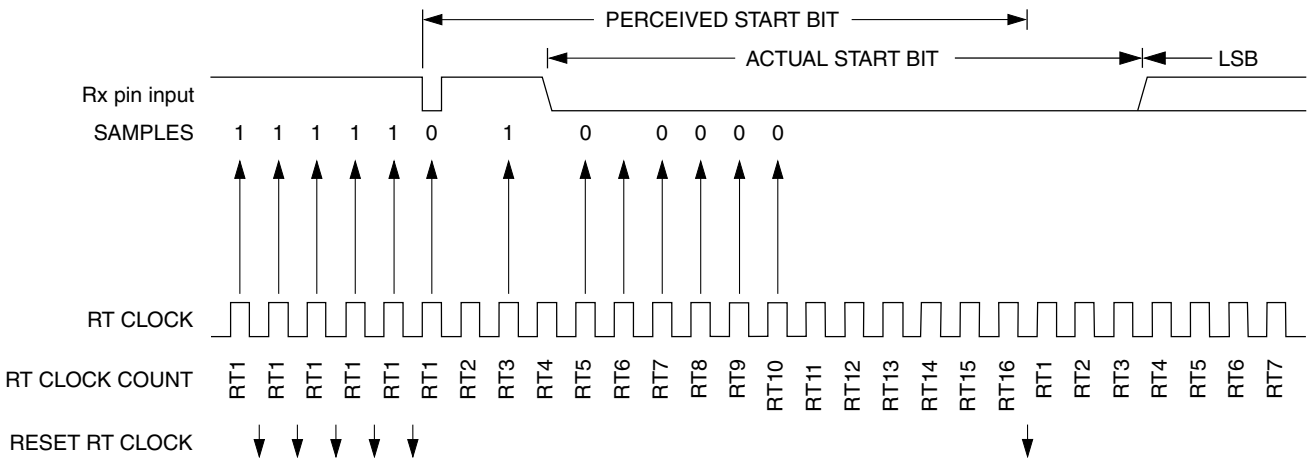


Figure 46-6. Start bit search example 2

In the following figure, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

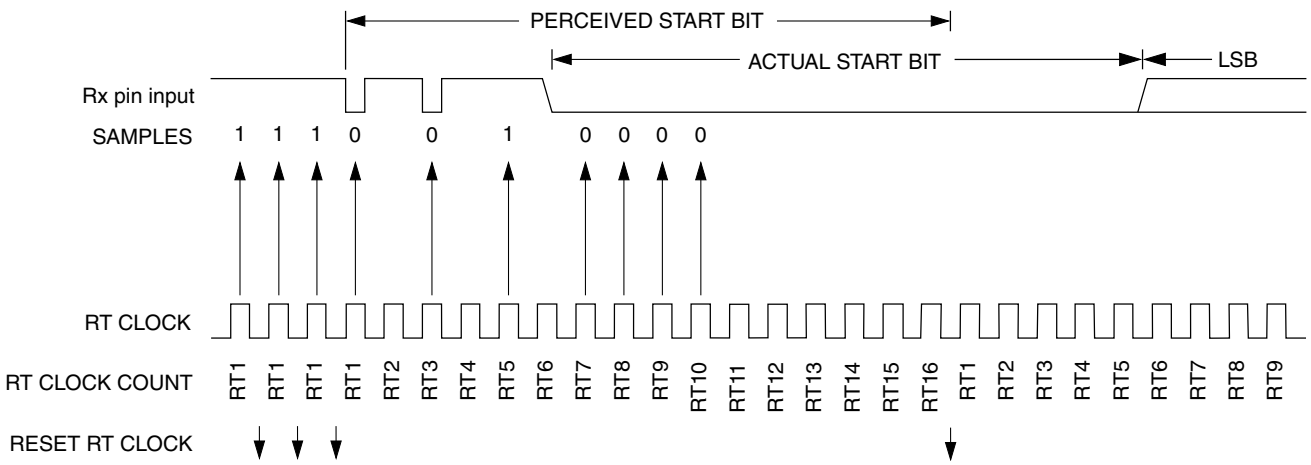


Figure 46-7. Start bit search example 3

The following figure shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

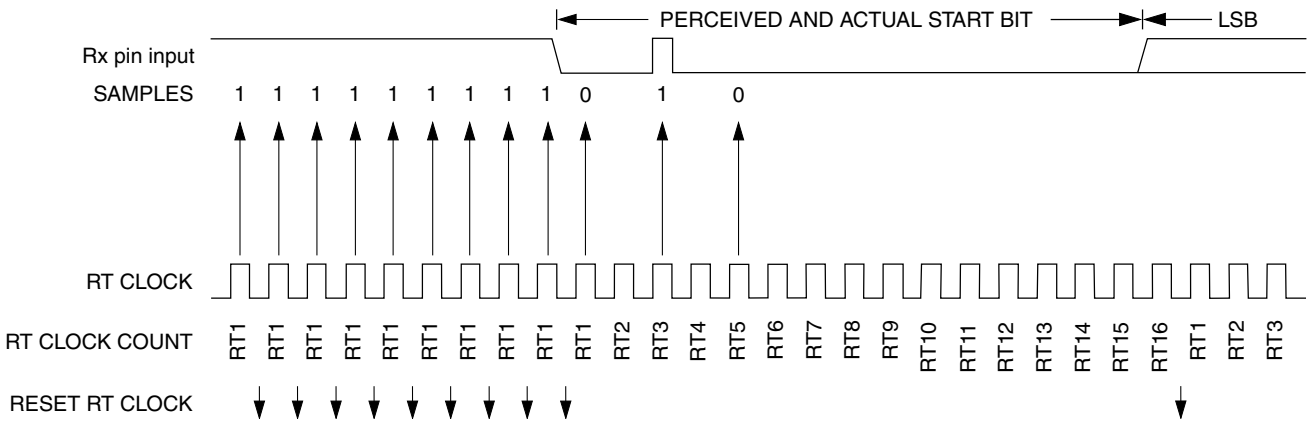


Figure 46-8. Start bit search example 4

The following figure shows a burst of noise near the beginning of the start bit that resets the RT clock. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

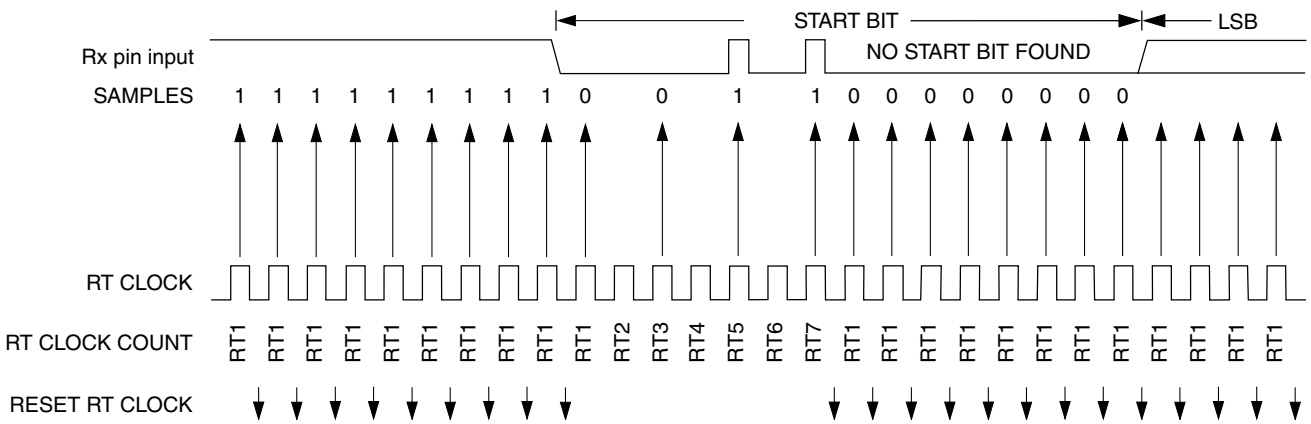


Figure 46-9. Start bit search example 5

In the following figure, a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

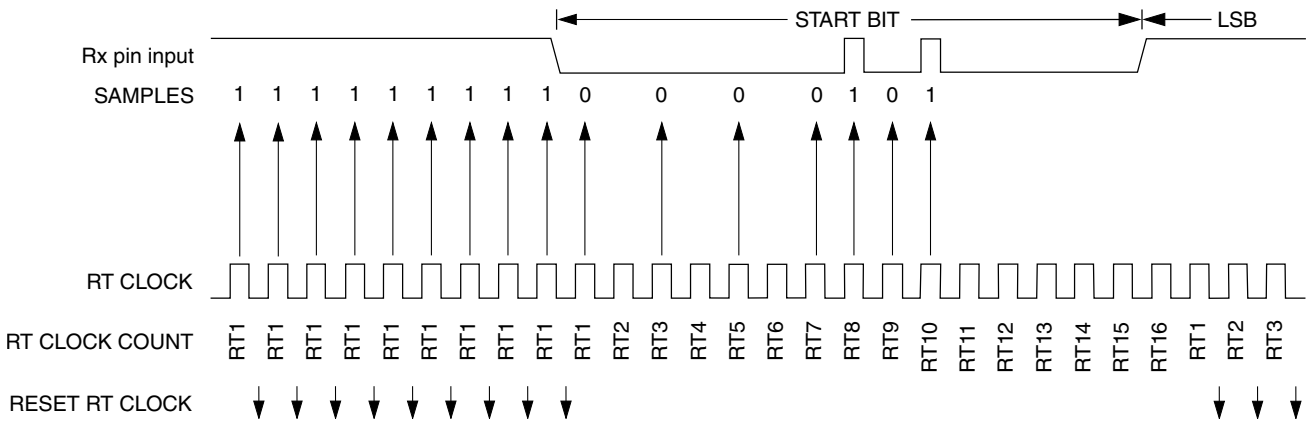


Figure 46-10. Start bit search example 6

46.5.2.5 Framing errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, S1[FE], if S2[LBKDE] is disabled. When S2[LBKDE] is disabled, a break character also sets the S1[FE] because a break character has no stop bit. S1[FE] is set at the same time that received data is placed in the receive data buffer.

46.5.2.6 Receiving break characters

The UART recognizes a break character when a start bit is followed by eight, nine, or ten logic 0 data bits and a logic 0 where the stop bit should be. Receiving a break character has these effects on UART registers:

- Sets the framing error flag, S1[FE].
- Writes an all 0 dataword to the data buffer, which may cause S1[RDRF] to set, depending on the watermark and number of values in the data buffer.
- May set the overrun flag, S1[OR], noise flag, S1[NF], parity error flag, S1[PE], or the receiver active flag, S2[RAF].

The detection threshold for a break character can be adjusted when using an internal oscillator in a LIN system by setting S2[LBKDE]. The UART break character detection threshold depends on C1[M], C1[PE], S2[LBKDE] and C4[M10]. See the following table.

Table 46-8. Receive break character detection threshold

LBKDE	SBNS	M	M10	PE	Threshold (bits)
0	0	0	—	—	10
0	1	0	—	—	11
0	0	1	0	—	11
0	1	1	0	—	12
0	0	1	1	1	12
0	1	1	1	1	13
1	—	0	—	—	11
1	—	1	—	—	12

While S2[LBKDE] is set, it will have these effects on the UART registers:

- Prevents S1[RDRF], S1[FE], S1[NF], and S1[PF] from being set. However, if they are already set, they will remain set.
- Sets the LIN break detect interrupt flag, S2[LBKDIF], if a LIN break character is received.

46.5.2.7 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert RTS.

- RTS remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts RTS if the number of characters in the receiver data register is equal to or greater than receiver data buffer's watermark, RWFIFO[RXWATER].
- The receiver asserts RTS when the number of characters in the receiver data register is less than the watermark. It is not affected if RDRF is asserted.
- Even if RTS is deasserted, the receiver continues to receive characters until the receiver data buffer is full or is overrun.
- If the receiver request-to-send functionality is disabled, the receiver RTS remains deasserted.

The following figure shows receiver hardware flow control functional timing. Along with the actual character itself, RXD shows the start bit. The stop bit can also indicated, with a dashed line, if necessary. The watermark is set to 2.

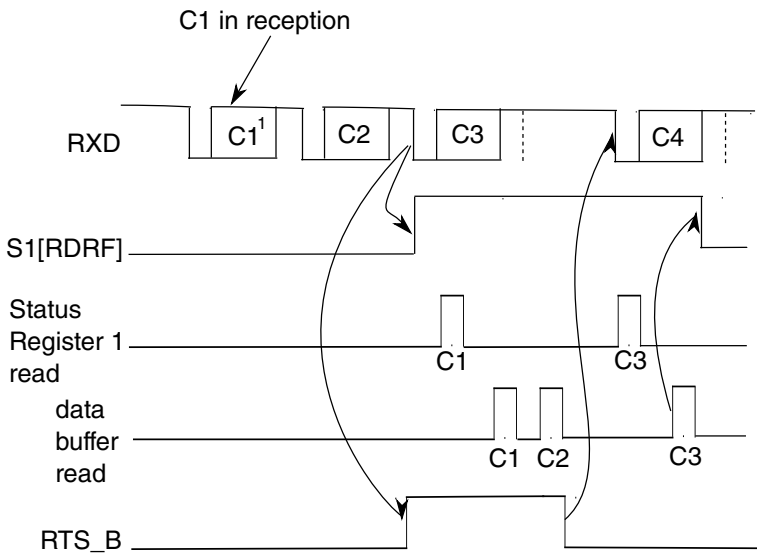


Figure 46-11. Receiver hardware flow control timing diagram

46.5.2.8 Baud rate tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic 0.

As the receiver samples an incoming frame, it resynchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames corrects a misalignment between transmitter bit times and receiver bit times.

46.5.2.8.1 Slow data tolerance

The following figure shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

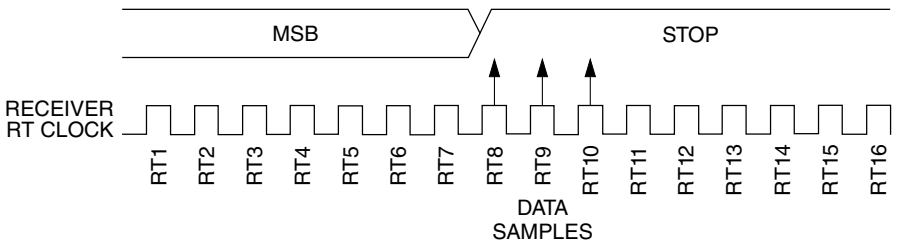


Figure 46-12. Slow data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 46-12, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 147 RT cycles (9 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$((154 - 147) \div 154) \times 100 = 4.54\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 46-12, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 163 RT cycles (10 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$((170 - 163) \div 170) \times 100 = 4.12\%$$

46.5.2.8.2 Fast data tolerance

The following figure shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

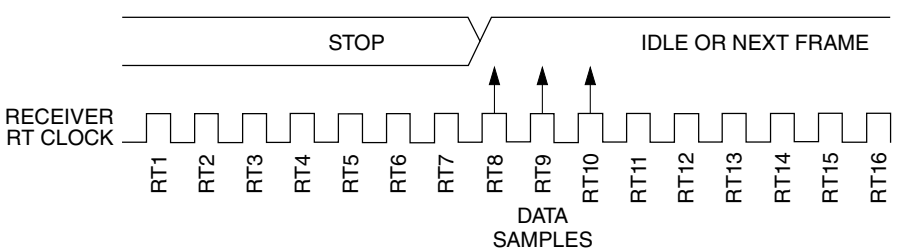


Figure 46-13. Fast data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 46-13, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 160 RT cycles (10 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$((154 - 160) \div 154) \times 100 = 3.90\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the Figure 46-13, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 176 RT cycles (11 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$((170 - 176) \div 170) \times 100 = 3.53\%$$

46.5.2.9 Receiver wakeup

C1[WAKE] determines how the UART is brought out of the standby state to process an incoming message. C1[WAKE] enables either idle line wakeup or address mark wakeup.

46.5.2.9.1 Idle input line wakeup (C1[WAKE] = 0)

In this wakeup method, an idle condition on the unsynchronized receiver input signal clears C2[RWU] and wakes the UART. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] and return to the standby state. C2[RWU] remains set and the receiver remains in standby until another idle character appears on the unsynchronized receiver input signal.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

When C2[RWU] is 1 and S2[RWUID] is 0, the idle character that wakes the receiver does not set S1[IDLE] or the receive data register full flag, S1[RDRF]. The receiver wakes and waits for the first data character of the next message which is stored in the receive data buffer. When S2[RWUID] and C2[RWU] are set and C1[WAKE] is cleared, any idle condition sets S1[IDLE] and generates an interrupt if enabled.

46.5.2.9.2 Address mark wakeup (C1[WAKE] = 1)

In this wakeup method, a logic 1 in the bit position immediately preceding the stop bit of a frame clears C2[RWU] and wakes the UART. A logic 1 in the bit position immediately preceeding the stop bit marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] and return to the standby state. C2[RWU] remains set and the receiver remains in standby until another address frame appears on the unsynchronized receiver input signal.

A logic 1 in the bit position immediately preceding the stop bit clears the receiver's C2[RWU] after the stop bit is received and places the received data into the receiver data buffer. Note that if Match Address operation is enabled i.e. C4[MAEN1] or C4[MAEN2] is set, then received frame is transferred to receive buffer only if the comparison matches.

Address mark wakeup allows messages to contain idle characters but requires that the bit position immediately preceding the stop bit be reserved for use in address frames.

If module is in standby mode and nothing triggers to wake the UART, no error flag is set even if an invalid error condition is detected on the receiving data line.

46.5.2.9.3 Match address operation

Match address operation is enabled when C4[MAEN1] or C4[MAEN2] is set. In this function, a frame received by the RX pin with a logic 1 in the bit position of the address mark is considered an address and is compared with the associated MA1 or MA2 register. The frame is transferred to the receive buffer, and S1[RDRF] is set, only if the comparison matches. All subsequent frames received with a logic 0 in the bit position of the address mark are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs, then no transfer is made to the receive data buffer, and all following frames with logic 0 in the bit position of the address mark are also discarded. If both C4[MAEN1] and C4[MAEN2] are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Match address operation functions in the same way for both MA1 and MA2 registers. Note that the position of the address mark is the same as the Parity Bit when parity is enabled for 8 bit and 9 bit data formats.

- If only one of C4[MAEN1] and C4[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If C4[MAEN1] and C4[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

46.5.3 Baud rate generation

A 13-bit modulus counter and a 5-bit fractional fine-adjust counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to SBR[12:0] determines the module clock divisor. The SBR bits are in the UART baud rate registers, BDH and BDL. The baud rate clock is synchronized with the module clock and drives the receiver. The fractional fine-adjust counter adds fractional delays to the baud rate clock to allow fine trimming of the baud rate to match the system baud rate. The transmitter is driven by the baud rate clock divided by 16. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to two sources of error:

- Integer division of the module clock may not give the exact target frequency. This error can be reduced with the fine-adjust counter.
- Synchronization with the module clock can cause phase shift.

The Table 46-9 lists the available baud divisor fine adjust values.

UART baud rate = UART module clock / (16 × (SBR[12:0] + BRFD))

The following table lists some examples of achieving target baud rates with a module clock frequency of 10.2 MHz, with and without fractional fine adjustment.

Table 46-9. Baud rates (example: module clock = 10.2 MHz)

Bits SBR (decimal)	Bits BRFA	BRFD value	Receiver clock (Hz)	Transmitter clock (Hz)	Target Baud rate	Error (%)
17	00000	0	600,000.0	37,500.0	38,400	2.3
16	10011	19/32=0.59375	614,689.3	38,418.08	38,400	0.047

Table continues on the next page...

Table 46-9. Baud rates (example: module clock = 10.2 MHz) (continued)

Bits SBR (decimal)	Bits BRFA	BRFD value	Receiver clock (Hz)	Transmitter clock (Hz)	Target Baud rate	Error (%)
33	00000	0	309,090.9	19,318.2	19,200	0.62
33	00110	6/32=0.1875	307,344.6	19,209.04	19,200	0.047
66	00000	0	154,545.5	9659.1	9600	0.62
133	00000	0	76,691.7	4793.2	4800	0.14
266	00000	0	38,345.9	2396.6	2400	0.14
531	00000	0	19,209.0	1200.6	1200	0.11
1062	00000	0	9604.5	600.3	600	0.05
2125	00000	0	4800.0	300.0	300	0.00
4250	00000	0	2400.0	150.0	150	0.00
5795	00000	0	1760.1	110.0	110	0.00

Table 46-10. Baud rate fine adjust

BRFA	Baud Rate Fractional Divisor (BRFD)
0 0 0 0 0	0/32 = 0
0 0 0 0 1	1/32 = 0.03125
0 0 0 1 0	2/32 = 0.0625
0 0 0 1 1	3/32 = 0.09375
0 0 1 0 0	4/32 = 0.125
0 0 1 0 1	5/32 = 0.15625
0 0 1 1 0	6/32 = 0.1875
0 0 1 1 1	7/32 = 0.21875
0 1 0 0 0	8/32 = 0.25
0 1 0 0 1	9/32 = 0.28125
0 1 0 1 0	10/32 = 0.3125
0 1 0 1 1	11/32 = 0.34375
0 1 1 0 0	12/32 = 0.375
0 1 1 0 1	13/32 = 0.40625
0 1 1 1 0	14/32 = 0.4375
0 1 1 1 1	15/32 = 0.46875
1 0 0 0 0	16/32 = 0.5
1 0 0 0 1	17/32 = 0.53125
1 0 0 1 0	18/32 = 0.5625
1 0 0 1 1	19/32 = 0.59375
1 0 1 0 0	20/32 = 0.625
1 0 1 0 1	21/32 = 0.65625
1 0 1 1 0	22/32 = 0.6875

Table continues on the next page...

Table 46-10. Baud rate fine adjust (continued)

BRFA	Baud Rate Fractional Divisor (BRFD)
1 0 1 1 1	23/32 = 0.71875
1 1 0 0 0	24/32 = 0.75
1 1 0 0 1	25/32 = 0.78125
1 1 0 1 0	26/32 = 0.8125
1 1 0 1 1	27/32 = 0.84375
1 1 1 0 0	28/32 = 0.875
1 1 1 0 1	29/32 = 0.90625
1 1 1 1 0	30/32 = 0.9375
1 1 1 1 1	31/32 = 0.96875

46.5.4 Data format

Each data character is contained in a frame that includes a start bit and a stop bit. The rest of the data format depends upon C1[M], C1[PE], S2[MSBF], BDH[SBNS] and C4[M10].

46.5.4.1 Eight-bit configuration

Clearing C1[M] configures the UART for 8-bit data characters, that is, eight bits are memory mapped in D. A frame with eight data bits has a total of 10 bits (This becomes 11 bits if BDH[SBNS] = 1). The most significant bit of the eight data bits can be used as an address mark to wake the receiver. If the most significant bit is used in this way, then it serves as an address or data indication, leaving the remaining seven bits as actual data. When C1[PE] is set, the eighth data bit is automatically calculated as the parity bit. See the following table.

Table 46-11. Configuration of 8-bit data format

UART_C1[PE]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	1	8	0	0	1
0	1	7	1[1]	0	1
1	1	7	0	1	1

1. The address bit identifies the frame as an address character. See Receiver wakeup.

NOTE

In the last column of the above table, the number of stop bits become 2 when BDH[SBNS] is set.

46.5.4.2 Nine-bit configuration

When C1[M] is set and C4[M10] is cleared and BDH[SBNS] is cleared, the UART is configured for 9-bit data characters. If C1[PE] is enabled, the ninth bit is either C3[T8/R8] or the internally generated parity bit. This results in a frame consisting of a total of 11 bits. In the event that the ninth data bit is selected to be C3[T8], it will remain unchanged after transmission and can be used repeatedly without rewriting it, unless the value needs to be changed. This feature may be useful when the ninth data bit is being used as an address mark.

When C1[M] and C4[M10] are set and BDH[SBNS] is cleared, the UART is configured for 9-bit data characters, but the frame consists of a total of 12 bits. The 12 bits include the start and stop bits, the 9 data character bits, and a tenth internal data bit. Note that if C4[M10] is set, C1[PE] must also be set. In this case, the tenth bit is the internally generated parity bit. The ninth bit can either be used as an address mark or a ninth data bit.

See the following table.

Table 46-12. Configuration of 9-bit data formats

C1[PE]	UC1[M]	C1[M10]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	0	0	See Eight-bit configuration				
0	0	1	Invalid configuration				
0	1	0	1	9	0	0	1
0	1	0	1	8	1 [1]	0	1
0	1	1	Invalid Configuration				
1	0	0	See Eight-bit configuration				
1	0	1	Invalid Configuration				
1	1	0	1	8	0	1	1
1	1	1	1	9	0	1	1
1	1	1	1	8	1[1]	1	1

1. The address bit identifies the frame as an address character.

NOTE

In the last column of the above table, the number of stop bits become 2 when BDH[SBNS] is set.

Note

Unless in 9-bit mode with M10 set, do not use address mark wakeup with parity enabled.

46.5.4.3 Timing examples

Timing examples of these configurations in the NRZ mark/space data format are illustrated in the following figures. The timing examples show all of the configurations in the following sub-sections along with the LSB and MSB first variations. This section explains the data formats available assuming single stop bit mode is selected.

46.5.4.3.1 Eight-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

Figure 46-14. Eight bits of data with LSB first

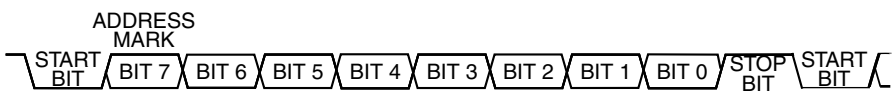


Figure 46-15. Eight bits of data with MSB first

46.5.4.3.2 Eight-bit format with parity enabled

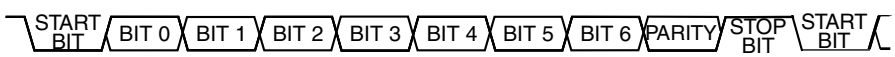


Figure 46-16. Seven bits of data with LSB first and parity

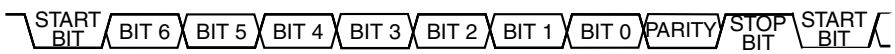


Figure 46-17. Seven bits of data with MSB first and parity

46.5.4.3.3 Nine-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

Figure 46-18. Nine bits of data with LSB first

Figure 46-19. Nine bits of data with MSB first

46.5.4.3.4 Nine-bit format with parity enabled

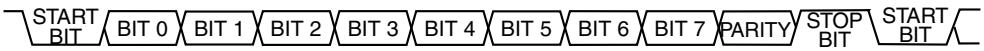


Figure 46-20. Eight bits of data with LSB first and parity

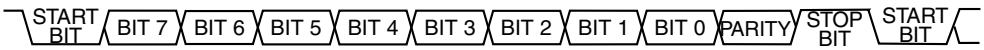


Figure 46-21. Eight bits of data with MSB first and parity

46.5.4.3.5 Non-memory mapped tenth bit for parity

The most significant memory-mapped bit can be used for address mark wakeup.

Figure 46-22. Nine bits of data with LSB first and parity

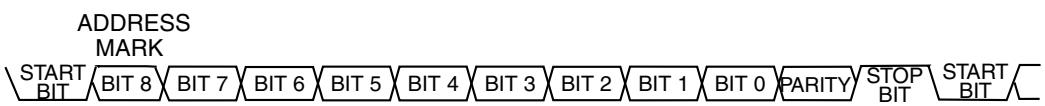


Figure 46-23. Nine bits of data with MSB first and parity

46.5.5 Single-wire operation

Normally, the UART uses two pins for transmitting and receiving. In single wire operation, the RXD pin is disconnected from the UART and the UART implements a half-duplex serial connection. The UART uses the TXD pin for both receiving and transmitting.

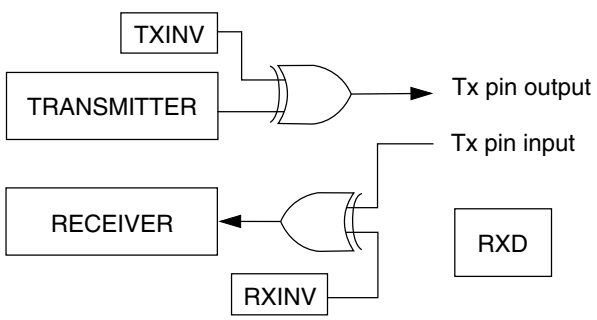


Figure 46-24. Single-wire operation (C1[LOOPS] = 1, C1[RSRC] = 1)

Enable single wire operation by setting C1[LOOPS] and the receiver source field, C1[RSRC]. Setting C1[LOOPS] disables the path from the unsynchronized receiver input signal to the receiver. Setting C1[RSRC] connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1).

46.5.6 Loop operation

In loop operation, the transmitter output goes to the receiver input. The unsynchronized receiver input signal is disconnected from the UART.

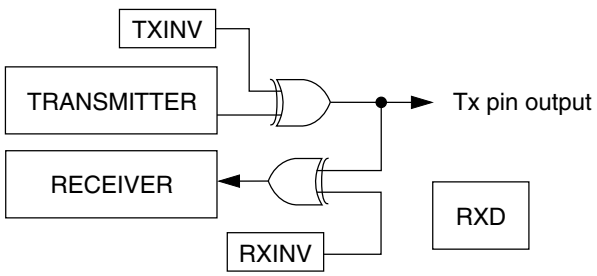


Figure 46-25. Loop operation (C1[LOOPS] = 1, C1[RSRC] = 0)

Enable loop operation by setting C1[LOOPS] and clearing C1[RSRC]. Setting C1[LOOPS] disables the path from the unsynchronized receiver input signal to the receiver. Clearing C1[RSRC] connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1).

46.6 Reset

All registers reset to a particular value are indicated in Memory map and registers.

46.7 System level interrupt sources

There are several interrupt signals that are sent from the UART. The following table lists the interrupt sources generated by the UART. The local enables for the UART interrupt sources are described in this table. Details regarding the individual operation of each interrupt are contained under various sub-sections of Memory map and registers. However, RXEDGIF description also outlines additional details regarding the RXEDGIF interrupt because of its complexity of operation. Any of the UART interrupt requests listed in the table can be used to bring the CPU out of Wait mode.

Table 46-13. UART interrupt sources

Interrupt Source	Flag	Local enable	DMA select
Transmitter	TDRE	TIE	TDMAS = 0
Transmitter	TC	TCIE	-
Receiver	IDLE	ILIE	-
Receiver	RDRF	RIE	RDMAS = 0
Receiver	LBKDIF	LBKDIE	LBKDDMAS = 0
Receiver	RXEDGIF	RXEDGIE	-
Receiver	OR	ORIE	-
Receiver	NF	NEIE	-
Receiver	FE	FEIE	-
Receiver	PF	PEIE	-
Receiver	RXUF	RXUFE	-
Transmitter	TXOF	TXOFE	-

46.7.1 RXEDGIF description

S2[RXEDGIF] is set when an active edge is detected on the RxD pin. Therefore, the active edge can be detected only when in two wire mode. A RXEDGIF interrupt is generated only when S2[RXEDGIF] is set. If RXEDGIE is not enabled before S2[RXEDGIF] is set, an interrupt is not generated.

46.7.1.1 RxD edge detect sensitivity

Edge sensitivity can be software programmed to be either falling or rising. The polarity of the edge sensitivity is selected using S2[RXINV]. To detect the falling edge, S2[RXINV] is programmed to 0. To detect the rising edge, S2[RXINV] is programmed to 1.

Synchronizing logic is used prior to detect edges. Prior to detecting an edge, the receive data on RxD input must be at the deasserted logic level. A falling edge is detected when the RxD input signal is seen as a logic 1 (the deasserted level) during one module clock cycle, and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input is seen as a logic 0 during one module clock cycle and then a logic 1 during the next cycle.

46.7.1.2 Clearing RXEDGIF interrupt request

Writing a logic 1 to S2[RXEDGIF] immediately clears the RXEDGIF interrupt request even if the RxD input remains asserted. S2[RXEDGIF] remains set if another active edge is detected on RxD while attempting to clear S2[RXEDGIF] by writing a 1 to it.

46.7.1.3 Exit from low-power modes

The receive input active edge detect circuit is still active on low power modes (Wait and Stop). An active edge on the receive input brings the CPU out of low power mode if the interrupt is not masked (S2[RXEDGIF] = 1).

46.8 DMA operation

In the transmitter, S1[TDRE] can be configured to assert a DMA transfer request. In the receiver, S1[RDRF],and S2[LBKDIF] can be configured to assert a DMA transfer request. The following table shows the configuration field settings required to configure each flag for DMA operation.

Table 46-14. DMA configuration

Flag	Request enable bit	DMA select bit
TDRE	TIE = 1	TDMAS = 1
RDRF	RIE = 1	RDMAS = 1
LBKDIF	LBKDIE = 1	LBKDDMAS = 1

When a flag is configured for a DMA request, its associated DMA request is asserted when the flag is set. When S1[RDRF] is configured as a DMA request, the clearing mechanism of reading S1, followed by reading D, does not clear the associated flag. The DMA request remains asserted until an indication is received that the DMA transactions are done. When this indication is received, the flag bit and the associated DMA request is cleared. If the DMA operation failed to remove the situation that caused the DMA request, another request is issued.

46.9 Application information

This section describes the UART application information.

46.9.1 Transmit/receive data buffer operation

The UART has independent receive and transmit buffers. The size of these buffers may vary depending on the implementation of the module. The implemented size of the buffers is a fixed constant via PFIFO[TXFIFOSIZE] and PFIFO[RXFIFOSIZE]. Additionally, legacy support is provided that allows for the FIFO structure to operate as a depth of one. This is the default/reset behavior of the module and can be adjusted using the PFIFO[RXFE] and PFIFO[TXFE] bits. Individual watermark levels are also provided for transmit and receive.

There are multiple ways to ensure that a data block, which is a set of characters, has completed transmission. These methods include:

1. Set TXFIFO[TXWATER] to 0. TDRE asserts when there is no further data in the transmit buffer. Alternatively the S1[TC] flag can be used to indicate when the transmit shift register is also empty.
2. Poll TCFIFO[TXCOUNT]. Assuming that only data for a data block has been put into the data buffer, when TCFIFO[TXCOUNT] = 0, all data has been transmitted or is in the process of transmission.
3. S1[TC] can be monitored. When S1[TC] asserts, it indicates that all data has been transmitted and there is no data currently being transmitted in the shift register.

46.9.2 Initialization sequence

To initiate a UART transmission:

1. Configure the UART.
    a. Select a baud rate. Write this value to the UART baud registers (BDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the BDH has no effect without also writing to BDL.
    b. Write to C1 to configure word length, parity, and other configuration bits (LOOPS, RSRC, M, WAKE, ILT, PE, and PT). Write to C4, MA1, and MA2 to configure.
    c. Enable the transmitter, interrupts, receiver, and wakeup as required, by writing to C2 (TIE, TCIE, RIE, ILIE, TE, RE, RWU, and SBK), S2 (MSBF and BRK13), and C3 (ORIE, NEIE, PEIE, and FEIE). A preamble or idle character is then shifted out of the transmitter shift register.

2. Transmit procedure for each byte.
    a. Monitor S1[TDRE] by reading S1 or responding to the TDRE interrupt. The amount of free space in the transmit buffer directly using TCFIFO[TXCOUNT] can also be monitored.
    b. If the TDRE flag is set, or there is space in the transmit buffer, write the data to be transmitted to (C3[T8]/D). A new transmission will not result until data exists in the transmit buffer.
3. Repeat step 2 for each subsequent transmission.

Note

During normal operation, S1[TDRE] is set when the shift register is loaded with the next data to be transmitted from the transmit buffer and the number of datawords contained in the transmit buffer is less than or equal to the value in TWFIFO[TXWATER]. This occurs 9/16ths of a bit time after the start of the stop bit of the previous frame.

To separate messages with preambles with minimum idle line time, use this sequence between messages.

1. Write the last dataword of the first message to C3[T8]/D.
2. Wait for S1[TDRE] to go high with TWFIFO[TXWATER] = 0, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting C2[TE].
4. Write the first and subsequent datawords of the second message to C3[T8]/D.

46.9.3 Overrun (OR) flag implications

To be flexible, the overrun flag (OR) operates slight differently depending on the mode of operation. There may be implications that need to be carefully considered. This section clarifies the behavior and the resulting implications. Regardless of mode, if a dataword is received while S1[OR] is set, S1[RDRF] and S1[IDLE] are blocked from asserting. If S1[RDRF] or S1[IDLE] were previously asserted, they will remain asserted until cleared.

46.9.3.1 Overrun operation

The assertion of S1[OR] indicates that a significant event has occurred. The assertion indicates that received data has been lost because there was a lack of room to store it in the data buffer. Therefore, while S1[OR] is set, no further data is stored in the data buffer until S1[OR] is cleared. This ensures that the application will be able to handle the overrun condition.

In most applications, because the total amount of lost data is known, the application will attempt to return the system to a known state. Before S1[OR] is cleared, all received data will be dropped. For this, the software does the following.

1. Remove data from the receive data buffer. This could be done by reading data from the data buffer and processing it if the data in the FIFO was still valuable when the overrun event occurred, or using CFIFO[RXFLUSH] to clear the buffer.
2. Clear S1[OR]. Note that if data was cleared using CFIFO[RXFLUSH], then clearing S1[OR] will result in SFIFO[RXUF] asserting. This is because the only way to clear S1[OR] requires reading additional information from the FIFO. Care should be taken to disable the SFIFO[RXUF] interrupt prior to clearing the OR flag and then clearing SFIFO[RXUF] after the OR flag has been cleared.

When LIN break detect (LBKDE) is asserted, S1[OR] has significantly different behavior than in other modes. S1[OR] will be set, regardless of how much space is actually available in the data buffer, if a LIN break character has been detected and the corresponding flag, S2[LBKDIF], is not cleared before the first data character is received after S2[LBKDIF] asserted. This behavior is intended to allow the software sufficient time to read the LIN break character from the data buffer to ensure that a break character was actually detected. The checking of the break character was used on some older implementations and is therefore supported for legacy reasons. Applications that do not require this checking can simply clear S2[LBKDIF] without checking the stored value to ensure it is a break character.

46.9.4 Match address registers

The two match address registers allow a second match address function for a broadcast or general call address to the serial bus, as an example.

46.9.5 Modem feature

This section describes the modem features.

46.9.5.1 Ready-to-receive using RTS

To help to stop overrun of the receiver data buffer, the RTS signal can be used by the receiver to indicate to another UART that it is ready to receive data. The other UART can send the data when its CTS signal is asserted. This handshaking conforms to the TIA-232-E standard. A transceiver is necessary if the required voltage levels of the communication link do not match the voltage levels of the UART's RTS and CTS signals.

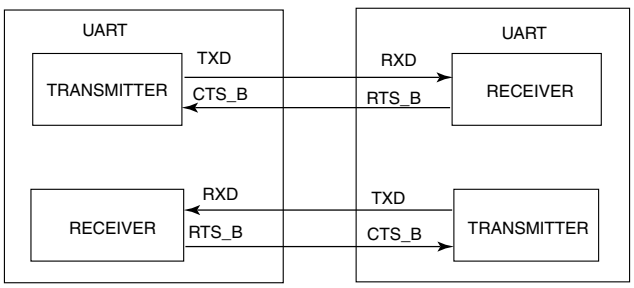


Figure 46-26. Ready-to-receive

The transmitter's CTS signal can be used for hardware flow control whether its RTS signal is used for hardware flow control, transceiver driver enable, or not at all.

46.9.5.2 Transceiver driver enable using RTS

RS-485 is a multiple drop communication protocol in which the UART transceiver's driver is 3-stated unless the UART is driving. The RTS signal can be used by the transmitter to enable the driver of a transceiver. The polarity of RTS can be matched to the polarity of the transceiver's driver enable signal. See the following figure.

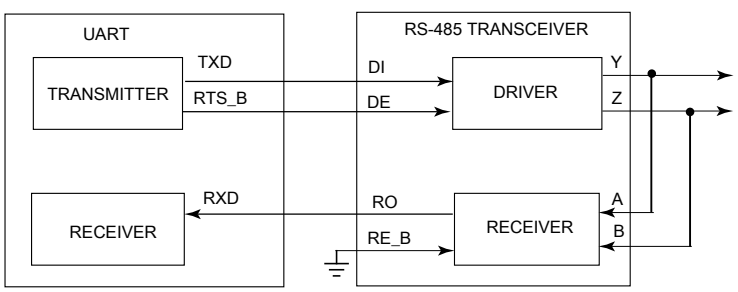


Figure 46-27. Transceiver driver enable using RTS

In the figure, the receiver enable signal is asserted. Another option for this connection is to connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled while driving. A pullup can pull RXD to a non-floating value during this time. This option can be refined further by operating the UART in single wire mode, freeing the RXD pin for other uses.

46.9.6 Legacy and reverse compatibility considerations

Recent versions of the UART have added several new features. Whenever reasonably possible, reverse compatibility was maintained. However, in some cases this was either not feasible or the behavior was deemed as not intended. This section describes several differences to legacy operation that resulted from these recent enhancements. If application code from previous versions is used, it must be reviewed and modified to take the following items into account. Depending on the application code, additional items that are not listed here may also need to be considered.

1. Various reserved registers and register bits are used, such as, MSFB and M10.
2. This module now generates an error when invalid address spaces are used.
3. While documentation indicated otherwise, in some cases it was possible for S1[IDLE] to assert even if S1[OR] was set.
4. S1[OR] will be set only if the data buffer (FIFO) does not have sufficient room. Previously, the data buffer was always a fixed size of one and the S1[OR] flag would set so long as S1[RDRF] was set even if there was room in the data buffer. While the clearing mechanism has remained the same for S1[RDRF], keeping the OR flag assertion tied to the RDRF event rather than the data buffer being full would have greatly reduced the usefulness of the buffer when its size is larger than one.
5. Previously, when C2[RWU] was set (and WAKE = 0), the IDLE flag could reassert up to every bit period causing an interrupt and requiring the host processor to reassert C2[RWU]. This behavior has been modified. Now, when C2[RWU] is set (and WAKE = 0), at least one non-idle bit must be detected before an idle can be detected.

Chapter 47
General-Purpose Input/Output (GPIO)

47.1 Chip-specific GPIO information

47.1.1 GPIO access protection

The GPIO module does not have access protection because it is not connected to a peripheral bridge slot.

47.1.2 Number of GPIO signals

The number of GPIO signals available on the devices covered by this document are detailed in Orderable part numbers.

Eight GPIO pins support a high drive capability - PTB0, PTB1, PTD4, PTD5, PTD6, PTD7, PTC3, and PTC4. All other GPIO support normal drive option only.

PTA4 includes a passive input filter that is enabled or disabled by PORTA_PCR4[PFE] control. This reset default is to have this function disabled.

PTC6 and PTC7 have true open-drain outputs thus when enabled as outputs, an external pull-resistor is required if output is logic 1.

47.2 Introduction

The GPIO registers support 8-bit, 16-bit or 32-bit accesses.

The GPIO data direction and output data registers control the direction and output data of each pin when the pin is configured for the GPIO function. The GPIO input data register displays the logic value on each pin when the pin is configured for any digital function, provided the corresponding Port Control and Interrupt module for that pin is enabled.

Efficient bit manipulation of the general-purpose outputs is supported through the addition of set, clear, and toggle write-only registers for each port output data register.

47.2.1 Features

Features of the GPIO module include:

- Port Data Input register visible in all digital pin-multiplexing modes
- Port Data Output register with corresponding set/clear/toggle registers
- Port Data Direction register

NOTE

The GPIO module is clocked by system clock.

47.2.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 47-1. Modes of operation

Modes of operation	Description
Run	The GPIO module operates normally.
Wait	The GPIO module operates normally.
Stop	The GPIO module is disabled.
Debug	The GPIO module operates normally.

47.2.3 GPIO signal descriptions

Table 47-2. GPIO signal descriptions

GPIO signal descriptions	Description	I/O
PORTA31–PORTA0	General-purpose input/output	I/O
PORTB31–PORTB0	General-purpose input/output	I/O
PORTC31–PORTC0	General-purpose input/output	I/O
PORTD31–PORTD0	General-purpose input/output	I/O
PORTE31–PORTE0	General-purpose input/output	I/O

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

47.2.3.1 Detailed signal description

Table 47-3. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
PORTA31–PORTA0 PORTB31–PORTB0 PORTC31–PORTC0 PORTD31–PORTD0 PORTE31–PORTE0	I/O	General-purpose input/output	
		State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.
		Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock. Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

47.3 Memory map and register definition

Any read or write access to the GPIO memory space that is outside the valid memory map results in a bus error.

GPIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F000	Port Data Output Register (GPIOA_PDOR)	32	R/W	0000_0000h	47.3.1/1325
400F_F004	Port Set Output Register (GPIOA_PSOR)	32	W (always reads 0)	0000_0000h	47.3.2/1326
400F_F008	Port Clear Output Register (GPIOA_PCOR)	32	W (always reads 0)	0000_0000h	47.3.3/1326

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F00C	Port Toggle Output Register (GPIOA_PTOR)	32	W (always reads 0)	0000_0000h	47.3.4/1327
400F_F010	Port Data Input Register (GPIOA_PDIR)	32	R	0000_0000h	47.3.5/1327
400F_F014	Port Data Direction Register (GPIOA_PDDR)	32	R/W	0000_0000h	47.3.6/1328
400F_F040	Port Data Output Register (GPIOB_PDOR)	32	R/W	0000_0000h	47.3.1/1325
400F_F044	Port Set Output Register (GPIOB_PSOR)	32	W (always reads 0)	0000_0000h	47.3.2/1326
400F_F048	Port Clear Output Register (GPIOB_PCOR)	32	W (always reads 0)	0000_0000h	47.3.3/1326
400F_F04C	Port Toggle Output Register (GPIOB_PTOR)	32	W (always reads 0)	0000_0000h	47.3.4/1327
400F_F050	Port Data Input Register (GPIOB_PDIR)	32	R	0000_0000h	47.3.5/1327
400F_F054	Port Data Direction Register (GPIOB_PDDR)	32	R/W	0000_0000h	47.3.6/1328
400F_F080	Port Data Output Register (GPIOC_PDOR)	32	R/W	0000_0000h	47.3.1/1325
400F_F084	Port Set Output Register (GPIOC_PSOR)	32	W (always reads 0)	0000_0000h	47.3.2/1326
400F_F088	Port Clear Output Register (GPIOC_PCOR)	32	W (always reads 0)	0000_0000h	47.3.3/1326
400F_F08C	Port Toggle Output Register (GPIOC_PTOR)	32	W (always reads 0)	0000_0000h	47.3.4/1327
400F_F090	Port Data Input Register (GPIOC_PDIR)	32	R	0000_0000h	47.3.5/1327
400F_F094	Port Data Direction Register (GPIOC_PDDR)	32	R/W	0000_0000h	47.3.6/1328
400F_F0C0	Port Data Output Register (GPIOD_PDOR)	32	R/W	0000_0000h	47.3.1/1325
400F_F0C4	Port Set Output Register (GPIOD_PSOR)	32	W (always reads 0)	0000_0000h	47.3.2/1326
400F_F0C8	Port Clear Output Register (GPIOD_PCOR)	32	W (always reads 0)	0000_0000h	47.3.3/1326
400F_F0CC	Port Toggle Output Register (GPIOD_PTOR)	32	W (always reads 0)	0000_0000h	47.3.4/1327
400F_F0D0	Port Data Input Register (GPIOD_PDIR)	32	R	0000_0000h	47.3.5/1327
400F_F0D4	Port Data Direction Register (GPIOD_PDDR)	32	R/W	0000_0000h	47.3.6/1328
400F_F100	Port Data Output Register (GPIOE_PDOR)	32	R/W	0000_0000h	47.3.1/1325

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F104	Port Set Output Register (GPIOE_PSOR)	32	W (always reads 0)	0000_0000h	47.3.2/1326
400F_F108	Port Clear Output Register (GPIOE_PCOR)	32	W (always reads 0)	0000_0000h	47.3.3/1326
400F_F10C	Port Toggle Output Register (GPIOE_PTOR)	32	W (always reads 0)	0000_0000h	47.3.4/1327
400F_F110	Port Data Input Register (GPIOE_PDIR)	32	R	0000_0000h	47.3.5/1327
400F_F114	Port Data Direction Register (GPIOE_PDDR)	32	R/W	0000_0000h	47.3.6/1328

47.3.1 Port Data Output Register (GPIOx_PDOR)

This register configures the logic levels that are driven on each general-purpose output pins.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 0h offset

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

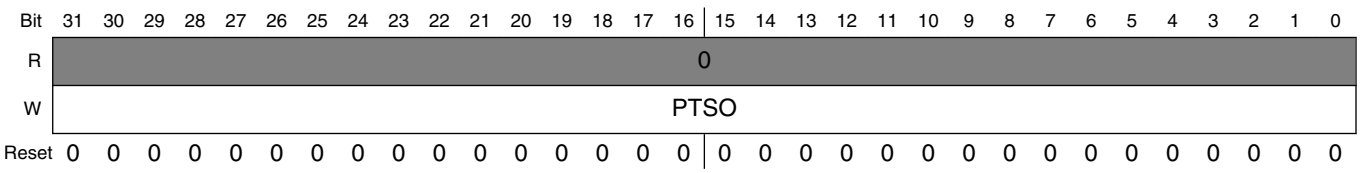
GPIOx_PDOR field descriptions

Field	Description
PDO	Port Data Output Register bits for unbonded pins return a undefined value when read. 0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

47.3.2 Port Set Output Register (GPIOx_PSOR)

This register configures whether to set the fields of the PDOR.

Address: Base address + 4h offset



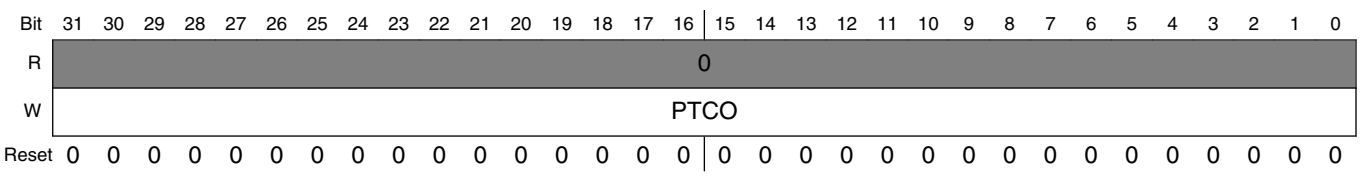
GPIOx_PSOR field descriptions

Field	Description
PTSO	Port Set Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic 1.

47.3.3 Port Clear Output Register (GPIOx_PCOR)

This register configures whether to clear the fields of PDOR.

Address: Base address + 8h offset



GPIOx_PCOR field descriptions

Field	Description
PTCO	Port Clear Output Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0.

47.3.4 Port Toggle Output Register (GPIOx_PTOR)

Address: Base address + Ch offset

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

GPIOx_PTOR field descriptions

Field	Description
PTTO	Port Toggle Output Writing to this register will update the contents of the corresponding bit in the PDOR as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

47.3.5 Port Data Input Register (GPIOx_PDIR)

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 10h offset

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

GPIOx_PDIR field descriptions

Field	Description
PDI	Port Data Input Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update. 0 Pin logic level is logic 0, or is not configured for use by digital function. 1 Pin logic level is logic 1.

47.3.6 Port Data Direction Register (GPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Address: Base address + 14h offset

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

GPIOx_PDDR field descriptions

Field	Description
PDD	Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

47.4 Functional description

47.4.1 General-purpose input

The logic state of each pin is available via the Port Data Input registers, provided the pin is configured for a digital function and the corresponding Port Control and Interrupt module is enabled.

The Port Data Input registers return the synchronized pin state after any enabled digital filter in the Port Control and Interrupt module. The input pin synchronizers are shared with the Port Control and Interrupt module, so that if the corresponding Port Control and Interrupt module is disabled, then synchronizers are also disabled. This reduces power consumption when a port is not required for general-purpose input functionality.

47.4.2 General-purpose output

The logic state of each pin can be controlled via the port data output registers and port data direction registers, provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

If	Then

Table continues on the next page...

A pin is configured for the GPIO function and the corresponding port data direction register bit is clear.	The pin is configured as an input.
A pin is configured for the GPIO function and the corresponding port data direction register bit is set.	The pin is configured as an output and and the logic state of the pin is equal to the corresponding port data output register.

To facilitate efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers exist to allow one or more outputs within one port to be set, cleared, or toggled from a single register write.

The corresponding Port Control and Interrupt module does not need to be enabled to update the state of the port data direction registers and port data output registers including the set/clear/toggle registers.

Chapter 48
JTAG Controller (JTAGC)

48.1 Introduction

The JTAGC block provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from the JTAGC block is communicated in serial format.

48.1.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. Refer to Register description for more information about the JTAGC registers.

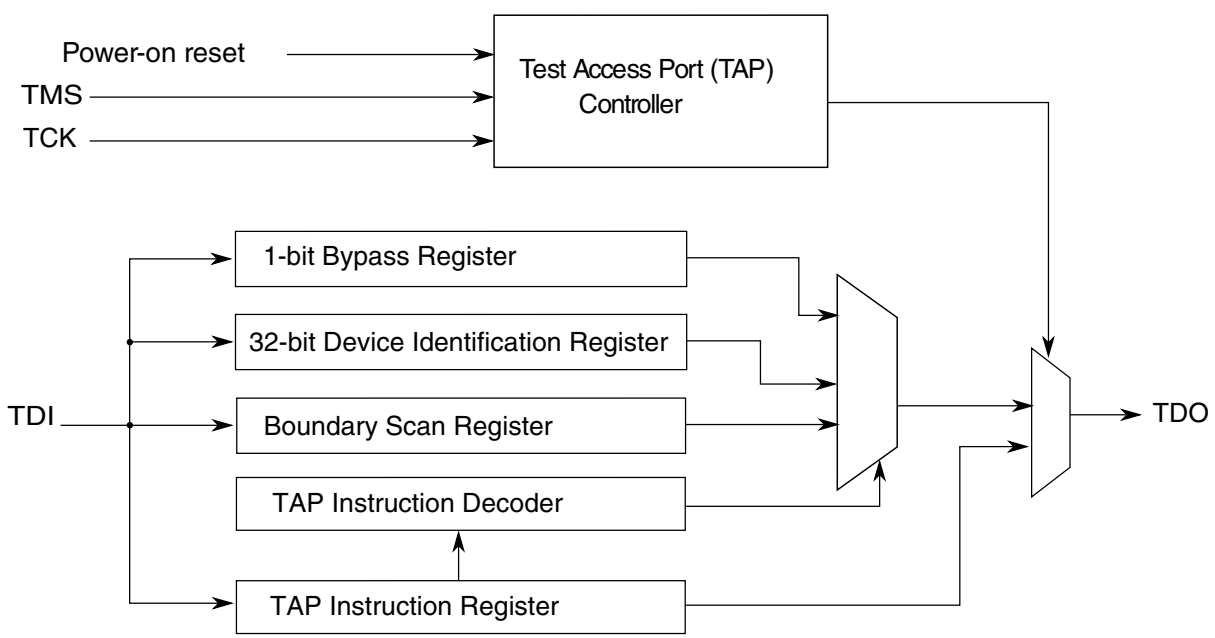


Figure 48-1. JTAG (IEEE 1149.1) block diagram

48.1.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 Test Access Port (TAP) interface
  - 4 pins (TDI, TMS, TCK, and TDO)
- Instruction register that supports several IEEE 1149.1-2001 defined instructions as well as several public and private device-specific instructions. Refer to Table 48-3 for a list of supported instructions.
- Bypass register, boundary scan register, and device identification register.
- TAP controller state machine that controls the operation of the data registers, instruction register and associated circuitry.

48.1.3 Modes of operation

The JTAGC block uses a power-on reset indication as its primary reset signals. Several IEEE 1149.1-2001 defined test modes are supported, as well as a bypass mode.

48.1.3.1 Reset

The JTAGC block is placed in reset when either power-on reset is asserted, or the TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state. Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset results in asynchronous entry into the reset state. While in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the on-chip system logic to continue unhindered
- The instruction register is loaded with the IDCODE instruction

48.1.3.2 IEEE 1149.1-2001 defined test modes

The JTAGC block supports several IEEE 1149.1-2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register while the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, SAMPLE and SAMPLE/PRELOAD. Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic while the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, SAMPLE or SAMPLE/PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the BYPASS, HIGHZ, CLAMP or reserved instructions are active. The functionality of each test mode is explained in more detail in JTAGC block instructions.

48.1.3.3 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. While in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

48.2 External signal description

The JTAGC consists of a set of signals that connect to off chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 48-1. JTAG signal properties

Name	I/O	Function	Reset State
TCK	Input	Test Clock	Weak pulldown
TDI	Input	Test Data In	Weak pullup
TDO	Output	Test Data Out	High Z[1]
TMS	Input	Test Mode Select	Weak pullup

1. TDO output buffer enable is negated when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

48.2.1 TCK—Test clock input

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

48.2.2 TDI—Test data input

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

48.2.3 TDO—Test data output

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is three-stateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

48.2.4 TMS—Test mode select

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

48.3 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

48.3.1 Instruction register

The JTAGC block uses a 4-bit instruction register as shown in the following figure. The instruction register allows instructions to be loaded into the block to select the test to be performed or the test data register to be accessed or both. Instructions are shifted in through TDI while the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states. Synchronous entry into the

Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the Capture-IR TAP controller state, the instruction shift register is loaded with the value 0001b , making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

	3	2	1	0
R	0	0	0	1
W	Instruction Code			
Reset:	0	0	0	1

Figure 48-2. Instruction register

48.3.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the BYPASS, CLAMP, HIGHZ or reserve instructions are active. After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

48.3.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP. The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state while the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Part Revision Number				Design Center						Part Identification Number					
W																
Reset	PRN				DC						PIN					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Part Identification Number				Manufacturer Identity Code											1
W																
Reset	PIN (contd.)				MIC											1

The following table describes the device identification register functions.

Table 48-2. Device identification register field descriptions

Field	Description
PRN	Part Revision Number. Contains the revision number of the part. Value is 0x0.
DC	Design Center. Indicates the design center. Value is .
PIN	Part Identification Number. Contains the part number of the device. .
MIC	Manufacturer Identity Code. Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID. Value is 0x00E.
IDCODE ID	IDCODE Register ID. Identifies this register as the device identification register and not the bypass register. Always set to 1.

48.3.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, SAMPLE or SAMPLE/PRELOAD instructions are active. It is used to capture input pin data, force fixed values on output pins, and select a logic value and direction for bidirectional pins. Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

48.4 Functional description

This section explains the JTAGC functional description.

48.4.1 JTAGC reset configuration

While in reset, the TAP controller is forced into the Test-Logic-Reset state, thus disabling the test logic and allowing normal operation of the on-chip system logic. In addition, the instruction register is loaded with the IDCODE instruction.

48.4.2 IEEE 1149.1-2001 (JTAG) Test Access Port

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction.

Data is shifted between TDI and TDO though the selected register starting with the least significant bit, as illustrated in the following figure. This applies for the instruction register, test data registers, and the bypass register.

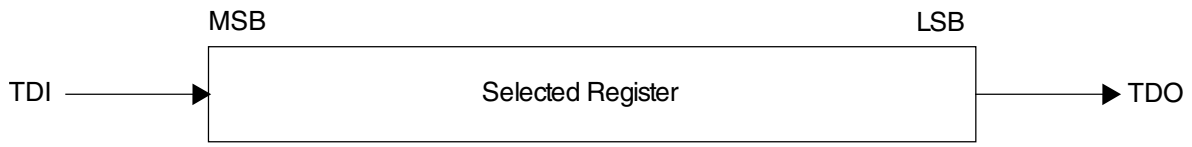
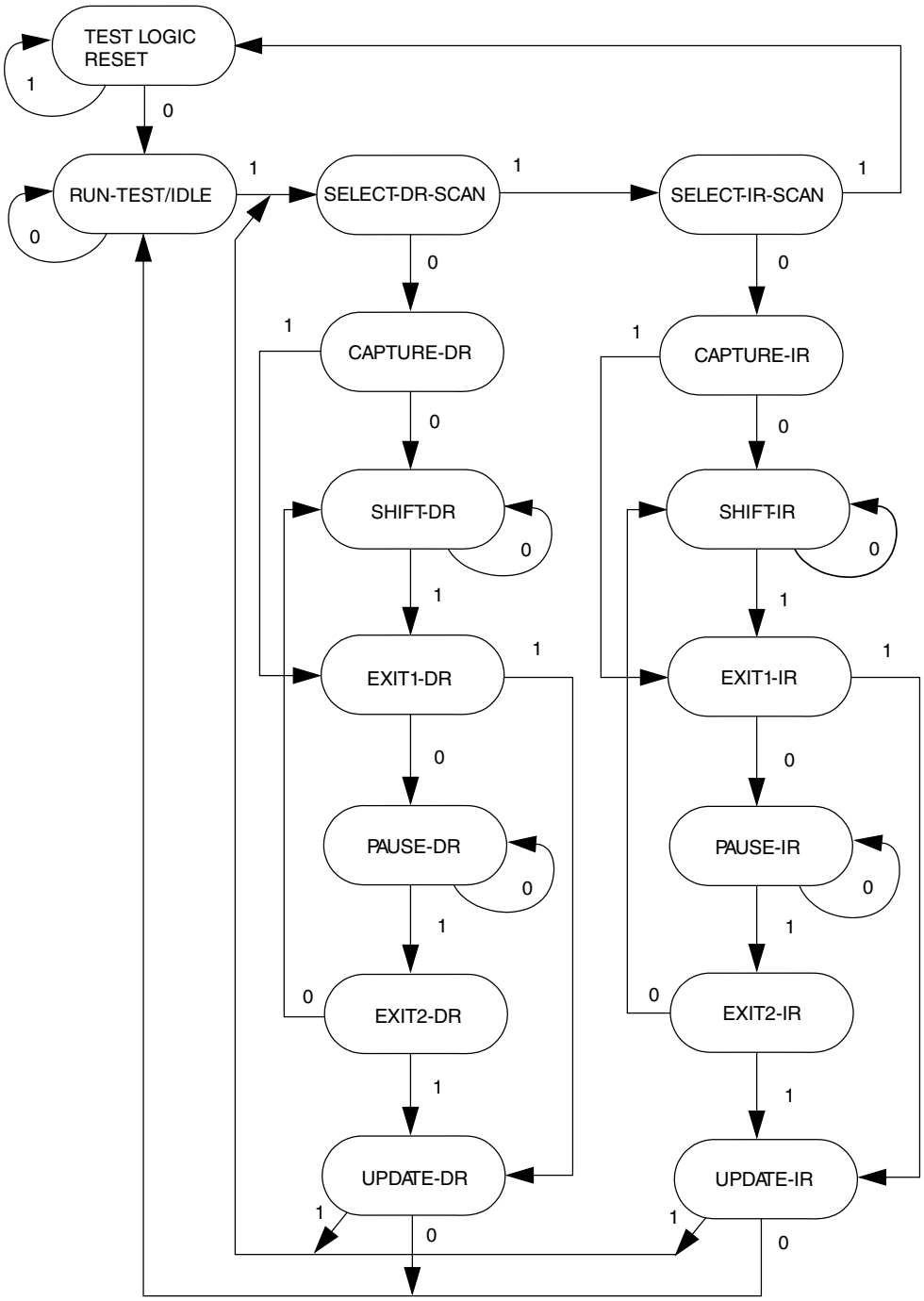


Figure 48-3. Shifting data through a register

48.4.3 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin. The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the following figure shows, holding TMS at logic 1 while clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 48-4. IEEE 1149.1-2001 TAP controller finite state machine

48.4.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting the JTAGC enable to a logic 1 value.

48.4.3.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions while the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated once the required number of bits have been acquired.

48.4.4 JTAGC block instructions

The JTAGC block implements the IEEE 1149.1-2001 defined instructions listed in the following table. This section gives an overview of each instruction; refer to the IEEE 1149.1-2001 standard for more details. All undefined opcodes are reserved.

Table 48-3. 4-bit JTAG instructions

Instruction	Code[3:0]	Instruction summary
IDCODE	0000	Selects device identification register for shift
SAMPLE/PRELOAD	0010	Selects boundary scan register for shifting, sampling, and preloading without disturbing functional operation
SAMPLE	0011	Selects boundary scan register for shifting and sampling without disturbing functional operation
EXTEST	0100	Selects boundary scan register and applies preloaded values to output pins. **NOTE:** Execution of this instruction asserts functional reset.
Factory debug reserved	0101	Intended for factory debug only
Factory debug reserved	0110	Intended for factory debug only
Factory debug reserved	0111	Intended for factory debug only
Arm JTAG-DP Reserved	1000	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
HIGHZ	1001	Selects bypass register and three-states all output pins. **NOTE:** Execution of this instruction asserts functional reset.
Arm JTAG-DP Reserved	1010	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.

Table continues on the next page...

Table 48-3. 4-bit JTAG instructions (continued)

Instruction	Code[3:0]	Instruction summary
Arm JTAG-DP Reserved	1011	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
CLAMP	1100	Selects bypass register and applies preloaded values to output pins. **NOTE:** Execution of this instruction asserts functional reset.
Arm JTAG-DP Reserved	1110	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
BYPASS	1111	Selects bypass register for data operations

48.4.4.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

48.4.4.2 SAMPLE/PRELOAD instruction

The SAMPLE/PRELOAD instruction has two functions:

- The SAMPLE portion of the instruction obtains a sample of the system data and control signals present at the MCU input pins and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE/PRELOAD instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. Both the data capture and the shift operation are transparent to system operation.
- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

48.4.4.3 SAMPLE instruction

The SAMPLE instruction obtains a sample of the system data and control signals present at the MCU input pins and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. There is no defined action in the Update-DR state. Both the data capture and the shift operation are transparent to system operation.

48.4.4.4 EXTEST External test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the preloaded data is loaded into the boundary scan register using the SAMPLE/PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state while performing external boundary scan operations.

48.4.4.5 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. While HIGHZ is active all output drivers are placed in an inactive drive state (e.g., high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

48.4.4.6 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register while the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) while conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

48.4.4.7 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. While the BYPASS instruction is active the system logic operates normally.

48.4.5 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

48.5 Initialization/Application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Place the JTAGC in reset through TAP controller state machine transitions controlled by TMS
2. Load the appropriate instruction for the test or action to be performed

Appendix A
Release Notes for Revision 5

A.1 General changes throughout document

- Removed KMS chapter and KMS related information from the document
- In chapter MCM, removed the second Functional description section
- Updated ARM to Arm as per branding guidelines
- Removed FAC feature from the document
- Removed the Note that stated 48-pin LQFP package is not available and is in included in Package Your Way program for Kinetis MCUs
- Added bitfield description of 10 as Reserved in FTM_SC[CLKS]

A.2 About This Document chapter changes

- No substantial content changes

A.3 Introduction chapter changes

- No substantial content changes

A.4 Core overview chapter changes

- No substantial content changes

A.5 Memories and Memory Interfaces chapter changes

- No substantial content changes

A.6 Memory map chapter changes

- No substantial content changes

A.7 Clock distribution chapter changes

- Updated the CLocking Diagram
- Added "max." after 100 MHz frequency in RUN mode table in Internal clocking requirements
- Removed the bullet "TRACE_CLK generated by simple..." from External clocks
- Added Trace clock generation diagram

A.8 Power management chapter changes

- No substantial content changes

A.9 Security chapter changes

- Updated Flash Security.
- Updated last sentence of section Security Interactions with Debug.

A.10 Debug chapter changes

- Added content to section The Debug Port.
- Removed the last sentence of section TPIU.

A.11 Reset and Boot chapter changes

- Added MCU Resets and Reset Pin section

A.12 Signal Multiplexing chapter changes

- Removed CLKOUT32 pin from pinout table and pinout diagrams

A.13 Port Control and Interrupts (PORT) changes

- In the "Pin Control Register n (PORT_PCRn)" section, clarified the last paragraph in the note.

A.14 SIM changes

- Changed LPTMR to LPTMR0 in SIM_SOPT1[OSC32KSEL] description and SIM_SCGC5 register

A.15 Kinetis Flashloader changes

A.15.1 Chip-specific Kinetis Flashloader changes

- No substantial content changes

A.15.2 Kinetis Flash Bootloader changes

- In Get/SetProperty Command Properties section, added mentions of internal memory regions, and 2 footnotes.
- In "LPUART Peripheral" section (or UART section, depending on your specific device), added autobaud ping packet accuracy requirement.
- In FlashEraseAll command section, in "FlashEraseAll Command Packet Format" table, added MemoryID parameter to command packet parameters

- In Get/SetProperty Command Properties section, table "Properties used by Get/SetProperty Commands, sorted by Value", updated the description of RAMStartAddress and RAMSizeInBytes properties.

- In Introduction section, added "(even when flash memory is non-secured)" to FlashEraseAll row in the table.

- In Introduction section, removed "(even when flash memory is non-secured)" from FlashEraseAll row in the table.
- Added figure "Kinetis Flashloader RAM Memory Map" in Memory Maps.
- Added figure "Kinetis Flashloader Start-up Flowchart" in Start-up Process.
- Added content in section Clock Configuration.
- Added the sentence "The framing packet described in this section is used for serial peripherals including UART, I2C, SPI, and CAN." in section Framing Packet.
- Added note in section Flashloader Command API.

• In FillMemory command section, corrected missing text for "Writing to flash requires the start address to be" • In Read memory command updated the sentence "This command can read any region of Flash memory, SRAM_L and SRAM_U memory accessible by the CPU and not protected by security."
• Updated figure "Protocol Sequence for FlashEraseRegion Command" and table "FlashEraseRegion Command Packet Format" in FlashEraseRegion command.
• Minor update in the section Read memory command : clarified as "Flash memory, SRAM_L and SRAM_U memory".
• Added '16-byte aligned' to incomplete statement in Flash Erase Region command section

A.16 Reset Control Module changes (RCM)

• No substantial content changes

A.17 System Mode Controller changes (SMC)

• No substantial content changes

A.18 MCM changes

• In the "Features" section, added "Control and counting logic for embedded trace buffer (ETB) almost full". • In Introduction section, updated Cortex-M7 to Cortex-M4

A.19 PMC changes

• Throughout the chapter, clarified that voltage monitoring applies only to the VDD power supply domain. • For the Low-voltage detect (LVD) system section, replaced "This protects memory contents and controls MCU system states during supply voltage variations." with clarification: "When the VDD supply falls below a specific trip point, the LVD circuit puts the device into a reset state, preventing the device from attempting to operate below its voltage operating range."
• Updated the reset value of PMC_REGSC from 0x04 to 0x24

A.20 LLWU changes

• In Memory map/register definition, clarified the *Chip Reset not VLLS* note regarding what resets affect the register bitfield values.

A.21 AXBS-Lite changes

A.21.1 Chip-specific AXBS-Lite changes

- No substantial content changes

A.21.2 Crossbar switch module changes

- No substantial content changes

A.22 AIPS-Lite module changes

- Memory map/register definition : MPRA : If a four-bit master field is reserved, the most-significant bit is ROZ.

A.23 DMAMUX module changes

- No substantial content changes

A.24 eDMA module changes

- Error Status Register (DMA_ES) : In description, replaced "See the Error Reporting and Handling section" with "See Fault reporting and handling.

- DONE : Added "The access of this field is W0C, write-zero-to-clear." to the field description.
- Performance : Added row for 48-MHz system speed to “eDMA peak transfer rates (Mbytes/sec)” table and “eDMA peak request rate (MReq/sec)” table.

- Added note beginning with "Disable a channel’s hardware service request..." to Enable Request Register (DMA_ERQ) and Clear Enable Request Register (DMA_CERQ) descriptions.
- Added new section: Suspend/resume a DMA channel with active hardware service requests.

- SSIZE : Added note about privileged data access to field description.

- Minor editorial changes.
- Replaced "DSPI" with "SPI".

- In Suspend/resume a DMA channel with active hardware service requests : Changed Sigma Delta Analog to Digital Convertor (SDADC) to ADC.

A.25 EWM changes

- Updated the section The $\overline{\text{EWM\_out}}$ Signal.

A.26 WDOG changes

A.26.1 Chip-specific WDOG changes

- No substantial content changes

A.26.2 WDOG changes

- In the "Unlocking and updating the watchdog" section, added guidance in the "context switch" note to disable global interrupts before starting an update or refresh sequence.

A.27 XBAR changes

- No substantial content changes

A.28 XBARB changes

- No substantial content changes

A.29 AOI changes

- No substantial content changes

A.30 OSC changes

- No substantial content changes

A.31 MCG changes

• Minor rewording in the section "Using a 32.768 kHz reference". • Typo correction (FCRDIV) in the section "MCG Internal Reference Clock".
• Minor update in the note of bitfield MCG_C4[DMX32]. • Number correction in the "Features" section: Fast clock with five trim bits.

A.32 FMC changes

• No substantial content changes

A.33 FTFA changes

A.33.1 Chip-specific FTFA changes

• Removed FAC feature related content

A.33.2 FTFA changes

• Remove FAC fetaure related content

A.34 CRC changes

• No substantial content changes

A.35 Cyclic ADC changes

A.35.1 Chip-specific Cyclic ADC changes

• No substantial content changes

A.35.2 Cyclic ADC changes

- No substantial content changes

A.36 CMP changes

A.36.1 Chip-specific CMP changes

- No substantial content changes

A.36.2 CMP changes

- No substantial content changes

A.37 DAC changes

- No substantial content changes

A.38 PWM changes

- Updated Fault Protection and Fault Pin Filter section
- Updated the Manual Fault Clearing section and updated Automatic Fault Clearing section
- Added Manual Fault Clearing (FCTRL[FSAFE]=0), fault clears before reload boundary figure and updated title of Manual Fault Clearing (FCTRL[FSAFE]=0) figure to Manual Fault Clearing (FCTRL[FSAFE]=0), fault clears after reload boundary
- Editorial Updates

A.39 PDB changes

- In Pulse-Out n Enable register (POEN), in bit POEN (PDB Pulse-Out Enable), changed "Enables the pulse output. Only lower Y bits are implemented in this MCU." to "Enables the pulse output. Only lower 8 bits are implemented in this MCU."

See Memory map and register definition.

- In Status and Control register (PDB_SC) register, updated Load Mode Select (LDMOD) field description; made PRESCALAR field more clear; updated PDB Interrupt Flag (PDBIF) field description.
- In Channel n Status register (PDB_CHnS) register, updated PDB Channel Flags (CF) field description.
- In PDB pre-trigger and trigger outputs section, replaced "When the PDB counter reaches the value set in IDLY register," with "When the PDB counter reaches the value (CNT + 1),"
- In Updating the delay registers section, "Circumstances of update to the delay registers" table was renamed to "When delay registers are updated" table; in that table, SC[LDMOD] rows 01 and 11 were updated using "The PDB counter reaches PDB_MOD[MOD] + 1 value".

See Memory map and register definition.

- In Status and Control register field SC[LDMOD], for 01 value: "The internal registers are loaded with the values from their buffers when the PDB counter reaches the MOD register value after 1 is written to LDOK." is changed to "The internal registers are loaded with the values from their buffers when the PDB counter (CNT) =MOD + 1 CNT delay elapsed, after 1 is written to LDOK."
- In Status and Control register field SC[LDMOD], for 11 value: "The internal registers are loaded with the values from their buffers when either the PDB counter reaches the MOD register value or a trigger input event is detected, after 1 is written to LDOK." is changed to "The internal registers are loaded with the values from their buffers when either the PDB counter (CNT) = MOD + 1 CNT delay elapsed, or a trigger input event is detected, after 1 is written to LDOK.".
- In Channel n Status register field PDB_CHnS[CF] (the PDB Channel Flags): "The CF[m] field is set when the PDB counter matches the CHnDLYm. Write 0 to clear these bits." is changed to "The CF[m] field is set when the PDB counter (PDB_CNT) matches the value CHnDLYm + 1. Write 0 to clear CF."
- In Status and Control Register (SC) Load OK field (LDOK, bit0): "After 1 is written to the LDOK field, the values in the internal buffers of these registers are not effective, and new values cannot be written to the internal buffers until the existing values in the internal buffers are loaded into their corresponding registers." is changed to "Before 1 is written to the LDOK field, the values in the internal buffers of these registers are not effective, and new values cannot be written to the internal buffers until the existing values in the internal buffers are loaded into their corresponding registers."
- In Counter Register (CNT) description,added a note: "Writing to this read-only register will generate a transfer error (and possibly a hard fault)."

- for CHC1[TOS] PDB Channel Pre-Trigger Output Select, changed SETRIG to SWTRIG (a correction)

- To Status and Control Register's PDB Interrupt Flag field description (SC[PDBIF]), added "Writing 1 to PDBIF has no effect."

A.40 FTM changes

A.40.1 Chip-specific FTM changes

- In Instantiation Information
  - Added Note:FTM0 and FTM3 has Quadrature Decoder registers but doesn't route its pins out, so QE function can not be used on FTM0 and FTM3.
  - Updated FTM2 by FTM3 in second Note

A.40.2 FTM changes

- No substantial content changes

A.41 PIT module changes

• Removed the bullet “Reserved registers will be read as 0. Writes will have no effect” under the topic “PIT register descriptions”.
• Added the Note "If the pause is initiated, when the timer is nearing 0 (CVALn =0x0), the pause command may not make it to the IP before the timer expires and generates a trigger. Pause will be ignored if (CVALn =0x0), but the trigger will remain asserted until the pause is removed. The user is recommended to remove Pause and then clear the interrupt TFLGn[TIF]. If the timers are to be paused, sufficient time must be ensured for the IP to react." under the topic Timers. • Added the note "Write request to a register must have the byte enables asserted for the bytes user wants to update" under the topic memory_map_and_register_description.Memory map/register description

A.42 ENC changes

• Edit, but no changes to technical content • Added colors to figures, with some clean-up

A.43 LPTMR changes

• No substantial content changes

A.44 FlexCAN changes

A.44.1 Chip-specific FlexCAN changes

• No substantial content changes

A.44.2 FlexCAN module changes

• Table in CAN_CTRL2[RFFN] field description was modified to show only information for 16 message buffers.
• In Protocol timing, in bullet point about Time Segment 1 changed "4 to 16 time quanta" to "2 to 16 time quanta." • In Figure 43-3, changed "4...16" to "2...16" for Time Segment 1.
• Editorial cleanup throughout chapter.
• In Module Configuration Register (CAN_MCR), changed access type of MCR[11] from RW to ROZ. • In Module Configuration Register (CAN_MCR), changed access type of MCR[14] from RU to ROZ. • In Error Counter (CAN_ECR), changed access type of ECR[24-31] from RW to ROZ.

• In Error Counter (CAN_ECR), changed access type of ECR[16-23] from RW to ROZ. • In Control 2 register (CAN_CTRL2), changed access type of CTRL2[31] from RW to ROZ.
• Modified description of Rx Individual Mask Registers (CAN_RXIMR*n*) to note option of using RXIMR memory region as general access memory. • Modified Bus interface to explain how to use RXIMR memory region as general access memory.
• Rewrote Transmit process for increased clarity, including adding a link to Table 43-6 and adding a NOTE. • In Transmission abort mechanism, removed summary of procedure at end because it was unnecessary.
• Added note about register read/write access to description of Rx FIFO Information Register (CAN_RXFIR).
• In Interrupt Masks 1 register (CAN_IMASK1), changed field BUF31TO0M into BUF15TO0M and a reserved 16-bit field. • In Interrupt Flags 1 register (CAN_IFLAG1), changed field BUF31TO8I into BUF15TO8I and a reserved 16-bit field.
• In Figure 43-1, changed word "Chip" to "FlexCAN."
• In Interrupt Masks 1 register (CAN_IMASK1), changed access type of reserved 16-bit field from RW to RU. • In Interrupt Flags 1 register (CAN_IFLAG1), changed access type of reserved 16-bit field from W1C to RU.
• In CTRL1[CLKSRC] field description in Control 1 register (CAN_CTRL1) and in Protocol timing, added note about referring to the clock distribution chapter (module clocks table) to identify the proper clock source. • Minor editorial cleanup throughout chapter.

• Added a note about referring to the CAN Protocol standard (ISO 11898-1) in these locations:
  • Register description in Control 1 register (CAN_CTRL1)
  • CLKSRC description in Control 1 register (CAN_CTRL1)
  • BOFFREC description in Control 1 register (CAN_CTRL1)
  • Register description in Error Counter (CAN_ECR)
  • Register description in Error and Status 1 register (CAN_ESR1)
  • Register description in CRC Register (CAN_CRCR)
  • Register description in CAN Bit Timing Register (CAN_CBT)

• Minor editorial changes in Memory map/register definition.
• Removed paragraph in Memory map/register definition that read "The memory maps for the message buffers are in FlexCAN message buffer memory map."
• Modified Transmit process.
• Added note "It is strongly recommended that all the fields in MB_CS word..." to Transmit process.
• Fixed issue in Message buffer structure. Bits MB[31], MB[30] and MB[29] are reserved for classical CAN instances.
• In Receive process, changed text of a step in procedure for recommended way to read frame received in mailbox. It now reads, "Read the free running timer to unlock the mailbox."

A.45 SPI changes

• In Serial Peripheral Interface SPI configuration removed redundant list item for "Transmit First In First Out (TX FIFO) buffering mechanism"
• In Transmit FIFO Fill Interrupt or DMA Request added text to Note: "Configure the DMA to fill only one FIFO location per transfer." • In Receive FIFO Drain Interrupt or DMA Request added text: "Configure the DMA to drain only one FIFO location per transfer."
• In PUSH TX FIFO Register In Master Mode (SPI_PUSHR) description, changed second sentence from "Only 16-bits can be written into TXDATA field" to "User must write 16-bits data into TXDATA field."

• In Status Register (SPI_SR), edited field description of:
  • RFDF
  • In TFFF, changed SR[TFFF] description to "The TFFF bit can be cleared by writing 1 to it (TFFF) or by acknowledgement from the DMA controller to the TX FIFO full request, when the TX FIFO is full."

• In MTFE, added note When MTFE=1 with continuous SCK enabled.. • In Modified SPI Transfer Format (MTFE = 1, CPHA = 1) removed note When MTFE=1 with continuous SCK enabled..

• In MCR[CLR_RXF], made editorial change.
• In External Stop Mode, added Note about Module disable mode for Master and Slave under power saving features.
• Editorial fixes in CLR_RXF
• Added Note in SPI_SR[TXNXTPTR] in TXNXTPTR.
• Corrected the module name throughout the chapter to SPI.
• Changed SR[TFFF] description to "The TFFF bit can be cleared by writing 1 to it (TFFF) or by acknowledgement from the DMA controller to the TX FIFO full request, when the TX FIFO is full." in TFFF
• SOUT of DSPI Slave is changed to SOUT of SPI Slave in figure: "SPI Modified Transfer Format (MTFE=1, CPHA=0, fsck = fsys/4)" in Modified SPI Transfer Format (MTFE = 1, CPHA = 0)
• In Serial Peripheral Interface SPI configuration changed 'MCR is 0b00' to 'MCR is 2b00'. • In Functional description changed 'SPIx_MCR is 0b00' to 'SPIx_MCR is 2b00'. • In Module Configuration Register (SPI_MCR) added table in bit field PCSIS. • In PUSH TX FIFO Register In Master Mode (SPI_PUSHR) added table in bit field PCS.
• In Memory Map/Register Definition added a note: "f_P and f_{sys} are used interchangeably in this chapter."
• In SR[TFFF] bit added paragraph: "If FIFO is filled manually, and not by DMA ....."
• Added section Command First In First Out (CMD FIFO) Buffering Mechanism.

A.46 I2C changes

• No substantial content changes

A.47 UART changes

• No substantial content changes

A.48 GPIO changes

• No substantial content changes

A.49 JTAGC module changes

• In Table 48-1 of External signal description :
  • Moved content from Pull column to Reset State column and removed Pull column
  • Updated "Up" to "Weak pullup"
  • Updated "Down" to "Weak pulldown"

• In Block diagram, Removed text "the chip-specific configuration information as well as"

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number KV4XP100M168RM
Revision 5, 08/2019