# UM10858

## PN7462 HW user manual

Rev. 1.2 — 08 September 2017
314512

User manual
COMPANY PUBLIC

**Document information**

| Info | Content |
|---|---|
| **Keywords** | PN7462AU, NFC reader |
| **Abstract** | This document describes how to use the PN7462AU. |

### Revision history

| Rev | Date | Description |
|---|---|---|
| 1.2 | 20170908 | GPIO wakeup condition corrected<br>Pull-up and pull-down configuration for SPIM pins corrected<br>CRC polynomial specified |
| 1.1 | 20170216 | Max enhanced ESD protection changed to 12 kV<br>Dynamic Power Control (DPC) added |
| 1.0 | 20160225 | Initial version |

## Contact information

For more information, please visit: http://www.nxp.com

# 1. Introduction

## 1.1 General description

The PN7462AU is an ARM Cortex-M0 based, low-cost 32-bit NFC micro controller offering performance, low power, simple instruction set and memory addressing, together with reduced code size compared to existing architectures. It operates at CPU frequencies up to 20 MHz.

The peripheral complement of the PN7462AU includes 160 K of flash memory, 12 kB of SRAM data memory, 4 kB EEPROM, and host interfaces. High Speed UART, SPI, I²C and 2.0 USB interface, I²C and SPI master interfaces, four general-purpose counter/timers, a Random Number Generator, one CRC coprocessor and up to 21 general-purpose I/O pins.

Equipped with a highly integrated high power output NFC-IC for contactless communication at 13.56 MHz, the PN7462AU NFC micro controller is extending NXP portfolio to offer a one chip solution to build contactless application, everywhere NFC interface is required. Its high-power transceiver allows to achieve EMV compliance on RF level without additional external active components.

It supports the following operating modes:

- ISO/IEC 14443-A & B, MIFARE
- JIS X 6319-4 (comparable with FeliCa scheme)
- ISO/IEC 15693, ISO/IEC 18000-3 Mode 3
- NFC protocols - tag Reader/ Writer, P2P
- ISO/IEC 14443-type A card emulation
- EMVCo compliance

By integrating an ISO/IEC 7816 interface on the same silicon, it is also the first product on the market to propose a solution for Dual Interface smart card readers. PN7462AU contact interface offers a high level of security for the card by performing current limitation, short-circuit detection, ESD protection as well as supply supervision. An additional UART output is also implemented to address applications where more than one contact card slot is need, which enable an easy connection to multiple smart-card slot interfaces like NXP's TDA8026.

- $V_{CC}$ regulation at 5 V, 3 V, and 1.8 V
- Thermal and short-circuit protection on all card contacts
- Automatic activation and deactivation sequences initiated by software, or by hardware in the event of a short-circuit, card take-off, and overheating
- EMVCo compliance

## 1.2 Features and benefits

### 1.2.1 Integrated contact interface front-end

- Class A, B and C cards support (1.8 V, 3 V, and 5 V)
- DC-to-DC converter for Class A support starting at 3 V, Class B support starting 2.7 V
- Thermal and short-circuit protection on all contact cards
- Automatic activation and deactivation sequence: initiated by software or by hardware in case of short-circuit, card take-off, overheating, VDD or VDD drop-out.
- Enhanced ESD protection (> 12 kV)
- ISO/IEC 7816 compliance
- EMVCo 4.3 compliance
- Clock generation up to 10 MHz
- Synchronous card support
- Possibility to extend the number of contact interface thanks to the addition of slot extender like TDA8026

### 1.2.2 Integrated contactless interface front end

- High RF output power frontend IC for transfer speeds up to 848 kbit/s
- NFC IP1 and NPFC IP2 support
- Full NFC Tag support (Type 1, Type 2, Type 3, Type 4 A and B)
- P2P active and passive, target and initiator
- Card emulation ISO14443 type A
- ISO/IEC 14443 Type A and B
- MIFARE Classic card
- ISO/IEC 15693 and ISO/IEC 18000-3 mode 3
- iClass serial number support
- Low power card detection function
- Compliance with EMV Contactless protocol specification
- Compliance with NFC standard

### 1.2.3 Cortex M0 microcontroller

- Processor core
  - 32-bit ARM cortex M0 processor
  - Built-in Nested Vectored Interrupt Controller (NVIC)
  - Non-maskable interrupt
  - System Tick Timer 24 bits
  - Running frequency up to 20 MHz
  - Clock management to enable low power consumption
- Memory

  - Flash: 160 kB
  - RAM: 12 kB
  - EEPROM: 4 kB
  - 40 K boot ROM included, including USB mass storage primary boot loader for code download.
- Debug Option
  - Serial Wire Debug interface (SWD)
- Master Interfaces*:*
  - SPI half-duplex, up to 6.78 Mbit/s
  - I²C supporting fast mode plus, and clock stretching
- Host Interfaces
  - HSUART for serial communication, supporting standards speeds from 9600 to 115200 bds, and faster speed up to 1.288 Mbit/s
  - SPI half-duplex and full duplex, up to 7 Mbit/s
  - I²C Host supporting standard, fast and high speed mode with multiple address support
  - USB 2.0 full speed, with USB 3.0 hub connection capability
- Up to 21 General-Purpose I/O (GPIO) with configurable pull-up/pull-down resistors
- GPIOs 1 to 12 can be used as edge and level sensitive interrupt sources
- Power
  - Two reduced power modes: Sleep mode and hard power down mode
  - Suspend mode for USB host interface
  - Processor wake-up from hard power down mode, stand-by mode, suspend mode via: host interface, contact card interface, GPIOs, RF field detection
  - Integrated PMU to automatically adjust internal regulators, to minimize the power consumption during all possible power modes.
  - Power-on reset
  - RF supply; external, or using an integrated LDO (TXLDO configurable with 3 V, 3.3 V, 3.6 V, 4.5 V, and 4.75 V)
  - Pad voltage supply: external 3.3 V or 1.8 V, or using an integrated LDO (3.3 V supply)
  - Integrated contact interface voltage regulation for 1.8 V, 3 V, and 5 V card supply, including a DC-to-DC converter for support of Class A and Class B cards
- Timers
  - Four general-purpose timers
  - Programmable WatchDog Timer (WDT)
- CRC coprocessor
- Random number generator
- Clocks

  - Crystal oscillator 27.12 MHz
  - Dedicated PLL 48 MHz for the USB
  - Integrated HFO 20 MHz and LFO 380 kHz
- General
  - HVQFN64 packaging
  - Temperature range: -40 °C to 85 °C

## 1.3 Ordering information

**Table 1. Ordering information**

| Type Number | Package | | |
|---|---|---|---|
| | Name | Description | Version |
| PN7462AU | HVQFN64 | Plastic thermal enhanced very thin quad flat package; no leads; 64 terminals; body 9 × 9 × 0.85 mm | SOT804-1 |

## 1.4 Block diagram

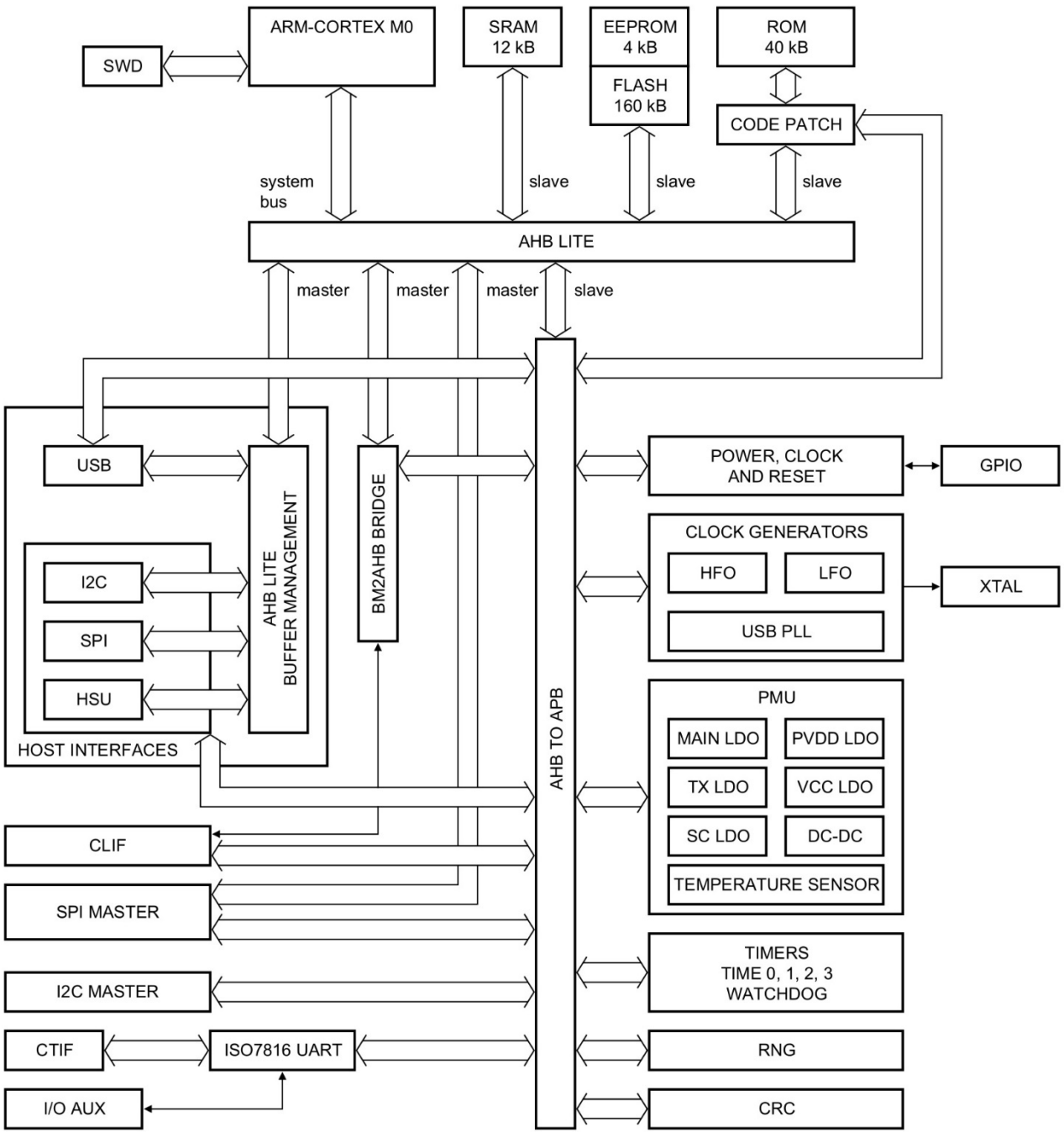



**Fig 1. Block diagram**

# 2. PN7462AU memory

## 2.1 Memory mapping

The PN7462AU incorporates several distinct memory regions. Fig 2 shows the overall map of the entire address space from user program viewpoint following reset. The APB peripheral area is 512 K in size, and is divided to allow for up to 32 peripherals. Only peripheral from 0 to 15 are accessible, and each one is allocated 16 kB of space, in order to simplify the address decoding for each peripheral. The APB peripheral memory map is shown on the right side of Fig 2.

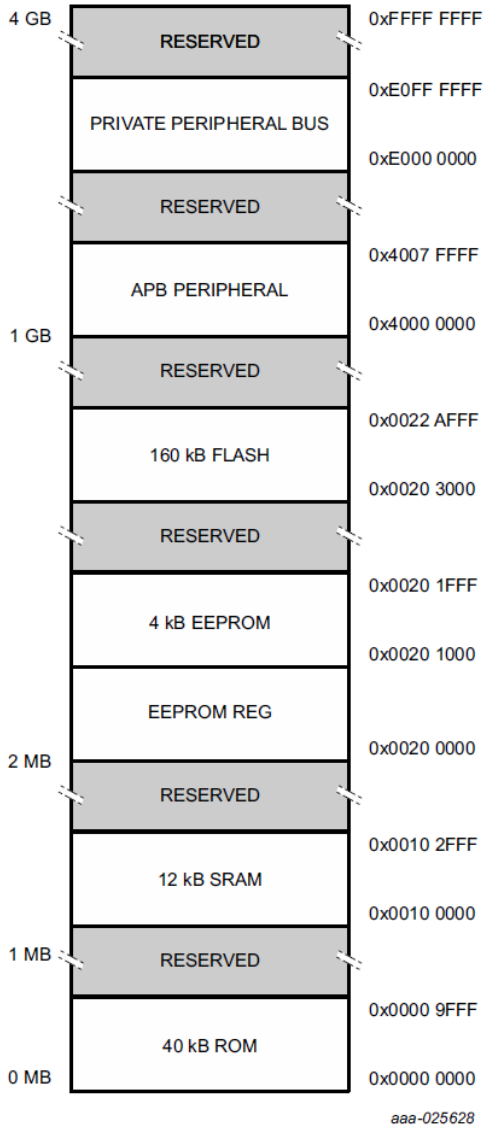



**Fig 2. PN7462 Memory map**

| APB ID | APB IF name | Connected IP | |
|---|---|---|---|
| 16 to 31 | Reserved | | 0x4004 8000 |
| 15 | Reserved | | 0x4004 0000 |
| 14 | Reserved | | 0x4003 C000 |
| 13 | SPIMASTER_APB | SPI Master IF | 0x4003 8000 |
| 12 | I2CMASTER_APB | I2C Master IF | 0x4003 4000 |
| 11 | Reserved | | 0x4003 0000 |
| 10 | USB_APB | HostIF (USB) IP | 0x4002 C000 |
| 9 | PCR_APB | PowerClockResetModule IP | 0x4002 8000 |
| 8 | HOST_APB | HostIF (I2C/SPI/HSU/BufMgt) IP | 0x4002 4000 |
| 7 | TIMERS_APB | Timer IP | 0x4002 0000 |
| 6 | RNG_APB | RNG IP | 0x4001 C000 |
| 5 | CTUART_APB | Contact UART IP | 0x4001 8000 |
| 4 | CLOCKGEN_APB | Clock Gen module | 0x4001 4000 |
| 3 | CRC_APB | CRC IP | 0x4001 0000 |
| 2 | PMU_APB | PMU modules | 0x4000 C000 |
| 1 | CL_APB | Contactless IP | 0x4000 8000 |
| 0 | Reserved | | 0x4000 4000 |
| | | | 0x4000 0000 |

**Fig 3. APB memory map**

## 2.2 On-chip flash memory map

The PN7462AU contains 160 kB on-chip flash program memory. The flash can be programmed using In-System Programming (ISP) or In-Application Programming (IAP) via the on-chip boot-loader software.

The flash memory is divided into two instances of 80 kB, with each sector consisting of individual pages of 64 Bytes. The flash memory map is described in Fig 4.

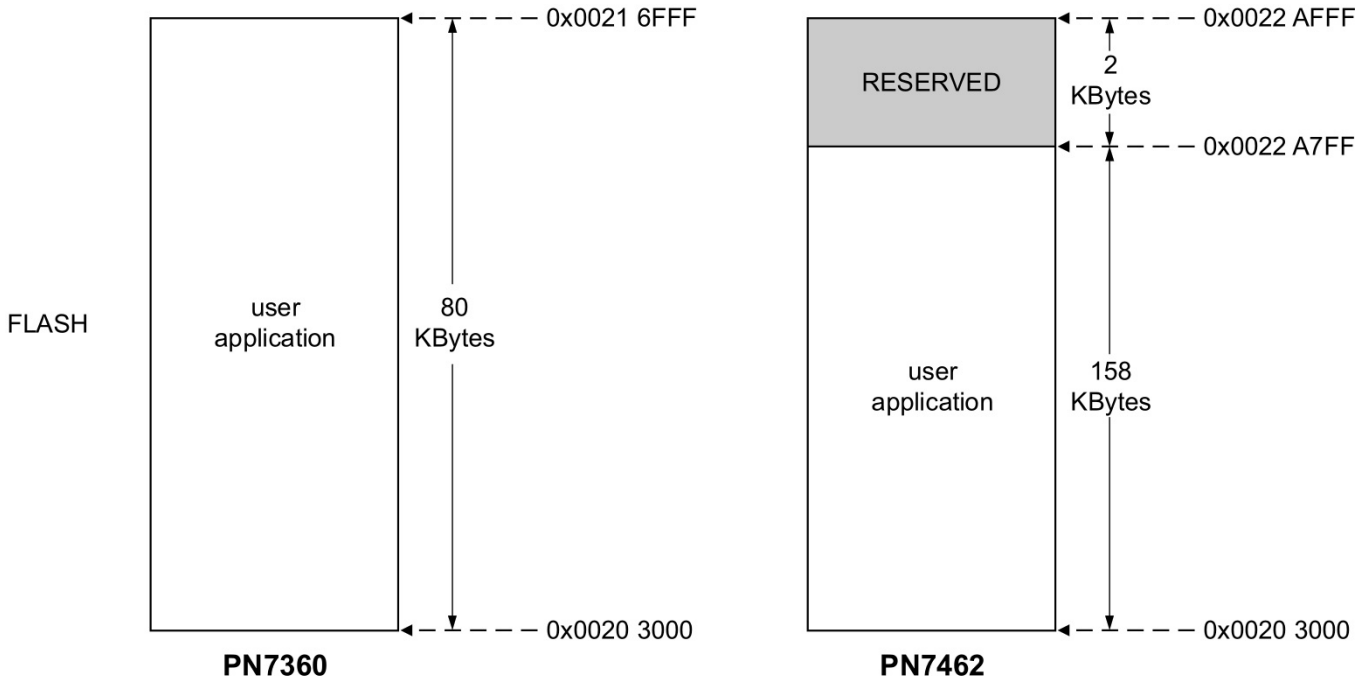



**Fig 4. Flash memory mapping**

## 2.3 EEPROM memory map

The PN7462AU embeds 4 kB of on-chip EEPROM data memory. The EEPROM memory map is shown in Fig 5.

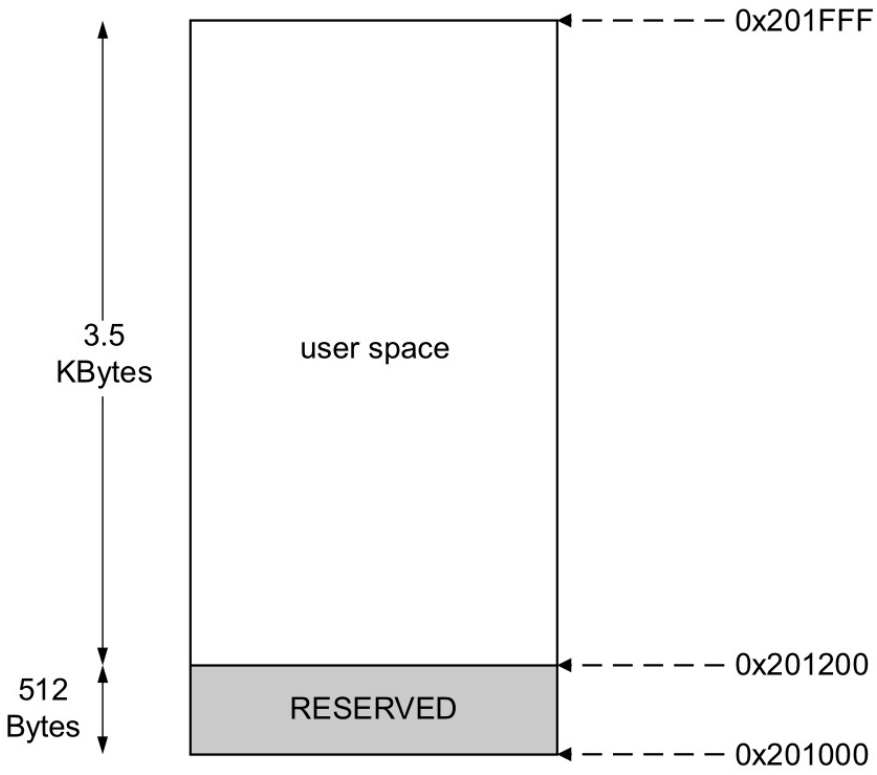



**Fig 5. EEPROM memory mapping**

## 2.4 SRAM memory map

The PN7462AU contains a total of 12 kB on-chip static RAM memory. The SRAM memory map is presented in Fig 6.

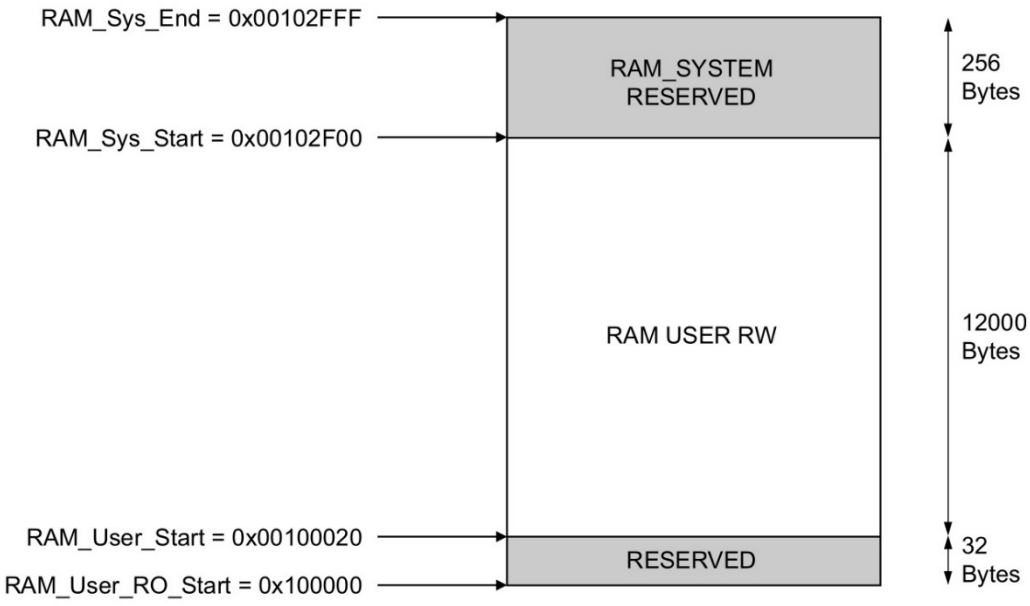



**Fig 6. SRAM memory mapping**

# 3. EEPROM CTRL module

The EEPROM controller module controls the access to 4 kB EEPROM and 160 kB On-chip Flash memories. This comprises:

- Reading the content of an EEPROM or Flash word in response to a read-transfer coming from the CPU
- Writing a word (8 ,16 and 32-bit word length transfer allowed for EEPROM, only 32-bit allowed for flash) in page file upon a write-transfer coming from the CPU (Full page has to be written for flash)
- Programming the Flash, meaning copying the content of the page into the relevant EEPROM/flash
- Implementing security bits (at boot time the EEPROM first word is read. Part of those bits are controlling the Flash access and the CMB enabling. These bits are available in a register). Fig 7 shows the block diagram EEPROM/flash controller module

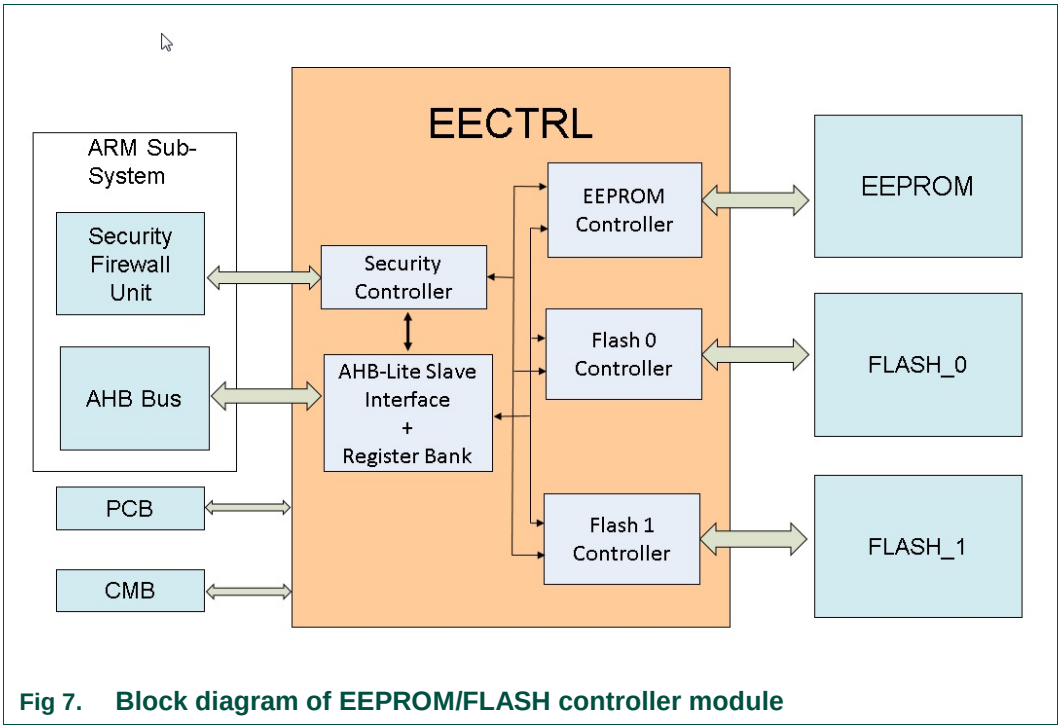



**Fig 7. Block diagram of EEPROM/FLASH controller module**

## 3.1 EEPROM/Flash controller features

- Three non-volatile memory ports: two for 80 x 2 kB of flash and one for a 4 kB of EEPROM
- 32-bit AHB-Lite slave interface with the CPU
- Read prefetching for flash memory to speed up reading from flash.
- Interrupt Request Device
- Register bank for control and status with user mode protection
- 2.8 ms maximum to write a page on the EEPROM, with a max. power consumption of 2 mA
- 2.5 ms to write a page on the Flash in typical condition
- 1.03 ms to write a page on the Flash with maximum clock speed, with a max. power consumption of 3 mA
- Test support Unit with CRC computation of the EEPROM and flash content

## 3.2 AHB interface

AMBA 3 AHB Lite slave interface is implemented to connect the CPU to the EECTRL. The wait states are inserted with individual length for a read or write access.

## 3.3 Memory map

The memory map for the EECTRL module is divided into four parts:

- 0x0020_0000 - 0x0020_0FFF - EECTRL registers area

- 0x0020_1000 - 0x0020_1FFF - 4 kBytes DATA EEPROM Area
- 0x0020_2000 - 0x0020_2FFF - UNDEFINED area
- 0x0020_3000 - 0x0022_AFFF - 160 kBytes flash area

## 3.4 EEPROM controller

### 3.4.1 Write operation

Write operation cannot be handled within one AHB clock cycle, therefore wait states are inserted by the AHB Slave Interface during page register write phase. The following table gives an overview of minimum idle time for a write operation. The EEPROM controller programs the data page by page. A page register for write access is used to internally store the data in a quick way and then a programming cycle can start. The targeted EEPROM is 16-bit oriented so 32-bit write access need additional processing to combine two 16-bit access leading to additional wait states.

**Table 2. Wait states for write access on EEPROM**

| Address width | Wait states FAST mode | | Wait states SLOW mode | |
|---|---|---|---|---|
| | AHB clock cycles | 20 MHz Clock | AHB clock cycles | 20 MHz Clock |
| 32-bit | 5 | 250 ns | 7 | 350 ns |
| 8/16-bit | 2 | 100 ns | 3 | 150 ns |

The Write operation to the EEPROM has to be done in two steps:

1. Write the 64-Byte short term storage page register.
2. Program the page register in one row of the EEPROM matrix.

### 3.4.2 Read operation

A read request cannot be handled within one AHB clock cycle, therefore wait states are inserted by the AHB Slave Interface. The targeted EEPROM is 16-bit oriented so 32-bit Read access need additional processing leading to additional wait states. Table 3 gives an overview of minimum idle time for a read operation.

**Table 3. Wait states for read access on EEPROM**

| Address Width | Wait states FAST mode | | Wait states SLOW mode | |
|---|---|---|---|---|
| | AHB clock cycles | 20 MHz clock | AHB clock cycles | 20 MHz clock |
| 32-bit | 5 | 250 ns | 7 | 350 ns |
| 8/16-bit | 2 | 100 ns | 3 | 150 ns |

## 3.5 Flash controller

The Flash is composed of two memory devices respectively controlled by one Flash controller integrated twice in the EECTRL module. The main difference to the EEPROM is a faster read access and a full-page programming.

### 3.5.1 Write operation

Write Access is done only with 32-bit access. A write operation cannot be handled within one AHB clock cycle, therefore wait states are inserted by the AHB Slave Interface during page register write phase. The following table gives an overview of minimum idle time for a write operation. The Flash controllers program the Flash_0 and Flash_1 data page by page.

A page register for write access is used to internally store the data in a quick way and then a programming cycle is necessary to store the data.

Flash_0 stores data corresponding to «even» AHB addresses with bit [2] = '0' and

Flash_1 stores data corresponding to «odd» AHB addresses with bit [2] = '1'.

**Table 4. Wait states for write access on EEPROM**

| Address Width | Wait states FAST mode | | Wait states SLOW mode | |
|---|---|---|---|---|
| | AHB clock cycles | ns (20 MHz clock) | AHB clock cycles | ns (20 MHz clock) |
| 32-bit access | 2 | 100 | 3 | 150 |

A write operation to the Flash has to be done in two steps:

1. Write the 64-Byte short term storage page register.
2. Program the page register in one row of the Flash matrix.

### 3.5.2 Read Operation

Even if 32-bit, 16-bit and 8-bit Read Access are supported at system level, read access is only done with 32-bit accesses at memory level because the Flash is 32-bit oriented. The read access is similar as for the EEPROM.

The following table gives idle time with AHB wait states insertion for a read operation

**Table 5. Wait states for read access on EEPROM**

| Address Width | Wait states FAST mode | | Wait states SLOW mode | |
|---|---|---|---|---|
| | AHB clock cycles | ns (20 MHz Clock) | AHB clock cycles | ns (20 MHz Clock) |
| 32- bit access | 2 | 100 | 3 | 150 |

## 3.6 Register overview

**Table 6. Clock generator register overview**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| EE_CTRL | 0000h | 32 | R/W | 001F_0000h | EECTRL general control register |
| EE_DYN | 0004h | 32 | W | 0000_0000h | EECTRL dynamic control register |
| EE_STAT_DAT | 0008h | 32 | R | 0X00_0000h | EEPROM status register |
| EE_STAT_COD | 000Ch | 32 | R | 0000_0000h | FLASH status register |
| EE_CRC_DAT | 0010h | 32 | R | FFFF_FFFFh | EEPROM CRC value |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| EE_CRC_DAT_ADDR | 0014h | 32 | R/W | 0FFF_0000h | EEPROM CRC start/end Addresses |
| EE_CRC_1_COD_INIT | 0018h | 32 | R/W | FFFF_FFFFh | FLASH_1 CRC init value |
| EE_CRC_1_COD | 001Ch | 32 | R | FFFF_FFFFh | FLASH_1 CRC value |
| EE_CRC_1_COD_ADDR | 0020h | 32 | R/W | 4FFF_0000h | FLASH_1 CRC start/end address |
| EE_CRC_0_COD_INIT | 0024h | 32 | R/W | FFFF_FFFFh | FLAH_0 CRC Init value |
| EE_CRC_0_COD | 0028h | 32 | R | FFFF_FFFFh | FLASH_0 CRC value |
| EE_CRC_0_COD_ADDR | 002Ch | 32 | R/W | 4FFF_0000h | FLASH_0 CRC start/end Addresses |
| RESERVED | 0030h | 32 | R/W | 0000_0000h | Reserved |
| RESERVED | 0034h | 32 | R/W | 0000_0000h | Reserved |
| RESERVED | 0038h | 32 | R/W | 0000_0000h | Reserved |
| EE_TRIMM | 003Ch | 32 | R/W | 0000_0000h | EEPROM/PFLASH trimming values |
| RESERVED | 0040h | 32 | R | 6000_01ACh | Reserved |
| EE_ECC_PF_AHB_ERROR_ADDR | 0044h | 32 | R | 0000_0000h | FLASH ECC Error address |
| Unused | 0048h - 0FD4h | 32 | - | - | Unused |
| EE_INT_CLR_ENABLE | 0FD8h | 32 | W | 0000_0000h | Interrupt CLR_ENABLE commands |
| EE_INT_SET_ENABLE | 0FDCh | 32 | W | 0000_0000h | Interrupt SET_ENABLE commands |
| EE_INT_STATUS | 0FE0h | 32 | R | 0000_0000h | Interrupt STATUS commands |
| EE_INT_ENABLE | 0FE4h | 32 | R | 0000_0000h | Interrupt ENABLE commands |
| EE_INT_CLR_STATUS | 0FE8h | 32 | W | 0000_0000h | Interrupt CLR_STATUS commands |
| EE_INT_SET_STATUS | 0FECh | 32 | W | 0000_0000h | Interrupt SET_STATUS commands |

## 3.7 Register description

**Table 7. EE_CTRL (address offset 0x0000h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | - | 0 | Reserved |
| 15 | ECC_PF_AHB_ERROR_ENABLE | R/W | 0 | when ’1’ enables the AHB error generation when FLASH read data cannot be corrected by the ECC mechanism and automatically set the FAST_COD bit to ’0’ to put the FLASH in slow mode. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 14 | PFLASH_READ_PREFETCH_DIS | R/W | 0 | When '1' disables read prefetching for the Flash memories |
| 13 | BLOCK_1_COD | R/W | 0 | Block mode for FLASH_1 |
| 12 | BNWSENS_1_COD | R/W | 0 | voltage drop sensor enable for FLASH_1 |
| 11 | SKIPPRG_1_COD | R/W | 0 | skip program if erase fails for FLASH_1 |
| 10 | STOP_1_COD | R/W | 0 | stop ramp-up at low power for FLASH_1 |
| 9 | PFLASH_DOUT_SYNCHRO_DIS | R/W | 0 | when '0' output PAGEFLASH data is synchronized with the system clock to ensure that following ECC calculation is made on stable data. It is automatically set to '1' if FAST_COD = '1'. |
| 8 | POWER_DOWN_1_COD | R/W | 0 | power down FLASH_1 block |
| 7 | BLOCK_0_COD | R/W | 0 | block mode for FLASH_0 |
| 6 | BNWSENS_0_COD | R/W | 0 | voltage drop sensor enable for FLASH_0 |
| 5 | SKIPPRG_0_COD | R/W | 0 | skip program if erase fails for FLASH_0 |
| 4 | STOP_0_COD | R/W | 0 | stop ramp-up at low power for FLASH_0 |
| 3 | FAST_COD | R/W | 0 | fast access for both FLASH_0 and FLASH_1. It is automatically set to '0' if EE_CTRL.ECC_PF_AHB_ERROR_ENABLE register is set to '1' |
| 2 | POWER_DOWN_0_COD | R/W | 0 | power down FLASH block |
| 1 | FAST_DAT | R/W | 0 | fast EEPROM data access |
| 0 | power_down_dat | R/W | 0 | power down EEPROM block |

**Table 8. EE_DYN (address offset 0x0004h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | - | 0 | Reserved |
| 23 | EE_RST_1_COD | W | 0 | resets the FLASH controller |
| 22 | CRC_CLEAR_1_COD | W | 0 | CRC clear function for FLASH |
| 21 | FULL_DUMP_READ_1_COD | W | 0 | performs a full or partial read of FLASH with CRC Calculation |
| 20 | PROG_1_COD | W | 0 | start programming data from the page register for FLASH_1 |
| 19 | EE_RST_0_COD | W | 0 | resets the FLASH Controller |
| 18 | CRC_CLEAR_0_COD | W | 0 | CRC clear function for EFLASH |
| 17 | FULL_DUMP_READ_0_COD | W | 0 | performs a full or partial read of FLASH with CRC calculation |
| 16 | PROG_0_COD | W | 0 | start programming data from the page register for FLASH_0 |
| 15:5 | RESERVED | - | 0 | Reserved |
| 4 | RESERVED | W | 0 | always write 0 |
| 3 | ee_rst_dat | W | 0 | resets the EEPROM Controller |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 2 | crc_clear_dat | W | 0 | CRC Clear Function for the EEPROM |
| 1 | full_dump_read_dat | W | 0 | performs a full or partial read of the EEPROM with CRC calculation |
| 0 | prog_dat | W | 0 | start programming data from the page register for the EEPROM |

**Table 9. EE_STAT_DAT (address offset 0x0008h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:23 | RESERVED | - | 0 | Reserved |
| 22 | ALL1_DAT | R | 0 | Single Fault Injection (SFI) detection flag for the EEPROM |
| 21 | ALL0_DAT | R | 0 | Single Fault Injection (SFI) detection flag for the EEPROM |
| 20 | BNWDROP_DAT | R | 0 | BNW sensor signal for the EEPROM |
| 19 | TMANALOG_DAT | R | 0 | analog test mode active, analog level at analog IO for the EEPROM |
| 18:3 | ee_edo_dat | R | 0 | EDO parity lines for EEPROM<br>EE_EDO_DAT[15:12]: parity bits corresponding to data[31:24]<br>EE_EDO_DAT[11:8]: parity bits corresponding to data[23:16]<br>EE_EDO_DAT[7:4]: parity bits corresponding to data[15:8]<br>EE_EDO_DAT[3:0]: parity bits corresponding to data[7:0] |
| 2 | readout_ongoing_dat | R | 0 | full or partial dump readout ongoing for the EEPROM |
| 1 | prog_dat | R | 0 | indicator if programming is ongoing for the EEPROM |
| 0 | hverr_dat | R | 0 | HV error signal for the EEPROM |

**Table 10. EE_STAT_COD (address offset 0x000Ch)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:26 | RESERVED | - | 0 | Reserved |
| 25 | ECC_READ_INVALID_1_COD | R | 0 | 1 means that read access can't be corrected, there is more than one bit error |
| 24 | ECC_READ_CORRECT_1_COD | R | 1 | 0 means that there is at least one bit error, check the ECC_READ_INVALID_1_COD to confirm if it has been corrected or not<br>1 means that no bit error occurred |
| 23:18 | EE_EDO_1_COD | R | X | EDO parity lines of the FLASH_1 32-bit data |
| 17 | readout_ongoing_1_COD | R | 0 | full or partial dump readout ongoing for the entire FLASH_1 |
| 16 | DROPSENS_1_COD | R | 0 | drop sensor output signal for FLASH_1 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 | VMPOK_1_COD | R | 0 | margin voltage flag for FLASH_1 |
| 14 | prog_1_COD | R | 0 | indicator if programming is ongoing for the FLASH_1 |
| 13 | hverr_1_COD | R | 0 | HV error signal for the FLASH_1 |
| 12 | ECC_READ_INVALID_0_COD | R | 0 | 1 means that read access can't be corrected, there is more than one bit error |
| 11 | ECC_READ_CORRECT_0_COD | R | 1 | 0 means that there is at least one bit error, check the ECC_READ_INVALID_0_COD to confirm if it has been corrected or not<br>1 means that no bit error occurred |
| 10:5 | EE_EDO_0_COD | R | X | EDO parity lines of the FLASH_0 32-bit data |
| 4 | readout_ongoing_0_COD | R | 0 | full or partial dump readout ongoing for the entire FLASH_0 |
| 3 | DROPSENS_0_COD | R | 0 | drop sensor output signal for FLASH_0 |
| 2 | VMPOK_0_COD | R | 0 | margin voltage flag for FLASH_0 |
| 1 | prog_0_COD | R | 0 | indicator if programming is ongoing for the FLASH_0 |
| 0 | hverr_0_COD | R | 0 | HV error signal for the FLASH_0 |

**Table 11. EE_CRC_DAT (address offset 0x0010h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | ee_crc_DAT | R | FFFFh | EEPROM CRC value |

**Table 12. EE_CRC_DAT_ADDR (address offset 0x0014h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | - | 0 | Reserved |
| 27:16 | ee_CRC_DAT_ADDR_END | R/W | 7FFh | EEPROM CRC calculation end address corresponding to a native 16-bit data access (AHB memory map divided by 2) |
| 15:12 | RESERVED | - | 0 | Reserved |
| 11:0 | ee_CRC_DAT_ADDR_START | R/W | 000h | EEPROM CRC calculation start address corresponding to a native 16-bit data access (AHB memory map divided by 2) |

**Table 13. EE_CRC_1_COD_INIT (address offset 0x0018h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | ee_crc_1_COD_INIT | R/W | FFFFh | FLASH_1 CRC Init Value loaded as soon as CRC_CLEAR_1_COD is high, meaning that FLASH_1 CRC must be set before CRC_CLEAR_1_COD. |

**Table 14. EE_CRC_1_COD (address offset 0x001Ch)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | ee_crc_1_COD | R | FFFFh | FLASH_1 CRC value |

Table 15. EE_CRC_1_COD_ADDR (address offset 0x0020h)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | ee_CRC_1_COD_ADDR_END | R/W | 4FFFh | FLASH_1 CRC calculation end address corresponding to a native 32-bit data access (AHB Memory Map divided by 8) |
| 15:0 | ee_CRC_1_COD_ADDR_START | R/W | 0000h | FLASH_1 CRC calculation start address corresponding to a native 32-bit data access (AHB memory map divided by 8) |

Table 16. EE_CRC_0_COD_INIT (address offset 0x0024h)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | ee_crc_0_COD_INIT | R/W | FFFFh | FLASH_0 CRC Init value loaded as soon as CRC_CLEAR_0_COD is high, meaning that FLASH_0 CRC must be set before CRC_CLEAR_0_COD. |

Table 17. EE_CRC_0_COD (address offset 0x0028h)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | ee_crc_0_COD | R | FFFFh | FLASH_0 CRC value |

Table 18. EE_CRC_0_COD_ADDR (address offset 0x002Ch)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | ee_CRC_0_COD_ADDR_END | R/W | 4FFFh | FLASH_0 CRC calculation end address corresponding to a native 32-bit data access (AHB memory map divided by 8) |
| 15:0 | ee_CRC_0_COD_ADDR_START | R/W | 0000h | FLASH_0 CRC Calculation start address corresponding to a native 32-bit data access (AHB memory map divided by 8) |

Table 19. EE_CRC_0_COD_ADDR (address offset 0x002Ch)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | ee_CRC_0_COD_ADDR_END | R/W | 4FFFh | FLASH_0 CRC calculation end address corresponding to a native 32-bit data access (AHB memory map divided by 8) |
| 15:0 | ee_CRC_0_COD_ADDR_START | R/W | 0000h | FLASH_0 CRC calculation start address corresponding to a native 32-bit data access (AHB memory map divided by 8) |

Table 20. EE_TRIMM (address offset 0x003Ch)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | - | 0 | Reserved |
| 23:20 | hvtrimw_1_COD | R/W | 0 | HV trimming value program for the FLASH_1 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 19:16 | hvtrime_1_COD | R/W | 0 | HV trimming value Erase for the FLASH_1 |
| 15:12 | hvtrimw_0_COD | R/W | 0 | HV trimming value Program for the FLASH_0 |
| 11:8 | hvtrime_0_COD | R/W | 0 | HV trimming value Erase for the FLASH_0 |
| 7:4 | hvtrimw_dat | R/W | 0 | HV trimming value Program for the EEPROM |
| 3:0 | hvtrime_dat | R/W | 0 | HV trimming value Erase for the EEPROM |

**Table 21. EE_ECC_PF_AHB_ERROR_ADDR (address offset 0x0044h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:18 | RESERVED | - | 0 | Reserved |
| 17:0 | ECC_PF_AHB_ERROR_ADDR | R | 0 | AHB address for which a flash read data was detected as invalid or corrected by the ECC module. |

**Table 22. EE_INT_CLR_ENABLE (address offset 0x0FD8h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | - | 0 | Reserved |
| 9 | EE_ECC_READ_NOT_CORRECT_1_COD_INT_CLR_ENABLE | W | 0 | FLASH_1 not correct ECC read interrupt clear enable command |
| 8 | EE_ECC_READ_INVALID_1_COD_INT_CLR_ENABLE | W | 0 | FLASH_1 Invalid ECC Read interrupt clear enable command |
| 7 | EE_ECC_READ_NOT_CORRECT_0_COD_INT_CLR_ENABLE | W | 0 | FLASH_0 Not Correct ECC Read interrupt clear enable command |
| 6 | EE_ECC_READ_INVALID_0_COD_INT_CLR_ENABLE | W | 0 | FLASH_0 Invalid ECC Read interrupt clear enable command |
| 5 | ee_hverr_1_cod_int_clr_enable | W | 0 | FLASH_1 High Voltage Error interrupt clear enable command |
| 4 | ee_hverr_0_cod_int_clr_enable | W | 0 | FLASH_0 High Voltage Error interrupt clear enable command |
| 3 | ee_hverr_dat_int_clr_enable | W | 0 | EEPROM High Voltage Error interrupt clear enable command |
| 2 | ee_prog_1_cod_completed_int_clr_enable | W | 0 | FLASH_1 Programming Completed interrupt clear enable command |
| 1 | ee_prog_0_cod_completed_int_clr_enable | W | 0 | FLASH_0 Programming Completed interrupt clear enable command |
| 0 | ee_prog_dat_completed_int_clr_enable | W | 0 | EEPROM Programming Completed interrupt clear enable command |

**Table 23. EE_INT_SET_ENABLE (address offset 0x0FDCh)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | Reserved | - | 0 | Reserved |
| 9 | EE_ECC_READ_NOT_CORRECT_1_COD_INT_SET_ENABLE | W | 0 | FLASH_1 not correct ECC read interrupt set enable command |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 8 | EE_ECC_READ_INVALID_1_COD_INT_SET_ENABLE | W | 0 | FLASH_1 Invalid ECC read interrupt set enable command |
| 7 | EE_ECC_READ_NOT_CORRECT_0_COD_INT_SET_ENABLE | W | 0 | FLASH_0 not correct ECC read interrupt set enable command |
| 6 | EE_ECC_READ_INVALID_0_COD_INT_SET_ENABLE | W | 0 | FLASH_0 invalid ECC read interrupt set enable command |
| 5 | ee_hverr_1_cod_int_SET_enable | W | 0 | FLASH_1 high voltage error interrupt set enable command |
| 4 | ee_hverr_0_cod_int_SET_enable | W | 0 | FLASH_0 high voltage error interrupt set enable command |
| 3 | ee_hverr_dat_int_SET_enable | W | 0 | EEPROM high voltage error interrupt set enable command |
| 2 | ee_prog_1_cod_completed_int_SET_enable | W | 0 | FLASH_1 programming completed interrupt set enable command |
| 1 | ee_prog_0_cod_completed_int_SET_enable | W | 0 | FLASH_0 programming completed interrupt set enable command |
| 0 | ee_prog_dat_completed_int_SET_enable | W | 0 | EEPROM programming completed interrupt set enable command |

**Table 24. EE_INT_STATUS (address offset 0x0FE0h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | Reserved | - | 0 | Reserved |
| 9 | EE_ECC_READ_NOT_CORRECT_1_COD_INT_STATUS | R | 0 | FLASH_1 not correct ECC read interrupt status variable |
| 8 | EE_ECC_READ_INVALID_1_COD_INT_STATUS | R | 0 | FLASH_1 Invalid ECC read interrupt status variable |
| 7 | EE_ECC_READ_NOT_CORRECT_0_COD_INT_STATUS | R | 0 | FLASH_0 not correct ECC read interrupt status variable |
| 6 | EE_ECC_READ_INVALID_0_COD_INT_STATUS | R | 0 | FLASH_0 Invalid ECC read interrupt status variable |
| 5 | ee_hverr_1_cod_int_STATUS | R | 0 | FLASH_1 high voltage error interrupt status variable |
| 4 | ee_hverr_0_cod_int_STATUS | R | 0 | PFLASH_0 high voltage error interrupt status variable |
| 3 | ee_hverr_dat_int_STATUS | R | 0 | EEPROM high voltage error interrupt status variable |
| 2 | ee_prog_1_cod_completed_int_STATUS | R | 0 | FLASH_1 programming completed interrupt status variable |
| 1 | ee_prog_0_cod_completed_int_STATUS | R | 0 | FLASH_0 programming completed interrupt status variable |
| 0 | ee_prog_dat_completed_int_STATUS | R | 0 | EEPROM programming completed interrupt status variable |

Table 25. EE_INT_ENABLE (address offset 0x0FE4h)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | - | 0 | Reserved |
| 9 | EE_ECC_READ_NOT_CORRECT_1_COD_INT_ENABLE | R | 0 | FLASH_1 not correct ECC read interrupt enable variable |
| 8 | EE_ECC_READ_INVALID_1_COD_INT_ENABLE | R | 0 | FLASH_1 Invalid ECC read interrupt enable variable |
| 7 | EE_ECC_READ_NOT_CORRECT_0_COD_INT_ENABLE | R | 0 | FLASH_0 Not Correct ECC read interrupt enable variable |
| 6 | EE_ECC_READ_INVALID_0_COD_INT_ENABLE | R | 0 | FLASH_0 invalid ECC read interrupt enable variable |
| 5 | ee_hverr_1_cod_int_SENABLE | R | 0 | FLASH_1 high voltage error interrupt enable variable |
| 4 | ee_hverr_0_cod_int_ENABLE | R | 0 | PFLASH_0 high voltage error interrupt enable variable |
| 3 | ee_hverr_dat_int_ENABLE | R | 0 | EEPROM high voltage error interrupt enable variable |
| 2 | ee_prog_1_cod_completed_int_ENABLE | R | 0 | FLASH_1 programming completed Interrupt Enable Variable |
| 1 | ee_prog_0_cod_completed_int_ENABLE | R | 0 | FLASH_0 programming completed interrupt enable variable |
| 0 | ee_prog_dat_completed_int_ENABLE | R | 0 | EEPROM programming completed interrupt enable variable |

Table 26. EE_INT_CLR_STATUS (address offset 0x0FE8h)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | - | 0 | Reserved |
| 9 | EE_ECC_READ_NOT_CORRECT_1_COD_INT_CLR_STATUS | W | 0 | FLASH_1 not correct ECC read interrupt clear status command |
| 8 | EE_ECC_READ_INVALID_1_COD_INT_CLR_STATUS | W | 0 | FLASH_1 Invalid ECC read interrupt clear status command |
| 7 | EE_ECC_READ_NOT_CORRECT_0_COD_INT_CLR_STATUS | W | 0 | FLASH_0 Not Correct ECC read interrupt clear status command |
| 6 | EE_ECC_READ_INVALID_0_COD_INT_CLR_STATUS | W | 0 | FLASH_0 Invalid ECC read interrupt clear status command |
| 5 | ee_hverr_1_cod_int_CLR_STATUS | W | 0 | FLASH_1 high voltage error interrupt clear status command |
| 4 | ee_hverr_0_cod_int_CLR_STATUS | W | 0 | FLASH_0 high voltage error interrupt clear status command |
| 3 | ee_hverr_dat_int_CLR_STATUS | W | 0 | EEPROM high voltage error interrupt clear status command |
| 2 | ee_prog_1_cod_completed_int_CLR_STATUS | W | 0 | FLASH_1 programming completed interrupt clear status command |
| 1 | ee_prog_0_cod_completed_int_CLR_STATUS | W | 0 | FLASH_0 programming completed interrupt clear status command |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 0 | ee_prog_dat_completed_int_CLR_STATUS | W | 0 | EEPROM programming completed interrupt clear status command |

**Table 27. EE_INT_SET_STATUS (address offset 0x0FECh)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | Reserved | - | 0 | Reserved |
| 9 | EE_ECC_READ_NOT_CORRECT_1_COD_INT_SET_STATUS | W | 0 | FLASH_1 not correct ECC read interrupt set status command |
| 8 | EE_ECC_READ_INVALID_1_COD_INT_SET_STATUS | W | 0 | PFLASH_1 Invalid ECC read interrupt set status command |
| 7 | EE_ECC_READ_NOT_CORRECT_0_COD_INT_SET_STATUS | W | 0 | FLASH_0 not correct ECC read interrupt set status command |
| 6 | EE_ECC_READ_INVALID_0_COD_INT_SET_STATUS | W | 0 | FLASH_0 invalid ECC read interrupt set status command |
| 5 | ee_hverr_1_cod_int_SET_STATUS | W | 0 | FLASH_1 high voltage error interrupt set status command |
| 4 | ee_hverr_0_cod_int_SET_STATUS | W | 0 | PFLASH_0 high voltage error interrupt set status command |
| 3 | ee_hverr_dat_int_SET_STATUS | W | 0 | EEPROM high voltage error interrupt set status command |
| 2 | ee_prog_1_cod_completed_int_SET_STATUS | W | 0 | FLASH_1 programming completed interrupt set status command |
| 1 | ee_prog_0_cod_completed_int_SET_STATUS | W | 0 | FLASH_0 programming completed interrupt set status command |
| 0 | ee_prog_dat_completed_int_SET_STATUS | W | 0 | EEPROM programming completed interrupt set status command |

# 4. Nested Vectored Interrupt Controller (NVIC)

The NVIC is an integral part of the Cortex-M0. The tight coupling to the CPU allows for a low interrupt latency and efficient processing of late arriving interrupts. The NVIC controls system exceptions and peripheral interrupts. Its control registers are accessible as memory-mapped devices.

## 4.1 NVIC features

- Controls system exceptions and peripheral interrupts
- Supports 32 vectored interrupts
- Four interrupt priority levels, with hardware priority level masking
- Non-mask able interrupt (NMI) connected to the watchdog interrupt.
- Software interrupt generation

## 4.2 Interrupt sources

The following table lists the interrupt sources available in the PN7462AU microcontroller

**Table 28. External interrupt sources**

| EIRQ# | Source | Description |
|---|---|---|
| 0 | Timer 0/1/2/3 | general-purpose timer 0/1/2/3 interrupt |
| 1 | - | Reserved |
| 2 | CLIF | contactless interface module interrupt |
| 3 | EECTRL | EEPROM controller |
| 4 | - | Reserved |
| 5 | - | Reserved |
| 6 | Host IF | TX or RX buffer from I2C, SPI, HSU, or USB module |
| 7 | Contact IF | ISO7816 contact module interrupt |
| 8 | - | Reserved |
| 9 | PMU/ TXLDO | power management unit (temperature sensor, TXLDO overcurrent detection, overload, VBUS level) |
| 10 | SPIMaster | TX or RX buffer from SPI master module |
| 11 | I2CMaster | TX or RX buffer from I2C master module |
| 12 | PCR | high temperature from temperature sensor 0 and 1, interrupt to CPU from PCR to indicate wakeup from suspend mode, out of standby, out of suspend, event on GPIO's configured as inputs. |
| 13 | PCR | interrupt common GPIO 1 to 12 |
| 14 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 1 |
| 15 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 2 |
| 16 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 3 |
| 17 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 4 |
| 18 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 5 |
| 19 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 6 |
| 20 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 7 |
| 21 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 8 |
| 22 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 9 |
| 23 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 10 |
| 24 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 11 |

| EIRQ# | Source | Description |
|---|---|---|
| 25 | PCR | interrupt (rise/fall/both-edge/level-high/level-low interrupt as programmed) GPIO 12 |
| 26 | - | Reserved |
| 27 | - | Reserved |
| 28 | - | Reserved |
| 29 | - | Reserved |
| 30 | - | Reserved |
| 31 | - | Reserved |
| NMI | WDT | the watchdog Interrupt is connected on the non-maskable interrupt |

## 4.3 NVIC register support in the SCS

The system control region includes status and configuration registers that apply to the NVIC as part of the general exception model.

All other external interrupt specific registers reside within the NVIC region of the SCS. Table 29 summarizes the NVIC specific registers in the SCS. Table 30 shows the NVIC_IPR bit assignments. In the table, N = 4n, where n is the NVIC_IPR register number. For example, for NVIC_IPR2, n is 2 and N is 8.

For more details, refer to DDI0419C_arm_architecture_v6m_reference_manual which can be found on the ARM webpage.

**Table 29. NVIC register overview**

| Address | Name | Type | Reset | Description |
|---|---|---|---|---|
| 0xE000E100 | NVIC_ISER | RW | 0x00000000 | enables, or reads the enabled state of one or more interrupts |
| 0xE000E104-0xE000E17F | | | | Reserved |
| 0xE000E180 | NVIC_ICER | RW | 0x00000000 | disables, or reads the enabled state of one or more interrupts |
| 0xE000E184-0xE000E1FF | | | | Reserved |
| 0xE000E200 | NVIC_ISPR | RW | 0x00000000 | on writes, sets the status of one or more interrupts to pending. On reads, shows the pending status of the interrupts. |
| 0xE000E204-0xE000E27F | | | | Reserved |
| 0xE000E280 | NVIC_ICPR | RW | 0x00000000 | On writes, clears the status of one or more interrupts to pending. On reads, shows the pending status of the interrupts. |
| 0xE000E300-0xE000E3FC | | | | Reserved |
| 0xE000E400- | NVIC_IPRn | RW | 0x00000000 | sets or reads interrupt priorities |

| Address | Name | Type | Reset | Description |
|---|---|---|---|---|
| 0xE000E41C | | | | |
| 0xE000E420-0xE000E43C | | | | Reserved |

**Table 30. NVIC_IPRn bit assignments**

| Bits | Name | Function |
|---|---|---|
| [31:30] | PRI_N3 | enables, or reads the enabled state of one or more interrupts |
| [29:24] | - | Reserved |
| [23:22] | PRI_N2 | disables, or reads the enabled state of one or more interrupts |
| [21:16] | - | Reserved |
| [15:14] | PRI_N1 | on writes, sets the status of one or more interrupts to pending. On reads, shows the pending status of the interrupts. |
| [13:8] | - | Reserved |
| [7:6] | PRI_N0 | on writes, clears the status of one or more interrupts to pending. On reads, shows the pending status of the interrupts. |
| [5:0] | - | Reserved |

## 4.4 SWD

Cortex-M0 processor-based devices use the Serial Wire ARM CoreSight™ debug technology. The Serial Wire Debug (SWD) signals are connected to the pads via the PCR (Power, Clock & Reset) described in Section 8. The SWD interface can be disabled in order to have code (or data) read/write access protection.

### 4.4.1 SWD features

- Run Control of the processor allowing to start and stop programs
- Single Step one source or assembler line
- Set breakpoints while the processor is running
- Write memory contents and peripheral registers on-the-fly
- “Printf” like debug messages through the SWD.

### 4.4.2 SWD limitations

The Pn7462AU does not allow breakpoint or single step debugging of ROM service APIs and boot code. Breakpoint or single step debugging of ROM service APIs and Boot code results into System reset.

Breakpoint and single step debugging is only allowed in the customer Application area.

### 4.4.3 Hardware connection of SWD

For using SWD it is recommended to connect an external pull-up from SWDCLK and SWDIO to PVDD_IN supply (see Table 31).

**Table 31. SWD pinning**

| Pin Number | Pin Name | Comment |
|---|---|---|
| 14 | SWDCLK | SWD clock |
| 15 | SWDIO | SWD I/O |

## 5. SysTick Timer (SysTick)

The SysTick timer is a 24-bit timer that counts down to zero and generates an interrupt. The SysTick timer is clocked from the system clock or from the reference clock, which is fixed to half the frequency of the system clock. In order to generate recurring interrupts at a specific interval, the SYST_RVR register must be initialized with the correct value for the desired interval. A default value is provided in the SYST_CALIB register and may be changed by software.

**Table 32. SysTick timer (base address 0xE000 E000)**

| Address | Name | Type | Reset | Description |
|---|---|---|---|---|
| 0xE000E010 | SYST_CSR | RW | 0x000 0000 | System Timer Control and status register |
| 0xE000E014 | SYST_RVR | RW | 0 | System Timer Reload value register |
| 0xE000E018 | SYST_CVR | RW | 0 | System Timer Current value register |
| 0xE000E1C | SYST_CALIB | RW | 0x4 | System Timer Calibration value register |

For more details, refer to DDI0419C_arm_architecture_v6m_reference_manual which can be found on the ARM webpage.

**Example timer calculation**

To use the system tick timer, do the following:

a. Program the SYST_RVR register with the reload value RELOAD to obtain the desired time interval.

b. Clear the SYST_CVR register by writing to it. This ensures that the timer will count from the SYST_RVR value rather than an arbitrary value when the timer is enabled.

c. Program the SYST_SCR register with the value 0x7 which enables the SysTick timer and the SysTick timer interrupt.

The following example illustrates selecting the SysTick timer reload value to obtain a 10 ms time interval with the system clock set to 20 MHz.

Example (system clock = 20 MHz)

The system tick clock = system clock = 20 MHz. Bit CLKSOURCE in the SYST_CSR register set to 1 (system clock).

RELOAD = (system tick clock frequency * 10 ms) -1 = (20 MHz * 10 ms) -1 = 200000-1= 199999 = 0x00030D3F.

# 6. PN7462AU Power management

## 6.1 Power supply

The PN7462AU is using following supply voltages:

- VBUS: Main supply voltage for internal analog modules, digital logic and memories
- VBUSP: Supply voltage for the contact interface
- TVDD_IN: Supply voltage for the contactless interface
- PVDD_IN: Pad voltage reference and supply voltage for the host interfaces (HSUART, USB, I2C and SPI) and the GPIOs
- PVDD_M_IN: Pad voltage reference and supply voltage for the master interfaces (SPI, I2C)
- DVDD: Supply voltage for the internal digital blocks
- VUP_TX: External supply voltage for the transmitter LDO (TXLDO)

### 6.1.1 Microcontroller supply

In order to use the PN7462AU as microcontroller with host interface following voltages (minimum requirements) need to be supplied:

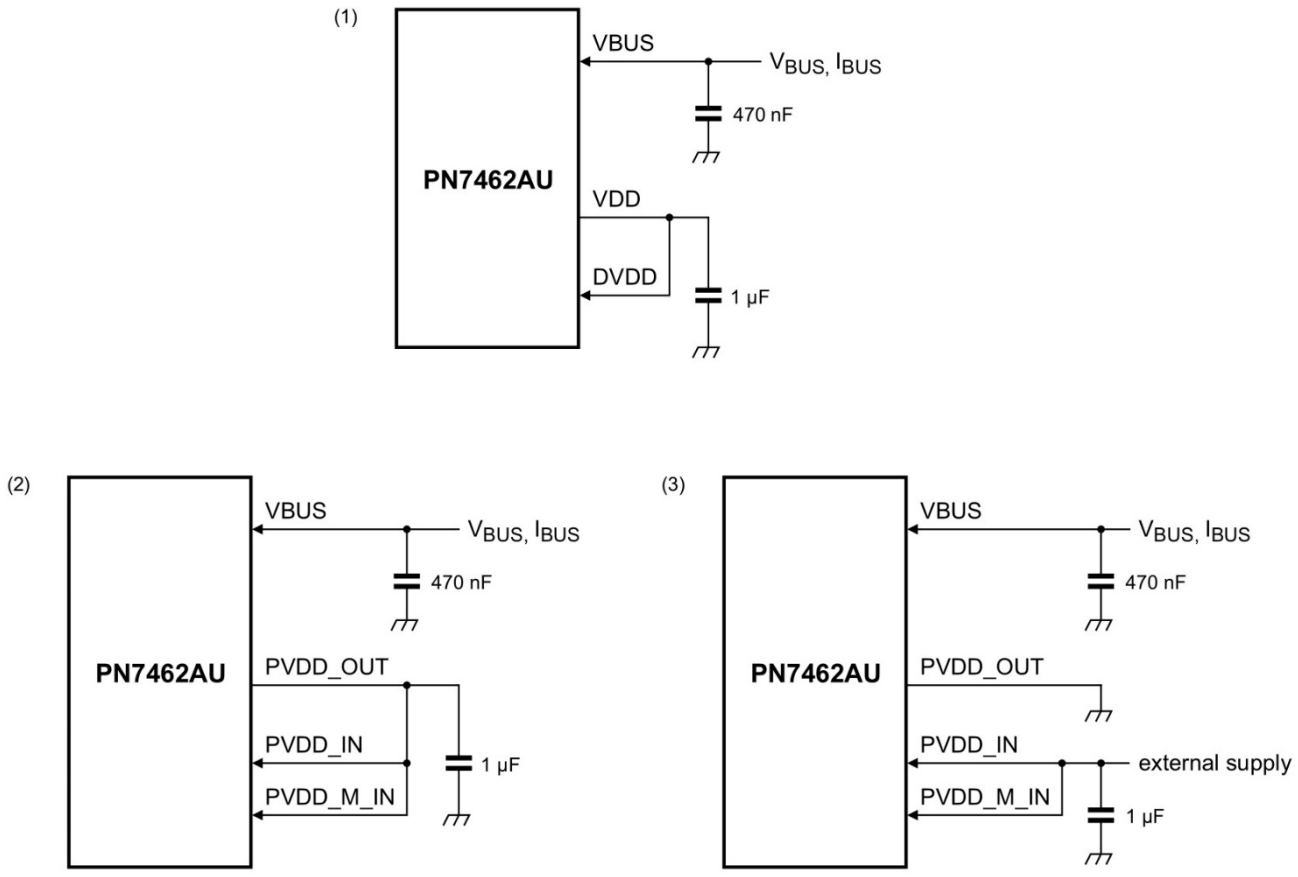
- VBUS
- PVDD_IN (if PVDD_OUT is not used, it needs to be shorted to GND)
- DVDD pin must be connected to VDD and buffered with 1 µF capacitor to GND

The SPI/ I²C master interface requires additional supply:

- PVDD_M_IN

Fig 8 shows the power supply of the chip (VBUS), including the supply of the digital blocks (DVDD). The host interface pads are supplied using PVDD_IN and master interface pads are supplied using PVDD_M_IN. The pads can be supplied using an internal LDO, or using an external supply. The internal LDO requires that VBUS > 4 V. When PVDD_LDO is used, the maximum total current available from PVDD_OUT for the pads supply is 30 mA.

When an external source is used for PVDD_IN and PVDD_IN_M, PVDD_OUT must be connected to the ground with a ground resistance of less than 10 Ω.




(1) Powering the microcontroller and the digital blocks (DVDD)

(2) 2 possibilities for powering the pad interfaces (PVDD_IN and PVDD_M_IN)

**Remark**: The capacitance must be chosen so that the capacitance value is correct 5 V.

**Fig 8. Powering the PN7462AU microcontroller**

### 6.1.2 Contactless reader supply

In order to use the PN7462AU as contactless reader the following voltages need to be supplied:

- VBUS
- PVDD_IN (if PVDD_OUT is not used, it needs to be shorted to GND)
- TVDD_IN (If supplied by TVDD_OUT, VUP_TX needs to be supplied)
- DVDD pin must be connected to VDD and buffered with 1µF capacitor to GND

For SPI/ I²C master following supply is also needed:

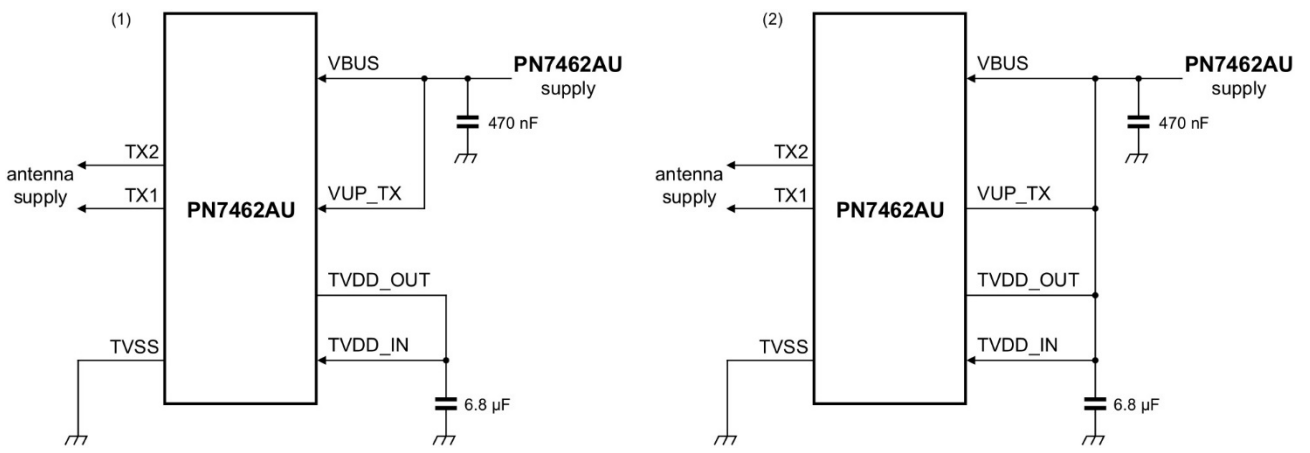
- PVDD_M_IN

The contactless interface is powered through TVDD_IN. This pin can be supplied either externally or by using TVDD_OUT.

In case TVDD_OUT is used

- VUP_TX needs to be supplied.

**Remark**: The TVDD_OUT pin must not be left floating, and needs to be at the same voltage as the TVDD_IN pin.

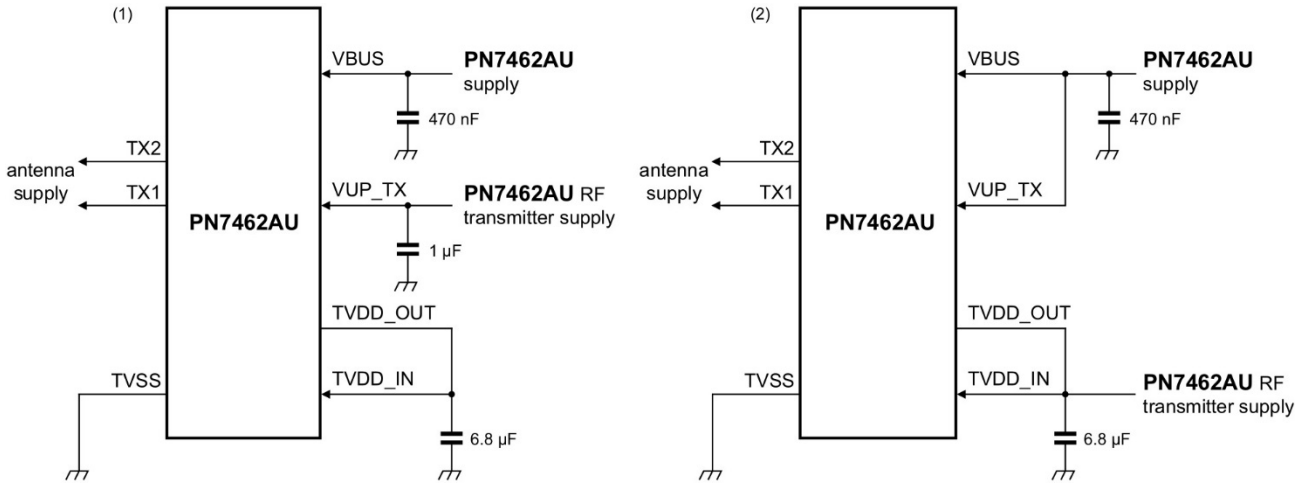



(1) Using the PN7462AU´s TXLDO

(2) Without using PN7462AU´s TXLDO

The capacitance value must be chosen so that the capacitance value is correct at 5 V

**Fig 9. Powering the contactless interface – using a single power supply**




(1) Using the PN7462AU´s TXLDO

(2) Without using PN7462AU´s TXLDO

The capacitance value must be chosen so that the capacitance value is correct at 5 V

**Fig 10. Powering the contactless interface – using an external RF transmitter supply**

### 6.1.3 Contact reader supply

In order to use the contact reader functionality of the PN7462AU following voltages need to be supplied:

- VBUS
- PVDD_IN (if PVDD_OUT is not used needs to be shorted to GND)
- VBUSP
- DVDD pin must be connected to VDD and buffered with 1 µF capacitor to GND

For SPI/ I²C master interface following supply is also needed

- PVDD_M_IN

The contact interface is powered through VBUSP which is connected to VBUS, as shown on the schematic in Fig 11.

The various ISO 7816 contact card classes (A, B, or C) require different voltages:

- VBUSP > 2.7 V: support of class B and class C contact cards
- VBUSP > 3 V: support of class A contact cards

**Remark:** To support Class A cards, DC-to-DC converter has to be used. To support Class B cards with VBUSP < 3.9 V, DC-to-DC converter also has to be used. The Fig 11.shows how to connect the contact interface related pins, when no contact interface is used.

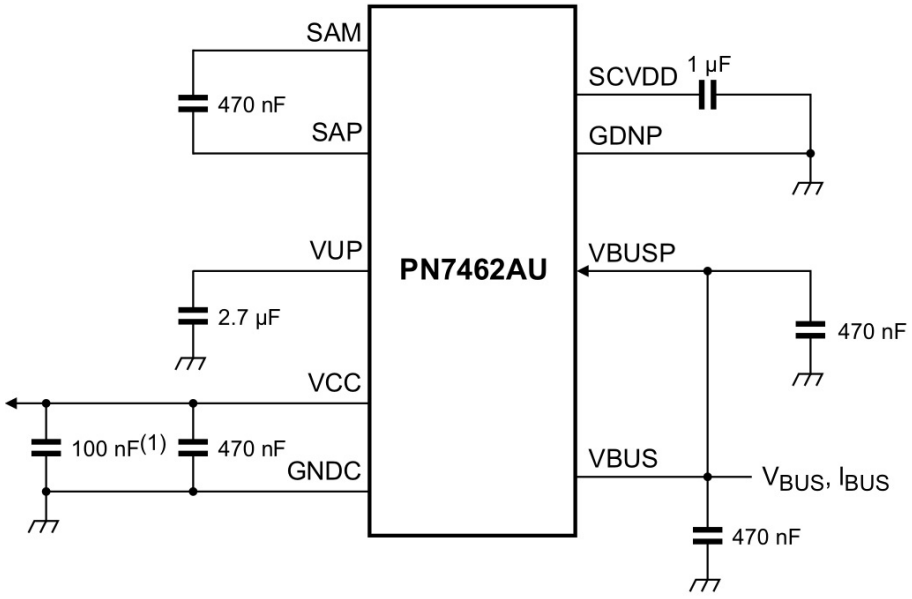



**Fig 11. Powering the contact interface**

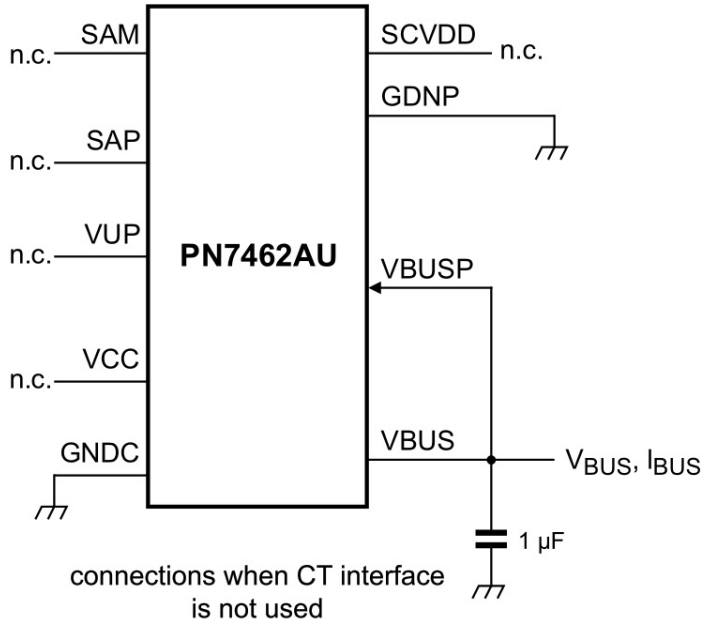



**Fig 12. Contact interface power supply connection when the contact interface is not used**

## 6.2 Power Management Unit

The Integrated Power Management Unit (PMU) supplies internal analog modules, internal digital logic, memories and pads. It also provides regulated voltages for the contactless and the contact interfaces. The PMU automatically adjusts the internal regulators to minimize the power consumption during all possible power modes. The power management unit embeds also mechanisms to prevent the IC from overheat, current overconsumption and overloading the DC-to-DC converter. For the RF transmitter stage a separated low-drop output regulator is embedded. This module also integrates a temperature sensor and Power On Reset generator. The PMU is made of analog modules and digital control unit embedding the registers

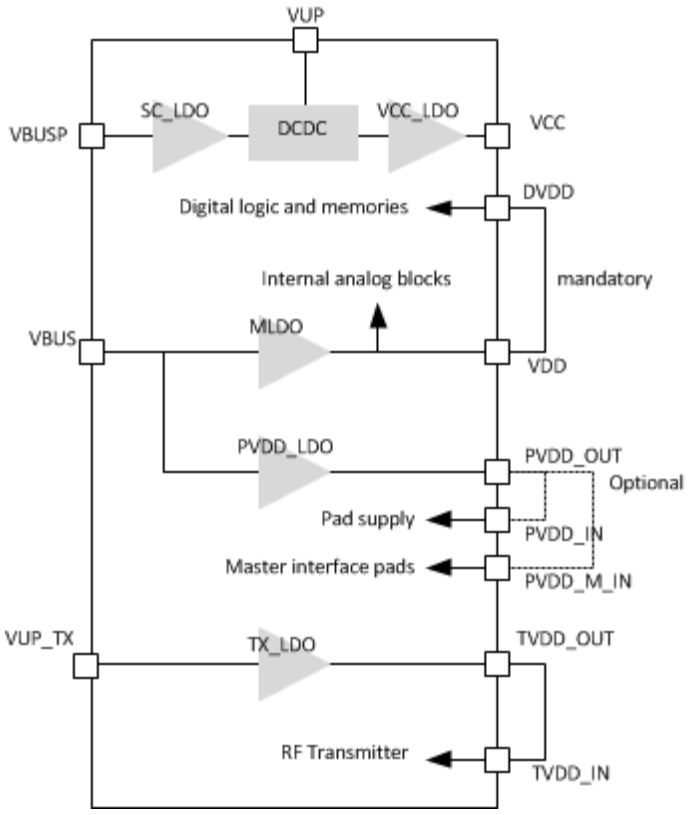



**Fig 13. PMU Module, LDOs and power pins overview**

**Table 33. Voltage and Supply pins connection overview**

| Pin No | Pin Name | IN(I)/Out(O) | Connection | Comment |
|---|---|---|---|---|
| 10 | PVDD_IN | I | externally powered or PVDD_OUT | pad supply for host interfaces (1.8 V or 3.3 V) |
| 11 | DVDD | I | connected to VDD | |

| Pin No | Pin Name | IN(I)/Out(O) | Connection | Comment |
|---|---|---|---|---|
| 30 | VMID | O | blocking capacitor to GND | recommended 100nF |
| 34 | TVDD_IN | I | externally powered or TVDD_OUT | |
| 37 | TVDD_OUT | O | | can be set in PMU_TXLDO_CONTROL_REG |
| 38 | VUP_TX | I | externally powered or VBUS | |
| 39 | VDD | O | | 1.8 V output for DVDD |
| 43 | VBUS | I | Externally powered | |
| 44 | PVDD_OUT | O | | 3.3V output; If connected to GND, PVDD_IN is externally supplied |
| 46 | SAM | I/O | 470 nF to SAP | |
| 47 | SCVDD | O | blocking capacitor to GND | Recommended 1 µF |
| 48 | VBUSP | I | connected to VBUS | Input for contact interface |
| 49 | SAP | I/O | 470 nF to SAM | |
| 50 | VUP | O | blocking capacitor to GND | Recommended 4,7µF |
| 51 | VCC | O | contact card supply | |
| 55 | USB_VBUS | I | | USB detection; If pulled high USB is connected |
| 59 | PVDD_M_IN | I | externally powered or PVDD_OUT | Pad supply for master interfaces (1.8V or 3.3V) |

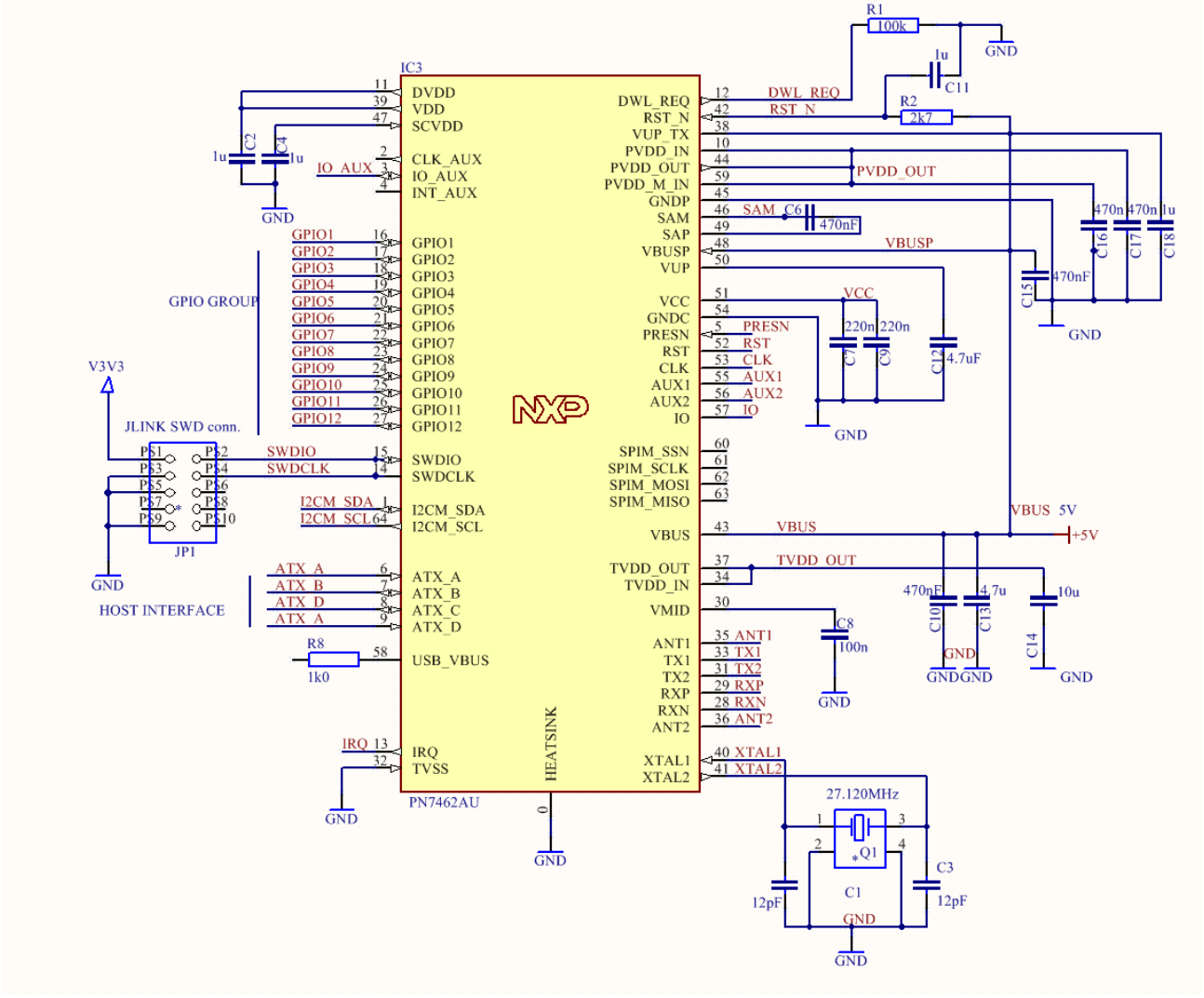



**Fig 14. PN7462AU basic schematic**

### 6.2.1 Low Drop-Out regulators

The PMU embeds several Low Drop-Out regulators (LDO) in order to ensure the stability of the power supply.

#### 6.2.1.1 Main LDO

The Main LDO (MLDO) provides 1.8 V for all internal analog, digital and memory modules. It draws its power from VBUS. It includes a current limiter to prevent damage of output transistors. The output supply is available on the VDD pin, which must be connected externally to the DVDD pin.

#### 6.2.1.2 PVDD_LDO

The PVDD_LDO provides 3.3 V for all digital pads. It is supplied by VBUS, and requires a minimum voltage of 4 V to be functional. It delivers a maximum current of 30 mA. The output of the PVDD_LDO is PVDD_OUT pin. This LDO is used to provide the necessary supply to PVDD_IN (pad supply for host interface) and PVDD_M_IN (pad supply for master interfaces). When an external power supply is used, the PVDD_OUT must be connected to the ground. The ROM boot detects automatically that the LDO output is connected to the ground, and switches it off. The PVDD LDO has a low power mode, which is used automatically by the PN7462AU when the chip is in Stand-by mode or Suspend mode. This enables to supply host pads and GPIOs, and to detect wake-up signals coming from these interfaces.

#### 6.2.1.3 TXLDO Transmitter supply

The PN7462AU integrates an internal transmitter LDO. The TXLDO can be used to maintain a constant output voltage for the RF interface. The TXLDO is designed to protect the chip from voltage ripple introduced by the power supply on the VUP_TX pin. It is powered through VUP_TX pin. The programmable output voltages are: 3.0 V, 3.3 V, 3.6 V, 4.5 V, and 4.75 V. For a given output voltage, VUP_TX shall always been higher of 0.3 V (i.e. to supply a 3 V output, the minimum voltage to be applied on VUP_TX is 3.3 V). If the voltage is no sufficient, then TVDD_OUT follows VUP_TX, lowered of 0.3 V. When it is not used, TVDD_OUT shall be connected to TVDD_IN, and TX_LDO shall be turned off. The TXLDO can be used in one of the following power modes

- Full power mode
- Low power mode
- Low power 2 mode
- Shutdown mode
- Standby mode

For corresponding register settings, please refer to Section 6.7.2.

#### 6.2.1.4 VCC LDO

The VCC LDO provides contact interface supply VCC.

#### 6.2.1.5 SCLDO

The SCLDO provides a regulated voltage to the DC-to-DC converter, to enable class B operation when 2.7 < VBUS < 3.9 V and class A operation.

#### 6.2.1.6 DC-to-DC converter

The PN7462AU includes a DC-to-DC converter, in order to support Class A and Class B cards, when the input voltage VBUSP is not sufficient. The DC-to-DC converter is a capacitance voltage doubler. It takes its power from the SCLDO. The DC-to-DC converter can be bypassed. Its output (VUP) is regulated from 3.3 to 5.5 V.

#### 6.2.1.7 Start-up times of LDOs

Table 34. Start-up times of LDOs

| LDO | Max Startup Time |
|---|---|
| PVDDLDO | 1.5 ms |
| TXLDO | 200 µs |
| VCC LDO | 22 ms (including DC-to-DC converter, SCLDO and VCCLDO startup times)<br>500 µs (excluding DC-to-DC and SCLDO) |

## 6.3 PN7462AU PMU digital control unit

The PMU digital control unit of the PN7462AU is used in the system as the gateway to configure all modes of supply for the product using the control registers. Note that additional registers related to PMU control are located in the power clock and reset (PCR) Unit for they need to be always powered up. The PMU digital control unit consists of the AMBA 3.0 APB interface and the associated register bank to drive the analog part of PMU, plus additional glue logic related to controlling the temperature, overcurrent, pad voltage, interrupts and calibration of temperature sensors.

Main blocks (see Fig 15) are:

- Register bank for PMU analog block (detailed description in Section 6.7)
- Temperature sensor controller
- 32-bit APB slave interface

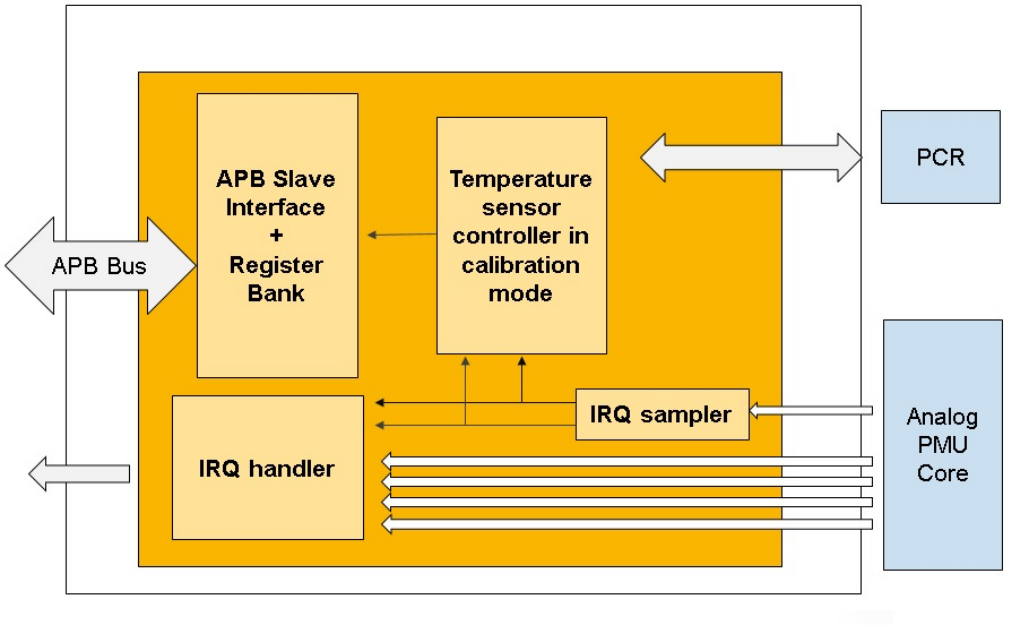



Fig 15. Block diagram of PMU

## 6.4 Interrupts

Interrupt signals are generated in the analog part of the PMU as a result of:

- TXLDO 5 V monitoring
- VCC current limiter
- DC-to-DC converter current overload
- SCVDD current overload
- TXLDO current overload
- Temperature sensor

All interrupts are “ORed” in the digital part of the PMU to output one unique PMU interrupt line. The software has to analyze the content of the PMU_INTERRUPT_STATUS_REG register to know which of the seven conditions caused the interrupt. All interrupts can be enabled or disabled (masked) separately using the PMU_INTERRUPT_ENABLE_REG register. Clearing the status bit field of one interrupt is performed by setting the corresponding bit field in the PMU_INTERRUPT_CLR_STATUS_REG register high. All bit fields of all PMU_INTERRUPT_CLR_STATUS_REG and PMU_INTERRUPT_SET_STATUS_REG registers are automatically cleared if set high after two system clock cycles.

## 6.5 Temperature sensors

The Power Management Unit of PN7462AU comprises two temperature sensors associated with the contactless TXLDO and the contact DC-to-DC converter interfaces. The main purpose of these sensors is to monitor the temperature and prevent the overheating, which could potentially cause the damage of the chip and the customer device. The triggering levels are configurable. Following temperatures can be chosen: 135°C, 130°C, 125°C or 120°C. By default, the temperature sensor is set to 120°C. When one of the sensors detects a temperature issue, an interrupt is generated and the PN7462AU will be put by software into the standby mode or the suspend mode if the USB interface is used.  The PN7462 registers indicate which of the two temperature sensors (contact interface, or contactless interface) generated the interrupt. When the temperature goes below the configured threshold temperature, the PN7462AU wakes up automatically. For a detailed description of the corresponding registers refer to PCR_TEMP_REG.

## 6.6 Voltage monitoring

The voltage monitoring is used to check if the voltages are within the appropriate range specified for a proper operation of the IC. The following power supplies are monitored: VBUS (2 voltage monitors) and VBUSP (1 voltage monitor). Table 35 summarizes the voltage monitors with the selectable thresholds values.

**Table 35. Voltage monitor - possible threshold configuration**

| Voltage Monitor | Threshold 1 | Threshold 2 | Threshold 3 |
|---|---|---|---|
| VBUS1 | 2.3 V | 2.7 V | N/A |

| Voltage Monitor | Threshold 1 | Threshold 2 | Threshold 3 |
|---|---|---|---|
| VBUS2 | 2.7 V | 4 V | N/A |
| VBUSP | 2.7 V | 3 V | 3.9 V |

### 6.6.1 VBUS monitor

The PN7462AU offers two selectable thresholds (2.3 V or 2.7 V) for monitoring the voltage on the VBUS pin. When VBUS voltage falls below the selected threshold and Auto Hard Power Down (HPD) feature is enabled in the Power, Clock and Reset unit (described in Section 8), the IC will enter the HPD mode. Alternatively, the software can monitor the signal by reading a dedicated status register and decide to put the IC into the HPD mode. The signal can be enabled for interrupt in Interrupt Enable register in the PCR to cause a CPU interrupt. By default the VBUS monitor is disabled during the power-up.

### 6.6.2 PVDD LDO (VBUS2) monitor

The PN7462AU offers two selectable thresholds (VBUS2: 2.7 V or 4.0 V) for monitoring the voltage of the PVDD LDO supply. The status of the VBUS2 monitor is available in the status register. The software has to check whether the voltage is sufficient before enabling the LDO. The PVDD LDO can be enabled when the input supply VBUS2 > 4.0 V.

### 6.6.3 VBUSP monitor

VBUSP monitor is used for the Contact interface supply. Three levels (2.7, 3.0, and 3.9 V) can be selected for monitoring the voltage on the VBUSP pin. The threshold is configured by firmware depending on the card type selected. (Class A, Class B, Class C)

When VBUSP < 2.7 V, no functionality is possible.

When VBUSP > 2.7 V, Class C type can be supported.

When VBUSP > 3.0 V, Class A type can be supported with DC-to-DC converter configured in the doubler mode.

When VBUSP > 2.7 V and VBUSP < 3.9 V, Class B type is supported with DC-to-DC converter configured in the doubler mode.

When VBUSP > 3.9 V, Class B type of card is supported with DC-to-DC converter configured in the follower mode.

When the voltage falls below the selected threshold value and CT automatic deactivation is enabled in the PCR System Register, the hardware automatically de-activates the CT interface. The signal can be enabled for interrupt in Interrupt Enable register in the PCR to cause a CPU interrupt. The software must check VBUSP monitor levels by reading dedicated status registers before starting card activation sequence.

### 6.6.4 Latency of voltage monitors

**Table 36. Latency of voltage monitors**

| Voltage Monitor | Latency |
|---|---|
| VBUS1 | 10 µs |
| VBUS2 | 10 µs |
| VBUSP | 75 µs |

## 6.7 Register overview and description

**Table 37. PMU register overview (base address 0x4000 8000)**

| Name | Address offset | Access | Reset value | Description |
|---|---|---|---|---|
| PMU_STATUS_REG | 0000h | R | 0000_0000h | Global PMU status register. To be used for observing signals in system mode. |
| PMU_ BG_MON_CONTROL_REG | 0004h | R/W | 0000_040Eh | Used to enable comparators and set thresholds for the monitors |
| PMU_TXLDO_CONTROL_REG | 0008h | R/W | 0400_0000h | TXLDO control register |
| PMU_LDO_CONTR OL_REG | 000Ch | R/W | 0000_0000h | DC-to-DC converter control register |
| INTERNAL_USE[1] | 00010h | R/W | 0000_0000h | For internal use |
| INTERNAL_USE[1] | 0014h | R/W | 0000_0000h | For internal use |
| PMU_INTERRUPT_CLR_ENABLE_REG | 3FD8h | W | 0000_0000h | PMU interrupt clear enable register with automatic clear if set by software. |
| PMU_INTERRUPT_SET_ENABLE_REG | 3FDCh | W | 0000_0000h | PMU interrupt set enable register with automatic clear if set by software |
| PMU_INTERRUPT_STATUS_REG | 3FE0h | R | 0000_0000h | PMU interrupt status register |
| PMU_INTERRUPT_ENABLE_REG | 3FE4h | R/W | 0000_0000h | PMU interrupt enable register |
| PMU_INTERRUPT_CLR_STATUS_REG | 3FE8h | W | 0000_0000h | PMU interrupt clear status register with automatic clear if set by software |
| PMU_INTERRUPT_SET_STATUS_REG | 3FECh | W | 0000_0000h | PMU interrupt set status register with automatic clear if set by software |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

### 6.7.1 Detailed register description

#### 6.7.1.1 PMU_STATUS_REG

This register is used to observe signals in system mode.

**Table 38. PMU_STATUS_REG (address offset 0x0000)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | R- | 0x00 | Reserved |
| 24 | MLDO_LOWPOWER_VBATBUF | R- | 0x00 | 1: Indicates that the VBAT LDO is in low power |
| 23 | INTERNAL_USE[1] | R- | 0x00 | For internal use |
| 22 | TXLDO_5VMON_OK | R- | 0x00 | TXLDO 5V monitor output<br>1: TXLDO 5V monitor output is set TVDD/VUP_TX value is greater than 5V. Source is selected in PMU_TXLDO register |
| 21 | POK_VBUSP | R- | 0x00 | Output of VBUSP monitor<br>1: VBUSP value is greater than threshold |
| 20 | POK_VBUSMON2 | R- | 0x00 | Output of VBUS monitor2<br>1: VBUS2 value is greater than threshold |
| 19 | POK_PVDDOUT | R- | 0x00 | Output of PVDD monitor<br>1: PVDDOUT is ok (above 3.3 V) |
| 18 | RESERVED | R- | 0x00 | Reserved |
| 17 | RESERVED | R- | 0x00 | Reserved |
| 16:12 | RESERVED | R- | 0x00 | Reserved |
| 11 | SCVDD_OVERLOAD | R- | 0x00 | 1: SCLDO overload error<br>0: No SCLDO overload error |
| 10 | DCDC_OVERLOAD | R- | 0x00 | 1: DC-to-DC converter overload error<br>0: No DC-to-DC converter overload error |
| 9:3 | TXLDO_DET_OUT | R- | 0x00 | [6:0] - Output of State Machine detection |
| 2 | TXLDO_DET_SM_OK | R- | 0x00 | Signal that indicates to Digital & SW that State Machine work is done for automatic current measurement (automatic mode) and current value is valid in TXLDO_DET_OUT |
| 1 | TXLDO_TVDD_OK | R- | 0x00 | Signal that indicates State Machine work is done |
| 0 | TXLDO_DET | R- | 0x00 | Indicates that the detection threshold for current measurement is reached |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

#### 6.7.1.2 PMU_BG_MON_CONTROL_REG

This register is used to enable comparators and set thresholds for the monitors.

**Table 39. PMU_BG_MON_CONTROL_REG (address offset 0x0004)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:11 | RESERVED | R/W | 0x00 | Reserved |
| 10 | ENABLE_PVDD_M_1V8_COMP | R/W | 0x01 | Activates 1.8 V comparator for PVDD_M<br>1: Enable 1.8 V comparator for PVDD_M<br>0: Disable 1.8 V comparator for PVDD_M |
| 9 | VBUSPMON_THRESHOLD | R/W | 0x00 | Sets threshold for VBUSPMON<br>00: 2.7 V<br>01: 3 V<br>10: 3.9 V |
| 7 | VBUSMON2_THRESHOLD | R/W | 0x00 | Sets threshold for VBUS MON2<br>0: 2.5 V<br>1: 4 V |
| 6 | ENABLE_VBUSPMON | R/W | 0x00 | Activates VBUSP monitor<br>1: Enable VBUSP monitor<br>0: Disable VBUSP monitor |
| 5 | ENABLE_VBUSMON2 | R/W | 0x00 | Activates VBUS monitor 2<br>1: Enable VBUS monitor 2<br>0: Disable VBUS monitor 2 |
| 4 | ENABLE_PVDDOUTMON | R/W | 0x00 | Activates PVDDOUT monitor<br>1: Enable PVDDOUT monitor<br>0: Disable PVDDOUT monitor |
| 3 | ENABLE_PVDD_M_3V_COMP | rw | 0x01 | Activates 3 V comparator for PVDD_M<br>1: Enable 3 V comparator for PVDD_M<br>0: Disable 3 V comparator for PVDD_M |
| 2 | ENABLE_PVDD_3V_COMP | rw | 0x01 | Activates 3 V comparator for PVDD<br>1: Enable 3 V comparator for PVDD<br>0: Disable 3 V comparator for PVDD |
| 1 | RESERVED | R/W | 0x01 | Reserved |
| 0 | RESERVED | R/W | 0x00 | reserved |

#### 6.7.1.3 PMU_TXLDO_CONTROL_REG

This register is used to control the TXLDO.

**Table 40. PMU_TXLDO_CONTROL_REG (address offset 0x0008)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R/W | 0x00 | Reserved |
| 28 | TXLDO_ENABLE_LP2 | R/W | 0x00 | 1: Enable 2nd regulator |

<table>
<tr><th>Bit</th><th>Symbol</th><th>Access</th><th>Value</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td></td><td>0: Disable 2nd regulator</td></tr>
<tr><td>27:25</td><td>RESERVED</td><td>R/W</td><td>0x02</td><td>Do not modify value</td></tr>
<tr><td>24</td><td>TXLDO_RSTN_SOURCE_SEL</td><td>R/W</td><td>0x00</td><td>Source selection bit of TXLDO_digi resetn.<br>0: resetn source = rst_pcr_system_n<br>1: resetn source = (~rst_pcr_system_n or TXLDO_enable)</td></tr>
<tr><td>23</td><td>TXLDO_OVERCURRENT_EN</td><td>R/W</td><td>0x00</td><td>1: Enable overcurrent detection<br>0: Disable overcurrent detection</td></tr>
<tr><td>22:21</td><td>TXLDO_SELECT_ANT</td><td>R/W</td><td>0x00</td><td>Offset selection for detection range of the TXLDO_DET_OUT<br>0 20mA<br>1 50mA<br>2 70mA<br>3 100mA<br>Measurement range of the 7 bit DAC with corresponding offset<br>20 to 70mA, 50 to 100mA, 70 to 120mA & 100 to 150mA</td></tr>
<tr><td>20</td><td>RESERVED</td><td>R/W</td><td>0x00</td><td>Always set to 0</td></tr>
<tr><td>19:15</td><td>RESERVED</td><td>R/W</td><td>0x00</td><td>Do not modify value</td></tr>
<tr><td>14:8</td><td>TXLDO_DET_IN</td><td>R/W</td><td>0x00</td><td>Selection of Current Detection value for manual current measurement. if value is reached TXLDO_DET is set to 1<br><table><tr><th>TXLDO_det_in[6:0]</th><th>Cur (mA)</th></tr><tr><td>0000000</td><td>0</td></tr><tr><td>0000001</td><td>0.39</td></tr><tr><td>0000010</td><td>0.78</td></tr><tr><td>0000100</td><td>1.56</td></tr><tr><td>0001000</td><td>3.12</td></tr><tr><td>0010000</td><td>6.24</td></tr><tr><td>0100000</td><td>12.48</td></tr><tr><td>1000000</td><td>24.96</td></tr><tr><td>1111111</td><td>50</td></tr></table>Minimum offset in TXLDO_SELECT_ANT is 20 mA (so 0 means minimum is 20 mA for the current threshold)</td></tr>
<tr><td>7:5</td><td>TXLDO_SELECT</td><td>R/W</td><td>0x00</td><td>Selection of Tvdd supply<br>0 = 3 V<br>1 = 3.3 V</td></tr>
</table>

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 2 = 3.6 V<br>3 = 4.5 V<br>4 and others=4.7 V |
| 4 | TXLDO_LIMITER_EN | R/W | 0x00 | Activates current limiter<br>1: Enable current limiter<br>0: Disable current limiter |
| 3 | TXLDO_LOW_POWER_EN | R/W | 0x00 | Activates the low-power mode<br>1: Enable low power mode<br>0: Disable low power mode |
| 2 | TXLDO_EN_DAC_SM | R/W | 0x00 | Enables the current measurement using the state machine (automatic mode) |
| 1 | TXLDO_DETECTOR_EN | R/W | 0x00 | Activates the Current Detection for DAC current measurement (manual mode)<br>1: Enable current detection for TXLDO<br>0: Disable current detection for TXLDO |
| 0 | TXLDO_ENABLE | R/W | 0x00 | Enable the whole TXLDO block<br>1: Enable TXLDO<br>0: Disable TXLDO |

#### 6.7.1.4 PMU_LDO_CONTROL_REG

This register is used to control the DC-to-DC converter.

**Table 41. PMU_LDO_CONTROL_REG (address offset 0x000C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R/W | 0x00 | Reserved |
| 11:7 | RESERVED | R/W | 0x00 | Reserved |
| 6 | RESERVED | R/W | 0x00 | Reserved |
| 5 | VBUSP_HI | R/W | 0x00 | CT channel detects the presence and starts the CT sequence.<br>1: VBUSP > 3.9 V.<br>0: VBUSP < = 3.9 V. |
| 4 | RESERVED | R/W | 0x00 | Reserved |
| 3 | DISABLE_VCC_IPROT | R/W | 0x00 | If set to 1, the VCC LDO current limit is disabled |
| 2 | CT_VCC_IPROT | R- | 0x00 | If set to 1 the VCC LDO current limit has been triggered |
| 1 | RESERVED | R/W | 0x00 | Reserved |

#### 6.7.1.5 PMU_INTERRUPT_CLR_ENABLE_REG

This register is a collection of Clear Interrupt Enable commands with automatic clear if set by software.

**Table 42. PMU_INTERRUPT_CLR_ENABLE_REG (address offset 0x3FD8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | -X | 0x00 | Reserved |
| 8 | VBUSMON2_LOW_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of Vbus monitor 2 going low interrupt |
| 7 | TXLDO_5V_MON_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of TXLDO 5 V monitor interrupt |
| 6 | VCC_ILIM_ACT_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of VCC current limiter. Automatically cleared after 2 cycles if set by software |
| 5 | RESERVED | -X | 0x00 | Reserved |
| 4 | PVDD_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of PVDD interrupt. Automatically cleared after 2 cycles if set by software |
| 3 | DCDC_OVERLOAD_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of DC-to-DC converter overload. Automatically cleared after 2 cycles if set by software |
| 2 | SCVDD_OVERLOAD_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of SCVDD overload. Automatically cleared after 2 cycles if set by software |
| 1 | TXLDO_OVERCURRENT_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of TXLDO overcurrent interrupt. Automatically cleared after 2 cycles if set by software |
| 0 | TEMPSENS_ERROR_IRQ_CLEAR_ENABLE | -X | 0x00 | clears enable state of Temperature sensor calibration interrupt. Automatically cleared after 2 cycles if set by software |

#### 6.7.1.6 PMU_INTERRUPT_SET_ENABLE_REG

This register is a collection of Set Interrupt Enable commands with automatic clear if set by software.

**Table 43. PMU_INTERRUPT_SET_ENABLE_REG (address offset 0x3FDC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | -X | 0x00 | Reserved |
| 8 | VBUSMON2_LOW_IRQ_SET_ENABLE | -X | 0x00 | enables Vbus monitor 2 going low interrupt |
| 7 | TXLDO_5V_MON_IRQ_SET_ENABLE | -X | 0x00 | Enables TXLDO 5 V monitor interrupt |
| 6 | VCC_ILIM_ACT_IRQ_SET_ENABLE | -X | 0x00 | Enables VCC current limiter. Automatically cleared after 2 cycles if set by software |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 5 | RESERVED | -X | 0x00 | Reserved |
| 4 | PVDD_IRQ_SET_ENABLE | -X | 0x00 | Enables PVDD interrupt. Automatically cleared after 2 cycles if set by software |
| 3 | DCDC_OVERLOAD_IRQ_SET_ENABLE | -X | 0x00 | Enables DC-to-DC converter overload. Automatically cleared after 2 cycles if set by software |
| 2 | SCVDD_OVERLOAD_IRQ_SET_ENABLE | -X | 0x00 | Enables SCVDD overload. Automatically cleared after 2 cycles if set by software |
| 1 | TXLDO_OVERCURRENT_IRQ_SET_ENABLE | -X | 0x00 | Enables TXLDO overcurrent interrupt. Automatically cleared after 2 cycles if set by software |
| 0 | TEMPSENS_ERROR_IRQ_SET_ENABLE | -X | 0x00 | Enables Temperature sensor calibration interrupt. Automatically cleared after 2 cycles if set by software |

#### 6.7.1.7 PMU_INTERRUPT_STATUS_REG

This register is a collection of Interrupt Status commands.

**Table 44. PMU_INTERRUPT_STATUS_REG (address offset 0x3FE0)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | R- | 0x00 | Reserved |
| 8 | VBUSMON2_LOW_IRQ_STATUS | R- | 0x00 | Indicates Vbus monitor 2 going low interrupt is set |
| 7 | TXLDO_5V_MON_IRQ_STATUS | R- | 0x00 | Indicates TXLDO 5 V monitor interrupt is set. Automatically cleared after 2 cycles if set by software |
| 6 | VCC_ILIM_ACT_IRQ_STATUS | R- | 0x00 | Indicates VCC current limiter active interrupt is set. Automatically cleared after 2 cycles if by software |
| 5 | RESERVED | R- | 0x00 | Reserved |
| 4 | PVDD_IRQ_STATUS | R- | 0x00 | Indicates PVDD interrupt is set. Automatically cleared after 2 cycles if by software |
| 3 | DCDC_OVERLOAD_IRQ_STATUS | R- | 0x00 | Indicates DC-to-DC converter overload interrupt is set. Automatically cleared after 2 cycles if by software |
| 2 | SCVDD_OVERLOAD_IRQ_STATUS | R- | 0x00 | Indicates SCVDD overload interrupt is set. Automatically cleared after 2 cycles if by software |
| 1 | TXLDO_OVERCURRENT_IRQ_STATUS | R- | 0x00 | Indicates TXLDO overcurrent interrupt is set. Automatically cleared after 2 cycles if by software |
| 0 | TEMPSENS_ERROR_IRQ_STATUS | R- | 0x00 | Indicates temperature sensor interrupt is set. Automatically cleared after 2 cycles if by software |

#### 6.7.1.8 PMU_INTERRUPT_ENABLE_REG

This register is a collection of Interrupt Enable commands.

**Table 45. PMU_INTERRUPT_ENABLE_REG (address offset 0x3FE4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | R- | 0x00 | Reserved |
| 8 | VBUSMON2_LOW_IRQ_ENABLE | R- | 0x00 | Indicates enabled VBUS monitor 2 going low interrupt |
| 7 | TXLDO_5V_MON_IRQ_ENABLE | R- | 0x00 | Indicates enabled TXLDO 5 V monitor interrupt. Automatically cleared after 2 cycles if set by software |
| 6 | VCC_ILIM_ACT_IRQ_ENABLE | R- | 0x00 | Indicates enabled VCC current limiter active interrupt. Automatically cleared after 2 cycles if by software |
| 5 | RESERVED | R- | 0x00 | Reserved |
| 4 | PVDD_IRQ_ENABLE | R- | 0x00 | Indicates enabled PVDD interrupt. Automatically cleared after 2 cycles if by software |
| 3 | DCDC_OVERLOAD_IRQ_ENABLE | R- | 0x00 | Indicates enabled DC-to-DC converter overload interrupt. Automatically cleared after 2 cycles if by software |
| 2 | SCVDD_OVERLOAD_IRQ_ENABLE | R- | 0x00 | Indicates enabled SCVD overload interrupt. Automatically cleared after 2 cycles if by software |
| 1 | TXLDO_OVERCURRENT_IRQ_ENABLE | R- | 0x00 | Indicates enabled TXLDO overcurrent interrupt. Automatically cleared after 2 cycles if by software |
| 0 | TEMPSENS_ERROR_IRQ_ENABLE | R- | 0x00 | Indicates enabled temperature sensor interrupt. Automatically cleared after 2 cycles if by software |

#### 6.7.1.9 PMU_INTERRUPT_CLR_STATUS_REG

This register is a collection of Clear Interrupt Status commands with automatic clear if set by software.

**Table 46. PMU_INTERRUPT_CLR_STATUS_REG (address offset 0x3FE8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | -X | 0x00 | Reserved |
| 8 | VBUSMON2_LOW_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of Vbus monitor 2 going low interrupt |
| 7 | TXLDO_5V_MON_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of TXLDO 5 V monitor interrupt. Automatically cleared after 2 cycles if set by software |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 6 | VCC_ILIM_ACT_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of VCC current limiter active interrupt. Automatically cleared after 2 cycles if clear by software |
| 5 | RESERVED | -X | 0x00 | Reserved |
| 4 | PVDD_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of PVDD interrupt. Automatically cleared after 2 cycles if clear by software |
| 3 | DCDC_OVERLOAD_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of DC-to-DC converter overload interrupt. Automatically cleared after 2 cycles if clear by software |
| 2 | SCVDD_OVERLOAD_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of SCVD overload interrupt. Automatically cleared after 2 cycles if clear by software |
| 1 | TXLDO_OVERCURRENT_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of TXLDO overcurrent interrupt. Automatically cleared after 2 cycles if set by software |
| 0 | TEMPSENS_ERROR_IRQ_CLEAR_STATUS | -X | 0x00 | Clears status of temperature sensor interrupt. Automatically cleared after 2 cycles if set by software |

#### 6.7.1.10 PMU_INTERRUPT_SET_STATUS_REG

This register is a collection of Set Interrupt Status commands with automatic clear if set by software.

**Table 47. PMU_INTERRUPT_SET_STATUS_REG (address offset 0x3FEC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | -X | 0x00 | Reserved |
| 8 | VBUSMON2_LOW_IRQ_SET_STATUS | -X | 0x00 | Sets status of Vbus monitor 2 going low interrupt |
| 7 | TXLDO_5V_MON_IRQ_SET_STATUS | -X | 0x00 | Sets status of TXLDO 5 V monitor interrupt. Automatically cleared after 2 cycles if set by software |
| 6 | VCC_ILIM_ACT_IRQ_SET_STATUS | -X | 0x00 | Sets status of VCC current limiter active interrupt. Automatically cleared after 2 cycles if clear by software |
| 5 | RESERVED | -X | 0x00 | Reserved |
| 4 | PVDD_IRQ_SET_STATUS | -X | 0x00 | Sets status of PVDD interrupt. Automatically cleared after 2 cycles if clear by software |
| 3 | DCDC_OVERLOAD_IRQ_SET_STATUS | -X | 0x00 | Sets status of DC-to-DC converter overload interrupt. Automatically cleared after 2 cycles if clear by software |
| 2 | SCVDD_OVERLOAD_IRQ_SET_STATUS | -X | 0x00 | Sets status of SCVD overload interrupt. Automatically cleared after 2 cycles if clear by software |
| 1 | TXLDO_OVERCURRENT_IRQ_SET_STATUS | -X | 0x00 | Sets status of TXLDO overcurrent interrupt. Automatically cleared after 2 cycles if set by software |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 0 | TEMPSENS_ERROR_IRQ_SET_STATUS | -X | 0x00 | Sets status of temperature sensor interrupt. Automatically cleared after 2 cycles if set by software |

### 6.7.2 TXLDO Register settings

**Table 48. TXLDO Register**

| Mode | Register | Bit Field |
|---|---|---|
| full power | PMU_TXLDO_CONTROL_REG | TXLDO_ENABLE =1<br>TXLDO_LOW_POWER_EN =0<br>TXLDO_ENABLE_LP2=0 |
| low Power | PMU_TXLDO_CONTROL_REG | TXLDO_ENABLE =0<br>TXLDO_LOW_POWER_EN =1<br>TXLDO_ENABLE_LP2=0 |
| low Power2 | PMU_TXLDO_CONTROL_REG | TXLDO_ENABLE =0<br>TXLDO_LOW_POWER_EN =0<br>TXLDO_ENABLE_LP2=1 |
| shutdown | PMU_TXLDO_CONTROL_REG | TXLDO_ENABLE =0<br>TXLDO_LOW_POWER_EN =0<br>TXLDO_ENABLE_LP2=0 |
| standby | PCR_PMU_REG | TXLDO_ENABLE_STANDBY |

# 7. Clock generator

The PN7462AU uses following clock sources:

- External 27.12 MHz crystal oscillator
- Internal 20 MHz HFO (High Frequency Oscillator)
- Internal 380 kHz LFO (Low Frequency Oscillator)
- Internal 48 MHz USB PLL (Phase Lock Loop)
- Internal CLIF PLL (Phase Lock Loop)

In addition to the clock sources, the clock generator comprises a digital control unit, which controls and monitors the signals coming from the clocks and integrated PLL. The registers are accessed using an APB register interface.

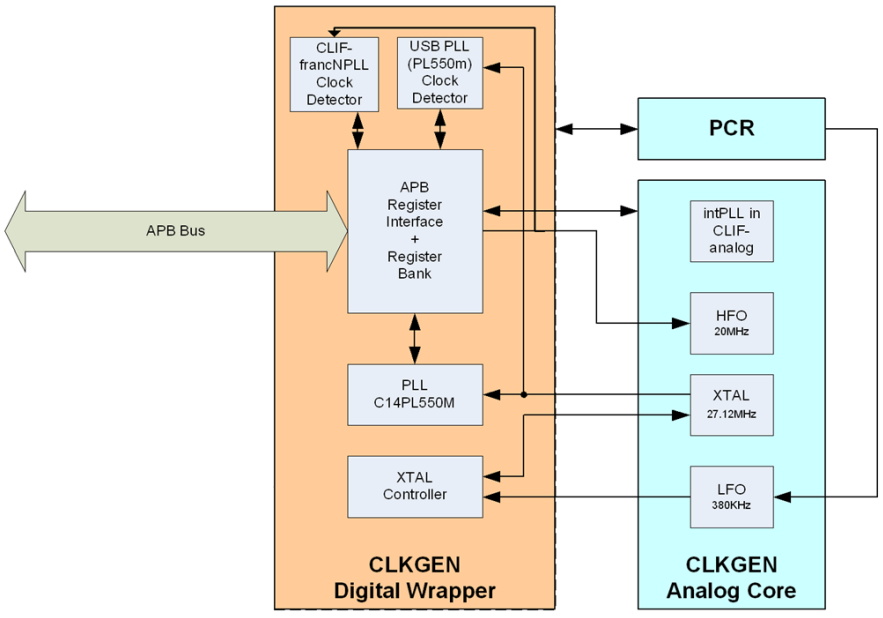



**Fig 16. Clock generator block diagram**

## 7.1 Oscillators

The PN7462AU includes three independent oscillators. Each oscillator can be used for more than one purpose. Upon reset, the PN7462AU will operate from the Internal HFO until it is switched to a different clock source. This allows systems to operate without any external crystal and the boot loader code to operate at a known frequency.

### 7.1.1 27.12 MHz Crystal oscillator (XTAL)

The 27.12 MHz crystal oscillator is used as a reference for all operations requiring high stability of the clock frequency. This includes: contactless interface, contact interface, SPI and I2C master interfaces, HSUART and USB PLL for the USB interface.

To ensure the stability of the clock frequency, it is recommended to adopt the circuit with the external quartz and the trimming capacitors shown in Fig 17. Table 49 summarizes the requirements for the crystals.

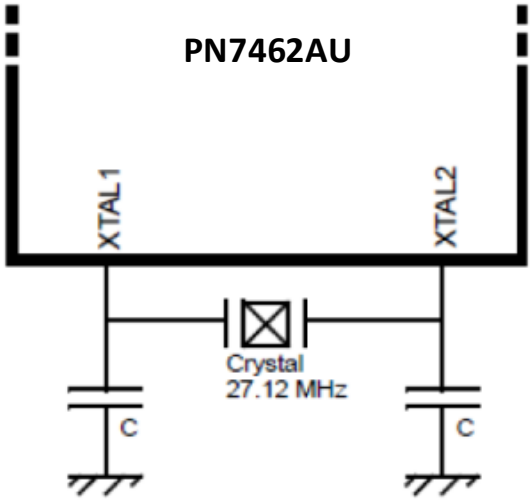



**Fig 17. 27.12 MHz crystal oscillator connection**

**Table 49. Crystal requirements**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $f_{CLK}$ | CLK frequency | ISO/ IEC and FCC compliancy | | 27.12 | | MHz |
| $f_{CLK\_ACC}$ | CLK frequency accuracy | [1] | -50 | | +50 | Ppm |
| ESR | Equivalent series resistance | | | 50 | 100 | Ohm |
| $C_{LOAD}$ | Load capacitance | | | 10 | | pF |
| $P_{CLK}$ | Drive level | | | | 100 | µW |

[1] This requirement is according FCC regulations requirements. To meet only ISO/IEC 14443 and ISO/IEC 18092 then +/- 14 kHz apply.

#### 7.1.1.1 XTAL activation sequence

The XTAL is automatically activated by the digital control unit as soon as the system reset is released. The LFO needs to be activated to start the XTAL Oscillator.

XTAL activation sequence description:

- Wait for system reset release
- Wait for 4 clk_lfo clock cycles (4x(1/380 KHz) ~ 10.53 us)
- Set XTAL_ENABLE_KICK to ’1’
- Wait for 4 clk_lfo clock cycles (4x(1/380 KHz) ~ 10.53 us)
- Set XTAL_ENABLE_KICK to ’1’

In case of error, the PCR can try to restart the XTAL activation sequence by activating the system reset again or switching to another system clock source.

The XTAL can also be controlled by software when CLKGEN_HFO_XTAL_REG.XTAL_CONTROL_SW is set to ’1’. Before activating the XTAL software control, the system clock should be switched from the XTAL to another clock. The following sequence must be followed step by step in order to activate XTAL by SW.

1. Set CLKGEN_HFO_XTAL_REG.XTAL_CONTROL_SW is set to ’1’.
2. Clear CLK_HFO_XTAL_REG.XTAL_SPARE0 to ‘0’
3. Wait for 4 clk_lfo clock cycles (4x(1/380 KHz) ~ 10.53 us)
4. Clear CLKGEN_HFO_XTAL_REG.XTALl_BYPASS to ‘0’
5. Clear CLKGEN_HFO_XTAL_REG.XTAL_SELECT_EXTERNAL_CLOCK to ‘0’
6. Set XTAL_ENABLE_KICK to ’1’
7. Wait for 4 clk_lfo clock cycles (4x(1/380 KHz) ~ 10.53 us)
8. Set XTAL_ENABLE_KICK to ’1’

### 7.1.2 High Frequency Oscillator (HFO)

The PN7462AU integrates an internal low power High Frequency Oscillator (HFO), generating a 20 MHz clock without using the PLL. The HFO can be used as a system clock. The HFO is activated by default with the CLKGEN_HFO_XTAL_REG.HFO_ENABLE register bit as soon as the system reset is released.

### 7.1.3 Low Frequency Oscillator (LFO)

The PN7462AU integrates an internal low power Low Frequency Oscillator working at 380 kHz. The LFO is used by the EEPROM, POR sequencer, Contactless interface, timers and watchdog. The LFO needs to be activated to start the XTAL Oscillator

## 7.2 Phase Locked Loop (USB PLL)

The PN7462AU integrates a dedicated USB PLL to generate a low-noise 48 MHz clock signal from the 27.12 MHz input signal coming from the external crystal (XTAL). The 48 MHz clock signal is used as the main clock for the USB interface.

**USB PLL features:**

- Low-skew, Peak-peak cycle-to-cycle jitter 48 MHz output clock (100 ps typical)
- Low Power in active more, low power-down current
- On-Chip loop filter, no external RC components needed

### 7.2.1 USB PLL Clock source selection

The USB PLL input can be selected between:

- Crystal oscillator output (default)

CLKGEN_HFO_XTAL_REG.XTAL_BYPASS = '0'

CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG.PLL_INPUT_BUFFER_BYPASS = '0'

- External clock input

CLKGEN_HFO_XTAL_REG. XTAL_BYPASS = '1'

CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG.PLL_INPUT_BUFFER_BYPASS = '0'

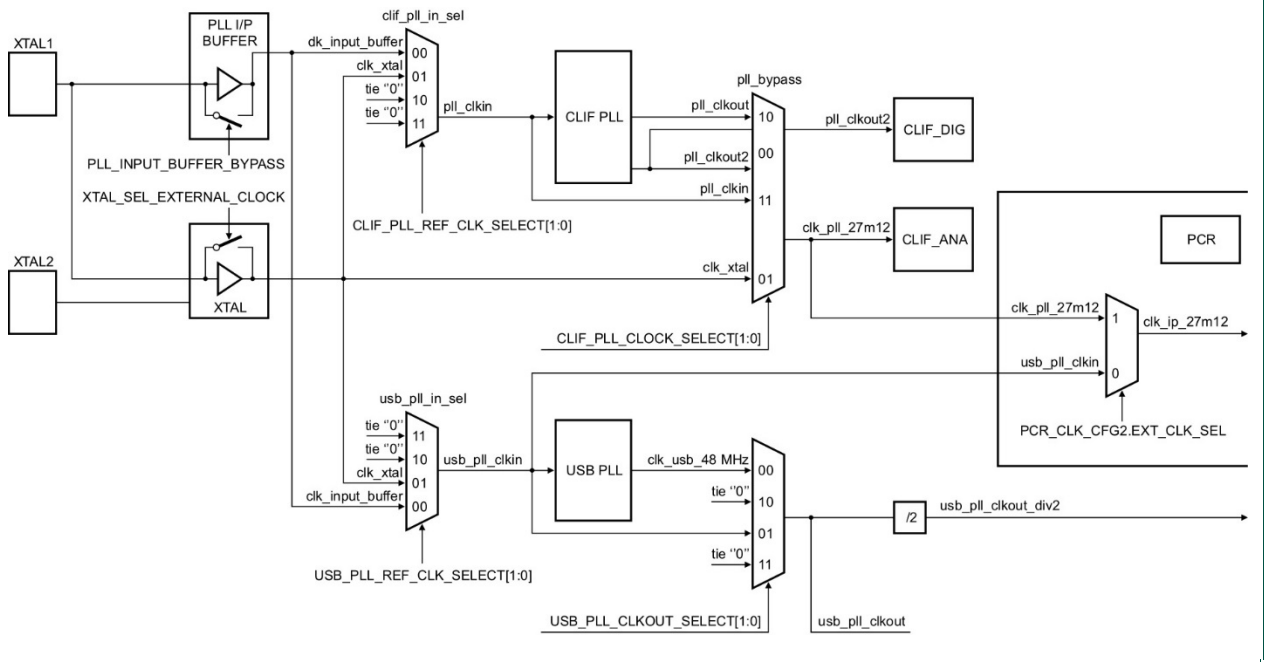



**Fig 18. PLL/ XTAL clock path diagram**

### 7.2.2 USB clock selection

The USB PLL output can be configured as:

- USB PLL Clock

CLKGEN_USB_PLL_CONTROL_REG.USB_PLL_CLKOUT_SELECT = ’00’

- Crystal Oscillator or External clock

  CLKGEN_USB_PLL_CONTROL_REG.USB_PLL_CLKOUT_SELECT = ’01’

- USB Clock Disable

  CLKGEN_USB_PLL_CONTROL_REG.USB_PLL_CLKOUT_SELECT = ’10’ or ‘11’

### 7.2.3 USB PLL frequency calculation

The USB PLL soft decoder selects pre-defined divider ratios and corresponding bandwidth of the PLL to guarantee stability. The soft decoder can only select two sets of divider parameters in order to have a ~48 MHz output clock from a 27.12 MHz input clock (Clkout=Clkinx(M/(NxP))).

CLKGEN_USB_PLL_CONTROL_REG.usb_pll_mnp_sel = ’0’ : M=69, N=13,P=3

Clkout=Clkinx(M/N.P) = 27.12 MHz × (69/(13 × 3)) = 48 MHz

CLKGEN_USB_PLL_CONTROL_REG.usb_pll_mnp_sel = ’1’ : M=92, N=13,P=4.

Clkout=Clkinx(M/N.P) = 27.12 MHz × (92/(13 × 4)) ~ 47.9815 MHz

The Soft Decoder can be bypassed in order to have the full control of the divider ratios. When CLKGEN_USB_PLL_CONTROL_REG.usb_pll_mnp_dec_selection is set to ’1’ M,N,P divider ratios are coming from CLKGEN_USB_PLL_MDEC_WO_SOFTDEC and CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC registers.

### 7.2.4 USB PLL Activation Sequence

The Activation Sequence with default MNP parameters comprises following steps:

1. Put the PLL in Power Down Mode

   CLKGEN_USB_PLL_CONTROL_REG.USB_PLL_PD = ’1’

2. Enable the PLL input buffer
   CLKGEN_USB_PLL_CONTROL_REG.USB_PLL_INPUT_BUFFER_ENABLE = ’1’

3. Set the expected PLL input clock frequency for the clock detector by setting the detection window length and the amount of expected detected input clock rising edges in this detection window respectively defined by the following registers CLKGEN_INPUT_CLOCK_DETECTOR_CONTROL_REG.

   By default, these register are set to detect a 27.12 MHz input clock.

4. Enable Input Clock Detector
   CLKGEN_INPUT_CLOCK_DETECTOR_CONTROL_REG.USB_CLK_DETECT_ENABLE = ’1’.
5. Poll for CLKGEN_STATUS_REG.CLK_IN_DETECT_DONE = ’1’ (after ~5.2 us by default). This will only indicate that the detection procedure is finished, not that there is clock and/or the frequency is the expected one.
6. Check that CLKGEN_STATUS_REG.CLK_IN_OK = ’1’.

   This will indicate if there is a clock of a frequency higher or equals to the expected one (27.12 MHz by default) at the input of the PLL. If this bit is 0 while CLK_IN_DETECT_DONE is high, this means either that there is no clock or that the clock has a frequency lower than the expected one (27.12 MHz), which will hamper the PLL functionality or give an unwanted PLL output frequency value.
7. Disable the Input Clock Detector
   CLKGEN_USB_PLL_CONTROL_REG. USB_CLK_DETECT_ENABLE = ’0’.
8. Exit the PLL from the Power Down Mode
   CLKGEN_USB_PLL_CONTROL_REG.PLL_PD = ’0’
9. Enable the PLL
   CLKGEN_USB_PLL_CONTROL_REG.PLL_CLKEN = ’1’
10. Poll for CLKGEN_STATUS_REG.PLL_LOCK = ’1’ to confirm the lock status of the PLL.

    Software can start a new PLL input clock detection at any time by generating a low to high transition on the CLKGEN_INPUT_CLOCK_DETECTOR_CONTROL_REG.USB_CLK_DETECT_ENABLE register.

## 7.3 CLIF PLL

The integrated CLIF PLL is designed to generate a low-noise 27.12 MHz clock, which is used as time reference for the Contactless Interface when PN7462AU is in reader mode or acting as ISO/IEC 18092 initiator.

The frequency value of the reference clock that is fed, can be selected using CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG register between:

1. Crystal oscillator output (default)
2. External clock input

The CLIF PLL output can be configured using CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG registers as:

1. Fractional PLL output
2. XTAL oscillator output Clock
3. Regular PLL output
4. PLL input

The clock generator module provides a PLL/XTAL clock presence indicator signal to the CLIF. This signal is active when the clock coming from the PLL or XTAL, or a

combination of both (depending on the clock settings), is available. This signal can be overridden using CLIF_PLL_CLK_IN_OK_BYPASS register.

### 7.3.1 Optimum divider settings for CLIF PLL

**Table 50. Optimum divider settings for PLL1 and PLL2**

| Fin [MHz] | n1 | m1 | p1 | m2 | p2 | Fref1 (MHz) | Fcco1 (MHz) | Fref2 (MHz) | Fcco2 (MHz) | Fpll_clkout (MHz) | Fpll_clkout2 (MHz) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 27.12 | 4 | 68 | 46 | 44 | 32.522 | 13.56 | 922.1 | 20.05 | 881.99 | 13.56 | 27.12 |

## 7.4 Register overview and description

### 7.4.1 Clock generator register overview

**Table 51. Clock generator register overview (base address 0x4001 0000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLKGEN_STATUS_REG | 0000h | 32 | R | 00000000h | CLKGEN status register |
| CLKGEN_HFO_XTAL_REG | 0004h | 32 | R/W | 00FFF001h | HFO and XTAL control register |
| CLKGEN_HFO_TRIM_REG | 0008h | 32 | R/W | 00000000h | HFO trimming value register |
| CLKGEN_USB_PLL_CONTROL_REG | 000Ch | 32 | R/W | 00F90001h | PLL Global Control Register |
| CLKGEN_USB_PLL_MDEC_WO_SOFTDEC_REG | 0010h | 32 | R/W | 00000000h | PLL M decoded divider ratio when the soft decoder is not used |
| CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC_REG | 0014h | 32 | R/W | 00000000h | PLL N and P decoded divider ratio when the soft decoder is not used |
| CLKGEN_CLIF_PLL1_CONTROL_REG | 0018h | 32 | R/W | 02E3B190h | CLIF PLL usage configurations |
| CLKGEN_CLIF_PLL2_CONTROL_REG | 001Ch | 32 | R/W | 02E121E0h | CLF PLL usage configurations |
| CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG | 0020h | 32 | R/W | 000000C8h | CLIF PLL integration configurations |
| CLKGEN_INPUT_CLOCK_DETECTOR_CONTROL_REG | 0024h | 32 | R/W | 0000100Dh | Input clock detector control |
| RESERVED | 0028h | 32 | R/W | 0000000Fh | Reserved |
| CLKGEN_CLOCK_PRESENCE_BYPASS_REG | 002Ch | 32 | R/W | 00000000h | clock presence for CLIF PLL |
| Unused | 0030h - 3FFFh | 32 | - | - | unused |

## 7.5 Clock Status Register description

**Table 52. CLKGEN_STATUS_REG (address 0000h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | - | 0 | Reserved |
| 26 | CLIF_CLOCK_PRESENCE_OK | R | 0 | Indicates the status of clkgen_clif_pll_lock2_o signal.<br>1: CLIF PLL2 lock signal set<br>0: CLIF PLL2 lock signal not set |
| 25 | XTAL_ACTIVATION_TIME OUT_ERROR | R | 0 | high if timeout for XTAL Activation is reached<br>1: Xtal activation has timed-out |
| 24 | XTAL_DETECT_OK | R | 0 | Indicates the presence of clock signal on clk_xtal if xtal_detect_enable is set.<br>1: Xtal detection done |
| 23 | XTAL_OSC_OK | R | 0 | high to indicate the clock is ready<br>1: Xtal osc clock is ready |
| 22 | CLIF_PLL_LOCK_OVERRIDEN | R | 0 | 1: pll_lock2 OR pll_bypass_lock2 is set<br>0: pll_lock2 OR pll_bypass_lock2 is not set |
| 21 | CLIF_CLK_IN_DETECT_DONE | R | 0 | CLIF PLL detection status<br>1: CLIF PLL clk_in detection done |
| 20 | CLIF_CLK_IN_OK | R | 0 | PLL input clock detector ok signal.<br>1: clk_in is present and correct<br>0: clk_in not ok |
| 19 | CLIF_PLL_LOCK2 | R | 0 | Lock detector Output for 2nd PLL<br>1: PLL2 lock is set<br>0: PLL2 lock is not set |
| 18 | CLIF_PLL_LOCK1 | R | 0 | Lock detector Output for 1st PLL.<br>1: PLL1 lock is set<br>0: PLL1 lock is not set |
| 17 | XTAL_OK | R | 0 | 1: XTAL oscillator is activated |
| 16 | XTAL_ENABLED | R | 0 | 1: XTAL oscillator is enabled<br>0: XTAL oscillator disabled |
| 15:8 | CLK_IN_EDGES_COUNTER | R | 0 | input clock edges counter value when the USB_PLL_CLK_IN detection completes successfully |
| 7 | CLK_IN_DETECT_DONE | R | 0 | USB_PLL input clock detection status<br>1: USB PLL input clock detection completed |
| 6 | CLK_IN_OK | R | 0 | USB_PLL input clock detector OK Status.<br>1: CLK_IN is present and the frequency matches expected frequency |
| 5 | USB_PLL_FR | R | 0 | USB_PLL free running detector status<br>1: USB PLL is in free running mode |
| 4 | USB_PLL_PACK | R | 0 | USB_PLL post-divider ratio change acknowledge<br>1: USB_PLL post-divider ratio change has been Acknowledged<br>0: No post-divider ratio change |
| 3 | USB_PLL_NACK | R | 0 | USB_PLL pre-divider ratio change acknowledge<br>1: USB_PLL pre-divider ratio change has been Acknowledged<br>0: No pre-divider ratio change |
| 2 | USB_PLL_MACK | R | 0 | USB_PLL feedback divider ratio change acknowledge<br>1: USB_PLL feedback divider ratio change has been |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | Acknowledged<br>0: No feedback divider ratio change |
| 1 | USB_PLL_LOCK_OVERRIDEN | R | 0 | USB_PLL lock overriden status<br>1: CLKGEN_STATUS_REG.USB_PLL_lock or CLKGEN_USB_PLL_GLOBAL_CONTROL_REG.USB_PLL_lock_bypass is high.<br>0: USB PLL Lock is not set |
| 0 | USB_PLL_LOCK | R | 0 | USB_PLL lock status<br>1: USB PLL lock set<br>0: USB PLL lock is not set |

### 7.5.1 Oscillators register description

Oscillators are controlled by the registers shown in Table 53. More detailed descriptions follow. Writes to any unused bits are ignored.

**Table 53. Oscillators Registers**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLKGEN_HFO_XTAL_REG | 0004h | 32 | R/W | 00FFF001h | HFO and XTAL control register |
| CLKGEN_HFO_TRIM_REG | 0008h | 32 | R/W | 00000000h | HFO trimming value register |

#### 7.5.1.1 HFO and XTAL control register

The CLKGEN_HFO_XTAL_REG register contains the bits that Activate/Enable XTAL, enable HFO.

**Table 54. CLKGEN_HFO_XTAL_REG (address 0004h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | R/W | 0x00 | Reserved |
| 23:12 | XTAL_ACTIVATION_TIMEOUT | R/W | 0xFFF | Set the XTAL activation timeout value (in LFO Clock Cycles + 8); Default value > 10 ms |
| 11 | XTAL_DETECT_ENABLE | R/W | 0x00 | Enables the XTAL output clock presence detection if clk_in_detect_enable is low.<br>1: Enable XTAL clk detection<br>0: Disable XTAL clk detection |
| 10 | XTAL_SPARE3 | R/W | 0x00 | controls XTAL Spare3 |
| 9 | XTAL_SPARE2 | R/W | 0x00 | controls XTAL Spare2 |
| 8 | XTAL_SPARE1 | R/W | 0x00 | controls XTAL Spare1 |
| 7 | XTAL_SPARE0 | R/W | 0x00 | controls XTAL Pull Down<br>1: enable strong pull-down on clk_xtal port of clkgen_ana sub-block<br>0: disable pull-down on clk_xtal port of clkgen_ana sub-block |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 6 | XTAL_VOLTAGE_MUX_CLOCK | R/W | 0x00 | controls XTAL voltage Mux |
| 5 | XTAL_SEL_EXTERNAL_CLOCK | R/W | 0x00 | Controls XTAL external clock selection if XTAL_CONTROL_SW='1'<br>1: Select External clock<br>0: Select XTAL Oscillator clock |
| 4 | XTAL_ENABLE | R/W | 0x00 | controls XTAL Enable if XTAL_CONTROL_SW='1'<br>1: Enable for XTAL oscillator<br>0: Disable XTAL Oscillator |
| 3 | XTAL_ENABLE_KICK | R/W | 0x00 | Controls XTAL Enable Kick if XTAL_CONTROL_SW='1'<br>1: Enable Kick of XTAL Oscillator |
| 2 | XTAL_BYPASS | R/W | 0x00 | controls XTAL Bypass if XTAL_CONTROL_SW='1'<br>1: Bypass XTAL<br>0: XTAL not Bypassed |
| 1 | XTAL_CONTROL_SW | R/W | 0x00 | high to control the XTAL oscillator by SW<br>1: Enable software control of XTAL oscillator<br>0: Disable software control of XTAL oscillator |
| 0 | HFO_ENABLE | R/W | 0x01 | enables the HFO (activated by default)<br>1: Enable HFO<br>0: Disable HFO |

#### 7.5.1.2 HFO Trimming Value Register

**Table 55. CLKGEN_HFO_TRIMM_REG (address 0008h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R/W | 0x00 | Reserved |
| 4:0 | HFO_TRIMM | R/W | 0x00 | HFO trimming values |

## 7.6 USB PLL register description

The USB PLL is controlled by the registers shown in Table 56. Writes to any unused bits are ignored.

**Warning: Improper setting of USB PLL values may result in incorrect operation of the USB.**

**Table 56. USB PLL Registers**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLKGEN_USB_PLL_CONTROL_REG | 000Ch | 32 | R/W | 00F90001h | PLL global control register |
| CLKGEN_USB_PLL_MDEC_WO_SOFTDEC_REG | 0010h | 32 | R/W | 00000000h | PLL M decoded divider ratio when the soft decoder is not used |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC_REG | 0014h | 32 | R/W | 00000000h | PLL N and P decoded divider ratio when the soft decoder is not used |

### 7.6.1 PLL Control Register (CLKGEN_USB_PLL_CONTROL_REG - 000Ch)

The CLKGEN_USB_PLL_CONTROL_REG register contains the bits that enable and connect PLL1. Enabling USB PLL allows it to attempt to lock to the current settings of the multiplier and divider values. Connecting USB PLL causes the USB subsystem to run from the USB PLL output clock. The USB PLL must be set up, enabled, and lock established before it may be used as a clock source for the USB.

**Table 57. CLKGEN_USB_PLL_CONTROL_REG (address 000Ch)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | RESERVED | R/W | 0x00 | Reserved |
| 30 | USB_PLL_MNPSEL | R/W | 0x00 | M,N,P selection values for the Soft Decoder<br>0: M=600,N=113,P=3<br>1: M=92,N=13,P=4 |
| 29:28 | USB_PLL_CLKOUT_SELECT | R/W | 0x00 | 00: USB_PLL output clock 2<br>01: USB_PLL_clkin<br>10: tie '0'<br>11: tie '0' |
| 27:26 | USB_PLL_REF_CLK_SELECT | R/W | 0x00 | Selects the reference clock for USB PLL<br>00: clk_input_buffer<br>01: clk_xtal<br>10: tie '0'<br>11: tie '0' |
| 25 | USB_PLL_LOCK_BYPASS | R/W | 0x00 | 1: Bypass the USB_PLL lock output |
| 24 | USB_PLL_MNP_DEC_SELECTION | R/W | 0x00 | 1: M,N,P divider ratio are not coming from the soft decoder but from the CLKGEN_USB_PLL_MDEC_WO_SOFTDEC and CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC registers<br>0: M, N, P divider ratio are taken from the Soft Decoder |
| 23:19 | USB_PLL_INSELP | R/W | 0x1F | select the bandwidth (don't care if USB_PLL_BANDSEL='0') |
| 18:15 | USB_PLL_INSELI | R/W | 0x02 | select the bandwidth (don't care if USB_PLL_BANDSEL='0') |
| 14:11 | USB_PLL_INSELR | R/W | 0x00 | select the bandwidth (don't care if USB_PLL_BANDSEL='0') |
| 10 | USB_PLL_BANDSEL | R/W | 0x00 | bandwidth adjustment (to modify externally the bandwidth of the USB_PLL) |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 9 | USB_PLL_PREQ | R/W | 0x00 | USB_PLL post-divider ratio change request<br>1: Request change of USB_PLL post-divider ratio (ratio taken from register) |
| 8 | USB_PLL_NREQ | R/W | 0x00 | USB_PLL pre-divider ratio change request<br>1: Request change of USB_PLL pre-divider ratio (ratio taken from register) |
| 7 | USB_PLL_MREQ | R/W | 0x00 | USB_PLL feedback divider ratio change request<br>1: Request change of USB_PLL feedback divider ratio (ratio taken from register) |
| 6 | USB_PLL_FRM | R/W | 0x00 | USB_PLL free running mode<br>1: Enable free running mode of USB PLL |
| 5 | USB_PLL_SKEW_EN | R/W | 0x00 | USB_PLL skew mode<br>1: Enable skew mode of USB_PLL |
| 4 | USB_PLL_DIRECTO | R/W | 0x00 | Bypass of the USB_PLL post-divider<br>1: Enable Bypass of USB_PLL post-divider<br>0: USB_PLL post-divider Bypass disable |
| 3 | USB_PLL_DIRECTI | R/W | 0x00 | Bypass of the USB_PLL pre-divider<br>1: Enable Bypass of USB_PLL pre-divider<br>0: USB_PLL pre-divider Bypass disable |
| 2 | USB_PLL_BYPASS | R/W | 0x00 | Bypass of the USB_PLL (clkout=clkin)<br>1: Enable Bypass of USB_PLL<br>0: USB_PLL Bypass disable |
| 1 | USB_PLL_CLKEN | R/W | 0x00 | Enable the USB_PLL output clock<br>1: Enable output clock from USB PLL<br>0: Disable output clock from USB PLL |
| 0 | USB_PLL_PD | R/W | 0x01 | USB_PLL Into power down<br>1: USB PLL powered down<br>0: USB PLL powered up |

### 7.6.2 PLL M decoded divider ratio

The CLKGEN_USB_PLL_MDEC_WO_SOFTDEC_REG register contains the USB PLL multiplier and divider values. Changes to CLKGEN_USB_PLL_MDEC_WO_SOFTDEC_REG register do not take effect until a correct USB feed sequence has been given (see Section 7.2). Calculations for the USB PLL frequency, and multiplier and divider values are found in Section 7.2.

**Table 58. CLKGEN_USB_PLL_MDEC_WO_SOFTDEC_REG (address 0010h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:17 | RESERVED | R/W | 0x00 | Reserved |
| 16:0 | USB_MDEC_WO_SOFTDEC | R/W | 0x00 | divider ratio code for M-divider when soft decoder is not used |

### 7.6.3 PLL N and P decoded divider ratio

The CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC_REG register contains the USB PLL multiplier and divider values. Changes to CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC register do not take effect until a correct USB feed sequence has been given (see Section 7.2). Calculations for the USB PLL frequency, and multiplier and divider values are found in Section 7.2.

**Table 59. CLKGEN_USB_PLL_NDEC_PDEC_WO_SOFTDEC_REG (address 0014h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | R/W | 0x00 | Reserved |
| 15:10 | USB_PDEC_WO_SOFTDEC | R/W | 0x00 | divider ratio code for P-divider when soft decoder is not used |
| 9:0 | USB_NDEC_WO_SOFTDEC | R/W | 0x00 | divider ratio code for N-divider when soft decoder is not used |

## 7.7 CLIF PLL register description

CLIF PLL is controlled by the registers shown in Table 60. More detailed descriptions follow. Writes to any unused bits are ignored. A read of any unused bits will return a logic zero.

**Warning: Improper setting of CLIF PLL values may result in incorrect operation of the Contactless System!**

**Table 60. CLIF PLL register overview**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLKGEN_CLIF_PLL1_CONTROL_REG | 0018h | 32 | R/W | 02E3B190h | Clif pll usage configurations |
| CLKGEN_CLIF_PLL2_CONTROL_REG | 001Ch | 32 | R/W | 02E121E0h | Clif pll usage configurations |
| CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG | 0020h | 32 | R/W | 000000C0h | Clif pll integration configurations |
| CLKGEN_INPUT_CLOCK_DETECTOR_CONTROL_REG | 0024h | 32 | R/W | 000011ADh | input clock detector control |
| CLKGEN_CLOCK_PRESENCE_BYPASS_REG | 002Ch | 32 | R/W | 0000000Fh | yes clock presence for clif_pll |

### 7.7.1 CLIF PLL CONTROL1 REG

The CLKGEN_CLIF_PLL1_CONTROL_REG register contains the CLIF PLL multiplier and divider values. Changes to CLKGEN_CLIF_PLL1_CONTROL_REG register do not take effect until a correct CLIF PLL feed sequence has been given. Calculations for the USB PLL frequency, and multiplier and divider values are found in Section 7.2.

Table 61. CLKGEN_CLIF_PLL1_CONTROL_REG (address 0018h)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R/W | 0x00 | Reserved |
| 27 | CLIF_PLL_LIMUP_OFF1 | R/W | 0x00 | pulse limiter for CLIF_PLL 1 |
| 26 | CLIF_PLL_FREQ_LIM1 | R/W | 0x00 | frequency limiter for CLIF_PLL 1 |
| 25:24 | CLIF_PLL_SELP1 | R/W | 0x02 | Pins to select the BW of CLIF_PLL 1 |
| 23:22 | CLIF_PLL_SELI1 | R/W | 0x03 | Pins to select the BW of CLIF_PLL 1 |
| 21:20 | CLIF_PLL_SELR1 | R/W | 0x02 | Pins to select the BW of CLIF_PLL 1 |
| 19 | CLIF_PLL_FUNC_TEST2_P1 | R/W | 0x00 | 1: Enable functional CLIF_PLL chain test of divider P1 |
| 18 | CLIF_PLL_FUNC_TEST1_P1 | R/W | 0x00 | 1: Enable functional divider test of divider P1 |
| 17:12 | CLIF_PLL_DIVP1 | R/W | 0x3B | Feedback divider ratio P1 |
| 11 | CLIF_PLL_FUNC_TEST2_M1 | R/W | 0x00 | 1: Enable functional CLIF_PLL test chain of divider M1 |
| 10 | CLIF_PLL_FUNC_TEST1_M1 | R/W | 0x00 | 1: Enable functional divider test of divider M1 |
| 9:3 | CLIF_PLL_DIVM1 | R/W | 0x32 | Feedback divider ratio M1 |
| 2 | CLIF_PLL_BYPASS_LOCK1 | R/W | 0x00 | Bypass of Lock1<br>1: Bypass clif_pll_lock1 |
| 1 | CLIF_PLL_FUNC_TEST2_LOCK1 | R/W | 0x00 | 1: Enable functional CLIF_PLL test chain of lock detector 1 |
| 0 | CLIF_PLL_FUNC_TEST1_LOCK1 | R/W | 0x00 | 1: Enable functional divider test of lock detector 1 |

### 7.7.2 CLIF PLL CONTROL2 REG

The CLKGEN_CLIF_PLL2_CONTROL_REG register contains the CLIF PLL multiplier and divider values. Changes to CLKGEN_CLIF_PLL2_CONTROL_REG register do not take effect until a correct CLIF PLL feed sequence has been given. Calculations for the USB PLL frequency, and multiplier and divider values are found Section 7.2.

Table 62. CLKGEN_CLIF_PLL2_CONTROL_REG (address 001Ch)

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R/W | 0x00 | Reserved |
| 27 | CLIF_PLL_LIMUP_OFF2 | R/W | 0x00 | pulse limiter for CLIF_PLL 2 |
| 26 | CLIF_PLL_FREQ_LIM2 | R/W | 0x00 | frequency limiter for CLIF_PLL 2 |
| 25:24 | CLIF_PLL_SELP2 | R/W | 0x02 | Pins to select the BW of CLIF_PLL 2 |
| 23:22 | CLIF_PLL_SELI2 | R/W | 0x03 | Pins to select the BW of CLIF_PLL 2 |
| 21:20 | CLIF_PLL_SELR2 | R/W | 0x02 | Pins to select the BW of CLIF_PLL 2 |
| 19 | CLIF_PLL_FUNC_TEST2_P2 | R/W | 0x00 | 1: Enable functional CLIF_PLL chain test of divider P2 |
| 18 | CLIF_PLL_FUNC_TEST1_P2 | R/W | 0x00 | 1: Enable functional divider test of divider P2 |
| 17:12 | CLIF_PLL_DIVP2 | R/W | 0x12 | Feedback divider ratio P2 |
| 11 | CLIF_PLL_FUNC_TEST2_M2 | R/W | 0x00 | 1: Enable functional CLIF_PLL test chain of divider M2 |
| 10 | CLIF_PLL_FUNC_TEST1_M2 | R/W | 0x00 | 1: Enable functional divider test of divider M2 |
| 9:3 | CLIF_PLL_DIVM2 | R/W | 0x3C | feedback divider ratio M2 |
| 2 | CLIF_PLL_BYPASS_LOCK2 | R/W | 0x00 | Bypass of Lock2 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Bypass clif_pll_lock2 |
| 1 | CLIF_PLL_FUNC_TEST2_LOCK2 | R/W | 0x00 | 1: Enable functional CLIF_PLL test chain of lock detector 2 |
| 0 | CLIF_PLL_FUNC_TEST1_LOCK2 | R/W | 0x00 | 1: Enable functional divider test of lock detector 2 |

### 7.7.3 CLIF PLL GLOBAL CONTROL REG

The CLIF PLL GLOBAL CONTROL REG register contains the bits that enable and connect CLIF PLL. Enabling CLIF PLL allows it to attempt to lock to the current settings of the multiplier and divider values.

**Table 63. CLKGEN_CLIF_PLL_GLOBAL_CONTROL_REG (address 0020h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:15 | RESERVED | R/W | 0x00 | Reserved |
| 14 | CLIF_PLL_CLK_IN_OK_BYPASS | R/W | 0x00 | CLIF PLL clk_in detection override<br>1: Override pll_clk_in detection |
| 13:12 | CLIF_PLL_REF_CLK_SELECT | R/W | 0x00 | Select the reference clock for CLIF PLL<br>00: Clk_input_buffer<br>01: clk_xtal<br>10: tie ‘0’<br>11: tie ‘0’ |
| 11 | RESERVED | R/W | 0x00 | Reserved |
| 10 | CLIF_CLK_DETECT_ENABLE | R/W | 0x00 | 1: Enable CLIF_PLL input clock detector (clk_in).<br>Higher prior than xtal_detect_enable. |
| 9:7 | CLIF_PLL_INPUT_FREQ_SEL | R/W | 0x01 | Select input frequency for the CLIF_PLL: 13, 19.2, 24, 26, 38.4 or 52 MHz |
| 6:5 | CLIF_PLL_CLOCK_SELECT | R/W | 0x02 | Selects output clock from CLIF_PLL<br>00: CLIF_PLL clockout2<br>01: clk_xtal<br>10: CLIF_PLL clockout<br>11: CLIF_PLL input clock |
| 4 | PLL_INPUT_BUFFER_BYPASS | R/W | 0x00 | Bypass PLL input buffer (the buffer for clock going into USB PLL and intPLL)<br>1: Bypass the PLL input buffer |
| 3 | PLL_INPUT_BUFFER_ENABLE | R/W | 0x01 | 1: Enable the PLL Input Buffer (the buffer for clock going into USB PLL and intPLL)<br>0: Disable the PLL input Buffer |
| 2 | CLIF_PLL_FUNC_TEST_N1 | R/W | 0x00 | 1: Enable functional divider test and CLIF_PLL test chain of divider M1<br>0: Disable functional divider test and CLIF_PLL test chain of divider M1 |
| 1 | CLIF_PLL_DIVN1 | R/W | 0x00 | Pre-divider selection for CLIF PLL1 |
| 0 | CLIF_PLL_ENABLE | R/W | 0x00 | 1: Enable the CLIF_PLL |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 0: Disable the CLIF PLL |

### 7.7.4 INPUT CLOCK DETECTOR CONTROL REGISTER

**Table 64. CLKGEN_INPUT_CLOCK_DETECTOR_CONTROL_REG (address 0024h)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:14 | RESERVED | R/W | 0x0 | Reserved |
| 14 | USB_PLL_CLK_IN_OK_BYPASS | R/W | 0x0 | usb pll clk_in detection override<br>1: usb_pll_clk_in detection overridden. Clk_in_ok set to '1' |
| 13 | USB_CLK_DETECT_ENABLE | R/W | 0x0 | Enable usb_pll_clk_in detect<br>1: Enable usb_pll_clk_in detection |
| 12:5 | INPUT_USB_CLOCK_EDGES_NUMBER | R/W | 0x80 | Defines the expected amount of input clock edges during the detection window length. Default value is set to detect a 27.12 MHz input clock. |
| 4:0 | DETECTION_WINDOW_LENGTH | R/W | 0x0D | Defines the detection window length (in HFO/8 clock cycles).<br>Default value is set to detect a 27.12 MHz input clock. |

### 7.7.5 CLOCK PRESENCE BYPASS REG

**Table 65. CLKGEN_CLOCK_PRESENCE_BYPASS_REG (address 002Ch)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:2 | RESERVED | R/W | 0x0 | Reserved |
| 1 | CLOCK_PRESENCE_BYPASS_VAL | R/W | 0x0 | Value to apply to clif_pll_lock2_o signal when corresponding enable bit is set<br>0: set clif_pll_lock2_o signal to 0<br>1: set clif_pll_lock2_o signal to 1 |
| 0 | CLOCK_PRESENCE_BYPASS_ENABLE | R/W | 0x0 | 1: Enable bypass of the clif_pll_lock2_o signal to the value stored in clock_presence_bypass_val<br>0: Disable Bypass of the clif_pll_lock2_o signal |

# 8. Power clock and reset (PCR)

The Power, Clock & Reset Unit (PCR) handles the digital startup of the PN7462AU and manages the behavior of the system in low power and active modes. The PCR unit is the only digital block that is powered in the standby mode.

The PCR unit provides following functionalities**:**

- Reset management
- Power on, standby, USB suspend and power off management
- Wake-up management
- Clock gating management for power consumption reduction
- I/O pad management

Fig 19 shows the block diagram of the PCR Unit with its main blocks:

- Reset Generator
- Standby Control
- Clock Box
- Pad Control

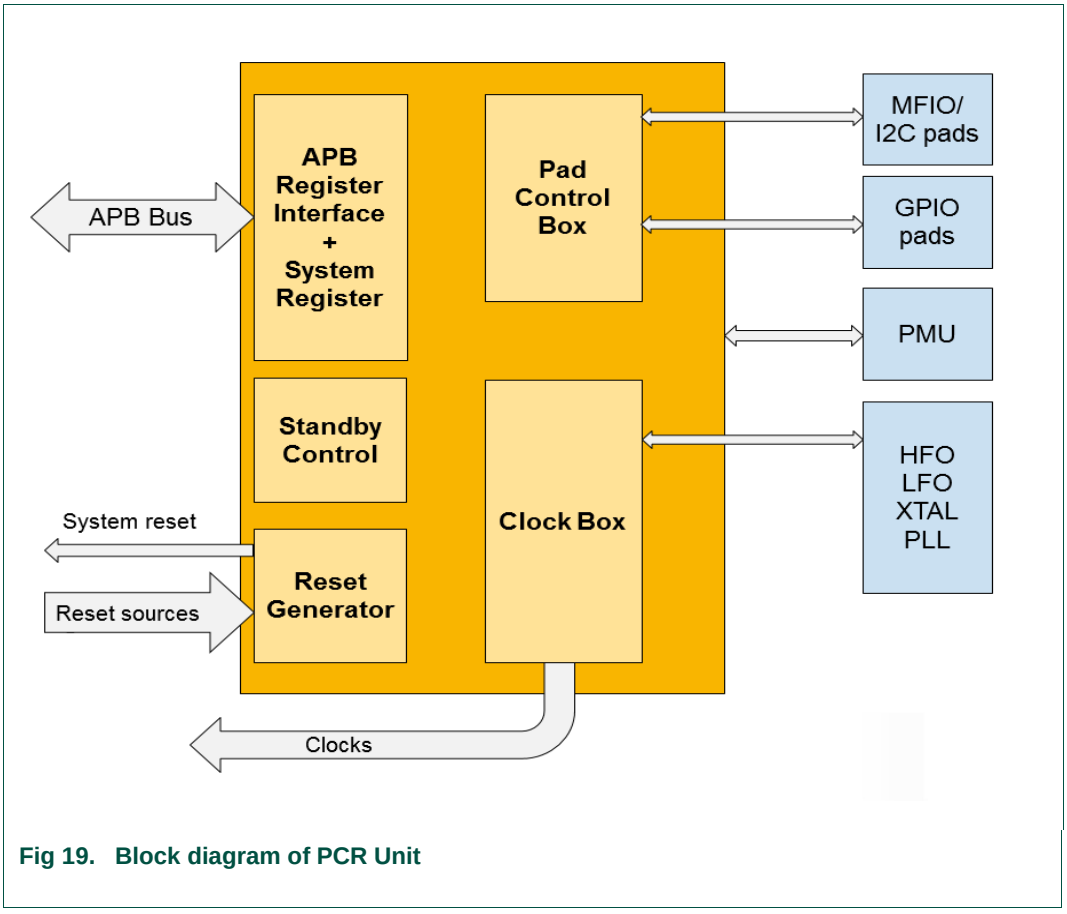



**Fig 19. Block diagram of PCR Unit**

## 8.1 Reset sources

The PN7462AU has 6 possible rest sources. These are described in detail in Table 66.

**Table 66. Reset sources**

| Source | Description |
|---|---|
| Software - PCR | Software reset from the PCR peripheral; Resets the whole chip except the PCR and the ARM debug interface |
| Software - ARM | Software reset from the ARM processor; Resets the whole chip except the PCR and the ARM debug interface |
| I2C interface | I2C Standard 3.0 defines a method to reset the chip via an I2C command; This feature can be disabled; Resets the whole chip except the PCR and the ARM debug interface |
| Watchdog | Reset the chip if the watchdog threshold is not periodically reloaded; Resets the whole chip except the PCR and the ARM debug interface |
| VBUS - PMU | Power-on reset sequence; Resets the complete chip when the voltage is above 2.3 V |
| RST_N pin | External reset triggered via reset pin; Resets the complete chip when the RST_N pin is set to low |

These reset sources trigger the reset generator that generates a global reset pulse. The Reset Generator is active high level sensitive to the reset sources. As long as one reset source is high, the global reset will be active. After releasing the reset source the reset pulse will be prolonged to at least one cycle. The power-on reset sequence is asserted when the device is powered up. It is used to keep the system in reset state until proper supply conditions are established. This point is achieved when the internal supply voltage reaches 1.55 V.

When the internal reset is removed, the processor begins executing at address 0, which is initially the reset vector mapped from the boot block. At that point, all of the processor and peripheral registers have been initialized to predetermined values.

## 8.2 Boot reason decoding

**Table 67. Boot reason decoding**

| PCR_STATUS.boot_reason | Values reflected in register | Description |
|---|---|---|
| startup_por | 0 | Analog Reset Sources (Startup Por or leave from HPD or VEN) |
| Rfld | 1 | RF Level Detector wakeup |
| wuc_cnt | 2 | Wakeup Timer |
| int_aux | 3 | Contact uart int_aux pad gives an interrupt |
| Ct | 4 | Contact card presence is detected |
| i2c | 5 | I2C address detected |
| RESERVED | 6 | Reserved |
| Spi | 7 | SPI slave received transaction |
| usb_resume | 8 | USB Resume signaling from Host |
| soft_reset | 9 | Soft reset given by software |
| Wdog | 10 | Watch dog timer timeout or ARM reset |
| Tvddmon | 11 | 5V detected by TVDD monitor |
| hif_reset | 12 | VEN from low to high back (only for test purpose) Ii2c slave or smb slave requested for reset |
| temp0 | 13 | Neg-edge detected for Temperature error from temperature sensor 0 |
| temp1 | 14 | Neg-edge detected for Temperature error from temperature sensor 1 |
| no_pvdd | 15 | PVDD dropped |
| pvdd_ilim | 16 | Pvdd current limiter input has become 0→1 |
| gpio | 17 | Gpio interrupt |
| hsu | 18 | HSUART transaction detected |

## 8.3 Power modes

The PN7462AU offers four different power modes allowing customer to optimize its energy consumption. These are:

- Active mode
- Standby mode
- USB suspend mode
- Hard Power Down mode

### 8.3.1 Full Power Mode (Active Mode)

In the active mode, all functionalities are available and all blocks are accessible. The PN7462AU is powered from the VBUS supply.

### 8.3.2 Standby Mode

In this mode, only small part of the IC is powered to maintain operation of the Power Control Unit, the LFO and a small set of registers for storing data during the standby operation. The MLDO is set to the low power mode. The possible wake-up sources are still powered. Depending on the application requirements, it is possible to put PVDDL_LDO into active mode, low power mode (default) or shut down mode.
The standby mode is triggered by the application firmware. Before entering the standby mode, the PN7462 executes automatically the deactivation of the contact card.
An internal mechanism prevents from entering into the standby mode either when no relevant wake up source is activated, or when conditions for a corresponding wakeup source are not present. The IC goes to power on mode again when a wake up is asserted.

#### 8.3.2.1 Entering Standby Mode

To enter standby mode, the firmware needs to operate in an infinite while loop of:

1. Programming the standby bit in the PCR_CTRL_REG.
2. Checking if there is any standby prevention reason.
3. If any reason found, then cater to the reason preventing entry of standby

   Go back to step 1.

This loop will be automatically broken when PN7462AU enters standby and comes out. As this will reset PN7462AU and restart the boot.

#### 8.3.2.2 Standby prevention root causes

Table below summarizes conditions preventing PN7462AU from entering the standby mode.

**Table 68. Standby prevention root causes**

| No. | Root cause description |
|---|---|
| 1 | A host communication is ongoing |
| 2 | If wake-up timer is enabled and its value is 0 |

| No. | Root cause description |
|---|---|
| 3 | Either RF level detector is activated as wakeup source and RF level detector is not enabled or RF field is already present |
| 4 | Host Interface is selected as wakeup source and no PVDD is available |
| 5 | PVDD current limiter input has seen a 0→1 |
| 6 | Negative-edge found on temperature error for temperature sensor 0, considering temperature sensor 0 is enabled for wakeup |
| 7 | Negative-edge found on temperature error for temperature sensor 1, considering temperature sensor 1 is enabled for wake-up |
| 8 | No host interface is selected |
| 9 | GPIO interrupt found |
| 10 | TVDD voltage has risen above 5 V |
| 11 | Card insertion or removal detected |
| 12 | Contact unart int_aux pin has given an interrupt |
| 13 | Contact deactivation is ongoing. |

### 8.3.3 USB suspend mode

In this mode, only a few parts of USB are still active but not clocked. All clock sources except LFO are stopped. PN7462AU will go into suspend state if there is no activity on the USB bus for more than 3 ms.

#### 8.3.3.1 Entering suspend mode

To enter suspend mode, the firmware needs to operate in an infinite while loop of:

1. Programming the suspend bit in the PCR_CTRL_REG
2. Wait for interrupt.
3. If the interrupt is SUSPEND_DEFAULTED then check for the standby prevention reason (same register used for standby).
4. If any reason found, then cater to the prevention reason.
5. Now go back to step 1

If the interrupt is SUSPEND, then it means P7462AU has entered and come out of suspend.

### 8.3.4 Wake-up from Standby/Suspend Mode

Standby/suspend mode is left if one of the following conditions is met:

- Host activity (SPI, I2C, HSU) for standby mode and USB Resume for suspend mode with following pre-conditions:
  - PVDD is available
  - One of interfaces is selected (hif_selection is != 0 )
  - In case of SPI being selected NSS==1
- Contact card insertion/removal detection
- Wake-up timer using a 6 bit counter and a match register with programmable standby/suspend mode duration from 50ms to 2.5s; Used to timely check for any contact or contactless card presence
- Active Reset Source: e.g. current overconsumption on the PVDD_OUT, voltage above 5V on TVDD_IN
- Disappearance of PVDD: Voltage drop below 1.8 V triggers wake-up; Always active
- RF level detection caused by activity on the CLIF interface e.g. by bringing card near to CLIF
- Temperature sensor threshold reached: when the temperature goes below the configured value, PN7462 wakes-up automatically; Each temperature sensor can be configured individually
- GPIO: transition from 0 to 1 on input GPIO pads can be used to wakeup

### 8.3.5 Hard Power Down Mode

This is the lowest power mode allowing for the highest reduction of the power consumption. All clocks are turned off, all LDOs are turned off, except the MLDO which is set to the low power mode

The PN7462AU enters the Hard Power Down mode when RST_N is set to zero or the VBUS voltage is going below 2.3 V.

The PN7462AU exits the Hard Power Down mode, when RST_N pin is set to high level and VBUS voltage goes above 2.3 V.

### 8.3.6 LDOs/PLLs in different power modes

**Table 69. LDOs/PLLs in power modes**

| LDO/PLL | Active | Standby | Suspend | HPD | Hardware/Software |
|---|---|---|---|---|---|
| LDOs | | | | | |
| MLDO | full power | low power | low power | low power | hardware |
| PVDDLDO | Full/Low power/OFF | Low power mode | Operational mode | Power down by hardware | software |

| LDO/PLL | Active | Standby | Suspend | HPD | Hardware/Software |
|---|---|---|---|---|---|
| DCDC/SCLDO/ VCCLDO | Full/Low power/OFF | OFF | OFF | enabling power switches. | software |
| TXLDO | Full/Low power/OFF | Standby-LDO mode | Standby-LDO mode | | software |
| PLLs and oscillators | | | | | |
| USB PLL | ON/OFF | OFF | OFF | OFF | OFF: hardware<br>ON: software |
| CLIF PLL | ON/OFF | OFF | OFF | OFF | software |
| XTAL | ON | OFF | OFF | OFF | hardware |
| HFO | ON | OFF | OFF | OFF | hardware |
| LFO | ON | ON | ON | OFF | hardware |

## 8.4 Clock box

The Clock Box is responsible for generating all clock signals for the system.
The PCR_CLK_CFG_REG and PCR_CLK_CFG2_REG are used by firmware to gate system and IP clocks going to different modules.

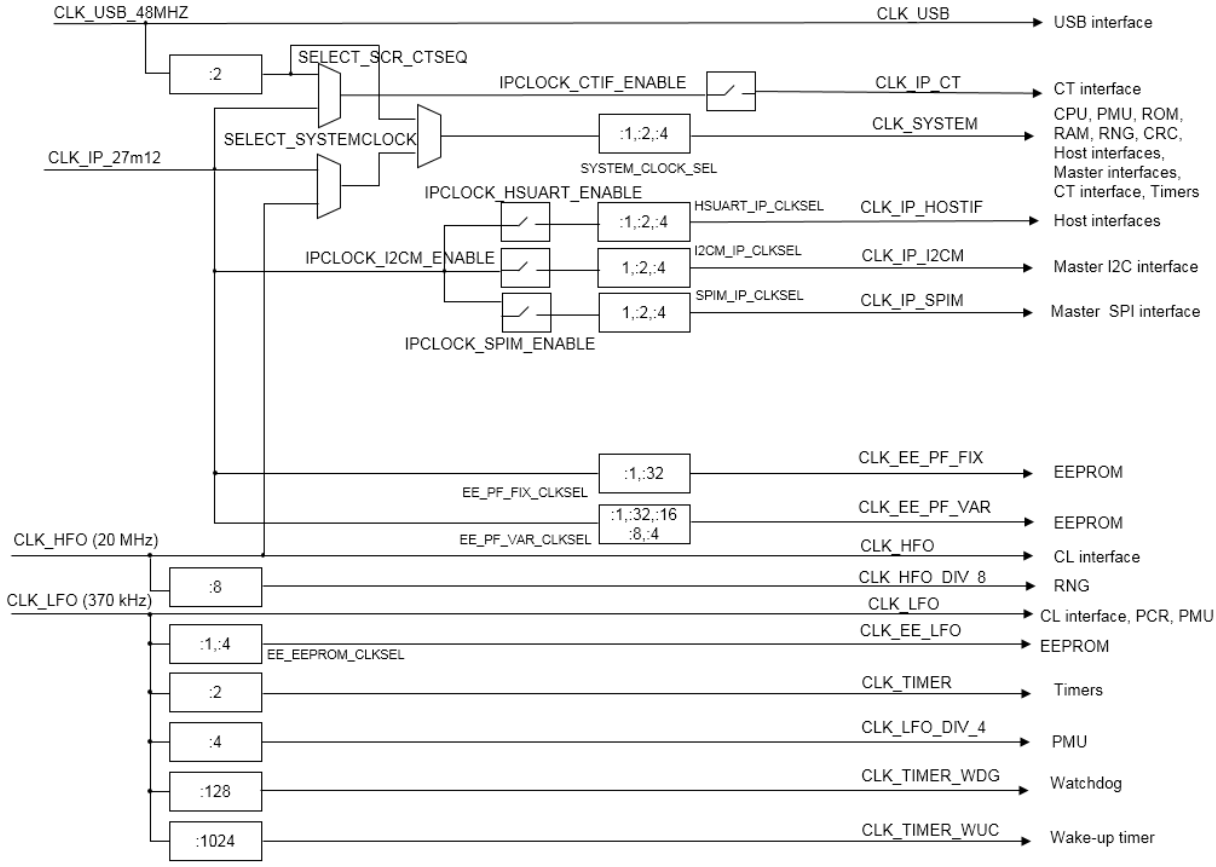



**Fig 20. Clock Box control**

The system clock source can be one among:

HFO→ 20 MHz

XTAL → 27.12 MHz (internal test purpose)

CLK_USB/2 → 24 Mhz. (only internal test purpose)

System clock must be always 20 MHz

The selection is done using the PCR_SELECT_SYSTEMCLOCK [2:0]:

001 ... 20 MHz clkHFO (default)

010 ... 24 MHz clkUSBPLL/2 (internal test purpose)

100 ... 27.12 MHz clkXtal (internal test purpose)

Others ... INVALID, should not be programmed

## 8.5 Clock Gating

In order to reduce the overall power consumption, the PN7462AU enables adjusting the system clock and integrates clock gating mechanisms.

The clocks of the following blocks can be activated or deactivated, depending on the peripherals used (see Fig 21):

- Contactless interface
- Contact interface
- Host interfaces
- $I^2C$ master interface
- SPI master interface
- CRC engine
- Timers
- Random generator
- System Clock
- EEPROM
- Flash memory

To enable the clock for this part, the corresponding bit in PCR_CLK_CFG_REG and PCR_CLK_CFG2_REG needs to be set.

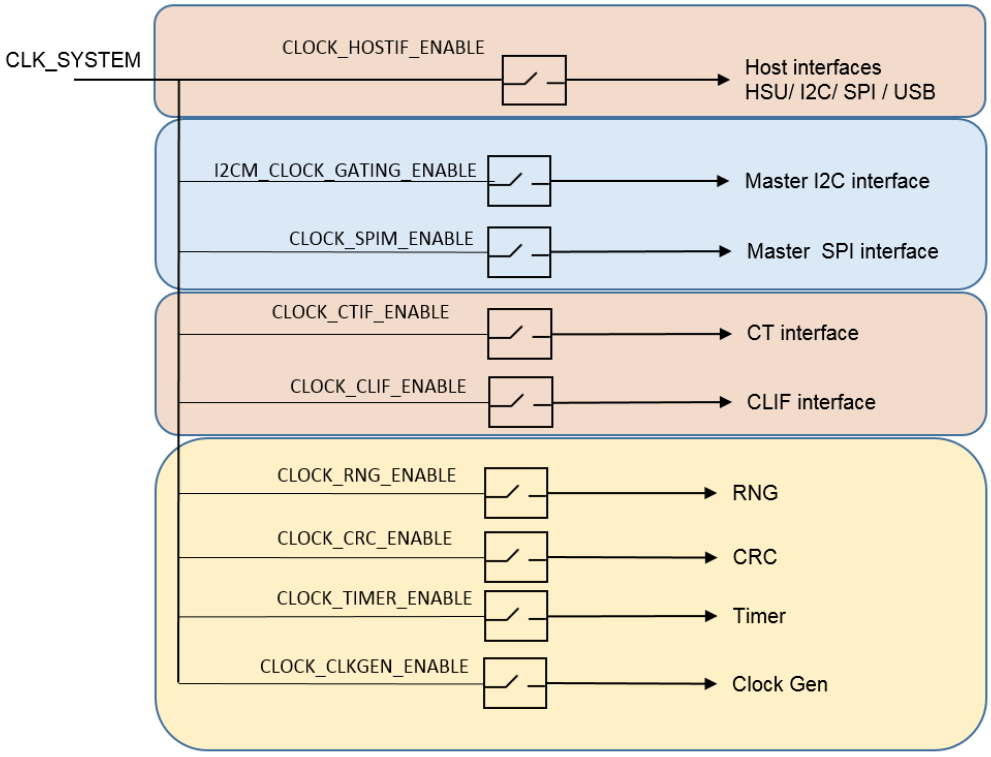



**Fig 21. Clock gating**

## 8.6 I/O Pad Management

I/O Pad Management allows:

- Connecting the GPIO/I2C/SPI to a peripheral IO for device pins that are not connected to a specific peripheral function
- Dynamic configuration as inputs or outputs or analog by FW
- Pull up, pull down or tri-state configuration

The GPIO read/write are made by the firmware using separate registers that allow reading, setting or clearing outputs. The value of the output register may be read back as well as the current state of the port pins. The pads controlled by the Pad Control Block are summarized in Table 70.

**Table 70. All digital controlled pads**

| PAD Name | Power Supply |
|---|---|
| DWL_REQ | PVDD_IN |
| ATX_A, ATX_B, ATC_C, ATX_D | PVDD_IN |
| CLK_AUX, INT_AUX, IO_AUX | PVDD_IN_M |
| D+, D- | PVDD_IN |

| PAD Name | Power Supply |
|---|---|
| MISO_M, MOSI_M, SCLK_M, NSS_M | PVDD_IN_M |
| SCL_M, SDA_M | PVDD_IN_M |
| GPIO1 to GPIO12 | PVDD_IN |
| SWDIO, SWDCLK | PVDD_IN |
| IRQ | PVDD_IN |

### 8.6.1 Hard Power Down (HPD) State of Pads

In the Hard Power Down mode, all digital pad signals will be masked.

### 8.6.2 Pad state in absence of PVDD

In absence of PVDD all input and output drivers will be disabled with a gate and all input signals from the PAD will be clamped.

### 8.6.3 Selecting host interface

The PN7462AU connects to host through four pads: ATX_A/ATX_B/ATX_C/ATX_D. There are three protocols by which PN7462AU connects to host through pads: I2C/high-speed-UART/SPI. The selection of which protocol to connect with is done by using configuration of PCR_SYS_REG.hif_selection bits in PCR_SYS_REG register described in Table 75.

## 8.7 Register overview

**Table 71. Register overview (base address 0x4002 4000)**

| Name | Address Offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| PCR_GPREG0_REG | 0x0000 | 32 | rw- | 0x00000000 | General-purpose register 0 for SW |
| PCR_GPREG1_REG | 0x0004 | 32 | rw- | 0x00000000 | General-purpose register 1 for SW |
| PCR_GPREG2_REG | 0x0008 | 32 | rw- | 0x00000000 | General-purpose register 2 for SW |
| PCR_SYS_REG | 0x000c | 32 | rw- | 0x00000100 | system configuration like Hostif selection and CT enabling |
| PCR_PMU_REG | 0x0010 | 32 | rw- | 0x0217010C | PMU interface. For LDO, bandgap, DC-to-DC converter configuration and sequences |
| PCR_RFLD_REG | 0x0014 | 32 | rw- | 0x00004032 | CLIF configuration |
| PCR_TEMP_REG | 0x0018 | 32 | rw- | 0x00058888 | temperature sensor calibration information |
| PCR_HOSTIF_WAKEUP_CFG_REG | 0x001c | 32 | rw- | 0x00000000 | configuring wake-up source for standby and Suspend |
| PCR_WAKEUP_CFG_REG | 0x0020 | 32 | rw- | 0x00000000 | configuring wake-up source for standby and Suspend |
| PCR_GPIO_WAKEUP_CFG_REG | 0x0024 | 32 | rw- | 0x000000FF | configuring wake-up source for standby and suspend |

| Name | Address Offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| PCR_BOOT_REG | 0x0028 | 32 | r-- | 0x00000000 | Bootup register with important values to be checked during bootup |
| PCR_CTRL_REG | 0x002C | 32 | rw- | 0x00000000 | Control register to enable standby/suspend/soft-restart/clearing boot register |
| PCR_CLK_CFG_REG | 0x0030 | 32 | rw- | 0x0087FE08 | Enable, gating, division value selection for clocks going to different Ips |
| PCR_CLK_CFG2_REG | 0x0034 | 32 | rw- | 0x00000B00 | Select lines for clock muxes, between different frequencies or between different division values |
| PCR_PADIN_REG | 0x0038 | 32 | r-- | 0x00000000 | Store value on input pads into this register |
| PCR_PADOUT_REG | 0x003C | 32 | rw- | 0x00000000 | output value to be driven onto output pads stored here |
| PCR_PAD_ATX_A_REG | 0x0040 | 32 | rw- | 0x00000000 | host if pad: i2c_scl/spi_nss/hsu_rx/usb_dp/smb_cl |
| PCR_PAD_ATX_B_REG | 0x0044 | 32 | rw- | 0x00000000 | host if pad: i2c_sda/spi_mosi/hsu_tx/usb_dm/smb_da |
| PCR_PAD_ATX_C_REG | 0x0048 | 32 | rw- | 0x00000000 | host if pad: i2c_adr0/spi_miso/hsu_rts_n/usb-N.A/smbalert |
| PCR_PAD_ATX_D_REG | 0x004C | 32 | rw- | 0x00000000 | host if pad: i2cadr1/spi_sck/hsu_cts_n/usb-N.A/smb-N.A |
| PCR_PADDWL_REQ_REG | 0x0050 | 32 | rw- | 0x00000001 | DWL_REQ pad configuration |
| PCR_PAD_INT_AUX_REG | 0x0054 | 32 | rw- | 0x00000001 | INT_AUX pad configuration |
| PCR_PAD_IO_AUX_REG | 0x0058 | 32 | rw- | 0x00000001 | IO_AUX pad configuration |
| PCR_PAD_CLK_AUX_REG | 0x005C | 32 | rw- | 0x00000002 | CLK_AUX pad configuration |
| PCR_PADIRQ_REG | 0x0060 | 32 | rw- | 0x00000002 | IRQ pad configuration |
| PCR_PADGPIO1_REG | 0x0064 | 32 | rw- | 0x00000001 | GPIO1 pad configuration |
| PCR_PADGPIO2_REG | 0x0068 | 32 | rw- | 0x00000001 | GPIO2 pad configuration |
| PCR_PADGPIO3_REG | 0x006C | 32 | rw- | 0x00000001 | GPIO3 pad configuration |
| PCR_PADGPIO4_REG | 0x0070 | 32 | rw- | 0x00000001 | GPIO4 pad configuration |
| PCR_PADGPIO5_REG | 0x0074 | 32 | rw- | 0x00000001 | GPIO5 pad configuration |
| PCR_PADGPIO6_REG | 0x0078 | 32 | rw- | 0x00000001 | GPIO6 pad configuration |
| PCR_PADGPIO7_REG | 0x007C | 32 | rw- | 0x00000001 | GPIO7 pad configuration |
| PCR_PADGPIO8_REG | 0x0080 | 32 | rw- | 0x00000001 | GPIO8 pad configuration |
| PCR_PADGPIO9_REG | 0x0084 | 32 | rw- | 0x00000001 | GPIO9 pad configuration |
| PCR_PADGPIO10_REG | 0x0088 | 32 | rw- | 0x00000001 | GPIO10 pad configuration |
| PCR_PADGPIO11_REG | 0x008C | 32 | rw- | 0x00000001 | GPIO11 pad configuration |
| PCR_PADGPIO12_REG | 0x0090 | 32 | rw- | 0x00000001 | GPIO12 pad configuration |

| Name | Address Offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| PCR_PADSWDIO_REG | 0x0094 | 32 | rw- | 0x00000006 | SWDIO pad slew rate configuration |
| INTERNAL_USE | 0x0098 | 32 | rw- | 0x00000006 | For internal use |
| RESERVED | 0x009C | 32 | rw- | 0x0000020C | Reserved |
| RESERVED | 0x00A0 | 32 | rw- | 0x00000008 | Reserved |
| PCR_PADIICM_REG | 0x00A4 | 32 | rw- | 0x00000280 | I2C master pad configuration |
| PCR_ANA_TX_STANDBY_REG | 0x00A8 | 32 | rw- | 0x00000000 | CLIF standby GSN value selection |
| PCR_ANA_TXPROT_REG | 0x00AC | 32 | rw- | 0x00000001 | CLIF configuration related to power down |
| INTERNAL_USE | 0x00B0 | 32 | rw- | 0x00000041 | For internal use |
| PCR_SPIM_REG | 0x00B4 | 32 | rw- | 0x00000040 | SPIM master pad configuration |
| PCR_CTIF_REG | 0x00B8 | 32 | rw- | 0x00000000 | CTIF presense detection pull-up |
| PCR_HOSTIF_SAVE1_REG | 0x00BC | 32 | rw- | 0x00000000 | host interface Tx/RX divider value storage during standby |
| PCR_HOSTIF_SAVE2_REG | 0x00C0 | 32 | rw- | 0x00000000 | host interface clock value storage during standby |
| PCR_TXLDO_MON_REG | 0x00C4 | 32 | rw- | 0x00000008 | TXLDO sequence management |
| PCR_BOOT2_REG | 0x00C8 | 32 | rw- | 0x00000000 | BOOT reason register extention. |
| PCR_GPREG3_REG | 0x00CC | 32 | rw- | 0x00000000 | general-purpose register 3 for SW |
| PCR_GPREG4_REG | 0x00D0 | 32 | rw- | 0x00000000 | general-purpose register 4 for SW |
| PCR_GPREG5_REG | 0x00D4 | 32 | rw- | 0x00000000 | general-purpose register 5 for SW |
| PCR_GPREG6_REG | 0x00D8 | 32 | rw- | 0x00000000 | general-purpose register 6 for SW |
| PCR_GPREG7_REG | 0x00DC | 32 | rw- | 0x00000000 | general-purpose register 7 for SW |
| PCR_GPIO_INT_ACTIVE_LOW_REG | 0x00E0 | 32 | rw- | 0x00000000 | register to program is GPIO interrupts are active low level/ falling edge sensitive |
| PCR_GPIO_INT_LEVEL_SENSE_REG | 0x00E4 | 32 | rw- | 0x00000000 | register to program if GPIO interrupts are level sensitive. |
| PCR_GPIO_INT_ACTIVE_BOH_EDGE_REG | 0x00E8 | 32 | rw- | 0x00000000 | register to program if GPIO interrupts are both edge sensitive |
| PCR_SELECT_SYSTEMCLOCK | 0x00EC | 32 | rw- | 0x00000001 | register to program the source for system clock. |
| PCR_ADV_RFLD_REG | 0x00F0 | 32 | rw- | 0x00000000 | register for configuring advanced RFLD detection FSM |
| PCR_ADV_RFLD_TEST_REG | 0x00F4 | 32 | rw- | 0x00000000 | configuration bits for testing advanced RFLD detection FSM |
| PCR_INT_CLR_ENABLE_REG | 0x3FD8 | 32 | -wm | 0x00000000 | interrupt clear enable |
| PCR_INT_SET_ENABLE_REG | 0x3FDC | 32 | -wm | 0x00000000 | interrupt set enable |
| PCR_INT_STATUS_REG | 0x3FE0 | 32 | r-m | 0x00000000 | interrupt status |

| Name | Address Offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| PCR_INT_ENABLE_REG | 0x3FE4 | 32 | r-m | 0x00000000 | interrupt enable |
| PCR_INT_CLR_STATUS_REG | 0x3FE8 | 32 | -wm | 0x00000000 | interrupt clear status |
| PCR_INT_SET_STATUS_REG | 0x3FEC | 32 | -wm | 0x00000000 | interrupt set status |

## 8.8 Register description

**Table 72. PCR_GPREG0_REG (address offset 0x00)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG0 | R/W | 0 | general-purpose register 0 for SW |

**Table 73. PCR_GPREG1_REG (address offset 0x04)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG1 | R/W | 0 | general-purpose register 1 for SW |

**Table 74. PCR_GPREG2_REG (address offset 0x08)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG2 | R/W | 0 | general-purpose register 2 for SW |

**Table 75. PCR_SYS_REG (address offset 0x0C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | rw | 0x00 | Reserved |
| 11 | AUTOMATIC_CT_DEACT | rw | 0x00 | 1: Enables automatic initiation of CT deactivation sequence when VBUSP voltage goes below programmed range. |
| 10 | AUTOMATIC_HPD | rw | 0x00 | 1: Enables PCR to go automatically into HPD state when the VBUS voltage goes below programmed voltage of 2.3 V/2.7 V<br>0: PCR stays in operating state even if VBUS goes below threshold voltage |
| 9 | PVDD_INT | rw | 0x00 | Indicates that PVDD is being supplied using internal PVDD LDO<br>1: Enable Internal PVDD LDO |
| 8 | ENABLE_CT | rw | 0x01 | 1: Enable the Contact interface.<br>0: Disable Contact Interface |
| 7 | PVDD_M_IRQ_VAL | rw | 0x00 | Selects the PVDD_M voltage trigger level<br>0: PVDD_M voltage trigger level 1.8 V<br>1: PVDD_M voltage trigger level 3.3 V |
| 6 | PVDD_M_IRQ_EN | rw | 0x00 | Enables the PVDD_M IRQ |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Enable PVDD_M IRQ<br>0: Disable PVDD_M IRQ |
| 5 | PVDD_IRQ_VAL | rw | 0x00 | Selects the PVDD voltage trigger level<br>0: PVDD voltage trigger level 1.8 V<br>1: PVDD voltage trigger level 3.3 V |
| 4 | PVDD_IRQ_EN | rw | 0x00 | Enables the PVDD IRQ<br>1: Enable PVDD IRQ<br>0: Disable PVDD IRQ |
| 3 | HOSTIF_SW_REGCONTROL_EN | rw | 0x00 | 1: Enabled control of USB D+,D- from ATX_A/B registers |
| 2:0 | HIF_SELECTION | rw | 0x00 | host interface selection<br>000: No Host interface selected<br>001: I2C selected as host interface<br>010: SPI selected as host interface<br>011: HSU selected as host interface<br>100 -USB selected as host interface<br>others - Invalid |

**Table 76. PCR_PMU_REG (address offset 0x10)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:29 | PBF_CONST_LOAD_VAL | rw | 0x00 | configuration bits for constant load on vdhf |
| 28 | PBF_EN_CONST_LOAD | rw | 0x00 | Power down signal to connect/disconnect a constant load to vdhf |
| 27 | VBATMON_OVERRIDE_VAL | rw | 0x00 | VBUS monitor override value<br>0: for 2.7 V<br>1: for 2.3 V |
| 26 | VBATMON_OVERRIDE_EN | rw | 0x00 | VBUS monitor override enable<br>1: Enable for VBUS monitor<br>0: Disable VBUS monitor |
| 25 | PD_PBF_FIELDSENS | rw | 0x01 | 1- Enable for pbf_pd_fieldsens |
| 24 | BG_TRIM_A | rw | 0x00 | bandgap trim bit |
| 23 | BG_TRIM_B | rw | 0x00 | bandgap trim bit |
| 22 | BG_TRIM_C | rw | 0x00 | bandgap trim bit |
| 21 | BG_TRIM_D | rw | 0x00 | bandgap trim bit |
| 17:20 | RESERVED | rw | 0x00 | Reserved |
| 16 | MLDO_LOWPOWER_BG_EN | rw | 0x01 | Controls mldo bandgap low power signals. |
| 15 | MLDO_LOWPOWER_VAL | rw | 0x00 | Value of mldo_low power signal |
| 14 | MLDO_LOWPOWER_EN | rw | 0x00 | Controls mldo low power signals |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Enable low power mode of MLDO |
| | | | | 0: Disable low power mode of MLDO |
| 13 | | | | To set the DC-to-DC converter in high impedance state enabling the testing of VCCLDO. VUP will be forced from outside. |
| | DCDC_OFF | rw | 0x00 | 1: Bypass DC-to-DC converter if DC-to-DC_off is low |
| 12 | RESERVED | rw | 0x00 | Reserved |
| 11 | | | | TXLDO clear standby mode |
| | TXLDO_STANDBY_CLEAR | rw | 0x00 | 1: Disable standby mode of TXLDO |
| 10 | | | | TXLDO enable standby mode |
| | TXLDO_ENABLE_STANDBY | rw | 0x00 | 1: Enable standby mode for low power consumption during standby/suspend |
| 9 | RESERVED | rw | 0x00 | Reserved |
| 8:4 | IBIAS_TRIMM | rw | 0x10 | IBIAS trim value |
| 3:2 | | | | Selects pvddldo mode normal/low power/soft start/power down |
| | | | | 11: power down |
| | | | | 10: soft start |
| | | | | 01: low power mode |
| | PVDDLDO_MODE | rw | 0x03 | 00: normal mode |
| 1:0 | RESERVED | rw | 0x00 | Reserved |

**Table 77. PCR_RFLD_REG (address offset 0x14)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | rw | 0x00 | Reserved |
| 15 | | | | HIGHER bias current for the env-detector |
| | RFLD_ENVDET_BOOST | rw | 0x00 | 1: Enable Higher bias current for the env-detector |
| 14 | | | | Higher bias current for comparator |
| | RFLD_COMP_BOOST | rw | 0x01 | 1: Enable Higher current for comparator |
| 13 | | | | Enable of automatic bias current regulation |
| | | | | 1: Enable of automatic bias current regulation |
| | RFLD_BIAS_ADPT_ENABLE | rw | 0x00 | 0: Disable of automatic bias current regulation |
| 12 | | | | Enable the chopper clock driver for the RF Level Detector. |
| | | | | 1: Enable the chopper clock driver for the RF Level Detector. |
| | RFLD_DRV_ENABLE | rw | 0x00 | 0: Disable the chopper clock driver for the RF Level Detector. |
| 11 | | | | Enable the chopper filter for RF Level Detector |
| | RFLD_FILTER_ENABLE | rw | 0x00 | 1: Enable the chopper filter for RF Level Detector |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 0: Disable the chopper filter for RF Level Detector |
| 10 | | | | Enable of RFLD comparator |
| | | | | 0: Disable RFLD comparator |
| | RFLD_COMP_ENABLE | rw | 0x00 | 1: Enable RFLD comparator |
| 9 | | | | Enable of RFLD envelope detector |
| | | | | 1: Enable RFLD envelope detector |
| | RFLD_ENVDET_ENABLE | rw | 0x00 | 0: Disable RFLD envelope detector |
| 8 | | | | Enable for RFLD reference voltage generator |
| | | | | 1: Enable RFLD reference voltage generator |
| | RFLD_VREF_ENABLE | rw | 0x00 | 0: Disable RFLD reference voltage generator |
| 7:4 | RFLD_REF_LO | rw | 0x03 | Higher reference value for RF level detector |
| 3:0 | RFLD_REF_HI | rw | 0x02 | Lower reference value for RF level detector |

**Table 78. PCR_TEMP_REG (address offset 0x18)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:26 | RESERVED | rw | 0x00 | Reserved |
| 25 | | | | Enable calibration of temperature sensor 1 |
| | | | | 1: Enable calibration of temperature sensor 1 |
| | TEMP_ENABLE_CAL_1 | rw | 0x00 | 0: Disable calibration of temperature sensor 1 |
| 24 | | | | Enable calibration of temperature sensor 0 |
| | | | | 1: Enable calibration of temperature sensor 0 |
| | TEMP_ENABLE_CAL_0 | rw | 0x00 | 0: Disable calibration of temperature sensor 0 |
| 23 | | | | Enable Hystere of temperature sensor 1 |
| | | | | 1: Enable Hystere of temperature sensor 1 |
| | TEMP_ENABLE_HYST_1 | rw | 0x00 | 0: Disable Hystere of temperature sensor 1 |
| 22 | | | | Enable Hystere of temperature sensor 0 |
| | | | | 1: Enable Hystere of temperature sensor 0 |
| | TEMP_ENABLE_HYST_0 | rw | 0x00 | 0: Disable Hystere of temperature sensor 0 |
| 21 | | | | Enable Temp Sensor 1 |
| | | | | 1: Enable Temp Sensor 1 |
| | TEMP_ENABLE_1 | rw | 0x00 | 0: Disable Temp Sensor 1 |
| 20 | | | | Enable Temp Sensor 0 |
| | | | | 1: Enable Temp Sensor 0 |
| | TEMP_ENABLE_0 | rw | 0x00 | 0: Disable Temp Sensor 0 |
| 19:18 | TEMP_DELTA_1 | rw | 0x01 | Selects temperature threshold detection for temperature sensor 1 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17:16 | TEMP_DELTA_0 | rw | 0x01 | Selects temperature threshold detection for temperature sensor 0 |
| 15:12 | TEMP_CAL_FINE_1 | rw | 0x08 | Trim value fine for temperature sensor 1 |
| 11:8 | TEMP_CAL_COURSE_1 | rw | 0x08 | Trim value course for temperature sensor 1 |
| 7:4 | TEMP_CAL_FINE_0 | rw | 0x08 | Trim value fine for temperature sensor 0 |
| 3:0 | TEMP_CAL_COURSE_0 | rw | 0x08 | Trim value course for temperature sensor 0 |

**Table 79. PCR_HOSTIF_WAKEUP_CFG_REG (address offset 0x1C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | rw | 0x00 | Reserved |
| 8:2 | I2C_ADDR | rw | 0x00 | I2C address for wake-up |
| 1 | RESERVED | rw | 0x00 | Reserved |
| 0 | EN_INTERFACE | rw | 0x00 | 1- Enable Wake-up host interface |

**Table 80. PCR_WAKEUP_CFG_REG (address offset 0x20)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:23 | RESERVED | rw | 0x00 | Reserved |
| 22 | EN_ADV_RFLD | rw | 0x00 | 1: Enable advanced RFLD level detector FSM. |
| 21 | EN_VBUS_LOW | rw | 0x00 | 1: Enable wake-up when vbus goes low |
| 20:11 | WUC_VALUE | rw | 0x00 | Wake-up timer value |
| 10 | EN_TVDD_MON | rw | 0x00 | 1: Enable wake-up from TVDD 5 V monitor |
| 9 | EN_INT_AUX | rw | 0x00 | 1: Enable wake-up is TDA (CTUART) gives a level high interrupt |
| 8 | EN_CT_PR | rw | 0x00 | 1: Enables wake-up if card is detected. |
| 7 | RESERVED | rw | 0x00 | Reserved |
| 6 | EN_PVDD_LIMITER | rw | 0x00 | 1: Enables wake-up if PVDD current limiter is risen |
| 5 | EN_GPIO_INT | rw | 0x00 | 1: Enables wake-up if GPIO gives any input |
| 4 | EN_TEMP1 | rw | 0x00 | 1: Enable temperature 1 error Wake-up |
| 3 | EN_TEMP0 | rw | 0x00 | 1: Enable temperature 0 error Wake-up |
| 2 | EN_RFLDT | rw | 0x00 | 1: Enable Wake-up RF level detector |
| 1 | RESERVED | rw | 0x00 | Set to “0” |
| 0 | EN_WUC | rw | 0x00 | 1: Enable Wake-up timer |

**Table 81. PCR_GPIO_WAKEUP_CFG_REG (address offset 0x24)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | rw | 0x00 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 11:0 | | | | Enables wake-up by the corresponding GPIO<br>0: gpio1<br>1: GPIO2 |
| | GPIO_WAKEUP_ENABLE | rw | 0x0FF | e.t.c …. |

**Table 82. PCR_BOOT_REG (address offset 0x28)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | RESERVED | rw | 0x00 | Reserved |
| 30 | USB_VBUS_OK | r- | 0x00 | Indicator for USB_VBUS is ok<br>1: USB_VBUS is available |
| 29 | POK_VBUS | r- | 0x00 | Indicator when VBUS > VBUSCritical when VBUSMonitor is enabled<br>1: VBUS > VBUSCritical<br>0: VBUS < VBUSCritical |
| 28 | POK_PVDD_M_3V | r- | 0x00 | Indicator for more than 3 V at PVDD_M pin<br>1: PVDD_M is available and over 3.3 V<br>0: PVDD_M is not over 3.3 V |
| 27 | POK_PVDD_3V | r- | 0x00 | Indicator for more than 3V at PVDD pin<br>1: PVDD is available and over 3.3 V<br>0: PVDD is not over 3.3 V |
| 26 | RESERVED | r- | 0x00 | Reserved |
| 25:22 | STBY_PREV_REASON | r- | 0x00 | Standby prevention reason |
| 21:2 | BOOT_REASON | r- | 0x00 | Boot up reason |
| 1 | POK_PVDD_M | r- | 0x00 | Indicator if PVDD_m is available<br>1: PVDD_m is available and over 1.8 V<br>0: PVDD_m is not available |
| 0 | POK_PVDD | r- | 0x00 | Indicator if PVDD is available<br>1: PVDD is available and over 1.8 V<br>0: PVDD is not available |

**Table 83. PCR_CTLR_REG (address offset 0x2C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | rw | 0x00 | Reserved |
| 5 | USB_VBUS_PULLDOWN | rw | 0x00 | 1: Enables the internal pulldown resistance to pulldown the USB_VBUS |
| 4 | CLR_BOOT_REGS | -x | 0x00 | Clearing Standby Prevention and Boot up register values in the PCR_BOOT_REG register |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Clear the boot register |
| 3 | RESERVED | -x | 0x00 | Reserved |
| 2 | | | | Enables entering suspend mode |
| | SUSPEND | -x | 0x00 | 1: Enter suspend mode |
| 1 | | | | entering standby mode |
| | STANDBY | -x | 0x00 | 1: Enter standby mode |
| 0 | | | | Trigger Soft Reset Source |
| | SOFT_RESET | -x | 0x00 | 1: Provide soft reset to the device |

**Table 84. PCR_CLK_CFG_REG (address offset 0x30)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | RESERVED | rw | 0x00 | Reserved |
| 30 | EECTRL_SYS_GATING_ENABLE | rw | 0x00 | 1: EEPROM controller system clock gating enable<br>0: EEPROM controller system clock gating disable |
| 29 | EECTRL_PF_GATING_ENABLE | rw | 0x00 | 1: EEPROM controller page flash clock gating enable<br>0: EEPROM controller page flash clock gating disable |
| 28 | EECTRL_EEPROM_GATING_ENABLE | rw | 0x00 | 1: EEPROM controller automatic clock gating enable<br>0: EEPROM controller automatic clock gating disable |
| 27 | IPCLOCK_CTIF_ENABLE | rw | 0x00 | 1: Enable contact interface IP clock<br>0: Disable contact interface IP clock |
| 26 | IPCLOCK_HSUART_ENABLE | rw | 0x00 | 1- Enable high speed UART IP clock<br>0: Disable high speed UART IP clock |
| 25 | IPCLOCK_SPIM_ENABLE | rw | 0x00 | 1: Enable SPI master UART IP clock<br>0: Disable SPI master UART IP clock |
| 24 | IPCLOCK_I2CM_ENABLE | rw | 0x00 | 1: Enable I2C master UART IP clock<br>0: Disable I2C master UART IP clock |
| 23 | CLOCK_CTIF_ENABLE | rw | 0x01 | 1: Enable contact source for Contact interface<br>0: Disable contact source for Contact interface |
| 22 | I2CM_CLOCK_GATING_ENABLE | rw | 0x00 | 1: Enable clock source for I2C master<br>0: Disable clock source for I2C master |
| 21 | CPU_CLKREQ_ENABLE | rw | 0x00 | 1: Enable the automatic clock request for ROM and RAM via the CPU |
| 20 | AUTOMATIC_CLOCKSTOP_AT_IDLE_ENABLE | rw | 0x00 | 1: Enable automatic clock gating for CRC, EECTRL, RNG and ROM when cpu is in idle mode |
| 19 | CLOCK_SPIM_ENABLE | rw | 0x01 | 1: Enable clock source for SPIM<br>0: Disable clock source for SPIM |
| 18 | RESERVED | rw | 0x00 | Reserved |
| 17 | CLOCK_HOSTIF_ENABLE | rw | 0x01 | 1: Enable clock source for HOSTIF<br>0: Disable clock source for HOSTIF |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 16 | CLOCK_TIMER_ENABLE | rw | | 1: Enable clock source for TIMER |
| | | | 0x01 | 0: Disable clock source for TIMER |
| 15 | CLOCK_CRC_ENABLE | rw | | 1: Enable clock source for CRC |
| | | | 0x01 | 0: Disable clock source for CRC |
| 14 | CLOCK_CLKGEN_ENABLE | rw | | 1: Enable clock source for CLKGEN |
| | | | 0x01 | 0: Disable clock source for CLKGEN |
| 13 | RESERVED | rw | 0x01 | Set to “0” |
| 12 | CLOCK_RNG_ENABLE | rw | | 1: Enable clock source for RNG |
| | | | 0x01 | 0: Disable clock source for RNG |
| 11 | CLOCK_CLIF_ENABLE | rw | | 1: Enable clock source for CLIF |
| | | | 0x01 | 0: Disable clock source for CLIF |
| 10 | LFO_EN | rw | | 1: Enable LFO |
| | | | 0x01 | 0 -Disable LFO |
| 9:4 | LFO_TRIMM | rw | 0x20 | Trim value for LFO |
| 3 | EN_SWIO_CLK | rw | 0x01 | 1: Enables the SWIO clock |
| 2 | SELECT_SCR_CTSEQ | rw | | Selects the clock source for the system clock generation<br>0 - clkXTAL (27.12 MHz) |
| | | | 0x00 | 1 - clkPLL/2 (24 MHz) |
| 1:0 | RESERVED | rw | 0x00 | Reserved |

**Table 85. PCR_CLK_CFG2_REG (address offset 0x34)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:18 | RESERVED | rw | 0x00 | Reserved |
| 19 | CTRL_TXLDO_CLK | rw | 0x00 | TXLDO clock division select. 0->LFO/2 , 1-> LFO |
| 18 | EXT_CLK_SEL | rw | 0x00 | Selects between XTAL clock and external clock.<br>0 - XTAL<br>1: external clock |
| 17:16 | HSUART_IP_CLKSEL | rw | 0x00 | Selects ip clock divider value for hsuart.<br>00: xtal/1<br>01: xtal/2<br>10: xtal/4<br>11: RESERVED |
| 15:14 | SPIM_IP_CLKSEL | rw | 0x00 | Selects ip clock divider value for spim.<br>00: xtal/1<br>01: xtal/2<br>10: xtal/4<br>11: RESERVED |
| 13:12 | I2CM_IP_CLKSEL | rw | 0x00 | Selects ip clock divider value for spim.<br>00: xtal/1<br>01 -.xtal/2<br>10: xtal/4<br>11: RESERVED |
| 11 | CTSEQ_CLKSEL | rw | 0x01 | 0: +10<br>1: +82 |
| 10 | EE_EEPROM_CLKSEL | rw | 0x01 | Selects the divider value for the lfo clock for the EEPROM module<br>0: LFO/1<br>1: lfo/4 |
| 9 | EE_PF_FIX_CLKSEL | rw | 0x01 | Selects between divided xtal value<br>0: No clock<br>1: xtal-clk/32 |
| 8:6 | EE_PF_VAR_CLKSEL | rw | 0x04 | Selects between and xtal divided clocks for page flash module.<br>0: No clock<br>1: Xtal-clk/4<br>2: Xtal-clk/8<br>3: Xtal-clk/16<br>4: Xtal-clk/32<br>5: 7 RESERVED |
| 5:4 | SPARE_CELL_CLK_CFGL | rw | 0x00 | Spare cells to be used for clock config. |
| 3:2 | RESERVED | rw | 0x00 | Reserved |
| 1:0 | SYSTEM_CLOCK_SEL | rw | 0x00 | Selects the divider for the system clock.<br>00: Divby1 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 01: DIVby2<br>10: DIVby4<br>11: Reserved |

**Table 86. PCR_PADIN_REG (address offset 0x38)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | - | 0x00 | Reserved |
| 27 | RESERVED | r- | 0x00 | Reserved |
| 26 | PADIN_CLK_AUX | r- | 0x00 | input value for CLK_AUX |
| 25 | PADIN_IO_AUX | r- | 0x00 | input value for IO_AUX |
| 24 | PADIN_INT_AUX | r- | 0x00 | input value for INT_AUX |
| 23 | PADIN_GPIO12 | r- | 0x00 | input value for GPIO12 |
| 22 | PADIN_GPIO11 | r- | 0x00 | input value for GPIO11 |
| 21 | PADIN_GPIO10 | r- | 0x00 | input value for GPIO10 |
| 20 | PADIN_GPIO9 | r- | 0x00 | input value for GPIO9 |
| 19 | PADIN_GPIO8 | r- | 0x00 | input value for GPIO8 |
| 18 | PADIN_GPIO7 | r- | 0x00 | input value for GPIO7 |
| 17 | PADIN_GPIO6 | r- | 0x00 | input value for GPIO6 |
| 16 | PADIN_GPIO5 | r- | 0x00 | input value for GPIO5 |
| 15 | PADIN_GPIO4 | r- | 0x00 | input value for GPIO4 |
| 14 | PADIN_GPIO3 | r- | 0x00 | input value for GPIO3 |
| 13 | PADIN_GPIO2 | r- | 0x00 | input value for GPIO2 |
| 12 | PADIN_GPIO1 | r- | 0x00 | input value for GPIO1 |
| 11 | PADIN_DWL_REQ | r- | 0x00 | input value for DWL_REQ |
| 10 | PADIN_MISO_M | r- | 0x00 | input value for MISO_M |
| 9 | PADIN_MOSI_M | r- | 0x00 | input value for MOSI_M |
| 8 | PADIN_SCLK_M | r- | 0x00 | input value for SCLK_M |
| 7 | PADIN_NSS_M | r- | 0x00 | input value for NSS_M |
| 6 | PADIN_SDA_M | r- | 0x00 | input Value SDA_M |
| 5 | PADIN_SCL_M | r- | 0x00 | input Value SCL_M |
| 4 | PADIN_IRQ | r- | 0x00 | input value for IRQ |
| 3 | PADIN_ATX_D | r- | 0x00 | input Value ATX_D |
| 2 | PADIN_ATX_C | r- | 0x00 | input Value ATX_C |
| 1 | PADIN_ATX_B | r- | 0x00 | input Value ATX_B |
| 0 | PADIN_ATX_A | r- | 0x00 | input Value ATX_A |

**Table 87. PCR_PADOUT_REG (address offset 0x3C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | - | 0x00 | Reserved |
| 28 | RESERVED | rw | 0x00 | Reserved |
| 27 | RESERVED | rw | 0x00 | Reserved |
| 26 | PADOUT_CLK_AUX | rw | 0x00 | output value for CLK_AUX |
| 25 | PADOUT_IO_AUX | rw | 0x00 | output value for IO_AUX |
| 24 | PADOUT_INT_AUX | rw | 0x00 | output value for INT_AUX |
| 23 | PADOUT_GPIO12 | rw | 0x00 | output value for GPIO12 |
| 22 | PADOUT_GPIO11 | rw | 0x00 | output value for GPIO11 |
| 21 | PADOUT_GPIO10 | rw | 0x00 | output value for GPIO10 |
| 20 | PADOUT_GPIO9 | rw | 0x00 | output value for GPIO9 |
| 19 | PADOUT_GPIO8 | rw | 0x00 | output value for GPIO8 |
| 18 | PADOUT_GPIO7 | rw | 0x00 | output value for GPIO7 |
| 17 | PADOUT_GPIO6 | rw | 0x00 | output value for GPIO6 |
| 16 | PADOUT_GPIO5 | rw | 0x00 | output value for GPIO5 |
| 15 | PADOUT_GPIO4 | rw | 0x00 | output value for GPIO4 |
| 14 | PADOUT_GPIO3 | rw | 0x00 | output value for GPIO3 |
| 13 | PADOUT_GPIO2 | rw | 0x00 | output value for GPIO2 |
| 12 | PADOUT_GPIO1 | rw | 0x00 | output value for GPIO1 |
| 11 | PADOUT_DWL_REQ | rw | 0x00 | output value for DWL_REQ |
| 10 | PADOUT_MISO_M | rw | 0x00 | output value for MISO_M |
| 9 | PADOUT_MOSI_M | rw | 0x00 | output value for MOSI_M |
| 8 | PADOUT_SCLK_M | rw | 0x00 | output value for SCLK_M |
| 7 | PADOUT_NSS_M | rw | 0x00 | output value for NSS_M |
| 6 | PADOUT_SDA_M | rw | 0x00 | output Value SDA_M |
| 5 | PADOUT_SCL_M | rw | 0x00 | output Value SCL_M |
| 4 | PADOUT_IRQ | rw | 0x00 | output value for IRQ |
| 3 | PADOUT_ATX_D | rw | 0x00 | output Value ATX_D |
| 2 | PADOUT_ATX_C | rw | 0x00 | output Value ATX_C |
| 1 | PADOUT_ATX_B | rw | 0x00 | output Value ATX_B |
| 0 | PADOUT_ATX_A | rw | 0x00 | output Value ATX_A |

**Table 88. PCR_PAD_ATX_A_REG (address offset 0x40)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for ATX_A |
| | ATX_A_SLEW_RATE | rw | 0x00 | 1: Enable slew for ATX_A |
| 3:2 | ATX_A_PUPD | rw | 0x00 | Enable PullUp/Down on ATX_A |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 10: Enable pull up |
| | | | | 11: Enable pull down |
| 1 | ATX_A_EN_OUT | rw | 0x00 | 1: Enables output driver for ATX_A |
| 0 | ATX_A_EN_IN | rw | 0x00 | 1: Enables input driver for ATX_A |

**Table 89. PCR_PAD_ATX_B_REG (address offset 0x44)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for ATX_B |
| | ATX_B_SLEW_RATE | rw | 0x00 | 1: Enable slew for ATX_B |
| 3:2 | | | | Enable Pull Up/Down on ATX_B |
| | | | | 10: Enable pull up |
| | ATX_B_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | ATX_B_EN_OUT | rw | 0x00 | 1: Enables output driver for ATX_B |
| 0 | ATX_B_EN_IN | rw | 0x00 | 1: Enables input driver for ATX_B |

**Table 90. PCR_PAD_ATX_C_REG (address offset 0x48)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for ATX_C |
| | ATX_C_SLEW_RATE | rw | 0x00 | 1: Enable slew for ATX_C |
| 3:2 | | | | Enable pull Up/Down on ATX_C |
| | | | | 10: Enable pull up |
| | ATX_C_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | ATX_C_EN_OUT | rw | 0x00 | 1: Enables output driver for ATX_C |
| 0 | ATX_C_EN_IN | rw | 0x00 | 1: Enables input driver for ATX_C |

**Table 91. PCR_PAD_ATX_D_REG (address offset 0x4C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | ATX_D_SLEW_RATE | rw | 0x00 | Select driver strength for ATX_D |
| 3:2 | | | | Enable pull Up/Down on ATX_D |
| | | | | 10: Enable pull up |
| | ATX_D_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | ATX_D_EN_OUT | rw | 0x00 | 1: Enables output driver for ATX_D |
| 0 | ATX_D_EN_IN | rw | 0x00 | 1: Enables input driver for ATX_D |

**Table 92. PCR_PADDWL_REQ_REG (address offset 0x50)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | DWLREQ_SLEW_RATE | rw | 0x00 | Select driver strength for DWLREQ<br>1: Enable slew for DWL_REQ |
| 3:2 | DWLREQ_PUPD | rw | 0x00 | Enable pull Up/Down on DWLREQ<br>10: Enable pull up<br>11: Enable pull down |
| 1 | DWLREQ_EN_OUT | rw | 0x00 | 1: Enables output driver for DWLREQ |
| 0 | DWLREQ_EN_IN | rw | 0x01 | 1: Enables input driver for DWLREQ |

**Table 93. PCR_PAD_INT_AUX_REG (address offset 0x54)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | rw | 0x00 | Reserved |
| 7 | INT_AUX_ACTIVE_LOW_EN | rw | 0x00 | Configures INT_AUX to be interpreted as active low signal<br>1: INT_AUX is active low<br>0: INT_AUX active high |
| 6 | INT_AUX_GPIOMODE_EN | rw | 0x00 | Puts the INT_AUX PAD in GPIO mode (By default in I2C mode)<br>1: Enable GPIO mode for INT_AUX pads<br>0: INT_AUX pad in functional mode |
| 5 | INT_AUX_SW_ENABLE | rw | 0x00 | Enabling software register control for INT_AUX<br>1: Enable software control for INT_AUX pad |
| 4 | INT_AUX_SLEW_RATE | rw | 0x00 | Select Driver Strength for INT_AUX<br>1: Enable slew for INT_AUX pad |
| 3:2 | INT_AUX_PUPD | rw | 0x00 | Enable PullUp/Down on INT_AUX<br>10: Enable Pull up<br>11: Enable Pull down |
| 1 | INT_AUX_EN_OUT | rw | 0x00 | 1: Enables output driver for INT_AUX |
| 0 | INT_AUX_EN_IN | rw | 0x01 | 1: Enables input driver for INT_AUX |

**Table 94. PCR_PAD_IO_AUX_REG (address offset 0x58)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:7 | RESERVED | rw | 0x00 | Reserved |
| 6 | IO_AUX_GPIOMODE_EN | rw | 0x00 | Puts the IO_AUX PAD in GPIO mode (By default in I2C mode)<br>1: IO_AUX pad in GPIO mode<br>0: IO_AUX pad in functional mode |
| 5 | IO_AUX_SW_ENABLE | rw | 0x00 | 1: Enabling software register control for IO_AUX |
| 4 | IO_AUX_SLEW_RATE | rw | 0x00 | Select Driver Strength for IO_AUX |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Enable Slew for IO_AUX |
| 3:2 | | | | Enable pull Up/Down on IO_AUX |
| | | | | 10: Enable pull up |
| | IO_AUX_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | IO_AUX_EN_OUT | rw | 0x00 | 1: Enables output driver for IO_AUX |
| 0 | IO_AUX_EN_IN | rw | 0x01 | 1: Enables input driver for IO_AUX |

**Table 95. PCR_PAD_CLK_AUX_REG (address offset 0x5C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:7 | RESERVED | rw | 0x00 | Reserved |
| 6 | | | | Enabling CLK_AUX pad in GPIO mode (By default in I2C mode) |
| | | | | 1: CLK_AUX pad in GPIO mode |
| | CLK_AUX_GPIOMODE_EN | rw | 0x00 | 0: CLK_AUX pad in functional mode |
| 5 | CLK_AUX_SW_ENABLE | rw | 0x00 | 1: Enabling software register control for CLK_AUX |
| 4 | | | | Select driver strength for CLK_AUX |
| | CLK_AUX_SLEW_RATE | rw | 0x00 | 1: Enable Slew for CLK_AUX |
| 3:2 | | | | Enable pull Up/Down on CLK_AUX |
| | | | | 10: Enable pull up |
| | CLK_AUX_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | CLK_AUX_EN_OUT | rw | 0x01 | 1: Enables output driver for CLK_AUX |
| 0 | CLK_AUX_EN_IN | rw | 0x00 | 1: Enables input driver for CLK_AUX |

**Table 96. PCR_PADIRQ_REG (address offset 0x60)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for IRQ |
| | IRQ_SLEW_RATE | rw | 0x00 | 1: Enable slew for IRQ |
| 3:2 | | | | Enable Pull Up/Down on IRQ |
| | | | | 10: Enable pull up |
| | IRQ_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | IRQ_EN_OUT | rw | 0x01 | 1: Enables output driver for IRQ |
| 0 | IRQ_EN_IN | Rw | 0x00 | 1: Enables input driver for IRQ |

**Table 97. PCR_PADGPIO1_REG (address offset 0x64)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | Rw | 0x00 | Reserved |
| 5 | RESERVED | Rw | 0x00 | Always set to “0” |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 4 | | | | Select driver strength for GPIO1 |
| | GPIO1_SLEW_RATE | Rw | 0x00 | 1: Enable Slew for GPIO1 |
| 3:2 | | | | Enable pull Up/Down on GPIO1 |
| | | | | 01: Enable pull up |
| | GPIO1_PUPD | Rw | 0x00 | 11: Enable pull down |
| 1 | GPIO1_EN_OUT | Rw | 0x00 | 1: Enables output driver for GPIO1 |
| 0 | GPIO1_EN_IN | Rw | 0x01 | 1: Enables input driver for GPIO1 |

**Table 98. PCR_PADGPIO2_REG (address offset 0x68)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | Rw | 0x00 | Reserved |
| 5 | RESERVED | Rw | 0x00 | Always set to “0” |
| 4 | | | | Select driver strength for GPIO2 |
| | GPIO2_SLEW_RATE | Rw | 0x00 | 1: Enable slew for GPIO2 |
| 3:2 | | | | Enable pull Up/Down on GPIO2 |
| | | | | 10: Enable pull up |
| | GPIO2_PUPD | Rw | 0x00 | 11: Enable pull down |
| 1 | GPIO2_EN_OUT | Rw | 0x00 | 1: Enables output driver for GPIO2 |
| 0 | GPIO2_EN_IN | Rw | 0x01 | 1: Enables input driver for GPIO2 |

**Table 99. PCR_PADGPIO3_REG (address offset 0x6C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | Rw | 0x00 | Reserved |
| 5 | RESERVED | Rw | 0x00 | Always set to “0” |
| 4 | | | | Select Driver Strength for GPIO3 |
| | GPIO3_SLEW_RATE | Rw | 0x00 | 1: Enable slew for GPIO3 |
| 3:2 | | | | Enable pull Up/Down on GPIO3 |
| | | | | 10: Enable pull up |
| | GPIO3_PUPD | Rw | 0x00 | 11: Enable pull down |
| 1 | GPIO3_EN_OUT | Rw | 0x00 | 1: Enables output driver for GPIO3 |
| 0 | GPIO3_EN_IN | Rw | 0x01 | 1: Enables input driver for GPIO3 |

**Table 100. PCR_PADGPIO4_REG (address offset 0x70)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | Rw | 0x00 | Reserved |
| 5 | RESERVED | Rw | 0x00 | Always set to “0” |
| 4 | | | | Select driver strength for GPIO4 |
| | GPIO4_SLEW_RATE | Rw | 0x00 | 1: Enable slew for GPIO4 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 3:2 | | | | Enable pull Up/Down on GPIO4 |
| | | | | 10: Enable pull up |
| | GPIO4_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | GPIO4_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO4 |
| 0 | GPIO4_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO4 |

**Table 101. PCR_PADGPIO5_REG (address offset 0x74)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | Rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for GPIO5 |
| | GPIO5_SLEW_RATE | Rw | 0x00 | 1: Enable slew for GPIO5 |
| 3:2 | | | | Enable pull Up/Down on GPIO5 |
| | | | | 10: Enable pull up |
| | GPIO5_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | GPIO5_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO5 |
| 0 | GPIO5_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO5 |

**Table 102. PCR_PADGPIO6_REG (address offset 0x78)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for GPIO6 |
| | GPIO6_SLEW_RATE | rw | 0x00 | 1: Enable slew for GPIO6 |
| 3:2 | | | | Enable Pull Up/Down on GPIO6 |
| | | | | 10: Enable pull up |
| | GPIO6_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | GPIO6_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO6 |
| 0 | GPIO6_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO6 |

**Table 103. PCR_PADGPIO7_REG (address offset 0x7C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for GPIO7 |
| | GPIO7_SLEW_RATE | rw | 0x00 | 1: Enable Slew for GPIO7 |
| 3:2 | | | | Enable Pull Up/Down on GPIO7 |
| | | | | 10: Enable Pull up |
| | GPIO7_PUPD | rw | 0x00 | 11: Enable Pull down |
| 1 | GPIO7_EN_OUT | rw | 0x00 | 1: Enables output DRIVER for GPIO7 |
| 0 | GPIO7_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO7 |

**Table 104. PCR_PADGPIO8_REG (address offset 0x80)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | GPIO8_SLEW_RATE | rw | 0x00 | Select driver strength for GPIO8<br>1: Enable slew for GPIO8 |
| 3:2 | GPIO8_PUPD | rw | 0x00 | Enable pull Up/Down on GPIO8<br>10: Enable pull up<br>11: Enable pull down |
| 1 | GPIO8_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO8 |
| 0 | GPIO8_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO8 |

**Table 105. PCR_PADGPIO9_REG (address offset 0x84)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | GPIO9_SLEW_RATE | rw | 0x00 | Select driver strength for GPIO9<br>1: Enable slew for GPIO9 |
| 3:2 | GPIO9_PUPD | rw | 0x00 | Enable pull Up/Down on GPIO9<br>10: Enable pull up<br>11: Enable pull down |
| 1 | GPIO9_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO9 |
| 0 | GPIO9_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO9 |

**Table 106. PCR_PADGPIO10_REG (address offset 0x88)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | GPIO10_SLEW_RATE | rw | 0x00 | Select driver strength for GPIO10<br>1: Enable slew for GPIO10 |
| 3:2 | GPIO10_PUPD | rw | 0x00 | Enable pull Up/Down on GPIO10<br>10: Enable pull up<br>11: Enable pull down |
| 1 | GPIO10_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO10 |
| 0 | GPIO10_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO10 |

**Table 107. PCR_PADGPIO11_REG (address offset 0x8C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | GPIO11_SLEW_RATE | rw | 0x00 | Select driver strength for GPIO11 |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Enable Slew for GPIO11 |
| 3:2 | | | | Enable Pull Up/Down on GPIO11 |
| | | | | 10: Enable pull up |
| | GPIO11_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | GPIO11_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO11 |
| 0 | GPIO11_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO11 |

**Table 108. PCR_PADGPIO12_REG (address offset 0x90)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | | | | Select driver strength for GPIO12 |
| | GPIO12_SLEW_RATE | rw | 0x00 | 1: Enable slew for GPIO12 |
| 3:2 | | | | Enable pull Up/Down on GPIO12 |
| | | | | 10: Enable pull up |
| | GPIO12_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | GPIO12_EN_OUT | rw | 0x00 | 1: Enables output driver for GPIO12 |
| 0 | GPIO12_EN_IN | rw | 0x01 | 1: Enables input driver for GPIO12 |

**Table 109. PCR_PADSWDIO_REG (address offset 0x94)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | rw | 0x00 | Reserved |
| 0 | | | | Select driver strength for SWDIO |
| | SWDIO_SLEW_RATE | rw | 0x00 | 1: Enable slew rate for SWDIO |

**Table 110. PCR_PADIICM_REG (address offset 0xA4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | rw | 0x00 | Reserved |
| 11 | | | | 1: Enable I2C master |
| | | | | 0: disable I2C Master. This allows programming I2C master pads as GPIO if IICM_SW_ENABLE =1 (software control mode) |
| | IICM_ENABLE | rw | 0x00 | |
| 10 | IICM_SW_ENABLE | rw | 0x00 | 1: Enables register control of I2C pads |
| 9 | SDA_EN_IN | rw | 0x01 | 1: Enables input driver for SDA |
| 8 | SDA_EN_OUT | rw | 0x00 | 1: Enables output driver for SDA |
| 7 | SCL_EN_IN | rw | 0x01 | 1: Enables input driver for SCL |
| 6 | SCL_EN_OUT | rw | 0x00 | 1: Enables output driver for SCL |
| 5:4 | | | | Enables pull up/down functionality on SDA |
| | SDA_PUPD | rw | 0x00 | 10: Enable pull up |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 11: Enable pull down |
| 3:2 | | | | Enables pull up/down functionality on SCL |
| | | | | 10: Enable pull up |
| | SCL_PUPD | rw | 0x00 | 11: Enable pull down |
| 1 | SDA_SLEW | rw | 0x00 | 1: Enables SDA slew rate |
| 0 | SCL_SLEW | rw | 0x00 | 1: Enables SCL slew rate |

**Table 111. PCR_ANA_TX_STANDBY_REG (address offset 0xA8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | TX_GSN_SRC_SEL | rw | 0x00 | Source of GSN value (0… PCR, 1… CLIF) |
| 3:0 | TX_GSN_CW_SB | rw | 0x00 | GSN Value for standby mode |

**Table 112. PCR_ANA_TXPROT_REG (address offset 0xAC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | rw | 0x00 | Reserved |
| 4 | RX_PROT_IDDQ | rw | 0x00 | 1: Set RX protection to power down for Iddq measurement |
| 3 | TXPROT_ENABLE_AUTO_FREEZE | rw | 0x00 | Enable automatic freeze during typeB demodulation: typeB_det =1 -> freeze; else -> no freeze |
| 2 | TXPROT_LIM_FREEZE | rw | 0x00 | Freeze limiter impedance, active high |
| 1 | TXPROT_PD_VREF | rw | p | Power down reference voltage generation for tx-prot |
| 0 | | | | 1: Enables tx protection |
| | TXPROT_ENABLE | rw | 0x01 | 0: Disables tx protection |

**Table 113. PCR_SPIM_REG (address offset 0xB4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:20 | RESERVED | rw | 0x00 | Reserved |
| 19 | | | | SPIM master second slave enabled: slave 1, slave 0 enabled by default. |
| | | | | 1: slave 1 enabled |
| | SPIM_NSS1_EN | rw | 0x00 | 0: slave 0 enabled |
| 18 | | | | SW control for SPIM_MOSI ECS when SPIM_SW_ENABLE='1' |
| | SPIM_MOSI_EN_OUT | rw | 0x00 | 1: SPIM MOSI enabled as output |
| 17 | | | | SW control for SPIM_MISO ECS when SPIM_SW_ENABLE='1' |
| | SPIM_MISO_EN_OUT | rw | 0x00 | 1: SPIM MISO enabled as output |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 16 | | | | SW control for SPIM_NSS ECS when SPIM_SW_ENABLE='1' |
| | SPIM_NSS_EN_OUT | Rw | 0x00 | 1: SPIM NSS enabled as output |
| 15 | | | | SW control for SPIM_SCK ECS when SPIM_SW_ENABLE='1' |
| | SPIM_SCK_EN_OUT | rw | 0x00 | 1: SPIM CLK enabled as output |
| 14 | | | | SW control for SPIM_MOSI ENZI (ENZI = ~EN_IN) when SPIM_SW_ENABLE='1' |
| | SPIM_MOSI_EN_IN | rw | 0x00 | 1: SPIM MOSI enabled as input |
| 13 | | | | SW control for SPIM_MISO ENZI (ENZI = ~EN_IN) when SPIM_SW_ENABLE='1' |
| | SPIM_MISO_EN_IN | rw | 0x00 | 1: SPIM MISO enabled as input |
| 12 | | | | SW control for SPIM_NSS ENZI (ENZI = ~EN_IN) when SPIM_SW_ENABLE='1' |
| | SPIM_NSS_EN_IN | rw | 0x00 | 1: SPIM NSS enabled as input |
| 11 | | | | SW control for SPIM_SCK ENZI (ENZI = ~EN_IN) when SPIM_SW_ENABLE='1' |
| | SPIM_SCK_EN_IN | rw | 0x00 | 1: SPIM CLK enabled as input |
| 10:9 | | | | SW control for SPIM_MISO EPUD/EPD when SPIM_SW_ENABLE='1' |
| | | | | 10: SPIM MOSI enabled as pull up |
| | SPIM_MISO_EPUD | rw | 0x00 | 11: SPIM MOSI enabled as pull down |
| 8:7 | | | | SW control for SPIM_MOSI EPUD/EPD when SPIM_SW_ENABLE='1' |
| | | | | 10: SPIM MISO enabled as pull up |
| | SPIM_MOSI_EPUD | rw | 0x00 | 11: SPIM MISO enabled as pull down |
| 6:5 | | | | SW control for SPIM_NSS EPUD/EPD when SPIM_SW_ENABLE='1' |
| | | | | 10: SPIM NSS enabled as Pull up |
| | SPIM_NSS_EPUD | rw | 0x02 | 11: SPIM NSS enabled as Pull down |
| 4:3 | | | | SW control for SPIM_SCK EPUD/EPD when SPIM_SW_ENABLE='1' |
| | | | | 10: SPIM CLK enabled as pull up |
| | SPIM_SCK_EPUD | rw | 0x00 | 11: SPIM CLK enabled as pull down |
| 2 | | | | Select driver strength for SPIM pads |
| | EHS | rw | 0x00 | 1: Enable slew (for high speed modes) for SPIM pads |
| 1 | SSUP | rw | 0x00 | supply voltage for SPIM pads |
| 0 | SPIM_SW_ENABLE | rw | 0x00 | 1: Enables Pad configuration for SPI master |

**Table 114. PCR_CTIF_REG (address offset 0xB8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | rw | 0x00 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 0 | | | | Value of ct_pres_enable_pup_i coming from CTIF is latched and is used to detect card activity during standby and suspend modes if CTIF is enabled as wakeup source |
| | CT_ENABLE_PUP | r- | 0x00 | 1: CT pres pull up enabled |

**Table 115. PCR_HOSTIF_SAVE1_REG (address offset 0xBC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:11 | RESERVED | rw | 0x00 | Reserved |
| 10 | HSU_TX_DIVIDER | rw | 0x00 | TX divider save (only save reg for stby) |
| 9:0 | HSU_RX_DIVIDER | rw | 0x00 | RX Divider save (only save reg for stby) |

**Table 116. PCR_HOSTIF_SAVE2_REG (address offset 0xC0)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | rw | 0x00 | Reserved |
| 23:11 | HSU_TX_CLK_CORRECT | rw | 0x00 | clock correction for TX (only save reg for stby) |
| 10:0 | HSU_RX_CLK_CORRECT | rw | 0x00 | Clock correction for RX (only save reg for stby) |

**Table 117. PCR_TXLDO_MON_REG (address offset 0xC4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | rw | 0x00 | Reserved |
| 3 | WELL_MNGT | rw | 0x01 | TXLDO well mngt |
| 2 | | | | Input of the 5 V monitor.<br>1: TVDD is the source of 5 V monitor |
| | SRC_5V_MONITOR | rw | 0x00 | 0: VUP_TX is the source of 5 V monitor |
| 1 | | | | Comparator threshold selector |
| | THRES_5V_MONITOR | rw | 0x00 | 1: Threshold of 5 V Monitor is 5 V |
| 0 | | | | 1: Enable the 5 V monitor on TVDD |
| | EN_5V_MONITOR | rw | 0x00 | 0: Disable the 5 V monitor on TVDD |

**Table 118. PCR_BOOT2_REG (address offset 0xC8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | SPARE_CELL3 | rw | 0x00 | Third set of spare cells. |
| 23 | BOOT_REASON_ACTIVE_HPD | r- | 0x00 | 1: Boot because of coming out of ACTIVE_HPD |
| 22 | BOOT_REASON_VBUS_LOW | r- | 0x00 | 1: Boot because of VBUS going low in suspend or standby. |
| 21:0 | SPARE_CELL2 | rw | 0x00000 | Second set of space cells |

**Table 119. PCR_GPREG3_REG (address offset 0xCC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG3 | rw | 0x00 | general-purpose register for SW |

**Table 120. PCR_GPREG4_REG (address offset 0xD0)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG4 | rw | 0x00 | general-purpose register for SW |

**Table 121. PCR_GPREG5_REG (address offset 0xD4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG5 | rw | 0x00 | general-purpose register for SW |

**Table 122. PCR_GPREG6_REG (address offset 0xD8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG6 | rw | 0x00 | general-purpose register for SW |

**Table 123. PCR_GPREG7_REG (address offset 0xDC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | PCR_GPREG7 | rw | 0x00 | general-purpose register for SW |

**Table 124. PCR_GPIO_INT_ACTIVE_LOW_REG (address offset 0xE0)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | rw | 0x00 | Reserved |
| 11 | GPIO12_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO12 interrupts are active low/falling edge.<br>0 - Active_high/Rising edge<br>1 - Active_low/falling edge. |
| 10 | GPIO11_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO11 interrupts are active low/falling edge.<br>0 - Active_high/Rising edge;<br>1 - Active_low/falling edge. |
| 9 | GPIO10_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO10 interrupts are active low/falling edge.<br>0 - Active_high/Rising edge;<br>1: Active_low/falling edge. |
| 8 | GPIO9_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO9 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1: Active_low/falling edge. |
| 7 | GPIO8_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO8 interrupts are active low/falling edge.<br>0 - Active_high/Rising edge; 1 - Active_low/falling edge. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 6 | GPIO7_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO7 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1 - Active_low/falling edge. |
| 5 | GPIO6_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO6 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1 - Active_low/falling edge. |
| 4 | GPIO5_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO5 interrupts are active low/falling edge.<br>0 - Active_high/Rising edge;<br>1: Active_low/falling edge. |
| 3 | GPIO4_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO4 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1: Active_low/falling edge. |
| 2 | GPIO3_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO3 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1: Active_low/falling edge. |
| 1 | GPIO2_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO2 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1Active_low/falling edge. |
| 0 | GPIO1_INTR_ACTIVE_LOW | rw | 0x00 | indicates if GPIO1 interrupts are active low/falling edge.<br>0: Active_high/Rising edge;<br>1: Active_low/falling edge. |

**Table 125. PCR_GPIO_INT_LEVEL_SENSE_REG (address offset 0xE4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | rw | 0x00 | Reserved |
| 11 | GPIO12_INTR_LEVEL_SENSITIVE | rw | 0x00 | indicates if GPIO12 interrupts are level sensitive/edge sensitive.<br>0: Edge sensitive;<br>1: Level sensitive. |
| 10 | GPIO11_INTR_LEVEL_SENSITIVE | rw | 0x00 | indicates if GPIO11 interrupts are level sensitive/edge sensitive.<br>0: Edge sensitive;<br>1: Level sensitive. |
| 9 | GPIO10_INTR_LEVEL_SENSITIVE | rw | 0x00 | indicates if GPIO10 interrupts are level sensitive/edge sensitive.<br>0: Edge sensitive;<br>1: Level sensitive. |
| 8 | GPIO9_INTR_LEVEL_SENSITIVE | rw | 0x00 | indicates if GPIO9 interrupts are level sensitive/edge sensitive.<br>0: Edge sensitive;<br>1: Level sensitive. |
| 7 | GPIO8_INTR_LEVEL_SENSITIVE | rw | 0x00 | indicates if GPIO8 interrupts are level sensitive/edge sensitive. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 0: Edge sensitive;<br>1 - Level sensitive. |
| 6 | | | | indicates if GPIO7 interrupts are level sensitive/edge sensitive. |
| | GPIO7_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |
| 5 | | | | indicates if GPIO6 interrupts are level sensitive/edge sensitive. |
| | GPIO6_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |
| 4 | | | | indicates if GPIO5 interrupts are level sensitive/edge sensitive. |
| | GPIO5_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |
| 3 | | | | indicates if GPIO4 interrupts are level sensitive/edge sensitive. |
| | GPIO4_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |
| 2 | | | | indicates if GPIO3 interrupts are level sensitive/edge sensitive. |
| | GPIO3_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |
| 1 | | | | indicates if GPIO2 interrupts are level sensitive/edge sensitive. |
| | GPIO2_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |
| 0 | | | | indicates if GPIO1 interrupts are level sensitive/edge sensitive. |
| | GPIO1_INTR_LEVEL_SENSITIVE | rw | 0x00 | 0: Edge sensitive;<br>1: Level sensitive. |

**Table 126. PCR_GPIO_INT_ACTIVE_BOH_EDGE_REG (address offset 0xE8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | rw | 0x00 | Reserved |
| 11 | | | | indicates if GPIO12 interrupts are both positive and negative edge triggered. |
| | GPIO12_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | 0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 10 | | | | indicates if GPIO11 interrupts are both positive and negative edge triggered. |
| | GPIO11_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | 0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 9 | GPIO10_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO10 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 8 | GPIO9_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO9 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 7 | GPIO8_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO8 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 6 | GPIO7_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO7 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 5 | GPIO6_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO6 interrupts are both positive and negative edge triggered.<br>0:  Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 4 | GPIO5_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO5 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 3 | GPIO4_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO4 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 2 | GPIO3_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO3 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 1 | GPIO2_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO2 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |
| 0 | GPIO1_INTR_BOTH_EDGE_SENSITIVE | rw | 0x00 | indicates if GPIO1 interrupts are both positive and negative edge triggered.<br>0: Not both edge triggered;<br>1: Both positive edge and negative edge triggered. |

**Table 127. PCR_SELECT_SYSTEMCLOCK_REG (address offset 0xEC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:3 | RESERVED | rw | 0x00 | Reserved |
| 2:0 | | | | Selects the clock source<br>001: clkHFO (20 MHz)<br>010: clkUSBPLL/2 (24 MHz) (internal test purpose)<br>100: clkXtal (27.12 MHz) (internal test purpose) |
| | SELECT_SYSTEMCLOCK | rw | 0x01 | Others: INVALID |

**Table 128. PCR_ADV_RFLD_REG (address offset 0xF0)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | rw | 0x00 | Reserved |
| 24:20 | ADV_RFLD_IBIAS_TIME | rw | 0x00 | Ibias Boot up Time |
| 19 | ADV_RFLD_BYPASS_FREQ_CHECK | rw | 0x00 | bypass the digital frequency check in case of malfunction of frequency check<br>1: Disable Frequency check during RF level detection |
| 18 | ADV_RFLD_BYPASS | rw | 0x00 | enable Bypass for pre-amplifier. To be set if normal RF Level detector is used for<br>1: Disable Pre-Amplifier |
| 17:16 | ADV_RFLD_PREAMP_GAIN | rw | 0x00 | gain setting for pre-amplifier |
| 15:12 | ADV_RFLD_CLKREC_TIME | rw | 0x00 | boot up time for clock recovery. Defines time between switching on the clock recovery until frequency check is enabled |
| 11:7 | ADV_RFLD_WAIT_TIME | rw | 0x00 | boot up time for advanced RF level detector. Defines time between switching on the RF level detector until field is checked. |
| 6:0 | ADV_RFLD_SLEEP_TIME | rw | 0x00 | defines the sleep Time (# LFO cycles) in the polling circuit where the Advanced RF Level Detector is switched off. |

**Table 129. PCR_ADV_RFLD_TEST_REG (address offset 0xF4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:3 | RESERVED | rw | 0x00 | Reserved |
| 2 | ADV_RFLD_TEST_BYPASS_FIELDDET | rw | 0x00 | bypass the field detection during SLEEP time<br>1: Bypass field detection during SLEEP time<br>0: Field detection enabled during SLEEP time |
| 1 | ADV_RFLD_TEST_PREAMP_ENABLE | rw | 0x00 | Enables the advanced RF level detector during standby mode. The RF level detector as wake-up source must be enabled to use this feature.<br>1: Enable Pre-amplifier<br>0: Disable Pre-amplifier for RF level detection |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 0 | ADV_RFLD_TEST_ENABLE | rw | 0x00 | Enable manual setting of advanced RF level detection.<br>1: Enable manual setting of RF level detection<br>0: Disable Manual setting of RF level detection |

**Table 130. PCR_INT_CLR_ENABLE_REG (address offset 0x3FD8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | -x | 0x00 | Reserved |
| 27 | GPIO12_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO12 pad going high interrupt clear enable<br>1: Disable GPIO12 interrupt |
| 26 | GPIO11_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO11 pad going high interrupt clear enable<br>1: Disable GPIO11 interrupt |
| 25 | GPIO10_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO10 pad going high interrupt clear enable<br>1: Disable GPIO10 interrupt |
| 24 | GPIO9_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO9 pad going high interrupt clear enable<br>1: Disable GPIO9 interrupt |
| 23 | GPIO8_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO8 pad going high interrupt clear enable<br>1: Disable GPIO8 interrupt |
| 22 | GPIO7_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO7 pad going high interrupt clear enable<br>1: Disable GPIO7 interrupt |
| 21 | GPIO6_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO6 pad going high interrupt clear enable<br>1: Disable GPIO6 interrupt |
| 20 | GPIO5_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO5 pad going high interrupt clear enable<br>1: Disable GPIO5 interrupt |
| 19 | GPIO4_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO4 pad going high interrupt clear enable<br>1: Disable GPIO4 interrupt |
| 18 | GPIO3_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO3 pad going high interrupt clear enable<br>1: Disable GPIO3 interrupt |
| 17 | GPIO2_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO2 pad going high interrupt clear enable<br>1: Disable GPIO2 interrupt |
| 16 | GPIO1_PAD_HIGH_INT_CLR_ENABLE | -x | 0x00 | GPIO1 pad going high interrupt clear enable<br>1: Disable GPIO1 interrupt |
| 15:11 | RESERVED | -x | 0x00 | Reserved |
| 10 | VBUSP_MON_HIGH_INT_CLR_ENABLE | -x | 0x00 | VBUSP monitor going high interrupt clear enable<br>1: Disable VBUSP monitor high interrupt |
| 9 | CLIF_RFLD_ACT_INT_CLR_ENABLE | -x | 0x00 | Clif RF field activity observed interrupt clear enable<br>1: Disable RF Level detected interrupt |
| 8 | VBUS_MON_LOW_INT_CLR_ENABLE | -x | 0x00 | VBUS monitor going low interrupt clear enable<br>1: Disable VBUS monitor low interrupt |
| 7 | VBUSP_MON_LOW_INT_CLR_ENABLE | -x | 0x00 | VBUSP monitor going low interrupt clear enable |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | 1: Disable VBUSP monitor low interrupt |
| 6 | PVDD_CURLIM_ACT_CLR_ENABLE | -x | 0x00 | PVDD current limiter active interrupt clear enable.<br>1: Disable PVDD current limiter interrupt |
| 5 | TEMPERROR1_INT_CLR_ENABLE | -x | 0x00 | Temperature error 1 interrupt clear enable<br>1: Disable temperature sensor 1 error interrupt |
| 4 | TEMPERROR0_INT_CLR_ENABLE | -x | 0x00 | Temperature error 0 interrupt clear enable<br>1: Disable temperature sensor 0 error interrupt |
| 3 | SUSPEND_DEFAULTED_INT_CLR_ENABLE | -x | 0x00 | Suspend defaulted interrupt clear enable<br>1: Disable suspend prevented interrupt |
| 2 | RESERVED | -x | 0x00 | Reserved |
| 1 | SUSPEND_INT_CLR_ENABLE | -x | 0x00 | suspend interrupt clear enable.<br>1: Disable suspend(exit) interrupt |
| 0 | GPIO_INT_CLR_ENABLE | -x | 0x00 | GPIO interrupt clear enable<br>1: Disable GPIO interrupt |

**Table 131. PCR_INT_SET_ENABLE_REG (address offset 0x3FDC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | -x | 0x00 | Reserved |
| 27 | GPIO12_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO12 pad going high interrupt set enable<br>1: Enable GPIO12 interrupt |
| 26 | GPIO11_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO11 pad going high interrupt set enable<br>1: Enable GPIO11 interrupt |
| 25 | GPIO10_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO10 pad going high interrupt set enable<br>1: Enable GPIO10 interrupt |
| 24 | GPIO9_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO9 pad going high interrupt set enable<br>1: Enable GPIO9 interrupt |
| 23 | GPIO8_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO8 pad going high interrupt set enable<br>1: Enable GPIO8 interrupt |
| 22 | GPIO7_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO7 pad going high interrupt set enable<br>1: Enable GPIO7 interrupt |
| 21 | GPIO6_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO6 pad going high interrupt set enable<br>1: Enable GPIO6 interrupt |
| 20 | GPIO5_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO5 pad going high interrupt set enable<br>1: Enable GPIO5 interrupt |
| 19 | GPIO4_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO4 pad going high interrupt set enable<br>1: Enable GPIO4 interrupt |
| 18 | GPIO3_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO3 pad going high interrupt set enable<br>1: Enable GPIO3 interrupt |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17 | GPIO2_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO2 pad going high interrupt set enable<br>1: Enable GPIO2 interrupt |
| 16 | GPIO1_PAD_HIGH_INT_SET_ENABLE | -x | 0x00 | GPIO1 pad going high interrupt set enable<br>1: Enable GPIO1 interrupt |
| 15:11 | RESERVED | -x | 0x00 | Reserved |
| 10 | VBUSP_MON_HIGH_INT_SET_ENABLE | -x | 0x00 | VBUSP monitor going high interrupt set enable<br>1: Enable VBUSP Monitor high interrupt |
| 9 | CLIF_RFLD_ACT_INT_SET_ENABLE | -x | 0x00 | Clif RF field activity observed interrupt set enable<br>1: Enable RF level detected interrupt |
| 8 | VBUS_MON_LOW_INT_SET_ENABLE | -x | 0x00 | VBUS monitor going low interrupt set enable<br>1 – Enable VBUS monitor low interrupt |
| 7 | VBUSP_MON_LOW_INT_SET_ENABLE | -x | 0x00 | VBUSP monitor going low interrupt set enable<br>1 – Enable VBUSP monitor low interrupt |
| 6 | PVDD_CURLIM_ACT_SET_ENABLE | -x | 0x00 | PVDD current limiter active interrupt set enable.<br>1 – Enable PVDD current limiter interrupt |
| 5 | TEMPERROR1_INT_SET_ENABLE | -x | 0x00 | Temperature error 1 interrupt set enable<br>1 – Enable temperature sensor 1 error interrupt |
| 4 | TEMPERROR0_INT_SET_ENABLE | -x | 0x00 | Temperature error 0 interrupt set enable<br>1 – Enable temperature sensor 0 error interrupt |
| 3 | SUSPEND_DEFAULTED_INT_SET_ENABLE | -x | 0x00 | Suspend defaulted interrupt set enable<br>1: Enable suspend prevented interrupt |
| 2 | RESERVED | rw | 0x00 | Reserved |
| 1 | SUSPEND_INT_SET_ENABLE | -x | 0x00 | suspend interrupt set enable.<br>1: Enable suspend(exit) interrupt |
| 0 | GPIO_INT_SET_ENABLE | -x | 0x00 | GPIO interrupt clear enable<br>1: Enable GPIO interrupt |

**Table 132. PCR_INT_STATUS_REG (address offset 0x3FE0)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | rw | 0x00 | Reserved |
| 27 | GPIO12_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO12 pad going high interrupt status<br>1: Indicates GPIO12 interrupt is set. |
| 26 | GPIO11_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO11 pad going high interrupt status<br>1: Indicates GPIO11 interrupt is set. |
| 25 | GPIO10_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO10 pad going high interrupt status<br>1: Indicates GPIO10 interrupt is set. |
| 24 | GPIO9_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO9 pad going high interrupt status<br>1: Indicates GPIO9 interrupt is set. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 23 | GPIO8_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO8 pad going high interrupt status<br>1: Indicates GPIO8 interrupt is set. |
| 22 | GPIO7_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO7 pad going high interrupt status<br>1: Indicates GPIO7 interrupt is set. |
| 21 | GPIO6_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO6 pad going high interrupt status<br>1: Indicates GPIO6 interrupt is set. |
| 20 | GPIO5_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO5 pad going high interrupt status<br>1: Indicates GPIO5 interrupt is set. |
| 19 | GPIO4_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO4 pad going high interrupt status<br>1: Indicates GPIO4 interrupt is set. |
| 18 | GPIO3_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO3 pad going high interrupt status<br>1: Indicates GPIO3 interrupt is set. |
| 17 | GPIO2_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO2 pad going high interrupt status<br>1: Indicates GPIO2 interrupt is set. |
| 16 | GPIO1_PAD_HIGH_INT_STATUS | r- | 0x00 | GPIO1 pad going high interrupt status<br>1: Indicates GPIO1 interrupt is set. |
| 15:11 | RESERVED | r- | 0x00 | Reserved |
| 10 | VBUSP_MON_HIGH_INT_STATUS | r- | 0x00 | VBUSP monitor going high interrupt status<br>1: Indicates VBUSP monitor high interrupt is set. |
| 9 | CLIF_RFLD_ACT_INT_STATUS | r- | 0x00 | Clif RF field activity observed interrupt status<br>1: Indicates RF Level detected interrupt is set. |
| 8 | VBUS_MON_LOW_INT_STATUS | r- | 0x00 | VBUS monitor going low interrupt clear status<br>1: Indicates VBUS monitor low interrupt is set. |
| 7 | VBUSP_MON_LOW_INT_STATUS | r- | 0x00 | VBUSP monitor going low interrupt clear status<br>1: Indicates VBUSP monitor low interrupt is set. |
| 6 | PVDD_CURLIM_ACT_INT_STATUS | r- | 0x00 | PVDD current limiter active interrupt status<br>1: Indicates PVDD current limiter interrupt is set. |
| 5 | TEMPERROR1_INT_STATUS | r- | 0x00 | Temperature error 1 interrupt status<br>1: Indicates temperature sensor 1 error interrupt is set. |
| 4 | TEMPERROR0_INT_STATUS | r- | 0x00 | Temperature error 0 interrupt status<br>1: Indicates temperature sensor 0 error interrupt is set. |
| 3 | SUSPEND_DEFAULTED_INT_STATUS | r- | 0x00 | Suspend defaulted interrupt status<br>1: Indicates suspend prevented interrupt is set. |
| 2 | RESERVED | r- | 0x00 | Reserved |
| 1 | SUSPEND_INT_STATUS | r- | 0x00 | Suspend interrupt status.<br>1: Indicates suspend (exit) interrupt is set. |
| 0 | GPIO_INT_STATUS | r- | 0x00 | GPIO interrupt clear enable<br>1: Indicates GPIO interrupt is set. |

**Table 133. PCR_INT_ENABLE_REG (address offset 0x3FE4)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | r- | 0x00 | Reserved |
| 27 | GPIO12_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO12 pad going high interrupt enable<br>1: Indicates GPIO12 interrupt enabled. |
| 26 | GPIO11_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO11 pad going high interrupt enable<br>1: Indicates GPIO11 interrupt enabled. |
| 25 | GPIO10_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO10 pad going high interrupt enable<br>1: Indicates GPIO10 interrupt enabled. |
| 24 | GPIO9_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO9 pad going high interrupt enable<br>1: Indicates GPIO9 interrupt enabled. |
| 23 | GPIO8_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO8 pad going high interrupt enable<br>1: Indicates GPIO8 interrupt enabled. |
| 22 | GPIO7_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO7 pad going high interrupt enable<br>1: Indicates GPIO7 interrupt enabled. |
| 21 | GPIO6_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO6 pad going high interrupt enable<br>1: Indicates GPIO6 interrupt enabled. |
| 20 | GPIO5_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO5 pad going high interrupt enable<br>1: Indicates GPIO5 interrupt enabled. |
| 19 | GPIO4_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO4 pad going high interrupt enable<br>1: Indicates GPIO4 interrupt enabled. |
| 18 | GPIO3_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO3 pad going high interrupt enable<br>1: Indicates GPIO3 interrupt enabled. |
| 17 | GPIO2_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO2 pad going high interrupt enable<br>1: Indicates GPIO2 interrupt enabled. |
| 16 | GPIO1_PAD_HIGH_INT_ENABLE | r- | 0x00 | GPIO1 pad going high interrupt enable<br>1: Indicates GPIO1 interrupt enabled. |
| 15:11 | RESERVED | r- | 0x00 | Reserved |
| 10 | VBUSP_MON_HIGH_INT_ENABLE | r- | 0x00 | VBUSP monitor going high interrupt enable<br>1: Indicates VBUSP monitor high interrupt enabled. |
| 9 | CLIF_RFLD_ACT_INT_ENABLE | r- | 0x00 | Clif RF field activity observed interrupt enable<br>1: Indicates RF level detected interrupt enabled. |
| 8 | VBUS_MON_LOW_INT_ENABLE | r- | 0x00 | VBUS monitor going low interrupt enable<br>1: Indicates VBUS monitor low interrupt enabled. |
| 7 | VBUSP_MON_LOW_INT_ENABLE | r- | 0x00 | VBUSP monitor going low interrupt enable<br>1: Indicates VBUSP monitor low interrupt enabled. |
| 6 | PVDD_CURLIM_ACT_INT_ENABLE | r- | 0x00 | PVDD current limiter active interrupt enable<br>1: Indicates PVDD current limiter interrupt enabled. |
| 5 | TEMPERROR1_INT_ENABLE | r- | 0x00 | Temperature error 1 interrupt enable<br>1: Indicates temperature sensor 1 error interrupt enabled. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 4 | TEMPERROR0_INT_ENABLE | r- | 0x00 | Temperature error 0 interrupt enable<br>1: Indicates temperature sensor 0 error interrupt enabled. |
| 3 | SUSPEND_DEFAULTED_INT_ENABLE | r- | 0x00 | Suspend defaulted interrupt enable<br>1: Indicates suspend prevented interrupt enabled. |
| 2 | RESERVED | r- | 0x00 | Reserved |
| 1 | SUSPEND_INT_ENABLE | r- | 0x00 | Suspend interrupt enable.<br>1: Indicates suspend (exit) interrupt enabled. |
| 0 | GPIO_INT_ENABLE | r- | 0x00 | GPIO interrupt clear enable<br>1: Indicates GPIO interrupt enabled. |

**Table 134. PCR_INT_CLR_STATUS_REG (address offset 0x3FE8)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | -x | 0x00 | Reserved |
| 27 | GPIO12_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO12 pad going high interrupt clear status<br>1: Clear GPIO12 interrupt. Auto clear after 2 cycles. |
| 26 | GPIO11_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO11 pad going high interrupt clear status<br>1: Clear GPIO11 interrupt. Auto clear after 2 cycles. |
| 25 | GPIO10_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO10 pad going high interrupt clear status<br>1: Clear GPIO10 interrupt. Auto clear after 2 cycles. |
| 24 | GPIO9_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO9 pad going high interrupt clear status<br>1: Clear GPIO9 interrupt. Auto clear after 2 cycles. |
| 23 | GPIO8_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO8 pad going high interrupt clear status<br>1: Clear GPIO8 interrupt. Auto clear after 2 cycles. |
| 22 | GPIO7_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO7 pad going high interrupt clear status<br>1: Clear GPIO7 interrupt. Auto clear after 2 cycles. |
| 21 | GPIO6_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO6 pad going high interrupt clear status<br>1: Clear GPIO6 interrupt. Auto clear after 2 cycles. |
| 20 | GPIO5_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO5 pad going high interrupt clear status<br>1: Clear GPIO5 interrupt. Auto clear after 2 cycles. |
| 19 | GPIO4_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO4 pad going high interrupt clear status<br>1: Clear GPIO4 interrupt. Auto clear after 2 cycles. |
| 18 | GPIO3_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO3 pad going high interrupt clear status<br>1: Clear GPIO3 interrupt. Auto clear after 2 cycles. |
| 17 | GPIO2_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO2 pad going high interrupt clear status<br>1: Clear GPIO2 interrupt. Auto clear after 2 cycles. |
| 16 | GPIO1_PAD_HIGH_INT_CLR_STATUS | -x | 0x00 | GPIO1 pad going high interrupt clear status<br>1: Clear GPIO1 interrupt. Auto clear after 2 cycles. |
| 15:11 | RESERVED | -x | 0x00 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 10 | VBUSP_MON_HIGH_INT_CLR_STATUS | -x | 0x00 | VBUSP monitor going high interrupt clear status<br>1: Clear VBUSP monitor high interrupt. Auto clear after 2 cycles. |
| 9 | CLIF_RFLD_ACT_INT_CLR_STATUS | -x | 0x00 | Clif RF field activity observed interrupt clear status<br>1: Clear RF level detected interrupt. Auto clear after 2 cycles. |
| 8 | VBUS_MON_LOW_INT_CLR_STATUS | -x | 0x00 | VBUS monitor going low interrupt clear status<br>1: Clear VBUS monitor low interrupt. Auto clear after 2 cycles. |
| 7 | VBUSP_MON_LOW_INT_CLR_STATUS | -x | 0x00 | VBUSP monitor going low interrupt clear status<br>1: Clear VBUSP monitor low interrupt. Auto clear after 2 cycles. |
| 6 | PVDD_CURLIM_ACT_CLR_STATUS | -x | 0x00 | PVDD current limiter active interrupt clear status<br>1: Clear PVDD current limiter interrupt. Auto clear after 2 cycles. |
| 5 | TEMPERROR1_INT_CLR_STATUS | -x | 0x00 | Temperature error 1 interrupt clear status<br>1: Clear temperature sensor 1 error interrupt. Auto clear after 2 cycles. |
| 4 | TEMPERROR0_INT_CLR_STATUS | -x | 0x00 | Temperature error 0 interrupt clear status<br>1: Clear temperature sensor 0 error interrupt. Auto clear after 2 cycles. |
| 3 | SUSPEND_DEFAULTED_INT_CLR_STATUS | -x | 0x00 | Suspend defaulted interrupt clear status<br>1: Clear suspend prevented interrupt. Auto clear after 2 cycles. |
| 2 | RESERVED | -x | 0x00 | Reserved |
| 1 | SUSPEND_INT_CLR_STATUS | -x | 0x00 | Suspend interrupt clear status.<br>1: Clear suspend (exit) interrupt. Auto clear after 2 cycles. |
| 0 | GPIO_INT_CLR_STATUS | -x | 0x00 | GPIO interrupt clear clear status<br>1: Clear GPIO interrupt. Auto clear after 2 cycles. |

**Table 135. PCR_INT_SET_STATUS_REG (address offset 0x3FEC)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | -x | 0x00 | Reserved |
| 27 | GPIO12_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO12 pad going high interrupt set status<br>1: Set GPIO12 interrupt. Auto clear after 2 cycles. |
| 26 | GPIO11_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO11 pad going high interrupt set status<br>1: Set GPIO11 interrupt. Auto clear after 2 cycles. |
| 25 | GPIO10_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO10 pad going high interrupt set status<br>1: Set GPIO10 interrupt. Auto clear after 2 cycles. |
| 24 | GPIO9_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO9 pad going high interrupt set status<br>1: Set GPIO9 interrupt. Auto clear after 2 cycles. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 23 | GPIO8_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO8 pad going high interrupt set status<br>1: Set GPIO8 interrupt. Auto clear after 2 cycles. |
| 22 | GPIO7_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO7 pad going high interrupt set status<br>1: Set GPIO7 interrupt. Auto clear after 2 cycles. |
| 21 | GPIO6_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO6 pad going high interrupt set status<br>1: Set GPIO6 interrupt. Auto clear after 2 cycles. |
| 20 | GPIO5_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO5 pad going high interrupt set status<br>1: Set GPIO5 interrupt. Auto clear after 2 cycles. |
| 19 | GPIO4_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO4 pad going high interrupt set status<br>1: Set GPIO4 Interrupt. Auto clear after 2 cycles. |
| 18 | GPIO3_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO3 pad going high interrupt set status<br>1: Set GPIO3 interrupt. Auto clear after 2 cycles. |
| 17 | GPIO2_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO2 pad going high interrupt set status<br>1: Set GPIO2 interrupt. Auto clear after 2 cycles. |
| 16 | GPIO1_PAD_HIGH_INT_SET_STATUS | -x | 0x00 | GPIO1 pad going high interrupt set status<br>1: Set GPIO1 interrupt. Auto clear after 2 cycles. |
| 15:11 | RESERVED | -x | 0x00 | Reserved |
| 10 | VBUSP_MON_HIGH_INT_SET_STATUS | -x | 0x00 | VBUSP monitor going high interrupt set status<br>1: Set VBUSP monitor high interrupt. Auto clear after 2 cycles. |
| 9 | CLIF_RFLD_ACT_INT_SET_STATUS | -x | 0x00 | Clif RF field activity observed interrupt set status<br>1: Set RF level detected interrupt. Auto clear after 2 cycles. |
| 8 | VBUS_MON_LOW_INT_SET_STATUS | -x | 0x00 | VBUS monitor going low interrupt set status<br>1: Set VBUS monitor low interrupt. Auto clear after 2 cycles. |
| 7 | VBUSP_MON_LOW_INT_SET_STATUS | -x | 0x00 | VBUSP monitor going low interrupt set status<br>1: Set VBUSP monitor low interrupt. Auto clear after 2 cycles. |
| 6 | PVDD_CURLIM_ACT_SET_STATUS | -x | 0x00 | PVDD current limiter active interrupt set status.<br>1: Set PVDD current limiter interrupt. Auto clear after 2 cycles. |
| 5 | TEMPERROR1_INT_SET_STATUS | -x | 0x00 | Temperature error 1 interrupt set status<br>1: Set temperature sensor 1 error interrupt. Auto clear after 2 cycles. |
| 4 | TEMPERROR0_INT_SET_STATUS | -x | 0x00 | Temperature error 0 interrupt set status<br>1: Set temperature sensor 0 error interrupt. Auto clear after 2 cycles. |
| 3 | SUSPEND_DEFAULTED_INT_SET_STATUS | -x | 0x00 | Suspend defaulted interrupt set status<br>1: Set suspend prevented interrupt. Auto clear after 2 cycles. |
| 2 | RESERVED | -x | 0x00 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 1 | SUSPEND_INT_SET_STATUS | -x | 0x00 | Suspend interrupt set status.<br>1: Set suspend (exit) interrupt. Auto clear after 2 cycles. |
| 0 | GPIO_INT_SET_STATUS | -x | 0x00 | GPIO interrupt clear status<br>1: Set GPIO interrupt. Auto clear after 2 cycles. |

# 9. CRC

This block implements a configurable 16/32-bit parallel CRC.

The 16-bit CRC is compliant to X.25 (CRC-CCITT, ISO/IEC13239) standard with a generator polynomial of:

$$g(x) = x^{16} + x^{12} + x^5 + 1 \quad (1)$$

The 32-bit CRC is compliant to the Ethernet / AAL5 (IEEE 802.3) standard with a generator polynomial of:

$$g(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1 \quad (2)$$

**Note:** No final XOR is performed.

CRC calculation is performed in parallel, meaning that one CRC calculation is performed in one clock cycle.

The standard CRC32 polynomial passes the fips140-2 tests.

## 9.1 CRC features

- Configurable CRC preset value
- Selectable LSB or MSB first
- Calculation based on 32 bits, 16 bits, 8 bits words

## 9.2 Parallel CRC

In parallel CRC calculation, each bit of the new CRC depends on the previous bits of the CRC and current bits of the input data. Using mathematical assumptions, one equation can be calculated for each bit of the new CRC which are derived from the serial CRC implementation. Using these equations, a CRC can be calculated in one clock cycle whereas serial CRC takes as many clock cycles as the number of bits in the CRC.

The method used to generate the equations for each bit of the new CRC requires 4 steps and depends on the input data width (W) and polynomial width (C). Therefore, each CRC configuration (polynomial width and input data width) requires dedicated equations.

Below is one simple example to generate the equations for CRC4-ITU (x4+x+1; C=4) and input data width W=4. The same method applies to any values of P and W.

Let's call C_new(n) the nth bit of the new CRC value, C_prev(n) the nth bit of the previous calculated CRC value (e.g. preloaded value) and W(n) the nth bit of the input data.

Step1: Implement a standard serial CRC calculation:

C_new(0)=C_prev(3) xor W;

C_new(1)=C_prev(0) xor C_prev(3) xor W;

C_new(2)=C_prev(1);

C_new(3)=C_prev(2);

A full C_new value can be calculated running 4 times these equations.

Step2: Parallel CRC calculation is a function of input data (W(n)) and previous CRC value (C_prev). Calculate C_new using the previous serial equations as a function of W(n) for each one hot encoded value of W(n):

C_prev=0x0;

W is one hot encoded: W=4 so the values are 0x1, 0x2, 0x4, 0x8.

C_new(W=0x1;C_prev=0x0)=4'b0011

C_new(W=0x2;C_prev=0x0)=4'b0110

C_new(W=0x4;C_prev=0x0)=4'b1100

C_new(W=0x8;C_prev=0x0)=4'b1011

The hereabove results lead to the matrix M1 shown in table below

**Table 136. CRC4 M1**

| | C_prev(0) | C_prev(1) | C_prev(2) | C_prev(3) |
|---|---|---|---|---|
| W(0) | 0 | 0 | 1 | 1 |
| W(1) | 0 | 1 | 1 | 0 |
| W(2) | 1 | 1 | 0 | 0 |
| W(3) | 1 | 0 | 1 | 1 |

Step3: Parallel CRC calculation is a function of input data (W(n)) and previous CRC

value (C_prev). Calculate C_new using the previous serial equations as a function of W(n) for each one hot encoded value of C_prev(n):

W=0x0;

C_prev is one hot encoded: C=4 so the values are 0x1,0x2,0x4,0x8.

C_new(W=0x0;C_prev=0x1)=4'b0011

C_new(W=0x0;C_prev=0x4)=4'b1100

C_new(W=0x0;C_prev=0x8)=4'b1011

The hereabove results give the matrix M2 shown in table below

**Table 137. CRC4 M2**

| | C_prev(0) | C_prev(1) | C_prev(2) | C_prev(3) |
|---|---|---|---|---|
| W(0) | 0 | 0 | 1 | 1 |
| W(1) | 0 | 1 | 1 | 0 |
| W(2) | 1 | 1 | 0 | 0 |
| W(3) | 1 | 0 | 1 | 1 |

Step 4: Compute the two matrices to generate the four equations for each CRC_new bit:

C_new(0)=W(0) xor W(3) xor C_prev(0) xor C_prev(3)

C_new(1)=W(0) xor W(1) xor W(3) xor C_prev(0) xor C_prev(1) xor C_prev(3)

C_new(2)=W(1) xor W(2) xor C_prev(1) xor C_prev(2)

C_new(3)=W(2) xor W(3) xor C_prev(2) xor C_prev(3)

Now these equations are used to calculate the parallel CRC in only one clock cycle.

## 9.3 LSB and MSB first functionality

**In MSB first mode:**

- Input data is written in CRCDAT_xx MSB first
- CRCDAT_xx is pushed LSB first in the internal LFSR
- CRC preset value is written in CRCDAT_PRELOAD MSB first
- CRCDAT_PRELOAD is pushed MSB first in the internal LFSR
- CRC value is read MSB first on the APB interface.

**In LSB first mode:**

- Input data is written in CRCDAT_xx MSB first
- CRCDAT_xx is pushed LSB first in the internal LFSR
- CRC preset value is written in CRCDAT_PRELOAD MSB first
- CRCDAT_PRELOAD is pushed LSB first in the internal LFSR

- CRC value is read MSB first on the APB interface

**Note**: The same equations are used for LSB first and for MSB first. A simple serial shift is performed on the input data, preset value and calculated CRC to output the correct CRC value.

## 9.4 Typical Usage Example

For an input buffer size of 7 bytes and required output of 32-bit CRC with LSB first input, the CRC_CONTROL_REG is configured with 0x00000015. The first 4 bytes of the buffer is written to CRC_DAT32, the next 2 bytes is written to CRC_DAT16 and the last byte is written to CRC_DAT8. The CRCDAT_CALC is returned that contains the computed CRC

Alternatively, after writing 4 bytes of buffer to CRC_DAT32 and the next 3 bytes can be recursively written to CRC_DAT8 also.

### CRC-16 set-up

Polynomial = $x^{16} + x^{15} + x^2 + 1$

Seed Value = 0xFFFF

Bit order reverse for data input: YES

1's complement for data input: NO

Bit order reverse for CRC sum: YES

1's complement for CRC sum: NO

CRC_MODE = 0x0000 0015

CRC_SEED = 0x0000 0000

### CRC-32 set-up

Polynomial = $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

Seed Value = 0xFFFF FFFF

Bit order reverse for data input: YES

1's complement for data input: NO

Bit order reverse for CRC sum: YES

1's complement for CRC sum: YES

CRC_CONTROL_REG = 0x0000 0015

CRC_SEED = 0xFFFF FFFF

## 9.5 Register overview and description

### 9.5.1 Register overview

**Table 138. CRC register overview (base address 0x4000 C000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CRC_CONTROL_REG | 00h | 32 | R/W | 00000000h | CRC configuration register |
| CRC_DAT32_REG | 04h | 32 | R/W | 00000000h | CRC data register for 32 AMBA bitstream |
| CRC_DAT16_REG | 08h | 32 | R/W | 00000000h | CRC data register for 16 AMBA bitstream |
| CRC_DAT8_REG | 0Ch | 32 | R/W | 00000000h | CRC data register for 8 AMBA bitstream |
| CRC_DAT_CALC_REG | 10h | 32 | R | 0000FFFFh | CRC calculated value for CRC16, CRC32 |
| CRC_DAT_PRELOAD_REG | 14h | 32 | R/W | 00000000h | CRC preload value of CRC data |
| INTERNAL_USE | 18h | 8 | R/W | 00h | For internal use |

### 9.5.2 Register description

**Table 139. CRC_CONTROL_REG (address offset 0x00)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:7 | RESERVED | - | 0 | Reserved |
| 6:4 | CRCMOD | R/W | 0,1 for each bit | Defines which type of CRC (CRC16 or CRC32) should be calculated:<br>0: CRC16<br>1: CRC32<br>2: Reserved<br>3: Reserved<br>4: Reserved<br>5: Reserved<br>6: Reserved<br>7: Reserved |
| 3 | - | - | - | Reserved |
| 2 | CRCINV | R/W | 0,1 | Inverted input data<br>0: MSB first<br>1: LSB first |
| 1 | RESERVED | - | - | Reserved |
| 0 | CRC_ENABLE | R/W | 0,1 | 0: disables the CRC (enable clock gating)<br>1: enables the CRC (disable clock gating) |

**Table 140. CRC_DAT32_REG (address offset 0x04)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | CRCDAT32 | R/W | 0, 1 for each bit | 32 bits CRC value.<br>Writing to CRCDAT32 triggers a new CRC calculation over a 32 bits written word. Reading from CRCDAT32 returns the currently addressed CRC byte (see CRCCON). |

**Table 141. CRC_DAT16_REG (address offset 0x08)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | - | 0 | Reserved |
| 15:0 | CRCDAT16 | R/W | 0,1 for each bit | 16 bits CRC value.<br>Writing to CRCDAT16 triggers a new CRC calculation over the 16 bits written word. Reading from CRCDAT16 returns the currently addressed CRC byte (see CRCCON). |

**Table 142. CRC_DAT8_REG (address offset 0x0C)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7:0 | CRCDAT8 | R/W | 0,1 for each bit | 8 bits CRC value.<br>Writing to CRCDAT8 triggers a new CRC calculation over the 8 bits written word. Reading from CRCDAT8 returns the currently addressed CRC byte (see CRCCON). |

**Table 143. CRC_DAT_CALC_REG (address offset 0x10)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | CRCDAT_CALC | R | 0,1 for each bit | CRCDAT_CALC[31:0]: 32bits CRC calculated value in CRC32 mode.<br>CRCDAT_CALC[15:0]: 16bits CRC calculated value in CRC16 mode. |

**Table 144. CRC_DAT_PRELOAD_REG (address offset 0x14)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | CRCDAT_PRELOAD | R/W | 0,1 for each bit | 32bits CRC preload value.<br>Writing to CRCDAT_PRELOAD automatically preloads the CRC data register with the 32 bits written word. Reading from CRCDAT_PRELOAD returns the previously written preload value. Preloading the internal CRC shift register takes one clock cycle. |

# 10. Random number generator

The integrated RNG consists of two main parts: an analog True Random Number Generator (TRNG) connected to a digital Pseudo Random Number Generator (PRNG). The analog TRNG is used to load a new seed into the PRNG.

## 10.1 RNG features

- Combination of an analog TRNG and a digital PRNG
- 80-bit LFSR + Substitution-Box
- Dedicated clock for TRNG, PRNG runs on system clock.
- Data Ready Indicator bit
- Compliant to BSI AIS20 and SP800-22 (includes FIPS 140-2) RNG tests
- Connected to the CPU via an APB 3.0 bus

## 10.2 Functional description

The RNG module generates random numbers continuously at HFO speed. The RNG module can be stopped for lower consumption by disabling the HFO clock in the PCR or resetting the RNG_CONTROL_REG.rng_enable bit.

Main blocks (shown in Fig 22):

- RNG digital core (PRNG)
- RNG analog core (TRNG)
- APB Slave Interface and Register Bank

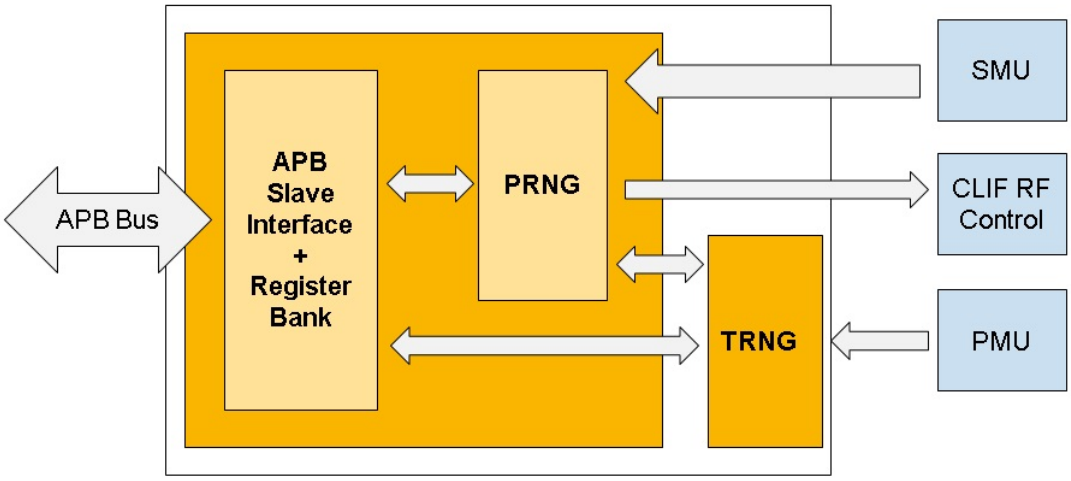



**Fig 22. Block diagram of RNG module**

### 10.2.1 PRNG digital core

The module delivers an 8-bit random number. The Pseudo Random Number Generator consists of an 80-bit LFSR with a feedback polynomial and an S-Box substitution for the output of the PRNG.

The feedback polynomial of the LFSR is:

$$x^{80} + x^{12} + x^{7} + x^{5} + 1 \quad (3)$$

The feedback polynomial is XOR with the TRNG input when a new seed is loaded. Some bits in the LFSR feeds an S-BOX lookup-table, taken from the DES Specification and the output of the SBOX is XOR with the feedback value from the LFSR.

### 10.2.2 RNG analog core (TRNG)

The basic function of TRNG is to provide a seed for the PRNG. The TRNG needs supply and references from the PMU. The TRNG consists of 2 comparators which monitor an analog signal to choose which offset has to be applied to this signal. The outputs of these 2 comparators are combined to generate the random number.

The TRNG generates an input random bit stream, which is xored to avoid continuous 0's or 1's as PRNG seed value.

## 10.3 Register overview and description

### 10.3.1 Random number generator register overview

**Table 145. Random number generator register overview (base address 0x4001 8000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| RNG_STATUS_REG | 00h | 16 | R | 8F00h | random number status register |
| RNG_CONTROL_REG | 04h | 16 | R/W | 2600h | random number control register |
| RESERVED | 08h | 4 | R/W | 0h | Reserved |

### 10.3.2 Register description

**Table 146. RNG_STATUS_REG**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | - | 0x00. | Reserved |
| 15:8 | rng | R | Ox8F | RNG number. Updated at each clock cycle but only considered random if rng_ready is high |
| 7:3 | RESERVED | - | 0x00. | Reserved |
| 2 | rng_ready | R | 0. | 1: a new RNG value is available.<br>0: the current value of RNG_STATUS_REG.rng is not random |
| 1 | rng_seed_error | R | 0 | 1: TRNG did not provide the random stream in time. Cleared with rng_enable rising edge.<br>0: If rng_seed_ready is high, TRNG is providing a random stream. If rng_seed_ready is low and rng_enable is high, TRNG is starting up. |
| 0 | Rng_seed_ready | R | 0 | 1: seeding process is done and PRNG is providing a new RNG value every cycle.<br>0: seeding process is ongoing |

**Table 147. RNG_CONTROL_REG**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | - | 0x00. | Reserved |
| 15:8 | trng_startup_time | R/W | 0x26 | Programmable wait time to release gated clocks feeding TRNG.An internal counter is started when RNG_CONTROL_REG.rng_enable is set, increments at the speed of clkHFO_div8, stops when it reaches the RNG_CONTROL_REG.trng_startup_time value and resets when RNG_CONTROL_REG.rng_enable is set low. Default value is 15.2us. |
| 7:1 | RESERVED | - | 0x00 | reserved |
| 0 | rng_enable | R/W | 0 | 1: enable PRNG (clkHFO is released)<br>0: disable PRNG (clkHFO is gated) |

# 11. General-purpose timers and watchdog timers

The PN7462AU includes two 12-bit general purpose timers (on LFO clock domain) with match capabilities two 32-bit general purpose timers (on HFO clock domain) and a Watch Dog Timer (WDT).

The purpose of the watchdog is to reset the microcontroller within a reasonable amount of time if it enters an erroneous state. When enabled, the watchdog will generate a system reset if the user program fails to 'feed' (or reload) the watchdog within a predetermined amount of time.

The timers and watchdog are software configurable via a 32-bit APB slave interface.

**Table 148. Timer characteristics**

| Name | Clock source | Frequency | Counter length (bit) | Resolution (ms) | Max delay(s) | Chaining possible |
|---|---|---|---|---|---|---|
| Timer 0 | LFO/2 | 190 KHz | 12 | 300 us | 1.2 | no |
| Timer 1 | LFO/2 | 190 KHz | 12 | 300 us | 1.2 | yes |
| Timer 2 | HFO | 20 MHz | 32 | 50 ns | 214 | no |
| Timer 3 | HFO | 20 MHz | 32 | 50 ns | 214 | no |
| Watchdog | LFO/128 | 2.96 KHz | 10 | 21.5 ms | 22 | no |

## 11.1 Timers features (Timers 0 and 1)

- Two 12-bit counters
- 1 match register per timer, no capture registers and capture trigger pins are needed
- 1 common output line gathering the four timer (Timer0, Timer1, Timer2, Timer3) Interrupts
- Timer0 and Timer1 can be concatenated (multiplied)
- Timer0 and Timer1 have two count modes: single-shot or free-running
- Timer0 and Timer1 timeout interrupts can be individually masked
- Timer0 and Timer1 Clock source is LFO clock

## 11.2 Timers features (Timer 2 and 3)

- Two 32-bit counters
- match register per timer, no capture registers and capture trigger pins are needed
- 1 common output line gathering the four timer (Timer0, Timer1, Timer2, Timer3) Interrupts
- Timer2 and Timer3 have two count modes: single-shot or free-running
- Timer2 and Timer3 timeout interrupts can be individually masked
- Timer2 and Timer3 clock source is HFO Clock

## 11.3 Watchdog features

- The watchdog has 10-bit counter

- The watchdog interrupt is connected to the ARM subsystem NMI (non-mask able Interrupt).
- The watchdog eventually sends a reset signal to the PCR to reset the chip if the threshold setting is not periodically reloaded by the FW.
- The watchdog reset is enabled by software, requires a system reset or a Watchdog reset/interrupt to be disabled. When enabled, The CPU must pulse the kick bit to reload the watchdog counter before it expires.

## 11.4 Register overview and description

### 11.4.1 Register overview

**Table 149. Timer register overview (base address 0x4001 C000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| TIMERS_TIMER0_CONTROL_REG | 0000h | 32 | R/W | 0000000h | Control of Timer 0 |
| TIMERS_TIMER0_TIMEOUT_REG | 0004h | 32 | R/W | 0000000h | Timeout value of Timer0 |
| TIMERS_TIMER0_COUNT_REG | 0008h | 32 | R | 0000000h | Current count value of Timer0 |
| TIMERS_TIMER1_CONTROL_REG | 000Ch | 32 | R/W | 0000000h | Control of Timer1 |
| TIMERS_TIMER1_TIMEOUT_REG | 0010h | 32 | R/W | 0000000h | Timeout value of Timer1 |
| TIMERS_TIMER1_COUNT_REG | 0014h | 32 | R | 0000000h | Current count value of Timer1 |
| TIMERS_TIMER2_CONTROL_REG | 0018h | 32 | R/W | 0000000h | Control of Timer 2 |
| TIMERS_TIMER2_TIMEOUT_REG | 001Ch | 32 | R/W | 0000000h | Timeout value of Timer2 |
| TIMERS_TIMER2_COUNT_REG | 0020h | 32 | R | 0000000h | Current count value of Timer2 |
| TIMERS_TIMER3_CONTROL_REG | 0024h | 32 | R/W | 0000000h | Control of Timer3 |
| TIMERS_TIMER3_TIMEOUT_REG | 0028h | 32 | R/W | 0000000h | Timeout value of Timer3 |
| TIMERS_TIMER3_COUNT_REG | 002Ch | 32 | R | 0000000h | Current count value of Timer3 |
| TIMERS_WDOG_CONTROL_REG | 0030h | 32 | R/W | 0000000h | Control of Watchdog Timer |
| TIMERS_WDOG_TIMEOUT_REG | 0034h | 32 | R/W | 0000000h | Timeout value of Watchdog Timer |
| TIMERS_WDOG_TRIGGER_INT_REG | 0038h | 32 | R/W | 0000000h | Count value of Watchdog Timer which triggers interrupt |
| TIMERS_WDOG_COUNT_REG | 003Ch | 32 | R | 0000000h | Current count value of Watchdog Timer |
| RESERVED | 0040h | 32 | R/W | 00000000h | Reserved |
| RESERVED | 0044h | 32 | R | 00000000h | Reserved |
| RESERVED | 0048h - 3FC8h | 32 | R | 00000000h | Reserved |
| TIMERS_WDOG_INT_STATUS_REG | 3FCCh | 32 | R | 00000000h | Watchdog interrupt status |
| TIMERS_WDOG_INT_CLR_STATUS_REG | 3FD0h | 32 | W | 00000000h | Watchdog clear interrupt |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| TIMERS_WDOG_INT_SET_STATUS_REG | 3FD4h | 32 | W | 00000000h | Watchdog set interrupt |
| TIMERS_INT_CLR_ENABLE_REG | 3FD8h | 32 | W | 00000000h | Timer clear interrupt enable |
| TIMERS_INT_SET_ENABLE_REG | 3FDCh | 32 | W | 00000000h | Timer set interrupt enable |
| TIMERS_INT_STATUS_REG | 3FE0h | 32 | R | 00000000h | Timer interrupt status |
| TIMERS_INT_ENABLE_REG | 3FE4h | 32 | R | 00000000h | Timer interrupt enable |
| TIMERS_INT_CLR_STATUS_REG | 3FE8h | 32 | W | 00000000h | Timer clear interrupt |
| TIMERS_INT_SET_STATUS_REG | 3FECh | 32 | W | 00000000h | Timer set interrupt |
| RESERVED | 3FF0h - 3FFCh | 32 | R | 00000000h | Reserved |

### 11.4.2 Register description

**Table 150. TIMERS_TIMER0_CONTROL_REG (address offset 0x0000)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | R | Reserved |
| 0 | TIMER0_MODE | 0 | R/W | 0 – single shot<br>1 – free running |

**Table 151. TIMERS_TIMER0_TIMEOUT_REG (address offset 0x0004)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | 0 | R | Reserved |
| 11:0 | TIMER0_TIMEOUT | 0 | R/W | Initial count value of Timer0 in step size of 0.30 ms[2]. If set to 0, this feature is disabled. |

**Table 152. TIMERS_TIMER0_COUNT_REG (address offset 0x0008)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | 0 | R | Reserved |
| 11:0 | TIMER0_COUNT | 0 | R | Current count value of Timer0 in step size of 0.30ms |

**Table 153. TIMERS_TIMER1_CONTROL_REG (address offset 0x000C)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:2 | RESERVED | 0 | R | Reserved |
| 1 | ENABLE_TIMER0_TRIGGER | 0 | R/W | 1- Timer1 will decrement once when Timer0 reaches its terminal count (assuming that field TIMER1_TIMEOUT is non-zero in register TIMERS_TIMER1_TIMEOUT)<br>0: Timer1 counts independently |

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 0 | TIMER1_MODE | 0 | R/W | 0: single shot 1 - free-running |

**Table 154. TIMERS_TIMER1_TIMEOUT_REG (address offset 0x0010)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | 0 | R | Reserved |
| 11:0 | TIMER1_TIMEOUT | 0 | R/W | Initial count value of Timer1 in step size of 0.30 ms[2]. If set to 0, this feature is disabled. |

**Table 155. TIMERS_TIMER1_COUNT_REG (address offset 0x0014)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | 0 | R | Reserved |
| 11:0 | TIMER1_COUNT | 0 | R | Current count value of Timer1 in step size of 0.30 ms |

**Table 156. TIMERS_TIMER2_CONTROL_REG (address offset 0x0018)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | R | Reserved |
| 0 | TIMER2_MODE | 0 | R/W | 0 – single shot<br>1 – free running |

**Table 157. TIMERS_TIMER2_TIMEOUT_REG (address offset 0x001C)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:0 | TIMER2_TIMEOUT | 0 | R/W | Initial count value of Timer2 in step size of 0.30 ms[2]. If set to 0, this feature is disabled. |

**Table 158. TIMERS_TIMER2_COUNT_REG (address offset 0x0020)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:0 | TIMER2_COUNT | 0 | R | Current count value of Timer2 in step size of 50 ns |

**Table 159. TIMERS_TIMER3_CONTROL_REG (address offset 0x0024)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | R | Reserved |
| 0 | TIMER3_MODE | 0 | R/W | 0: single shot<br>1: free running |

**Table 160. TIMERS_TIMER3_TIMEOUT_REG (address offset 0x0028)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:0 | TIMER3_TIMEOUT | 0 | R/W | Initial count value of Timer3 in step size of 0.30 ms[2]. If set to 0, this feature is disabled. |

**Table 161. TIMERS_TIMER3_COUNT_REG (address offset 0x002C)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:0 | TIMER3_COUNT | 0 | R | Current count value of Timer3 in step size of 50 ns |

**Table 162. TIMERS_WDOG_CONTROL_REG (address offset 0x0030)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | R | reserved |
| 0 | WDOG_KICK | 0 | D | 1: re-initialize the Watchdog Timer to value WDOG_TIMEOUT<br>0: no effect |

**Table 163. TIMERS_WDOG_TIMEOUT_REG (address offset 0x0034)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | 0 | R | reserved |
| 9:0 | WDOG_TIMEOUT | 0 | R/W | Initial count value of Watchdog Timer in step size of 21.5 ms<br>If set to 0, this feature is disabled. |

**Table 164. TIMERS_WDOG_TRIGGER_INT_REG (address offset 0x0038)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | 0 | R | reserved |
| 9:0 | WDOG_INT_THRESHOLD | 0 | R/W | Count value of Watchdog Timer which triggers interrupt *intreq_wdog_o* |

**Table 165. TIMERS_WDOG_COUNT_REG (address offset 0x003C)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | 0 | R | reserved |
| 9:0 | WDOG_COUNT | 0 | R | Current count value of Watchdog Timer in step size of 21.5 ms |

**Table 166. TIMERS_WDOG_INT_STATUS_REG (address offset 0x3FCC)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | R | reserved |
| 0 | WDOG_TIMEOUT_ STATUS | 0 | R | Watchdog timeout interrupt status |

**Table 167. TIMERS_WDOG_INT_CLR_STATUS_REG (address offset 0x3FD0)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | W | reserved |
| 0 | WDOG_TIMEOUT_CLR_STATUS | 0 | W | 1: clear Watchdog timeout interrupt<br>0: no effect |

**Table 168. TIMERS_WDOG_INT_SET_STATUS_REG (address offset 0x3FD4)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:1 | RESERVED | 0 | W | reserved |
| 0 | WDOG_TIMEOUT_SET_STATUS | 0 | W | 1: set Watchdog timeout interrupt<br>0: no effect |

**Table 169. TIMERS_INT_CLR_ENABLE_REG (address offset 0x3FD8)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | 0 | W | reserved |
| 3 | TIMER3_TIMEOUT_CLR_ENABLE | 0 | W | 1: clear enable for Timer3 timeout interrupt<br>0: no effect |
| 2 | TIMER2_TIMEOUT_CLR_ENABLE | 0 | W | 1: clear enable for Timer2 timeout interrupt<br>0: no effect |
| 1 | TIMER1_TIMEOUT_CLR_ENABLE | 0 | W | 1: clear enable for Timer1 timeout interrupt<br>0: no effect |
| 0 | TIMER0_TIMEOUT_CLR_ENABLE | 0 | W | 1: clear enable for Timer0 timeout interrupt<br>0: no effect |

**Table 170. TIMERS_INT_SET_ENABLE_REG (address offset 0x3FDC)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | 0 | W | reserved |
| 3 | TIMER3_TIMEOUT_SET_ENABLE | 0 | W | 1: set enable for Timer3 timeout interrupt<br>0: no effect |
| 2 | TIMER2_TIMEOUT_SET_ENABLE | 0 | W | 1: set enable for Timer2 timeout interrupt<br>0: no effect |
| 1 | TIMER1_TIMEOUT_SET_ENABLE | 0 | W | 1: set enable for Timer1 timeout interrupt |

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| | | | | 0: no effect |
| 0 | TIMER0_TIMEOUT_SET_ENABLE | 0 | W | 1: set enable for Timer0 timeout interrupt<br>0: no effect |

**Table 171. TIMERS_INT_STATUS_REG (address offset 0x3FE0)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | 0 | R | reserved |
| 3 | TIMER3_TIMEOUT_STATUS | 0 | R | Timer3 timeout interrupt status |
| 2 | TIMER2_TIMEOUT_STATUS | 0 | R | Timer2 timeout interrupt status |
| 1 | TIMER1_TIMEOUT_STATUS | 0 | R | Timer1 timeout interrupt status |
| 0 | TIMER0_TIMEOUT_STATUS | 0 | R | Timer0 timeout interrupt status |

**Table 172. TIMERS_INT_ENABLE_REG (address offset 0x3FE4)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | 0 | R | Reserved |
| 3 | TIMER3_TIMEOUT_ENABLE | 0 | R | Timer3 timeout interrupt enable |
| 2 | TIMER2_TIMEOUT_ENABLE | 0 | R | Timer2 timeout interrupt enable |
| 1 | TIMER1_TIMEOUT_ENABLE | 0 | R | Timer1 timeout interrupt enable |
| 0 | TIMER0_TIMEOUT_ENABLE | 0 | R | Timer0 timeout interrupt enable |

**Table 173. TIMERS_INT_CLR_STATUS_REG (address offset 0x3FE8)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | 0 | W | reserved |
| 3 | TIMER3_TIMEOUT_CLR_STATUS | 0 | W | 1: clear Timer3 timeout interrupt<br>0: no effect |
| 2 | TIMER2_TIMEOUT_CLR_STATUS | 0 | W | 1: clear Timer2 timeout interrupt<br>0: no effect |
| 1 | TIMER1_TIMEOUT_CLR_STATUS | 0 | W | 1: clear Timer1 timeout interrupt<br>0: no effect |
| 0 | TIMER0_TIMEOUT_CLR_STATUS | 0 | W | 1: clear Timer0 timeout interrupt<br>0: no effect |

**Table 174. TIMERS_INT_SET_STATUS_REG (address offset 0x3FEC)**

| Bit | Symbol | Reset Value | Access Type | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | 0 | W | reserved |
| 3 | TIMER3_TIMEOUT_SET_STATUS | 0 | W | 1: set Timer3 timeout interrupt<br>0: no effect |
| 2 | TIMER2_TIMEOUT_SET_STATUS | 0 | W | 1: set Timer2 timeout interrupt<br>0: no effect |
| 1 | TIMER1_TIMEOUT_SET_STATUS | 0 | W | 1: set Timer1 timeout interrupt<br>0: no effect |
| 0 | TIMER0_TIMEOUT_SET_STATUS | 0 | W | 1: set Timer0 timeout interrupt<br>0: no effect |

# 12. PN7462AU Contactless interface

The PN7462AU embeds a high power 13.56 RF front end. The RF interface implements the RF functionality, like antenna driving, and the receiver circuitry and all the low-level functionalities, to enable to realize an NFC forum or an EMV Co compliant reader. The PN7462AU allows different voltages for the RF drivers. The PN7462AU uses an external oscillator, working at 27.12 MHz, as a clock source for generating the RF field and its internal operation.

## 12.1 Contactless interface features

- ISO/IEC 14443 type A & B compliant
- MIFARE functionality, including MIFARE Classic encryption in read/write mode
- ISO/IEC 15693 compliant
- ICLASS UID
- NC Forum - NFCIP-1 & NFC IP2 compliant
- P2P, active and passive mode
- Reading of NF C Forum Tag Types 1,2,3,4,5
- FeliCa
- ISO/IEC18000-3 Mode 3
- EPC UID
- EMVCo contactless
    - RF level can be achieved without the need of booster circuitry (for some antenna topologies the EMV
    - RF-level compliance might physically not be achievable)
- Card mode - enabling the emulation of an ISO14443 Type A card
    - Supports PLM (Passive Load Modulation) and ALM (Active Load Modulation)
- LPCD - Low Power Card Detection
- Adjustable Rx-Voltage level

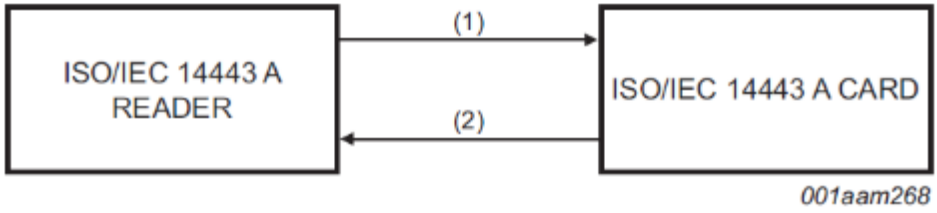
A minimum voltage of 2.3 V enable to use card emulation, and P2P passive target functionality - in passive load modulation. A voltage above 2.7 V enables the complete contactless functionality.

## 12.2 Reader/Writer modes

### 12.2.1 ISO/ IEC14443 A/ MIFARE and Jewel/Topaz PCD modes

The ISO/IEC 14443A/MIFARE PCD mode is the general reader to card communication scheme according to the ISO/IEC 14443A specification. This modulation scheme is as well used for communications with Jewel/Topaz cards.

The physical level of the communication is shown in Fig 23.




(1) Reader to Card 100 % ASK, Modified Miller Coded, Transfer speed 106 kbit/s to 848 kbit/s

(2) Card to Reader, Subcarrier Load Modulation Manchester Coded or BPSK, transfer speed 106 kbit/s to 848 kbit/s

**Fig 23. ISO/IEC 14443 A/MIFARE read/write mode communication diagram**

The physical parameters are described in Table 175

**Table 175. Communication overview for ISO/IEC 14443 A/MIFARE reader/ writer**

| Communication direction | Signal type | Transfer speed | | | |
|---|---|---|---|---|---|
| | | 106 kbit/s | 212 kbit/s | 424 kbit/s | 848 kbit/s |
| Reader to card<br>$f_c$ = 13.56 MHz | Reader side modulation | 100 % ASK | 100 % ASK | 100 % ASK | 100 % ASK |
| | Bit encoding | Modified Miller encoding | Modified Miller encoding | Modified Miller encoding | Modified Miller encoding |
| | Bit rate [kbit/s] | $f_c$/128 | $f_c$/64 | $f_c$/32 | $f_c$/16 |
| Card to reader | Card side modulation | Subcarrier load modulation | Subcarrier load modulation | Subcarrier load modulation | Subcarrier load modulation |
| | Subcarrier frequency | $f_c$/16 | $f_c$/16 | $f_c$/16 | $f_c$/16 |
| | Bit encoding | Manchester encoding | BPSK | BPSK | BPSK |

No connection to a host is required to manage the complete ISO/IEC 14443 A/MIFARE protocol. Fig 24 shows the data coding and framing according to ISO/IEC 14443 A/ MIFARE.

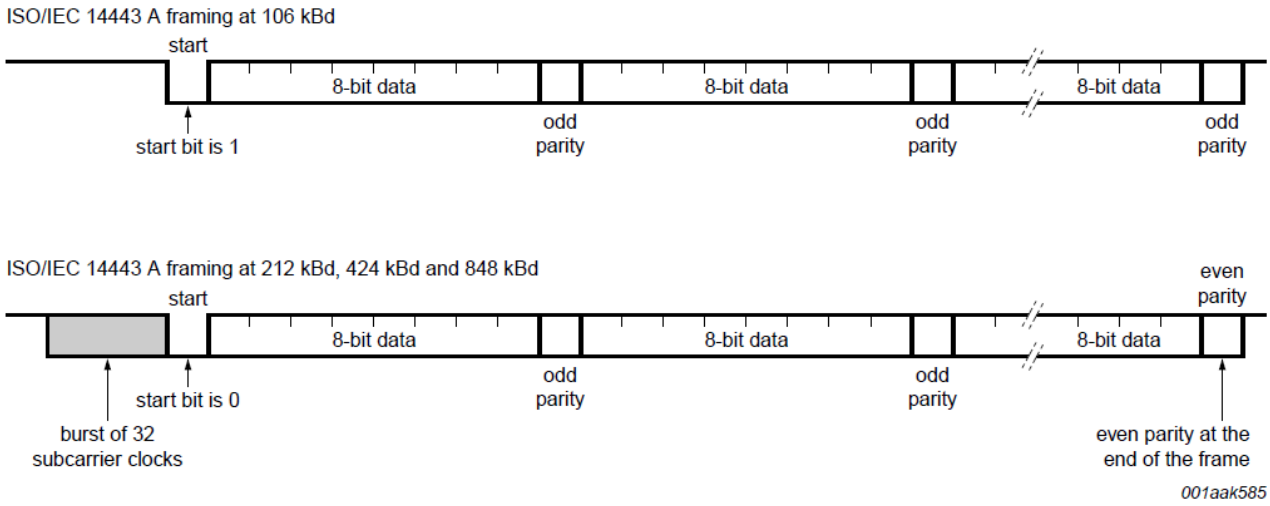
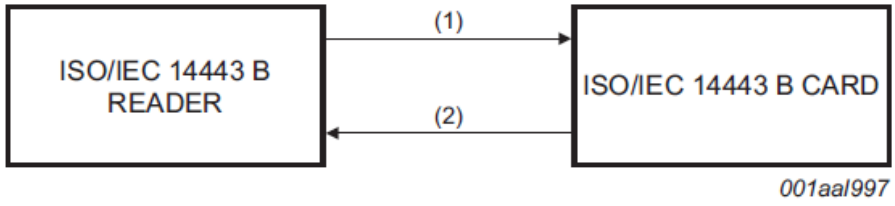



**Fig 24. Data coding and framing according to ISO/IEC 14443 A card response**

The internal CRC coprocessor calculates the CRC value based on the selected protocol. In card mode for higher baud rates, the parity is automatically inverted as end of communication indicator.

### 12.2.2 ISO/IEC14443 B PCD mode

The physical level of the communication is shown in Fig 25




(1) Reader to Card NRZ, transfer speed 106 kbit/s to 848 kbit/s

(2) Card to reader, Subcarrier Load Modulation Manchester Coded or BPSK, transfer speed 106 kbit/s to 848 kbit/s

**Fig 25. ISO/IEC 14443B read/write mode communication diagram**

The physical parameters are described in Table 176.

**Table 176. Communication overview for ISO/IEC 14443 B reader/ writer**

| Communication direction | Signal type | Transfer speed | | | |
|---|---|---|---|---|---|
| | | 106 kbit/s | 212 kbit/s | 424kbit/s | 848 kbit/s |
| Reader to card | Reader side modulation | 10 % ASK | 10 % ASK | 10 % ASK | 10 % ASK |
| | Bit encoding | NRZ | NRZ | NRZ | NRZ |
| | Bit rate [kbit/s] | $f_c/128$ | $f_c/64$ | $f_c/32$ | $f_c/16$ |
| Card to reader | Card side modulation | Subcarrier load modulation | Subcarrier load modulation | Subcarrier load modulation | Subcarrier load modulation |
| | Subcarrier frequency | $f_c/16$ | $f_c/16$ | $f_c/16$ | $f_c/16$ |
| | Bit encoding | BPSK | BPSK | BPSK | BPSK |

### 12.2.3 FeliCa PCD mode

The FeliCa mode is the general reader/ writer to card communication scheme according to the FeliCa specification. The communication on a physical level is shown in Fig 26

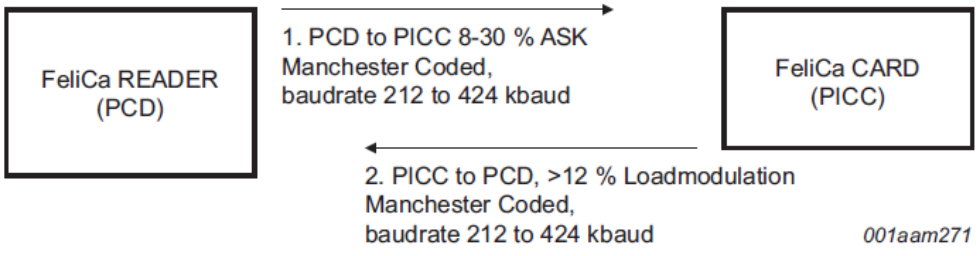



**Fig 26. FeliCa read/ write communication diagram**

The physical parameters are described in Table 177

**Table 177. Communication overview for FeliCa reader/ writer**

| Communication direction | Signal type | Transfer speed FeliCa | FeliCa higher transfer speed |
|---|---|---|---|
| | | 212 kbit/s | 424 kbit/s |
| Reader to card | Reader side modulation | 8 to 30 % ASK | 8 to 30 % ASK |
| | Bit encoding | Manchester encoding | Manchester encoding |
| | Bit rate | $f_c/64$ | $f_c/32$ |
| Card to reader | Card side modulation | Load modulation | Load modulation |
| | Bit encoding | Manchester encoding | Manchester encoding |

#### 12.2.3.1 Multiple reception cycles (RXMultiple)

For FeliCa timeslot handling in PCD mode PN7462AU implements multiple reception cycles. The feature is enabled by setting the control bit RX_MULTIPLE_ENABLE in the register TRANSCEIVE_CONTROL_REG in combination with the transceive or the receive command.

Unlike for normal operation the receiver is enabled again after a reception is finished. It is necessary to issue the IDLE command in order to leave the RXMultiple cycle. As there is only one receive buffer available but several responses are expected the buffer is split into sub buffers of 32-byte length. Hence, the maximum number of responses which can be handled is limited to 8. As the maximum length defined for a FeliCa response is 20 bytes the buffer size defined does fulfill the requirements for that use-case. The first data frame received is copied onto buffer address 0. The subsequent frames will be copied to the buffer address 32 * NumberOfReceivedFrames. The maximum number of data bytes allowed per frame is limited to 28.

All bytes in the buffer between the payload and the status byte are un-initialized and therefore invalid. FW has to take care that these bytes are not used. The last word of the sub buffer (position 28 to 31) contains a status word. The status word contains the number of received bytes (may vary from the FeliCa length in case of an error), the CLError flag indicating any error in the reception (which is a combination of 3 individual error flags DATA_INTEGRITY_ERROR || PROTOCOL_ERROR || COLLISION_DETECTED) the individual error flags and the LenError flag indicating an incorrect length byte (either length byte is greater than 28 or the number of received bytes is shorter than indicated by the length byte). All unused bits (RESERVED) are masked to 0.

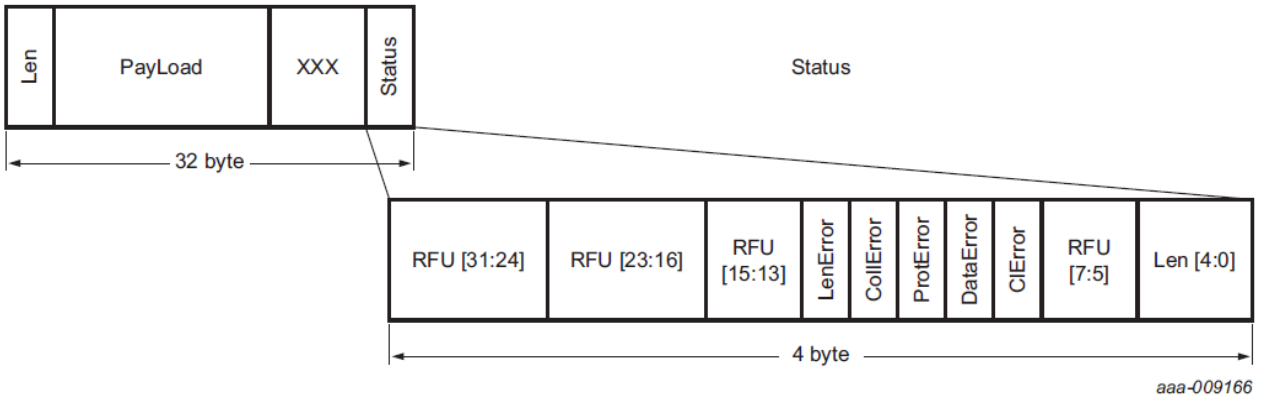



**Fig 27. RXMultiple data format**

There are 4 different scenarios possible for reception:

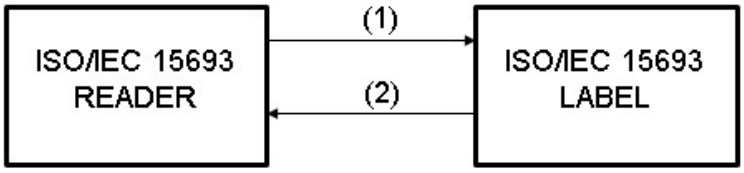
1. Correct reception - Data integrity is correct (no CRC error), and additionally the number of bytes received is equal to the length byte. Data is written to the buffer. No error set in status byte.

2. Erroneous reception - Data is incorrect (data integrity error - CRC wrong) but frame length is correct. Data is written to buffer and the bits CLError and DataError in the status byte are set.
3. Erroneous reception - the length byte received indicates a frame length greater than 28. No data is copied to buffer but status byte with LenError bit set is written.
4. Erroneous reception - the length byte is larger than the number of data bytes, which have been received. Data received is written to buffer and the ProtocolError bit in the status byte is set.

For each reception, the RX_IRQ in the GENERAL_IRQ_STATUS_REG is set. FW can disable the IRQ and use a timer for time-out after the last time slot to avoid excessive interaction with the hardware. At the end of the reception additionally the bit field RX_NUM_FRAMES_RECEIVED in the register RX_STATUS_REG is updated to indicate the number of received frames. After the reception of the 8th frame (which is the maximum supported) a state change to next expected state is executed (IDLE for Receive command and WaitTransmit for transceive command). Consequently, the reception is stopped. Upon start of a new reception cycle the flag RX_NUM_FRAMES_RECEIVED is cleared. The duration between deactivate and reactivate is at minimum 2 RF cycles and can last up to 2 us.

### 12.2.4 ISO/IEC 15693 PCD mode

The physical level of the communication is shown in Fig 28.




(1) Reader to Label: 1/256 and 1/4 encoding
(2) Label to Reader: Manchester coding

**Fig 28. ISO/IEC 15693 read/write mode communication diagram**

The physical parameters are described in Table 178.

**Table 178. ISO/IEC 15693 reader/writer to label communication overview**

| Communication direction | Signal type | Transfer speed | | | |
|---|---|---|---|---|---|
| | | **$f_c$/8192 kbit/s** | | **$f_c$/512 kbit/s** | |
| Reader to card | Reader side modulation | 10 to 30 % ASK or 100 % ASK | | 10 to 30 % ASK 90 % to 100 % ASK | |
| | Bit encoding | 1/256 | | 1/4 | |
| | Bit length | 4.833 ms | | 302.08 us | |
| | | **6.62 (6.67) kbit/s** | **13.24 kbit/s** | **26.48 (26.69) kbit/s** | **52.96 kbit/s** |
| Card to reader | Card side modulation | Not supported | Not supported | single (dual) subcarrier load modulation ASK | single subcarrier load modulation ASK |
| | Bit length | - | - | 37.76(3.746) us | 18.88 us |
| | Bit encoding | - | - | Manchester coding | Manchester coding |
| | Sub-carrier frequency | - | - | $f_c$ /32 ($f_c$ /28) MHz | $f_c$ /32 MHz |

### 12.2.5 ISO/IEC18000-3 Mode 3 VCD mode

The ISO/IEC 18000-3 mode 3 is not described in this document. For a detailed explanation of the protocol, refer to the ISO/IEC 18000-3 standard.

## 12.3 NFC modes

### 12.3.1 NFCIP – 1 modes

#### 12.3.1.1 Overview

The NFCIP-1 communication differentiates between an Active and a Passive Communication Mode.

Active Communication mode means both the initiator and the target are using their own RF field to transmit data.

- Passive Communication mode means that the target answers to an initiator command in a load modulation scheme. The initiator is active in terms of generating the RF field.
- Initiator: generates RF field at 13.56 MHz and starts the NFCIP-1 communication
- Target: responds to initiator command either in a load modulation scheme in Passive Communication mode or using a self-generated and self-modulated RF field for Active Communication mode.

In order to fully support the NFCIP-1 standard the PN7462AU supports the Active and Passive Communication mode at the transfer speeds 106 kbit/s, 212 kbit/s and 424 kbit/s as defined in the NFCIP-1 standard.

#### 12.3.1.2 Active communication mode

Active communication mode means both the initiator and the target are using their own RF field to transmit data.

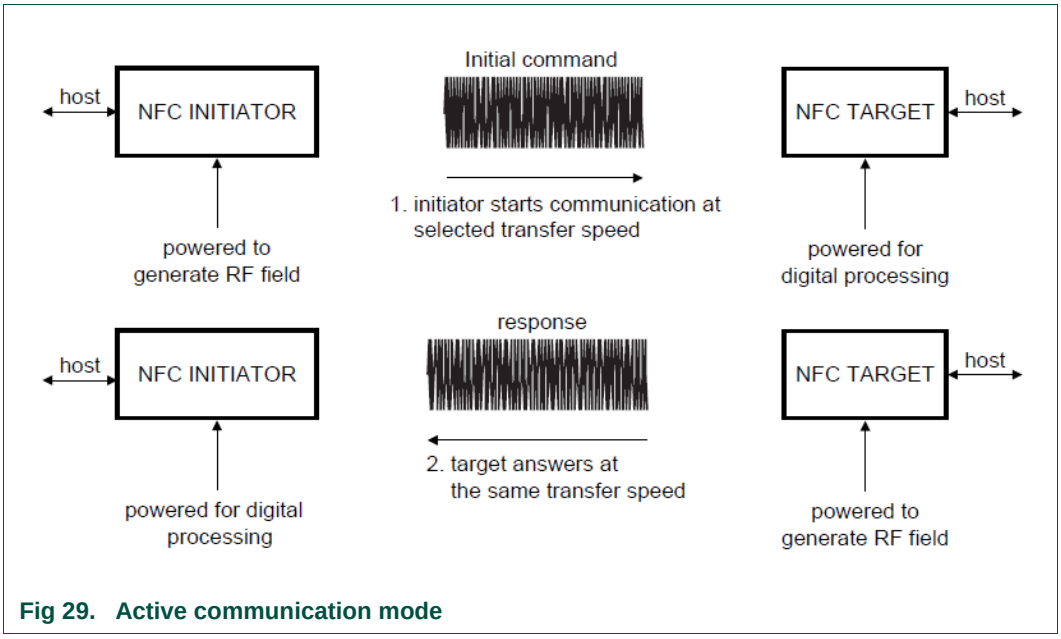



**Fig 29. Active communication mode**

**Table 179. Communication overview for active communication mode**

| Communication direction | 106 kbit/s | 212 kbit/s | 424 kbit/s |
|---|---|---|---|
| Initiator → Target | According to ISO/IEC 14443A 100% ASK, modified Miller Coded | According to FeliCa Manchester Coded | 8 – 30 % ASK |
| Target → Initiator | | | |

A dedicated host controller firmware is required to handle the NFCIP-1 protocol.

Note: Transfer Speeds above 424 kbit/s are not defined in the NFCIP-1 standard.

#### 12.3.1.3 Passive communication mode

Passive Communication mode means that the target answers to an initiator command in a load modulation scheme. The initiator is active (powered) to generate the RF field.

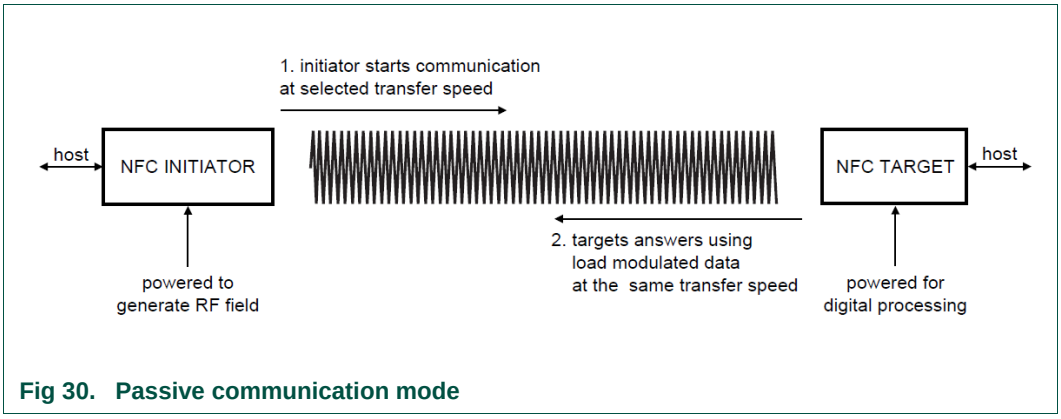



**Fig 30. Passive communication mode**

**Table 180. Communication overview for active communication mode**

| Communication direction | 106 kbit/s | 212 kbit/s | 424 kbit/s |
|---|---|---|---|
| Initiator → Target | According to ISO/IEC 14443A 100% ASK, modified Miller Coded | According to FeliCa Manchester Coded | 8 – 30 % ASK |
| Target → Initiator | According to ISO/IEC 14443A @ 106 kbit/s modified Miller Coded | According to FeliCa Manchester Coded | >12 % ASK |

A dedicated host controller firmware is required to handle the NFCIP-1 protocol.

**Note**: Transfer Speeds above 424 kbit/s are not defined in the NFCIP-1 standard.

#### 12.3.1.4 ISO/IEC14443 A/MIFARE Card operation mode

PN7462AU can be addressed as a ISO/IEC 14443 A card.

This means that PN7462AU can generate an answer in a load modulation scheme according to the ISO/IEC 14443 A interface description.

The following tables describes the physical parameters of an ISO/IEC14443 A card mode:

**Table 181. ISO/IEC14443 A Card operation mode**

| Communication direction | Signal type | Transfer speed | | |
|---|---|---|---|---|
| | | 106 kbit/s | 212 kbit/s | 424 kbit/s |
| Reader/ writer to PN7462AU | Reader side modulation | 100 % ASK | >25 % ASK | >25 % ASK |
| | Bit encoding | Modified Miller e | Modified Miller | Modified Miller |
| | Bit length | 128/$f_c$ us | 64/$f_c$ us | 32/$f_c$ us |
| PN7462AU to reader/ writer | Card side modulation | Subcarrier load modulation | Subcarrier load modulation | Subcarrier load modulation |
| | Subcarrier frequency | $f_c$/16 | $f_c$/16 | $f_c$/16 |
| | Bit coding | Manchester | BPSK | BPSK |

| Communication direction | Signal type | Transfer speed | | |
|---|---|---|---|---|
| | | 106 kbit/s | 212 kbit/s | 424 kbit/s |
| | Bit encoding | Manchester encoding | BPSK | BPSK |

#### 12.3.1.5 ISO/IEC14443 B Card operation mode

PN7462AU can be also addressed as an ISO/IEC 14443 B card. The following table describes the physical parameters of an ISO/IEC14443 B card mode:

**Table 182. ISO/IEC1443 B Card operation mode**

| Communication direction | Signal type | Transfer speed | | |
|---|---|---|---|---|
| | | 106 kbit/s | 212 kbit/s | 424 kbit/s |
| Reader/ writer to PN7462AU | Reader side modulation | 8-14 % ASK | 8-14 % ASK | 8-14 % ASK |
| | Bit encoding | NRZ | NRZ | NRZ |
| | Bit length | 128/$f_c$ us | 64/$f_c$ us | 32/$f_c$ us |
| PN7462AU to reader/ writer | Card side modulation | Subcarrier load modulation | Subcarrier load modulation | Subcarrier load modulation |
| | Subcarrier frequency | $f_c$/16 | $f_c$/16 | $f_c$/16 |
| | Bit coding | Manchester | BPSK | BPSK |
| | Bit encoding | BPSK | BPSK | BPSK |

#### 12.3.1.6 NFCIP-1 framing and coding

The NFCIP-1 framing and coding in Active and Passive Communication mode is defined in the NFCIP-1 standard. The following data rates are supported by the PN7462AU:

**Table 183. Framing and coding**

| Transfer speed | Framing and Coding |
|---|---|
| 106 kbit/s | According to the ISO/IEC 1443 A/ MIFARE scheme |
| 212 kbit/s | According to the FeliCa scheme |
| 424 kbit/s | According to the FeliCa scheme |

#### 12.3.1.7 NFCIP-1 protocol support

The NFCIP-1 protocol is not completely described in this document. The PN7462AU does not implement any of the high-level protocol functions. This higher level protocol function need to be provided by the host. For detailed explanation of the protocol refer to the NFCIP-1 standard. However, the data link layer is according to the following policy:

- Speed shall not be changed while continuous data exchange in a transaction.
- Transaction includes initialization, anti-collision methods and data exchange (in continuous way, meaning no interruption by another transaction).

In order not to disturb current infrastructure based on 13.56 MHz, the following general rules to start a NFCIP-1 communication are defined:

1. Per default a NFCIP-1 device is in Target mode - meaning its RF field is switched off.
2. The RF level detector is active.
3. Only if it is required by the application the NFCIP-1 device shall switches to Initiator mode.
4. An initiator shall only switch on its RF field if no external RF field is detected by the RF Level detector during a time of $T_{IDT}$.
5. The initiator performs initialization according to the selected mode.

### 12.3.2 NFC Configuration

The NFC protocol defines for 106kbps mode an additional Sync-Byte (0xF0 + parity) after the normal start bit transmitted. As this Sync-Byte has a parity it can be handled by a host FW as a normal data byte. The PN7462AU however, provides all means to automatically handle the Sync-Byte.

There are four different areas where adaptations in respect to the default configuration can be done:

- Reader mode signal processing to remove the Sync-Byte for NFC-Passive-Initiator
- mode
- Card mode signal processing configuration for removal of the Sync-Byte for all other
- NFC-modes (Passive-Target, Active-Target and Initiator)
- Transmitter settings to automatically add the Sync-Byte for transmission
- Adapted CRC preset value to correctly calculate the CRC

### 12.3.3 Card mode detection

The PN7462AU provides the functionality of automatic mode detection for the card mode. If activated and as soon as the receiver is enabled the signal processing module does permanently check for an incoming communication at one of the supported protocols (Type A, B, F). As soon as a Start-of-Frame for one protocol is detected all reception based CLIF registers are configured automatically to allow reception of the incoming frame. This includes all Rx protocol framing configuration (data-rate, parity, CRC, start-/stop-bit, EoF-detection, tx_wait guard time, miller-synchronization).

It is not allowed that FW modifies any of the registers set by the mode-detector while communication is ongoing. Of course, if necessary setting can be changed later on (e.g. for changing baudrates). But as all relevant configuration is done by the mode-detector for the ongoing reception any change made by FW can lead to a fail of the reception.

All transmission related configuration must be set-up by FW. As Type A is the most timing critical protocol and additionally the activation is done in hardware by the CMA the default configuration for the mode-detection is Type A.

Consequently, FW needs to set all TX registers according to Type A protocol up-front. The CRC however needs to be disabled because it is not used for the activation.

#### 12.3.3.1 General target mode configuration (GTM)

Depending on the configuration a certain protocol can be selected/disabled for the card mode detection:

- Miller decoder is enabled > TypeA/NFC 106kbps is detected
- NRZ decoder is enabled > TypeB 106kbps is detected
- Manchester decoder is enabled > Felica / NFC 212/424kbps is detected

It is not possible to enable only one baudrate for the Felica protocol.

#### 12.3.3.2 Card mode detection - CMA active

Usually, in the card mode the CMA is enabled to handle the Type A activation automatically. In case a Type A command is detected, the CMA takes over the activation process including the transceive flow control. In such a case after the initial MODE_DETECTED_IRQ all subsequent IRQs are disabled until either the Type A activation is complete (indicated by an CARD_ACTIVATED_IRQ) or a different protocol is detected. For the later scenario, the MODE_DETECTED_IRQ is set and all IRQs are unmasked again.

#### 12.3.3.3 Card mode detection - FW handling

In case a protocol different to Type A (Type B or Felica) is detected FW needs to handle the complete activation process. In such a case CMA is inactive and all IRQs are passed to FW.

### 12.3.4 Active Load Modulation

When PN7462AU is used in the card emulation mode, or P2P passive target mode, it has to modulate the field emitted by the external reader or NFC passive initiator.

To modulate the field, PN7462AU has two possibilities:

- Passive Load Modulation (PLM): the reader modifies its antenna characteristics, which will be detecting by the reader through the antenna coupling.
- Active Load Modulation (ALM): the reader generates a small field, in phase opposition with the reader's emitted field, which will be detected by the reader reception stage.

The modulation type to use depends on the external reader and PN7462AU antenna, and must be chosen depending on the application.

### 12.3.5 Low Power Card Detection

If a card is moved into the RF field of the reader, then in many cases the antenna is detuned and the voltage value at the reader changes. This fact is exploited for low power card detection, where the AGC is used to detect small changes of the RF field. The AGC is used in fast-mode in order to keep the operational time to a minimum.

The following steps are performed in a loop:

1. Write the Rx-Divider nominal value into the CLIF_AGC_VALUE_REG.AGC_VALUE_RM

2. Activate Fast-tracking mode
   a. Set AGC_FAST_MODE_ENABLE = 1
   b. Set AGC_FAST_MODE_STEP
   c. Set AGC_FAST_MODE_DURATION
3. Configure to load the Rx-Divider to nominal value
   a. Disable the AGC operation by writing “0” to CLIF_AGC_CONFIG0_REG.AGC_MODE_ENABLE.
   b. Configure the AGC to manual mode by writing “0” to CLIF_AGC_CONFIG0_REG.AGC_MODE_SEL.
   c. Configure the AGC input to card-mode by writing “1” to CLIF_AGC_CONFIG0_REG.AGC_INPUT_SEL.
   d. Configure the AGC time constant and the CLIF_AGC_CONFIG0_REG.AGC_TIME_CONSTANT (value to be added) and CLIF_AGC_CONFIG0_REG.AGC_THRESHOLD (value to be added)
4. Load the Rx-Divider to nominal value (first four steps match 3.)
   a. Disable the AGC operation by writing “0” to CLIF_AGC_CONFIG0_REG.AGC_MODE_ENABLE.
   b. Configure the AGC to manual mode by writing “0” to CLIF_AGC_CONFIG0_REG.AGC_MODE_SEL.
   c. Configure the AGC input to card-mode by writing “1” to CLIF_AGC_CONFIG0_REG.AGC_INPUT_SEL.
   d. Configure the AGC time constant and the CLIF_AGC_CONFIG0_REG.AGC_TIME_CONSTANT (value to be added) and CLIF_AGC_CONFIG0_REG.AGC_THRESHOLD (value to be added)
   e. Load the CM value CLIF_AGC_VALUE_REG.AGC_VALUE_RM into the AGC register by writing “1” to CLIF_AGC_CONFIG_REG.AGC_LOAD.
5. Start AGC feedback operation
   a. Enable the AGC by writing “1” to CLIF_AGC_CONFIG_REG.AGC_MODE_ENABLE.
   b. Configure the AGC to feedback mode by writing “1” to CLIF_AGC_CONFIG0_REG.AGC_MODE_SEL.
6. Wait a certain time for AGC to settle.
7. Card detection:
   a. Compare AGC control value with nominal value. If the change exceeds a certain threshold, then a card is detected.
   b. Perform protocol check using Reader Mode configuration for the AGC.
   c. If no card responds to the request, then the actual AGC value is stored as nominal value for the next comparison.

## 12.4 Functional Description

Fig 31 shows an overview of the Contactless interface.

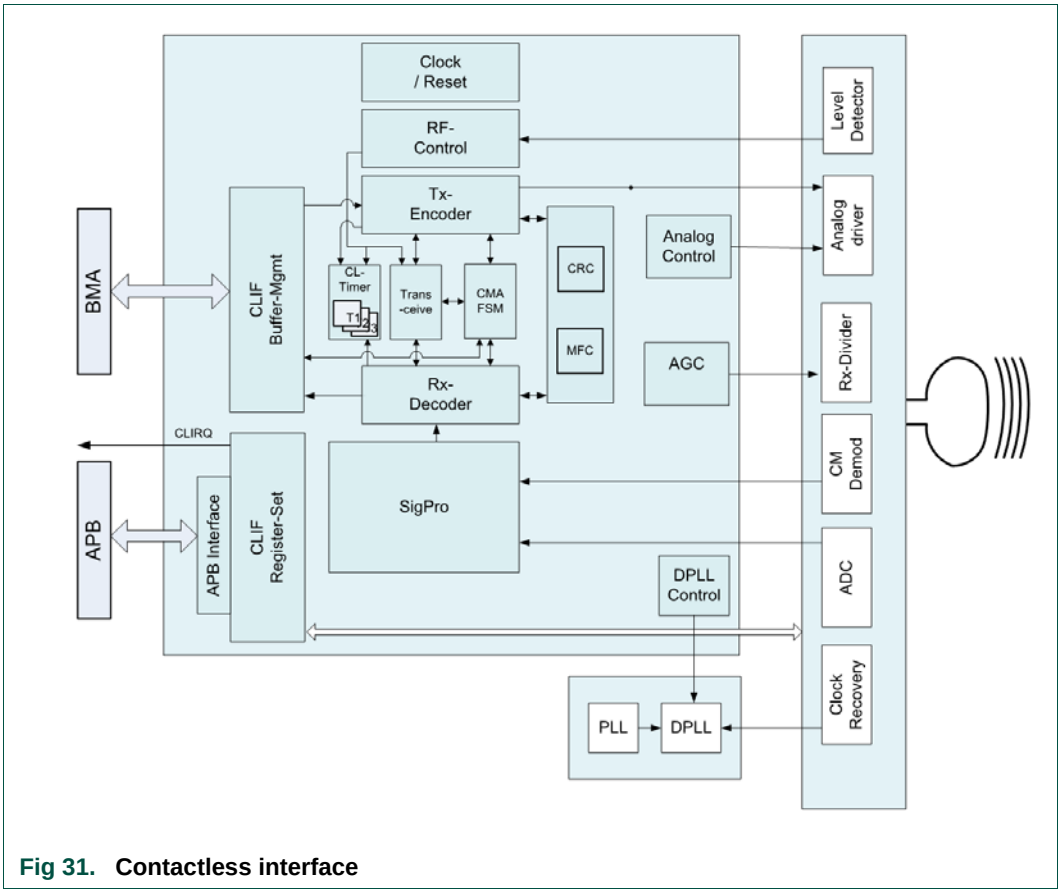



**Fig 31. Contactless interface**

### 12.4.1 Clock/Reset Block

The Clock/Reset module is the source of all clocks and reset used in the CLIF.

Table 184 summarized the input clocks of the CLIF module.

**Table 184. Input clocks of CLIF**

| Symbol | Source | Frequency | Description |
|---|---|---|---|
| clk_pll_27m12 | PCR | 27.12 MHz | IP clock for active contactless communication. |
| clk_rf_27m12 | Clock Recovery | 27.12 MHz | Clock recovered from the RF-Field by the analog CLIF clock recovery |
| clk_pcr_lfo | PCR | 380 KHz | Low frequency oscillator clock |
| clk_pcr_hfo | PCR | 20 MHz | High frequency oscillator clock |

The main clock for the contactless interface is the clk13 - a 13.5 6MHz clock derived from 27.12 MHz clock either from PLL or the analog RF clock recovery. The RF clock is only

present when the device is within the RF-Field. PLL clock is available as soon as the CLIF PLL is enabled and locked to the input clock.

The HFO (high frequency oscillator – 20 MHz nominal) clock is always available as soon systems is active. Some functional blocks use this clock as the functionality must be available even if the main contactless clock is not available. Modules interfacing the ARM sub-system (APB register interface and Buffer Manager) also operate at system clock (nominal 20 MHz).

### 12.4.2 Digital PLL (DPLL) Control

For Active Load Modulation (ALM) with a single loop antenna the PLL is extended by a digital PLL (DPLL), which is controlled by this module. For ALM with a single loop antenna the generated frequency by the PLL must be aligned with the field clock in phase and frequency. During transmission, the frequency is kept constant (as close as possible to the frequency of the reader) such that the phase drift between the RF signal generated by the reader and the transmitted signal by the card (ALM) is kept to a minimum.

### 12.4.3 CLIF Signal Processing (SigPro) Block

This SigPro module performs digital signal processing. The input is the demodulated envelope signal and the output is a digital data signal, which is further processed in the

Rx-Decoder. The SigPro module consists of three modes i.e., reader-mode, card-mode, and ADC based card-mode. The SigPro is the interface between the analog frontend and the Rx-Decoder. The SigPro generates a bit stream out of the input signal (analog demodulator signal or ADC data signal). The output data is further processed in the Rx-Decoder for protocol handling.

#### 12.4.3.1 Functional features

- Decode card-mode ISO14443 Type A (106 kBd, 212 kBd, 424 kBd, 848 kBd: MIFARE): Miller decoder (100 % ASK modulation)
- Decode card-mode ISO14443 Type B (106 kBd, 212 kBd, 424 kBd, 848 kBd): NRZ decoder (10 % ASK modulation)
- Decode card-mode ISO18092 (106kBd): A: Miller decoder (100 % ASK modulation)
- Decode card-mode ISO18092 (212kBd, 424kBd: Felica): Manchester decoder (10 % ASK Modulation)
- Target mode detection (Type A / Type B / FeliCa)
- Decode reader-mode ISO14443 Type A (106 kBd: MIFARE): Manchester decoder (848kHz subcarrier with load modulation)
- Decode reader-mode ISO14443 Type A (212 kBd, 424 kBd, 848 kBd: MIFARE): BPSK decoder (848 kHz subcarrier with load modulation)
- Decode reader-mode ISO14443 Type B BPSK decoder (848 kHz subcarrier with load modulation).

- Decode reader-mode ISO18092 (106 kBd): Manchester decoder (848 kHz subcarrier with load modulation)
- Decode reader-mode ISO18092 (212 kBd, 424 kBd: Felica): Manchester decoder (848 kHz subcarrier with load modulation
- Decode Felica (212 kBd, 424 kBd): Manchester decoder
- Decode ISO15693 (SSC, DSC (FSK))
- Decode ISO 18000-3 Mode 3 (M=10, M=11 only, no PJM)
- Decode active-mode ISO18092 (106 kBd, 212 kBd, 424 kBd): Miller decoder (100 % ASK modulation) for 106 kBd

### 12.4.4 TX-Encoder

The TX-Encoder handles all kinds of data processing for data transmission via the contactless interface. This includes character encoding, comprising parallel-serial conversion, start- and stop bit generation, calculation and appending of parity bits, byte adjustments for bit-oriented frames and MIFARE encryption, and on the other hand frame generation, which covers creation of start-of-frame, end-of-frame and special protocol symbols as well as appending of CRC values.

#### 12.4.4.1 Functional features

- Encoder for ISO14443 type A (PICC & PCD
- Encoder for ISO14443 type B (PICC & PCD)
- Encoder for ISO18092 (PICC & PCD)
- Encoder for FeliCa (PICC & PCD)
- Encoder for I-Code EPC-V2 (only PCD)
- Encoder for I-Code ISO15693 (only PCD)
- Encoder for Jewel (only PCS)
- Parallel-to-serial conversion
- “1 out of 4”, “1 out of 256”, NRZ, PIE code generation
- Start bit, stop bit, EGT generation
- Parity calculation and appending
- Byte adjustment for bit-oriented frames
- MIFARE encryption
- Bitwise CRC (5-bit and 16-bit) appending
- CRC extension for Jewel (extend 7bit frames to 8bits => emulates byte-wise CRC)
- 16-bit to define protocol symbols (e.g., SOF and EOF)
- Flexible baudrate and subcarrier selection
- Modulation width and type adjustable
- Special envelope combinations selectable
- Frame-Step mode

### 12.4.5 RX-Decoder

The Rx-Decoder decodes the bit stream from the signal processing block CLIF SigPro) and performs serial-to-parallel conversion, frame checking (including stop-bit check as well as EOF check) and collision detection. Additionally, the Rx-Decoder handles data integrity checking which means it performs a parity check and hands the received data to the CRC coprocessor for CRC check.

#### 12.4.5.1 Functional features

- Serial-parallel conversion
- Stop bit detection and check
- Data integrity check (Parity and CRC)
- Byte alignment and bit count for bit oriented frames
- Collision detection
- MIFARE decryption
- Configurable EOF detection
- ISO 14443B EOF detection
- Configurable stop conditions
- Frame check

### 12.4.6 Contactless Transceive module

The Transceive module is handling the overall flow for contactless communication (transmission and reception) by controlling the Tx-Encoder and the Rx-Decoder. It also has a timer to accurately control FDT as well as FWT.

#### 12.4.6.1 Functional features

- Automatic switching between transmission and reception
- Accurate control of FDT and FWT
- Firmware controlled start of transmission

### 12.4.7 Buffer Management

The CLIF buffer manager handles data transfer between RAM and the CLIF for transmission and reception. It manages two 32 bit buffers which are used to synchronize data between the CLIF clock domain and the System clock. On CLIF side the Rx-Decoder and the Tx-Encoder provide/request data byte wise so the buffer manager and it handle word to byte conversion or vice versa.

### 12.4.8 Contactless CRC module

The contactless CRC module performs the CRC calculation for data transmission and reception. It is used by the TX-Encoder and the RX-Decoder. The CRC is configurable to

comply with ISO14443 Type A and B, ISO18092, FeliCa, ISO15693, Jewel, and I-Code EPC-V2 (ISO18000).

#### 12.4.8.1 Functional features

- Supports different preset values of the CRC register
- Supports 5-bit and 16-bit CRC calculation
- Supports inverted CRC output

### 12.4.9 Analog Control Block

The Analog Control block is responsible for switching correctly analog control signals depending on the actual system state.

### 12.4.10 RF Control Block

The RF Control block is responsible for switching on/off of the analog drivers. To be compliant with the ISO18092 specification for peer-to-peer communication special guard times ($T_{IDT}$/$T_{ADT}$) are handled. Collision avoidance - variation of the guard waiting time – is handled as well if enabled by software.

RF Control block contain analog Level detector. Status indication for external field detection is performed by analog Level detector. This is necessary on the one hand to know whenever an external field is present in card mode and other hand to avoid overlapping with an external field when the ICs driver shall be activated. The external field indication includes masking a self-generated RF-Field (which is detected by analog level detector too) as well as RF clock frequency checking. As the analog level detector has a latency of approximately 15 us. It is necessary to mask the analog level detect signal to avoid external field indication after a detected RF-Field disappeared. A period of 25 us is implemented in the RFControl module for masking to level detector signal. This corresponds to 512 HFO clock cycles (nominal 20 MHz).

### 12.4.11 Automatic Gain Control (AGC) block

The AGC is used to control the analog Rx-Divider in order to keep the RX-voltage level constant, i.e., ideally independent of the field strength. Consequently, the full dynamic range of the ADC can be exploit and also the performance of the analog demodulators is increased.

### 12.4.12 Card Mode Activation (CMA) Block

The CMA module manages the Type-A activation flow including all state transitions from **IDLE** state to **ACTIVE** state. If enabled, it interacts with all necessary modules to handle the activation process fully automated. When reaching the **ACTIVE** state an IRQ flag is set and the control is handed over to software. After reaching the **ACTIVE** state, the firmware disables the CMA.

### 12.4.13 MFC Block

The MFC (MIFARE Crypto) block generates the cryptographic data to encipher and decipher data for communication with MIFARE cards. It is supported by a dedicated PRNG.

#### 12.4.13.1 Functional features

- MIFARE authentication for PCD and PICC
- MIFARE encryption and decryption
- MIFARE pseudo random number generation (PRNG)

### 12.4.14 Timers

There are three general purpose timers T0, T1 and T2 running with the CLIF clock and one additional timer, T3, operating on HFO (~ 20MHz) or LFO (~ 375 kHz).

**RF Timers:** Timers T0, T1 and T2 have 20 bits and may be operated at clock frequencies derived from the 13.56MHz system clock.

**Additional Timer:** Timer T3 has 20 bits and can be operated with the LFO (~375 kHz) or the HFO (~20MHz) clock. Note that this timer requires the RF clock (`clk13`) to be functional.

At expiration of the timer a flag is raised and an IRQ is triggered. Refer to the Register-Set specifications for details on the IRQ handling.

#### 12.4.14.1 Functional features

**Timers T0, T1 and T2**

- 20-bit
- Runs on RF clock
- The 13.56 MHz RF clock may be divided by 2, 4, 8, 16, 32, 64, 128, and 256
- Several Start events: Start now, Start on external RF field on/off, Start on Rx/Tx started/ended, Start on timer T3 running,
- Reload
- IRQ is triggered at expiration

**Timer T3 for oscillator trimming**

- 20-bit
- Start -event: Start now
- Runs on HFO or LFO clock

## 12.5 CLIF register overview

**Table 185. CLIF Register overview (base address 0x4000 4000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLIF_CONTROL_REG | 0000h | 32 | R/W | 00000000h | Main control register for CLIF digital |
| CLIF_CLOCK_CONTROL_REG | 0004h | 32 | R/W | 00000000h | CLIF digital clock control register |
| CLIF_STATUS_REG | 0008h | 32 | R | 10000000h | CLIF digital status register |
| CLIF_TRANSCEIVE_CONTROL_REG | 000Ch | 32 | R/W | 00000000h | Transceiver control register |
| CLIF_TX_WAIT_REG | 0010h | 32 | R/W | 00000000h | Transceive TxWait register |
| INTERNAL_USE | 0014h | 32 | R/W | 00000000h | For internal use |
| CLIF_RX_WAIT_REG | 0018h | 32 | R/W | 00000000h | Transceive RxWait register |
| INTERNAL_USE | 001Ch | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0020h | 32 | R/W | 00000000h | For internal use |
| CLIF_TX_WATERLEVEL_REG | 0024h | 32 | R/W | 00000000h | Buffer manager TX water level register |
| CLIF_RX_WATERLEVEL_REG | 0028h | 32 | R/W | 00000000h | Buffer Manager RX water level register |
| CLIF_RF_CONTROL_REG | 002Ch | 32 | R/W | 00000000h | RFControl control register |
| CLIF_TX_DATA_CONFIG_REG | 0030h | 32 | R/W | 00000000h | TxEncoder config register |
| CLIF_TX_DATA_MOD_REG | 0034h | 32 | R/W | 00000000h | TXEncoder config register |
| CLIF_TX_FRAME_CONFIG_REG | 0038h | 32 | R/W | 00000200h | TxEncoder config register |
| INTERNAL_USE | 003Ch | 32 | R/W | 00000000h | For internal use |
| CLIF_TX_SYMBOL0_DEF_REG | 0040h | 32 | R/W | 00000000h | TxEncoder config register |
| CLIF_TX_SYMBOL1_DEF_REG | 0044h | 32 | R/W | 00000000h | TxEncoder config register |
| CLIF_TX_SYMBOL23_DEF_REG | 0048h | 32 | R/W | 00000000h | TxEncoder config register |
| CLIF_TX_SYMBOL01_MOD_REG | 004Ch | 32 | R/W | 00000000h | TxEncoder config register |
| CLIF_TX_SYMBOL23_MOD_REG | 0050h | 32 | R/W | 00000000h | TXEncoder config register |
| CLIF_TX_OVERSHOOT_CONFIG_REG | 0054h | 32 | R/W | 00000000h | TXEncoder overshot prevention config register |
| CLIF_TX_UNDERSHOOT_CONFIG_REG | 0058h | 32 | R/W | 00000000h | TXEncoder undershot prevention config register |
| CLIF_RX_CONFIG_REG | 005Ch | 32 | R/W | 00000002h | RXDecoder registers |
| CLIF_RX_STATUS_REG | 0060h | 32 | R | 00000000h | RXDecoder registers |
| INTERNAL_USE | 0064h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0068h | 32 | R/W | 00000000h | For internal use |
| CLIF_CRC_RX_CONFIG_REG | 006Ch | 32 | R/W | 00000000h | CRC registers |
| CLIF_CRC_TX_CONFIG_REG | 0070h | 32 | R/W | 00000000h | CRC registers |
| INTERNAL_USE | 0074h | 32 | R/W | 00000000h | For internal use |
| CLIF_TIMER0_CONFIG_REG | 0078h | 32 | R/W | 00000000h | CLTimer register |
| CLIF_TIMER1_CONFIG_REG | 007Ch | 32 | R/W | 00000000h | CLTimer register |
| CLIF_TIMER2_CONFIG_REG | 0080h | 32 | R/W | 00000000h | CLTimer register |
| CLIF_TIMER0_RELOAD_REG | 0084h | 32 | R/W | 00000000h | CLTimer register |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CLIF_TIMER1_RELOAD_REG | 0088h | 32 | R/W | 00000000h | CLTimer register |
| CLIF_TIMER2_RELOAD_REG | 008Ch | 32 | R/W | 00000000h | CLTimer register |
| CLIF_TIMER0_OUTPUT_REG | 0090h | 32 | R | 00000000h | CLTimer register |
| CLIF_TIMER1_OUTPUT_REG | 0094h | 32 | R | 00000000h | CLTimer register |
| CLIF_TIMER2_OUTPUT_REG | 0098h | 32 | R | 00000000h | CLTimer register |
| CLIF_TIMER3_CONFIG_REG | 009Ch | 32 | R/W | 00000000h | CLTimer register |
| INTERNAL_USE | 00A0h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 00A4h | 32 | R | 00000000h | For internal use |
| INTERNAL_USE | 00A8h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 00ACh | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 00B0h | 32 | R/W | 00000000h | For internal use |
| CLIF_SIGPRO_RM_CONFIG1_REG | 00B4h | 32 | R/W | 00000000h | SigPro register |
| INTERNAL_USE | 00B8h | 32 | R/W | 00000004h | For internal use |
| INTERNAL_USE | 00BCh | 32 | R/W | 01000000h | For internal use |
| CLIF_SIGPRO_ADCBCM _THRESHOLD_REG | 00C0h | 32 | R/W | 00000000h | SigPro register |
| INTERNAL_USE | 00C4h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 00C8h | 32 | R/W | 00000000h | For internal use |
| CLIF_AGC_CONFIG0_REG | 00CCh | 32 | R/W | 00000000h | AGC register |
| INTERNAL_USE | 00D0h | 32 | R/W | 00000000h | For internal use |
| CLIF_AGC_INPUT_REG | 00D4h | 32 | R/W | 00000000h | AGC register |
| CLIF_RSSI_REG | 00D8h | 32 | R/W | 00000000h | AGC register |
| CLIF_TX_CONTROL_REG | 00DCh | 32 | R/W | 00000000h | TX control register |
| INTERNAL_USE | 00E0h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 00E4h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 00E8h | 32 | R | 00000000h | For internal use |
| CLIF_RX_DATA_BUFFER_REG | 00ECh | 32 | R | 00000000h | BufferManager Rx data register |
| CLIF_TEST_CONTROL_REG | 00FCh | 32 | R/W | 00000000h | Digital test control register |
| CLIF_ANA_NFCLD_REG | 0100h | 32 | R/W | 00000011h | NFC Level Detector control register |
| CLIF_ANA_TX_CLK_CONTROL_REG | 0104h | 32 | R/W | 00000083h | Analog TX clock control register |
| CLIF_ANA_TX_AMPLITUDE_REG | 0108h | 32 | R/W | 00000003h | Analog TX amplitude control register |
| INTERNAL_USE | 010Ch | 32 | R/W | 000000A0h | For internal use |
| CLIF_ANA_RX_REG | 0110h | 32 | R/W | 0003D820h | Analog RM receiver control register |
| CLIF_ANA_CM_CONFIG_REG | 0114h | 32 | R/W | 0000C080h | Analog CM control register |
| INTERNAL_USE | 0118h | 32 | R/W | 00010200h | For internal use |
| CLIF_ANA_AGC_REG | 011Ch | 32 | R/W | 00000004h | Analog AGC control register |
| CLIF_ANA_CLK_MAN_REG | 0120h | 32 | R/W | 00000000h | Analog clock management control register |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| INTERNAL_USE | 0124h | 32 | R/W | 00000000h | For internal use |
| CLIF_ANA_TX_SHAPE_CONTROL_REG | 0128h | 32 | R/W | 00000000h | Analog TX shaping control register |
| INTERNAL_USE | 012Ch | 32 | R/W | 00000000h | For internal use |
| CLIF_ANA_TEST_REG | 01FCh | 32 | R/W | 0050004Ah | Analog test control register |
| INTERNAL_USE | 0200h | 32 | R/W | FF000000h | For internal use |
| INTERNAL_USE | 0204h | 32 | R/W | 00000000h | For internal use |
| CLIF_DPLL_INIT_REG | 0208h | 32 | R/W | 00000000h | DPLL Configuration Register |
| INTERNAL_USE | 020Ch | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0210h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0214h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0218h | 32 | R/W | 00000000h | For internal use |
| CLIF_INT_CLR_ENABLE_REG | 3FD8h | 32 | W | 00000000h | Interrupt register |
| CLIF_INT_SET_ENABLE_REG | 3FDCh | 32 | W | 00000000h | Interrupt register |
| CLIF_INT_STATUS_REG | 3FE0h | 32 | R | 00000000h | Interrupt register |
| CLIF_INT_ENABLE_REG | 3FE4h | 32 | R | 00000000h | Interrupt register |
| CLIF_INT_CLR_STATUS_REG | 3FE8h | 32 | W | 00000000h | Interrupt register |
| CLIF_INT_SET_STATUS_REG | 3FECh | 32 | W | 00000000h | Interrupt register |

## 12.6 Register description

**Table 186. CLIF_CONTROL_REG register (address 0000h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | R | 0 | Reserved |
| 3 | START_SEND | D | 0*, 1 | Set to logic 1, the data transmission is started.<br>**Note:** This bit is only valid in combination with the transceive command<br>**Note:** If TXWait is set to a value other than zero the TXWait period configured must be expired as well that the transmission starts<br>As soon as the transmission started this bit is cleared by hardware. |
| 2:0 | COMMAND | D | 0 - 5 | This registers hold the command bits |
| | | | 0* | IDLE/StopCom Command; stops all ongoing communication and set the CLIF to IDLE mode; reset value |
| | | | 1 | Transmit command; starts a transmission immediately |
| | | | 2 | Receive command; enables the receiver. After end of reception the bits are clear and IDLE |
| | | | 3 | Transceive command; initiates a transceive cycle.<br>**Note:** Depending on the value of the Initiator bit a transmission is started or the receiver is enabled |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | **Note**: The transceive command does not finish automatically. It stays in the transceive cycle until stopped via the IDLE/ StopCom command |
| | | | 4 | KeepCommand command; This command does not change the content of the command register and might be used in case other bits in the register are to be changed |
| | | | 5 | LoopBack command; This command is for test purposes only. It starts a transmission and at the same enables the receiver. |
| | | | 6 | PRBS command: This command will start a transmission with the given protocol settings, transmitting a predefined pseudo-random data stream. The command will not return automatically. |
| | | | 7 | Reserved: Do not use this setting |

**Table 187. CLIF_CLOCK_CONTROL_REG register (address 0004h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:3 | RESERVED | R | 0 | reserved |
| 5 | FORCE_TEMP_CLK_ ON_RFOFF | R/W | 0*, 1 | If set, to 1, upon RFOFF event the clock is always switched to temporary-clock, no matter if pll-clock is available. |
| 4 | FORCE_PLL_CLOCK_ ON_TXACTIVE | R/W | 0*, 1 | If set, to 1, clock is automatically switched to DPLL clock for transmission |
| 3 | RESERVED | R | 0 | reserved |
| 2 | RELEASE_TEMP_CLOCK | W | 0*, 1 | Setting this register bit to 1 will release the temporary clock and switch back to functional mode.<br>**Note**: In normal operation, this bit must not be set. |
| 1:0 | CLIF_CLOCK_SELECT | R/W | 0* - 3 | This register controls the source of the CLIF clock (13.56 MHz contactless frequency clock) |
| | | | 0* | Automatic clock selection. The source of the clock is automatically chosen depending on the RF-Field status. If an external RF-Field is detected the<br>RF-clock is chosen otherwise the PLL-clock (independent if it is available) |
| | | | 1 | Force RF-clock. RF-clock is chosen as source, no automatic switching is performed. |
| | | | 2 | Force PLL-clock. PLL-clock is chosen as source, no automatic switching is performed. |
| | | | 3 | Force Temporary-clock. Temp-clock is chosen as source, no automatic switching is performed |

**Table 188. CLIF_STATUS_REG register (address 0008h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | RESERVED | R | 0 | reserved |
| 30 | DPLL_ENABLE | R | 0*, 1 | This bit indicates that the DPLL controller has enabled the DPLL |
| 29 | AGC_RFOFF_DET | R | 0*, 1 | This bit indicates than the AGC has detected the external RF-Field was switched off while transmitting in SL-ALM mode.<br>**Note**: Only valid if the detection mode is enabled with the register bit-field AGC_RF_DETECT_SEL. |
| 28 | CRC_OK | R | 0, 1 | This bit indicates the status of the actual CRC calculation. If 1 the CRC is correct, meaning the CRC register has the value 0 or the residue value if inverted CRC is used.<br>**Note**: This flag should only by evaluated at the end of a communication |
| 27 | SC_DETECTED | R | 0*, 1 | Status signal indicating that a subcarrier is detected |
| 26 | SOF_DETECTED | R | 0*, 1 | Status signal indicating that a SOF has been detected |
| 25 | TX_RF_STATUS | R | 0*, 1 | If set to 1 this bit indicates that the drivers are turned on, meaning an RF-Field is created by the device itself |
| 24 | RF_DET_STATUS | R | 0*, 1 | If set to 1 this bit indicates that an external RF-Field is detected by the rf level detectors (after digital filtering) |
| 23:16 | RESERVED | R | 0 | reserved |
| 15 | CLOCK_ERROR | R | 0, 1 | If set to 1 CLIF is operating on the temporary clock. |
| 14 | BMA_TRANSFER_ONGOING | R | 0, 1 | Status signal from Buffer Manager to indicate that a transfer is actually ongoing. |
| 13 | TX_READ_ERROR | R | 0*, 1 | |
| 12 | TX_LATENCY_ERROR | R | 0*, 1 | |
| 11 | TX_NO_DATA_ERROR | R | 0*, 1 | |
| 10:8 | RF_ACTIVE_ERROR_CAUSE | R | 0* - 5 | |
| | | | 0* | No Error; reset value |
| | | | 1 | External field was detected on within TIDT timing |
| | | | 2 | External field was detected on within TADT timing |
| | | | 3 | No external field was detected within TADT timings |
| | | | 4 | Peer did switch off RF Field without but no Rx event was raised (no data received) |
| | | | 5 - 7 | Reserved |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5 | RX_ENABLE | R | 0*, 1 | This bit indicates if the RXDecoder is enabled. If 1 the RXDecoder was enabled by the Transceive Unit and is now ready for data reception. |
| 4 | TX_ACTIVE | R | 0*, 1 | This bit indicates activity of the TXEncoder. If 1 a transmission is ongoing, otherwise the TXEncoder is in idle state. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 3 | RX_ACTIVE | R | 0*, 1 | This bit indicates activity of the RXDecoder. If 1 a data reception is ongoing, otherwise the RXDecoder is in idle state. |
| 2:0 | TRANSCEIVE_STATE | R | 0* - 5 | This registers hold the command bits |
| | | | 0* | IDLE state |
| | | | 1 | WaitTransmit state |
| | | | 2 | Transmitting state |
| | | | 3 | WaitReceive state |
| | | | 4 | WaitForData state |
| | | | 5 | Receiving state |
| | | | 6 | WaitLoopBack state |
| | | | 7 | LoopBack state |

**Table 189. CLIF_TRANSCEIVE_CONTROL_REG register (address 000Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | R | 0 | reserved |
| 29:24 | INTERNAL_USE[1] | R/W | 0* - 111111b | for internal use |
| | | | xxxxx1 | IDLE state enabled to trigger IRQ |
| | | | xxxx1x | WaitTransmit state enabled to trigger IRQ |
| | | | xxx1xx | Transmitting state enabled to trigger IRQ |
| | | | xx1xxx | WaitReceive state enabled to trigger IRQ |
| | | | x1xxxx | WaitForData state enabled to trigger IRQ |
| | | | 1xxxxx | Receiving state enabled to trigger IRQ |
| 23:18 | RESERVED | R | 0 | Reserved |
| 17 | INTERNAL_USE[1] | R/W | 0*, 1 | for internal use |
| 16 | INTERNAL_USE[1] | R/W | 0*, 1 | for internal use |
| 15:8 | TX_BITPHASE | R/W | 0* - FFh | Defines the number of 13.56 MHz cycles used for adjustment of TX_wait to meet the FDT. |
| 7 | RESERVED | R | 0 | reserved |
| 6 | INTERNAL_USE[1] | R/W | 0*, 1 | for internal use |
| 5 | INTERNAL_USE[1] | D | 0*, 1 | for internal use |
| 4 | INTERNAL_USE[1] | R/W | 0*, 1 | for internal use |
| 3 | INTERNAL_USE[1] | R/W | 0*, 1 | for internal use |
| 2 | RX_MULTIPLE_ENABLE | R/W | 0*, 1 | If this bit is set to 1, the receiver is re-activated after the end of a reception. A status byte is written to the RAM containing all relevant status information of the frame.<br>Note: Data in RAM is word aligned therefore empty bytes of a data Word in RAM are padded with 0x00 bytes. SW has to calculate the correct address for the following frame. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 1 | INTERNAL_USE[1] | D | 0*, 1 | for internal use |
| 0 | INITIATOR | R/W | 0*, 1 | Set to 1, the CLIF is configured for initiator mode. Depending on this setting the behavior of the transceive command is different |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 190. CLIF_TX_WAIT_REG register (address 0010h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R | 0 | Reserved |
| 27:8 | TX_WAIT_VALUE | D | 0* - FFFFFh | Defines the tx_wait timer reload value.<br>**Note:** If set to 00000h the tx_wait guard time is Disabled<br>**Note**: This bit is set by HW a protocol is detected in automatic mode detection |
| 7:0 | TX_WAIT-PRESCALER | D | 0*-FFh | Defines the prescaler reload value for the tx_wait timer.<br>**Note:** This bit is set by HW a protocol is detected in automatic mode detection |

**Table 191. CLIF_RX_WAIT_REG register (address 0018h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R | 0 | Reserved |
| 27:8 | RX_WAIT_VALUE | R/W | 0* - FFFFFh | Defines the rx_wait timer reload value.<br>**Note:** If set to 00000h the rx_wait guard time is Disabled |
| 7:0 | RX_WAIT-PRESCALER | R/W | 0*-FFh | Defines the prescaler reload value for the Rx_wait timer. |

**Table 192. CLIF_TX_WATERLEVEL_REG register (address 0024h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | R | 0 | Reserved |
| 7:0 | TX_WATERLEVEL | R/W | 0* - 0xFF | Defines a warning level to indicate that TX_WATERLEVEL number of words were already transmitted in the actual frame. When this level is reached the corresponding IRQ is set.<br>**Note**: 0 disables the water level<br>**Note**: In case a header byte offset is set the water level refers to word fetched from RAM |

**Table 193. CLIF_RX_WATERLEVEL_REG register (address 0028h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | R | 0 | Reserved |
| 7:0 | RX_WATERLEVEL | R/W | 0* - 0xFF | Defines a warning level to indicate that RX_WATERLEVEL number of words were already received in the actual frame. When this level is reached the corresponding IRQ is set.<br>**Note**: 0 disables the water level |

**Table 194. CLIF_RF_CONTROL_REG register (address 002Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R | 0 | Reserved |
| 4 | TX_RF_ENABLE | D | 0*, 1 | Set to 1, turning on the driver in reader or active mode is requested.<br>**Note**: According to the setting of InitialRFOn, AutoRFOn and CAOn the driver is turned after a defined time and depended on the presence of an external RF field.<br>**Note**: In case of an RFActiveError this bit is cleared by hardware |
| 3 | TX_COLL_AV_ENABLE | R/W | 0*, 1 | Set to 1, enables automatic collision avoidance. See ISO18092 for more details. |
| 2 | TX_INITIAL_RFON | D | 0*, 1 | Set to 1, the drivers are automatically turned on when no external field is present. In the case an external field is detected, the turning on the drivers is delay until the external field vanishes.<br>**Note**: The driver on procedure must be triggered by setting TX_RF_ENABLE.<br>**Note**: This bit is reset to 0 as soon as the drivers turned on. |
| 1 | TX_AUTO_RFON | R/W | 0*, 1 | Set to 1, the drivers are automatically turned on after a before present external field vanished.<br>**Note**: The driver on procedure must be triggered by setting TX_RF_ENABLE. |
| 0 | TX_AUTO_RFOFF | R/W | 0*, 1 | Set to 1, the drivers are automatically turned off after data is transmitted |

**Table 195. CLIF_TX_DATA_CONFIG_REG register (address 0030h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | R | 0 | Reserved |
| 24:16 | RESERVED | R/W | 0 - 104h | Reserved |
| 15 | RESERVED | R | 0 | Reserved |
| 14:12 | TX_LAST_BITS | R/W | 0*- 7 | Defines how many bits of the last data byte to be sent. If set to 000b all bits of the last data byte are sent. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | **Note**: Bits are skipped at the end of the byte |
| 11:9 | TX_FIRST_BITS | R/W | 0*- 7 | Defines how many bits of the first data byte to be sent.<br>If set to 000b all bits of the last data byte are sent.<br>**Note**: Bits are skipped at the beginning of the byte |
| 8:0 | TX_NUM_BYTES_2_SEND | R/W | 0 - 104h | Defines the number of bytes to be transmit. The maximum number of bytes is 260 (0x104).<br>If the Transmission Of Data Is Enabled (TX_DATA_ENABLE) and TX_NUM_BYTES_2_SEND is zero, then a NO_DATA_ERROR occurs. |

**Table 196. CLIF_TX_DATA_MOD_REG register (address 0034h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | R | 0 | Reserved |
| 23:16 | TX_DATA_MOD_WIDTH | R/W | 0* - FFh | Specifies the length of a pulse for sending data with miller pulse modulation enabled. The length is given by the number of carrier clocks + 1. |
| 15:9 | RESERVED | R | 0 | Reserved |
| 8 | TX_DATA_MILLER_ENABLE | R/W | 0*, 1 | If set to 1, pulse modulation is applied according to modified miller coding |
| 7 | TX_DATA_INV_ENV | R/W | 0*, 1 | If set to 1, the output envelope is inverted |
| 6:4 | TX_DATA_ENV_TYPE | R/W | 0*, 7h | Specifies the type of envelope used for transmission of data packets. The selected envelope type is applied to the pseudo bit stream. |
| | | | 000 | Direct output |
| | | | 001 | Manchester code |
| | | | 010 | Manchester code with subcarrier |
| | | | 011 | BPSK |
| | | | 100 | RZ (pulse of half bit length at beginning of second half of bit) |
| | | | 101 | RZ (pulse of half bit length at beginning of bit) |
| | | | 110 | Manchester tuple coding |
| | | | 111 | Reserved |
| 3 | TX_DATA_SC_FREQ | R/W | 0*, 1 | Specifies the frequency of the subcarrier. |
| | | | 0 | 424 kHz |
| | | | 1 | 848 kHz |
| 2:0 | TX_DATA_BIT_FREQ | R/W | 0* - 7h | Specifies the frequency of the bit-stream. |
| | | | 000 | 1.695 MHz |
| | | | 001 | 3.39 MHz |
| | | | 010 | 26 kHz |
| | | | 011 | 53 kHz |
| | | | 100 | 106 kHz |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | 101 | 212 kHz |
| | | | 110 | 424 kHz |
| | | | 111 | 848 kHz |

**Table 197. CLIF_TX_FRAME_CONFIG_REG register (address 0038h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:19 | RESERVED | R | 0 | Reserved |
| 18:16 | TX_DATA_CODE_TYPE | R/W | 0*- 7 | Specifies the type of encoding of data to be used |
| | | | 000b | No special code |
| | | | 001b | 1 out of 4 code [I-Code SLI] |
| | | | 010b | 1 out of 256 code [I-Code SLI] |
| | | | 011b | Pulse interval encoding (PIE) [I-Code EPC-V2] |
| | | | 100b | 2-bit tuple code (intended only for test purposes) |
| | | | 101-111b | Reserved |
| 15:13 | TX_STOPBIT_TYPE | R/W | 0*- 7 | Enables the stop bit (logic “1”) and extra guard time (logic “1”). The value 0 disables transmission of stop-bits. |
| | | | 000b | no stop-bit, no EGT |
| | | | 001b | stop-bit, no EGT |
| | | | 010b | stop-bit + 1 EGT |
| | | | 011b | stop-bit + 2 EGT |
| | | | 100b | stop-bit + 3 EGT |
| | | | 101b | stop-bit + 4 EGT |
| | | | 110b | stop-bit + 5 EGT |
| | | | 111b | stop-bit + 6 EGT |
| 12 | TX_STARTBIT_ENABLE | R/W | 0*- 1 | If set to 1, a start-bit (logic “0”) will be send |
| 11 | TX_MSB_FIRST | R/W | 0* | If set to 1, data bytes are interpreted MSB first for data transmission |
| 10 | TX_PARITY_LAST_INV_ENABLE | R/W | 0* | If set to 1, the parity bit of last byte (data or crc) is inverted |
| 9 | TX_PARITY_TYPE | R/W | 0*- 1 | Defines the type of the parity bit |
| | | | 0 | Even Parity is calculated |
| | | | 1 | Odd parity is calculated |
| 8 | TX_PARITY_ENABLE | R/W | 0*- 1 | If set to 1, a parity bit is calculated and appended to each byte transmitted.<br>If the Transmission Of Data Is Enabled and TX_NUM_BYTES_2_SEND is zero, then a NO_DATA_ERROR occurs. |
| 7:5 | RESERVED | R | 0 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 4 | TX_DATA_ENABLE | R/W | 0* | If set to 1, transmission of data is enabled otherwise only symbols are transmitted. |
| 3:2 | TX_STOP_SYMBOL | R/W | 0*- 3h | Defines which pattern symbol is sent as frame stop-symbol |
| | | | 00b | No symbol is sent |
| | | | 01b | Symbol 1 is sent |
| | | | 10b | Symbol 2 is sent |
| | | | 11b | Symbol 3 is sent |
| 1:0 | TX_START_SYMBOL | R/W | 0*-3h | Defines which symbol pattern is sent as frame start-symbol |
| | | | 00b | No symbol is sent |
| | | | 01b | Symbol 1 is sent |
| | | | 10b | Symbol 2 is sent |
| | | | 11b | Symbol 3 is sent |

**Table 198. CLIF_TX_SYMBOL0_DEF_REG register (address 0040h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R | 0 | Reserved |
| 27:0 | TX_SYMBOL0_DEF | R/W | 0* - FFFFFFFh | Pattern definition for Symbol0 |

**Table 199. CLIF_TX_SYMBOL1_DEF_REG register (address 0044h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R | 0 | Reserved |
| 27:0 | TX_SYMBOL1_DEF | R/W | 0* - FFFFFFFh | Pattern definition for Symbol1 |

**Table 200. CLIF_TX_SYMBOL23_DEF_REG register (address 0048h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | R | 0 | Reserved |
| 23:16 | TX_SYMBOL3_DEF | R/W | 0* - FFh | Pattern definition for Symbol3 |
| 15:8 | RESERVED | R | 0 | RESERVED |
| 7:0 | TX_SYMBOL2_DEF | R/W | 0* - FFh | Pattern definition for Symbol2 |

**Table 201. CLIF_TX_SYMBOL01_MOD_REG register (address 004Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | R | 0 | Reserved |
| 23:16 | TX_SYMBOL3_DEF | R/W | 0* - FFh | Pattern definition for Symbol3 |
| 15:9 | RESERVED | R | 0 | Reserved |
| 8 | TX_S01_MILLER_ENABLE | R/W | 0*, 1 | If set to 1, pulse modulation is applied according to modified miller coding.<br>**Note**: Modified miller coding may be used only for direct envelope output (TX_S01_ENV_TYPE == 0) |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 7 | TX_S01_INV_ENV | R/W | 0*, 1 | If set to 1, the output envelope is inverted. |
| 6:4 | TX_S01_ENV_TYPE | R/W | 0*, 7 | Specifies the type of envelope used for transmission of data packets. The selected envelope type is applied to the pseudo bit stream. |
| | | | 000b | Direct output |
| | | | 001b | Manchester code |
| | | | 010b | Manchester code with subcarrier |
| | | | 011b | BPSK |
| | | | 100b | RZ (pulse of half bit length at beginning of second half of bit) |
| | | | 101b | RZ (pulse of half bit length at beginning of bit) |
| | | | 110b | Manchester tupple |
| | | | 111b | Reserved |
| 3 | TX_S01_SC_FREQ | R/W | 0*, 1 | Specifies the frequency of the subcarrier. |
| | | | 0 | 424 kHz |
| | | | 1 | 848 kHz |
| 2:0 | TX_S01_BIT_FREQ | R/W | 000*-111 | Specifies the frequency of the bit-stream. |
| | | | 000* | 1.695 MHz |
| | | | 001 | Reserved for test |
| | | | 010 | 26 kHz |
| | | | 011 | 53 kHz |
| | | | 100 | 106 kHz |
| | | | 101 | 212 kHz |
| | | | 110 | 424 kHz |
| | | | 111 | 848 kHz |

**Table 202. CLIF_TX_SYMBOL23_MOD_REG register (address 0050h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:24 | RESERVED | R | 0 | Reserved |
| 23:16 | TX_S23_MODWIDTH | R/W | 0*-FFh | Specifies the length of a pulse for sending data of symbol 2/3. The length is given by the number of carrier clocks + 1. |
| 15:9 | RESERVED | R | 0 | Reserved |
| 8 | TX_S23_MILLER_ENABLE | R/W | 0*, 1 | If set to 1, pulse modulation is applied according to modified miller coding.<br>**Note:** Modified miller coding may be used for direct envelope output (TX_S23_ENV_TYPE ==0) |
| 7 | TX_S23_INV_ENV | R/W | 0*, 1 | If set to 1, the output envelope is inverted. |
| 6:4 | TX_S23_ENV_TYPE | R/W | 0* - 7h | Specifies the type of envelope used for transmission of data packets. The selected envelope type is applied to the pseudo bit stream. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | 000 | Direct output |
| | | | 001 | Manchester code |
| | | | 010 | Manchester code with subcarrier |
| | | | 011 | BPSK |
| | | | 100 | RZ (pulse of half bit length at beginning of second half of bit) |
| | | | 101 | RZ (pulse of half bit length at beginning of bit) |
| | | | 110 | Manchester tuple |
| | | | 111 | Reserved |
| 3 | TX_S23_SC_FREQ | R/W | 0*, 1 | Specifies the frequency of the subcarrier. |
| | | | 0 | 424 kHz |
| | | | 1 | 848 kHz |
| 2:0 | TX_S23_BIT_FREQ | R/W | 000* - 111 | Specifies the frequency of the bit-stream. |
| | | | 000* | 1.695 MHz |
| | | | 001 | 3.39 MHz |
| | | | 010 | 26 kHz |
| | | | 011 | 53 kHz |
| | | | 100 | 106 kHz |
| | | | 101 | 212 kHz |
| | | | 110 | 424 kHz |
| | | | 111 | 848 kHz |

**Table 203. CLIF_TX_OVERSHOOT_CONFIG_REG register (address 0054h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | TX_OVERSHOOT_PATTERN | R/W | 0* - FFFFh | Overshoot pattern which is transmitted after each rising edge. |
| 15:5 | RESERVED | R | 0 | Reserved |
| 4:1 | TX_OVERSHOOT_PATTERN_LEN | R/” | 0* - Fh | Defines length of the overshoot prevention pattern (value +1). The pattern is applied starting from the MSB of the defined pattern, all other bits are ignored. |
| 0 | TX_OVERSHOOT_PROT_ENABLE | R/W | 0*, 1 | If set to 1, the overshoot protection is enabled. |

**Table 204. CLIF_TX_UNDERSHOOT_CONFIG_REG register (address 0058h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | TX_UNDERSHOOT_PATTERN | R/W | 0* - FFFFh | Undershoot pattern which is transmitted after each rising edge. |
| 15:6 | RESERVED | R | 0 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 5 | TX_UNDERSHOOT_PROT_LAST_SC_ENABLE | R/W | 0* - 1 | This mode activates the undershoot prevention circuit only for the last sub-carrier cycle for card-mode transmission.<br>Note: The bit TX_UNDERSHOOT_PROT_ENABLE must not be set for this mode. |
| 4:1 | TX_UNDERSHOOT_PATTERN_LEN | R/” | 0* - Fh | Defines length of the undershoot prevention pattern (value +1). The pattern is applied starting from the MSB of the defined pattern, all other bits are ignored. |
| 0 | TX_UNDERSHOOT_PROT_ENABLE | R/W | 0*, 1 | If set to 1, the undershoot protection is enabled. |

**Table 205. CLIF_RX_CONFIG_REG register (address 005Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | RX_PARITY_EMD_ON_SO VER | R/W | 0*,1 | If set, decision if EMD due to parity error is taken at saver |
| 30 | RX_MISSING_PARITY_IS_EMD | R/W | 0*,1 | If set, a missing parity bit in the 4th byte is treated as - EMD (for EMD option 3/4 |
| 29 | RX_ADVANCED_EMD_ENABLE | R/W | 0*,1 | If set, new EMD options for PN7462AU are enabled |
| 28 | RX_PARITY_ERROR_IS_EMD | R/W | 0*, 1 | If set to 1 a parity error in the 3rd/4th byte (depending on RX_EMD_SUP setting) is interpreted as an EMD error. Otherwise it is interpreted as a parity error. |
| 27:25 | RX_EMD_SUP | R/W | 0*- 7h | Defines EMD suppression mechanism |
| | | | 000 | Off |
| | | | 001 | EMD suppression according to ISO14443 |
| | | | 010 | EMD suppression according to NFC Forum (with respect to the first 3 characters) |
| | | | 011 | EMD suppression according to NFC Forum (with respect to the first 4characters) |
| | | | 100 | EMD suppression according to NFC Forum (with respect to the first 4characters, all valid frames <4 bytes are ignored) |
| | | | 101 - 111 | reserved |
| 24 | RX_COLL_IS_DATA_ERROR | R/W | 0*, 1 | If set to 1, a collision is treated as a data integrity error (especially for ISO14443-4) |
| 23 | VALUES_AFTER_COLLISION | R/W | 0*, 1 | This bit defined the value of bits received after a collision occurred. |
| | | | 0 | All received bits after a collision will be cleared. |
| | | | 1 | All received bits after a collision keep their value. |
| 22 | RX_CRC_ALLOW_BITS | R/W | 0*, 1 | Set to 1, a frame with less than one byte length is written to ram, when the CRC is enabled otherwise it is discarded. |
| 21 | RX_FORCE_CRC_WRITE | R/W | 0*, 1 | Set to 1, the received CRC byte(s) are written to ram. In normal operation (if this bit is set to 0) CRC bytes are only checked and removed from the data stream. |
| 20 | RESERVED | R | 0 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 19 | RX_PARITY_ERR_8 BITS_E NABLE | R/W | 0*, 1 | If set, parity error is detected when only 8 bits in last byte received |
| 18:16 | RX_BIT_ALIGN | R/W | 0* - 7h | RxAlign defines the bit position within the byte for the first bit received. Further received bits are stored at the following bit positions. |
| 15:8 | RX_EOF_SYMBOL | D | 0* - FFh | This value defines the pattern of the EOF symbol. Note: All bits are clear upon enabling of the receiver if the automatic mode detection is enabled. If in such a case an ISO14443A communication is detected bit0 is set by the mod-detector. |
| 7 | RESERVED | R | 0 | Reserved |
| 6 | RX_EOF_TYPE | D | 0*, 1 | Defines the EOF handling, modified by the mode-detector |
| | | | 0 | EOF as defined in the register field RX_EOF_SYMBOL is expected. |
| | | | 1 | EOF as defined in ISO14443B is expected. |
| 5:4 | RX_STOP_CONDITI ON | D | 00b*-11b | This bit field defines the condition to stop a reception.<br>**Note**: This bit is set by the mod-detector if automatic mode detection is enabled and the corresponding communication is detected. |
| | | | 00b | Reception is stopped only if end of data communication is detected by the signal processing Note: This value is set by the mode-detector if a ISO14443A communication is detected |
| | | | 01b | Reception is stopped upon reception of the EOF pattern selected by the register field RX_EOF_TYPE and RX_EOF_SYMBOL<br>**Note**: This value is set by the mode-detector if an ISO14443B communication is detected |
| | | | 10b | Reception is stopped on detection of an incorrect parity bit. |
| | | | 11b | Reception is stopped if the number of received bytes reaches the defined frame length (which extracted from the first data-byte received).<br>**Note**: This value is set by the mod-detector if a FeliCa communication is detected |
| 3 | RX_MSB_FIRST | D | 0*, 1 | If set to 1, data bytes are interpreted MSB first for reception |
| 2 | RX_STOPBIT_ENAB LE | D | 0*, 1 | Set to 1, a stop-bit is expected and will be checked and extracted from data stream.<br>**Note**: The stop-bit is not configurable and fixed to a logic 1.<br>**Note**: This bit is set by the mod-detector if automatic mode detection is enabled and ISO14443B communication is detected. |
| 1 | RX_PARITY_TYPE | D | 0, 1* | Defines which type of the parity-bit is used<br>**Note**: This bit is set by the mod-detector if automatic mode detection is enabled and ISO14443A communication is detected. |
| | | | 0 | Even parity calculation is used |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | 1 | Odd parity calculation is used |
| 0 | RX_PARITY_ENABLE | D | 0*, 1 | If set to 1, a parity-bit for each byte is expected, will be extracted from data stream and checked for correctness. In case the parity-bit is incorrect, the RX_DATA_INTEGRITY_ERROR flag is set.<br>Nevertheless, the reception is continued.<br>**Note**: This bit is set by the mod-detector if automatic mode detection is enabled and ISO14443A communication is detected. |

**Table 206. CLIF_RX_STATUS_REG register (address 0060h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | RESERVED | R | 0 | Reserved |
| 30:24 | RX_COLL_POS | R | 0* - 7Fh | These bits show the bit position of the first detected collision in a received frame (only data bits are interpreted).<br>**Note**: These bits shall only be interpreted in passive communication mode at 106 kbit/s or ISO14443A /MIFARE® reader / writer or ICODE SLIX reader/ writer mode if bit CollPosValid is set to 1.<br>**Note**: If RX_ALIGN is set to a value different to 0, this value is included in the RX_COLL_POS. |
| 23 | RX_WRITE_ERROR | R | 0*, 1 | This error flag is set to 1, if for an ongoing reception data is not copied to RAM in time (BMA encountered write error) and therefor the reception is aborted |
| 22 | RX_LATENCY_ERROR | R | 0*, 1 | This error flag is set to 1, if for an ongoing reception data is not copied to RAM in time (BMA latency to big) and therefor the reception is aborted |
| 21 | RX_BUFFER_OVFL_ERROR | R | 0*, 1 | This flag is set to 1 when the number of received bytes exceeds the size of the RXBuffer |
| 20 | RX_CL_ERROR | R | 0*, 1 | General contactless error, is a combination of data integrity, protocol and collision error |
| 19 | RX_LEN_ERROR | R | 0*, 1 | This flag is set to 1, if RXMultiple is activate and the LEN byte indicates more than 28 bytes |
| 18 | RX_COLLISION_DETECTED | R | 0*, 1 | This flag is set to 1, when a collision has accrued. The position of the first collision is shown in the register RX_COLLPOS |
| 17 | RX_PROTOCOL_ERROR | R | 0*, 1 | This flag is set to 1, when a protocol error has accrued. A protocol error can be a wrong stop bit, a missing or wrong ISO14443B EOF or SOF or a wrong number of received data bytes.<br>**Note**: When a protocol error is detected, data reception is stopped.<br>**Note**: The flag is automatically cleared at start of next reception. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 16 | RX_DATA_INTEGRITY_ERROR | R | 0*, 1 | This flag is set to 1, if a data integrity error has been detected. Possible caused can be a wrong parity or a wrong CRC.<br>**Note**: On a data integrity error the reception is continued<br>**Note**: The flag is automatically cleared at start of next reception.<br>**Note**: If a reversed parity bit is a stop criteria, the flag is not set to 1 in case of a wrong parity. |
| 15:13 | RX_NUM_LAST_BITS | R | 0* - 7h | Defines the number of valid bits of the last data byte received in bit-oriented communications. If zero, the whole byte is valid. |
| 12:9 | RX_NUM_FRAMES_RECEIVED | R | 0* - 8h | Indicates the number of frames received. The value is updated when the RxIRQ is raised.<br>**Note**: This bit field is only valid when the RxMultiple is active (bit RX_MULTIPLE_ENABLE set) |
| 8:0 | RX_NUM_BYTES_RECEIVED | R | 0* - 104h | Indicates the number of bytes received. The value is valid when the RxIRQ is raised until the receiver is enabled again. |

**Table 207. CLIF_CRC_RX_ CONFIG_REG register (address 006Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RX_CRC_PRESET_VALUE | R/W | 0*- FFFFh | Arbitrary preset value for the Rx-Decoder CRC calculation. |
| 15:6 | RESERVED | R | 0 | Reserved |
| 5:3 | RX_CRC_PRESET_SEL | D | 000-101b | Preset value of the CRC register for the Rx-Decoder. For a CRC calculation using 5-bits, only the LSByte is used. |
| | | | 000b* | 0000h, reset value<br>**Note:** That this configuration is set by the Mode detector for FeliCa. |
| | | | 001b | 6363h<br>**Note**: That this configuration is set by the Mode detector for ISO14443 type A. |
| | | | 010b | A671h |
| | | | 011b | FFFFh<br>**Note**: That this configuration is set by the Mode detector for ISO14443 type B. |
| | | | 100b | 0012h |
| | | | 101b | E012h |
| | | | 110b | Reserved |
| | | | 111b | Use arbitrary preset value RX_CRC_PRESET_VALUE |
| 2 | RX_CRC_TYPE | R/W | 0, 1 | Controls the type of CRC calculation for the Rx-Decoder. |
| | | | 0* | 16-bit CRC calculation, reset value |
| | | | 1 | 5-bit CRC calculation |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 1 | RX_CRC_INV | D | 0, 1 | Controls the comparison of the CRC checksum for the Rx-Decoder |
| | | | 0* | Not inverted CRC value: 0000h, reset value<br>**Note**: That this nit is cleared by the Mode detector for ISO14443 type A and FeliCa. |
| | | | 1 | Inverted CRC value: F0B8h<br>**Note**: That this bit is set by the Mode detector for ISO14443 type B. |
| 0 | RX_CRC_ENABLE | D | 0*, 1 | If set, the Rx-Decoder will check the CRC for correctness.<br>**Note**: That this bit is set by the Mode Detector, when ISO14443 type B, or FeliCa (212 or 424kBd) is detected. |

**Table 208. CLIF_CRC_TX_ CONFIG_REG register (address 0070h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | TX_CRC_PRESET_VALUE | R/W | 0*- FFFFh | Arbitrary preset value for the Tx-Encoder CRC calculation. |
| 15:7 | RESERVED | R | 0 | Reserved |
| 6 | TX_CRC_BYTE2_ENABLE | R/W | 0*, 1 | If set, the CRC is calculated from the 2nd byte onwards (intended for HID).<br>Note: That this option is used in the Tx-Encoder. |
| 5:3 | TX_CRC_PRESET_SEL | R/W | 000-101b | Preset value of the CRC register for the Tx-Encoder. For a CRC calculation using 5bits, only the LSByte is used. |
| | | | 000b* | 0000h, reset value |
| | | | 001b | 6363h |
| | | | 010b | A671h |
| | | | 011b | FFFFh |
| | | | 100b | 0012h |
| | | | 101b | E012h |
| | | | 110b | reserved |
| | | | 111b | Use arbitrary preset value TX_CRC_PRESET_VALUE |
| 2 | TX_CRC_TYPE | R/W | 0, 1 | Controls the type of CRC calculation for the Tx-Encoder |
| | | | 0* | 16-bit CRC calculation, reset value |
| | | | 1 | 5-bit CRC calculation |
| 1 | TX_CRC_INV | R/W | 0, 1 | Controls the sending of an inverted CRC value by the Tx-Encoder |
| | | | 0* | Controls the sending of an inverted CRC value by the Tx-Encoder |
| | | | 1 | Inverted CRC checksum |
| 0 | TX_CRC_ENABLE | R/W | 0*, 1 | If set to one, the Tx-Encoder will compute and transmit a CRC. |

**Table 209. CLIF_TIMER0_CONFIG_REG register (address 0078h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | T0_STOP_ON_T3_EXPIRED | R/W | 0*, 1 | T0_STOP_EVENT: If set the timer T0 is stopped when timer T3 raises its expiration flag. |
| 30 | T0_STOP_ON_RX_STARTED | R/W | 0*, 1 | T0_STOP_EVENT: If set the timer T0 is stopped when a data reception begins (1st bit is received). |
| 29 | T0_STOP_ON_TX_STARTED | R/W | 0*, 1 | T0_STOP_EVENT: If set the timer T0 is stopped when a data transmission begins. |
| 28 | T0_STOP_ON_RF_ON_EXT | R/W | 0*, 1 | T0_STOP_EVENT: If set the timer T0 is stopped when the external RF field is detected. |
| 27 | T0_STOP_ON_RF_OFF_EXT | R/W | 0*, 1 | T0_STOP_EVENT: If set the timer T0 is stopped when the external RF field vanishes. |
| 26 | T0_STOP_ON_RF_ON_INT | R/W | 0*,1 | T0_STOP_EVENT: If set, the timer T0 is stopped when the internal RF field is turned on. |
| 25 | T0_STOP_ON_RF_OFF_INT | R/W | 0*,1 | T0_STOP_EVENT: If set, the timer T0 is stopped when the internal RF field is turned off. |
| 24 | T0_STOP_ON_SIGIN_ACT | R/W | 0*,1 | T0_STOP_EVENT: If set, the timer T0 is stopped when an activity on SigIn is detected. |
| 23:19 | RESERVED | R | 0 | Reserved |
| 18 | T0_START_ON_T3_RUNNING | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when the timer T3 starts running. |
| 17 | T0_START_ON_RX_STARTED | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when a data reception begins (1st bit is received). |
| 16 | T0_START_ON_RX_ENDED | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when a data reception ends. |
| 15 | T0_START_ON_TX_STARTED | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when a data transmission begins. |
| 14 | T0_START_ON_TX_ENDED | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when a data transmission ends. |
| 13 | T0_START_ON_RF_ON_EXT | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when the external RF field is detected. |
| 12 | T0_START_ON_RF_OFF_EXT | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when the external RF field is not detected any more. |
| 11 | T0_START_ON_RF_ON_INT | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when an internal RF field is turned on. |
| 10 | T0_START_ON_RF_OFF_INT | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when an internal RF field is turned off. |
| 9 | T0_START_ON_SIGIN_ACT | R/W | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started when an activity on SigIn is detected. |
| 8 | T0_START_NOW | D | 0*, 1 | T0_START_EVENT: If set, the timer T0 is started immediately. |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5:3 | T0_PRESCALE_SEL | R/W | 0h-7h | Controls frequency/period of the timer T0 when the prescaler is activated in T0_MODE_SEL. |
| | | | 000 | 6.78 MHz counter |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | 001 | 3.39 MHz counter |
| | | | 010 | 1.70 MHz counter |
| | | | 011 | 848 kHz counter |
| | | | 100 | 424 kHz counter |
| | | | 101 | 212 kHz counter |
| | | | 110 | 106 kHz counter |
| | | | 111 | 53 kHz counter |
| 2 | T0_MODE_SEL | R/W | 0 – 1 | Configuration of the timer T0 clock. |
| | | | 0* | Prescaler is disabled: the timer frequency matches CLIF clock frequency (13.56 MHz). |
| | | | 1 | Prescaler is enabled: the timer operates on the prescaler signal frequency (chosen by T0_PRESCALE_SEL). |
| 1 | T0_RELOAD_ENABLE | R/W | 0*, 1 | If set to 0, the timer T0 will stop on expiration. |
| | | | 0* | After expiration, the timer T0 will stop counting, i.e., remain zero, reset value. |
| | | | 1 | After expiration, the timer T0 will reload its preset value and continue counting down. |
| 0 | T0_ENABLE | R/W | 0*, 1 | Enables the timer T0 |

**Table 210. CLIF_TIMER1_CONFIG_REG register (address 007Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | T1_STOP_ON_T3_EXPIRED | R/W | 0*, 1 | T1_STOP_EVENT: If set the timer T1 is stopped when timer T3 raises its expiration flag. |
| 30 | T1_STOP_ON_RX_STARTED | R/W | 0*, 1 | T1_STOP_EVENT: If set the timer T1 is stopped when a data reception begins (1st bit is received). |
| 29 | T1_STOP_ON_TX_STARTED | R/W | 0*, 1 | T1_STOP_EVENT: If set the timer T1 is stopped when a data transmission begins. |
| 28 | T1_STOP_ON_RF_ON_EXT | R/W | 0*, 1 | T1_STOP_EVENT: If set the timer T1 is stopped when the external RF field is detected. |
| 27 | T1_STOP_ON_RF_OFF_EXT | R/W | 0*, 1 | T1_STOP_EVENT: If set the timer T1 is stopped when the external RF field vanishes. |
| 26 | T1_STOP_ON_RF_ON_INT | R/W | 0*,1 | T1_STOP_EVENT: If set, the timer T1 is stopped when the internal RF field is turned on. |
| 25 | T1_STOP_ON_RF_OFF_INT | R/W | 0*,1 | T1_STOP_EVENT: If set, the timer T1 is stopped when the internal RF field is turned off. |
| 24 | T1_STOP_ON_SIGIN_ACT | R/W | 0*,1 | T1_STOP_EVENT: If set, the timer T1 is stopped when an activity on SigIn is detected. |
| 23:19 | RESERVED | R | 0 | Reserved |
| 18 | T1_START_ON_T3_RUNNING | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when the timer T3 starts running. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17 | T1_START_ON_RX_STARTED | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when a data reception begins (1st bit is received). |
| 16 | T1_START_ON_RX_ENDED | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when a data reception ends. |
| 15 | T1_START_ON_TX_STARTED | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when a data transmission begins. |
| 14 | T1_START_ON_TX_ENDED | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when a data transmission ends. |
| 13 | T1_START_ON_RF_ON_EXT | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when the external RF field is detected. |
| 12 | T1_START_ON_RF_OFF_EXT | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when the external RF field is not detected any more. |
| 11 | T1_START_ON_RF_ON_INT | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when an internal RF field is turned on. |
| 10 | T1_START_ON_RF_OFF_INT | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when an internal RF field is turned off. |
| 9 | T1_START_ON_SIGIN_ACT | R/W | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started when an activity on SigIn is detected. |
| 8 | T1_START_NOW | D | 0*, 1 | T1_START_EVENT: If set, the timer T1 is started immediately. |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5:3 | T1_PRESCALE_SEL | R/W | 0h-7h | Controls frequency/period of the timer T1 when the prescaler is activated in T1_MODE_SEL. |
| | | | 000 | 6.78 MHz counter |
| | | | 001 | 3.39 MHz counter |
| | | | 010 | 1.70 MHz counter |
| | | | 011 | 848 kHz counter |
| | | | 100 | 424 kHz counter |
| | | | 101 | 212 kHz counter |
| | | | 110 | 106 kHz counter |
| | | | 111 | 53 kHz counter |
| 2 | T1_MODE_SEL | R/W | 0 – 1 | Configuration of the timer T1 clock. |
| | | | 0* | Prescaler is disabled: the timer frequency matches CLIF clock frequency (13.56MHz). |
| | | | 1 | Prescaler is enabled: the timer operates on the prescaler signal frequency (chosen by T1_PRESCALE_SEL). |
| 1 | T1_RELOAD_ENABLE | R/W | 0*, 1 | If set to 0, the timer T1 will stop on expiration. |
| | | | 0* | After expiration, the timer T1 will stop counting, i.e., remain zero, reset value. |
| | | | 1 | After expiration, the timer T1 will reload its preset value and continue counting down. |
| 0 | T1_ENABLE | R/W | 0*, 1 | Enables the timer T1 |

**Table 211. CLIF_TIMER2_CONFIG_REG register (address 0080h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | T2_STOP_ON_T3_EXPIRED | R/W | 0*, 1 | T2_STOP_EVENT: If set the timer T2 is stopped when timer T3 raises its expiration flag. |
| 30 | T2_STOP_ON_RX_STARTED | R/W | 0*, 1 | T2_STOP_EVENT: If set the timer T2 is stopped when a data reception begins (1st bit is received). |
| 29 | T2_STOP_ON_TX_STARTED | R/W | 0*, 1 | T2_STOP_EVENT: If set the timer T2 is stopped when a data transmission begins. |
| 28 | T2_STOP_ON_RF_ON_EXT | R/W | 0*, 1 | T2_STOP_EVENT: If set the timer T2 is stopped when the external RF field is detected. |
| 27 | T2_STOP_ON_RF_OFF_EXT | R/W | 0*, 1 | T2_STOP_EVENT: If set the timer T2 is stopped when the external RF field vanishes. |
| 26 | T2_STOP_ON_RF_ON_INT | R/W | 0*,1 | T2_STOP_EVENT: If set, the timer T2 is stopped when the internal RF field is turned on. |
| 25 | T2_STOP_ON_RF_OFF_INT | R/W | 0*,1 | T2_STOP_EVENT: If set, the timer T2 is stopped when the internal RF field is turned off. |
| 24 | T2_STOP_ON_SIGIN_ACT | R/W | 0*,1 | T2_STOP_EVENT: If set, the timer T2 is stopped when an activity on SigIn is detected. |
| 23:19 | RESERVED | R | 0 | Reserved |
| 18 | T2_START_ON_T3_RUNNING | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when the timer T3 starts running. |
| 17 | T2_START_ON_RX_STARTED | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when a data reception begins (1st bit is received). |
| 16 | T2_START_ON_RX_ENDED | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when a data reception ends. |
| 15 | T2_START_ON_TX_STARTED | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when a data transmission begins. |
| 14 | T2_START_ON_TX_ENDED | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when a data transmission ends. |
| 13 | T2_START_ON_RF_ON_EXT | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when the external RF field is detected. |
| 12 | T2_START_ON_RF_OFF_EXT | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when the external RF field is not detected any more. |
| 11 | T2_START_ON_RF_ON_INT | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when an internal RF field is turned on. |
| 10 | T2_START_ON_RF_OFF_INT | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when an internal RF field is turned off. |
| 9 | T2_START_ON_SIGIN_ACT | R/W | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started when an activity on SigIn is detected. |
| 8 | T2_START_NOW | D | 0*, 1 | T2_START_EVENT: If set, the timer T2 is started immediately. |
| 7:6 | RESERVED | R | 0 | reserved |
| 5:3 | T2_PRESCALE_SEL | R/W | 0h-7h | Controls frequency/period of the timer T2 when the prescaler is activated in T2_MODE_SEL. |
| | | | 000 | 6.78 MHz counter |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | 001 | 3.39 MHz counter |
| | | | 010 | 1.70 MHz counter |
| | | | 011 | 848 kHz counter |
| | | | 100 | 424 kHz counter |
| | | | 101 | 212 kHz counter |
| | | | 110 | 106 kHz counter |
| | | | 111 | 53 kHz counter |
| 2 | T2_MODE_SEL | R/W | 0 – 1 | Configuration of the timer T2 clock. |
| | | | 0* | Prescaler is disabled: the timer frequency matches CLIF clock frequency (13.56MHz). |
| | | | 1 | Prescaler is enabled: the timer operates on the prescaler signal frequency (chosen by T2_PRESCALE_SEL). |
| 1 | T2_RELOAD_ENABLE | R/W | 0*, 1 | If set to 0, the timer T2 will stop on expiration. |
| | | | 0* | After expiration, the timer T2 will stop counting, i.e., remain zero, reset value. |
| | | | 1 | After expiration, the timer T2 will reload its preset value and continue counting down. |
| 0 | T2_ENABLE | R/W | 0*, 1 | Enables the timer T2 |

**Table 212. CLIF_TIMER0_RELOAD_REG register (address 0084h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:20 | RESERVED | R | 0 | Reserved |
| 19:0 | T0_RELOAD_VALUE | R/W | 0000h - FFFFFh | Reload value of the timer T0. |
| | | | 00h* | reset value |

**Table 213. CLIF_TIMER1_RELOAD_REG register (address 0088h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:20 | RESERVED | R | 0 | Reserved |
| 19:0 | T1_RELOAD_VALUE | R/W | 0000h - FFFFFh | Reload value of the timer T1. |
| | | | 00h* | reset value |

**Table 214. CLIF_TIMER2_RELOAD_REG register (address 008Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:20 | RESERVED | R | 0 | Reserved |
| 19:0 | T2_RELOAD_VALUE | R/W | 0000h - FFFFFh | Reload value of the timer T2. |
| | | | 00h* | reset value |

**Table 215. CLIF_TIMER0_OUTPUT_REG register (address 0090h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | R | 0 | Reserved |
| 24 | T0_RUNNING | R | 0*, 1 | Indicates that timer T0 is running (busy) |
| 23:20 | RESERVED | R | 0 | Reserved |
| 19:0 | T0_VALUE | R | 00000h - FFFFFh | Value of 20-bit counter in timer T0 |
| | | | 00h* | reset value |

**Table 216. CLIF_TIMER1_OUTPUT_REG register (address 0094h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | R | 0 | Reserved |
| 24 | T1_RUNNING | R | 0*, 1 | Indicates that timer T1 is running (busy) |
| 23:20 | RESERVED | R | 0 | Reserved |
| 19:0 | T1_VALUE | R | 00000h - FFFFFh | Value of 20-bit counter in timer T1 |
| | | | 00h* | reset value |

**Table 217. CLIF_TIMER2_OUTPUT_REG register (address 0098h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | R | 0 | Reserved |
| 24 | T2_RUNNING | R | 0*, 1 | Indicates that timer T2 is running (busy) |
| 23:20 | RESERVED | R | 0 | Reserved |
| 19:0 | T2_VALUE | R | 00000h - FFFFFh | Value of 20-bit counter in timer T2 |
| | | | 00h* | reset value |

**Table 218. CLIF_TIMER3_CONFIG_REG register (address 009Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | T3_RUNNING | R | 0*, 1 | Indicates that timer T3 is running (busy) |
| 30:28 | RESERVED | R | 0*, 1 | Reserved |
| 27:8 | T3_START_VALUE | R/W | Oh*- FFFFFh | Start value of timer T3 |
| 7:3 | RESERVED | R | 0 | Reserved |
| 2 | T3_START_NOW | D | 0*, 1 | Start value of timer T3 |
| 1 | T3_CLOCK_SEL | R/W | 0*, 1 | Select the timer clock frequency |
| | | | 0 | HFO clock |
| | | | 1 | LFO clock |
| 0 | T3_ENABLE | R/W | 0*, 1 | Enables timer T3 |

**Table 219. CLIF_SIGPRO_RM_CONFIG1_REG register (address 00B4h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:25 | RESERVED | R | 0 | Reserved |
| 24 | DYNAMIC_BPSK_TH_ENABLE | R/W | 0-1* | If set to 1 the threshold for BPSK demodulation is dynamically adapted while reception.<br>Note: This setting is only valid in BPSK modes – otherwise it is ignored |
| 23:21 | BPSK_IQ_MODE | R/W | 000*-111 | Defines strategy of signal processing regarding I and Q channel |
| | | | 000* | Both channels (I and Q) are used for signal processing |
| | | | 001 | Use only I channel |
| | | | 010 | Use only Q channel |
| | | | 011 | Reserved |
| | | | 100 | Use the strongest channel |
| | | | 101 | Use the first channel |
| | | | 110-111 | Reserved |
| 20 | BPSK_FILT6 | R/W | 0*-1 | This bit changes the Spike filter for the Phase shift detection from order 5 to order 6 (default = order 6). |
| 19 | RESYNC_EQ_ON | R/W | 0-1* | Resynchronization during the SOF for an equal correlation value is done (default = activated). |
| 18 | CORR_RESET_ON | R/W | 0-1* | The correlator is reset at a reset (default = activated). |
| 17 | VALID_FILT_OFF | R/W | 0*-1 | Disables a special filter in BPSK mode. If set to 0, the correlation of 0110 is filtered with the correlation of 1110 and 0111. Otherwise the demodulation is done using the correlation with 0110 |
| 16 | DATA_BEFORE_MIN | R/W | 0*-1 | Data is received even before the first minimum at the SOF (default: = deactivated). |
| 15:12 | MIN_LEVEL | R/W | 0*-Fh | Defines the MinLevel of the reception<br>**Note**: The MinLevel should be higher than the noise level in the system<br>**Note**: Used for BPSK and Manchester with Subcarrier |
| 11:8 | MIN_LEVEL_P | R/W | 0* - Fh | Defines the MinLevel for the phase shift detector unit |
| 7 | USE_SMALL_EVAL | R/W | 0* - 1 | Defines the length of the even period for the correlator for Manchester subcarrier communication types |
| 6:5 | COLL_LEVEL | R/W | 00* - 11 | Defines how strong a signal must be to be interpreted as a collision for Manchester subcarrier communication types. |
| | | | 00* | >12.5 % |
| | | | 01 | >25 % |
| | | | 10 | >50 % |
| | | | 11 | No Collison |
| 4 | PRE_FILTER | R/W | 0* - 1 | If set to 1 four samples are combined to one data. (average) |
| 3 | RECT_FILTER | R/W | 0* - 1 | If set to one, the ADC-values are changed to a more rectangular wave shape. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 2 | SYNC_HIGH | R/W | 0* - 1 | Defines if the bit grid is fixed at maximum (1) or at a minimum (0) value of the correlation. |
| 1 | FSK | R/W | 0*-1 | If set to 1, the demodulation scheme is FSK. |
| 0 | BPSK | R/W | 0*-1 | If set to 1, the demodulation scheme is BPSK. |

**Table 220. CLIF_SIGPRO_ADCBCM_THRESHOLD_REG register (address 00C0h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R | 0 | Reserved |
| 28:16 | EDGE_DETECT_TH | R/W | 0000h* - 1FFFFh | Threshold for the edge decision block of the ADCBCM |
| 15:13 | RESERVED | R | 0 | Reserved |
| 12:0 | BIT_DETECT_TH | R/W | 0000h* - 1FFFFh | Threshold for the “bit” decision block of the ADCBCM. |

**Table 221. CLIF_AGC_CONFIG0_REG register (address 00CCh)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | INTERNAL_USE[1] | R/W | 0 | For internal use |
| 30:24 | FOR INTERAL USE[1] | R/W | 0*-7Fh | For internal use |
| 23:15 | INTERNAL_USE[1] | R/W | 0*-1FFh | For internal use |
| 14:5 | AGC_TIME_CONSTANT | R/W | 0*-3FFh | Time constant for the AGC update. An AGC period is given by (AGC_TIME_CONSTANT+1) * 13.56 MHz |
| 4 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 3 | AGC_INPUT_SEL | R/W | 0*, 1 | Selects the AGC value to be loaded into the AGC and the source for manual mode: |
| 2 | AGC_LOAD | W | 0*, 1 | If set, one AGC control value is loaded from |
| 1 | AGC_MODE_SEL | R/W | 0*, 1 | Selects the operation mode of the AGC: |
| | | | 0* | Rx-Divider is controlled by the register CLIF_AGC_INPUT_REG.AGC_CM_VALUE or CLIF_AGC_INPUT_REG.AGC_RM_VALUE (Dependent on AGC_INPUT_SEL). |
| | | | 1 | Rx-Divider value is controlled by the AGC. |
| 0 | AGC_MODE_ENABLE | R/W | 0*-1 | If set, the AGC is enabled. If not set, the Rx-Divider is controlled by either the internal AGC register or a register value (dependent on AGC_MODE_SEL). |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 222. CLIF_AGC_INPUT_REG register (address 00D4h)**
*** = reset value**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0 | Reserved |
| 25:16 | AGC_RM_VALUE | R/W | 0*-3FFh | Static AGC value used for reader mode CLIF_AGC_CONFIG0_REG.AGC_INPUT_SEL = 1 & CLIF_AGC_CONFIG0_REG.MODE_SEL = 0 |
| | | | 0h* | Most sensitive: largest Rx-resistor, i.e., none of the switchable resistors is added in parallel |
| | | | 3FFh | Most robust: smallest Rx-resistor, i.e., all switchable resistors are added in parallel |
| 15:10 | RESERVED | R | 0 | Reserved |
| 9:0 | AGC_CM_VALUE | R/W | 0*-3FFh | Static AGC value used for card mode<br>CLIF_AGC_CONFIG0_REG.AGC_INPUT_SEL = 0 & CLIF_AGC_CONFIG0_REG.MODE_SEL = 0.<br>**Note**: That in reset CLIF_AGC_CONFIG0_REG = 0x00 and CLIF_AGC_INPUT_REG = 0x00 and therefore the AGC output is AGC_CM_VALUE = 0x00. |
| | | | 0h* | Most sensitive: largest Rx-resistor, i.e., none of the switchable resistors is added in parallel |
| | | | 3FFh | Most robust: smallest Rx-resistor, i.e., all switchable resistors are added in parallel |

**Table 223. CLIF_RSSI_REG register (address 00D8h)**
** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | R | 0 | Reserved |
| 15:6 | AGC_VALUE | R | 0*-3FFh | RSSI: Current value of the AGC |
| | | | 0h* | Most sensitive: largest Rx-resistor, i.e., none of the switchable resistors are added in parallel |
| | | | 3FFH | Most robust: smallest Rx-resistor, i.e., all switchable resistors are added in parallel |
| 5:0 | ADC_DATA_Q | R | 0h* - 3Fh | RSSI: Q-channel ADC value (for RSSI DC coupling shall be used) |

**Table 224. CLIF_TX_CONTROL_REG register (address 00DCh)**
** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:18 | INTERNAL_USE[1] | R | 0 | For internal use |
| 17 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 16 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 | INTERNAL_USE[1] | R | 0 | For internal use |
| 14:13 | INTERNAL_USE[1] | R/W | 0*, 3 | For internal use |
| 12 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 11 | INTERNAL_USE[1] | R | 0 | For internal use |
| 10 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 9 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 8 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 7 | INTERNAL_USE[1] | R | 0 | For internal use |
| 6 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 5 | TX_ALM_TYPE_SELECT | R/W | 0*, 1 | Defines which ALM type is used. If set to 1 Dual-Loop-ALM is used, default (0) is Single-Loop-ALM. |
| 4 | INTERNAL_USE[1] | R/W | 0*, 1 | Selects the driver to be used for ALM.<br>For internal use |
| 3 | TX_ALM_ENABLE | R/W | 0*, 1 | If set to 1 ALM is used for transmission in card mode |
| 2 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 1:0 | INTERNAL_USE[1] | R/W | 0* - 3h | For internal use |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 225. CLIF_RX_DATA_BUFFER_REG register (address 00ECh)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | RX_DATA_BUFFER | R | 0h-FFFFFFFFh | Data buffer for APB based Buffer Manager data transfer. |

**Table 226. CLIF_TEST_CONTROL_REG register (address 00FCh)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 30 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 29 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 28:27 | INTERNAL_USE[1] | R | 0 | For internal use |
| 26 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 25 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 24 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 23 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 22:21 | TADT_EXTEND_SELECT | R/W | 0*, 3 | By setting a value != 0 the minimum time for $T_{ADT}$ can be increased |
| 20 | PRBS_TYPE | R/W | 0*, 1 | Defines the PRBS type; If set to 1, PRBS15 is selected, default value 0 selects PRBS9 |
| 19 | LONG_LD_MASK_ENABLE | R/W | 0*, 1 | If set to 1, the masking time for the NFC-LD after TxRFOff is longer. |
| 18 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 16 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 15 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 14 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 13 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 12 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 11 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 10 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 9:8 | RFDET_SOURCE_SEL | R/W | 0*- 3 | Select the source for RF-Field detection |
| | | | 0* | NFC-Level detector indication signal is used |
| | | | 1 | RF-Level detector indication signal is used |
| | | | 2 | NFC- and RF-Level detector indication signal is used |
| | | | 3 | Override - RF-Field detected is emulated |
| 7 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 6 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 5 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 4 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 3 | INTERNAL_USE[1] | R | 0 | For internal use |
| 2 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 1 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 0 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 227. CLIF_ANA_NFCLD_REG register (address 0100h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | R | 0 | Reserved |
| 8 | CM_PD_NFC_DET | R/W | 0 – 1* | Power Down NFC level detector |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5:0 | CM_RFL_NFC | R/W | 0h,4h* - Fh | Programming of detection level |

**Table 228. CLIF_ANA_TX_CLK_CONTROL_REG register (address 0104h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:11 | RESERVED | R | 0 | Reserved |
| 10:8 | TX_CLK_MODE_OVUN_PREV | R/W | 0*- 7 | Defines the TX clockmode for the period the overshoot/undershoot prevention is active |
| 7 | TX2_INV_RM | R/W | 0, 1* | if 1 -> TX output is inverted (clk_13m56_n is used); 0 -> clk_13m56 is used |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 6 | TX2_INV_CM | R/W | 0*, 1 | if 1 -> TX output is inverted (clk_13m56_n is used); 0 -> clk_13m56 is used |
| 5 | TX1_INV_RM | R/W | 0*, 1 | if 1 -> TX output is inverted (clk_13m56_n is used); 0 -> clk_13m56 is used |
| 4 | TX1_INV_CM | R/W | 0*, 1 | if 1 -> TX output is inverted (clk_13m56_n is used); 0 -> clk_13m56 is used |
| 3:1 | TX_CLK_MODE_RM | R/W | 0-7h,1* | TX clockmode |
| 0 | TX_PD_CLK_GEN | R/W | 0, 1* | TX clock generation disabled and thus also envelope synchronization to clk is disabled. Only valid in combination with the corresponding override bit. In normal operation, the pd is controlled by HW. |

**Table 229. CLIF_ANA_TX_AMPLITUDE_REG register (address 0108h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | TX_GSN_CW_RM | R/W | 0* - Fh | gsn settings @ continuous wave in reader mode |
| | | | | 0000: lowest power level |
| | | | | 1111: highest power level |
| 27:24 | INTERNAL_USE[1] | R/W | 0* - Fh | For internal use |
| 23:20 | INTERNAL_USE[1] | R/W | 0* - Fh | For internal use |
| 19:16 | INTERNAL_USE[1] | R/W | 0* - Fh | For internal use |
| 15 | RESERVED | R | 0 | Reserved |
| 14 | TX_ASKMODE_SELECT | R/W | 0* - 1 | Selects the output voltage mode if the transmitter. Default value is 1 |
| | | | 0 | Setting used for ALM mode: TX output voltage is independent of the envelope and hence TX output amplitude can be adjusted by the bitfield TX_RESIDUAL_CARRIER (0X1F is maximum, 0x00 minimum amplitude) |
| | | | 1 | Setting used for standard modes (reader and active mode) |
| 13:12 | TX_CW_AMPLITUDE_RM | R/W | 0* - 3h | sets amplitude of unmodulated carrier (continuous wave) @ reader mode |
| | | | 0 | Amplitude is set to TVDD - 150 mV |
| | | | 1 | Amplitude is set to TVDD - 250 mV |
| | | | 2 | Amplitude is set to TVDD - 500 mV |
| | | | 3 | Amplitude is set to TVDD - 1000 mV |
| 11:10 | RESERVED | R | 0 | Reserved |
| 9:8 | TX_CW_AMPLITUDE_ALM_CM | R/W | 0* - 3h | Set amplitude of unmodulated carrier @ card mode |
| 7:3 | TX_RESIDUAL_CARRIER | R/W | 0* - 1Fh | Set residual carrier (0=100 %, 1F = 0 %) |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 2 | TX_CW_TO_MAX_RM | R/W | 0* - 1h | TX HI output is the maximum voltage obtainable from charge pump (RM setting); if set to 1 -> TX_CW_AMPLITUDE_RM is overruled. |
| 1 | TX_CW_TO_MAX_ALM_CM | R/W | 0-1*h | TX HI output is the maximum voltage obtainable from charge pump (CM setting); if set to 1 -> TX_CW_AMPLITUDE_CM is overruled. |
| 0 | TX_PD | R/W | 0-1h* | charge pumps clamped to tvdd |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 230. CLIF_ANA_RX_REG register (address 0110h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:19 | INTERNAL_USE[1] | R | 0* | For internal use |
| 18 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 17 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 16 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 15 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 14 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 13 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 12 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 11 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 10 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 9:8 | INTERNAL_USE[1] | R/W | 0h*-3h | For internal use |
| 7 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 6 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 5 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 4 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 3:2 | RX_HPCF | R/W | 0h*-3h | Lower Corner Frequency: 00->45 kHz, 01->85 kHz, 10->150 kHz, 11->250 kHz |
| 1:0 | RX_GAIN | R/W | 0h*-3h | Gain Adjustment BBA: 00->33 dB, 01->40 dB, 10->50 dB, 11->57 dB |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 231. CLIF_ANA_CM_CONFIG_REG register (address 0114h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:17 | RESERVED | R | 0* | Reserved |
| 16 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 15 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 14 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 13:12 | CM_MILLER_SENS | R/W | 0h*-3h | Configuration bits for reference level of Miller demodulator (with respect to peak-peak input voltage) |
| | | | 0* | 30 % |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | 1 | 50 % |
| | | | 2 | 65 % |
| | | | 3 | 80 % |
| 11:10 | CM_MILLER_TAU | R/W | 0h – 3h | Configuration bits for the time constant of the reference generation in Miller demodulator |
| | | | 0* | 8 µs (cap of 125 f) |
| | | | 1 | 2 µs (cap of 500 f) |
| | | | 2 | 5 µs (cap of 200 f) |
| | | | 3 | Not allowed |
| 9 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 8 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 7 | INTERNAL_USE[1] | R/W | 0-1* | For internal use |
| 6 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 5 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 4 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 3 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 2 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 1:0 | INTERNAL_USE[1] | R/W | 0h*-3h | For internal use |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 232. CLIF_ANA_AGC_REG register (address 011Ch)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | R | 0* | Reserved |
| 3 | RESERVED | R/W | 0*, 1 | Reserved always set to 0 |
| 2 | AGC_PD | R/W | 0*-1 | AGC power down |
| 1:0 | AGC_VREF_SEL | R/W | 0h*-3h | Select the comparison reference voltage |
| | | | 0* | $V_{ref}$ = 1.15 V which results in $V_{Rx}$ = 0.5 $V_{pp}$ |
| | | | 1* | $V_{ref}$ f = 1.4 V which results in $V_{Rx}$ = 1 $V_{pp}$ |
| | | | 2* | $V_{ref}$ = 1.5 V which results in $V_{Rx}$ = 1.2 $V_{pp}$ |
| | | | 3* | $V_{ref}$ = 1.6 V which results in $V_{Rx}$ = 1.4 $V_{pp}$ |

**Table 233. CLIF_ANA_CLK_MAN_REG register (address 0120h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:7 | RESERVED | R | 0* | Reserved |
| 6 | INTERNAL_USE[1] | R/W | 0*, 1 | For internal use |
| 5 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 4 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 3 | INTERNAL_USE[1] | R/W | 0*-1 | For internal use |
| 2:0 | CLOCK_CONFIG_DLL_ALM | R/W | 0h*-7h | Select DLL clock phase in 45°C steps. |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| | | | | Note that the phase settings can be refined (on top of this configuration) in 5°C steps using CLIF_DPLL_INIT_REG. DPLL_CLOCK_CONFIG_ALM. |
| | | | 0* | 0 °C |
| | | | 1 | 45 °C |
| | | | 2 | 90 °C |
| | | | 3 | 135 °C |
| | | | 4 | 180 °C |
| | | | 5 | 225 °C |
| | | | 6 | 270 °C |
| | | | 7 | 315 °C |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 234. CLIF_ANA_TX_SHAPE_CONTROL_REG register (address 0128h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R | 0 | Reserved |
| 28:24 | TX_RESIDUAL_CARRIER_OV_ PREV | R/W | 0* - 1Fh | Defines the value for the residual carrier for the period the overshoot prevention pattern is active. |
| 23:18 | RESERVED | R | 0 | Reserved |
| 17 | TX _SET_BYPASS_SC_SHAPING | R/W | 0*, 1 | Bypasses switched capacitor shaping of the Transmitter Signal |
| 16:12 | RESERVED | R | 0 | Reserved |
| 11:8 | TX_SET_SLEW_SHUNTREG | R/W | 0h*-Fh | Set slew rate for shunt regulator |
| 7:4 | TX_SET_TAU_MOD_FALLING | R/W | 0h*-Fh | Transmitter TAU setting for falling edge of modulation shape. In AnalogControl module the output signal is switched with the tx_envelope. Only valid is TX_SET_BYPASS_SC_SHAPING is set |
| 3:0 | TX_SET_TAU_MOD_RISING | R/W | 0h*-Fh | Transmitter TAU setting for rising edge of modulation shape. In AnalogControl module the output signal is switched with the tx_envelope. Only valid is TX_SET_BYPASS_SC_SHAPING is set |

**Table 235. CLIF_ANA_TEST_REG register (address 01FCh)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | R | 0* | Reserved |
| 26 | RESERVED | R/W | 0*, 1 | Reserved |
| 25 | RESERVED | R/W | 0*, 1 | Reserved |
| 24 | RESERVED | R/W | 0*, 1 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 23 | RESERVED | R/W | 0*, 1 | Reserved |
| 22:19 | TX_SET_ILOAD | R/W | 0h-Fh, Ah* | settings for set_iload and corresponding assumed load current in the output path |
| 18:13 | RESERVED | R/W | 0h*-3Fh | Reserved |
| 12:7 | RESERVED | R/W | 0h*-3Fh | Reserved |
| 6 | RESERVED | R/W | 0*, 1 | Reserved |
| 5 | RESERVED | R | 0 | Reserved |
| 4:1 | RESERVED | R/W | 0h-Fh, 5h* | Reserved |
| 0 | TX_CW_AMP_REF2TVDD | R/W | 0*, 1 | If set to 1, the reference of the unmodulated carrier is defined relative to TVDD. Used for 5 V driver input |

**Table 236. CLIF_DPLL_INIT_REG register (address 0208h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R | 0 | Reserved |
| 27 | DPLL_CLOCK_CONFIG_ALM_SIGN | R/W | 0*, 1 | Defines the sign of the fine phase-changes |
| | | | 0* | apply positive phase offset |
| | | | 1 | apply negative phase offset |
| 26:24 | DPLL_CLOCK_CONFIG_ALM | R/W | 0* - 7 | Is the fine tuning of the phase configuration done on top of the coarse setting CLIF_ANA_CLK_MAN_REG.CLOCK_CONFIG_DLL_ALM |
| | | | 0 | 0 °C (mode disabled) |
| | | | 1 | 5 °C |
| | | | 2 | 10 °C |
| | | | 3 | 15 °C |
| | | | 4 | 20 °C |
| | | | 5 | 25 °C (not recommended) |
| | | | 6 | 30 °C (not recommended) |
| | | | 7 | 35 °C (not recommended) |
| 23:22 | RESERVED | R | 0 | Reserved |
| 21:20 | INTERNAL_USE[1] | R/W | 0*-3h | For internal use |
| | | | 0h* | Wakeup mode disabled |
| | | | 1h | ~37.8 us |
| | | | 2h | ~75.5 us |
| | | | 3h | ~113 us |
| 19 | RESERVED | R | 0 | Reserved |
| 18 | INTERNAL_USE[1] | R/W | 0*,1 | For internal use |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17:16 | INTERNAL_USE[1] | R/W | 0*-3h | For internal use |
| | | | 0* | Proportional: 2^(-3) and Integral: 2^(-20) |
| | | | 1 | Proportional: 2^(-4) and Integral: 2^(-21) |
| | | | 2 | Proportional: 2^(-5) and Integral: 2^(-22) |
| | | | 3 | Proportional: 2^(-6) and Integral: 2^(-23) |
| 15:14 | RESERVED | R | 0 | Reserved |
| 13:12 | INTERNAL_USE[1] | R/W | 0*-3h | For internal use |
| | | | 0* | ~37.8 us |
| | | | 1 | ~75.5 us |
| | | | 2 | ~151 us |
| | | | 3 | ~300 us |
| 11 | RESERVED | R | 0 | Reserved |
| 10:8 | INTERNAL_USE[1] | R/W | 0*-7h | For internal use |
| | | | 0* | 2^(-10) |
| | | | 7 | 2^(-17) |
| 7 | RESERVED | R | 0 | Reserved |
| 6:4 | INTERNAL_USE[1] | R/W | 0*-7h | For internal use |
| | | | 0* | 2^(-3) |
| | | | 7 | 2^(-10) |
| 3:2 | RESERVED | R | 0 | Reserved |
| 1:0 | RESERVED | R/W | 0*-3h | Reserved |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 237. CLIF_INT_CLR_ENABLE_REG register (address 3FD8h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | W | 0 | Reserved |
| 29 | AGC_RFOFF_DET_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 28 | TX_DATA_REQ_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 27 | RX_DATA_AV_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 26 | RX_BUFFER_OVERFLOW_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 25 | TX_WATERLEVEL_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 24 | RX_WATERLEVEL_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 23 | RESERVED | W | 0 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 22 | RX_SC_DET_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 21 | RX_SOF_DET_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 20 | RX_EMD_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 19 | TIMER3_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 18 | TIMER2_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 17 | TIMER1_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding |
| 16 | TIMER0_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE |
| 15 | CLOCK_ERROR_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE |
| 14 | INTERNAL_USE[1] | W | 0, 1 | For internal use |
| 13 | INTERNAL_USE[1] | W | 0, 1 | For internal use |
| 12 | RF_ACTIVE_ERROR_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 11 | TX_RFON_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 10 | TX_RFOFF_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 9 | RFON_DET_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 8 | RFOFF_DET_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 7:6 | RESERVED | W | 0 | Reserved |
| 5 | STATE_CHANGE_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 4 | CARD_ACTIVATED_IRQ_CLR_ ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 3 | MODE_DETECTED_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 2 | IDLE_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 1 | TX_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 0 | RX_IRQ_CLR_ENABLE | W | 0, 1 | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 238. CLIF_INT_SET_ENABLE_REG register (address 3FDCh)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | W | 0 | Reserved |
| 29 | AGC_RFOFF_DET_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 28 | TX_DATA_REQ_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 27 | RX_DATA_AV_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 26 | RX_BUFFER_OVERFLOW_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 25 | TX_WATERLEVEL_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 24 | RX_WATERLEVEL_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 23 | RESERVED | W | 0 | Reserved |
| 22 | RX_SC_DET_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 21 | RX_SOF_DET_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 20 | RX_EMD_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 19 | TIMER3_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 18 | TIMER2_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 17 | TIMER1_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding |
| 16 | TIMER0_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE |
| 15 | CLOCK_ERROR_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE |
| 14 | INTERNAL_USE[1] | W | 0, 1 | For internal use |
| 13 | INTERNAL_USE[1] | W | 0, 1 | For internal use |
| 12 | RF_ACTIVE_ERROR_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 11 | TX_RFON_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 10 | TX_RFOFF_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 9 | RFON_DET_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 8 | RFOFF_DET_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 7:6 | RESERVED | W | 0 | Reserved |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 5 | STATE_CHANGE_IRQ_SET_ ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 4 | CARD_ACTIVATED_IRQ_SE T_ ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 3 | MODE_DETECTED_IRQ_SE T_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 2 | IDLE_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 1 | TX_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 0 | RX_IRQ_SET_ENABLE | W | 0, 1 | Writing 1 to this register does set the corresponding IRQ ENABLE flag |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 239. CLIF_INT_STATUS_REG register (address 3FE0h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | R | 0 | Reserved |
| 29 | AGC_RFOFF_DET_ IRQ | R | 0, 1 | Set to 1 by hardware, when the AGC has detected the external RF-Field was switched off while transmitting in SL-ALM mode.<br>**Note**: Only valid if the detection mode is enabled with the register bit-field AGC_RF_DETECT_SEL. |
| 28 | TX_DATA_REQ_IRQ | R | 0*, 1 | Set to 1 by hardware, when the Buffer Manager requests data for transmission from RAM.<br>**Note**: Only valid if the bit DIRECT_DATA_ACCESS_ ENABLE is set to 1 |
| 27 | RX_DATA_AV_IRQ | R | 0*, 1 | Set to 1 by hardware, when the Buffer Manager holds received data from reception to be written to RAM.<br>**Note**: Only valid if the bit DIRECT_DATA_ACCESS_ ENABLE is set to 1 |
| 26 | RX_BUFFER_OVERFLOW_I RQ | R | 0*, 1 | Set to 1 by hardware, when the number of bytes received exceeds the size of the RX buffer.<br>**Note**: Reception is stopped in that case.<br>**Note**: If RX_MULTIPLE is set to 1 this IRQ is raised when the sum of all frames exceed the RX buffer size |
| 25 | TX_WATERLEVEL_IRQ | R | 0*, 1 | Set to 1 by hardware, when the number of bytes transmitted is equal to the TX_WATERLEVEL |
| 24 | RX_WATERLEVEL_IRQ | R | 0*, 1 | Set to 1 by hardware, when the number of bytes received is equal to the RX_WATERLEVEL |
| 23 | RESERVED | R | 0 | Reserved |
| 22 | RX_SC_DET_IRQ | R | 0*, 1 | Set to 1 by hardware, when in reader mode a subcarrier is detected |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 21 | RX_SOF_DET_IRQ | R | 0*, 1 | Set to 1 by hardware, when in reader mode an SOF is detected |
| 20 | RX_EMD_IRQ | R | 0*, 1 | Set to 1 by hardware, when an EMD event is detected |
| 19 | TIMER3_IRQ | R | 0*, 1 | Set to 1 by hardware, when the Timer3 is expired. |
| 18 | TIMER2_IRQ | R | 0*, 1 | Set to 1 by hardware, when the Timer2 is expired. |
| 17 | TIMER1_IRQ | R | 0*, 1 | Set to 1 by hardware, when the Timer1 is expired. |
| 16 | TIMER0_IRQ | R | 0*, 1 | Set to 1 by hardware, when the Timer0 is expired. |
| 15 | CLOCK_ERROR_IRQ | R | 0*, 1 | Set to 1 by hardware, when RF-Field vanished (and consequently the RF-Clock is not present) and the clock of the system PLL is not available |
| 14 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 13 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 12 | RF_ACTIVE_ERROR_IRQ | R | 0*, 1 | Set to 1 by hardware, when an RF error case occurred |
| 11 | TX_RFON_IRQ | R | 0*, 1 | Set to 1 by hardware, when the internally generated RF-field was switched on. |
| 10 | TX_RFOFF_IRQ | R | 0*, 1 | Set to 1 by hardware, when the internally generated RF-field was switched off. |
| 9 | RFON_DET_IRQ | R | 0*, 1 | Set to 1 by hardware, when an external RF-field is detected. |
| 8 | RFOFF_DET_IRQ | R | 0*, 1 | Set to 1 by hardware, when an external RF-field is switched off. |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5 | STATE_CHANGE_IRQ | R | 0*, 1 | Set to 1 by hardware, when a transceive state is entered selected in the register field STATE_TRIGGER_SELECT |
| 4 | CARD_ACTIVATED_IRQ | R | 0*, 1 | Set to 1 by hardware, when TypeA card mode activation FSM reached the ACTIVATED or ACTIVATE_S state |
| 3 | MODE_DETECTED_IRQ | R | 0*, 1 | Set to 1 by hardware, when the card mode has been detected by the ModeDetector<br>**Note**: While the TypeA activation FSM is active no IRQ is issued any more |
| 2 | IDLE_IRQ | R | 0*, 1 | Set to 1 by hardware, when the **IDLE** state is entered |
| 1 | TX_IRQ | R | 0*, 1 | Set to 1 by hardware, when an ongoing transmission is finished. |
| 0 | RX_IRQ | R | 0*, 1 | Set to 1 by hardware, when an ongoing reception is finished |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 240. CLIF_INT_ENABLE_REG register (address 3FE4h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | R | 0 | Reserved |
| 29 | AGC_RFOFF_DET_IRQ_ENABLE | R | 0, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 28 | TX_DATA_REQ_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 27 | RX_DATA_AV_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 26 | RX_BUFFER_OVERFLOW_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 25 | TX_WATERLEVEL_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 24 | RX_WATERLEVEL_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 23 | RESERVED | R | 0 | Reserved |
| 22 | RX_SC_DET_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 21 | RX_SOF_DET_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 20 | RX_EMD_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 19 | TIMER3_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 18 | TIMER2_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 17 | TIMER1_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 16 | TIMER0_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 15 | CLOCK_ERROR_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 14 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 13 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 12 | RF_ACTIVE_ERROR_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 11 | TX_RFON_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 10 | TX_RFOFF_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 9 | RFON_DET_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 8 | RFOFF_DET_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5 | STATE_CHANGE_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 4 | CARD_ACTIVATED_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 3 | MODE_DETECTED_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 2 | IDLE_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 1 | TX_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 0 | RX_IRQ_ENABLE | R | 0*, 1 | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 241. CLIF_INT_CLR_STATUS_REG register (address 3FE8h)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | R | 0 | Reserved |
| 29 | AGC_RFOFF_DET_IRQ_CLR_STATUS | R | 0, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 28 | TX_DATA_REQ_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 27 | RX_DATA_AV_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 26 | RX_BUFFER_OVERFLOW_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 25 | TX_WATERLEVEL_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 24 | RX_WATERLEVEL_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 23 | RESERVED | R | 0 | Reserved |
| 22 | RX_SC_DET_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 21 | RX_SOF_DET_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 20 | RX_EMD_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 19 | TIMER3_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 18 | TIMER2_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 17 | TIMER1_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 16 | TIMER0_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 15 | CLOCK_ERROR_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 14 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 13 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 12 | RF_ACTIVE_ERROR_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 11 | TX_RFON_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 10 | TX_RFOFF_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 9 | RFON_DET_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 8 | RFOFF_DET_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5 | STATE_CHANGE_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 4 | CARD_ACTIVATED_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 3 | MODE_DETECTED_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 2 | IDLE_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 1 | TX_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 0 | RX_IRQ_CLR_STATUS | R | 0*, 1 | Writing 1 to this register does clear the corresponding IRQ STATUS flag |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

**Table 242. CLIF_INT_SET_STATUS_REG register (address 3FECh)**

** = reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | R | 0 | Reserved |
| 29 | AGC_RFOFF_DET_IRQ_SET_STATUS | R | 0, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 28 | TX_DATA_REQ_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 27 | RX_DATA_AV_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 26 | RX_BUFFER_OVERFLOW_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 25 | TX_WATERLEVEL_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 24 | RX_WATERLEVEL_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 23 | RESERVED | R | 0 | reserved |
| 22 | RX_SC_DET_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 21 | RX_SOF_DET_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 20 | RX_EMD_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 19 | TIMER3_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 18 | TIMER2_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 17 | TIMER1_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 16 | TIMER0_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 15 | CLOCK_ERROR_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 14 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 13 | INTERNAL_USE[1] | R | 0*, 1 | For internal use |
| 12 | RF_ACTIVE_ERROR_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 11 | TX_RFON_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 10 | TX_RFOFF_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 9 | RFON_DET_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 8 | RFOFF_DET_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 7:6 | RESERVED | R | 0 | Reserved |
| 5 | STATE_CHANGE_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 4 | CARD_ACTIVATED_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 3 | MODE_DETECTED_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 2 | IDLE_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 1 | TX_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 0 | RX_IRQ_SET_STATUS | R | 0*, 1 | Writing 1 to this register does set the corresponding IRQ STATUS flag |

[1] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

## 12.7 Dynamic Power Control (DPC)

### 12.7.1 DPC description

The Dynamic Power Control allows to adjust the Transmitter output current dependent on the loading condition of the antenna.

A lookup table (LUT) is used to configure the output voltage and by this control the transmitter current. In addition to the control of the transmitter current, wave shaping settings can be controlled as well dependent on the selected protocol and the measured antenna load.

The PN7462 allows to measure periodically the RX voltage. The RX voltage is used as indicator for the actual antenna current. The voltage measurement is done with the help of the AGC. The time interval between two measurements can be configured with the DPC_TIME byte in the EEPROM.

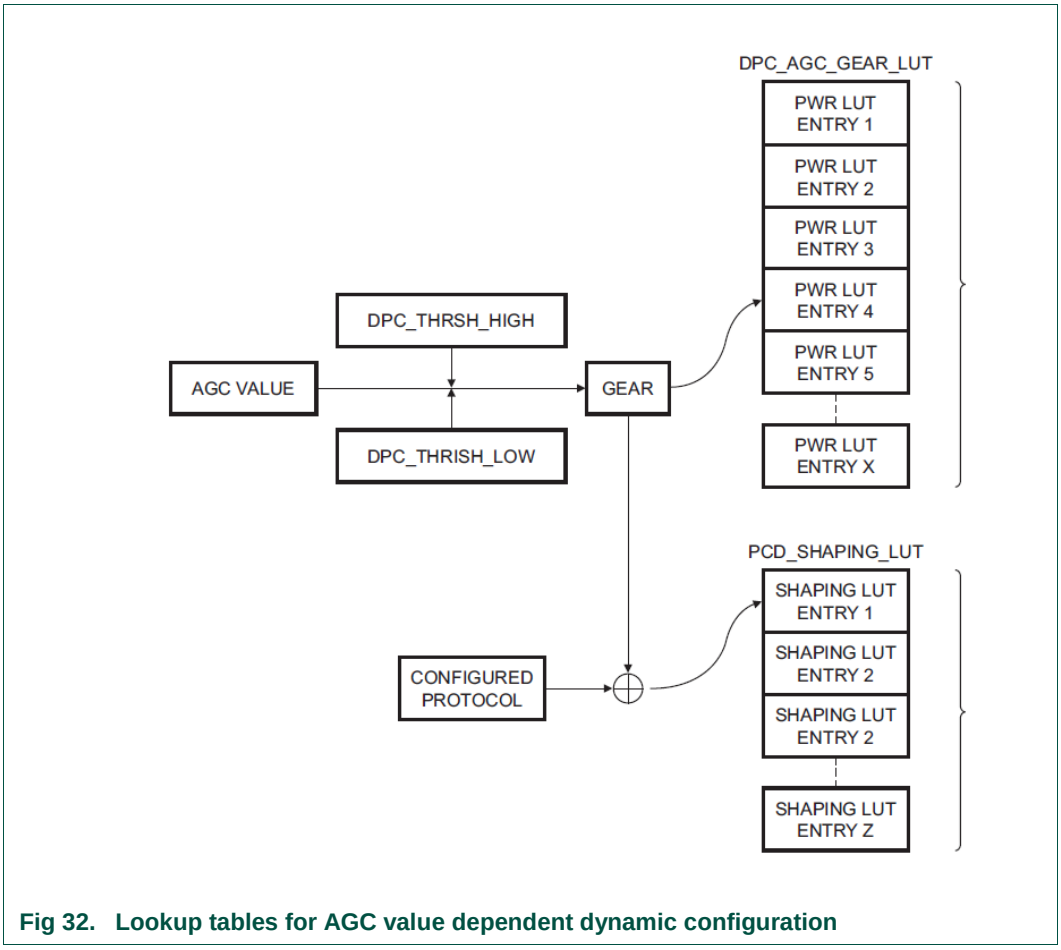



**Fig 32. Lookup tables for AGC value dependent dynamic configuration**

The AGC value is compared to a maximum and minimum threshold value which is stored in EEPROM.
If the AGC value is exceeding one of the thresholds, a new gear configuring another transmitter supply driver voltage will be activated. The number of gears - and by these

transmitter supply voltage configurations - can be defined by the application, up to 15 gears are available.

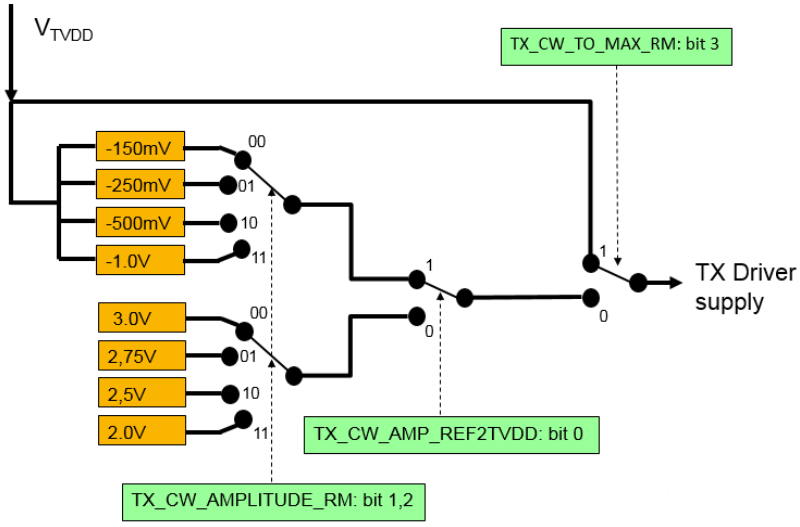



**Fig 33. Transmitter supply voltage configuration, $V_{DD(TVDD)}$ > 3.5 V**

### 12.7.2 DPC EEPROM values

Following table (Table 243) shows the EEPROM address, variable name as well as the description and access rights of all DPC values.

**Table 243. DPC values**

*DPC related EEPROM values*

| Value | EEPROM | Variable | ACCESS | Size (bytes) | Comment |
|---|---|---|---|---|---|
| DPC_CONTROL | 0x201352 | bOcProtControl | R/W | 1 | bit7..4 START_GEAR; binary definition of start gear,<br>bit4=LSB of start gear number<br>bit3..1 GEAR_STEP_SIZE: binary definition of gear<br>step size, bit1=LSB of gear step size<br>bit0 DPC_ENABLE cleared: OFF; set: ENABLE |

| | | | | | |
|---|---|---|---|---|---|
| DPC_TIME | 0x201328 | wControlCycle | R/W | 2 | Sets the value for the periodic regulation. Time base is<br>1/20 MHz. (Example: Value of 20000 is equal to 1 ms) |
| DPC_XI | 0x201353 | bAgcXi | R/W | 1 | Trim Value of the AGC value |
| AGC_CONTROL | 0x20132A | wAgcFastModeConfig | R/W | 2 | Controls the AGC loop<br>bit15..14 RFU<br>bit13 StepSize Enable<br>bit12..11 StepSize<br>bit10... Duration Enable<br>bit9..0 Duration |
| DPC_THRSH_HIGH | 0x201334 | wAgcTrshHigh | R/W | 30 | Defines the AGC high threshold for each gear.<br>Number of gears can be 1..15 |
| DPC_THRSH_LOW | 0x20132C | wAgcTrshLow | R/W | 2 | Defines the AGC low threshold for initial gear. |
| DPC_DEBUG | 0x201354 | bDebug | R/W | 1 | Enables the debug signals |
| DPC_AGC_SHIFT_VALUE | 0x201355 | bAgcShiftValue | R/W | 1 | Shift Value for the AGC dynamic low adoption to<br>prevent oscillation |
| DPC_AGC_GEAR_LUT_SIZE | 0x201356 | bSizeOfLUT | R/W | 1 | Defines the number of gears for the lookup table (LUT,<br>value can be between 1...15) |
| DPC_AGC_GEAR_LUT | 0x201357 | bConfigLUT | R/W | 15 | Defines the Gear Setting for each step size starting<br>with Gear0 up to 15 gears. Each entry contains a<br>definition for the DPC_CONFIG register content. Bits |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | 8:11 are not taken into account. |
| DPC_GUARD_FAST_MODE | 0x20132E | wGuardTimeFastMode | R/W | 2 | Guard time after AGC fast mode has been triggered.<br>This happens in the following scenarios:<br>- End of Receive<br>- End of Transmit<br>- After a gear switch<br>Time base is 1/20 MHz (Example: Value of 2000 is<br>equal to 100us) |
| DPC_GUARD_SOF_DETECTED | 0x201330 | wGuardTimeSofDetected | R/W | 2 | Guard time after SoF or SC detection. This is to avoid any DPC regulation between SoF/SC and actual begin of reception. Time base is 1/20MHz (Example: Value of<br>2000 is equal to 100us) |
| DPC_GUARD_FIELD_ON | 0x201332 | wGuardTimeFieldOn | R/W | 2 | Guard time after Gear Switch during FieldOn instruction. Time base is 1/20MHz (Example: Value of<br>2000 is equal to 100us) |
| PCD_SHAPING_LUT_SIZE | 0x2013A8 | bSizeOfLUT | R/W | 1 | Number of elements for the PCD Shaping |
| PCD_SHAPING_LUT | 0x201368-0x2013a7 | dwConfiguration | R/W | 64 | PCD Shaping configuration lookup table: Each word contains the following information:<br>0..3: DPC Gear<br>4..7: TAU_MOD_FALLING (Sign bit + 3-bit value) |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | 8..11: TAU_MOD_RISING (Sign bit + 3-bit value)<br>12..15: RESIDUAL_CARRIER (Sign bit + 3-bit value)<br>16..31: Bitmask identifying technology and baud rate |

## 12.8 Adaptive Waveform Control (AWC)

Depending on the level of detected detuning of the antenna, RF wave shaping related register settings can be automatically updated. The shaping related register settings are stored in a lookup table located in EEPROM, and selected dependent on the actual gear. The gear numbers need to be provided as part of the lookup table entries and need to be provided in ascending order in the EEPROM. Each lookup table entry allows to configure not only a dedicated wave shaping configuration for the corresponding gear, but in additionally it is possible to configure for this gear the wave shaping configuration dependent on the different protocols.
Each lookup table item contains a bitmask of technology and baud rate (in order to use an entry for multiple technologies and baud rates), the DPC Gear and a relative value (change compared to actual setting of register CLIF_ANA_TX_SHAPE_CONTROL_REG) for TX_SET_TAU_MOD_FALLING, TX_SET_TAU_MOD_RISING and TX_RESIDUAL_CARRIER.

**Table 244. PCD Shaping entry definition**
*Each entry contains 32 bit.*

| bit # | Definition | |
|---|---|---|
| 31 … 29 | RFU, don't use. | Must be 0. |
| 28 | 180003m3 | Enabling a protocol enables the defined PCD shaping setting changes, as defined in bit 15 … 4 for the defined gear as defined in bit 3 … 0.<br>One or more protocols can be enabled. |
| 27 | 15693 @ASK100% | |
| 26 | 15693 @ASK10% | |
| 25 | NFC type F @424 | |
| 24 | NFC type F @212 | |
| 23 | 14443 type B @848 | |
| 22 | 14443 type B @424 | |
| 21 | 14443 type B @212 | |
| 20 | 14443 type B @106 | |
| 19 | 14443 type A @848 | |
| 18 | 14443 type A @424 | |
| 17 | 14443 type A @212 | |
| 16 | 14443 type A @106 | |
| 15 …12 | Residual carrier | Sign bit + 3 bit value |
| 11 … 8 | TAU_MOD_RISING | Sign bit + 3 bit value |
| 7 … 4 | TAU_MOD_FALLING | Sign bit + 3 bit value |
| 3 … 0 | Gear number | Gear number 1 … 14 |

In case of a gear switch, a EEPROM lookup is performed if the current gear (at current protocol and baud rate) has an assigned wave shaping configuration. In case of an execution of a LoadProtocol command, this lookup will be performed (example: switching from baud rate A106 to A424) as well. The change from the wave shaping configuration as configured by `phhalRf_LoadProtocol` is relative, which means that bits are added or

subtracted from the existing configuration. For an increasing gear value, the defined change is cumulative.

The PCD shaping LUT (LookUp Table) value is stored beginning at EEPROM address 0x201368. Each value is stored LSB first.

# 13. PN7462AU Contact interface

The PN7462AU integrates contact interface to enable communication with ISO7816 and EMVCo contact smart cards, without the need for an external contact front end. It offers a high level of security for the cards by performing current limitation, short-circuit detection, ESD protection as well as supply supervision.

PN7462AU also offer the possibility to extend the number of contact interfaces available, using an I/O Aux interface to connect a slot extension (TDA8007 - 2 slots, TDA8020 - 2 slots, TDA8026 - 5 slots).

## 13.1 Contact interface features

- Support of Class A (5), Class B (3 V) and Class C (1.8 V) contact smart cards
- Compliant with ISO7816 and EMV Co 4.3 standards
- Protection of the smart card
  - Thermal and current limitation in the event of short-circuit (pins I/O, VCC)
  - Vcc regulation: 5 V, 3 V, and 1.8 V
  - Automatic activation and deactivation sequences initiated by software, or by hardware in the event of a short-circuit, card take-off, overheating, falling of PN7462AU supply
  - Enhanced card-side Electro Static Discharge (ESD) protection of (>12 kV)
  - Supply supervisor for killing spikes during power on and off
- DC-to-DC converter for VCC generation to enable support of Class A and Class B cards with low input voltages
- Built-In debouncing on card presence contact
- Card Clock generation up to 13.56 MHz using the external crystal oscillator (27.12MHz) with synchronous frequency changes of fXTAL/2, fXTAL/3, fXTAL/4, fXTAL/5, fXTAL/6, fXTAL/8, fXTAL/16
- Specific ISO UART with APB access for automatic convention processing, variable baud rate through frequency or division ratio programming, error management at character level for T=0 and extra guard time register
- FIFO 1 to 32 characters in both reception and transmission mode
- Parity error counter in reception mode and transmission mode with automatic re-transmission
- Cards clock stop (at HIGH or LOW level)
- Automatic activation and deactivation sequence through a sequencer
- Supports the asynchronous protocols T=0 and T=1 in accordance with ISO7816 and EMV

- Versatile 24-bit time-out counter for Answer To Reset (ATR) and waiting times processing
- Specific Elementary Time Unit (ETU) counter for Block Guard Time (BGT): 22 in T=1 and 16 in T=0
- Supports synchronous cards

## 13.2 Typical application schematics

The following figure shows the basic application schematics to use the embedded contact smart card controller.

For more details on how to connect a smart card interface, refer to the dedicated PN7462AU contact interface application note.

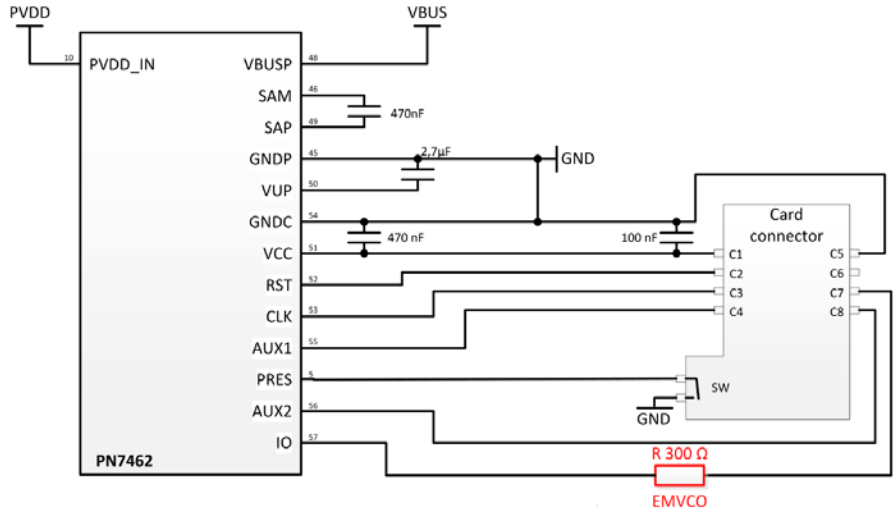



**Fig 34. Contact Application schematics**

## 13.3 Pin description and power supply

In order to control one or several smart card, the PN7462AU has to be supplied with respect to the smart card needs. Then some pins are required to generate the right smart card power supply, and control the smart card activation and data exchange.

There are two sets of pins: one for the main smart card, controlled by the embedded smart card controller, one for the extra smart cards, electrically handled by an external TDA:

**Table 245. Contact Smart card pins – Main slot**

*Pins description for the main contact smart card slot*

| Pin name | Pin n° | Description | Specific requirement |
|---|---|---|---|
| VBUSP | 48 | PN7462AU for the contact smart card interface | Must be > 3 V for Class A cards |
| GNDP | 45 | Power supply main ground pin | Must be connected with a short and low resistive connection to the PCB main ground plane |
| SAM/SAP | 46/49 | DC-to-DC converter capacitor pins | Connected together through a 470 nF capacitor. Needed to generate a 5V power supply out of a > 3 V VBUSP |
| VUP | 50 | DC-to-DC converter power output pin | Must be connected to a capacitor, connected to GNDP. Connection to GNDP must be as short as possible |
| VCC | 51 | Smart card power supply | Must be decoupled through 2 capacitors |
| GNDC | 54 | Ground pin for the smart card | Must be connected to the smart card connector, and to the main PCB ground plane |
| RST | 52 | Reset pin for the smart card | |
| CLK | 53 | Clock output to the smart card | |
| IO | 57 | Data pin to/from smart card | |
| AUX1 | 55 | C4 smart card pin | |
| AUX2 | 56 | C8 smart card pin | |
| PRESN | 5 | Presence detection for the smart card. | Referenced to PVDD_IN |

**Table 246. Contact Smart card pins – Auxiliary slots**

*Pins description for the auxiliary contact smart card slots*

| Pin name | Pin n° | Description | Specific requirement |
|---|---|---|---|
| CLK_AUX | 2 | Output clock to the external TDA device | |
| IO_AUX | 3 | Data pin to/from smart card | To be connected to the external TDA IOUC pin |
| INT_AUX | 4 | Interrupt input from the external TDA | To be connected to the interrupt output from the TDA (INT or OFFN, depending on the TDA type) |
| I2C MASTER_SDA | 1 | SDA Master line to control the TDA | Needed for a TDA8023 or TDA8026 |
| I2C MASTER_SCL | 64 | SCL Master line to control the TDA | Needed for a TDA8023 or TDA8026 |
| GPIOx | X | GPIOs to control the TDA | Needed for TDA with GPIO interface as TDA8035/8034/8037 |

For more details about the pins and power supply, refer to the PN7462AU datasheet. For deeper description of usage of each pin, please refer to the PN7462AU contact interface application note.

## 13.4 Specific blocks

### 13.4.1 ATR counter

The sequencer manages the activation and deactivation sequences. In addition to the sequencer, the ATR counter is used to manage RST and check the asynchronous card ATR on the full slot. In case of synchronous cards, RST is controlled via RSTIN bit (see the registers description) and the card ATR is not checked. The operating mode (asynchronous or synchronous) has to be selected by the application (see the registers description).

The ATR counter block is composed of two counters. One checks the early answer and the second checks if the card is mute.

The early answer counter is composed of a fixed part that counts up to 200d CLK cycles.

An additional part counts up to EC7-EC0 bits value CLK cycles (see the registers description). The default value of EC7-EC0 bits is 170d, which gives a total default count of 370d CLK cycles. The additional configurable count enables to follow a potential standard change.

The mute counter counts up to MCL15-MCL0 bits value CLK cycles when RST is LOW and up to MCH15-MCH0 bits value CLK cycles when RST is HIGH (see the registers description). The default value of MCL15-MCL0 & MCH15-MCH0 bits is 42100d, which gives a default count of 42100d CLK cycles. The value chosen for MCL15-MCL0 bits can be different from the one of MCH15-MCH0 bits. This configurable count enables to support ISO7816 and EMV compliant cards and to follow a potential standard change.

Let’s have a look to an asynchronous card activation and ATR. First, the application starts the activation (START bit) after having configured the slot 1 (activation voltage). The sequencer performs the activation sequence. The DC-to-DC converter is started, then $V_{CC}$ goes to logic level one, I/O is enabled and CLK starts. RST is at logic level zero.

Then the ATR counter checks the following steps:

1. If a start bit is detected on I/O during the first 200d CLK cycles, it is ignored and the count goes on.
2. If a start bit is detected whilst RST is at logic level zero between 200d and 42100d (or the value written in MCL15-MCL0 bits) CLK cycles, the bits EARLY and MUTE are set to logic level one. RST will remain at logic level zero, it is up to the application to decide whether accepting the card or not.
3. If no start bit has been detected until 42100d (or the value written in MCL15-MCL0 bits) CLK cycles, RST is set to logic level one.
4. If a start bit is detected within the first 370d (or 200d + the value written in EC7-EC0 bits) CLK cycles with RST at logic level one, the bit EARLY is set to logic level one.
5. If the card does not answer before 42100d (or the value written in MCH15-MCH0 bits)

6. CLK cycles with RST at logic level one, the bit MUTE is set to logic level one.
7. If the card answers within the correct time window, the CLK cycles count is stopped and the application may send commands to the card.

The picture below shows the timings checked by the ATR counter:

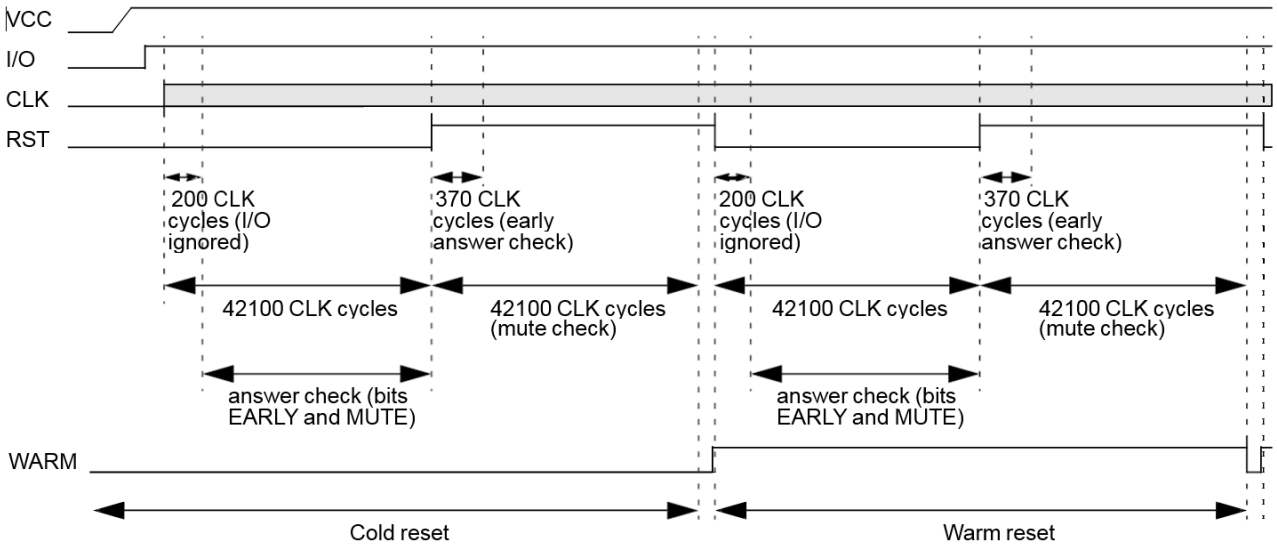



**Fig 35. ATR counter timings checked (with default values)**

The bits EARLY and MUTE signal an interrupt when set to logic level one (see the registers description).

The sequence mentioned above relates to a cold reset (left part of Fig 35). If the card is mute (has not answered), the application may start a warm reset by setting WARM bit to logic level one (see the registers description). Then, the ATR counter set RST to logic level zero and performs the same timing checks (right part of Fig 35).

### 13.4.2 FIFO

The FIFO is used for both Reception and Transmission modes. This block receives characters from the card via the Reception block and provides characters written by the micro-controller to the Transmission block. Its depth is 32 bytes.

- Reception:
  - In reception mode, when a received character is correct (no parity error) at 10.5 ETUs (T=0), it is loaded into the FIFO which size pointer is incremented.
  - If the FIFO size pointer equals 32, no more character can be loaded into the FIFO. An Overrun interrupt will be generated by the ct_usr1_reg register to mean that at least one character will be lost.

- When the micro-controller reads a character, the FIFO size pointer is decremented.
- A read operation when the FIFO is empty will not cause any action for the FIFO (size pointer unchanged). In this case, 0 is read.

- Transmission:
  - The micro-controller can write a character into the FIFO whilst the UART is in transmission mode and if the FIFO is not full. If the FIFO contains already 32 characters, the write operation is not taken into account.
  - The FIFO commands the transmission. If its size pointer is one or more, the FIFO starts the transmission by loading the first character to transmit in the Transmission block. Then, the Transmission block will manage the transmission to the card. The FIFO size pointer is decremented at 9.5 ETUs.
  - If a parity error interrupt occurs after one or more retransmission(s) (T=0), there are two cases:

PEC=0: no action, the transmission doesn't stop.

PEC>0 (automatic mode): the transmission stops. The software will deactivate the card: the parity error counter will be reset by hardware when activating. If necessary, the firmware has the possibility to pursue the transmission. By reading the number of bytes present into the FIFO (ffl bits), it can determine which character has been naked PEC +1 times by the card. It will then flush the FIFO (FIFO flush bit). The next step consists in unlocking the transmission using dispe bit. By writing this bit at logic level one (and then at logic level zero if the firmware still wants to check parity errors), the transmission is unlocked. The firmware can now write bytes into the FIFO.

- Turnaround Reception -> Transmission:
  - There is a hardware protection when switching from reception to transmission mode. If the micro-controller sets to logic 1 the bit T/R for example between 10.5 (ft occurs) and 11.8 ETUs (reception finished), only the FIFO switches in transmission mode and not the rest of the UART which remains in reception mode. This allows the micro-controller to write characters into the FIFO. At 11.8 ETUs, the whole UART switches in transmission mode.
  - The FIFO is in transmission mode when the bit T/R is set to logic 1, else in reception mode. The transmission starts when the whole UART is in transmission mode, that is to say when the internal bit T/R in register ct_ucr1_reg is set to logic 1.

## 13.5 Connect an external TDA

The PN7462AU can handle more than one smart card by controlling an extra contact interface front-end (typically TDA product from NXP).

In this use case, the PN7462AU is the main controller for the electrical and protocol part for the main card slot, while the secondary slots are electrically controlled by an extra contact front-end interface (TDA), the PN7462AU being the protocol controller for these extra slots.

When such a design is used, several smart cards can be activated at the same time, but the communication with each smart card has to go sequentially: it is not possible to

communicate with two smart cards at the same time as there is only one protocol control block for all cards.

The figure below shows an overview of the connections when PN7462AU is used to control several smart cards.

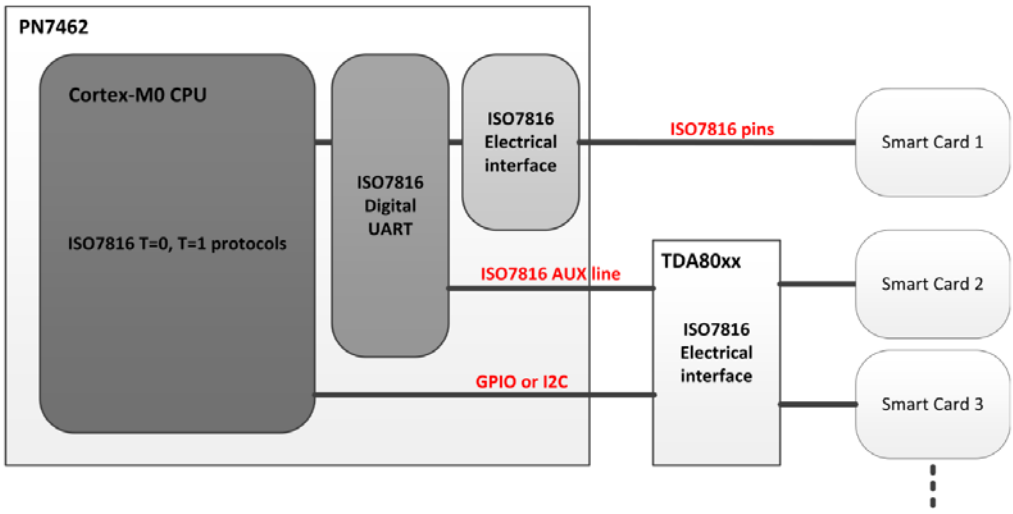
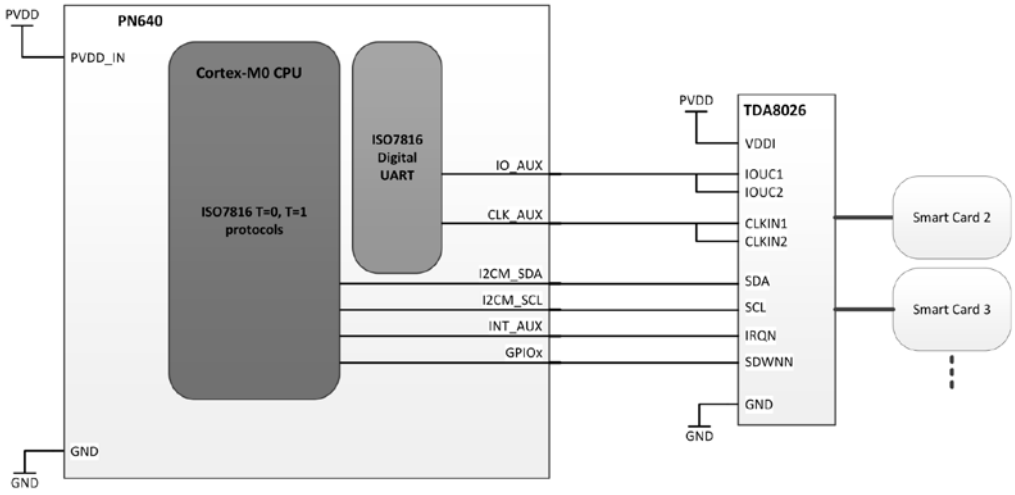



**Fig 36. Contact interface architecture**

The external TDA is required to handle the smart card electrical interface. The connection between the PN7462AU and the TDA is composed of 2 parts:

- The host interface control, where the PN7462AU is the master, controlling the TDA behavior: card activation, deactivation, TDA configuration (voltage level, clock division, slew rates…)
- The ISO7816 link: the PN7462AU handles the ISO7816 communication protocol and uses the TDA as a level shifter for the clock and I/O signals.

The figure below summarizes this connection.




(1) This figure only shows the link between PN7462AU and TDA8026, but is not an application schematics. External components are not showed here.

**Fig 37. External TDA8026 connection**

The default PN7462AU FW embeds a SW controlling a TDA8026.

But it is also possible to connect other external contact interface, such as TDA8035. The difference is the number of slots (TDA8035 controls only 1 slot), and the host interface: on TDA8035, the host interface doesn't use I2C, but it uses a set of GPIOs. It is possible to control this TDA8035 by changing the connection between the PN7462AU and the TDA8035, and by updating the FW to control the TDA8035 through GPIOs, instead of the TDA8026 through I2C.

Here the SW modifications only apply to the TDA control part: configuration, and card activation/deactivation. The whole communication protocol remains unchanged, through IO_AUX and CLK_AUX.

## 13.6 Registers

### 13.6.1 Register overview

**Table 247. Contact interface – Register overview (base address 0x4001 4000)**

| Name | Address offset | Width (bits) | Access | Reset Value | Description |
|---|---|---|---|---|---|
| ct_ssr_reg | 0000h | 5 | R/W | 00000001h | Slot Select Register |
| ct_pdr1_lsb_reg | 0004h | 8 | R/W | 00000074h | Programmable Divider Register (LSB) slot 1 |
| ct_pdr2_lsb_reg | 0004h | 8 | R/W | 00000074h | Programmable Divider Register (LSB) slot 2 |
| ct_pdr1_msb_reg | 0008h | 8 | R/W | 00000001h | Programmable Divider Register (MSB) slot 1 |
| ct_pdr2_msb_reg | 0008h | 8 | R/W | 00000001h | Programmable Divider Register (MSB) slot 2 |
| ct_fcr_reg | 000Ch | 8 | R/W | 00000001h | FIFO Control Register |

| Name | Address offset | Width (bits) | Access | Reset Value | Description |
|---|---|---|---|---|---|
| ct_gtr1_reg | 0010h | 8 | R/W | 00000000h | Guard Time Register slot 1 |
| ct_gtr2_reg | 0010h | 8 | R/W | 00000000h | Guard Time Register slot 2 |
| ct_ucr11_reg | 0014h | 6 | R/W | 00000000h | UART Configuration Register 1 slot 1 |
| ct_ucr12_reg | 0014h | 6 | R/W | 00000000h | UART Configuration Register 1 slot 2 |
| ct_ucr21_reg | 0018h | 8 | R/W | 00000000h | UART Configuration Register 2 slot 1 |
| ct_ucr22_reg | 0018h | 7 | R/W | 00000000h | UART Configuration Register 2 slot 2 |
| ct_ccr1_reg | 001Ch | 6 | R/W | 00000000h | Clock Configuration Register slot 1 |
| ct_ccr2_reg | 001Ch | 5 | R/W | 00000000h | Clock Configuration Register slot 2 |
| ct_pcr_reg | 0020h | 7 | R/W | 000000C0h | Power Control Register |
| ct_ecr_reg | 0024h | 8 | R/W | 000000AAh | Early answer CounteR register |
| ct_mcrl_lsb_reg | 0028h | 8 | R/W | 00000074h | Mute card CounteR RST Low register (LSB) |
| ct_mcrl_msb_reg | 002Ch | 8 | R/W | 000000A4h | Mute card CounteR RST Low register (MSB) |
| ct_mcrh_lsb_reg | 0030h | 8 | R/W | 00000074h | Mute card CounteR RST High register (LSB) |
| ct_mcrh_msb_reg | 0034h | 8 | R/W | 000000A4h | Mute card CounteR RST High register (MSB) |
| ct_srr_reg | 0038h | 6 | R/W | 00000000h | Slew Rate configuration Register |
| ct_urr_reg | 003Ch, 0040h 0044h, 0048h | 32 | R | 00000000h | UART Receive Register |
| ct_utr_reg | 003Ch, 0040h 0044h, 0048h | 32 | W | 00000000h | UART Transmit Register |
| ct_tor1_reg | 004Ch | 8 | W | 00000000h | Time-Out Register 1 |
| ct_tor2_reg | 0050h | 8 | W | 00000000h | Time-Out Register 2 |
| ct_tor3_reg | 0054h | 8 | W | 00000000h | Time-Out Register 3 |
| ct_toc_reg | 0058h | 8 | R/W | 00000000h | Time-Out Configuration register |
| ct_fsr_reg | 005Ch | 6 | R | 00000000h | FIFO Status register |
| ct_msr_reg | 0060h | 3 | R | 00000000h | Mixed Status register |
| ct_usr1_reg | 0064h | 6 | R | 00000000h | UART Status register 1 |
| ct_usr2_reg | 0068h | 8 | R | 00000000h | UART Status register 2 |
| INTERNAL_USE | 006Ch | 4 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0070h | 8 | R | 00000000h | For internal use |
| INTERNAL_USE | 0074h | 4 | R/W | 00000000h | For internal use |

**Table 248. Register bit overview**

| Name | Access | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset value |
|---|---|---|---|---|---|---|---|---|---|---|
| ssr | R/W | RESERVED | RESERVED | RESERVED | pres con no | pres pup en | CLKAUX en | IOAUXen | softreset | XXXX 0001 |
| pdr LSB[2] | R/W | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 | 0111 0100 |
| pdr MSB[2] | R/W | PD15 | PD14 | PD13 | PD12 | PD11 | PD10 | PD9 | PD8 | 0000 0001 |
| Fcr | R/W | PEC2 | PEC1 | PEC0 | ftc4 | ftc3 | ftc2 | ftc1 | ftc0 | 0000 0001 |
| gtr[2] | R/W | GT7 | GT6 | GT5 | GT4 | GT3 | GT2 | GT1 | GT0 | 0000 0000 |
| ucr1[2] | R/W | RESERVED | RESERVED | FIP | FC | PROT | T/R | LCT | CONV | XX00 0000 |
| ucr2[2] | R/W | wrdacc | FIFO flush | disintaux | disATR counter[4] | dispe | disft | MAN BGT | AUTO CONV | 0000 0000 |
| ccr[2] | R/W | RESERVED | RESERVED | SHL | CST[5] | SAN | ACC2 | ACC1 | ACC0 | XX00 0000 |
| pcr | R/W | C8 | C4 | RESERVED | RSTIN | vccsel1 | vccsel0 | WARM | START | 11X0 0000 |
| ecr | R/W | EC7 | EC6 | EC5 | EC4 | EC3 | EC2 | EC1 | EC0 | 1010 1010 |
| mcrl LSB | R/W | MCL7 | MCL6 | MCL5 | MCL4 | MCL3 | MCL2 | MCL1 | MCL0 | 0111 0100 |
| mcrl MSB | R/W | MCL15 | MCL14 | MCL13 | MCL12 | MCL11 | MCL10 | MCL9 | MCL8 | 1010 0100 |
| mcrh LSB | R/W | MCH7 | MCH6 | MCH5 | MCH4 | MCH3 | MCH2 | MCH1 | MCH0 | 0111 0100 |
| mcrh MSB | R/W | MCH15 | MCH14 | MCH13 | MCH12 | MCH11 | MCH10 | MCH9 | MCH8 | 1010 0100 |
| srr | R/W | RESERVED | RESERVED | vcc rise sel1 | vcc rise sel0 | clk_sr1 | clk_sr0 | io_sr1 | io_sr0 | XX00 0000 |
| urr[3] | R | UR7 | UR6 | UR5 | UR4 | UR3 | UR2 | UR1 | UR0 | 0000 0000 |
| utr[3] | W | UT7 | UT6 | UT5 | UT4 | UT3 | UT2 | UT1 | Ut0 | 0000 0000 |
| tor1 | W | TOL7 | TOL6 | TOL5 | TOL4 | TOL3 | TOL2 | TOL1 | TOL0 | 0000 0000 |
| tor2 | W | TOL15 | TOL14 | TOL13 | TOL12 | TOL11 | TOL10 | TOL9 | TOL8 | 0000 0000 |
| tor3 | W | TOL23 | TOL22 | TOL21 | TOL20 | TOL19 | TOL18 | TOL17 | TOL16 | 0000 0000 |
| toc | R/W | TOC7 | TOC6 | TOC5 | TOC4 | TOC3 | TOC2 | TOC1 | TOC0 | 0000 0000 |
| fsr | R | RESERVED | RESERVED | ffl5 | ffl4 | ffl3 | ffl2 | ffl1 | ffl0 | XX00 0000 |

| Name | Access | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset value |
|---|---|---|---|---|---|---|---|---|---|---|
| msr | R | RESERVED | RESERVED | RESERVED | RESERVED | RESERVED | INTAUX | BGT | PRES | XXXXX000 |
| usr1 | R | RESERVED | RESERVED | MUTE | EARLY | pe | ovr | fer | ft | XX000000 |
| usr2 | R | to3 | to2 | to1 | wrdaccerr | INTAUXL | PROTL | PRESL | PTL | 00000000 |

[2] Configuration registers doubled (one by slot) and sharing the same address, selection is done with bit IOauxen in register ssr.

[3] 32-bit FIFO access registers are shown here 8-bit for representation convenience.

[4] disATRcounter bit is only available for slot 1 since ATR counter is dedicated to slot 1.

[5] CST bit is only available for slot 1, this feature is ensured by CLKAUXen bit in ssr register for the auxiliary slot.

Another view of the registers can be obtained by looking to the functions they controlled.

We can form 4 groups: general, ISO UART, slot 1 and slot AUX (2).

The general register allows to perform a software reset of the Contact Interface, to select the slot used, to configure the time-out counters (used to process real-time tasks like WWT, CWT).

The ISO UART registers allows to control data reception and transmission through configuration and status registers.

Slot 1 and AUXiliary (2) have dedicated registers to control card power (only slot 1), card ATR (only slot 1), I/O and CLK pins slew rate (only slot 1), data baud rate, protocol and card clock configuration. Since each slot has its own registers and since these slot registers share the same addresses (grey rectangles on the following picture), the slot inherent registers are addressed when the slot is selected using general register.

This is depicted by the following picture:

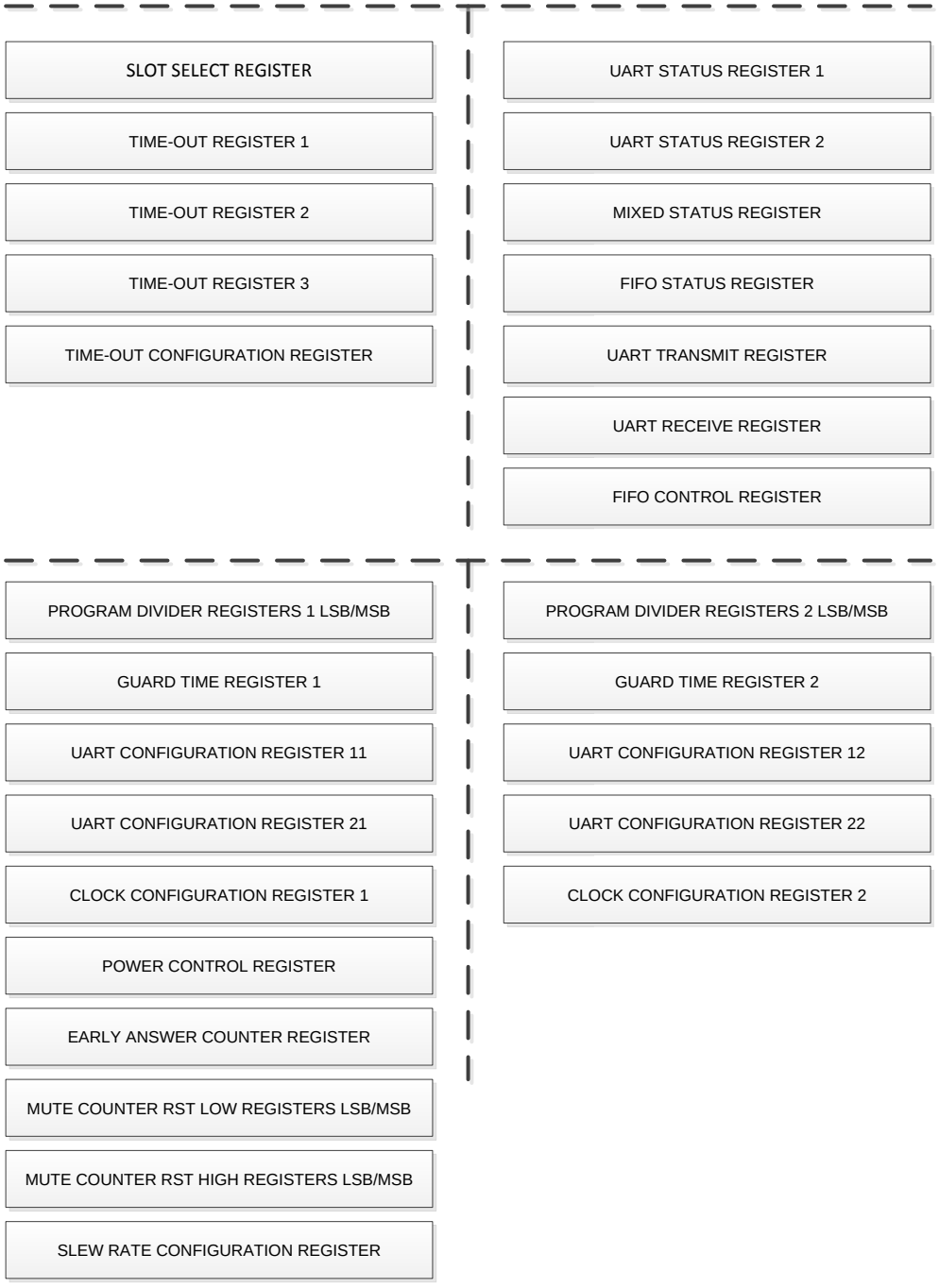



**Fig 38. Registers overview per function**

### 13.6.2 Registers description

#### 13.6.2.1 Register ct_ssr_reg (Slot Select Register)

This configuration register enables to select the I/O line used and also to reset the whole Contact UART.

**Table 249. Ct_ssr_reg (address 0000h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | - | 0 | Reserved |
| 4 | pres_con_no | R/W | 0 | PRESN CONnector Normally Open<br><br>When set to logic 0, the type of connector used for card detection is normally closed.<br>When set to logic 1, the type of connector used for card detection is normally open.<br>In conjunction with pres_pup_en bit, this enables to support the two types of connector (normally open and normally closed) for card detection.<br>Remark: this bit has to be set to the value corresponding to the connector used before any activation. |
| 3 | pres_pup_en | R/W | 0 | PRESN internal Pull UP ENable<br>When set to logic 0, an internal pull down resistance is connected to PRESN pin for card detection.<br>When set to logic 1, an internal pull up resistance is connected to PRESN pin for card detection.<br>In conjunction with pres_con_no bit, this enables to support the two types of connector (normally open and normally closed) for card detection.<br>Remark: this bit has to be set to the value corresponding to the connector used before any activation. |
| 2 | CLKAUXen | R/W | 0 | CLK AUXiliary enable<br>When set to logic 1, outputs CLKAUX clock on CLKAUX pin. This can be used also to perform clock stop; CST bit is therefore not available for the auxiliary slot in ct_ccr2_reg register. |
| 1 | IOAUXen | R/W | 0 | I/O AUXiliary enable<br>Select the second (auxiliary) slot.<br>**Note:** Changing the value of this bit (switching from one slot to the other one) resets the ISO UART block, the interrupt bits ft, fer, ovr, pe, EARLY, MUTE (ct_usr1_reg register), wrdaccerr, to1, to2 and to3 (ct_usr2_reg register), the Timers block and ct_toc_reg register. |
| 0 | softreset | R/W | 1 | When set to logic 0 this bit resets the whole Contact UART (software reset), sets to logic 1 automatically by hardware after after one clock cycle if slot 1 is not activated else after one clock cycle after slot 1 has been automatically deactivated. Software should check soft reset is finished by reading ct_ssr_reg register before any further action. |

#### 13.6.2.2 Register ct_pdr1_lsb_reg/ct_pdr2_lsb_reg (Programmable Divider Register Least Significant Byte)

This configuration register is the least significant byte of a 16-bit counter used to define the ETU.

This register is doubled: ct_pdr1_lsb_reg is dedicated to the full slot and ct_pdr2_lsb_reg is dedicated to the auxiliary slot. Both registers share the same address, the selection is done via bit IOauxen of register ct_ssr_reg.

**Table 250. ct_pdr1_lsb_reg/ct_pdr2_lsb_reg (address 0004h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7:0 | PD7 - PD0 | R/W | 0111 0100b | Least significant byte of a 16-bit counter defining the ETU. The ETU counter counts a number of cycles of the Contact Interface clock, this defines the ETU. The minimum acceptable value is 0001 0000b. |

#### 13.6.2.3 Register ct_pdr1_msb_reg/ct_pdr2_msb_reg (Programmable Divider Register Most Significant Byte)

This configuration register is the most significant byte of a 16-bit counter used to define the ETU.

This register is doubled: ct_pdr1_msb_reg is dedicated to the full slot and ct_pdr2_msb_reg is dedicated to the auxiliary slot. Both registers share the same address, the selection is done via bit IOauxen of register ct_ssr_reg.

**Table 251. ct_pdr1_msb_reg/ct_pdr2_msb_reg (address 000Ch) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7:0 | PD15 – PD8 | R/W | 0000 0001b | Most significant byte of a 16-bit counter defining the ETU. The ETU counter counts a number of cycles of the Contact Interface clock, this defines the ETU. |

The ETU (and also the card clock CLK) generation is shown by the following figure:

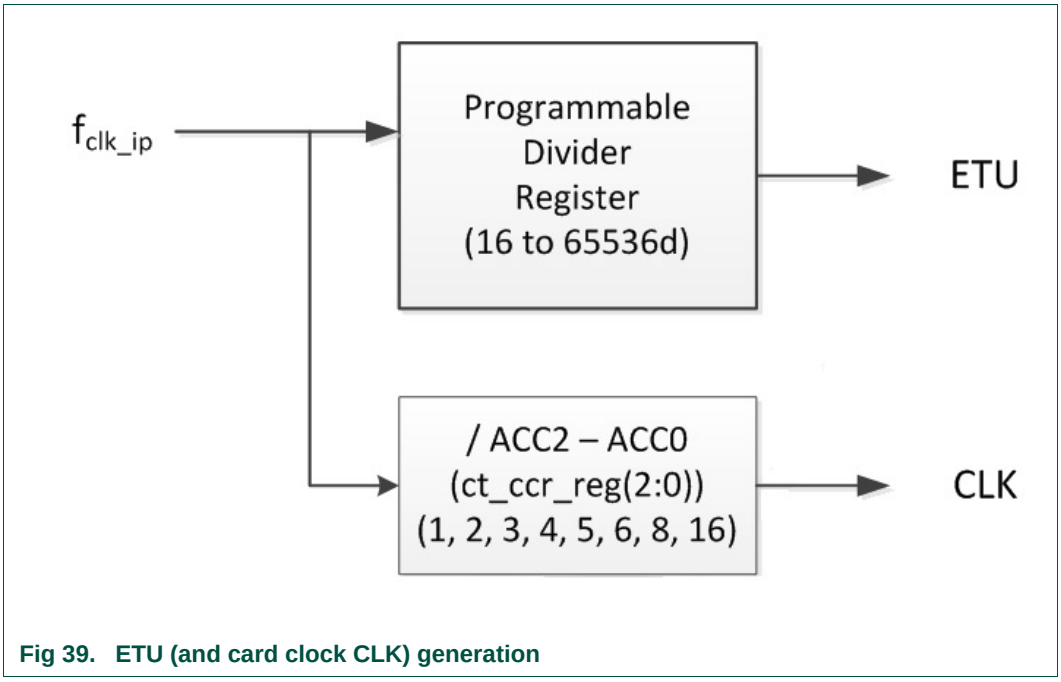



**Fig 39. ETU (and card clock CLK) generation**

#### 13.6.2.4 Register ct_fcr_reg (FIFO Control Register)

This configuration register defines the FIFO threshold (interrupt signaled by the ft bit of register ct_usr1_reg) and the number of repetition of character in case of Parity Error (interrupt signaled by the pe bit of register ct_usr1_reg).

**Table 252. ct_fcr_reg (address 000Ch) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7:5 | PEC2 - PEC0 | R/W | 000b | Parity Error Count:<br>- In protocol T = 0:<br>Set the number of allowed repetitions in reception or transmission mode before setting pe in ct_usr1_reg. The value 000 indicates that, if only one parity error has occurred, bit pe is set at logic 1; the value 111 indicates that bit pe will be set at logic 1 after 8 parity errors.<br>• If a correct character is received before the programmed error number is reached, the error counter will be reset.<br>• If the programmed number of allowed parity errors is reached, bit pe in register ct_usr1_reg will be set at logic 1.<br>• If a transmitted character has been nacked by the card, then the Contact UART will automatically retransmit it up to a number of times equal to the value programmed in bits PEC(2:0); the character will be resent at 15 ETU.<br>• If a transmitted character is considered as correct by the card after having been naked a number of times less than the value programmed in bits PEC(2:0) +1, the error counter will be reset.<br>• If a transmitted has been naked by the card a number of times equal to the value programmed in bits PEC(2:0) +1, the transmission stops and bit pe in register ct_usr1_reg is set at logic 1. The firmware is supposed to deactivate |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| | | | | the card. If not, the firmware has the possibility to pursue the transmission. By reading the number of bytes present into the FIFO (ffl bits), it can determine which character has been naked PEC +1 times by the card. It will then flush the FIFO (FIFO flush bit). The next step consits in unlocking the transmission using dispe bit. By writing this bit at logic level one (and then at logic level zero if the firmware still wants to check parity errors), the transmission is unlocked. The firmware can now write bytes into the FIFO.<br>• In transmission mode, if bits PEC(2:0) are at logic 0, then the automatic retransmission is invalidated. There is no retransmission; the transmission continues with the next character sent at 13 ETU.<br><br>- In protocol T = 1:<br>The error counter has no action: bit pe is set at logic 1 at the first wrong received character. |
| 4:0 | ftc(4:0) | R/W | 0001b | FIFO Threshold Configuration<br>Define the number of received or transmitted characters in the FIFO triggering the ft bit in ct_usr1_reg. The FIFO depth is 32 bytes.<br>In reception mode, it enables to know that a number equals to ftc(4:0) + 1 bytes have been received. In transmission mode, ftc(4:0) equals to the number of remaining bytes into the FIFO.<br>Be careful: in reception mode 00000 = length 1, and in transmission mode 00000 = length 0. |

#### 13.6.2.5 Register ct_gtr1_reg/ct_gtr2_reg (Guard Time Register)

This configuration register is used to store the guard time given by the card during ATR.

This register is doubled: ct_gtr1_reg is dedicated to the full slot and ct_gtr2_reg is dedicated to the auxiliary slot. Both registers share the same address, the selection is done via bit IOauxen of register ct_ssr_reg.

**Table 253. ct_gtr1_reg/ct_gtr2_reg (address 0010h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7:0 | GT7-GT0 | R/W | 0000 0000b | Guard Time<br>Value used by the Contact UART notably in transmission mode.<br>The Contact UART will wait this number of ETUs before transmitting the character.<br>In protocol T=1, gtr = FFh means operation at 11 ETUs.<br>In protocol T=0, gtr = FFh means operation at 12 ETUs. |

#### 13.6.2.6 Register ct_ucr11_reg/ct_ucr12_reg (UART Configuration Register 1)

This configuration register defines the reception and transmission settings.

This register is doubled: ct_ucr11_reg is dedicated to the full slot and ct_ucr12_reg is dedicated to the auxiliary slot. Both registers share the same address, the selection is done via bit IOauxen of register ct_ssr_reg.

**Table 254. ct_ucr11_reg/ct_ucr12_reg (address 0014h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | - | 0 | Reserved |
| 5 | FIP | R/W | 0b | Force Inverse Parity<br>If bit FIP is set to logic 1, the Contact UART will NAK a correctly received character, and will transmit characters with wrong parity bits. |
| 4 | FC | R/W | 0b | |
| 3 | PROT | R/W | 0b | PROTocol<br>Selects the protocol: logic 1 means T=1 and logic 0 T=0. |
| 2 | T/R | R/W | 0b | Transmit/Receive<br>Defines the mode: logic 1 means transmission and logic 0 reception.<br>Bit T/R is set by software for transmission mode. Bit T/R is automatically reset to logic 0 by hardware, if bit LCT has been used before transmitting the last character.<br>Note that when switching from/to reception to/from transmission mode, the FIFO is flushed. Any remaining bytes are lost. |
| 1 | LCT | R/W | 0b | Last Character to Transmit<br>Bit LCT is set to logic 1 by software before writing the last character to be transmitted in register ct_utr_reg. It allows automatic change to reception mode. It is reset to logic 0 by hardware at the end of a successful transmission after 11.75 ETUs in protocol T = 0 and after 10.75 ETUs in protocol T = 1. When bit LCT is being reset to logic 0, bit T/R is also reset to logic 0 and the UART is ready to receive a character.<br>LCT bit can be set to logic 1 by software not only when writing the last character to be transmitted but also during the transmission or even at the beginning of the transmission. It will be taken into account when the FIFO becomes empty, which implies for the software to be able to regularly re-load the FIFO when transmitting more than 32 bytes to ensure there is at least one byte into the FIFO as long as the transmission is not finished. Else, a switch to reception mode will prematurely occur before having transmitted all the bytes. |
| 0 | CONV | R/W | 0b | CONVention<br>Bit CONV is set to logic 1 if the convention is direct. Bit CONV is either automatically written by hardware according to the convention detected during ATR, or by software if the bit AUTOCONV in register ct_ucr1_reg is set to logic 1. |

#### 13.6.2.7 Register ct_ucr21_reg/ct_ucr22_reg (UART Configuration Register 2)

This configuration register defines the reception and transmission settings.

This register is doubled: ct_ucr21_reg is dedicated to the full slot and ct_ucr22_reg is dedicated to the auxiliary slot. Both registers share the same address, the selection is done via bit IOauxen of register ct_ssr_reg.

**Table 255. ct_ucr21_reg/ct_ucr22_reg (address 0018h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7 | wrdacc | R/W | 0b | FIFO WoRD ACCess<br>When set to logic 1, the FIFO supports word (4 bytes) access (read and write), access failure is indicated by bit wrdaccerr in register ct_usr2_reg.<br>When set to logic 0, the FIFO supports byte access (read and write). |
| 6 | FIFO flush | R/W | 0b | FIFO flush<br>When set to logic 1, the FIFO is flushed whatever the mode (reception or transmission) is. It can be used before any reception or transmission of characters but not while receiving or transmitting a character.<br>It is reset to logic 0 by hardware after one *clk_ip* cycle. |
| 5 | disintaux | R/W | 0b | DISable INTAUX<br>When set to logic 1 the bit INTAUXL in register ct_usr2_reg will not generate interrupt. |
| 4 | disATRcounter | R/W | 0b | DISable ATR counter<br>- Slot 1:<br>When set to logic 1, the bits EARLY and MUTE in register ct_usr1_reg will not generate interrupt.<br>This bit should be set before activating.<br>- Slot AUX:<br>This bit is not available for the auxiliary slot (ct_ucr22_reg) since ATR counter is dedicated to slot 1. |
| 3 | dispe | R/W | 0b | DISable Parity Error interrupt bit<br>When set to logic 1, the parity is not checked in both reception and transmission modes, the bit pe in register ct_usr1_reg will not generate interrupt. |
| 2 | disft | R/W | 0b | DISable Fifo Threshold interrupt bit<br>When set to logic 1 the bit ft in register ct_usr1_reg will not generate interrupt. |
| 1 | MAN BGT | R/W | 0b | MANual BGT<br>When set to logic 1, BGT is managed by software, else by hardware. |

#### 13.6.2.8 Register ct_ccr1_reg/ct_ccr2_reg (Clock Configuration Register)

This configuration register defines the card clock frequency.

This register is doubled: ct_ccr1_reg is dedicated to the full slot (card 1) and ct_ccr2_reg is dedicated to the auxiliary slot (card 2). Both registers share the same address, the selection is done via bit IOauxen of register ct_ssr_reg.

**Table 256. ct_ccr1_reg/ct_ccr2_reg (address 001Ch) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | - | 0 | reserved |
| 5 | FIP | R/W | 0b | Stop HIGH or LOW<br>- Slot 1:<br>If bits SAN = 0 and CST = 1, then the clock is stopped at LOW level if bit SHL = 0, and at HIGH level if bit SHL = 1.<br>If bit SAN = 1, then contact CLK is the copy of the value of bit SHL.<br><br>- Slot AUX:<br>If bits SAN = 0 and CLKAUXen = 0, then the clock is stopped at LOW level if bit SHL = 0, and at HIGH level if bit SHL = 1 (CLKAUX output is inverted). SHL bit value should not change while CLKAUXen = 1: dynamic change is not supported. SHL bit value should be chosen before enabling CLKAUX clock with CLKAUXen bit.<br>If bit SAN = 1, then contact CLKAUX is the copy of the value of bit SHL. |
| 4 | CST | R/W | 0b | Clock STop<br>- Slot 1:<br>In the case of an asynchronous card, bit CST defines whether the clock to the card is stopped or not; if bit CST is reset to logic 0, then the clock is determined by bits ACC0, ACC1 and ACC2.<br>- Slot AUX:<br>This bit is not available for the auxiliary slot (ct_ccr2_reg) since clock stop feature is supported using CLKAUXen bit in ct_ssr_reg register. |
| 3 | SAN | R/W | 0b | Synchronous/Asynchronous Card<br>- Slot 1:<br>When set to logic 1, the Contact UART supports synchronous card. The Contact UART is then bypassed, only bit 0 of registers ct_urr_reg and ct_utr_reg is connected to pin I/O. In this case, the card clock is controlled by bit SHL and RST card is controlled by bit RSTIN in register ct_pcr_reg. When set to logic 0, the Contact UART supports asynchronous card. Dynamic change (while activated) is not supported. The choice should be done before activating the card.<br>- Slot AUX:<br>When set to logic 1, the Contact UART supports synchronous card. The Contact UART is then bypassed, only bit 0 of registers ct_urr_reg and ct_utr_reg is connected to pin I/O. In this case, the card clock is controlled by bit SHL. When set to logic 0, the Contact UART supports asynchronous card. Dynamic change (while CLKAUXen = 1) is not supported. The choice should be done before enabling CLKAUX clock. |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 2:0 | ACC2 - ACC0 | R/W | 000b | Asynchronous Card Clock<br>Defines the card clock frequency:<br>• 000: card clock frequency = fclk_ip<br>• 001: card clock frequency = fclk_ip /2<br>• 010: card clock frequency = fclk_ip /3<br>• 011: card clock frequency = fclk_ip /4<br>• 100: card clock frequency = fclk_ip /5<br>• 101: card clock frequency = fclk_ip /6<br>• 110: card clock frequency = fclk_ip /8<br>• 111: card clock frequency = fclk_ip /16<br>All frequency changes are synchronous, thus ensuring that no spikes or unwanted pulse widths occur during changes.<br>In conjunction with registers ct_etucr_lsb_reg and ct_etucr_msb_reg, the bits ACC2, ACC1 and ACC0 defines the baudrate used by the Contact UART. |

#### 13.6.2.9 Register ct_pcr_reg (Power Control Register)

This configuration register enables to start or stop card sessions, define the card supply voltage (5V, 3V, 1.8V) and manage the card contacts C4 and C8 (also known as AUX1 and AUX2).

**Table 257. ct_pcr_reg (address 0020h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7 | C8 | R/W | 1b | Contact 8<br>Writing C8 bit writes the corresponding value on C8 pin. Reading C8 bit reads the state of C8 pin. |
| 6 | C4 | R/W | 1b | Contact 4<br>Writing C4 bit writes the corresponding value on C4 pin.<br>Reading C4 bit reads the state of C4 pin. |
| 5 | RESERVED | - | 0 | reserved |
| 4 | RSTIN | | | Reset bit<br>Synchronous card: when set to logic 1, RST pin is set to logic 1;<br>when set to logic 0, RST pin is set to logic 0.<br>Asynchronous card: RST is controlled by hardware (ATR management). |
| 3:2 | vccsel1 - vccsel0 | R/W | 00b | "VCC selection<br>Defines VCC voltage:<br>• 00: VCC = 5 V<br>• 01: VCC = 3 V<br>• 10 or 11: VCC = 1.8 V<br>Dynamic change (while activated) is not supported. The choice should be done before activating the card. |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 1 | WARM | R/W | 0b | When set to logic 1: a warm reset procedure is started.<br>Set to logic 0 by hardware when a START bit is detected or when MUTE bit is set to logic 1. |
| 0 | START | R/W | 0b | When set to logic 1: starts the activation sequence and cold reset procedure (only if the PTL and PROTL bits in register ct_usr2_reg are logic 0 and the PRES bit in register ct_msr_reg is logic 1).<br>When set to logic 0: starts the deactivation sequence.<br>Remark: the bits pres_pup_en and pres_con_no in ct_ssr_reg register should have been set prior to any activation. |

#### 13.6.2.10 Register ct_ecr_reg (Early answer CounteR)

This configuration register enables to program the value of an 8-bit counter used to check whether the card has answered too early.

**Table 258. ct_ecr_reg (address 0024h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7:0 | EC7 - EC0 | R/W | 1010 1010b | Early answer Counter<br>Programmable 8-bit clock counter (see ATR counter Functional Description). |

#### 13.6.2.11 Register ct_mclr_lsb_reg (Mute card CounteR RST Low register Least Significant Byte)

This configuration register is the least significant byte of a 16-bit counter used to check whether the card answers when RST is logic 0.

**Table 259. ct_mclr_lsb_reg (address 0028h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7:0 | MCL7 - MCL0 | R/W | 0111 0100b | Least significant byte of a programmable 16-bit clock counter (see ATR counter Functional Description). |

#### 13.6.2.12 Register ct_mclr_msb_reg (Mute card CounteR RST Low register Most Significant Byte)

This configuration register is the most significant byte of a 16-bit counter used to check whether the card answers when RST is logic 0.

**Table 260. ct_mclr_msb_reg (address 002Ch) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 7:0 | MCL15 – MCL8 | R/W | 1010 0100b | Most significant byte of a programmable 16-bit clock counter (see ATR counter Functional Description). |

#### 13.6.2.13 Register ct_mchr_lsb_reg (Mute card CounteR RST High register Least Significant Byte)

This configuration register is the least significant byte of a 16-bit counter used to check whether the card is mute when RST is logic 1.

**Table 261. ct_mchr_lsb_reg (address 0030h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | rReserved |
| 7:0 | MCH7 - MCH0 | R/W | 0111 0100b | Least significant byte of a programmable 16-bit clock counter (see ATR counter Functional Description). |

#### 13.6.2.14 Register ct_mchr_msb_reg (Mute card CounteR RST High register Most Significant Byte)

This configuration register is the most significant byte of a 16-bit counter used to check whether the card is mute when RST is logic 1.

**Table 262. ct_mchr_msb_reg (address 0034h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7:0 | MCH15 – MCH8 | R/W | 1010 0100b | Most significant byte of a programmable 16-bit clock counter (see ATR counter Functional Description). |

#### 13.6.2.15 Register ct_srr_reg (Slew Rate configuration Register)

This configuration register defines the slew rate values on the I/O and CLK lines.

**Table 263. ct_ssr_reg (address 0038h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | - | 0 | reserved |
| 5:4 | vcc rise sel1 - vcc rise sel0 | R/W | 00b | VCC rise window selection<br>To rise from 0 to its selected value (see ct_pcr_reg register), the" time needed by VCC depends on the capacitance. For more flexibility, it is possible to choose between 4 window time:<br>• 00: t = 5T (~120 us)<br>• 01: t = 10T (~240 us)<br>• 10: t = 16T (~380 us)<br>• 11: t = 21T (~500 us) - recommended<br>"See the activation sequence description for T definition. |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| | | | | Dynamic change (while activated) is not supported. The choice" should be done before activating the card. |
| 3:2 | clk_sr1 - clk_sr0 | R/W | 00b | CLK slew rate<br>Card clock slew rate selection. |
| 1:0 | io_sr1 - io_sr0 | R/W | 00b | I/O slew rate<br>Card I/O slew rate selection. |

#### 13.6.2.16 Register ct_urr_reg/ct_utr_reg (UART Receive Register/UART Transmit Register)

These two data registers share the same address but the first one can only be read and the second only be written. They constitute the access to the Contact UART FIFO respectively in reception and transmission modes.

**Table 264. ct_urr_reg/ct_utr_reg (address 003Ch, 0040h, 0044h, 0048h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:0 | URR31 - URR0 | R | 0000 0000h | Uart Receive Register<br>Asynchronous card:<br>When the controller wants to read a character from the card stored into the FIFO, it reads it from this register in direct convention.<br>- In case of byte access (bit wrdacc = 0 in register ct_ucr2_reg), the byte is read on the 8 least significant bits:<br>URR31 - URR8: 000000h (unvalid bytes)<br>URR7 - URR0: byte (received byte)<br>- In case of word access (bit wrdacc = 1 in register ct_ucr2_reg), 4 characters are read:<br>URR31 - URR24: byte 4 (last received byte)<br>URR23 - URR16: byte 3<br>URR15 - URR8: byte 2<br>URR7 - URR0: byte 1 (first received byte)<br>If less than 4 bytes are present into the FIFO, the missing bytes are read 00h or previous values and interrupt bit wdraccerr in register ct_usr2_reg is set to logic 1.<br>Synchronous card:<br>In reception mode, the data from the card is available to bit UR0 after a read operation of register ct_urr_reg; the FIFO is bypassed. |
| 31:0 | UTR31 - UTR0 | W | 0000 0000h | Uart Transmit Register<br>Asynchronous card:<br>When the micro-controller wants to transmit a character to the card, it writes the byte in direct convention in the utr register (this byte will be stored into the FIFO).<br>- In case of byte access (bit wrdacc = 0 in register ct_ucr2_reg), 1 byte is written into the FIFO:<br>UTR31 - UTR8: 000000h (unused bytes)<br>UTR7 - UTR0 : byte (byte to transmit) |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| | | | | - In case of word access (bit wrdacc = 1 in register ct_ucr2_reg), 4 bytes are written into the FIFO:<br>UTR31 - UTR24: byte 4 (last byte to transmit)<br>UTR23 - UTR16: byte 3<br>UTR15 - UTR8 : byte 2<br>UTR7 - UTR0 : byte 1 (first byte to transmit)<br>Synchronous card:<br>In transmission mode, the data (UTR0) is written on the I/O line of the card when register ct_utr_reg has been written to; the FIFO is bypassed. |

#### 13.6.2.17 Register ct_tor1_reg (Time-Out Register 1)

This configuration register enables to program the value of an 8-bit ETU counter used to check some timings (CWT, BWT...).

**Table 265. ct_tor1_reg (address 004Ch) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7:0 | TOL7 - TOL0 | W | 0000 | Time-Out Latched |
| | | | 0000b | Programmable 8-bit ETU counter (see register ct_toc_reg). |

#### 13.6.2.18 Register ct_tor2_reg (Time-Out Register 2)

This configuration register enables to program the value of an 8-bit ETU counter used to check some timings (CWT, BWT...).

**Table 266. ct_tor2_reg (address 0050h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7:0 | TOL15 - TOL8 | W | 0000 | Time-Out Latched |
| | | | 0000b | Programmable 8-bit ETU counter (see register ct_toc_reg). |

#### 13.6.2.19 Register ct_tor3_reg (Time-Out Register 3)

This configuration register enables to program the value of an 8-bit ETU counter used to check some timings (CWT, BWT...).

**Table 267. ct_tor3_reg (address 0054h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | reserved |
| 7:0 | TOL24 – TOL16 | W | 0000 | **Time-Out Latched:** |
| | | | 0000b | Programmable 8-bit ETU counter (see register ct_toc_reg). |

#### 13.6.2.20 Register ct_toc_reg (Time-Out Configuration)

This configuration register is used for setting different configurations of the time-out counter as given in Table 269 “Timer settings”; all other configurations are undefined.

**Table 268. ct_toc_reg (address 0058h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7:0 | TOC7 – TOC0 | R/W | 0000 | Time-Out Configuration |
| | | | 0000b | Time-out counter mode selection (see table below). |

**Table 269. Timer settings**

| TOC Value | Operating Mode |
|---|---|
| 00h | All counters are stopped |
| 05h | Counter 1 is an 8-bit auto-reload counter, and counters 3 and 2 form a 16-bit counter. Counters 2 and 3 are stopped; counter 1 continues to operate in auto-reload mode. |
| 07h | Counters 1, 2 and 3 are three independent 8-bit counters. Counters 2 and 3 are stopped; counter 1 continues to operate in auto-reload mode. |
| 61h | Counter 1 is stopped, and counters 3 and 2 form a 16-bit counter. Counting the value stored in registers ct_tor3_reg and ct_tor2_reg is started after 61h is written in register ct_toc_reg. An interrupt is given, and bit TO3 is set within register ct_usr2_reg when the terminal count is reached. The counter is stopped by writing 00h in register ct_toc_reg, and sHALl be stopped before reloading new values in registers ct_tor2_reg and ct_tor3_reg. In this configuration, registers ct_tor3_reg and ct_tor2_reg must not be all zero. |
| 65h | Counter 1 is an 8-bit auto-reload counter, and counters 3 and 2 form a 16-bit counter. Counter 1 starts counting the content of register ct_tor1_reg on the first start bit (reception or transmission) detected on pin I/O after 65h is written in register ct_toc_reg. When counter 1 reaches its terminal count, an interrupt is given, bit TO1 in register ct_usr2_reg is set, and the counter automatically restarts the same count until it is stopped. Changing the content of register ct_tor1_reg during a count is not allowed. Counting the value stored in registers ct_tor3_reg and ct_tor2_reg is started after 65H is written in register ct_toc_reg. When the counter reaches its terminal count, an interrupt is given and bit TO3 is set within register ct_usr2_reg. Both counters are stopped when 00H is written in register ct_toc_reg. Counters 3 and 2 sHALl be stopped by writing 05h in register ct_toc_reg before reloading new values in registers ct_tor2_reg and ct_tor3_reg. In this configuration, registers ct_tor3_reg, ct_tor2_reg and ct_tor1_reg must not be all zero. |
| 71h | Counter 1 is stopped, and counters 3 and 2 form a 16-bit counter. Counting the value stored in registers ct_tor3_reg and ct_tor2_reg is started on the first start bit detected on pin I/O (reception or transmission) after the value has been written, and then on each subsequent start bit. It is possible to change the content of registers ct_tor3_reg and ct_tor2_reg during a count; the current count will not be affected and the new count value will be taken into account at the next start bit. An interrupt is given, and bit TO3 is set within register ct_usr2_reg when the terminal count is reached. The counter is stopped by writing 00h in ct_toc_reg register. In this configuration, registers ct_tor3_reg and ct_tor2_reg must not be all zero. |
| 73h | Counters 1, 2 and 3 are three independent 8-bit counters. Counter 1 is stopped. Counter 2 starts counting the content of register ct_tor2_reg on the first start bit (reception or transmission) detected on pin I/O after 73h is written in register ct_toc_reg, and then on each subsequent start bit. It is possible to change the content of register ct_tor2_reg during a count; the current |

| TOC Value | Operating Mode |
|---|---|
| | count will not be affected and the new count value will be taken into account at the next start bit. An interrupt is given, and bit TO2 is set within register ct_usr2_reg when the terminal count is reached. Counting the value stored in register ct_tor3_reg is started after 73h is written in register ct_toc_reg. An interrupt is given, and bit TO3 is set within register ct_usr2_reg when the terminal count is reached. The counter sHALl be stopped before reloading new value in ct_tor3_reg register. The counters are stopped by writing 00h in register ct_toc_reg. In this configuration, registers ct_tor3_reg and ct_tor2_reg must not be all zero. |
| 75h | Counter 1 is an 8-bit auto-reload counter, and counters 3 and 2 form a 16-bit counter. Counter 1 starts counting the content of register ct_tor1_reg on the first start bit (reception or transmission) detected on pin I/O after 75h is written in register ct_toc_reg. When counter 1 reaches its terminal count, an interrupt is given, bit TO1 in register ct_usr2_reg is set, and the counter automatically restarts the same count until it is stopped. Changing the content of register ct_tor1_reg during a count is not allowed. Counting the value stored in registers ct_tor3_reg and ct_tor2_reg is started on the first start bit detected on pin I/O (reception or transmission) after the value has been written, and then on each subsequent start bit. It is possible to change the content of registers ct_tor3_reg and ct_tor2_reg during a count; the current count will not be affected and the new count value will be taken into account at the next start bit. An interrupt is given, and bit TO3 is set within register ct_usr2_reg when the terminal count is reached. The counters are stopped by writing 00h in ct_toc_reg register. In this configuration, registers ct_tor3_reg, ct_tor2_reg and ct_tor1_reg must not be all zero. |
| 77h | Counters 1, 2 and 3 are three independent 8-bit counters. Counter 1 is an 8-bit auto-reload counter which starts counting the content of register ct_tor1_reg on the first start bit (reception or transmission) detected on pin I/O after 77h is written in register ct_toc_reg. When counter 1 reaches its terminal count, an interrupt is given, bit TO1 in register ct_usr2_reg is set, and the counter automatically restarts the same count until it is stopped. Changing the content of register ct_tor1_reg during a count is not allowed. Counter 2 starts counting the content of register ct_tor2_reg on the first start bit (reception or transmission) detected on pin I/O after 77h is written in register ct_toc_reg, and then on each subsequent start bit. It is possible to change the content of register ct_tor2_reg during a count; the current count will not be affected and the new count value will be taken into account at the next start bit. An interrupt is given, and bit TO2 is set within register ct_usr2_reg when the terminal count is reached. Counting the value stored in register ct_tor3_reg is started after 77h is written in register ct_toc_reg. An interrupt is given, and bit TO3 is set within register ct_usr2_reg when the terminal count is reached. The counter sHALl be stopped before reloading new value in ct_tor3_reg register. The counters are stopped by writing 00h in register ct_toc_reg. In this configuration, registers ct_tor3_reg, ct_tor2_reg and ct_tor1_reg must not be all zero. |
| 7Ch | Counters 3, 2 and 1 are wired as a single 24-bit counter. Counting the value stored in registers ct_tor3_reg, ct_tor2_reg and ct_tor1_reg is started on the first start bit detected on pin I/O (reception or transmission) after the value has been written, and then on each subsequent start bit. It is possible to change the content of registers ct_tor3_reg, ct_tor2_reg and ct_tor1_reg during a count; the current count will not be affected and the new count value will be taken into account at the next start bit. An interrupt is given, and bit TO3 is set within register ct_usr2_reg when the terminal count is reached. The counter is stopped by writing 00H in register ct_toc_reg. In this configuration, registers ct_tor3_reg, ct_tor2_reg and ct_tor1_reg must not be all zero. |
| F3h | Same configuration as value 73h, except that the 8-bit counters will be stopped at the end of the 12th ETU following the first start bit detected after F3h has been written in ct_toc_reg register. |
| F7h | Same configuration as value 77h, except that the 8-bit counters will be stopped at the end of the 12th ETU following the first start bit detected after F7h has been written in ct_toc_reg register. |

#### 13.6.2.21 Register ct_fsr_reg (FIFO Status register)

This status register enables to know how many bytes are present into the FIFO; it doesn't generate any interrupt.

**Table 270. ct_fsr_reg (address 005Ch) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | - | 0 | reserved |
| 5:0 | ffl5 – ffl0 | R | 0 0000b | FIFO Fulfilment Level<br>Gives the number of bytes present into the FIFO. |

#### 13.6.2.22 Register ct_msr_reg (Mixed Status Register)

This status register is intended for polling; it doesn't generate any interrupt.

**Table 271. ct_msr_reg (address 0060h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:3 | RESERVED | - | 0 | reserved |
| 2 | INTAUX | R | 0b | Auxiliary interrupt<br>Bit INTAUX is the copy of pin INTAUX. |
| 1 | BGT | R | 0b | Block Guard Time<br>In protocol T = 1, bit BGT is linked with a 22-ETU counter, which is started at every start bit on pin I/O. Bit BGT is set to logic level one, if the count is finished before the next start bit. This helps to verify that the card has not answered before 22 ETU after the last transmitted character, or that the reader is not transmitting a character before 22 ETU after the last received character.<br>In protocol T = 0, bit BGT is linked with a 16-ETU counter, which is started at every start bit on pin I/O. Bit BGT is set to logic level one, if the count is finished before the next start bit. This helps to verify that the card has not answered before 16 ETU after the last transmitted character, or that the reader is not transmitting a character before 16 ETU after the last received character. |
| 0 | PRES | R | 0b | PRESence<br>Set to logic 1 when the card is present.<br>Set to logic 0 when the card is not present or has been removed.<br>Remark: the bits pres_pup_en and pres_con_no in ct_ssr_reg register should have been set prior to any check of card presence. |

#### 13.6.2.23 Register ct_usr1_reg (UART Status Register 1)

This register is an interrupt register (together with ct_usr2_reg register): these bits coming from the Contact UART core are used to manage the reception & transmission of characters. Read this register enables to know what the cause of the interrupt is. The bits are set to logic 1 by hardware and set to logic 0 by reading (with a hardware mechanism avoiding the loss of incoming interrupt while reading).

**Table 272. ct_usr1_reg (address 0064h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | - | 0 | Reserved |
| 5 | MUTE | R | 0b | During ATR, set to logic 1 when the card has not answered (it has not sent its ATR) within the time defined by the mute counter while RST was at logic 1 (see registers ct_mcrh_lsb_reg & ct_mcrh_msb_reg) or when the card has answered while RST was at logic 0 (see registers ct_mcrl_lsb_reg & ct_mcrl_lsb_reg).<br>Set to logic 0 after reading the byte. |
| 4 | EARLY | R | 0b | During ATR, set to logic 1 when the card has answered two early within the time defined by the early counter (see register ct_ecr_reg) while RST was at logic 1 or when the card has answered while RST was at logic 0.<br>Set to logic 0 after reading the byte. |
| 3 | pe | R | 0b | Parity Error<br>- In protocol T=0, it is high if the character has been received with parity error a number of time equals to the number written in PEC(2:0) of register ct_fcr_reg +1 or if the transmitted character has been NAKed by the card a number of times equal to the value programmed in bits PEC(2:0) +1. It is set at 10.5 ETU in the reception mode and at 11.5 ETU in the transmission mode. A character received with a parity error is not stored into the FIFO and the card is supposed to repeat this character.<br>- In protocol T=1, it is high when the parity error has been detected.<br>A character with a parity error is stored into the FIFO and the parity error counter is not active.<br>Set to logic 0 after reading the byte. |
| 2 | ovr | R | 0b | OVerRun<br>Set to logic 1 when a new character has been received whilst the FIFO was full. In this case, at least one character has been lost.<br>Set to logic 0 after reading the byte. |
| 1 | fer | R | 0b | Framing ERror<br>Set to logic 1 when the I/O line was not in the high-impedance state at 10.25 ETUs after a start bit.<br>Set to logic 0 after reading the byte. |
| 0 | Ft | R | 0b | Fifo Threshold<br>Set to logic 1 in reception mode if the number of received bytes in the FIFO equals to the number written in bits ft(4:0) + 1 of the register ct_fcr_reg. This bit goes high 10.5 ETUs after the start bit of the (ft(4:0) + 1)th received character. In transmission mode, it is set to logic 1 when the number of remaining bytes to transmit in the FIFO equals to the number written in bits ft(4:0) of the register ct_fcr_reg. This bit goes high 9.5 ETUs after the start bit of the (32 - ft(4:0))th transmitted character.<br>Set to logic 0 after reading the byte. |

**Remark:** When one of the bits ft, fer, ovr, pe, MUTE, EARLY is set at logic 1, then the interrupt line is set at logic 1.

**Remark:** When both bits EARLY and MUTE are high, it means that the card answered (sent its ATR) too early between 200 CLK periods after I/O line goes high and before RST goes high.

#### 13.6.2.24 Register ct_usr2_reg (UART Status Register 2)

This register is an interrupt register (together with ct_usr1_reg register): these bits coming from the Contact UART core are used to manage the reception & transmission of characters. Read this register enables to know what the cause of the interrupt is. The bits are set to logic 1 by hardware and set to logic 0 by reading (with a hardware mechanism avoiding the loss of incoming interrupt while reading).

**Table 273. ct_usr2_reg (address 0068h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | - | 0 | Reserved |
| 7 | to3 | R | 0b | **Time Out counter3:**<br>Set to logic 1 when the timer counter 3 has reached its terminal count.<br>Set to logic 0 after reading the byte. |
| 6 | to2 | R | 0b | **Time Out counter2:**<br>Set to logic 1 when the timer counter 2 has reached its terminal count.<br>Set to logic 0 after reading the byte. |
| 5 | to1 | R | 0b | **Time Out counter1:**<br>Set to logic 1 when the timer counter 1 has reached its terminal count.<br>Set to logic 0 after reading the byte. |
| 4 | wrdaccerr | R | 0b | **WoRD ACCess ERRor:**<br>Set to logic 1 if a word (bit wrdacc = 1 in register ct_ucr21_reg/ct_ucr22_reg) read access is attempted with less than 4 bytes present into the FIFO.<br>Set to logic 0 after reading the byte. |
| 3 | INTAUXL | R | 0b | **AUXiliary INTerrupt Latched:**<br>Set to logic 1 if the level on pin INTAUX has been changed.<br>Set to logic 0 after reading the byte. |
| 2 | PROTL | R | 0b | **PROTection Latched:**<br>Set to logic 1 when an overload occurs.<br>Set to logic 0 after reading the byte. |
| 1 | PRESL | R | 0b | **PRESence Latched:**<br>Set to logic 1 when the card has been inserted or extracted.<br>Set to logic 0 after reading the byte.<br>Remark: the bits pres_pup_en and pres_con_no in ct_ssr_reg register should have been set prior to any check of card presence. |
| 0 | PTL | R | 0b | **Protection Temperature Latched:**<br>Set to logic 1 when an overheating occurs.<br>Set to logic 0 after reading the byte. |

# 14. PN7462AU Interfaces

## 14.1 I2C Master Interface

The I2C bus is a simple two-wire bi-directional serial communication interface that is intended for inter-IC communication over short distances. The I2C-bus uses only two wires: a serial clock line (SCL) and a serial data line (SDA). Each device is recognized by a unique address. The I2C master controller is one of two I2C controllers supported by PN7462AU. The I2C slave controller is described in Section 14.3.1.

### 14.1.1 I2C Master features:

- Standard I2C compliant bus interface with open-drain pins.
- Supports standard-mode, fast mode and fast mode plus (up to 1 MBds)
- Supports I2C master mode only
- Programmable clocks allowing versatile rate control
- Clock stretching
- 7-bits and 10-bits I2C slave addressing
- LDM/STM instruction support
- Maximum data frame size up to 1024 bytes

### 14.1.2 General description

A typical I ²C bus configuration is shown in Fig 40. The master device generates all of the serial clock pulses and the START and STOP conditions. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I2C-bus:

- Data transfer from a master transmitter to a slave receiver. The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte.
- Data transfer from a slave transmitter to a master receiver. The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. Next follows the data bytes transmitted by the slave to the master. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a “not acknowledge” is returned.

A transfer is ended with a STOP condition or with a Repeated START condition. Since a Repeated START condition is also the beginning of the next serial transfer, the I2C bus will not be released.

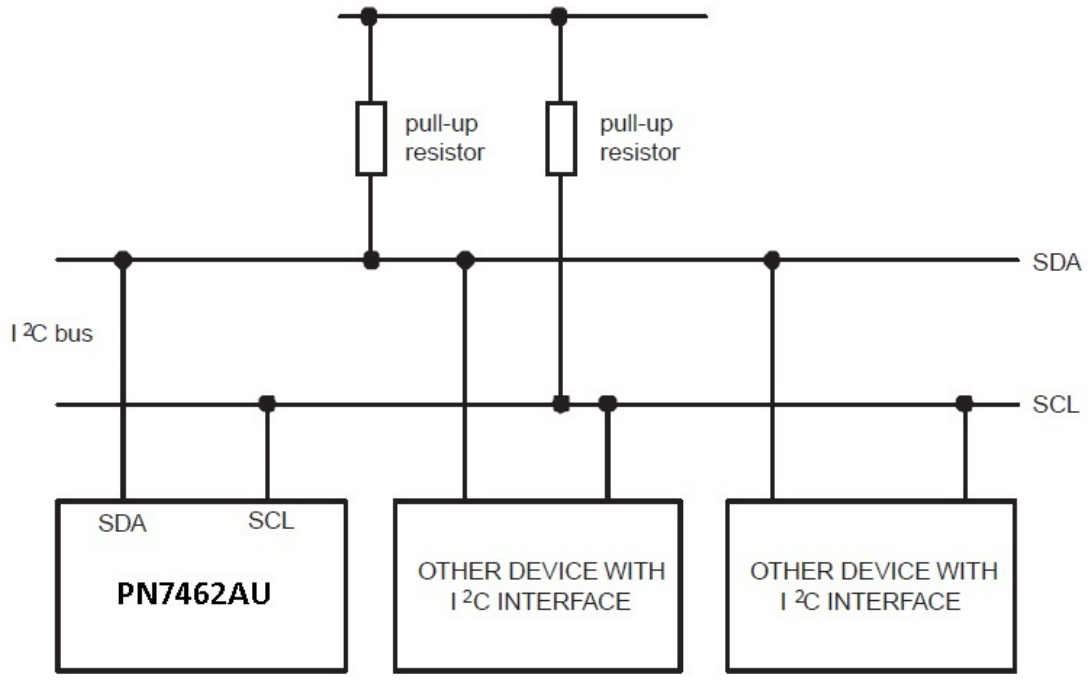



**Fig 40. I2C bus configuration**

The I2C Master can operate in two modes: I2C Master Transmitter and I2C Master Receiver.

In the master transmitter mode data is transmitted from master to slave. The first byte transmitted contains the slave address and Write bit. The data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. The I2C Master in Master Transmitter mode supports the automatic continuation of the I2C transmission without the need of altering the data in the FIFO or the register content. The Stop condition is generated when the byte sent is equal to the content of the byte count register. In case of error during the ongoing communication, the FIFO has to be cleared (corresponding register to clear FIFO based on interrupt error). When the FIFO is empty during the transmission, the block will automatically stretch the clock of the previous byte (after slave acknowledge) until a new byte is written.

The transfer in the master receiver mode is initiated in the same way as in the master transmitter mode. The number of bytes to receive must be specified in the byte count register. The I2C slave can always stretch the clock if it is not able to transmit any more data. The Stop condition is generated when the number of bytes received is equal to the byte count register content. When the FIFO is full, the clock is stretched before sending the bit acknowledge to the slave. An interrupt will be raised to indicate that some FIFO data need to be fetched.

### 14.1.3 Pin description

**Table 274. I²C-bus pin description**

| Pin Name | Pin description |
|---|---|
| SDA | I²C Serial Data |
| SCL | I²C Serial Clock |

### 14.1.4 TX/RX programming Flowcharts

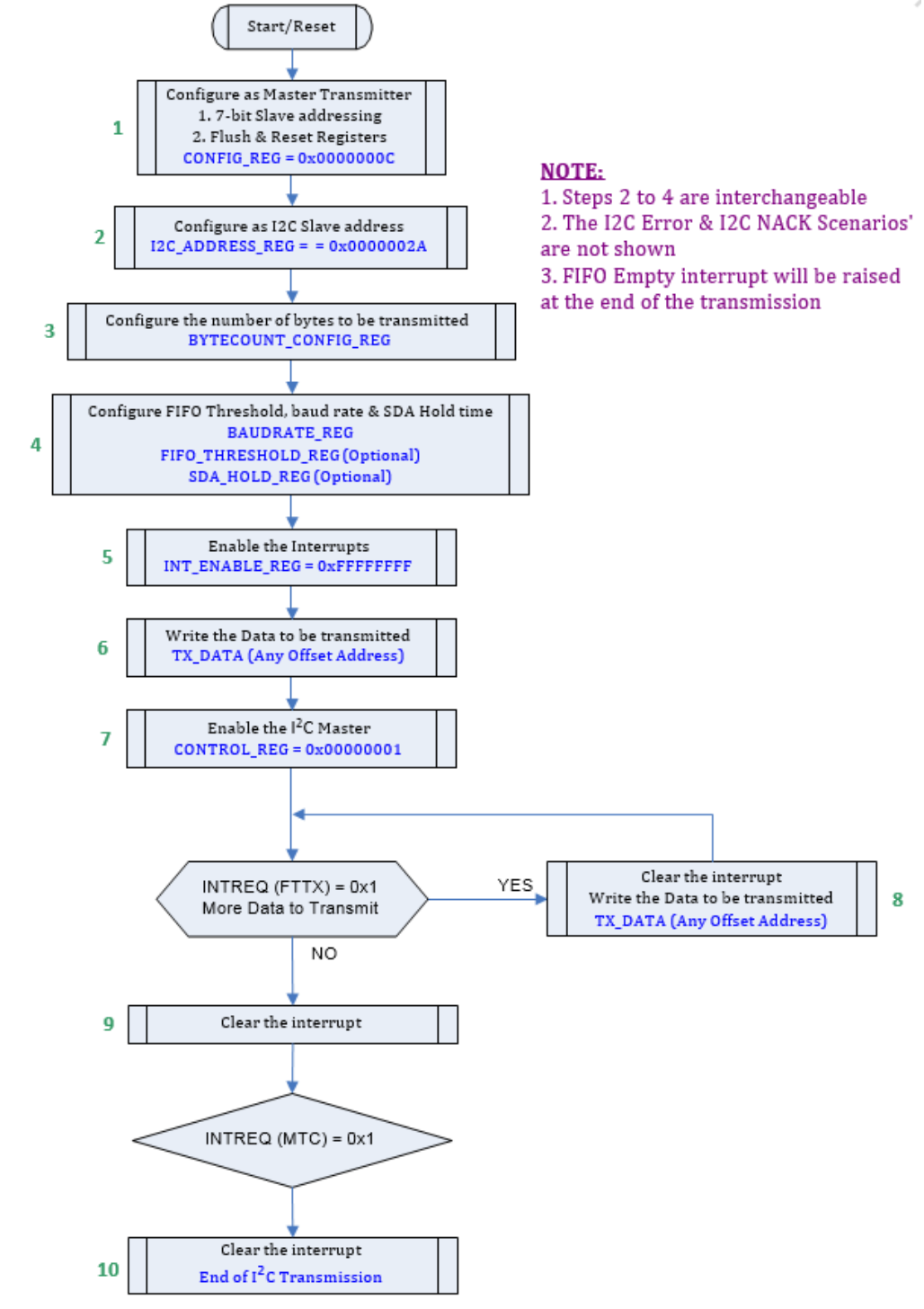



**Fig 41. TX programming Flowchart**

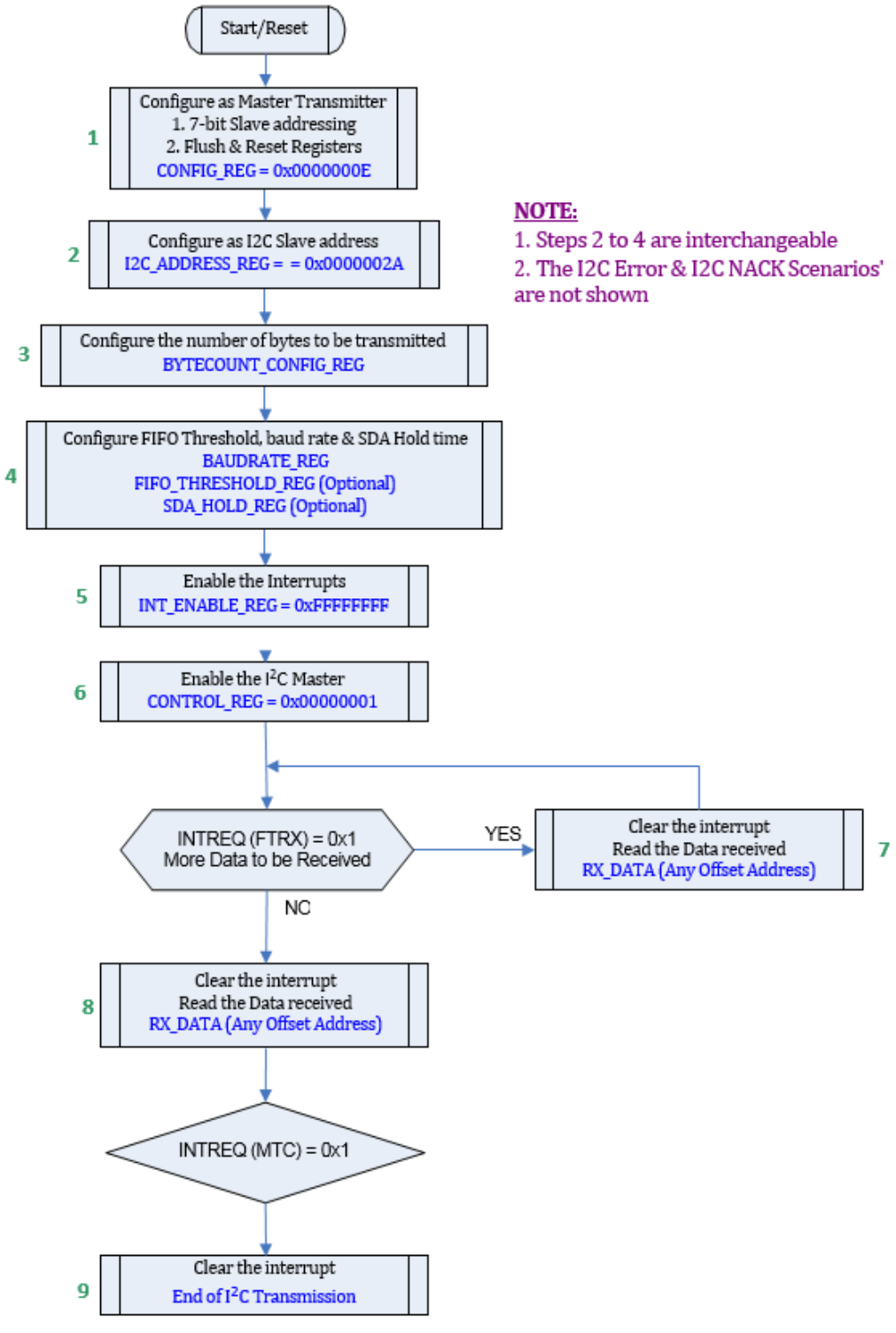



**Fig 42. RX programming Flowchart**

### 14.1.5 Baud rate calculation

The SCL (serial clock line) frequency calculation is based on 27.12 MHz clock. The BAUDRATE_REG register (see Table 277) controls the speed of the I2C bus transmissions. The SCL (serial clock) line frequency is determined as follows.

$$SCL\ Frequency = \frac{27.12}{27 + BAUDRATE}\ MHz$$

The baud rate parameter can be calculated using the following equation.

Examples:

If BAUDRATE_REG = 0x00000000 then SCL frequency = 1MHz (1 Mbit/s)

If BAUDRATE_REG = 0x00000029 then SCL frequency = 400kHz (400 kbit/s)

If BAUDRATE_REG = 0x0000006D then SCL frequency = 100kHz to 100 kbit/s

### 14.1.6 SDA_Hold calculation

The I2C-Bus specification requires a hold time for the SDA signal of at least 300 ns to bridge the undefined region of the falling edge of SCL in Standard and Fast modes. This 300 ns hold time is computed based on the 27.12 MHz clock (27.12 MHz) and written to the SDA_HOLD_REG register (Table 278).

The SDA hold requirements in Fast-mode Plus show that this 300 ns window is no longer necessary to ensure a proper functionality of the controller. It is therefore recommended to set the SDA_HOLD to 0x00 when operating in Fast-mode Plus, in order for the data to change as early as possible after the falling edge of the SCL line.

Example:

If the SCL frequency is 339; then the BAUDRATE register bit field is 0x35.

So, the SDA_HOLD register bit field is would be:

Here, SDA_HOLD can be any integer value between 9 & 19.

**Note**:

1. Setting the Value (Non-zero) for the SDA_HOLD register bit field is not mandatory. If the SDA_HOLD register bit field is 0x0, then the I2C Master IP will internal take care of 300 ns hold time for I2C standard and fast modes
2. The SDA_HOLD register bit field is applicable only for I2C standard and fast modes.

### 14.1.7 External wiring

The I2C controller operates on an external clock. For proper operation, external blocks must supply a clock signal to the apb_clk pin. Furthermore, as can be seen in Fig 43, the analog characteristics of the I2C bus (data and clock are connected to pull-up resistors) cause that the SCL and SDA lines must be driven by a pull-down circuit also called “I2C”. The signals scl_out and sda_out are then used to enable the pull-down circuit (when ‘0’) or disable it when ‘1’.

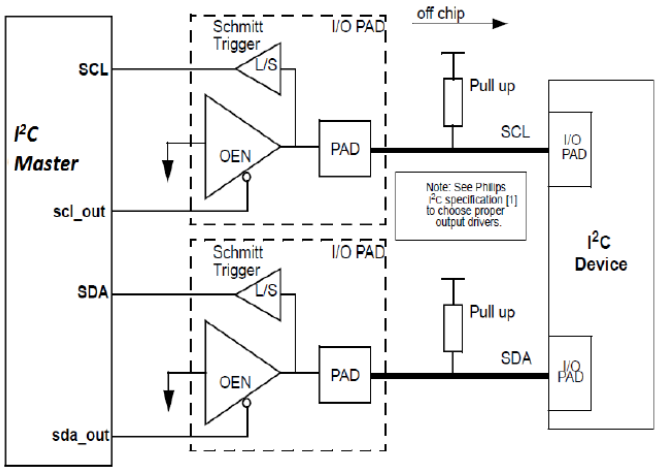



**Fig 43. I2C external wiring diagram**

### 14.1.8 I2C Register overview

**Table 275. I2CM Register overview (base address 0x4003 0000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| CONFIG_REG [1] | 0x0000 | 32 | RW | 0x00000000 | Register fields to configure the I2C Master & I2C (Transmission/Reception) mode of operation. |
| BAUDRATE_REG [1] | 0x0004 | 32 | RW | 0x000000F4 | Register field to generate the I2C Serial Clock Frequency. Default: 1 MHz Serial Clock Frequency at input system/IP Clock Frequency of 27 MHz |
| SDA_HOLD_REG [1] | 0x0008 | 32 | RW | 0x00000009 | Used to set the SDA hold time (SDA generation with respect to falling edge on SCL) - Must be set to 0x00 when operating in Fast-mode Plus (freq SCL> 400 kHz) |
| I2C_ADDRESS_REG [1] | 0x000C | 32 | RW | 0x0000002A | Contains the I2C Slave address of the device. |
| FIFO_THRESHOLD_REG | 0x0010 | 32 | RW | 0x00000701 | Register field to set the FIFO Threshold Level for the Interrupt Request Generation |
| BYTECOUNT_CONFIG_REG | 0x0014 | 32 | RW | 0x00000000 | Register bit field to configure the number of bytes to be transmitted or received |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| BYTECOUNT_STATUS_REG | 0x0018 | 32 | R | 0x00000000 | Register bit field to indicate the status of number of bytes transmitted or received |
| STATUS_REG | 0x001C | 32 | R | 0x0000002C | Register bit fields to indicate the current status of the I2C master |
| CONTROL_REG [2] | 0x0020 | 32 | W | 0x00000000 | Register bit field to enable the I2C Transmission or reception. |
| INTERNAL_USE [3] | 0x0024 | 32 | RW | 0x00000000 | For internal use |
| INTERNAL_USE [3] | 0x0028 | 32 | R | 0x00000000 | For internal use |
| TX_DATA | 0x0040 | 32 | W | 0x00000000 | These register fields are used to fill the FIFO with the data to be transmitted & accessible as write only registers; if I2C Master is configured for I2C Transmission. |
| | 0x0044 | 32 | W | 0x00000000 | |
| | 0x0048 | 32 | W | 0x00000000 | |
| | 0x004C | 32 | W | 0x00000000 | |
| RX_DATA | 0x0050 | 32 | R | 0x00000000 | These register fields are used to read the received data from FIFO & accessible as read only registers; if I2C Master is configured for I2C Reception. |
| | 0x0054 | 32 | R | 0x00000000 | |
| | 0x0058 | 32 | R | 0x00000000 | |
| | 0x005C | 32 | R | 0x00000000 | |
| INT_CLR_ENABLE_REG | 0x3FD8 | 32 | W | 0x00000000 | Collection of Clear Interrupt Enable commands |
| INT_SET_ENABLE_REG | 0x3FDC | 32 | W | 0x00000000 | Collection of Set Interrupt Enable commands |
| INT_STATUS_REG | 0x3FE0 | 32 | R | 0x00000000 | Collection of Interrupt Status |
| INT_ENABLE_REG | 0x3FE4 | 32 | R | 0x00000000 | Collection of Interrupt Enable |
| INT_CLR_STATUS_REG | 0x3FE8 | 32 | W | 0x00000000 | Collection of Clear Interrupt Status commands |
| INT_SET_STATUS_REG | 0x3FEC | 32 | W | 0x00000000 | Collection of Set Interrupt Status commands |

[1] Read-Only Register Access during I2C Transmission/Reception is on-going.

[2] Write Set Only Register Access: Once the bit is set, the I2C transmission or reception starts and once the I2C transmission or reception is completed the bit field is automatically cleared. Writing 1 to bit field during an active I2C transmission or reception has no effect.

[3] Bit-field are either set by HAL or use default value from CLIF EEPROM default settings

### 14.1.9 Register description

#### 14.1.9.1 CONFIG_REG

This register is used to configure the I2C master.

**Table 276. CONFIG_REG (address offset 0x0000)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R | 0x0000000* | reserved |
| 4 | RESET_I2C_CORE | -X | 0x0* | Set Only (Pulse) Register Bit Field to Reset the $I^2C$ Core Block. |
| 3 | RESET_REG | -X | 0x0* | Set Only (Pulse) Register Bit Field to Reset Selected Registers of $I^2C$ Master |
| 2 | FIFO_FLUSH | -X | 0x0* | Set Only (Pulse) Register Bit Field to flush the FIFO irrespective of $I^2C$ Master Mode of operation |
| 1 | SLV_ADDRESSING | R/W | 0x0* | $I^2C$ Slave 7/10 bit address selection |
| 0 | IP_MODE | R/W | 0x0* | $I^2C$ Master Mode selection |

**IP_MODE**

IP_MODE bit field determines the I2C master mode of operation.

- IP_MODE = 0x0: I2Cmaster as transmitter
- IP_MODE = 0x1: I2C master as receiver

**SLV_ADDRESSING**

SLV_ADDRESSING bit field determines I2C master slave addressing.

- SLV_ADDRESSING = 0x0: 7-bit I2C slave addressing
- SLV_ADDRESSING = 0x1: 10-bit I2C slave addressing

**FIFO_FLUSH**

Writing 1 to the FIFO_FLUSH bit field will generate a pulse to flush the content of the FIFO irrespective of I2C master mode of operation.

**RESET_REG**

Writing 1 to the RESET_REG bit field will generate a pulse to reset the following registers to their default value.

1. BAUDRATE_REG
2. SDAHOLD_REG
3. I2C_ADDRESS_REG
4. FIFO_THRESHOLD_REG
5. BYTECOUNT_CONFIG_REG

**RESET_I2C_CORE**

Writing 1 to the RESET_I2C_CORE bit field will generate a pulse to reset the I2C Core block/state-machine.

#### 14.1.9.2 BAUDRATE_REG

This register is used to generate the I2C serial clock frequency. The default is 1MHz serial clock frequency at input system clock Frequency of 27 MHz

**Table 277. BAUDRATE_REG (address offset 0x0004)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | R | 0x0000000* | reserved |
| 7:0 | BAUDRATE | R/W | 0xF4* | BAUDRATE bit field is used for the I2C Serial Clock Frequency calculation. For a detailed calculation, refer to Section 14.1.5 |

#### 14.1.9.3 SDA_HOLD_REG

This register is used to set the SDA hold time (SDA generation with respect to the falling edge on SCL). It must be set to 0x00 when operating in Fast-mode Plus (freq SCL> 400 kHz).

**Table 278. SDA_HOLD_REG (address offset 0x0008)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R | 0x0000000* | reserved |
| 4:0 | SDA_HOLD | R/W | 0x09* | SDA Hold time; For a detailed calculation refer to Section 14.1.6 |

#### 14.1.9.4 I2C_ADDRESS_REG

This register contains the I2C slave address of the device.

**Table 279. I2C_ADDRESS_REG (address offset 0x000C)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0x0000000* | reserved |
| 9:0 | SLAVE_ADDRESS | R/W | 0x2A* | $I^2C$ Slave 7 bit or 10-bit address |

#### 14.1.9.5 FIFO_THRESHOLD_REG

This register is used to set the FIFO threshold level for interrupt request generation.

**Table 280. FIFO_THRESHOLD_REG (address offset 0x0010)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:11 | RESERVED | R | 0x0000000* | Reserved |
| 10:8 | RXMODE_THRESHOLD | R/W | 0x7* | FIFO Threshold level for Interrupt Request generation when $I^2C$ Master is configured for $I^2C$ Reception mode. |
| 7:3 | RESERVED | R | 0x0000000* | |
| 2:0 | TXMODE_THRESHOLD | R/W | 0x1* | FIFO Threshold level for Interrupt Request generation when $I^2C$ Master is configured for $I^2C$ Transmission mode. |

**TXMODE_THRESHOLD**
FIFO Threshold level for interrupt request generation when I2C master is configured for transmission mode.

1. TXMODE_THRESHOLD register field is valid only during I2C transmission.
2. Interrupt request is generated when the TXMODE_THRESHOLD equals the I2C master internal FIFO.
3. Byte granularity will be interpreted by the BYTECOUNT_CONFIG_REG.

**RXMODE_THRESHOLD**
FIFO threshold level for interrupt request generation when I2C master is configured for reception mode.

1. RXMODE_THRESHOLD register field is valid only during I2C transmission.
2. Interrupt request is generated when the RXMODE_THRESHOLD equals the I2C master internal FIFO.
3. Byte granularity must be interpreted by the BYTECOUNT_CONFIG_REG.

#### 14.1.9.6 BYTECOUNT_CONFIG_REG

This register's function is to configure the number of bytes to be transmitted or received.

**Table 281. BYTECOUNT_CONFIG_REG (address offset 0x0014)**
*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0x0000000* | reserved |
| 9:0 | BYTE_COUNT_CONFIG | R/W | 0x000* | This register bit field is used to configure the number of byte to be Transmitted or Received. A maximum of 1023 byte can be transmitted or received in a frame. |

#### 14.1.9.7 BYTECOUNT_STATUS_REG

This register reflects the status of number of bytes being transmitted or received.

**Table 282. BYTECOUNT_STATUS_REG (address offset 0x0018)**
*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0x0000000* | reserved |
| 9:0 | BYTE_COUNT_CONFIG | R | 0x000* | This register bit field is used to provide the status of number of byte currently Transmitted or Received. A maximum of 1023 byte can be transmitted or received in a frame. |

#### 14.1.9.8 STATUS_REG

This register indicates the current status of the I2C master.

**Table 283. STATUS_REG (address offset 0x001C)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0000000* | reserved |
| 11:8 | FIFO_LEVEL | R | 0x0* | Indicates the current FIFO level irrespective of $I^2C$ Master Mode of operation |
| 7:6 | RESERVED | R | 0x0000000* | Reserved |
| 5 | FIFO_EMPTY_STATUS | R | 0x1* | Indicates the FIFO empty condition irrespective of $I^2C$ Master Mode of operation |
| 4 | FIFO_FULL_STATUS | R | 0x0* | Indicates the FIFO full condition irrespective of $I^2C$ Master Mode of operation |
| 3 | SCL | R | U* | Current Status of scl_a line |
| 2 | SDA | R | U* | Current Status of sda_a line |
| 1 | I2C_MASTER_MODE | R | 0x0* | $I^2C$ Mode of operation. Status from the IP_MODE bit field of CONFIG_REG register |
| 0 | I2C_BUS_ACTIVE | R | 0x0* | Indicates $I^2C$ Transmission or Reception is On-going |

**I2C_BUS_ACTIVE**

The I2C_BUS_ACTIVE bit field is set when I2C transmission or reception is enabled (I2C_ENABLE bit field in the CONTROL_REG register). The I2C_BUS_ACTIVE bit field is cleared when 2C stop condition is detected.

**I2C_MASTER_MODE**

The I2C_MASTER_MODE bit field is set when I2C master is configured for I2C reception. The I2C_MASTER_MODE bit field is cleared when I2C master is configured for I2C transmission.

**SDA**

The SDA bit field indicates the current status of sda_a line after 2 flops synchronization.

**SCL**

The SCL bit field indicates the current status of scl_a line after 2 flops synchronization.

#### 14.1.9.9 CONTROL_REG

This register is used to enable the I2C transmission or reception.

**Table 284. CONTROL_REG (address offset 0x0020)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0x0000000* | reserved |
| 9:0 | BYTE_COUNT_CONFIG | R | 0x000* | This register bit field is used to provide the status of number of byte currently Transmitted or Received. A maximum of 1023 byte can be transmitted or received in a frame. |

#### 14.1.9.10 TX_DATA

This register is used to fill the FIFO with the data to be transmitted & accessible as write only registers, if the I2C Master is configured for transmission.

**Table 285. TX_DATA (address offset 0x0040 to 0x004C)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | TX_DATA | W | 0x0000000* | Data to be transmitted |

1. The TX_DATA register fields are accessible as write-only during I2C master configured as I2C transmitter. (IP_MODE = 0x0 in CONFIG_REG)

2. The Data written to TX_DATA register fields are considered to be WORD size, however the byte granularity will be interpreted by the BYTECOUNT_CONFIG_REG register.

3. The TX_DATA from offset address 0x0040 to 0x004C, is order independent. The TX data can be written to any of these offset address 0x0040 or 0x0044 or 0x0048 or 0x004C in any order.

#### 14.1.9.11 RX_DATA

This register is used to read the received data from FIFO & accessible as read only registers if the I2C Master is configured for reception.

**Table 286. RX_DATA (address offset 0x0050 to 0x005C)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:0 | RX_DATA | R | 0x0000000* | Data received |

1. The RX_DATA register fields are accessible as read-only during I2C master configured as I2C receiver. (IP_MODE = 0x1 in CONFIG_REG)
2. The Data from RX_DATA register fields are considered to be WORD size, however the byte granularity will be interpreted by the BYTECOUNT_CONFIG_REG register.
3. The RX_DATA from offset address 0x0050 to 0x005C, is order independent. The RX data can be read in any of these offset address 0x0050 or 0x0054 or 0x0058 or 0x005C in any order.

#### 14.1.9.12 INT_CLR_ENABLE_REG

This register is a collection of Clear Interrupt Enable commands. Writing 1 to this register does clear the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 287. INT_CLR_ENABLE_REG (address offset 0x3FD8)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0* | reserved |
| 11 | CLR_ENABLE_TX_FIFO_THRES | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 10 | CLR_ENABLE_RX_FIFO_THRES | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 9 | CLR_ENABLE_FIFO_EMPTY | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 8 | CLR_ENABLE_FIFO_FULL | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 7:4 | RESERVED | R | 0x0* | reserved |
| 3 | CLR_ENABLE_I2C_BUS_ERROR | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 2 | CLR_ENABLE_NACK | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 1 | CLR_ENABLE_ARB_FAILURE | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |
| 0 | CLR_ENABLE_TRN_COMPLETED | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ ENABLE flag |

#### 14.1.9.13 INT_SET_ENABLE_REG

This register is a collection of Set Interrupt Enable commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 288. INT_SET_ENABLE_REG (address offset 0x3FDC)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0* | reserved |
| 11 | SET_ENABLE_TX_FIFO_THRES | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 10 | SET_ENABLE_RX_FIFO_THRES | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 9 | SET_ENABLE_FIFO_EMPTY | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 8 | SET_ENABLE_FIFO_FULL | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 7:4 | RESERVED | R | 0x0* | reserved |
| 3 | SET_ENABLE_I2C_BUS_ERROR | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 2 | SET_ENABLE_NACK | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 1 | SET_ENABLE_ARB_FAILURE | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |
| 0 | SET_ENABLE_TRN_COMPLETED | W | 0x0* | Writing 1 to this register does set the corresponding IRQ ENABLE flag |

#### 14.1.9.14 INT_STATUS_REG

This register is a collection of Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 289. INT_STATUS_REG (address offset 0x3FE0)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0* | reserved |
| 11 | TX_FIFO_THRES | R | 0x0* | Indicates that the FIFO threshold is reached while I2C transmission is on-going |
| 10 | RX_FIFO_THRES | R | 0x0* | Indicates that the FIFO threshold is reached while I2C reception is on-going |
| 9 | FIFO_EMPTY | R | 0x0* | Indicates that the FIFO empty condition is reached while I2C transmission is on-going |
| 8 | FIFO_FULL | R | 0x0* | Indicates that the FIFO full condition is reached while I2C reception is on-going |
| 7:4 | RESERVED | R | 0x0* | reserved |
| 3 | I2C_BUS_ERROR | R | 0x0* | Indicates an I2C bus error occurred |
| 2 | NACK | R | 0x0* | Indicates an I2C slave didn't acknowledge the I2C master request |
| 1 | ARB_FAILURE | R | 0x0* | Indicates an I2C master arbitration failure |
| 0 | TRN_COMPLETED | R | 0x0* | Indicates an I2C master completed the I2C transmission or I2C reception |

#### 14.1.9.15 INT_ENABLE_REG

This register is a collection of Interrupt Enable commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 290. INT_ENABLE_REG (address offset 0x3FE4)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0* | reserved |
| 11 | ENABLE_TX_FIFO_THRES | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 10 | ENABLE_RX_FIFO_THRES | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 9 | ENABLE_FIFO_EMPTY | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 8 | ENABLE_FIFO_FULL | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 7:4 | RESERVED | R | 0x0* | reserved |
| 3 | ENABLE_I2C_BUS_ERROR | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 2 | ENABLE_NACK | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 1 | ENABLE_ARB_FAILURE | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |
| 0 | ENABLE_TRN_COMPLETED | R | 0x0* | If this bit is 1 the corresponding IRQ can propagate to the CPUs IRQ controller |

#### 14.1.9.16 INT_CLR_STATUS_REG

This register is a collection of Clear Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 291. INT_CLR_STATUS_REG (address offset 0x3FE8)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0* | reserved |
| 11 | CLR_STATUS_TX_FIFO_THRES | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 10 | CLR_STATUS_RX_FIFO_THRES | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 9 | CLR_STATUS_FIFO_EMPTY | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 8 | CLR_STATUS_FIFO_FULL | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 7:4 | RESERVED | R | 0x0* | reserved |
| 3 | CLR_STATUS_I2C_BUS_ERROR | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 2 | CLR_STATUS_NACK | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 1 | CLR_STATUS_ARB_FAILURE | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |
| 0 | CLR_STATUS_TRN_COMPLETED | W | 0x0* | Writing 1 to this register does clear the corresponding IRQ STATUS flag |

#### 14.1.9.17 INT_SET_STATUS_REG

This register is a collection of Set Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 292. INT_SET_STATUS_REG (address offset 0x3FEC)**

*Legend: * reset value; <= mandatory value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:12 | RESERVED | R | 0x0* | reserved |
| 11 | SET_STATUS_TX_FIFO_THRES | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 10 | SET_STATUS_RX_FIFO_THRES | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 9 | SET_STATUS_FIFO_EMPTY | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 8 | SET_STATUS_FIFO_FULL | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 7:4 | RESERVED | R | 0x0* | reserved |
| 3 | SET_STATUS_I2C_BUS_ERROR | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 2 | SET_STATUS_NACK | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 1 | SET_STATUS_ARB_FAILURE | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |
| 0 | SET_STATUS_TRN_COMPLETED | W | 0x0* | Writing 1 to this register does set the corresponding IRQ STATUS flag |

### 14.1.10 Using example

#### 14.1.10.1 I2C master transmitter example

Example with:

1. Slave address = 0x2A
2. 9 bytes to be transmitted
3. I2C clock frequency = 1 MHz

**Step 1**: Configure the I2C Master in Master Transmitter mode with an APB Write Transaction using write data 0x0000000C at offset address 0x00000 {CONFIG_REG}.

**Step 2**: Configure the I2C slave address with an APB Write Transaction using write data 0x0000002A at offset address 0x0000C {I2C_ADDRESS_REG}.

**Step 3**: Configure the I2C clock frequency with an APB Write Transaction using write data 0x00000000 at offset address 0x00004 {BAUDRATE_REG}.

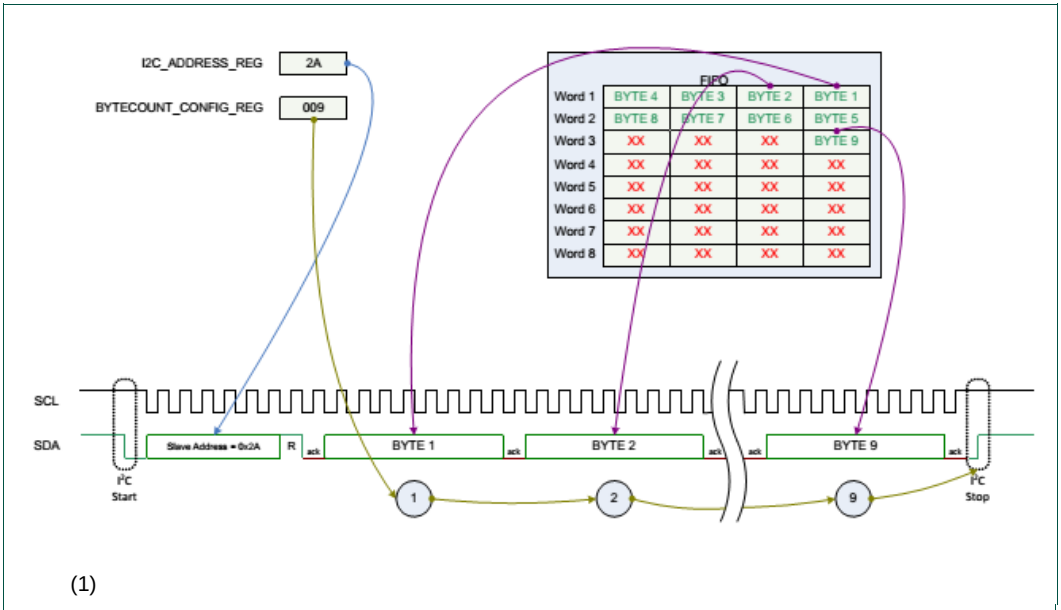
**Step 4**: Configure the byte count with an APB Write Transaction using write data 0x00000009 at offset address 0x00014 {BYTECOUNT_CONFIG_REG}.

**Step 5**: Configure the I2C data to be transmitted with three APB Write Transaction at offset address 0x00040 - 0x0004C {TX_DATA}.

1. First APB Write Transaction using write data = {BYTE4, BYTE3, BYTE2, BYTE1}
2. Second APB Write Transaction using write data = {BYTE8, BYTE7, BYTE6, BYTE5}
3. Third APB Write Transaction using write data = {XX, XX, XX, BYTE9}

**Step 6**: Enable the I2C Master transmission with an APB Write Transaction using write data 0x00000001 at offset address 0x00020 {CONTROL_REG}.

**Note**: In the Example, the interrupts are not considered & targeted for the explanation of data integrity & programming procedure for I2C transmission.




(1)

**Fig 44. I2C master transmitter example**

### 14.1.10.2 I2C master receiver example

Example with:

1. Slave address = 0x2A
2. 7 bytes to be received
3. I2C clock frequency = 1 MHz

**Step 1**: Configure the I2C Master in Master Receiver mode with an APB Write Transaction using write data 0x0000000D at offset address 0x00000 {CONFIG_REG}.

**Step 2**: Configure the I2C slave address with an APB Write Transaction using write data 0x0000002A at offset address 0x0000C {I2C_ADDRESS_REG}.

**Step 3**: Configure the I2C clock frequency with an APB Write Transaction using write data 0x00000000 at offset address 0x00004 {BAUDRATE_REG}.

**Step 4**: Configure the byte count with an APB Write Transaction using write data 0x00000007 at offset address 0x00014 {BYTECOUNT_CONFIG_REG}.

**Step 5**: Enable the I2C Master reception with an APB Write Transaction using write data 0x00000001 at offset address 0x00020 {CONTROL_REG} & wait until the I2C reception is completed.

**Step 6**: Read the I2C data received with an two APB Read Transaction at offset address 0x00050 - 0x0005C {RX_DATA}.

1. First APB Read Transaction using read data = {BYTE4, BYTE3, BYTE2, BYTE1}

2. Second APB Read Transaction using read data = {XX, BYTE7, BYTE6, BYTE5}

**Note**: In the Example, the interrupts are not considered & targeted for the explanation of data integrity & the programming procedure for the I2C reception.

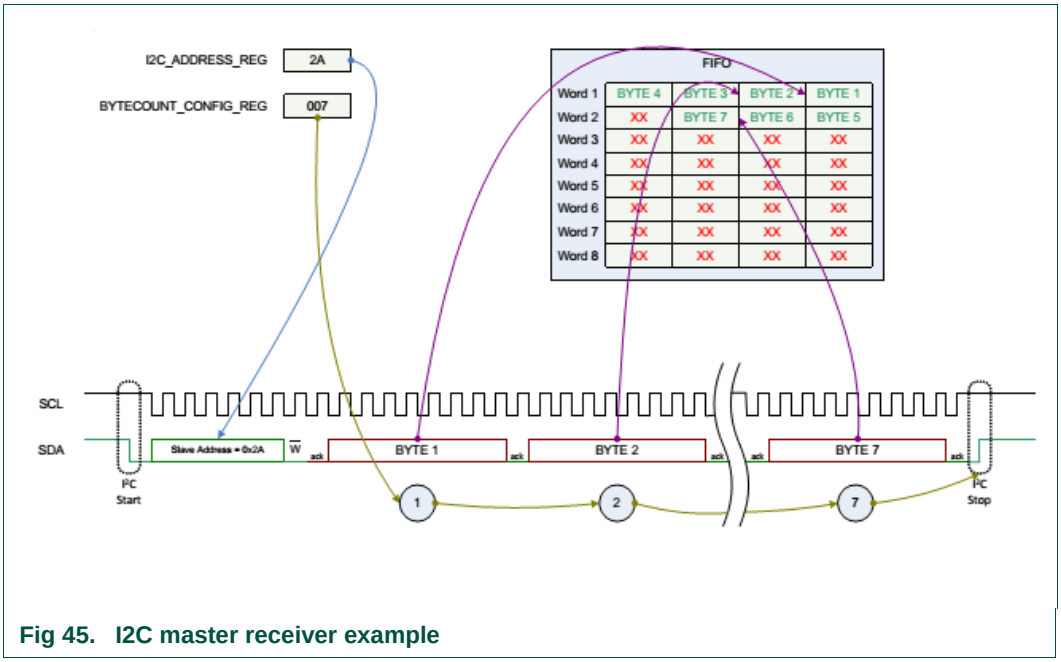



**Fig 45. I2C master receiver example**

## 14.2 SPI Master Interface

The SPI is a 4-wire serial interface designed to interface with a large range of serial peripheral or memory devices. The SPI master controller is one of two SPI controllers supported by PN7462AU. The SPI slave controller is described in Section 11.3.3.

### 14.2.1 SPI Master features

- Half-Duplex synchronous transfers
- Supports Motorola SPI frame formats only (SPI Block Guide V04.0114 (Freescale) specification)
- Multiple data rates -1, 1.51, 2.09, 2.47, 3.01, 4.52, 5.42 and 6.78 Mbit/s
- Up to two Slave Select, with selectable polarity
- Programmable clock polarity and phase
- Supports 8-bit transfers only
- Maximum frame size: 511 data bytes payload + 1 CRC Byte
- AHB Master interface for data transfer
- Optional CRC calculation (1 byte) on all data of TX and RX buffer

### 14.2.2 General description

Fig 46 shows the connection SPI master with two SPI slaves. Only a single master and a single slave can communicate on the bus during a given data transfer. Data transfers are in principle full duplex, with frames of 4 bits to 16 bits of data flowing from the master to the slave and from the slave to the master.

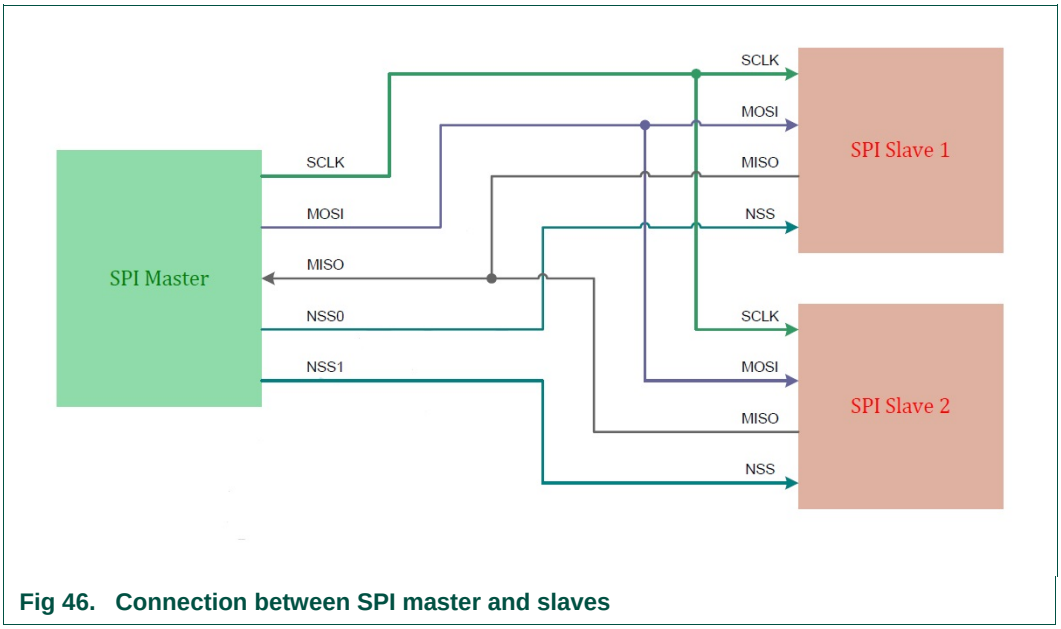



**Fig 46. Connection between SPI master and slaves**

### 14.2.3 Pin description

**Table 293. SPI pin description**

| Pin Name | Pin description |
|---|---|
| SCK0/1 | Serial Clock is clock signal, which is used to synchronize the transfer of data between master and slave. It can be programmed to be always active high or active low, and it will only switch during a data transfer. |
| SSEL0/1 | Slave Select is used to wake up the slave or in case of multiple slaves and select the slave to talk to |
| MISO0/1 | Master In Slave Out signal transfers serial data from the slave to the master and deals with different situation, depending on whether the SPI is a slave a master or not selected by the Slave Select. |
| MOSI0/1 | Master Out Slave in signal transfers serial data from the master to the slave. And handles different cases, where the SPI is either the master or the slave |

### 14.2.4 Configuring the SPI interface

The following parameters can be configured via the register SPIM_CONFIG_REG:

- NSS Polarity: defines the level of activity for NSS (0 by default for NSS active low)
- NSS Pulses: a 1 on this field (0 by default) means the SPI master will generate pulses between every byte.

These parameters are not allowed to be changed during an on-going SPI transmission.

Following figure shows the SPI protocol with CPHA = 0 & NSS_PULSE = 0:

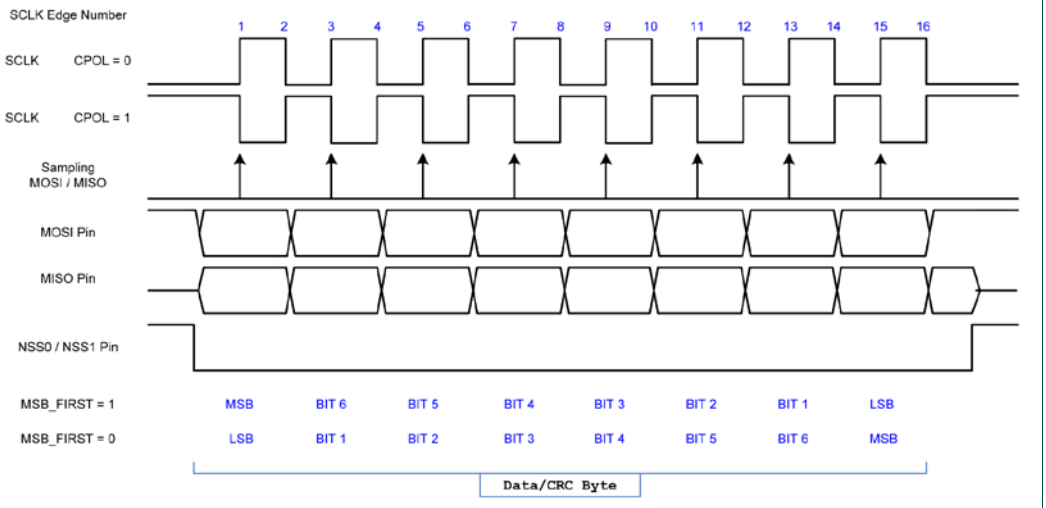



**Fig 47. SPI protocol with CPHA = 0 & NSS_PULSE = 0**

Following Figure shows the SPI protocol CPHA = 1& NSS_PULSE = 0

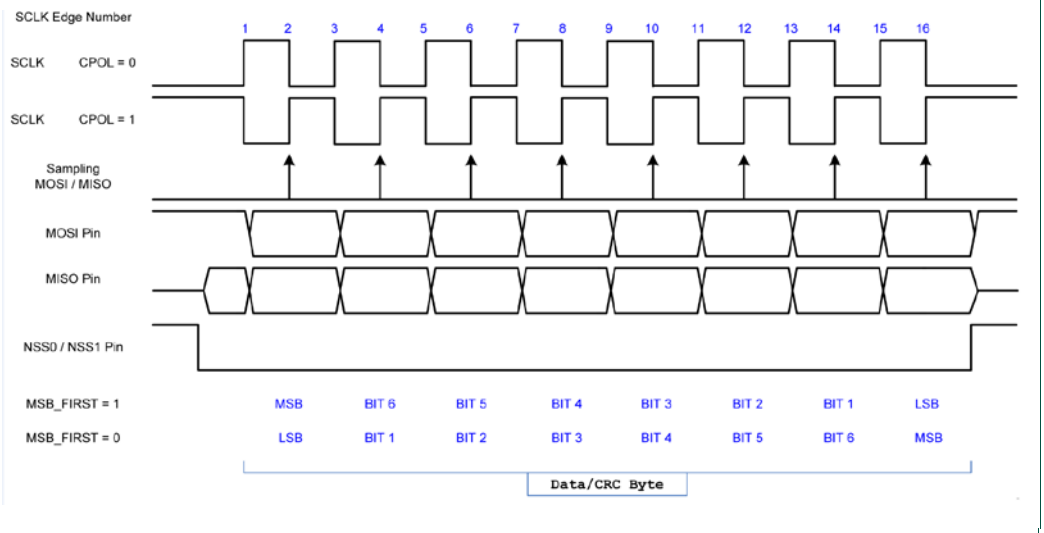



**Fig 48. SPI protocol with CPHA = 1 & NSS_PULSE = 0**

Following figure shows the SPI protocol CPHA = 0 & NSS_PULSE = 0.

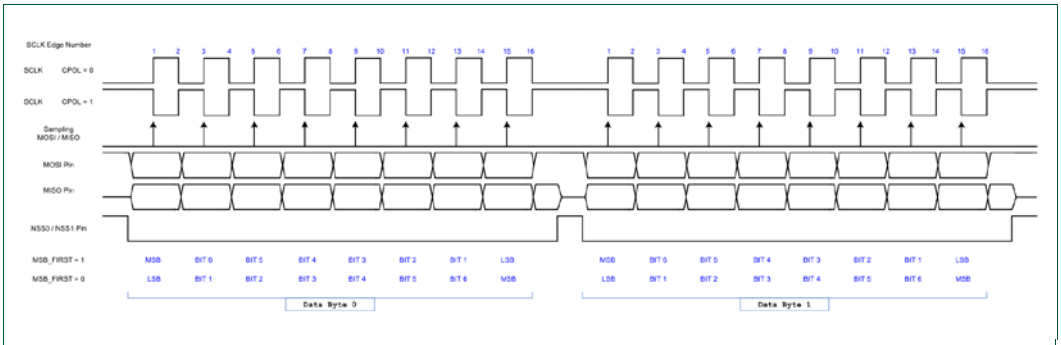


**Fig 49. SPI protocol with CPHA = 0 & NSS_PULSE = 1**

The SPI master automatically generates NSS signal. Its polarity can be configured, and it can be chosen to generate a pulse on NSS between each byte. If more flexibility is needed, then FW can program itself the value to output to NSS, using NSS_CTRL=1 (override internal value), and choosing the value to output to NSS_VAL.

### 14.2.5 Packet structure

The structure of the packet to be sent/received is described in Fig 50. The packet consists of:

- Payload: up to 511 bytes
- CRC: Redundancy code, computed on payload

All data bytes are transferred from memory to SPI or from SPI to memory. An exception is CRC, which in case of TX can be automatically computed and appended to the sent frame.

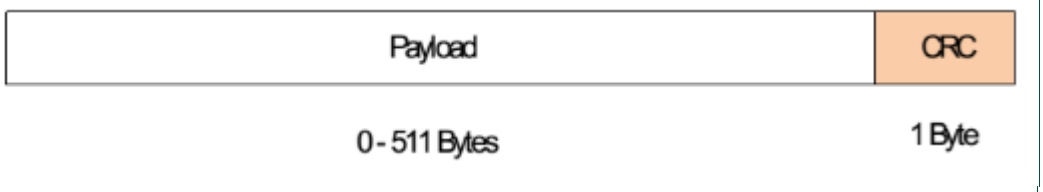



**Fig 50. SPI packed structure**

### 14.2.6 Buffer initialization

#### 14.2.6.1 RX buffers

Before reading data from the SPI slave, firmware should configure the receive buffer. This is done by using SPIM_BUFFER_MAPPING_REG, SPIM_RX_BUFFER_REG and SPIM_RX_BUFFER_CRC_REG registers. The SPIM_RX_BUFFER_REG specifies the buffer start address (inside System RAM address space), and length of the read access. The RX_LENGTH of 0 is not a valid buffer length. Special attention should be paid to the fact that the RX buffer should not exceed the maximum address size. RX_START_ADDR + RX_LENGTH must be kept in System RAM address range. The address is a byte address, which means that the data will be stored byte-aligned in the memory. Note that in case the address is not multiple of 4, all bytes of first word may be overwritten during data reception: content of first bytes (from word_start_addr to start_addr-1) is undefined. The same applies for the last byte written to memory: all bytes of the last word will be overwritten with undefined values. This is shown in Fig 51.

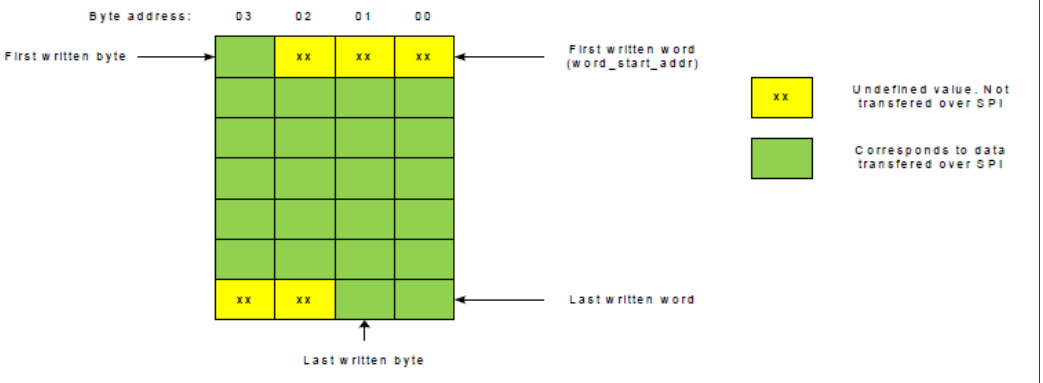



**Fig 51. RX data storage into memory. Defined/ undefined data**

Second register defines the reset value of the CRC (Should be programmed to 0xff), and the number of bytes to skip from the CRC computation from the received packets: default value is 0. When FW wants to reset CRC, then RX_SET_CRC bit of SPIM_CONTROL_REG will have to be written. This can be done during same write access as writing RX_START.

#### 14.2.6.2 TX buffers

Before writing data to the SPI Slave, firmware should configure the transmit buffer. This is done by using SPIM_TX_BUFFER_REG and SPIM_TX_BUFFER_CRC_REG registers. Compared to the RX buffer, there is one more bit in SPIM_TX_BUFFER_CRC_REG, called TX_APPEND_CRC. If this bit is set, then the SPI Master will send an extra byte (after sending TX_LENGTH bytes over SPI) with the automatically computed CRC value. The TX_LENGTH can be programmed to 0 if TX_APPEND_CRC is 1 (in that case, SPIM would transfer CRC over SPI lines).

### 14.2.7 CRC

The CRC is computed for all data of a RX/TX buffer. RX/TX_CRC_PAYLOAD_OFFSET allows to define a number of bytes to be skipped from the CRC computation. The initial value is 0xFF, and the CRC is a xor of all data bytes: CRC = 0xFF xor (Payload bytes)

#### 14.2.7.1 TX CRC automatically appended

For transmission, the CRC can be automatically computed and appended after payload (optional feature), when TX_APPEND_CRC is set. This is illustrated in Fig 52.

In that case, the number of sent bytes is TX_LENGTH + 1, and TX_LENGTH should be programmed to Payload_size.

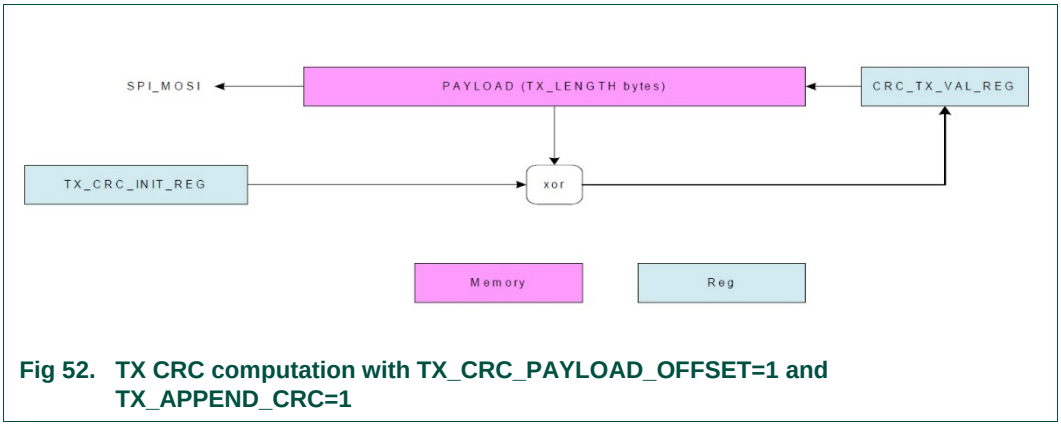



**Fig 52. TX CRC computation with TX_CRC_PAYLOAD_OFFSET=1 and TX_APPEND_CRC=1**

#### 14.2.7.2 TX CRC not appended

If for any reason, the CRC is already computed in memory (e.g. by firmware), then resetting TX_APPEND_CRC will prevent hardware from sending an extra CRC byte. This is shown in Fig 53. In that case, the number of sent bytes is TX_LENGTH, and TX_LENGTH should be programmed to Payload_size + 1 (CRC). Note that in that case, if CRC was correctly computed by firmware, CRC_TX_VAL_REG should be equal to 0, as result of CRC xor CRC.

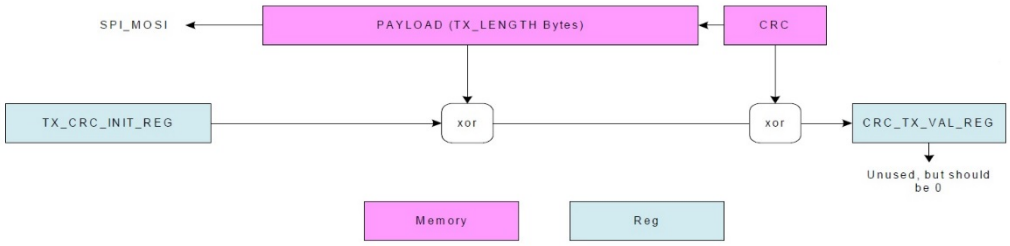



**Fig 53. TX CRC computation with TX_CRC_PAYLOAD_OFFSET=1 and TX_APPEND_CRC=0**

#### 14.2.7.3 RX CRC

For reception, the CRC is considered as data, and is stored in the memory. Number of received data is RX_LENGTH, where RX_LENGTH is programmed to Payload_size + 1. As a result of the reception, the firmware can read back the received CRC byte in memory (last received byte), and can check in CRC register that CRC_RX_VAL is 0 (CRC xor CRC). This is shown on figure Fig 54.

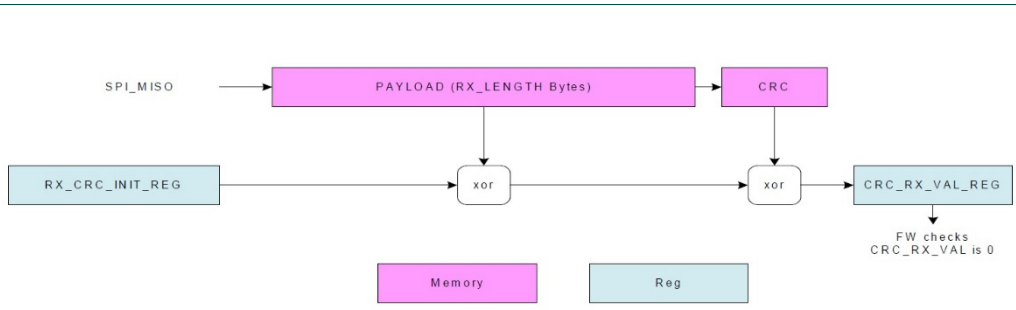



**Fig 54. RX CRC computation with RX_CRC_PAYLOAD_OFFSET=1**

### 14.2.8 SPI Register overview

**Table 294. SPI master register overview (base address 0x4003 4000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| SPIM_STATUS_REG | 0000h | 32 | R | 00000000h | Status |
| SPIM_CONFIG_REG | 0004h | 32 | R/W | 00000002h | Configuration of SPI Master |
| SPIM_CONTROL_REG | 0008h | 32 | W | 00000000h | RX/TX control |
| SPIM_RX_BUFFER_REG | 000Ch | 32 | R/W | 00000000h | Configuration of RX buffer |
| SPIM_RX_BUFFER_CRC_REG | 0010h | 32 | R/W | 000001FFh | Configuration of RX CRC |
| SPIM_TX_BUFFER_REG | 0014h | 32 | R/W | 00000000h | Configuration of TX buffer |
| SPIM_TX_BUFFER_CRC_REG | 0018h | 32 | R/W | 000201FFh | Configuration of TX CRC |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| SPIM_CRC_STATUS_REG | 001Ch | 32 | R | 0000FFFFh | RX/TX CRC values |
| SPIM_WATERLEVEL_REG | 0020h | 32 | R/W | 00000000h | Water level |
| INTERNAL_USE | 0024h | 32 | R/W | 00000000h | For internal use |
| INTERNAL_USE | 0028h | 32 | R | 00000000h | For internal use |
| SPIM_BUFFER_MAPPING_REG RAM | 002Ch | 32 | R/W | 00002FFFh | Buffer Mapping |
| SPIM_INT_CLR_ENABLE_REG | 3FD8h | 32 | W | 00000000h | Clear interrupt enable |
| SPIM_INT_SET_ENABLE_REG | 3FDCh | 32 | W | 00000000h | Set interrupt enable |
| SPIM_INT_STATUS_REG | 3FE0h | 32 | R | 00000000h | Interrupt status |
| SPIM_INT_ENABLE_REG | 3FE4h | 32 | R | 00000000h | Interrupt enable |
| SPIM_INT_CLR_STATUS_REG | 3FE8h | 32 | W | 00000000h | Clear interrupt |
| SPIM_INT_SET_STATUS_REG | 3FECh | 32 | W | 00000000h | Set interrupt |

[1] The reserved address 0x30 is mapped to a spare register (8 bits). The value read from this location is 0x000000F0. Bits 7:4 are read-write; bits 3:0 are read-only.

### 14.2.9 Register description

#### 14.2.9.1 SPIM_STATUS_REG

This register reflects the current status of the SPI master.

*Table 295.* **SPIM_STATUS_REG (address offset 0000h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:2 | RESERVED | R | 0 | Reserved |
| 1 | TX_ONGOING | R | 0 | 1 - TX buffer is currently in use by the hardware. Any TX_START/RX_START command will be ignored while TX_ONGOING=1 |
| 0 | RX_ONGOING | R | 0 | 1 - RX buffer is currently in use by the hardware. Any TX_START/RX_START command will be ignored while RX_ONGOING=1 |

#### 14.2.9.2 SPIM_CONFIG_REG

This register is used to configure the SPI Master.

*Table 296.* **SPIM_CONFIG_REG (address offset 0004h) bit description**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:19 | RESERVED | R | 0 | Reserved |
| 18:16 | BAUDRATE[1] | R/W | 0 | SCK frequency (Baudrate) Selection |
| 15:9 | RESERVED | R | 0 | Reserved |
| 8 | SLAVE_SELECT[1] | R/W | 0 | 1: Slave 1 is selected (Pin: GPIO1)<br>0: Slave 0 is selected (Pin: SPIM_SSN). |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 7 | MSB_FIRST [1] | R/W | 0 | If set, MS bit of a byte is transmitted & received first.<br>If cleared, LS bit of a byte is transmitted & received first. |
| 6 | CPHA [1] | R/W | 0 | This bit is used to select the SPI clock format. A change of this bit will abort a transmission in progress and force the SPI system into idle state.<br>1 = Sampling of data occurs at even edges (2,4,6,...,16) of the SCK clock<br>0 = Sampling of data occurs at odd edges (1,3,5,...,15) of the SCK clock |
| 5 | CPOL [1] | R/W | 0 | This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. A change of this bit will abort a transmission in progress and force the SPI system into idle state.<br>1 = Active-low clocks selected. In idle state SCK is high.<br>0 = Active-high clocks selected. In idle state SCK is low. |
| 4 | NSS_VAL | R/W | 0 | Value to output to nss for the selected slave if NSS_CTRL=1 |
| 3 | NSS_CTRL | R/W | 0 | 1: Override NSS value with NSS_VAL |
| 2 | RESERVED | R | 0 | Reserved |
| 1 | NSS_PULSE [1] | R/W | 1 | 1: a pulse on NSS is generated between 2 bytes |
| 0 | NSS_POLARITY [1] | R/W | 0 | 0: NSS active low |

[1] The following Register bit fields are not updated during an on-going SPI communication.

- BAUDRATE
- SLAVE_SELECT
- MSB_FIRST
- CPHA
- CPOL
- NSS_PULSE
- NSS_POLARITY

#### 14.2.9.3 SPIM_CONTROL_REG

This register is used to control the SPI transmission or reception.

**Table 297. SPIM_CONTROL_REG (address offset 0x0008)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:4 | RESERVED | W | 0 | Reserved |
| 3 | TX_SET_CRC | W | 0 | Flag to init internal CRC to TX_CRC_INIT. Automatically returns to 0. |
| 2 | TX_START | W | 0 | 1: Start TX. Automatically returns to 0 |
| 1 | RX_SET_CRC | W | 0 | Flag to init internal CRC to RX_CRC_INIT. Automatically returns to 0 |
| 0 | RX_START | W | 0 | 1: Start RX. Automatically returns to 0 |

#### 14.2.9.4 SPIM_RX_BUFFER_REG

This register is used to configure the RX buffer.

**Table 298. SPIM_RX_BUFFER_REG (address offset 0x000C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:23 | RESERVED | R | 0 | Reserved |
| 22:14 | RX_LENGTH | R/W | 0 | Size of RX transfer.<br>RX_LENGTH = 0: RX transfer of size 0 is not allowed<br>RX_LENGTH = 1: RX transfer payload of size 1 byte<br>RX_LENGTH = 2: RX transfer payload of size 2 byte<br>RX_LENGTH = 511: RX transfer payload of size 511 byte |
| 13:0 | RX_START_ADDR | R/W | 0 | Byte start address of RX buffer: start_addr+RX_length must not exceed maximum address range |

#### 14.2.9.5 SPIM_RX_BUFFER_CRC_REG

This register is used to configure the RX CRC.

**Table 299. SPIM_RX_BUFFER_CRC_REG (address offset 0x0010)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:17 | RESERVED | R | 0 | Reserved |
| 16:8 | RX_CRC_PAYLOAD_OFFSET | R/W | 0x01 | Number of bytes to skip for CRC computation |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 7:0 | RX_CRC_INIT | R/W | 0xFF | RX CRC init value. Only used if RX_SET_CRC is set |

#### 14.2.9.6 SPIM_TX_BUFFER_REG

This register is used to configure the TX buffer.

**Table 300. SPIM_TX_BUFFER_REG (address offset 0x0014)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:23 | RESERVED | R | 0 | Reserved |
| 22:14 | TX_LENGTH | R | 0 | Size of TX transfer.<br>TX_LENGTH = 0: TX transfer of size 0 is allowed only if TX_APPEND_CRC is set<br>TX_LENGTH = 1: TX transfer payload of size 1 byte<br>TX_LENGTH = 2: TX transfer payload of size 2 byte<br>…<br>TX_LENGTH = 511: TX transfer payload of size 511 byte |
| 13:0 | TX_START_ADDR | R/W | 0 | Byte start address of TX buffer: start_addr+TX_length must not exceed maximum address range |

#### 14.2.9.7 SPIM_TX_BUFFER_CRC_REG

This register is used to configure TX CRC.

**Table 301. SPIM_TX_BUFFER_CRC_REG (address offset 0x0018)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:18 | RESERVED | R | 0 | Reserved |
| 17 | TX_APPEND_CRC | R/W | 1 | 1: CRC is sent after transmission of TX_LENGTH bytes. If TX_LENGTH is 0, then only CRC will be sent. |
| 16:8 | TX_CRC_PAYLOAD_OFFSET | R/W | 0x01 | Number of bytes to skip for CRC computation |
| 7:0 | TX_CRC_INIT | R/W | 0xFF | TX CRC init value. Only used if TX_SET_CRC is set |

#### 14.2.9.8 SPIM_CRC_STATUS_REG

This register reflects the RX/TX CRC values.

**Table 302. SPIM_CRC_STATUS_REG (address offset 0x001C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | R | 0 | Reserved |
| 15:8 | CRC_TX_VAL | R | 0xFF | Value of internal TX CRC |
| 7:0 | CRC_RX_VAL | R | 0xFF | Value of internal RX CRC |

#### 14.2.9.9 SPIM_WATERLEVEL_REG

This register is used to indicate the water level.

**Table 303. SPIM_WATERLEVEL_REG (address offset 0x0020)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:9 | RESERVED | R | 0 | Reserved |
| 8:0 | WATERLEVEL | R/W | 0 | Number of bytes received in incoming frame, or sent in out coming frame before triggering an interrupt.<br>If set to 0, this feature is disabled. |

#### 14.2.9.10 SPIM_BUFFER_MAPPING_REG

REGION_SIZE – Size of buffer region in system RAM. 0x1FFF (8K) is the current RAM size in PN7462U. Note that software need to initialize REGION_SIZE <= 0x1FFF always before using the Buffer.

**Table 304. SPIM_BUFFER_MAPPING_REG (address offset 0x002C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:30 | RESERVED | R | 0 | Reserved |
| 29:16 | REGION_START_ADDR | R/W | 0 | Start address of buffer region in system RAM.<br>Any AHB transaction address would be 0x00100000 (RAM address for PN7462AU) + REGION_START_ADDR.<br>Also REGION_START_ADDR must be less than or equal to REGION_SIZE. |
| 15:14 | RESERVED | R | 0 | Reserved |
| 13:0 | REGION_SIZE | R/W | 0x2FFF | Size of buffer region in system RAM 0x2FFF is the Current RAM Size in PN7462AU |

#### 14.2.9.11 SPIM_INT_CLR_ENABLE_REG

This register is a collection of Clear Interrupt Enable commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 305. SPIM_INT_CLR_ENABLE_REG (address offset 0x3FD8)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | W | 0 | Reserved |
| 9 | AHB_ADDR_ERROR_CLR_ENABLE | W | 0 | 1 - clear enable for AHB address overflow interrupt |
| 8 | AHB_ERROR_CLR_ENABLE | W | 0 | 1 - clear enable for AHB Slave error interrupt<br>0 - no effect |
| 7:3 | RESERVED | W | 0 | Reserved |
| 2 | WATERLEVEL_REACHED_CLR_ENABLE | W | 0 | 1 - clear enable for water level reached interrupt<br>0 - no effect |
| 1 | EOT_CLR_ENABLE | W | 0 | 1 - clear enable for EOT interrupt<br>0 – no effect |
| 0 | EOR_CLR_ENABLE | W | 0 | 1 - clear enable for EOR interrupt<br>0 - no effect |

#### 14.2.9.12 SPIM_INT_SET_ENABLE_REG

This register is a collection of Set Interrupt Enable commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 306. SPIM_INT_SET_ENABLE_REG (address offset 0x3FDC)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | W | 0 | Reserved |
| 9 | AHB_ADDR_ERROR_SET_ENABLE | W | 0 | 1 - set enable for AHB address overflow interrupt |
| 8 | AHB_ERROR_SET_ENABLE | W | 0 | 1 - set enable for AHB Slave error interrupt<br>0 - no effect |
| 7:3 | RESERVED | W | 0 | Reserved |
| 2 | WATERLEVEL_REACHED_SET_ENABLE | W | 0 | 1 - set enable for water level reached interrupt<br>0 - no effect |
| 1 | EOT_SET_ENABLE | W | 0 | 1 - set enable for EOT interrupt<br>0 – no effect |
| 0 | EOR_SET_ENABLE | W | 0 | 1 - set enable for EOR interrupt<br>0 - no effect |

#### 14.2.9.13 SPIM_INT_STATUS_REG

This register is a collection of Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 307. SPIM_INT_STATUS_REG (address offset 0x3FE0)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0 | Reserved |
| 9 | AHB_ADDR_ERROR_STATUS | R | 0 | AHB address overflow Error interrupt status |
| 8 | AHB_ERROR_STATUS | R | 0 | AHB Slave Error interrupt status |
| 7:3 | RESERVED | R | 0 | Reserved |
| 2 | WATERLEVEL_REACHED_STATUS | R | 0 | Water level reached interrupt status |
| 1 | EOT_STATUS | R | 0 | EOT interrupt status |
| 0 | EOR_STATUS | R | 0 | EOR interrupt status |

#### 14.2.9.14 SPIM_INT_ENABLE_REG

This register is a collection of Interrupt Enable commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 308. SPIM_INT_ENABLE_REG (address offset 0x3FE4)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | R | 0 | Reserved |
| 9 | AHB_ADDR_ERROR_ENABLE | R | 0 | AHB address overflow Error interrupt enable |
| 8 | AHB_ERROR_ENABLE | R | 0 | AHB Slave Error interrupt enable |
| 7:3 | RESERVED | R | 0 | Reserved |
| 2 | WATERLEVEL_REACHED_ENABLE | R | 0 | Water level reached interrupt enable |
| 1 | EOT_ENABLE | R | 0 | EOT interrupt enable |
| 0 | EOR_ENABLE | R | 0 | EOR interrupt enable |

#### 14.2.9.15 SPIM_INT_CLR_STATUS_REG

This register is a collection of Clear Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 309. SPIM_INT_CLR_STATUS_REG (address offset 0x3FE8)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | W | 0 | Reserved |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 9 | AHB_ADDR_ERROR_CLR_STATUS | W | 0 | 1 - clear AHB address overflow Error interrupt<br>0 – no effect |
| 8 | AHB_ERROR_CLR_STATUS | W | 0 | 1 - clear AHB Slave Error interrupt<br>0 – no effect |
| 7:3 | RESERVED | W | 0 | Reserved |
| 2 | WATERLEVEL_REACHED_CLR_STATUS | W | 0 | 1 - clear water level reached interrupt<br>0 – no effect |
| 1 | EOT_CLR_STATUS | W | 0 | 1 - clear EOT interrupt<br>0 – no effect |
| 0 | EOR_CLR_STATUS | W | 0 | 1 - clear EOR interrupt<br>0 – no effect |

#### 14.2.9.16 SPIM_INT_SET_STATUS_REG

This register is a collection of Set Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 310. SPIM_INT_SET_STATUS_REG (address offset 0x3FEC)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:10 | RESERVED | W | 0 | Reserved |
| 9 | AHB_ADDR_ERROR_SET_STATUS | W | 0 | 1 - set AHB address overflow Error interrupt<br>0 – no effect |
| 8 | AHB_ERROR_SET_STATUS | W | 0 | 1 - set AHB Slave Error interrupt<br>0 – no effect |
| 7:3 | RESERVED | W | 0 | Reserved |
| 2 | WATERLEVEL_REACHED_SET_STATUS | W | 0 | 1 - set water level reached interrupt<br>0 – no effect |
| 1 | EOT_SET_STATUS | W | 0 | 1 - set EOT interrupt<br>0 – no effect |
| 0 | EOR_SET_STATUS | W | 0 | 1 - set EOR interrupt<br>0 – no effect |

## 14.3 Host interfaces

The Host interface block comprises four sub-blocks shown in Fig 55 .These are:

1. I2C Slave
2. SPI Slave
3. HSU Slave
4. USB

The host interface block selects the interface according to the input port *HIF_SELECTION [3:1]* (all listed in Table 311) from the PCR, and connects this interface to the buffer manager. Only one interface is active at a time.

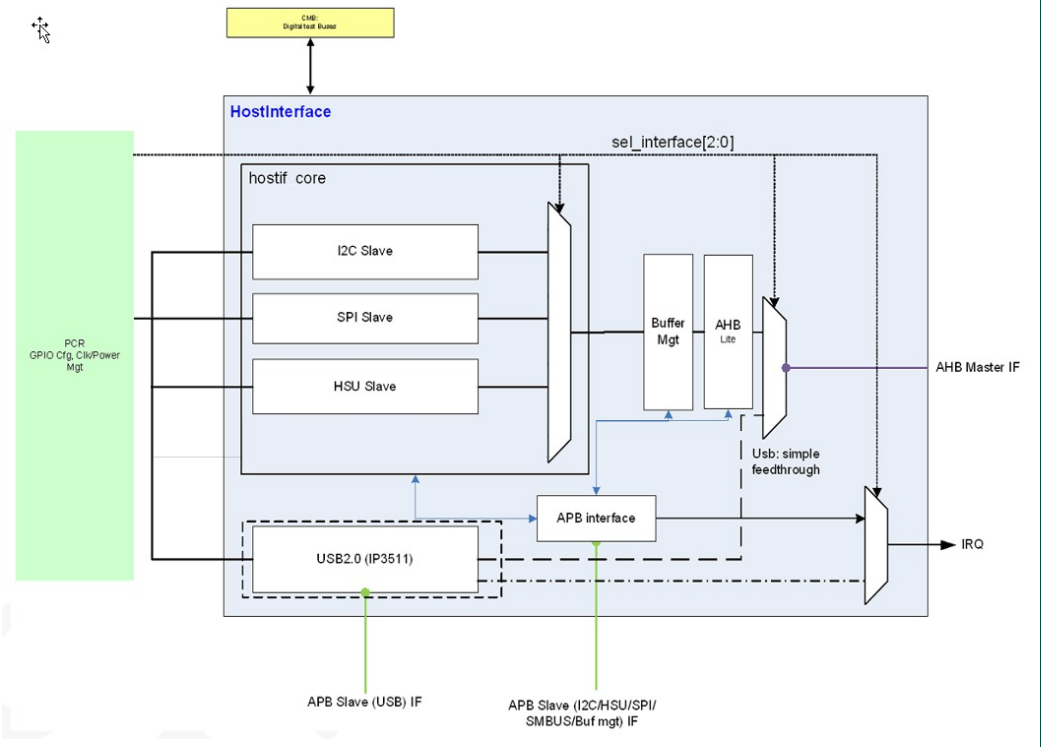



**Fig 55. Block diagram of Host interface block**

**Table 311. Interface selection**

| HIF_Selection | Host interface type |
|---|---|
| 000 | no interface selected |
| 001 | I2C |
| 100 | SPI |
| 011 | HSU |
| 100 | USB |

The Host Interface selection is done by FW/Application program by writing into register in PCR --> PCR_SYS_REG bits [2:0]. It's the responsibility of application developer to correctly select the Host Interface.

### 14.3.1 I2C Slave interface

The host interface of PN7462AU can be also used as I2C slave interface. This is the second of the two I2C controllers supported by PN7462AU. For more details on the I2C master controller please refer to Section 14.1. The I2C interface is compliant with the I2C Bus Specification V3.0.

#### 14.3.1.1 I2C Slave features

- Supports slave I2C bus
- Standard mode, Fast mode (extended to 1Mbds) and High speed mode
- Supports 7-bit addressing mode only
- Selection of I2C address done by 2 pins
  - Support multiple addresses
  - The upper bits of address are hard coded; The value (01010XXb) corresponds to NXP identifier for I2C blocks
- General call (Software reset only)
- Software reset (in Standard and Fast mode only)

#### 14.3.1.2 Pin description

**Table 312. I²C pinning and signal assignments**

| Pin Number | Pin Name | I²C slave |
|---|---|---|
| 6 | ATX_A | SCL |
| 7 | ATX_B | SDA |
| 8 | ATX_C | Adress0 |
| 9 | ATX_D | Adress1 |

#### 14.3.1.3 Configuring the I2C interface

The following parameters can be configured via the register HOSTIF_I2C_CONTROL_REG (see Table 323):

- I2C slave address (2 LSBs)
- High-speed mode (will select high speed mode of pads). Fast mode and standard mode are both enabled by default
- I2C soft reset enable
- I2C Device ID enable

The Host Interface ensures that changes to these parameters will not take effect while output from the Host Interface is BUSY (logic high).

#### 14.3.1.4 Clock requirements

The I2C block uses the system clock to generate the request lines for data exchange. The main I2C functionality is done with the clock supported via the SCL pin. Two 25 ns filters are needed: one each for Start and Stop bit detection.

#### 14.3.1.5 Transfer speed

The host interface supports Standard-mode (Sm), Fast-mode (Fm) and High-speed mode (Hs-mode). Each mode is half-duplex only. The bitrate speeds for each mode are listed in the following table.

**Table 313. I2C Mode Maximum Bitrates**

| Mode | Max Bitrate |
|---|---|
| Standard-mode (Sm) | 100 kbit/s |
| Fast-mode (Fm) | 400 kbit/s |
| High-speed mode (Hs-mode) | 3.4 Mbit/s |

There is no difference in functionality between Standard-mode and Fast-mode; only the frequency of SCL changes. To enable High-speed mode, set bit I2C_HS_ENABLE in register HOSTIF_I2C_CONTROL_REG. With this bit set, the Host Interface responds with an ACK to the slave address following the High-speed mode, Master Code and Repeated Start sequence. In addition, the output port is set logic high to drive the I2C pads in the correct configuration. The I2C interface does not support clock stretching.

#### 14.3.1.6 I2C address

The upper bits of the I2C slave address are hardcoded. The value corresponds to the NXP identifier for I2C blocks. The value is 01010XXb. The lower two bits are defined via field I2C_ADDR in register HOSTIF_I2C_CONTROL_REG. The 10-bit addressing is not supported.

#### 14.3.1.7 ACK/NACK behavior

The following sections describe the conditions under which the I2C block generates a NACK/ACK.

**Host write to I2C:**

The slave address+W will be ACKed provided that:

- Selected host interface is I2C

- I2C address matches the value stored in the Host IF
- RX buffer is available
- No frame overflow
- No buffer overflow

Otherwise, the Host Interface will return a NACK in response to the address phase.

The Host Interface will send an ACK back to the Host for each data byte, provided the following conditions are met:

- RX buffer is available
- No frame overflow
- No buffer overflow

Under all other conditions, the Host Interface sends a NACK back to the Host and the frame is discarded.

Note that in Native mode, frame overflow is not applicable.

**Host read from I2C:**

The Slave Address+R will be ACKed when:

- Selected host interface is I2C.
- I2C address matches the value stored in the Host IF
- TX buffer contains a frame.

Otherwise, the Host Interface will return a NACK in response to the address phase.

#### 14.3.1.8 IDLE byte generation

IDLE Bytes are defined in the I2C standard by setting the SDA line high. The Host receives 0xFF bytes. An IDLE byte will be transmitted if one of the following conditions occurs:

- No frame for transmission is available
- This can also occur if during transmission the buffer becomes empty
- If a NACK is sent without a following stop or restart condition, even when data are available

#### 14.3.1.9 Special modes

**Device ID:** The I2C interface supports the Device ID feature. This can be enabled by setting bit I2C_DEVID_ENABLE in register HOSTIF_I2C_CONTROL_REG.

If HOSTIF_I2C_CONTROL_REG.I2C_DEVID_ENABLE is logic high and the Host sends a Device ID request, the Host Interface Core asserts the corresponding output simultaneously with the Data Request. The Buffer Manager responds to the next 3 byte reads with the Device ID. The Device ID bit stream is shown in Fig 56.

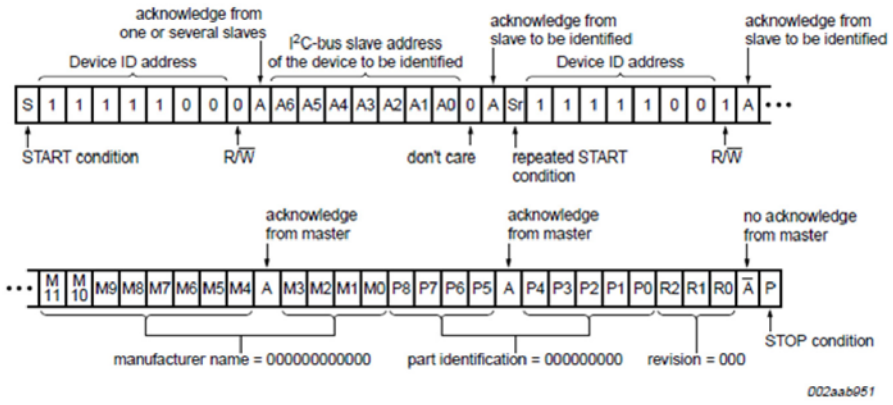



**Fig 56. I2C Device ID Bit stream**

If HOSTIF_I2C_CONTROL_REG.I2C_DEVID_ENABLE is logic low, the Host Interface will respond to the Device ID Address and the I2C Slave Address with NACK.

The Device ID sequence only operates in Standard-mode or Fast-mode.

**Soft Reset via I2C:** The I2C interface can reset the IC in response to a Software reset from the I2C master (Host). This feature can be enabled by setting bit I2C_RESET_ENABLE logic high in register HOSTIF_I2C_CONTROL_REG. When the Host then issues a General Call (00h) followed by 06h, the I2C interface asserts logic high to the corresponding reset output. This will remain logic high until either the PCR responds by asserting the system reset or an *EOF* is detected. As a result, the I2C reset output signal will be driven logic low.

Following the assertion of system reset, HOSTIF_I2C_CONTROL_REG.I2C_RESET_ENABLE is cleared to logic low. This means that any subsequent request by the Host to perform an I2C Soft Reset will be NACKed by the Host Interface.

**Note**: Since the I2C Soft Reset is the only type of General Call address supported by the Host Interface, when I2C_RESET_ENABLE is logic low, the Host Interface responds with to the General Call address (0x00) itself, as well as the I2C Soft Reset command (0x06). The Soft Reset sequence only operates in Standard-mode or Fast-mode.

#### 14.3.1.10 Pads and pad control

In order to be compliant with Fast-mode and High-speed mode, the SCL and SDA pads have spike suppression (50 ns in Standard and Fast-mode and 10ns in High-speed mode) and a Schmitt trigger (100 mV hysteresis) on the input stage. The output stage has a slope control. For High-speed mode only, the rise time of the SCL pulse is shortened using a current-source pull-up circuit within the pad. The pad data and control signals pass through the PCR unit.

### 14.3.2 SPI Slave interface

The host interface of PN7462 can be also used as SPI slave interface. The SPI slave controller is the second of the two SPI controllers supported by PN7462. For more details on the I2C master controller please refer to Section 14.2.

#### 14.3.2.1 SPI Slave features

- Speed up to 7 Mbit/s
- Slave mode only
- 8-bit data format only
- Programmable clock polarity and phase
- Slave selection fixed to positive polarity
- Supports all 4 modes of SPI (CPOL and CPHA)
- Half duplex in HDLL Mode
- Full duplex mode in native mode

#### 14.3.2.2 SPI Slave pin description

**Table 314. SPI pinning**

| Pin Number | Pin Name | SPI slave | Description |
|---|---|---|---|
| 6 | ATX_A | NSS | SPI active-low Slave Select (NSS) |
| 7 | ATX_B | MOSI | SPI Master Output Slave Input (MOSI) |
| 8 | ATX_C | MISO | SPI Master Input Slave Output (MISO) |
| 9 | ATX_D | SCK | SPI Serial clock (SCK) |

#### 14.3.2.3 Configuring the SPI interface

The following parameters can be configured via the register HOSTIF_SPI_CONTROL_REG:

- SPI_CPOL: polarity of SPI clock
- SPI_CPHA: phase of SPIO clock

The Host Interface Core ensures that changes to these parameters will not take effect while the interface core is BUSY (logic high).

#### 14.3.2.4 HDLL Mode

The HDLL mode requires a transfer direction detector, because only half duplex is supported. For this reason, the first byte is evaluated. If the first bit of the first byte is logic low (SOF true), the SPI frame is treated as a Host Write. The Host Interface transmits FFh on MISO. If this bit and all other bits in the byte are logic high, the SPI frame is

treated as a Host Read and the Host Interface transmits the required data bytes on MISO. All other values of the first byte are treated as an invalid access and the Host Interface transmits FFh on MISO. Fig 57 illustrates these three types of accesses.

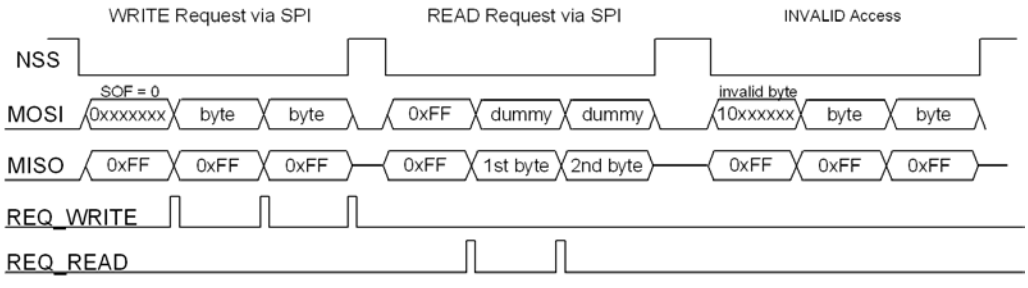



**Fig 57. SPI HDLL mode examples**

#### 14.3.2.5 NCI Mode

The NCI mode described in this section is a low-level protocol not to be confused with the NFC Controller Interface (NCI). The SPI can support NCI with and without CRC, which determines the format of the transport bytes. When the CRC is not being used (fields NCI_CRC_DISABLE and NCI_LENGTH_MODE both logic low in register HOSTIF_CONTROL_REG), the same transport direction detector bytes are used (e.g. if the first bit is logic low then it is a Host Write; otherwise for a Host Read, the MISO is set to 0xFF. If the CRC is included in the protocol (fields NCI_CRC_DISABLE and NCI_LENGTH_MODE both logic high in register HOSTIF_CONTROL_REG), then there is no transport direction detector byte for a Host Write. The Host Read is indicated by the first byte on MISO being 0xFF.

#### 14.3.2.6 Native Mode

The Native mode supports full duplex transfer. Therefore, a Data Request and Data Ready pulse can occur after a short time. Please refer to the timing diagram in Fig 58.

Note that the host, which is always the master, is responsible for ensuring that the length of the TX frame is exactly equal to the RX frame. If this is not the case, then the rising edge of SPI Slave Select in response to the shorter frame (or fragment) would truncate the longer frame.

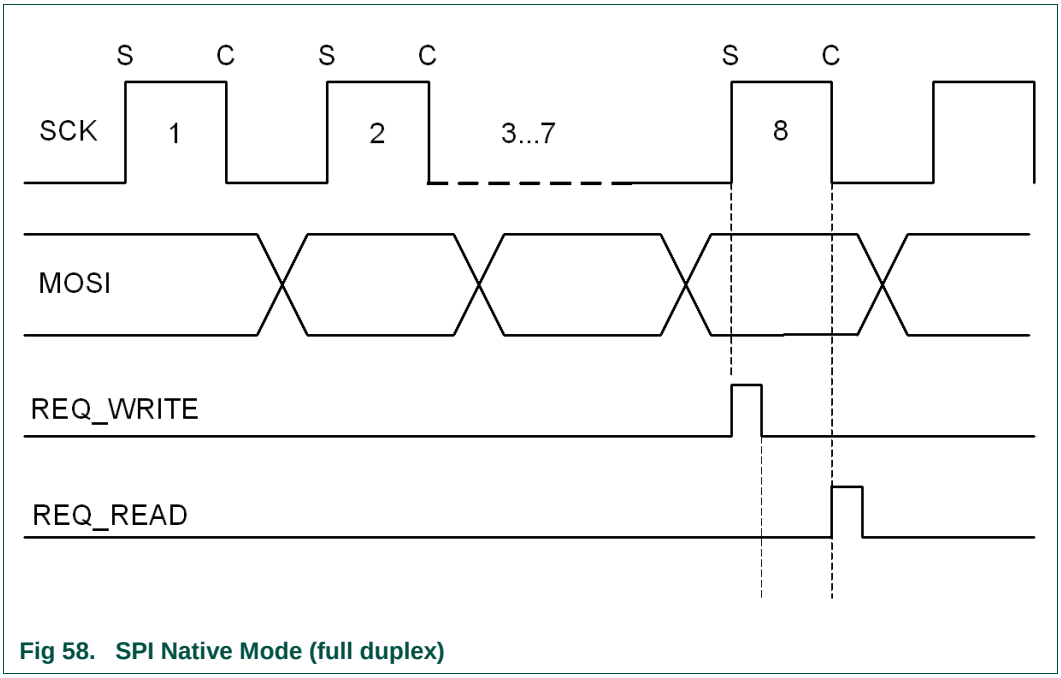



**Fig 58. SPI Native Mode (full duplex)**

#### 14.3.2.7 Response to SPI Host Write

The Host Interface does not assert Data Ready to the Buffer Manager and discards the frame in response to a Host Write unless all of the following conditions are met:

- Selected host interface is SPI
- Input port *go* is logic high
- RX buffer is available
- No frame overflow
- No buffer overflow

Note that in Native mode, frame overflow is not applicable.

#### 14.3.2.8 Response to SPI Host Read

The Host Interface returns 0xFF back to the Host and does not assert Data Request to the Buffer Manager in response to a Host Read unless all the following conditions are met:

- Selected host interface is SPI
- TX buffer contains a frame

#### 14.3.2.9 SPI protocol detection

The SPI protocol is synchronous with the SCK clock based on the system clock, which is fixed at 27.12 MHz.

**Data Request:** Signal Data Request is asserted after the changing edge of the last bit.

**Data Ready:** Signal Data Ready is asserted after the sampling edge of the last bit.

**End of Frame:** Signal *EOF* is generated on the rising edge of SPI Slave Select since a Host must write the frame in a single write access.

#### 14.3.2.10 Pads and pad control

The SPI interface signals are routed to the pads via the PCR.

### 14.3.3 High speed UART

The High Speed Asynchronous Receiver/Transmitter (HS UART) is another host interface supported by PN7462. The HSUART is designed to support bit rates up to 1.288 Mbit/s. The HSUART can operate only in slave mode.

#### 14.3.3.1 HS UART features

- Standard bit-rates - 9600, 19200, 38400, 57600, 115200, and faster speed rate up to 1.288 Mbit/s
- Full Duplex supported
- Supports only one operational mode: Start bit, 8 data bits (LSB), Stop bit(s)
- Number of “stop bits” programmable for RX and TX (1 or 2)
- Configurable length of EOF (1 to 122 bits)

#### 14.3.3.2 HS UART pin description

**Table 315. HS UART pinning and signal assignments**

| Pin Number | Pin Name | HSU | Description |
|---|---|---|---|
| 6 | ATX_A | HSUART RX | High Speed UART - Reception Line |
| 7 | ATX_B | HSUART TX | High Speed UART - Transmission Line |
| 8 | ATX_C | HSUART RTS | High Speed UART – Host interface is ready to receive |
| 9 | ATX_D | HSUART CTS | High Speed UART - Host is ready to receive |

#### 14.3.3.3 HS UART transmission

This interface comprises four wires connected to pads via the PCR:

- Transmit Data (HSU_TX)
- Receive Data (HSU_RX)
- Clear to Send (HSU_CTS_n)
- Request to Send (HSU_RTS_n)

The HS UART takes bytes of data and transmits the individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes. The term “Asynchronous” means no clock being transferred over the line (both the transmitter and the receiver know the baud rate before transmission). The frame is made of several independent bytes. In the receive mode the HS UART works as a slave, while

in transmit mode it will act as a master. The flow control is performed using CTS/RTS. When CTS is low, the PN7462AU HS UART is allowed to send data to the host. PN7462AU will drive RTS low when a non-empty receive buffer is available.

#### 14.3.3.4 HS UART operational mode

The HSUART supports only one operational mode with following characteristics:

- Start bit: The start bit detected when a 0 is asserted on the RX line
- 8 data bits: The data bits are sent/received in the LSB first order
- Stop bit:
  - During reception, the stop bit values are checked (should be equal to 1. If this is not the case, RX_fer interrupt is raised). The number of stop bits is programmable. It can be 1 or 2 (recommended value is 1)
  - During transmission, after the complete data bit transmission, some '1' are transmitted. The number of stop bits is programmable. It varies from 1 to 2 (recommended value is 1).
- EOF: length of EOF is programmable (in bit number). The EOF can vary from 1 to 122 bits. The EOF is sampled at reception side before raising EOR interrupt, and EOF is sent at the end of transmission before sending EOT interrupt.

This is shown in Fig 61.


**Fig 59. HSU byte transmission**

#### 14.3.3.5 HS UART Baud rate generator

To reach the high-speed transfer rate, the HS UART has its own baud rate generator based on sample clock (27.12 MHz). The baud rate generator uses an integer division of this clock, using the division factor defined in register HSU_RX_DIVIDER. The divided clock is used for both RX sampler and TX generator, and can be corrected for both RX and TX, by adding if needed one cycle of sample clock to each bit duration. This is necessary because an integer division of sample clock would not be accurate enough to receive/transmit data over HSU. The clock correction is done using HOSTIF_SAMPLE_REG (fields HSU_RX_CLK_CORRECT and HSU_TX_CLK_CORRECT). See Fig 60 for more details. The RX/TX_CLK_CORRECT fields are 11 bits long. The RX_CLK_CORRECT[0] corrects start bit duration, the RX_CLK_CORRECT[1..8] corrects data bit [0 to 7], and the RX_CLK_CORRECT[9..10] correct the one or two stop bits. Note that RX baud rate generator aims at measuring precisely start time + 1.5bit, 2.5 bit, ... up to 8.5 bit, while TX baud rate generator aims at measuring 1-bit, 2-bit, 3-bit ... Hence a small difference in RX/TX baud rate generators, and in the proposed CLK_CORRECT parameter values.

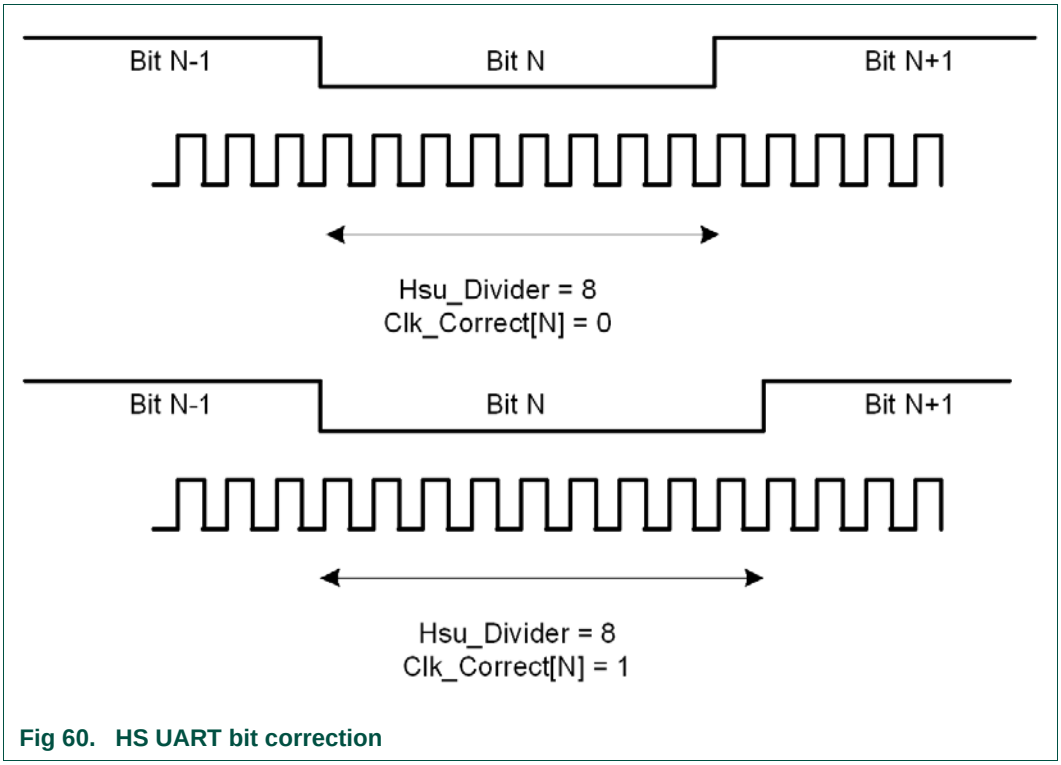



**Fig 60. HS UART bit correction**

Table 316 shows the programming values of HSU_RX/TX_DIVIDER and HSU_RX/TX_CORRECT for various baud rates when sample clock is 27.12 MHz.

**Table 316. HSU baud rates for 27.12 MHz+/-1.5 % sample clock**

| Bit rate (kBaud) | HSU_RX_DIVIDER | HSU_TX_DIVIDER | HSU_RX_CLK_CORRECT | HSU_TX_CLK_CORRECT |
|---|---|---|---|---|
| 9.6 | 2825 | 0 | 000_0000_0000 | 000_0000_0000 |
| 19.2 | 1412 | 0 | 000_0000_0000 | 001_0101_0101 |
| 38.4 | 706 | 0 | 000_0000_0000 | 010_0010_0010 |
| 57.6 | 470 | 0 | 000_0110_1001 | 001_1111_0111 |
| 115.2 | 235 | 0 | 000_0000_0011 | 001_0010_1010 |
| 230.4 | 117 | 0 | 011_0110_0101 | 010_1101_1101 |
| 460.8 | 58 | 0 | 000_1011_1111 | 011_1111_0111 |
| 921.6 | 29 | 0 | 000_0001_1101 | 001_0010_1010 |
| 1288 | 21 | 0 | 000_0000_0000 | 001_0000_0000 |
| 2400 | 11 | 0 | 000_0000_0111 | 001_0001_0010 |
| 3500[1] | 7 | 0 | 011_0011_0111 | 001_1101_1101 |
| 3750[1] | 7 | 0 | 000_0000_0011 | 000_0100_0100 |
| 4000 [1] | 6 | 0 | 011_1100_1011 | 011_1011_1011 |
| 5000 [1] | 5 | 0 | 001_0001_0011 | 001_0010_1010 |

[1] Baud rates above 3.4 Mbit/s depend on the clock accuracy. It is suggested to use only baud rates with a ratio between sample clock and UART clock greater than 8.

#### 14.3.3.6 EOF detection

For HS UART, the EOF is determined by the sampling of EOF_SIZE bits of value one. To prevent the race condition between EOF detection and retransmission from the host, an additional time of 5 bits is required before retransmitting the data. The EOF_size defines the maximum inter-byte duration in a frame, while the EOF_size+5 defines the minimum inter-frame duration between 2 frames.

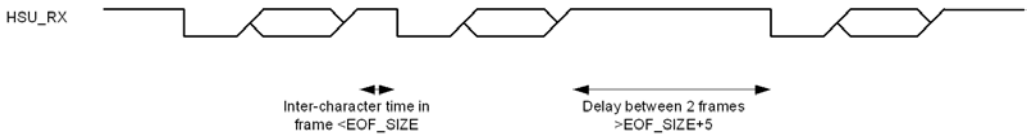



**Fig 61. EOF_SIZE: max inter-byte delay and min inter-frame delay**

**Table 317. EOF duration in us**

| Bit rate (kBaud) | EOF_SIZE=10 | EOF_SIZE=50 | EOF_SIZE=100 |
|---|---|---|---|
| 57.6 | 174 | 869 | 1737 |
| 115.2 | 87 | 434 | 869 |
| 921.6 | 11 | 55 | 109 |
| 1288 | 8 | 39 | 78 |
| 3500 | 2.9 | 14.3 | 28.6 |
| 3750 | 2.7 | 13.4 | 26.7 |
| 4000 | 2.5 | 12.5 | 25 |
| 5000 | 2 | 10 | 20 |

#### 14.3.3.7 Baud rate estimation

In order to be independent of clock ratio between HS UART baud rate and host interface sample clock, a baud rate estimator was implemented. **The principle of this estimator is that the host sends a calibration byte of value 0x00 at beginning of each frame. First transmission from the host interface to the host must occur after previous reception of such a frame when HSU_BR_ESTIMATOR_MODE is 1 (Hence using baud rate estimator in automatic mode is not possible if full-duplex HSU is used).** This byte is used to measure sample clock for reception of next bytes and transmission of next frame. Usage of this baud rate estimator assumes that no important clock frequency deviation will happen from reception of this first byte until the last byte to be sent by host interface. Hence, this is well adapted to receive a message and answer to this message after firmware processing. But if the host interface wants to transmit data with no prior data reception (or with a long time since prior reception), then clock frequency may have changed and an accurate clock will be needed.

The baud rate estimator is active when bit HSU_BR_ESTIMATOR_MODE of HSU_CONTROL_REG is set to 1 or 2. When active, the first received byte must be 0x00, which will be sent as 9 consecutive ‘0’ bits on RX line. Then the number of edges of sample clock are counted and divided by 9: The result of this division is used to

compute HSU_RX_DIVIDER and HSU_TX_DIVDER, and the remainder of this division is used to compute HSU_RX_CORRECT and HSU_TX_CORRECT. The computed value is sent to registers for reading/checking by firmware.

If HSU_BR_ESTIMATOR_MODE is 1, then the computed values (RX/TX_CLK_DIVIDER and RX/TX_CLK_CORRECT) are automatically used for any further data sampling or data sending. If HSU_BR_ESTIMATOR_MODE is 2, then data sampling and data sending is still performed using programmed values. In case the baud rate estimator is active in automatic mode, and TX happens BEFORE RX (should not be done anyway), then the parameters used for clock divider/clock correction are the one programmed in HSU_CONTROL_REG/HSU_SAMPLE_REG. Once a reception occurs, the TX parameters will be taken from HSU_EST_xx registers.

Note that the first calibration byte can be stored in memory by setting HSU_STORE_BR_BYTE to 1. This must not be used for HDLL (since no extra byte is expected before header for payload length analysis).

If the baud rate estimator overflows (duration of first received 0x00 byte is too long), then RX_FER interrupt will be raised, and the value of HSU_EST_RX_DIVIDER_REG is set to 1023.

#### 14.3.3.8 RTS/CTS flow control

The CTS input is used to indicate that the host can (CTS=0) or cannot (CTS=1) receive the data. Before sending a byte, the host interface will sample the value of CTS. If the value is 0, then the whole byte to send will be sent over TX line. If not, then the host interface will wait for CTS to be 0 before sending the byte.

The RTS is used to signal to the host that at least one buffer is ready to receive data (RTS=0). This is shown in Fig 62. Once one RX buffer is available (programmed by firmware), the RTS changes from 1 to 0, allowing the host to send a frame. At the end of the frame reception, the buffer becomes unavailable and the RTS changes to 1 if no other buffer is available, until firmware programs a new buffer. Note that after reset, all buffers are enabled, so by default RTS is '0' and the host can send data. The firmware should ensure that the buffers are correctly configured, or disabled, before enabling the RTS pad.

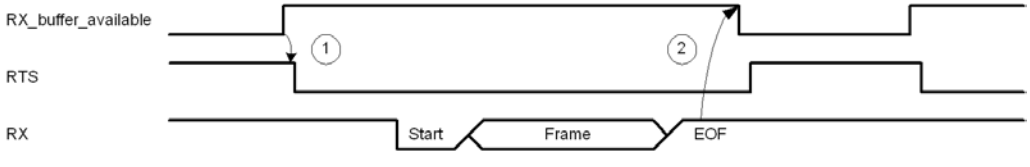



**Fig 62. RTS generation, frame based**

Note that in case the host sends a frame longer than buffer size, a buffer_overflow interrupt will be sent to the firmware. The RTS will not toggle to high. Data received after the error are not stored into memory. The error is signaled to the firmware by an interrupt.

#### 14.3.3.9 Data sampling

Basically, the RX sampler will detect a falling edge on HSU_RX input signal, resynchronize it to sampling clock, then will start counting a number of sample clock cycles before sampling each data bit. The RX data sampling is based on the ratio between the sampling clock (27.12 MHz) and the communication baud rate: integer part of this ratio is stored in HSU_RX_DIVIDER, and the decimal part of this ratio will be reflected in HSU_RX_CORRECT, by distributing '1' in this register with a proportion equivalent to the decimal part of the ratio. The RX sampler will count up to this value to determine the bit length. The sampled time (ideally at the middle of bit length) is determined as HSU_RX_DIVIDER/2 (this is an integer). When the ratio between the sampling clock frequency and the baud rate is not an integer, then HSU_RX_CORRECT provides a mean to extend individually the duration of each sampled bit of one sample clock cycle: the proportion of '1' in this register should reflect the decimal part of the ratio between sampling clock and baud rate. At start bit detection, the counter starts with value 3 to compensate for the start bit detection and re synchronization which consumed 2 to 3 clock cycles (not deterministic due to asynchronicity between RX and sampling clock). Due to clock uncertainty given by start bit detection, average sample time for first bit is HSU_RX_DIVIDER/2+0.5 clock cycles when HSU_RX_DIVIDER is even, and HSU_RX_DIVIDER/2 clock cycles when HSU_RX_DIVIDER is odd. Table 318 shows the duration of each sampled bit. If we only consider 1 stop bit, and we call the number of '1' in RX_correct[9:0], we can compute the average bit duration: (RX_divider+d/10+0.05)xsample_clock_period. This is why d should approximately reflect the decimal part of sample_clock_frequency/baud rate.

**Table 318. Bit duration in sample clock cycle**

| Bit | Duration (# of sample clock cycles) |
|---|---|
| Start bit | RX_divider+0.5+RX correct[0] |
| Bit 0 | RX_divider+RX_correct[1] |
| Bit 1 | RX_divider+RX_correct[2] |
| Bit 2 | RX_divider+RX_correct[3] |
| Bit 3 | RX_divider+RX_correct[4] |
| Bit 4 | RX_divider+RX_correc[5] |
| Bit 5 | RX_divider+RX_correct[6] |
| Bit 6 | RX_divider+RX_correct[7] |
| Bit 7 | RX_divider+RX_correct[8] |
| Stop Bit 0 | RX_divider+RX_correct[9] |
| Stop Bit 1 | RX_divider+RX_correct[10] |

As a conclusion, RX_DIVIDER will be programmed to the integer part of 27.12/baud rate. RX_CORRECT will be programmed with an evenly distributed proportion of 1 equal to the decimal part of this ratio (for example, if ratio if 8.25, then RX_correct will have 1/4 of '1' bits, for example: 01000100). A script is provided which can give one of the best values for RX_correct (best sampling results for given clock accuracy of both the host and the hostif).

#### 14.3.3.10 HS UART reception from host to host interface

The UART reception should be programmed in the following order:

- Program HSU_RX_DIVIDER and HSU_RX_CLK_CORRECT for the expected reception baud rates. Or program HSU_USE_BR_ESTIMATOR to 1
- Program host interface RX_DATA_READY to free a buffer for reception, which will have the effect of driving RTS low, and allow the host to send data
- Wait for End of Reception Interrupt.

At the end of data reception, CPU can read back memory to analyze received data like for other interface.

#### 14.3.3.11 HS UART interrupt

The HS UART can provide an interrupt HSU_RX_FER, (which is not the case for all other interfaces). When this interrupt is set, this means that stop bit was not correctly sampled for one byte. The HSU_RX_FER interrupt does not stop receiving the frame.

#### 14.3.3.12 HS UART transmission from host interface to host

The TX transmission should be performed in the following order:

- Program HSU_RX_DIVIDER and HSU_TX_CLK_CORRECT for the transmission baud rate (if HSU_USE_BR_ESTIMATOR is not already set by previous Reception)
- Program the TX frame in memory
- Program TX_BUFFER_READY to start the buffer manager state machine
- Wait for End Of Transmission Interrupt

The HS UART transmit state machine is a slave to buffer manager state machine: it waits for data from the buffer manager, then sends them to HSU_TX, so the protocol between HS UART and the buffer manager is a bit different from the one used for SPI or I2C. The buffer manager sends a signal that a new data is available. The HS UART acknowledges the received data (at the end of transmission). This can be seen on picture Fig 63. At the end of transmission, the HS UART with generate the EOF signal, by sending as many '1' on TX line as specified by EOF_SIZE. The end of transmission will only be sent to firmware after this EOF transmission, thus allowing firmware to directly send a new frame without any wait time.

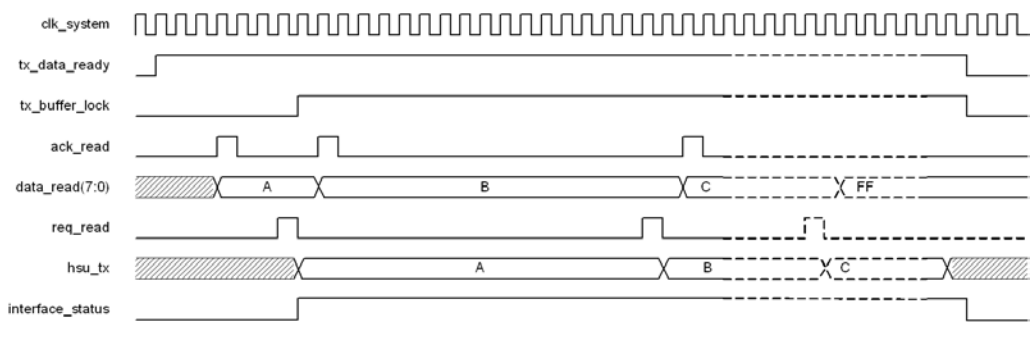



**Fig 63. HS UART-Buf_mgt protocol for transmit sequence**

The TX_DIVIDER will be programmed to 0 (RX_DIVIDER value is used), and a provided script will be used to compute TX_CORRECT.

#### 14.3.3.13 Data generation

The data generation is a bit easier than data sampling, since there is no need to compute when the middle of a bit happens. Also there is nothing to resynchronize. The output data will be in sync with the sample clock. The TX generator is a simple clock counter, using HSU_DIVIDER registers and HSU_TX_CORRECT for each bit to send. For TX, HSU_DIVIDER = HSU_RX_DIVIDER + HSU_TX_DIVIDER, with TX_DIVIDER=0 or 1 only. This is due to the fact that for optimization, the RX_divider is sometimes decreased by 1 to finely tune when sampling happens. At the end of each period, next bit is sent over HSU_TX.

#### 14.3.3.14 HS UART and HDLL mode

In HDLL mode, the host is expected to send HDLL frames starting with the 2 bytes HDLL header, followed by payload then 2 bytes CRC) with NO extra bytes. All data except CRC will be stored in the memory.  No difference to I2C/SPI is expected.

#### 14.3.3.15 HS UART and NCI mode

In NCI mode, the host may start each frame with 1 to 3 dummy bytes. This dummy byte will be always stored in the memory. The reason why these dummy bytes are transferred is due to the fact that during the standby, the host interface is inactive and may not be waken fast enough to sample the first bytes sent by the host. In case the host communicates during the standby, the PCR will raise the RTS to 1 to prevent the host from sending other data, and we assume that no more than 3 bytes will be sent. These 3 bytes will be lost. The host cannot determine when PN7462AU is in the standby. Therefore, we recommend sending always 3 dummy bytes at beginning of frames in the NCI mode.

#### 14.3.3.16 Wake-up from standby

In NCI mode, it is expected that up to dummy 3 bytes are lost when the host wakes-up the PN7462AU from the standby. Reception of these bytes can be simulated in order to ease writing of the firmware. For this reason, there is a control bit (WAKEUP_STANDY in HSU_CONTROL_REG) which can be used only once. When this bit is set, the HSUART will simulate reception of 1 to 3 dummy bytes (number configured by WAKEUP_BYTES) at first address of reception buffer. Next frames will be stored in memory without any byte addition.

In HDLL mode, WAKEUP_STANDY should not be programmed. The message sent by the host to wake-up the PN7462AU can be a simple byte which will be discarded by PCR and not seen by the host interface.

#### 14.3.3.17 Pads and pad control

The UART interface signals are routed to the pads via the PCR.

### 14.3.4 Buffer Manager

The buffer manager transfers data between the host interface and the SRAM. It is shared between High-Speed UART, I2C and SPI interfaces whereas USB has its own buffer management. It processes the incoming frames (extracts the packet length, verifies the

HDLL CRC) and checks for frame over/underflow and inter-character timeout. For outgoing frames, the buffer manager monitors the delay between bytes. Exceeding a threshold will trigger an interrupt. It supports up to 4 receive buffers and 1 transmit buffer listed in Table 319.

**Table 319. Buffer ID assignment**

| Buffer Name | Buffer ID |
|---|---|
| RX0 | 0 |
| RX1 | 1 |
| RX2 | 2 |
| RX3 | 3 |
| TX | 4 |

The buffer manager supports various transport streams (HDLL, native, NCI) which are not supported equally for each interface (depending on physical/logical difference in protocols).

**Table 320. ed format per interface**

| Interface | HDLL | Native (debug only) | NCU, No Header, No CRC | NCI, No Header, CRC | NCI, Header, No CRC | NCI, Header, CRC |
|---|---|---|---|---|---|---|
| I2C | Yes | Yes | Yes | Yes | Yes | Yes |
| SPI | Yes | Yes | Yes | Yes | Yes | Yes |
| HSU | Yes | Yes | Yes | Yes | Yes | Yes |

#### 14.3.4.1 Buffer initialization

##### RX buffers

Each of the RX buffers may be independently disabled by setting bit RX<n>_BUFFER_DISABLE logic high in register HOSTIF_BUFFER_RX<n>_CFG_REG.

Each of the RX buffers must be configured with the following parameters, which are defined in HOSTIF_BUFFER_RX<n>_CFG_REG:

- Maximum buffer size
- Start address
- Normal or short frame assignment
- Header offset

The maximum buffer size should be set to be greater than or equal to the sum of:

- HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_HEADER_OFFSET
- Number of header bytes
- Payload

It is used to detect an RX buffer overflow (described in Section 14.3.4.11).

The start address is defined in field

HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_START_ADDR. The memory address is incremented after a word has been received. The address is only incremented internally meaning that the memory start address is not changed.

The RX buffers can be labelled to receive either non-short frames or short frames depending on the value of bit

HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_SHORT_FRAME_BUFFER. This is not applicable to the Native Mode.

HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_HEADER_OFFSET defines the number of padding bytes to add to the frame header before writing the first word to the RX buffer.

By default, this value is set to 2 since the standard HDLL header is 2 bytes long. **Since the minimum length of the header is 2 bytes, an offset of 3 is not permitted since this would result in the header being stored across the first two words in the buffer.**

#### TX buffer

There is no explicit disable field in register HOSTIF_BUFFER_TX_CFG_REG since the

TX buffer is effectively disabled when

HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY is cleared to logic low.

The TX buffer must be configured with the following parameters, which are defined in

HOSTIF_BUFFER_TX_CFG_REG:

- Start address
- Header offset
- Empty payload enable

The start address bit 0-14 is defined in field
HOSTIF_BUFFER_TX_CFG_REG.TX_START_ADDR.
Base address is fix in SRAM (address 0x100000).

The header offset is defined in field

HOSTIF_BUFFER_TX_CFG_REG.TX_HEADER_OFFSET. It states the number of bytes to skip before sending them to the Host Interface Core

To configure the buffer manager to send only the header and its calculated CRC, set

HOSTIF_BUFFER_TX_CFG_REG.TX_EMPTY_PAYLOAD_ENABLE to logic high.

This is not applicable to native mode.

### 14.3.4.2 HDLL format

The HDLL frame format is shown in Fig 64. Both the HDLL header and payload are written to SRAM. The CRC is not written.

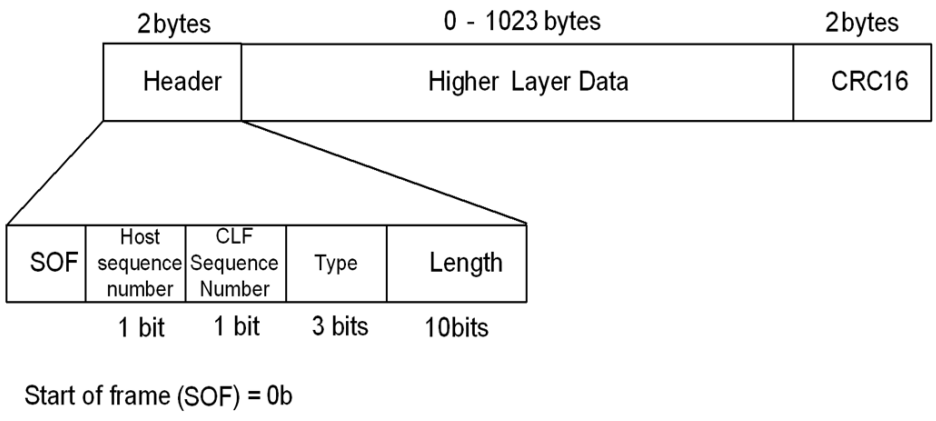



**Fig 64. Data-link layer frame format**

**Frame reception**

The buffer manager extracts the frame length from the header. This gives the number of bytes in the payload. It is compared with the length of a short frame as defined by the field SHORT_FRAME_LEN in register HOSTIF_CONTROL_REG. If the frame length is equal to or less than SHORT_FRAME_LEN, the buffer manager will try to find an available RX buffer, which has been labelled as a “short frame buffer”. Otherwise it will search for an available non-short RX buffer. The frame header is stored in the first word of the selected RX buffer (with the number of padding bytes defined by

HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_HEADER_OFFSET). The payload is stored from the second word onwards. The CRC is extracted by the CRC checker for verification; it is not stored in the RX buffer.

**Frame transmission**

The buffer manager reads the first word from the TX buffer and removes the *m* most significant bytes (where m = HOSTIF_BUFFER_TX_CFG_REG.TX_HEADER_OFFSET) to get the header, from which it extracts the frame length. It then reads out the number of bytes corresponding to this frame length. Once the payload has been read out, the CRC

Generator appends the CRC to the end of the TX frame.

#### 14.3.4.3 Native format

The Native frame format has neither a header nor a CRC: it is a free-format payload.

**Frame reception**

The payload is stored from the second word of the RX buffer onwards. When the buffer manager sees that EOF is asserted, it writes the number of received bytes into the first word of the buffer. The format to write this number of bytes is similar to HDLL header with offset 0: byte at address 0 is Counter MSB (3 bits), byte at address 1 is Counter LSB (8 bits). 16 MSB of 1st word are 0.

**Frame transmission**

The buffer manager reads the first word from the TX buffer to get the number of bytes to send. It then reads out this number of payload bytes, starting from the second word in the buffer. The contents of the first word (number of bytes) is not sent.

#### 14.3.4.4 NCI modes

The possible NCI modes with corresponding register settings are summarized in Table 321.

**Table 321. Summary of possible NCI modes**

| Mode Name | Options | | HOSTIF_CONTROL_REG | | | | Description |
|---|---|---|---|---|---|---|---|
| | CRC | Length Info | BUFFER_FORMAT | NCI_MODE | NCI_CRC_DISABLE | NCI_LENGTH_MODE | |
| NCI over I2C (no CRC) | N | N | 0 | 1 | 1 | 1 | No additional transport layer specified by NCI<br>Number of received bytes stored in RX Length register<br>Number of transmitted bytes stored in TX Length register |
| NCI over I2C (with CRC) | Y | Y | 0 | 1 | 0 | 0 | Supported but not required for NCI |
| NCI over SPI (no CRC) | N | N | 0 | 1 | 1 | 1 | Uses HDLL (half-duplex) implementation<br>Transport mapping:<br>Existing 1-byte header as used in HDLL mode in Tx mode<br>New extra byte in Rx mode<br>Number of bytes received stored in RX Length register<br>Number of bytes to transmit stored in TX Length register |
| NCI over SPI (with CRC) | Y | Y | 0 | 1 | 0 | 0 | Uses HDLL (half-duplex) implementation<br>Transport mapping:<br>Existing 1-byte header as used in HDLL mode in Tx mode<br>No extra byte in Rx mode |
| NCI over HSU (no CRC) | N | N | 0 | 1 | 1 | 1 | No additional transport layer specified by NCI<br>Number of bytes received stored in RX Length register<br>Number of bytes to transmit stored in TX Length register<br>Special attention to the fact that all wake-up bytes are stored into memory and counted in length register. |

| Mode Name | Options | | HOSTIF_CONTROL_REG | | | | Description |
|---|---|---|---|---|---|---|---|
| | CRC | Length Info | BUFFER_ FORMAT | NCI_MODE | NCI_CRC_ DISABLE | NCI_LENGTH_ MODE | |
| | | | | | | | Baudrate estimator byte can also be stored in memory |
| NCI over HSU (with CRC) | Y | Y | 0 | 1 | 0 | 0 | Supported but not required for NCI.<br>In this case, extra wake-up bytes should not be sent by the host |

#### 14.3.4.5 Buffer handshaking

Each of the four receive buffers, as well as the transmit buffer, has an associated status bit in the HOSTIF_DATA_READY_STATUS_REG register. In the case of an RX buffer, the buffer manager sets bit HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY to indicate to the firmware that the buffer contains an error-free frame received from the Host. For TX, the buffer manager clears HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY bit to indicate to the firmware that it has sent the entire frame to the host which is in the TX buffer.

The firmware access to these bits is via the registers HOSTIF_SET_DATA_READY_REG and HOSTIF_CLR_DATA_READY_REG.

**Receiving frames**

The Buffer Manager examines the Frame Length in the header and searches the next available buffer by checking that all the following conditions are true:

- HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_BUFFER_DISABLE = 0
- HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY = 0; HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_MAX_SIZE >= (Frame Length+2)

If the frame is a short frame (Frame Length = HOSTIF_CONTROL_REG.SHORT_FRAME_LEN) then the Buffer Manager also checks:

- HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_SHORT_FRAME_BUFFER = 1.

Else it looks for a buffer dedicated to receive non-short frames:

- HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_SHORT_FRAME_BUFFER = 0

The buffer manager indicates that it has locked the selected RX buffer for its own use by setting bit RX<n>_BUFFER_LOCK in register HOSTIF_STATUS_REG. When the buffer manager signals an EOR event, indicating that the complete frame has been transferred to the selected RX buffer, it sets bit HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY logic high and clears HOSTIF_STATUS_REG.RX<n>_BUFFER_LOCK logic low.

Note that the search order when checking for an available RX buffer is incremental and circular. For example, assume that RX buffers with Buffer ID=0, 1 and 2 have been assigned to store non-short frames and that RX buffer with Buffer ID=3 will store short frames. If the previous non-short frame was stored in RX buffer with Buffer ID=0, then the

Buffer Manager will first check if the RX buffer with Buffer ID=1 is available. If it is not available, it will check RX buffer with Buffer ID=2 and failing that the search will wrap around and check RX buffer with Buffer ID=0.

If there are no suitable RX buffers available, the buffer manager sets RX_BUFFER_NOT_AVAILABLE_STATUS in register HOSTIF_INT_STATUS_REG and asserts the corresponding outputs logic high. The output will remain high until the firmware sets RX_BUFFER_NOT_AVAILABLE_CLR_STATUS bit in register HOSTIF_INT_CLR_STATUS_REG.

**NOTES:**

1. The buffer manager can only set HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY bit.
2. Setting HOSTIF_SET_DATA_READY_REG.SET_RX<n>_DATA_READY bit by the firmware will only cause HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY bit to be set if the Buffer Manager is not using the RX<n> buffer (HOSTIF_STATUS_REG.RX<n>_BUFFER_LOCK bit is 0). This bit is only intended for debugging purposes.
3. Setting HOSTIF_CLR_DATA_READY_REG.CLR_RX_DATA_READY bit by the firmware will only cause HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY bit to be cleared if the buffer manager is not using the RX<n> buffer (HOSTIF_STATUS_REG.RX<n>_BUFFER_LOCK bit is 0).

### Transmitting frames

Once the firmware has loaded the TX buffer with a frame, it should set bit HOSTIF_SET_DATA_READY_REG.SET_TX_DATA_READY logic high to indicate to the buffer manager that the frame is ready for sending. This causes bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY to go logic high. When the buffer manager receives the first read request, it checks that the following conditions are satisfied:

- HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY = 1
- HOSTIF_STATUS_REG.TX_BUFFER_PREFETCH_OK = 1

If both are true, the buffer manager locks the TX buffer for its own use by setting bit HOSTIF_STATUS_REG.TX_BUFFER_LOCK and starts to send bytes. If it is not set, it will set TX_FRAME_NOT_AVAILABLE_STATUS flag in HOSTIF_INT_STATUS_REG register. Once the entire frame has been sent, the buffer manager performs the following actions:

- generates an EOT event
- clears bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY logic low
- clears bit HOSTIF_STATUS_REG.TX_BUFFER_LOCK logic low
- clears bit HOSTIF_STATUS_REG.TX_BUFFER_PREFETCH_OK logic low

**NOTES**

1. The buffer manager can only clear bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY.
2. Setting bit HOSTIF_SET_DATA_READY_REG.SET_TX_DATA_READY by the firmware will only cause bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY to be set if the buffer manager is not using the TX buffer (bit HOSTIF_STATUS_REGTX_BUFFER_LOCK is 0). Setting bit HOSTIF_CLR_DATA_READY_REG.CLR_TX_DATA_READY by the firmware will only cause bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY to be cleared if the buffer manager is not using the TX buffer (bit HOSTIF_STATUS_REGTX_BUFFER_LOCK is 0).

#### 14.3.4.6 EOR

When the EOF logic is asserted high and the Buffer Manager has written the last word to the buffer, an EOR event is generated. The EOR_STATUS field in register HOSTIF_INT_STATUS_REG is set.

#### 14.3.4.7 EOT

When the EOF logic is asserted high and the last word has been sent from the TX buffer, the Buffer Manager generates an EOT event. The EOT_STATUS field in register HOSTIF_INT_STATUS_REG is set and bit HOSTIF_STATUS_REG.TX_BUFFER_LOCK is cleared logic low.

#### 14.3.4.8 Waterlevel reached

If the field WATERLEVEL in register HOSTIF_WATERLEVEL_REG is non-zero, then the Buffer Manager generates a Waterlevel Reached event when the number of bytes received is equal to WATERLEVEL. The field WATERLEVEL_REACHED_STATUS in register HOSTIF_INT_STATUS_REG is set. An EOR event is not generated until the end of the frame as normal.

#### 14.3.4.9 RX frame underflow

If fewer bytes have been received than indicated in the Frame Length of the Data-link header when the EOF logic is asserted high, the buffer manager sets RX_FRAME_UNDERFLOW_STATUS in register HOSTIF_INT_STATUS_REG. The frame is discarded by not setting HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY. An EOR event is not generated. The EOR_STATUS flag in register HOSTIF_INT_STATUS_REG maintains logic low.

Note that there is no RX Frame Underflow detection in native mode.

#### 14.3.4.10 RX frame overflow

If a byte is received after a correct CRC, the buffer manager sets RX_FRAME_OVERFLOW_STATUS in register HOSTIF_INT_STATUS_REG. No bytes beyond the frame length limit are stored and the frame is discarded by not setting HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY. The corresponding output will remain high until the firmware sets RX_BUFFER_FRAME_OVERFLOW_CLR_STATUS in register

HOSTIF_INT_CLR_STATUS_REG. An EOR event is not generated. The EOR_STATUS flag in register HOSTIF_INT_STATUS_REG maintains logic low.

Note that there is no RX Frame Overflow detection in native mode.

Note that extra bytes received before the expected CRC bytes will result in a CRC error and not a RX frame overflow error.

#### 14.3.4.11 RX buffer overflow

If the number of bytes received exceeds the maximum size of the buffer (defined in HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_MAX_SIZE) then the Buffer Manager will set RX_BUFFER_OVERFLOW_STATUS in register HOSTIF_INT_STATUS_REG and assert the corresponding output. In the Native Mode, if the number of bytes received exceeds (MAX_SIZE - 4) where MAX_SIZE is defined in HOSTIF_BUFFER_RX<n>_CFG_REG.RX<n>_MAX_SIZE, then the Buffer Manager will set RX_BUFFER_OVERFLOW_STATUS in HOSTIF_INT_STATUS_REG register and assert the corresponding output to the Host Interface Core. Note that the threshold is (MAX_SIZE - 4) because the first word in the buffer is reserved for storing the number of bytes received. The output will remain high until the firmware sets RX_BUFFER_OVERFLOW_CLR_STATUS in register HOSTIF_INT_CLR_STATUS_REG. An EOR event is not generated. The EOR_STATUS flag in register HOSTIF_INT_STATUS_REG maintains logic low. The frame is discarded by not setting HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY.

#### 14.3.4.12 CRC verification and generation

The 2-byte data-link CRC is appended to the end of the data-link frame as shown in and is described by the CRC-16-CCITT polynomial:

$$x^{16}+x^{12}+x^{5}+1$$

This is implemented using the polynomial (1)1021, operating on the bit sequence MSB first. It has an initial value of 0xFFFF and is calculated over the entire data-link frame (header and payload). For received frames from the Host Interface Core, the Buffer Manager continually calculates the CRC. At the end of the frame, it extracts the sent CRC and compares it with its internally calculated CRC. If there is a mismatch, the Buffer Manager sets CRC_NOK_STATUS in register HOSTIF_INT_STATUS_REG. The frame is discarded by not setting HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY.

An EOR event is not generated. The EOR_STATUS flag in register HOSTIF_INT_STATUS_REG maintains logic low.

For transmitted frames, the buffer manager calculates the 2-byte CRC and appends it to the end of the frame.

Note that there is no CRC in native mode.

#### 14.3.4.13 TX frame not available

If there is no data ready to be sent in the transmit buffer (bit TX_DATA_READY in register HOSTIF_DATA_READY_STATUS_REG is logic low) when the Host Interface Core Asserts *Data Request*, the Buffer Manager sets TX_FRAME_NOT_AVAILABLE_STATUS in register HOSTIF_INT_STATUS_REG and asserts the corresponding output logic high. The output will remain logic high until the firmware sets TX_FRAME_NOT_AVAILABLE_CLR_STATUS in register HOSTIF_INT_CLR_STATUS_REG.

For I2C: if the host reads a 1st data while no buffer is available, data will be NACKed and no IRQ will be raised.

For SPI: if the host reads a 1st data while no buffer is available, 0xff will be sent and no IRQ will be raised.

For HSUART: not applicable since the host interface is master of HSUART TX transaction.

#### 14.3.4.14 TX Inter-Character (TIC) timeout

The TX Inter-Character (TIC) timer is used to measure the delay between sending characters. The following features are implemented:

- Programmable timeout
- Timer is reloaded with the value in
- HOSTIF_TIC_TIMEOUT_REG.TX_TIMEOUT_VALUE after the transmission of each character
- Automatic start of timer after first character has been transmitted
- Automatic stop of timer after number of transmitted bytes is equal to the frame length.
- Timeout range 200 ms with a granularity of 5 us

A divide-by-100 prescaler divides down the high frequency oscillator clock in order to generate a 3.6us tick. This tick clocks the 16-bit TIC timer, whose initial value is specified in field TX_TIMEOUT_VALUE of register HOSTIF_TIC_TIMER_REG.

**This field must be loaded with a non-zero value. Since the power-on-reset value is zero, the TIC timer is disabled by default.**

The TIC counter will only start decrementing when HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY=1 (this is to avoid the case where the host attempts to read an empty buffer).

If the TIC timer expires during transmission, the buffer manager sets TX_TIMEOUT_STATUS in register HOSTIF_INT_STATUS_REG. The buffer manager also clears bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY logic low to abort the frame transmission

**The firmware is responsible for re-sending the frame in the event of a TIC timeout by setting once again HOSTIF_SET_DATA_READY_REG.SET_TX_DATA_READY.** This will cause the Buffer Manager to initiate a new read prefetch sequence. An EOT event is not generated. The EOT_STATUS flag in register HOSTIF_INT_STATUS_REG

maintains logic low. The HOSTIF_STATUS_REG.TX_BUFFER_LOCK bit is cleared logic low.

#### 14.3.4.15 Storing erroneous RX frames

By default, the incoming frames are discarded by the buffer manager if it detects an error. This is achieved by *not* setting bit RX<n>_DATA_READY in register HOSTIF_DATA_READY_STATUS_REG. In parallel, the buffer manager sets the bit associated with the error event in register HOSTIF_INT_STATUS in order to signal an interrupt to the CPU. If one of the following errors occurs when a frame is received, it may still be stored in the RX buffer such that they may be analyzed for debug.

- Frame underflow
- Frame overflow
- CRC error

This mode is enabled by setting bit STORE_RX_ERROR_FRAMES in register HOSTIF_CONTROL_REG.
HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY will be set logic high even when the received frame contains an error. The interrupt associated with the error type is also still generated. An EOR event is not generated. The EOR_STATUS flag in register HOSTIF_INT_STATUS_REG maintains logic low. This feature is not applicable in native mode since none of the above error conditions can be detected.

#### 14.3.4.16 Sending frames directly from the CLIF RX buffer (“direct copy”)

Due to the fact that the HDLL frame payload is stored word-aligned in the buffers, with the first word in buffer reserved for the header bytes, the frame data can be transferred directly from one interface to another without an intermediate copy.

In order to send out a frame via host interface whose payload is that of a frame previously received via the CLIF, the firmware must configure the start address of the host interface TX buffer to be the same as the CLIF RX buffer. Once the frame has been received, the firmware only needs to update the first word in the buffer to convert the CLIF frame header to the HDLL header for host interface communication. The direct copy is not applicable in native mode since the frame format is unknown a priori.

### 14.3.5 Host Interface register overview and description

#### 14.3.5.1 Register overview

**Table 322. Register overview (base address 0x4002 0000)**

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| HOSTIF_STATUS_REG | 0000h | 32 | R | 00000000 | Status |
| HOSTIF_CONTROL_REG | 0004h | 32 | R/W | 0017ff80 | Buffer manager control |
| HOSTIF_HEADER_CONTROL_REG | 0008h | 32 | R/W | 00000520 | Header description control |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| HOSTIF_I2C_CONTROL_REG | 000Ch | 32 | R/W | 00000000 | I2C interface control |
| HOSTIF_SPI_CONTROL_REG | 0010h | 32 | R/W | 00000000 | SPI interface control |
| HOSTIF_HSU_CONTROL_REG | 0014h | 32 | R/W | 01410000 | HSU control register |
| HOSTIF_HSU_SAMPLE_REG | 0018h | 32 | R/W | 00000000 | HSU register for sample clock |
| HOSTIF_HSU_EST_CLOCK_DIVIDER_ REG | 001Ch | 32 | R | 00000000 | HSU estimated clock dividers |
| HOSTIF_HSU_EST_CLOCK_CORREC T_REG | 0020h | 32 | R | 00000000 | HSU estimated clock correction parameters |
| INTERNAL_USE | 0024h | 32 | R/W | 00061000 | For internal use |
| INTERNAL_USE | 0028h | 32 | R/W | 00000000 | For internal use |
| INTERNAL_USE | 002Ch | 32 | R/W | 00000000 | For internal use |
| INTERNAL_USE | 0030h | 32 | R/W | 00000000 | For internal use |
| INTERNAL_USE | 0034h | 32 | R/W | 00000000 | For internal use |
| HOSTIF_BUFFER_RX0_CFG_REG | 0038h | 32 | R/W | 08000000 | Configuration of RX buffer 0 |
| HOSTIF_BUFFER_RX1_CFG_REG | 003Ch | 32 | R/W | 08000000 | Configuration of RX buffer 1 |
| HOSTIF_BUFFER_RX2_CFG_REG | 0040h | 32 | R/W | 08000000 | Configuration of RX buffer 2 |
| HOSTIF_BUFFER_RX3_CFG_REG | 0044h | 32 | R/W | 08000000 | Configuration of RX buffer 3 |
| HOSTIF_BUFFER_TX_CFG_REG | 0048h | 32 | R/W | 00008000 | Configuration of TX buffer |
| HOSTIF_BUFFER_RX0_LEN_REG | 004Ch | 32 | R | 00000000 | Number of bytes stored in RX buffer 0 |
| HOSTIF_BUFFER_RX1_LEN_REG | 0050h | 32 | R | 00000000 | Number of bytes stored in RX buffer 1 |
| HOSTIF_BUFFER_RX2_LEN_REG | 0054h | 32 | R | 00000000 | Number of bytes stored in RX buffer 2 |
| HOSTIF_BUFFER_RX3_LEN_REG | 0058h | 32 | R | 00000000 | Number of bytes stored in RX buffer 3 |
| HOSTIF_BUFFER_TX_LEN_REG | 005Ch | 32 | R/W | 00000000 | Number of bytes stored in TX buffer |
| HOSTIF_TIC_TIMEOUT_REG | 0060h | 32 | R/W | 00000000 | Inter-character timeout |
| HOSTIF_WATERLEVEL_REG | 0064h | 32 | R/W | 00000000 | Water level |
| HOSTIF_SET_DATA_READY_REG | 0068h | 32 | W | 00000000 | Set data ready flags for buffers |

| Name | Address offset | Width (bits) | Access | Reset value | Description |
|---|---|---|---|---|---|
| HOSTIF_CLR_DATA_READY_REG | 006Ch | 32 | W | 00000000 | Clear data ready flags for buffers |
| HOSTIF_DATA_READY_STATUS_REG | 0070h | 32 | R | 00000000 | Status of data ready flags for buffers |
| HOSTIF_DBG_RX_REG | 0074h | 32 | R | 00000000 | Debug Received data |
| HOSTIF_DBG_RX_ADDR_REG | 0078h | 32 | R | 00000000 | Debug receive address |
| INTERNAL_USE | 007Ch | 32 | R/W | 00000000 | For internal use |
| INTERNAL_USE | 0080h | 32 | R | 00000000 | For internal use |
| INTERNAL_USE | 0084h | 32 | R/W | 000000Fh | For internal use |
| HOSTIF_INT_CLR_ENABLE_REG | 3FD8h | 32 | W | 00000000 | Clear interrupt enable |
| HOSTIF_INT_SET_ENABLE_REG | 3FDCh | 32 | W | 00000000 | Set interrupt enable |
| HOSTIF_INT_STATUS_REG | 3FE0h | 32 | R | 00000000 | Interrupt status |
| HOSTIF_INT_ENABLE_REG | 3FE4h | 32 | R | 00000000 | Interrupt enable |
| HOSTIF_INT_CLR_STATUS_REG | 3FE8h | 32 | W | 00000000 | Clear interrupt |

#### 14.3.5.2 Register description

**HOSTIF_STATUS_REG**

This register reflects the current status of the host Interface.

**Table 323. HOSTIF_STATUS_REG (address offset 0x0000)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:6 | RESERVED | R | 0 | Reserved |
| 5 | TX_BUFFER_PREFETCH_OK | R | 0 | 1 - Read prefetch of TX buffer completed. |
| 4 | TX_BUFFER_LOCK | R | 0 | 1 - TX buffer is currently in use by the hardware. |
| 3 | RX3_BUFFER_LOCK | R | 0 | 1 - RX buffer 3 is currently in use by the hardware. |
| 2 | RX2_BUFFER_LOCK | R | 0 | 1 - RX buffer 2 is currently in use by the hardware. |
| 1 | RX1_BUFFER_LOCK | R | 0 | 1 - RX buffer 1 is currently in use by the hardware. |
| 0 | RX0_BUFFER_LOCK | R | 0 | 1 - RX buffer 0 is currently in use by the hardware. |

**HOSTIF_CONTROL_REG**

This register is used to control the buffer manager.

**Table 324. HOSTIF_CONTROL_REG (address offset 0x0004)**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31:21 | RESERVED | R | 0 | Reserved |
| 20:7 | BUFFERS_SIZE | R/W | 0x2FFF | RAM aperture size |
| 6 | HIF_ENABLE | R/W | 0 | 0- All inputs disabled. Should be programmed to 1 when host if is fully configured only. Includes USB. |
| 5 | NCI_LENGTH_MODE [2] | R/W | 0 | 0 - Length in memory and transferred over HDLL header<br>1 - Length in register. Not transferred over physical interface |
| 4 | NCI_CRC_DISABLE [2] | R/W | 0 | 0 – CRC active<br>1 – No CRC |
| 3 | NCI_MODE | R/W | 0 | 0 – NCI not used<br>1 – NCI mode active |
| 2 | SHORT_FRAME_LEN | R/W | 0 | Maximum number of payload bytes in a short frame<br>0 – 2 bytes<br>1 – 3 bytes |
| 1 | STORE_RX_ERROR_FRAMES | R/W | 0 | Store erroneous RX frames.<br>HOSTIF_DATA_READY_STATUS_REG.<br>RX<n>_DATA_READY is set by Host IF as if frame were received error-free. |
| 0 | BUFFER_FORMAT | R/W | 0 | 0 - HDLL Frames<br>1 - Native format |

[2] This bit only has effect if NCI_MODE = 1; For NCI mode description see chapter 14.3.2.5

HOSTIF_HEADER_CONTROL_REG
This register is used control the header description.

**Table 325. HOSTIF_HEADER_CONTROL_REG (address offset 0x0008)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | R | 0 | Reserved |
| 11:10 | HEADER_SIZE | W | 1 | Number of bytes -1 of header. Default is for HDLL header. |
| 9:8 | LENGTH_MSB_BYTE_POS | W | 0 | Byte position of Length MSB (in reception order). Default is for HDLL header. |
| 7:5 | LENGTH_MSB_BIT_POS | W | 1 | Bit position of Length MSB. Default is for HDLL header. |
| 4:3 | LENGTH_LSB_BYTE_POS | W | 1 | Byte position of Length LSB (in reception order). Default is for HDLL header. |
| 2:0 | LENGTH_LSB_BIT_POS | W | 0 | Bit position of Length LSB. Default is for HDLL header. |

### HOSTIF_I2C_CONTROL_REG

This register is used is to control the I²C host interface.

**Table 326. HOSTIF_I2C_CONTROL_REG (address offset 0x000C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:8 | RESERVED | R | 0 | Reserved |
| 7:5 | I2C_REV_ID | R/W | 0 | Die Revision of Device ID |
| 4 | I2C_DEVID_ENABLE | R/W | 0 | 1 - Enable Device ID defined in the I2C standard |
| 3 | I2C_RESET_ENABLE | R/W | 0 | 1 - Enable Soft Reset sequence defined in the I2C standard |
| 2 | I2C_HS_ENABLE | R/W | 0 | 1 - Enable High-speed mode |
| 1:0 | I2C_ADDR | R/W | 0 | Set two LSBs of the I2C address |

### HOSTIF_SPI_CONTROL_REG

This register is used to control the SPI host interface.

**Table 327. HOSTIF_SPI_CONTROL_REG (address offset 0x0010)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:2 | RESERVED | R | 0 | Reserved |
| 1 | SPI_CPHA | R/W | 0 | SPI clock phase |
| 0 | SPI_CPOL | R/W | 0 | SPI clock polarity |

### HOSTIF_HSU_CONTROL_REG

This register is used to control the High-speed UART host interface

**Table 328. HOSTIF_HSU_CONTROL_REG (address offset 0x0014)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:28 | RESERVED | R | 0 | Reserved |
| 27:21 | HSU_EOF_SIZE | R/W | 0x10 | EOF duration in bit number - 1 (1-122) |
| 20 | HSU_STORE_BR_BYTE | R/W | 0 | 1: the first 0 used to estimate baud rate is stored in memory |
| 19:18 | HSU_BR_ESTIMATOR_MODE | R/W | 0 | 00: Baud rate estimator inactive<br>01: Baud rate estimator active with automatic clock setting<br>10: Baud rate estimator active without clock setting<br>11: Reserved |
| 17:16 | HSU_WAKEUP_BYTES | R/W | 1 | Number of bytes lost during wakeup at RTS rising edge |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 15 | HSU_WAKEUP_STANDBY | R/W | 0 | 1: simulate reception of bytes lost during wakeup from standby phase (number of bytes specified in HSU_WAKEUP_BYTES). Can only be used once |
| 14 | HSU_STOPBIT | R/W | 0 | 0: 1 stop bit<br>1: 2 stop bits |
| 13 | HSU_TX_DIVIDER | R/W | 0 | 0: use HSU_RX_DIVIDER<br>1: use HSU_RX_DIVIDER+1 |
| 12:0 | HSU_RX_DIVIDER | R/W | 0 | Clock divider for RX sampling |

### HOSTIF_HSU_CONTROL_REG

This register is used for the HSU Sample Clock.

**Table 329. HOSTIF_HSU_SAMPLE_REG (address offset 0x0018)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:22 | RESERVED | R | 0 | Reserved |
| 21:11 | HSU_TX_CLK_CORRECT | R/W | 0 | Used to correct clock division. If TX_CLK_CORRECT[i]=1, then duration of bit[i] will be extended of one clock cycle |
| 10:0 | HSU_RX_CLK_CORRECT | R/W | 0 | Used to correct clock division. If RX_CLK_CORRECT[i]=1, then duration of bit[i] will be extended of one clock cycle |

### HOSTIF_HSU_CONTROL_REG

This register is used for the HSU estimated clock dividers.

**Table 330. HOSTIF_HSU_EST_CLOCK_DIVIDER_REG (address offset 0x001C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:14 | RESERVED | R | 0 | Reserved |
| 13 | HSU_EST_TX_DIVIDER | R | 0 | Estimated clock divider for TX sampling:<br>0: use HSU_EST_RX_DIVIDER<br>1: use HSU_EST_RX_DIVIDER + 1 |
| 12:0 | HSU_EST_RX_DIVIDER | R | 0 | Estimated clock divider for RX sampling |

### HOSTIF_HSU_EST_CLOCK_CORRECT_REG

This register is used for the HSU estimated clock correction.

**Table 331. HOSTIF_HSU_EST_CLOCK_CORRECT_REG (address offset 0x0020)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:20 | RESERVED | R | 0 | Reserved |
| 21:11 | HSU_EST_TX_CLK_CORRECT | R | 0 | Estimated value for HSU_TX_CLK_CORRECT |
| 10:0 | HSU_EST_RX_CLK_CORRECT | R | 0 | Estimated value for HSU_RX_CLK_CORRECT |

### HOSTIF_BUFFER_RX0_CFG_REG

This register is used to configure the RX buffer 0

**Table 332. HOSTIF_BUFFER_RX0_CFG_REG (address offset 0x0038)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R | 0 | Reserved |
| 28 | RX0_BUFFER_DISABLE [1] | R/W | 0 | 1 - buffer is disabled |
| 27:26 | RX0_HEADER_OFFSET [1][5][6] | R/W | 0x02 | Number of padding bytes to add before writing frame header to first word of RX buffer 0. Not applicable to Native Mode. |
| 25 | RX0_SHORT_FRAME_BUFFER [1][2][3][5] | R/W | 0 | 1 - buffer assigned to short frames<br>0 - buffer assigned to non-short frames<br>Not applicable to Native Mode. |
| 24:14 | RX0_MAX_SIZE [1][2][4][5] | R/W | 0 | Maximum size (bytes) of RX buffer 0 |
| 13:0 | RX0_START_ADDR [1][5] | R/W | 0 | Word start address of RX buffer 0. Bits [1:0] are unused. |

[1] Any change to this register, is only taken into account if the buffer is not in use (RX0_BUFFER_LOCK = 0). However, the register itself is updated.

[3] The frame length of a short frame is defined in field SHORT_FRAME_LEN of HOSTIF_CONTROL_REG

[4] Value must be greater than or equal to RX0_HEADER_OFFSET + + header + largest payload expected (rounded up to the next word).

[5] Attempting to change this field when the buffer is in use (RX0_BUFFER_LOCK = 1) will generate a System Error.

[6] Only values 0,1 or 2 are permitted

### HOSTIF_BUFFER_RX1_CFG_REG

This register is used to configure the RX buffer 1

**Table 333. HOSTIF_BUFFER_RX1_CFG_REG (address offset 0x003C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R | 0 | Reserved |
| 28 | RX1_BUFFER_DISABLE [1] | R/W | 0 | 1 - buffer is disabled |
| 27:26 | RX1_HEADER_OFFSET [1][5][6] | R/W | 0x02 | Number of padding bytes to add before writing frame header to first word of RX buffer 1.Not applicable to Native Mode. |
| 25 | RX1_SHORT_FRAME_BUFFER [1][3][5] | R/W | 0 | 1 - buffer assigned to short frames<br>0 - buffer assigned to non-short frames<br>Not applicable to Native Mode. |
| 24:14 | RX1_MAX_SIZE [1][4][5] | R/W | 0 | Maximum size (bytes) of RX buffer 1 |
| 13:0 | RX1_START_ADDR [1][5] | R/W | 0 | Word start address of RX buffer 1. Bits [1:0] are unused. |

[1] Any change to this register, is only taken into account if the buffer is not in use (RX1_BUFFER_LOCK = 0). However, the register itself is updated.

[3] The frame length of a short frame is defined in field SHORT_FRAME_LEN of HOSTIF_CONTROL_REG

[4] Value must be greater than or equal to RX1_HEADER_OFFSET + + header + largest payload expected (rounded up to the next word).

[5] Attempting to change this field when the buffer is in use (RX1_BUFFER_LOCK = 1) will generate a System Error.

[6] Only values 0,1 or 2 are permitted

### HOSTIF_BUFFER_RX2_CFG_REG

This register is used to configure the RX buffer 2

**Table 334. HOSTIF_BUFFER_RX2_CFG_REG (address offset 0x0040)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R | 0 | Reserved |
| 28 | RX2_BUFFER_DISABLE [1] | R/W | 0 | 1 - buffer is disabled |
| 27:26 | RX2_HEADER_OFFSET [1][5][6] | R/W | 0x02 | Number of padding bytes to add before writing frame header to first word of RX buffer 2. Not applicable to Native Mode. |
| 25 | RX2_SHORT_FRAME_BUFFER [1][3][5] | R/W | 0 | 1 - buffer assigned to short frames<br>0 - buffer assigned to non-short frames<br>Not applicable to Native Mode. |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 24:14 | RX2_MAX_SIZE [1][4][5] | R/W | 0 | Maximum size (bytes) of RX buffer 2 |
| 13:0 | RX2_START_ADDR [1][5] | R/W | 0 | Word start address of RX buffer 2. Bits [1:0] are unused. |

[1] Any change to this register, is only taken into account if the buffer is not in use (RX<n>_BUFFER_LOCK = 0). However, the register itself is updated.

[3] The frame length of a short frame is defined in field SHORT_FRAME_LEN of HOSTIF_CONTROL_REG

[4] Value must be greater than or equal to RX<n>_HEADER_OFFSET + + header + largest payload expected (rounded up to the next word).

[5] Attempting to change this field when the buffer is in use (RX<n>_BUFFER_LOCK = 1) will generate a System Error.

[6] Only values 0,1 or 2 are permitted

### HOSTIF_BUFFER_RX3_CFG_REG

This register is used to configure the RX buffer 3.

**Table 335. HOSTIF_BUFFER_RX3_CFG_REG (address offset 0x0044)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:29 | RESERVED | R | 0 | Reserved |
| 28 | RX3_BUFFER_DISABLE [1] | R/W | 0 | 1 - buffer is disabled |
| 27:26 | RX3_HEADER_OFFSET [1][5][6] | R/W | 0x02 | Number of padding bytes to add before writing frame header to first word of RX buffer 3.Not applicable to Native Mode. |
| 25 | RX3_SHORT_FRAME_BUFFER [1][3][5] | R/W | 0 | 1 - buffer assigned to short frames<br>0 - buffer assigned to non-short frames<br>Not applicable to Native Mode. |
| 24:14 | RX3_MAX_SIZE [1][4][5] | R/W | 0 | Maximum size (bytes) of RX buffer 3 |
| 13:0 | RX3_START_ADDR [1][5] | R/W | 0 | Word start address of RX buffer 3. Bits [1:0] are unused. |

[1] Any change to this register, is only taken into account if the buffer is not in use (RX3_BUFFER_LOCK = 0). However, the register itself is updated.

[3] The frame length of a short frame is defined in field SHORT_FRAME_LEN of HOSTIF_CONTROL_REG

[4] Value must be greater than or equal to RX3_HEADER_OFFSET + + header + largest payload expected (rounded up to the next word).

[5] Attempting to change this field when the buffer is in use (RX3_BUFFER_LOCK = 1) will generate a System Error.

[6] Only values 0,1 or 2 are permitted

### HOSTIF_BUFFER_TX_CFG_REG

This register is used to configure the TX buffer

**Table 336. HOSTIF_BUFFER_TX_CFG_REG (address offset 0x0048)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:17 | RESERVED | R | 0 | Reserved |
| 16 | TX_EMPTY_PAYLOAD_ENABLE[1] | R/W | 0 | 1 - send empty payload packets (header and CRC only).<br>Not applicable to Native Mode. |
| 15:14 | TX_HEADER_OFFSET [1][2] | R/W | 0x02 | Number of bytes to skip in first word of buffer before sending to Host.<br>Not applicable to Native Mode. |
| 13:0 | TX_START_ADDR [1] | R/W | 0 | Word start address of TX buffer. Bits [1:0] are unused. |

[1] Any change to this register, is only taken into account if the buffer is not in use (TX_BUFFER_LOCK = 0). However, the register itself is updated.

[2] Only values 0,1 or 2 are permitted

### HOSTIF_BUFFER_RX0_LEN_REG

This register is used to indicate the number of bytes stored in RX buffer 0

**HOSTIF_BUFFER_RX0_LEN_REG (address offset 0x004C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:13 | RESERVED | R | 0 | Reserved |
| 12 | RX0_PEC_OK | R | 0 | 1 - Last byte received matched computed PEC. Only valid if RX0_PEC_RECEIVED is 1 |
| 11 | RX0_PEC_RECEIVED | R | 0 | 1 - Last received was a PEC. 0- hardware could not detected if last received byte was a PEC, payload analysis required. |
| 10:0 | RX0_LENGTH | R | 0 | Number of bytes received in buffer RX0 (active when HOSTIF_CONTROL_REG.NCI_LENGTH_MODE=1) |

### HOSTIF_BUFFER_RX1_LEN_REG

This register is used to indicate the number of bytes stored in RX buffer 1

**Table 337. HOSTIF_BUFFER_RX1_LEN_REG (address offset 0x0050)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:13 | RESERVED | R | 0 | Reserved |
| 12 | RX1_PEC_OK | R | 0 | 1 - Last byte received matched computed PEC. Only valid if RX1_PEC_RECEIVED is 1 |
| 11 | RX1_PEC_RECEIVED | R | 0 | 1 - Last received was a PEC. 0- hardware could not detected if last received byte was a PEC, payload analysis required. |
| 10:0 | RX1_LENGTH | R | 0 | Number of bytes received in buffer RX1 (active when HOSTIF_CONTROL_REG.NCI_LENGTH_MODE=1) |

### HOSTIF_BUFFER_RX2_LEN_REG

This register is used to indicate the number of bytes stored in RX buffer 2

**Table 338. HOSTIF_BUFFER_RX2_LEN_REG (address offset 0x0054)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:13 | RESERVED | R | 0 | Reserved |
| 12 | RX2_PEC_OK | R | 0 | 1 - Last byte received matched computed PEC. Only valid if RX2_PEC_RECEIVED is 1 |
| 11 | RX2_PEC_RECEIVED | R | 0 | 1 - Last received was a PEC. 0- hardware could not detected if last received byte was a PEC, payload analysis required. |
| 10:0 | RX2_LENGTH | R | 0 | Number of bytes received in buffer RX2 (active when HOSTIF_CONTROL_REG.NCI_LENGTH_MODE=1) |

### HOSTIF_BUFFER_RX3_LEN_REG

This register is used to indicate the number of bytes stored in RX buffer 3

**Table 339. HOSTIF_BUFFER_RX3_LEN_REG (address offset 0x0058)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:13 | RESERVED | R | 0 | Reserved |
| 12 | RX3_PEC_OK | R | 0 | 1 - Last byte received matched computed PEC. Only valid if RX3_PEC_RECEIVED is 1 |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 11 | RX3_PEC_RECEIVED | R | 0 | 1 - Last received was a PEC. 0- hardware could not detected if last received byte was a PEC, payload analysis required. |
| 10:0 | RX3_LENGTH | R | 0 | Number of bytes received in buffer RX3 (active when HOSTIF_CONTROL_REG.NCI_LENGTH_MODE=1) |

### HOSTIF_BUFFER_TX_LEN_REG

This register is used to indicate the number of bytes stored in the TX buffer

**Table 340. HOSTIF_BUFFER_TX_LEN_REG (address offset 0x005C)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:11 | RESERVED | R | 0 | Reserved |
| 10:0 | TX_LENGTH | R/W | 0 | Number of bytes to send from buffer TX (active when HOSTIF_CONTROL_REG.NCI_LENGTH_MODE=1) |

### HOSTIF_TIC_TIMEOUT_REG

This register is used to configure the inter-character timeout.

**Table 341. HOSTIF_TIC_TIMEOUT_REG (address offset 0x0060)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | R | 0 | Reserved |
| 15:0 | TX_TIMEOUT_VALUE | R/W | 0 | Inter-character timeout in 3.6us steps (max delay 200ms). If set to 0, this feature is disabled. |

### HOSTIF_WATERLEVEL_REG

This register is to used indicate the water level.

**Table 342. HOSTIF_WATERLEVEL_REG (address offset 0x0064)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:11 | RESERVED | R | 0 | Reserved |
| 10:0 | WATERLEVEL | R/W | 0 | Number of bytes received in incoming frame before triggering an interrupt (pre-empting EOR). If set to 0, this feature is disabled. |

### HOSTIF_SET_DATA_READY_REG

This register is used to set data ready flags for buffers.

**Table 343. HOSTIF_SET_DATA_READY_REG (address offset 0x0068)**

| Bit | Symbol | Access | Reset value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R | 0 | Reserved |
| 4 | SET_TX_DATA_READY [1] | W | 0 | Set TX_DATA_READY |
| 3 | SET_RX3_DATA_READY[2] | W | 0 | Set RX3_DATA_READY |
| 2 | SET_RX2_DATA_READY[2] | W | 0 | Set RX2_DATA_READY |
| 1 | SET_RX1_DATA_READY[2] | W | 0 | Set RX1_DATA_READY |
| 0 | SET_RX0_DATA_READY [2] | W | 0 | Set RX0_DATA_READY |

[1] Setting this bit will only cause bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY to be set if the buffer is not in use by the buffer manager (HOSTIF_STATUS_REG.TX_BUFFER_LOCK = 0).

[2] Setting this bit will only cause bit HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY to be set if the buffer is not in use by the buffer manager (HOSTIF_STATUS_REG.RX<n>_BUFFER_LOCK = 0).

### HOSTIF_CLR_DATA_READY_REG

This register is used to clear data ready flags for buffers.

**Table 344. HOSTIF_CLR_DATA_READY_REG (address offset 0x006C)**

| Bit | Symbol | Access | Reset value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R | 0 | Reserved |
| 4 | CLR_TX_DATA_READY [1] | W | 0 | Clear TX_DATA_READY |
| 3 | CLR_RX3_DATA_READY[2] | W | 0 | Clear RX3_DATA_READY |

| Bit | Symbol | Access | Reset value | Description |
|---|---|---|---|---|
| 2 | CLR_RX2_DATA_READY[2] | W | 0 | Clear RX2_DATA_READY |
| 1 | CLR_RX1_DATA_READY[2] | W | 0 | Clear RX1_DATA_READY |
| 0 | CLR_RX0_DATA_READY[2] | W | 0 | Clear RX0_DATA_READY |

[1] Setting this bit will only cause bit HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY to be cleared if the buffer is not in use by the Buffer Manager (HOSTIF_STATUS_REG.TX_BUFFER_LOCK = 0).

[2] Setting this bit will only cause bit HOSTIF_DATA_READY_STATUS_REG.RX<n>_DATA_READY to be cleared if the buffer is not in use by the Buffer Manager (HOSTIF_STATUS_REG.RX<n>_BUFFER_LOCK = 0).

### HOSTIF_DATA_READY_STATUS_REG

This register is used to indicate the status of data ready flags for buffers.

**Table 345. HOSTIF_DATA_READY_STATUS_REG (address offset 0x0070)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:5 | RESERVED | R | 0 | Reserved |
| 4 | TX_DATA_READY [1][2] | R | 0 | Frame valid bit for TX buffer.<br>1 - buffer loaded with frame by FW to be sent to Host<br>0 - frame successfully sent to Host. |
| 3 | RX3_DATA_READY [1][3] | R | 0 | Frame valid bit for RX buffer 3.<br>1 - buffer contains an error-free frame received from Host<br>0 - frame has been processed by FW and buffer is free to receive a new frame. |
| 2 | RX2_DATA_READY [1][3] | R | 0 | Frame valid bit for RX buffer 2.<br>1 - buffer contains an error-free frame received from Host<br>0 - frame has been processed by FW and buffer is free to receive a new frame. |
| 1 | RX1_DATA_READY [1][3] | R | 0 | Frame valid bit for RX buffer 1.<br>1 - buffer contains an error-free frame received from Host<br>0 - frame has been processed by FW and buffer is free to receive a new frame. |
| 0 | RX0_DATA_READY [1][3] | R | 0 | Frame valid bit for RX buffer 0.<br>1 - buffer contains an error-free frame received from Host<br>0 - frame has been processed by FW and buffer is free to receive a new frame. |

[1] Can be overwritten by FW using either register HOSTIF_SET_DATA_READY_REG or register HOSTIF_CLR_DATA_READY_REG.

[2] The buffer manager can only clear this bit.

[3] The buffer manager can only set this bit.

### HOSTIF_DBG_RX_REG

This register is used for the debug received data.

**Table 346. HOSTIF_DBG_RX_REG (address offset 0x0074)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:5 | RX_REG | R | 0 | Contains byte the last received bytes to be written into memory as one word |

### HOSTIF_DBG_RX_REG

This register is used to indicate the debug receive address.

**Table 347. HOSTIF_DBG_RX_ADDR_REG (address offset 0x0078)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:16 | RESERVED | R | 0 | Reserved |
| 15:14 | WR_PTR | R | 0 | Pointer to next byte to write into RX_REG (next byte inside word to write to memory) |
| 13:0 | WR_ADDR | R | 0 | Next AHB write address |

### HOSTIF_INT_CLR_ENABLE_REG

This register is a collection of clear interrupt enable commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 348. HOSTIF_INT_CLR_ENABLE_REG (address offset 0x3FD8)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | W | 0 | Reserved |
| 26 | HSU_RX_FER_CLR_ENABLE | W | 0 | 1 - clear enable for HSU RX frame error interrupt0 - no effect |
| 25 | BUFFER_CFG_CHANGED_ERROR_CLR_ENABLE | W | 0 | 1 - clear enable for buffer config changed during use interrupt<br>0 - no effect |
| 24 | AHB_WR_SLOW_CLR_ENABLE | W | 0 | 1 - clear enable for slow AHB during write operation interrupt<br>0 - no effect |
| 23 | AHB_RD_SLOW_CLR_ENABLE | W | 0 | 1 - clear enable for slow AHB during read operation interrupt<br>0 - no effect |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 22 | AHB_ERROR_CLR_ENABLE | W | 0 | 1 - clear enable for ahb_error (hresp=1, oraddress overflow) interrupt<br>0 - no effect |
| 21 | WATERLEVEL_REACHED_CLR_ENABLE | W | 0 | 1 - clear enable for water level reached interrupt<br>0 - no effect |
| 20:17 | RX_BUFFER_OVERFLOW_CLR_ENABLE | W | 0 | 0001 - clear enable for max buffer size interrupt for RX buffer 0<br>0010 - clear enable for max buffer size interrupt for RX buffer 1<br>0100 - clear enable for max buffer size interrupt for RX buffer 2<br>1000 - clear enable for max buffer size interrupt for RX buffer 3<br>0000 - no effect |
| 16 | CRC_NOK_CLR_ENABLE | W | 0 | 1 - clear enable for data-link Layer CRC error interrupt<br>0 - no effect |
| 15 | TX_TIMEOUT_CLR_ENABLE | W | 0 | 1 - clear enable for inter-character timeout (TIC) exceeded on transmitted frame interrupt<br>0 - no effect |
| 14:11 | RX_FRAME_OVERFLOW_CLR_ENABLE | W | 0 | 0001 - clear enable for frame overflow interrupt for RX buffer 0<br>0010 - clear enable for frame overflow interrupt for RX buffer 1<br>0100 - clear enable for frame overflow interrupt for RX buffer 2<br>1000 - clear enable for frame overflow interrupt for RX buffer 3<br>0000 - no effect |
| 10:7 | RX_FRAME_UNDERFLOW_CLR_ENABLE | W | 0 | 0001 - clear enable for frame underflow interrupt for RX buffer 0<br>0010 - clear enable for frame underflow interrupt for RX buffer 1<br>0100 - clear enable for frame underflow interrupt for RX buffer 2<br>1000 - clear enable for frame underflow interrupt for RX buffer 3<br>0000 - no effect |
| 6 | TX_FRAME_NOT_AVAILABLE_CLR_ENABLE | W | 0 | 1 - clear enable for TX frame not available interrupt<br>0 - no effect |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 5 | RX_BUFFER_NOT_AVAILABLE_CLR_ENABLE | W | 0 | 1 - clear enable for no receive buffers available interrupt<br>0 - no effect |
| 4 | EOT_CLR_ENABLE | W | 0 | 1 - clear enable for EOT interrupt<br>0 - no effect |
| 3:0 | EOR_CLR_ENABLE | W | 0 | 1 - clear enable for End of Reception (EOR) in buffer N (0<=N<=3) interrupt<br>0001 - clear enable for EOR interrupt for RX buffer 0<br>0010 - clear enable for EOR interrupt for RX buffer 1<br>0100 - clear enable for EOR interrupt for RX buffer 2<br>1000 - clear enable for EOR interrupt for RX buffer 3<br>0000 - no effect |

[1] An interrupt event which has its clear enable bit set simply means that the external interrupt is not asserted. However, the event itself is still triggered. Thus, even if the CPU is using a polling mechanism instead of being interrupt-driven, the firmware must still ensure that the event is cleared by setting the associated bit in register HOSTIF_INT_CLR_STATUS_REG.

### HOSTIF_INT_SET_ENABLE_REG

This register is a collection of set interrupt enable commands. Writing 1 to this register does set the corresponding interrupt request enable flag. Writing 0 to this register has no effect.

**Table 349. HOSTIF_INT_SET_ENABLE_REG (address offset 0x3FDC)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | W | 0 | Reserved |
| 26 | HSU_RX_FER_SET_ENABLE | W | 0 | 1 - set enable for HSU RX frame error interrupt0 - no effect |
| 25 | BUFFER_CFG_CHANGED_ERROR_SET_ENABLE | W | 0 | 1 - set enable for buffer configuration changed during use interrupt<br>0 - no effect |
| 24 | AHB_WR_SLOW_SET_ENABLE | W | 0 | 1 - set enable for slow AHB during write operation interrupt<br>0 - no effect |
| 23 | AHB_RD_SLOW_SET_ENABLE | W | 0 | 1 - set enable for slow AHB during read operation interrupt<br>0 - no effect |
| 22 | AHB_ERROR_SET_ENABLE | W | 0 | 1 - set enable for ahb_error (hresp=1, oraddress overflow) interrupt<br>0 - no effect |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 21 | WATERLEVEL_REACHED_SET_ENABLE | W | 0 | 1 - set enable for water level reached interrupt<br>0 - no effect |
| 20:17 | RX_BUFFER_OVERFLOW_SET_ENABLE | W | 0 | 0001 - set enable for max buffer size interrupt for RX buffer 0<br>0010 - set enable for max buffer size interrupt for RX buffer 1<br>0100 - set enable for max buffer size interrupt for RX buffer 2<br>1000 - set enable for max buffer size interrupt for RX buffer 3<br>0000 - no effect |
| 16 | CRC_NOK_SET_ENABLE | W | 0 | 1 - set enable for data-link Layer CRC error interrupt<br>0 - no effect |
| 15 | TX_TIMEOUT_SET_ENABLE | W | 0 | 1 - set enable for inter-character timeout (TIC) exceeded on transmitted frame interrupt<br>0 - no effect |
| 14:11 | RX_FRAME_OVERFLOW_SET_ENABLE | W | 0 | 0001 - set enable for frame overflow interrupt for RX buffer 0<br>0010 - set enable for frame overflow interrupt for RX buffer 1<br>0100 - set enable for frame overflow interrupt for RX buffer 2<br>1000 - set enable for frame overflow interrupt for RX buffer 3<br>0000 - no effect |
| 10:7 | RX_FRAME_UNDERFLOW_SET_ENABLE | W | 0 | 0001 - set enable for frame underflow interrupt for RX buffer 0<br>0010 - set enable for frame underflow interrupt for RX buffer 1<br>0100 - set enable for frame underflow interrupt for RX buffer 2<br>1000 - set enable for frame underflow interrupt for RX buffer 3<br>0000 - no effect |
| 6 | TX_FRAME_NOT_AVAILABLE_SET_ENABLE | W | 0 | 1 - set enable for TX frame not available interrupt<br>0 - no effect |
| 5 | RX_BUFFER_NOT_AVAILABLE_SET_ENABLE | W | 0 | 1 - set enable for no receive buffers available interrupt<br>0 - no effect |
| 4 | EOT_SET_ENABLE | W | 0 | 1 - set enable for EOT interrupt<br>0 - no effect |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 3:0 | EOR_SET_ENABLE | W | 0 | 1 - set enable for End of Reception in buffer N (0<=N<=3) interrupt<br>0001 - set enable for EOR interrupt for RX buffer 0<br>0010 - set enable for EOR interrupt for RX buffer 1<br>0100 - set enable for EOR interrupt for RX buffer 2<br>1000 - set enable for EOR interrupt for RX buffer 3<br>0000 - no effect |

### HOSTIF_INT_STATUS_REG

This register is a collection of interrupt status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 350. HOSTIF_INT_STATUS_REG (address offset 0x3FE0)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | W | 0 | Reserved |
| 26 | HSU_RX_FER_STATUS | W | 0 | HSU RX frame error interrupt |
| 25 | BUFFER_CFG_CHANGED_ERROR_STATUS | W | 0 | Buffer configuration changed during use interrupt |
| 24 | AHB_WR_SLOW_STATUS | W | 0 | Slow AHB during write operation interrupt |
| 23 | AHB_RD_SLOW_STATUS | W | 0 | Slow AHB during read operation interrupt |
| 22 | AHB_ERROR_STATUS | W | 0 | Ahb_error (hresp=1, or address overflow) interrupt |
| 21 | WATERLEVEL_REACHED_STATUS | W | 0 | Water level reached interrupt status |
| 20:17 | RX_BUFFER_OVERFLOW_STATUS | W | 0 | 0001 - maximum buffer size exceeded interrupt status for RX buffer 0<br>0010 - maximum buffer size exceeded interrupt status for RX buffer 1<br>0100 - maximum buffer size exceeded interrupt status for RX buffer 2<br>1000 - maximum buffer size exceeded interrupt status for RX buffer 3 |
| 16 | CRC_NOK_STATUS | W | 0 | Data-link layer CRC error interrupt status |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 15 | TX_TIMEOUT_STATUS | W | 0 | Inter-character timeout (TIC) exceeded on transmitted frame interrupt status |
| 14:11 | RX_FRAME_OVERFLOW_STATUS | W | 0 | 0001 - Frame overflow interrupt status for RX buffer 0<br>0010 - Frame overflow interrupt status for RX buffer 1<br>0100 - Frame overflow interrupt status for RX buffer 2<br>1000 - Frame overflow interrupt status for RX buffer 3 |
| 10:7 | RX_FRAME_UNDERFLOW_STATUS | W | 0 | 0001 - Frame underflow interrupt status for RX buffer 0<br>0010 - Frame underflow interrupt status for RX buffer 1<br>0100 - Frame underflow interrupt status for RX buffer 2<br>1000 - Frame underflow interrupt status for RX buffer 3 |
| 6 | TX_FRAME_NOT_AVAILABLE_STATUS | W | 0 | HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY=0 when Host read request interrupt status |
| 5 | RX_BUFFER_NOT_AVAILABLE_STATUS | W | 0 | No receive buffers available interrupt status |
| 4 | EOT_STATUS | W | 0 | EOT interrupt status |
| 3:0 | EOR_STATUS | W | 0 | 0001 - EOR interrupt status for RX buffer 0<br>0010 - EOR interrupt status for RX buffer 1<br>0100 - EOR interrupt status for RX buffer 2<br>1000 - EOR interrupt status for RX buffer 3 |

### HOSTIF_INT_ENABLE_REG

This register is a collection of interrupt enable commands. Writing 1 to this register does set the corresponding interrupt request enable flag. Writing 0 to this register has no effect.

**Table 351. HOSTIF_INT_ENABLE_REG (address offset 0x3FE4)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | W | 0 | Reserved |
| 26 | HSU_RX_FER_ENABLE | W | 0 | HSU RX frame error interrupt enable |
| 25 | BUFFER_CFG_CHANGED_ERROR_ENABLE | W | 0 | Buffer configuration changed during use interrupt enable |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 24 | AHB_WR_SLOW_ENABLE | W | 0 | Slow AHB during write operation interrupt enable |
| 23 | AHB_RD_SLOW_ENABLE | W | 0 | Slow AHB during read operation interrupt enable |
| 22 | AHB_ERROR_ENABLE | W | 0 | Ahb_error (hresp=1, or address overflow) interrupt enable |
| 21 | WATERLEVEL_REACHED_ENABLE | W | 0 | Water level reached interrupt enable |
| 20:17 | RX_BUFFER_OVERFLOW_ENABLE | W | 0 | 0001 - Maximum buffer size exceeded interrupt enable for RX buffer 0<br>0010 - Maximum buffer size exceeded interrupt enable for RX buffer 1<br>0100 - Maximum buffer size exceeded interrupt enable for RX buffer 2<br>1000 - Maximum buffer size exceeded interrupt enable for RX buffer 3 |
| 16 | CRC_NOK_ENABLE | W | 0 | Data-link layer CRC error interrupt enable |
| 15 | TX_TIMEOUT_ENABLE | W | 0 | Inter-character timeout (TIC) exceeded on transmitted frame interrupt status |
| 14:11 | RX_FRAME_OVERFLOW_ENABLE | W | 0 | 0001 - Frame overflow interrupt enable for RX buffer 0<br>0010 - Frame overflow interrupt enable for RX buffer 1<br>0100 - Frame overflow interrupt enable for RX buffer 2<br>1000 - Frame overflow interrupt enable for RX buffer 3 |
| 10:7 | RX_FRAME_UNDERFLOW_ENABLE | W | 0 | 0001 - Frame underflow interrupt enable for RX buffer 0<br>0010 - Frame underflow interrupt enable for RX buffer 1<br>0100 - Frame underflow interrupt enable for RX buffer 2<br>1000 - Frame underflow interrupt enable for RX buffer 3 |
| 6 | TX_FRAME_NOT_AVAILABLE_ENABLE | W | 0 | HOSTIF_DATA_READY_STATUS_REG.TX_DATA_READY=0 when host read request interrupt enable |
| 5 | RX_BUFFER_NOT_AVAILABLE_ENABLE | W | 0 | No receive buffers available interrupt enable |
| 4 | EOT_ENABLE | W | 0 | EOT interrupt enable |
| 3:0 | EOR_ENABLE | W | 0 | 0001 - EOR interrupt enable for RX buffer 0<br>0010 - EOR interrupt enable for RX buffer 1 |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| | | | | 0100 - EOR interrupt enable for RX buffer 2<br>1000 - EOR interrupt enable for RX buffer 3 |

### HOSTIF_INT_CLR_STATUS_REG

This register is a collection of clear interrupt status commands. Writing 1 to this register does set the corresponding interrupt request enable flag. Writing 0 to this register has no effect.

**Table 352. HOSTIF_INT_CLR_STATUS_REG (address offset 0x3FE8)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | W | 0 | Reserved |
| 26 | HSU_RX_FER_CLR_STATUS | W | 0 | 1 - Clear HSU RX frame error interrupt status<br>0 - no effect |
| 25 | BUFFER_CFG_CHANGED_ERROR_CLR_STATUS | W | 0 | 1 - Clear buffer config changed during use interrupt status<br>0 - no effect |
| 24 | AHB_WR_SLOW_CLR_STATUS | W | 0 | 1 - Clear slow AHB during write operation interrupt status<br>0 - no effect |
| 23 | AHB_RD_SLOW_CLR_STATUS | W | 0 | 1 - Clear slow AHB during read operation interrupt status<br>0 - no effect |
| 22 | AHB_ERROR_CLR_STATUS | W | 0 | 1- Clear ahb_error (hresp=1, or address overflow) interrupt status<br>0 - no effect |
| 21 | WATERLEVEL_REACHED_CLR_STATUS | W | 0 | 1 - clear water level reached interrupt<br>0 - no effect |
| 20:17 | RX_BUFFER_OVERFLOW_CLR_STATUS | W | 0 | 0001 - clear max buffer size interrupt for RX buffer 0<br>0010 - clear max buffer size interrupt for RX buffer 1<br>0100 - clear max buffer size interrupt for RX buffer 2<br>1000 - clear max buffer size interrupt for RX buffer 30000 - no effect |
| 16 | CRC_NOK_CLR_STATUS | W | 0 | 1 - clear data-link Layer CRC error interrupt<br>0 - no effect |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 15 | TX_TIMEOUT_CLR_STATUS | W | 0 | 1 - clear inter-character timeout (TIC) exceeded on transmitted frame interrupt<br>0 - no effect |
| 14:11 | RX_FRAME_OVERFLOW_CLR_STATUS | W | 0 | 0001 - clear frame overflow interrupt for RX buffer 0<br>0010 - clear frame overflow interrupt for RX buffer 1<br>0100 - clear frame overflow interrupt for RX buffer 2<br>1000 - clear frame overflow interrupt for RX buffer 3<br>0000 - no effect |
| 10:7 | RX_FRAME_UNDERFLOW_CLR_STATUS | W | 0 | 0001 - clear frame underflow interrupt for RX buffer 0<br>0010 - clear frame underflow interrupt for RX buffer 1<br>0100 - clear frame underflow interrupt for RX buffer 2<br>1000 - clear frame underflow interrupt for RX buffer 3<br>0000 - no effect |
| 6 | TX_FRAME_NOT_AVAILABLE_CLR_STATUS | W | 0 | 1 - clear TX frame not available interrupt<br>0 - no effect |
| 5 | RX_BUFFER_NOT_AVAILABLE_CLR_STATUS | W | 0 | 1 - clear no receive buffers available interrupt<br>0 - no effect |
| 4 | EOT_CLR_STATUS | W | 0 | 1 - clear EOT interrupt<br>0 - no effect |
| 3:0 | EOR_CLR_STATUS | W | 0 | 0001 - clear EOR interrupt for RX buffer 0<br>0010 - clear EOR interrupt for RX buffer 1<br>0100 - clear EOR interrupt for RX buffer 2<br>1000 - clear EOR interrupt for RX buffer 3<br>0000 - no effect |

### HOSTIF_INT_SET_STATUS_REG

This register is a collection of Set Interrupt Status commands. Writing 1 to this register does set the corresponding Interrupt Request ENABLE flag. Writing 0 to this register has no effect.

**Table 353. HOSTIF_INT_SET_STATUS_REG (address offset 0x3FEC)**

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 31:27 | RESERVED | W | 0 | Reserved |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| 26 | HSU_RX_FER_SET_STATUS | W | 0 | 1 - Set HSU RX frame error interrupt status<br>0 - no effect |
| 25 | BUFFER_CFG_CHANGED_ERROR_SET_STATUS | W | 0 | 1 - Set buffer config changed during use interrupt status<br>0 - no effect |
| 24 | AHB_WR_SLOW_SET_STATUS | W | 0 | 1 - Set slow AHB during write operation interrupt status<br>0 - no effect |
| 23 | AHB_RD_SLOW_SET_STATUS | W | 0 | 1 - Set slow AHB during read operation interrupt status<br>0 - no effect |
| 22 | AHB_ERROR_SET_STATUS | W | 0 | 1- Set ahb_error (hresp=1, or address overflow) interrupt status<br>0 - no effect |
| 21 | WATERLEVEL_REACHED_SET_STATUS | W | 0 | 1 - set water level reached interrupt<br>0 - no effect |
| 20:17 | RX_BUFFER_OVERFLOW_SET_STATUS | W | 0 | 0001 - set max buffer size interrupt for RX buffer 0<br>0010 - set max buffer size interrupt for RX buffer 1<br>0100 - set max buffer size interrupt for RX buffer 2<br>1000 - set max buffer size interrupt for RX buffer 30000 - no effect |
| 16 | CRC_NOK_SET_STATUS | W | 0 | 1 - set data-link Layer CRC error interrupt<br>0 - no effect |
| 15 | TX_TIMEOUT_SET_STATUS | W | 0 | 1 - set inter-character timeout (TIC) exceeded on transmitted frame interrupt<br>0 - no effect |
| 14:11 | RX_FRAME_OVERFLOW_SET_STATUS | W | 0 | 0001 - set frame overflow interrupt for RX buffer 0<br>0010 - set frame overflow interrupt for RX buffer 1<br>0100 - set frame overflow interrupt for RX buffer 2<br>1000 - set frame overflow interrupt for RX buffer 3<br>0000 - no effect |
| 10:7 | RX_FRAME_UNDERFLOW_SET_STATUS | W | 0 | 0001 - set frame underflow interrupt for RX buffer 0<br>0010 - set frame underflow interrupt for RX buffer 1 |

| Bit | Symbol | Access | Reset Value | Description |
|---|---|---|---|---|
| | | | | 0100 - set frame underflow interrupt for RX buffer 2<br>1000 - set frame underflow interrupt for RX buffer 3<br>0000 - no effect |
| 6 | TX_FRAME_NOT_AVAILABLE_SET_STATUS | W | 0 | 1 - set TX frame not available interrupt<br>0 - no effect |
| 5 | RX_BUFFER_NOT_AVAILABLE_SET_STATUS | W | 0 | 1 - set no receive buffers available interrupt<br>0 - no effect |
| 4 | EOT_SET_STATUS | W | 0 | 1 - set EOT interrupt<br>0 - no effect |
| 3:0 | EOR_SET_STATUS | W | 0 | 0001 - set EOR interrupt for RX buffer 0<br>0010 - set EOR interrupt for RX buffer 1<br>0100 - set EOR interrupt for RX buffer 2<br>1000 - set EOR interrupt for RX buffer 3<br>0000 - no effect |

# 15. PN7462AU USB

The Universal Serial Bus (USB) is a four-wire bus that supports communication between a host and one or more (up to 127) peripherals. The host controller allocates the USB bandwidth to attached devices through a token-based protocol. The bus supports hot plugging and dynamic configuration of the devices. All transactions are initiated by the host controller.

## 15.1 USB features

- Fully compliant with USB 2.0 specification (full speed)
- Dedicated USB PLL available
- Supports 14 physical (7 logical) endpoints including one control endpoint
- Each non-control endpoint supports bulk, interrupt, or isochronous endpoint types.
- Single or double buffering allowed
- Support wake-up from Suspend mode on USB activity and remote wake-up
- Soft connect supported

## 15.2 General description

The host schedules transactions in 1 ms frames. Each frame contains a Start-Of-Frame (SOF) marker and transactions that transfer data to or from device endpoints.

There are four types of transfers defined for the endpoints. Control transfers are used to configure the device.

Interrupt transfers are used for periodic data transfer. Bulk transfers are used when the latency of transfer is not critical. Isochronous transfers have guaranteed delivery time but no error correction.

The USB device controller on the PN7462AU enables full-speed (12 Mbit/s) data exchange with a USB host controller.

The picture below shows the block diagram of the USB device controller.

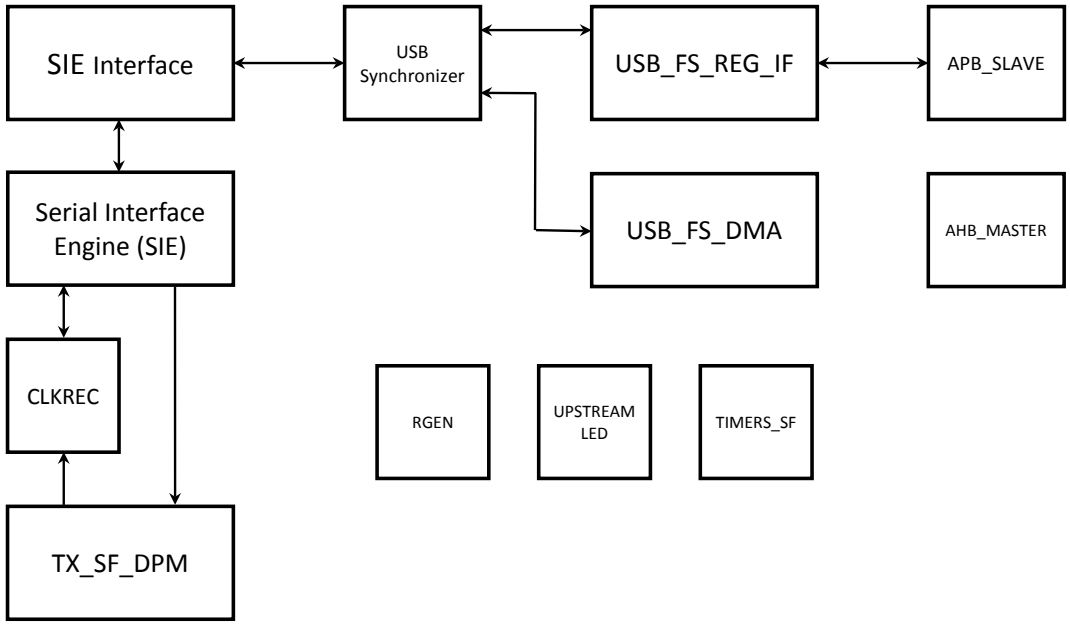



**Fig 65. USB block diagram**

### 15.2.1 USB software interface

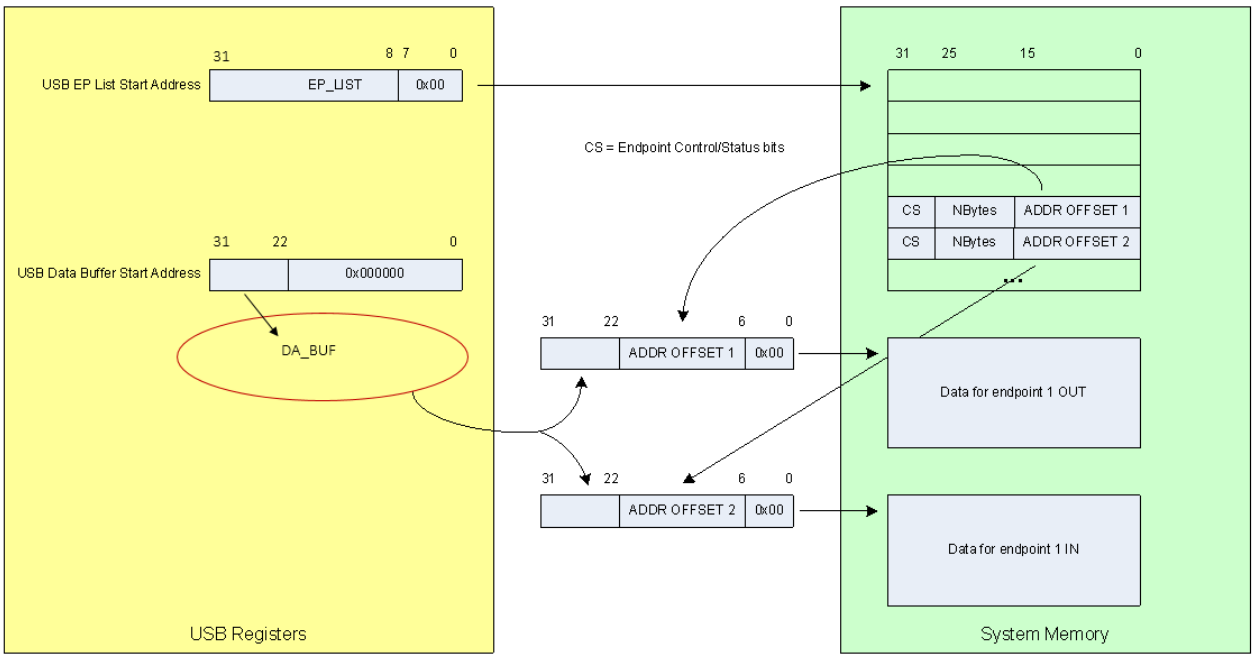
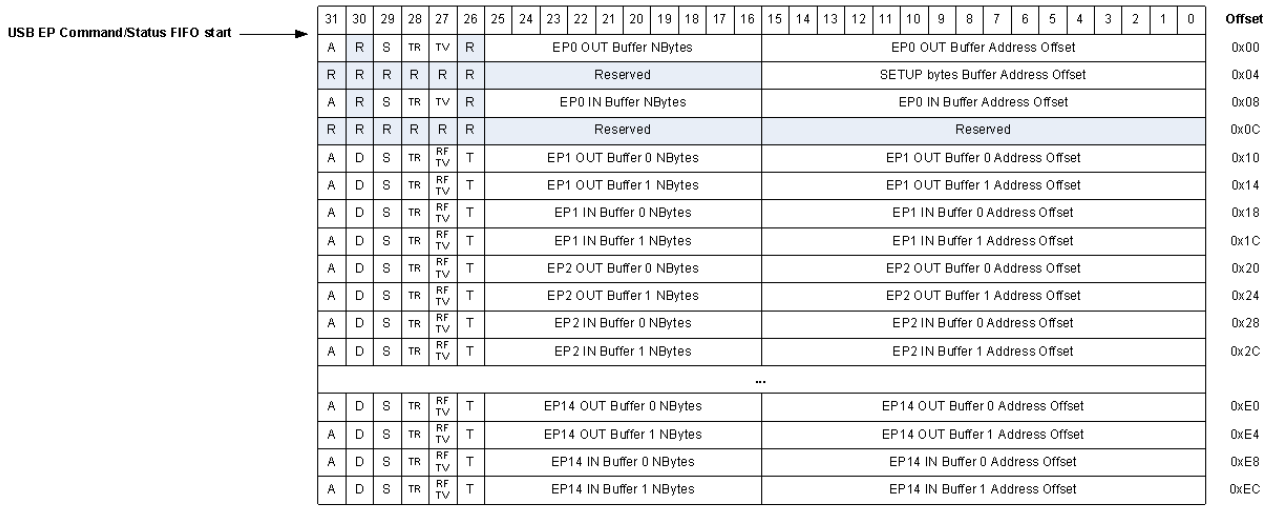



**Fig 66. USB software interface**

## 15.3 Functional description

### 15.3.1 Endpoint command/status list

The picture below gives an overview on how the endpoint list is organized in the memory. The USB EP Command/Status List start register points to the start of the list that contains all the endpoint information in the memory. The order of the endpoints is fixed as shown in the picture.




| 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 | Offset |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | R | S | TR | TV | R | EP0 OUT Buffer NBytes | | | | | | | | | | EP0 OUT Buffer Address Offset | | | | | | | | | | | | | | | | | 0x00 |
| R | R | R | R | R | R | Reserved | | | | | | | | | | SETUP bytes Buffer Address Offset | | | | | | | | | | | | | | | | | 0x04 |
| A | R | S | TR | TV | R | EP0 IN Buffer NBytes | | | | | | | | | | EP0 IN Buffer Address Offset | | | | | | | | | | | | | | | | | 0x08 |
| R | R | R | R | R | R | Reserved | | | | | | | | | | Reserved | | | | | | | | | | | | | | | | | 0x0C |
| A | D | S | TR | RF TV | T | EP1 OUT Buffer 0 NBytes | | | | | | | | | | EP1 OUT Buffer 0 Address Offset | | | | | | | | | | | | | | | | | 0x10 |
| A | D | S | TR | RF TV | T | EP1 OUT Buffer 1 NBytes | | | | | | | | | | EP1 OUT Buffer 1 Address Offset | | | | | | | | | | | | | | | | | 0x14 |
| A | D | S | TR | RF TV | T | EP1 IN Buffer 0 NBytes | | | | | | | | | | EP1 IN Buffer 0 Address Offset | | | | | | | | | | | | | | | | | 0x18 |
| A | D | S | TR | RF TV | T | EP1 IN Buffer 1 NBytes | | | | | | | | | | EP1 IN Buffer 1 Address Offset | | | | | | | | | | | | | | | | | 0x1C |
| A | D | S | TR | RF TV | T | EP2 OUT Buffer 0 NBytes | | | | | | | | | | EP2 OUT Buffer 0 Address Offset | | | | | | | | | | | | | | | | | 0x20 |
| A | D | S | TR | RF TV | T | EP2 OUT Buffer 1 NBytes | | | | | | | | | | EP2 OUT Buffer 1 Address Offset | | | | | | | | | | | | | | | | | 0x24 |
| A | D | S | TR | RF TV | T | EP2 IN Buffer 0 NBytes | | | | | | | | | | EP2 IN Buffer 0 Address Offset | | | | | | | | | | | | | | | | | 0x28 |
| A | D | S | TR | RF TV | T | EP2 IN Buffer 1 NBytes | | | | | | | | | | EP2 IN Buffer 1 Address Offset | | | | | | | | | | | | | | | | | 0x2C |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | D | S | TR | RF TV | T | EP14 OUT Buffer 0 NBytes | | | | | | | | | | EP14 OUT Buffer 0 Address Offset | | | | | | | | | | | | | | | | | 0xE0 |
| A | D | S | TR | RF TV | T | EP14 OUT Buffer 1 NBytes | | | | | | | | | | EP14 OUT Buffer 1 Address Offset | | | | | | | | | | | | | | | | | 0xE4 |
| A | D | S | TR | RF TV | T | EP14 IN Buffer 0 NBytes | | | | | | | | | | EP14 IN Buffer 0 Address Offset | | | | | | | | | | | | | | | | | 0xE8 |
| A | D | S | TR | RF TV | T | EP14 IN Buffer 1 NBytes | | | | | | | | | | EP14 IN Buffer 1 Address Offset | | | | | | | | | | | | | | | | | 0xEC |

**Fig 67. Endpoint command/status list**

**Table 354. Endpoint Commands**

| Symbol | Access | Description |
|---|---|---|
| A | R/W | Active<br>The buffer is enabled. hardware can use the buffer to store received OUT data or to transmit data on the IN endpoint.<br>Firmware can only set this bit to „1". As long as this bit is set to one, firmware is not allowed to update any of the values in this 32-bit word. In case firmware wants to deactivate the buffer, it must write a one to the corresponding “skip” bit in the USB Endpoint skip register. Hardware can only write this bit to zero. It will do this when it receives a short packet or when the NBytes field transitions to zero or when firmware has written a one to the “skip” bit. |
| D | R/W | Disabled<br>„0": The selected endpoint is enabled.<br>„1": The selected endpoint is disabled.<br>When a bus reset is received, firmware must set the disable bit of all endpoints to „1".<br>Firmware can only modify this bit when the active bit is zero. |
| S | R/W | Stall<br>„0": The selected endpoint is not stalled<br>„1": The selected endpoint is stalled<br>The Active bit has always higher priority than the Stall bit. This means that a Stall handshake is only sent when the active bit is zero.<br>Firmware can only modify this bit when the active bit is zero. |
| TR | R/W | Toggle reset<br>When firmware set this bit to one, the hardware will set the toggle value equal to the value indicated in the “toggle value” (TV) bit.<br>For the control endpoint zero, this is not needed to be used because the hardware resets the endpoint toggle to one for both directions when a setup token is received. |

| Symbol | Access | Description |
|---|---|---|
| | | For the other endpoints, the toggle can only be reset to zero when the endpoint is reset. |
| RF/ TV | R/W | Rate feedback mode / toggle value<br>For bulk endpoints and isochronous endpoints this bit is reserved and must be set to zero.<br>For the control endpoint zero this bit is used as the toggle value. When the toggle reset bit is set, the data toggle is updated with the value programmed in this bit.<br>When the endpoint is used as an interrupt endpoint, it can be set to the following values.<br>„0": Interrupt endpoint in „toggle mode"<br>„1": Interrupt endpoint in „rate feedback mode". This means that the data toggle is fixed to zero for all data packets.<br>When the interrupt endpoint is in „rate feedback mode", the TR bit must always be set to zero. |
| T | R/W | Endpoint type<br>„0": Generic endpoint. The endpoint is configured as a bulk or interrupt endpoint<br>„1": Isochronous endpoint |
| NBytes | R/W | For OUT endpoints this is the number of bytes that can be received in this buffer.<br>For IN endpoints this is the number of bytes that must be transmitted.<br>Hardware decrements this value with the packet size every time when a packet is successfully transferred.<br>Note: If a short packet is received on an OUT endpoint, the active bit will be cleared and the NBytes value indicates the remaining buffer space that is not used. Firmware calculates the received number of bytes by subtracting the remaining NBytes from the programmed value. |
| Address Offset | | Bits 21 down to 6 of the buffer start address.<br>This address offset is updated by hardware after each successful reception/transmission of a packet. Hardware increments the original value with the rounded up integer value when the packet size is divided by 64. |

Note: When receiving a SETUP token for endpoint zero, the hardware will only read the SETUP bytes buffer address offset to know where it has to store the received SETUP bytes. Hardware will ignore all other fields. In case the SETUP stage contains more than 8 bytes, it will only write the first 8 bytes to memory. A USB compliant host must never send more than 8 bytes during the SETUP stage.

### 15.3.2 Control endpoint zero

The flow charts in this section indicate how the firmware can handle the different endpoints.

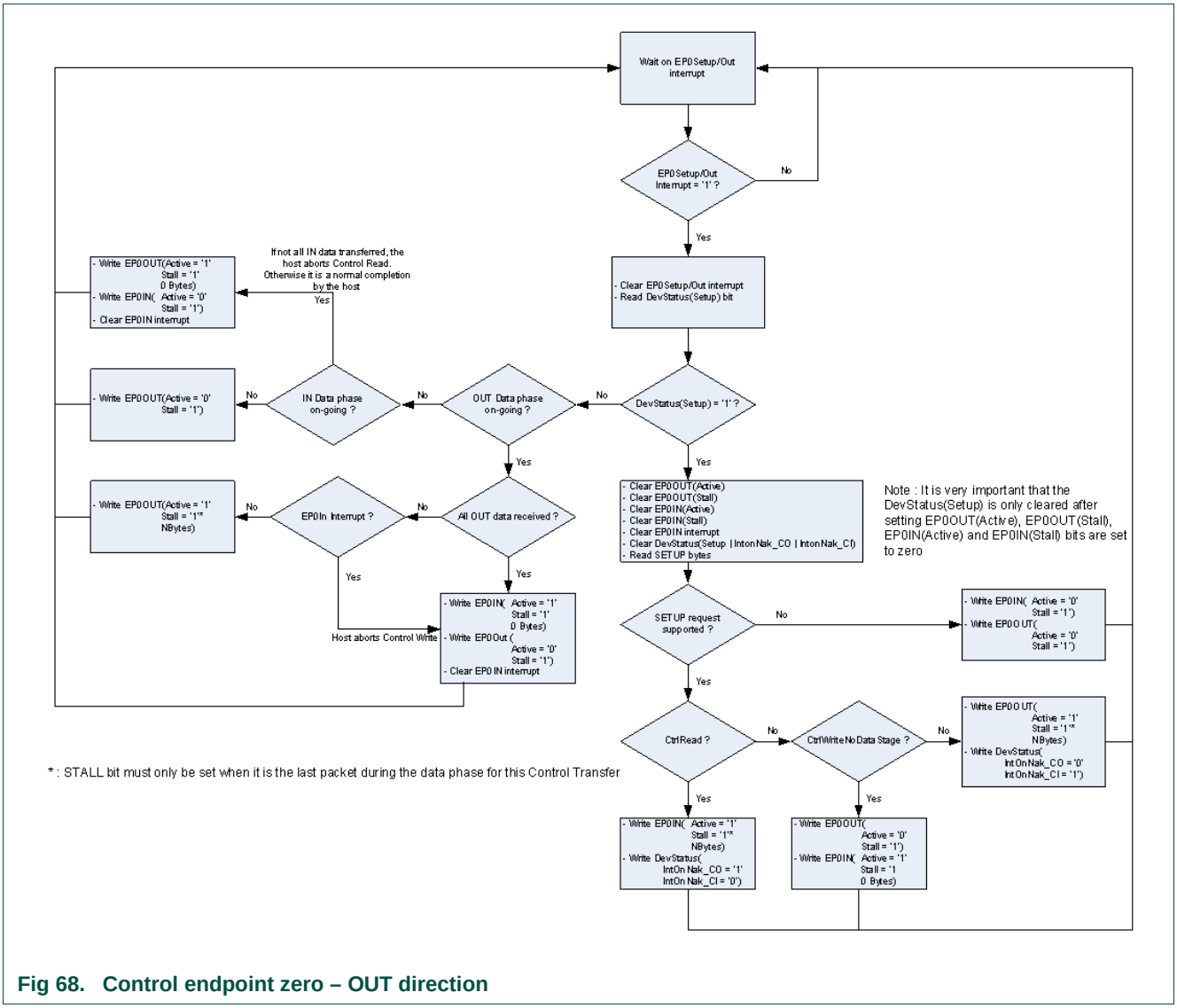



**Fig 68. Control endpoint zero – OUT direction**

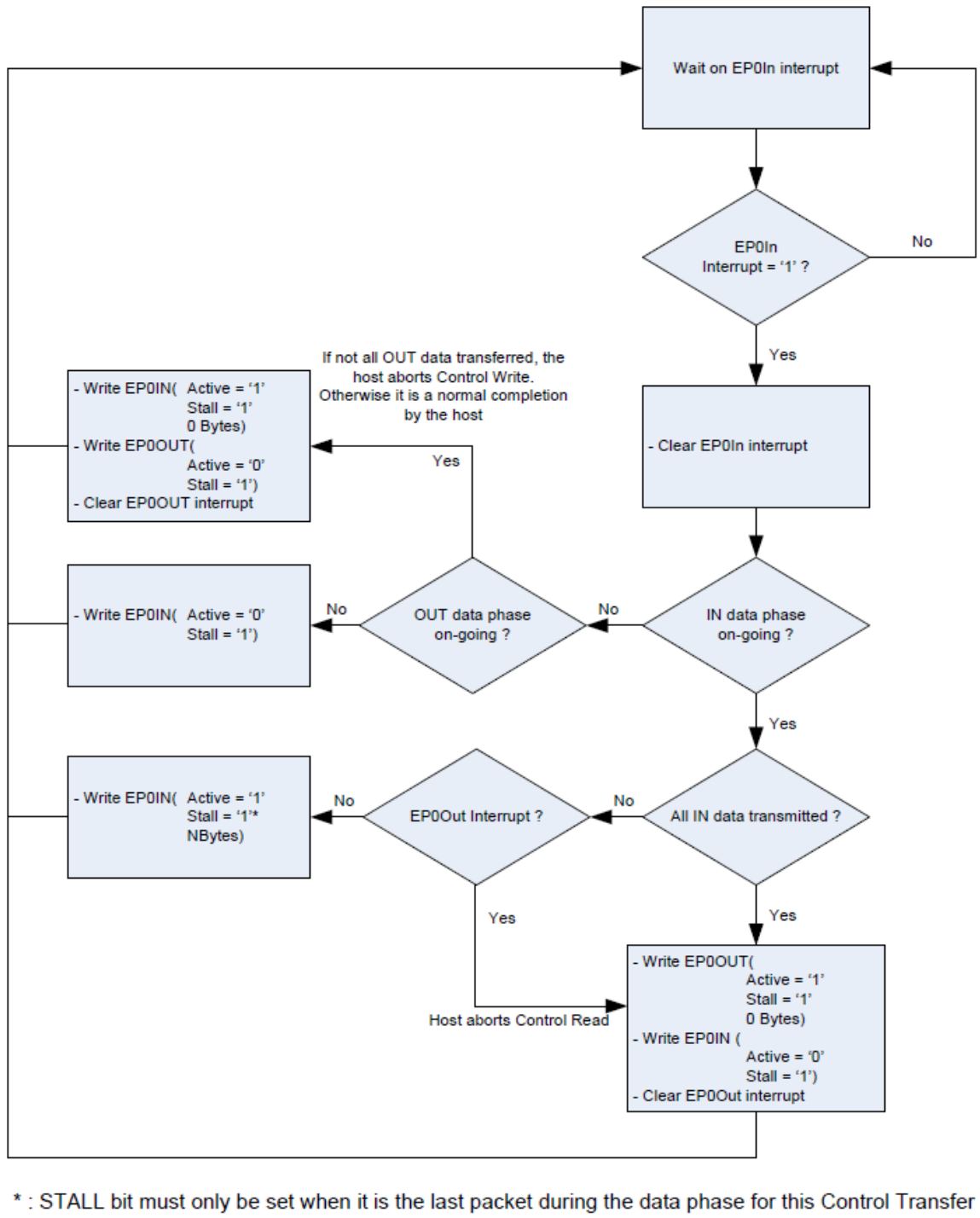



**Fig 69. Control endpoint 0 – IN direction**

### 15.3.3 Generic endpoint

#### 15.3.3.1 Generic endpoint – single buffering

To enable single buffering, firmware must set the corresponding “USB EP Buffer Config” bit to zero. In the “USB EP Buffer in use” register, firmware can indicate which buffer is used in this case.

When firmware wants to transfer data, it programs the different bits in the endpoint command/status entry and sets the active bits. Hardware will transmit/receive multiple packets for this endpoint until the NBytes value is equal to zero. When NBytes goes to zero, hardware clears the active bit and sets the corresponding interrupt status bit.

Firmware must wait until hardware has cleared the active bit to change some of the command/status bits. This prevents that hardware will overwrite a new value programmed by firmware with some old values that were still cached.

If firmware wants to disable the active bit before the hardware has finished handling complete buffer, it can do this by setting the corresponding endpoint skip bit in USB endpoint skip register.

#### 15.3.3.2 Generic endpoint – double buffering

To enable double buffering, firmware must set the corresponding “USB EP Buffer Config” bit to one. The “USB EP Buffer in use” register indicates which buffer will be used by hardware when the next token is received.

When hardware clears the active bit of the current buffer in use, it will switch the buffer in use. Firmware can also force hardware to use a certain buffer by writing to the “USB EP Buffer in use” bit.

### 15.3.4 Special cases

#### 15.3.4.1 Usage of the Active bit

The use of active bit is a different between OUT and IN endpoints.

When data must be received for the OUT endpoint, the firmware will set the Active bit to one and program the NBytes field to the maximum number of bytes it can receive.

When data must be transmitted for an IN endpoint, the firmware sets the Active bit to one and programs the NBytes field to the number of bytes that must be transmitted.

#### 15.3.4.2 Generation of a STALL handshake

Special care must be taken in programming the endpoint to send a STALL handshake. A STALL handshake is only sent in the following situations:

- The endpoint is enabled (Disabled bit = “0")
- The active bit of the endpoint is set to zero (No packet needs to be received/transmitted for that endpoint)

#### 15.3.4.3 Clear feature (Endpoint HALt)

When a non-control endpoint has returned a STALL handshake, the host will send a Clear Feature (Endpoint HALt) for that endpoint. When the device receives this request, the endpoint must be unstalled and the toggle bit for that endpoint must be reset back to zero. In order to do that the firmware must program the following items for the endpoint that is indicated.

If the endpoint is used in single buffer mode, program the following:

- Set STALL bit (S) to zero
- Set toggle reset bit (TR) to one and set toggle value bit (TV) to zero

If the endpoint is used in double buffer mode, program the following:

- Set the STALL bit of buffer 0 and buffer 1 to 0.
- Read the buffer in use bit for this endpoint
- Set the toggle reset bit (TR) to 1 and set the toggle value bit (TV) to 0 for the buffer indicated by the buffer in use bit.

#### 15.3.4.4 Set configuration

When a set configuration request is received with a configuration value different from zero, the device firmware must enable all endpoints that will be used in this configuration and reset all the toggle values. To do so, it must generate the procedure explained in clear feature (endpoint HALT) for every endpoint that will be used in this configuration.

For all endpoints that are not used in this configuration, it must set the disabled bit (D) to one.

#### 15.3.4.5 Suspend and resume

The USB suspend mode has been introduced in Section 8.2. The USB protocol enforces power management by the USB device. This becomes even more important if the device draws power from the bus (bus-powered device). The following constraints should be met by the bus-powered device:

- A device in the non-configured state should draw only a maximum of 100mA from the USB bus.
- A configured device can draw only up to what is specified in the Max Power field of the configuration descriptor. The maximum value is 500 mA
- A suspended device should draw only a maximum of 500 uA.

A device will go into the suspend mode if there is no activity on the USB bus for more than 3 ms. A suspended device wakes up, if there is transmission from the host (host-initiated wake up). The USB interface also supports software initiated remote wake up. To initiate this, software on the device must enable all clocks and clear the suspend bit. This will cause the hardware to generate a remote wake up signal upstream.

The assertion of the output signal indicates that there was no activity on the USB bus for the last 3 ms. At this time an interrupt is sent to the processor on which the software can start preparing the device for suspend.

If there is no activity for the next 2 ms, the signal indicating need for the clock will go low. This means that the main clock can be switched off. When activity is detected on the USB bus, USB suspend is deactivated and the signal indicating signal indicating need for the clock is activated.

## 15.4 Register overview and functions

This section gives an overview of the register set used to control the hardware functionalities of the USB full speed device controller.

### 15.4.1 Register overview

**Table 355. Register overview (base address 0x4002 8000)**

| Address offset | Register name | Access | Description |
|---|---|---|---|
| 0x00 | USB device Command/Status register | R/W (C) | This register contains all the fields to control the behavior of the USB device |
| 0x04 | USB Info register | RO | This register contains the frame number of the last received SOF, the ChipID and the error code |
| 0x08 | USB Endpoint List start address | R/W | This register contains the start address of the endpoint list that are stored in memory. |
| 0x0C | USB Data Buffer start address | R/W | This register contains the start address of the endpoint data buffers in memory |
| 0x10 | USB Link Power Management | R/W | This register contains the fields for the link power management support |
| 0x14 | USB EP skip | R/W | This register is used to indicate to hardware that it has to deactivate the corresponding endpoint (set active bit to zero) |
| 0x18 | USB EP Buffer in use | R/W | This bit is used for double buffering. It indicates which buffer is in-use for each endpoint |
| 0x1C | USB EP Buffer configuration register | R/W | This bit indicates if the endpoint has single buffering or double buffering |
| 0x20 | USB Interrupt status register | R/W | This register contains the status bits of the different interrupts |
| 0x24 | USB Interrupt enable register | R/W | This register contains the enable bits of the different interrupts. If this bit is set and the corresponding interrupt status bit is set a hardware interrupt is generated |
| 0x28 | USB Set Interrupt status register | R/W | If ‘1’ is written to one of the bits of this register, the corresponding interrupt status bit is set to one. When this register is read, it returns the same value as the USB Interrupt status register |
| 0x2C | USB Interrupt routing register | R/W | Each interrupt bit has a corresponding interrupt routing bit. If the interrupt routing bit is set to zero, a hardware interrupt will be generated on the IRQ line if both the corresponding interrupt status and interrupt enable bits are set.<br>If the interrupt routing bit is set to one, a hardware interrupt is generated on the FIQ line if both the corresponding interrupt status and interrupt enable bits are set |
| 0x30 | USB configuration | RO | This contains the configuration values as specified in section 6 (R&D document) |
| 0x34 | USB EP toggle | RO | This debug register is used to indicate the current data toggle value of the corresponding endpoint |

### 15.4.2 Register description

The following chapter describes the register on a bit level.

#### 15.4.2.1 USB device command/ Status register

**Table 356. USB device command/ Status register (address offset = 0x00)**

*Reset value: 0x00000800*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 6:0 | DEV_ADDR | | USB device address. After bus reset, the address is reset to 0x00. If the enable bit is set, the device will respond on packets for function address DEV_ADDR. When receiving a SetAddress Control Request from the USB host, firmware must program the new address before completing the status phase of the SetAddress control request. | 0 | R/W |
| 7 | DEV_EN | | USB device enable. If this bit is set, the hardware will start responding on packets for function address DEV_ADDR | 0 | R/W |
| 8 | SETUP | | SETUP token received. If a SETUP token is received and acknowledged by the device, this bit is set. As long as this bit is set all received IN and OUT tokens will be NAKed by hardware. SW must clear this bit by writing a one. If this bit is zero, hardware will handle the tokens to the CTRL EP0 as indicated by the CTRL EP0 IN and OUT data information programmed by SW. | 0 | R/W/C |
| 9 | PLL_ON | | Always PLL Clock on: | 0 | R/W |
| | | 0 | USB_NeedClk functional | | |
| | | 1 | USB_NeedClk always „1". Clock will not be stopped in case of suspend | | |
| 10 | RESERVED | | Reserved | 0 | RO |
| 11 | LPM_SUP | | LPM supported: | 1 | R/W |
| | | 0 | LPM not supported. hardware returns no handshake when receiving an LPM token | | |
| | | 1 | LPM supported | | |
| 12 | IntOnNAK_AO | | Interrupt on NAK for interrupt and bulk OUT EP | 0 | R/W |
| | | 0 | Only acknowledged packets generate an interrupt | | |
| | | 1 | Both acknowledged and NAKed packets generate interrupts | | |
| 13 | IntOnNAK_AI | | Interrupt on NAK for interrupt and bulk IN EP | 0 | R/W |
| | | 0 | Only acknowledged packets generate an interrupt | | |
| | | 1 | Both acknowledged and NAKed packets generate interrupts | | |
| 14 | IntOnNAK_CO | | Interrupt on NAK for control OUT EP | 0 | R/W |
| | | 0 | Only acknowledged packets generate an interrupt | | |
| | | 1 | Both acknowledged and NAKed packets generate interrupts | | |
| 15 | IntOnNAK_CI | | Interrupt on NAK for control IN EP | 0 | R/W |
| | | 0 | Only acknowledged packets generate an interrupt | | |
| | | 1 | Both acknowledged and NAKed packets generate interrupts | | |

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 16 | DCON | | Device status – connect. The connect bit must be set by SW to indicate that the device must signal connect. The pull-up resistor on D+ will be enabled when this bit is set and the VbusDebounced bit is one. | 0 | R/W |
| 17 | DSUS | | Device status – suspend. The suspend bit indicates the current suspend state. It is set to 1 when the device hasn´t seen any activity on its upstream port for more than 3 milliseconds. It is reset to 0 on any activity. When the device is suspended (Suspend bit = 1) and the firmware writes a 0 to it, the device will generate a remote wakeup. This will only happen when the device is connected (Connect bit = 1). When the device is not connected or not suspended, a writing a 0 has no effect. Writing a 1 has never an effect. | 0 | R/W |
| 18 | RESERVED | | Reserved | 0 | RO |
| 19 | LPM_SUS | | Device status – LPM Suspend. This bit represents the current LPM suspend state. It is set to 1 by hardware when the device has acknowledged the LPM request from the USB host and the Token Retry Time of 10 µs has elapsed. When the device is in the LPM suspended state (LPM suspend bit = 1) and the firmware writes a zero to this bit, the device will generate a remote wakeup. Firmware can only write a zero to this bit when the LPM_REWP bit is set to 1. Hardware resets this bit when it receives a host initiated resume. Hardware only updates the LPM_SUS bit when the LPM_SUPP bit is equal to one. | 0 | R/W |
| 20 | LPM_REWP | | LPM Remote Wakeup Enabled by USB host. Hardware sets this bit to one when the bRemoteWake bit in the LPM extended token is set to 1. Hardware will reset this bit to 0 when it receives the host initiated LPM resume, when a remote wakeup is sent by the device or when a USB bus reset is received. Firmware can use this bit to check if the remote wake up feature is enabled by the host for the LPM transaction. | 0 | R/W |
| 23:21 | RESERVED | | Reserved | 0 | RO |
| 24 | DCON_C | | Device status – connect change. The connect change bit is set when the devices pull-up resistor is disconnected because VBus disappeared. The bit is reset by firmware writing a one to it. | 0 | R/W/C |
| 25 | DSUS_C | | Device status – suspend change. The suspend change bit is set to 1 when the suspend bit toggles. The suspend bit can toggle because:<br>- The device goes in the suspended state<br>- The device is disconnected<br>- The device receives resume signaling on its upstream port<br>The bit is reset by firmware writing a one to it. | 0 | R/W/C |
| 26 | DRES_C | | Device status – reset change. This bit is set when the device received a bus reset. On a bus reset the device will automatically go to the default state (unconfigured and responding to address 0). The bit is reset by firmware writing a one to it. | 0 | R/W/C |
| 27 | RESERVED | | Reserved | 0 | RO |

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 28 | VbusDebounced | | This bit indicates if Vbus is detected or not. The bit raises immediately when Vbus becomes high. It drops to zero if Vbus is low for at least 3 ms. If this bit is high and the DCon bit is set, the hardware will enable the pull-up resistor to signal a connect. | 0 | RO |
| 31:29 | RESERVED | | Reserved | 0 | RO |

#### 15.4.2.2 USB info register

**Table 357. USB info register (address offset = 0x04)**

*Reset value: depends on the CHIP_ID value*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 10:0 | FRAME_NR | | Frame number. It contains the frame number of the last successfully received SOF. In case no SOF was received by the device at the beginning of a frame, the frame number returned is that of the last successfully received SOF. In case the SOF frame number contained a CRC error, the frame number returned will be the corrupted frame number as received by the device. | 0 | RO |
| 14:11 | ERR_CODE | | error code which last occurred: | 0 | R/W |
| | | 0x0 | No error | | |
| | | 0x1 | PID encoding error | | |
| | | 0x2 | PID unknown | | |
| | | 0x3 | Packet unexpected | | |
| | | 0x4 | Token CRC error | | |
| | | 0x5 | Data CRC error | | |
| | | 0x6 | Time out | | |
| | | 0x7 | Babble | | |
| | | 0x8 | Truncated EOP | | |
| | | 0x9 | Sent/receive NAK | | |
| | | 0xA | Sent stall | | |
| | | 0xB | Overrun | | |
| | | 0xC | Sent empty packet | | |
| | | 0xD | Bitstuff error | | |
| | | 0xE | Sync error | | |
| | | 0xF | Wrong data toggle | | |
| 15 | RESERVED | | Reserved | 0 | RO |
| 31:16 | CHIP_ID | | Chip identification: indicates major and minor revision of the IP<br>- [31:24] = Major revision<br>- [23:16] = Minor revision | 0 | RO |

#### 15.4.2.3 USB EP command/ Status list start address

This 32-bit register indicates the start address of the USB EP command/ status list. Only a subset of these bits is programmable by software. The 8 least-significant bits are hardcoded to zero because the list must start on a 256 byte boundary. Bits 31 to 8 can be programmed by software.

**Table 358. USB EP command/status list address (address offset = 0x08)**

*Reset value: see configuration values*

| Bit | Symbol | Description | Reset value | Access |
|---|---|---|---|---|
| 7:0 | RESERVED | Reserved | | RO |
| 31:8 | EP_LIST | These are the programmable bits for firmware to indicate the start address of the USB EP Command/Status List. | | R/W |

#### 15.4.2.4 USB data buffer start address

This 32-bit register indicates the page of the AHB address where the endpoint data can be located

**Table 359. USB data buffer start address (address offset = 0x0C)**

*Reset value: see configuration values*

| Bit | Symbol | Description | Reset Value | Access |
|---|---|---|---|---|
| 21:0 | RESERVED | Reserved | | RO |
| 31:22 | DA_BUF | These are the programmable bits for firmware to indicate the buffer pointer page where all endpoint data buffers are located. | | R/W |

#### 15.4.2.5 Link Power Management register

**Table 360. Link Power Management register (address offset = 0x10)**

*Reset value: 0x00000000*

| Bit | Symbol | Description | Reset Value | Access |
|---|---|---|---|---|
| 3:0 | HIRD_HW | Host initiated resume duration - hardware<br>This is the HIRD value from the last received LPM token | 0 | RO |
| 7:4 | HIRD_SW | Host initiated resume duration - software<br>This is the time duration required by the USB device system to come out of LPM initiated suspend after receiving the host initiated LPM resume. | 0 | R/W |
| 8 | Data Pending | As long as this bit is set to one and LPM supported bit is set to one, hardware will return a NYET handshake on every LPM token it receives. If LPM supported bit is set to one and this bit is zero, hardware will return an ACK handshake on every LPM token it receives. If SW has still data pending and LPM is supported, it must set this bit to „1". | 0 | R/W |
| 31:9 | RESERVED | Reserved | 0 | RO |

#### 15.4.2.6 USB Endpoint skip

**Table 361. USB Endpoint skip (address offset = 0x14)**

*Reset value: 0x00000000*

| Bit | Symbol | Description | Reset Value | Access |
|---|---|---|---|---|
| 29:0 | SKIP | Endpoint skip: Writing „1 to one of these bits, will indicate to hardware that it must deactivate the buffer assigned to this endpoint and return control back to firmware. When hardware has deactivated the endpoint it will clear this bit. But it will not modify the “Buffer in use” bit. An interrupt will be generated when the Active bit goes from „1“ to „0“.<br>Note: In case of double buffering, hardware will only clear the Active bit of the buffer indicated by the “Buffer in use” bit. | 0 | R/W |
| 31:30 | RESERVED | Reserved | 0 | RO |

#### 15.4.2.7 USB Endpoint Buffer in use

**Table 362. USB Endpoint Buffer in use (address offset = 0x18)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 1:0 | - | | Reserved | 0 | RO |
| 29:2 | BUF | | Buffer in use – This bit has one bit per physical endpoint | 0 | R/W |
| | | 0 | hardware is accessing buffer 0 | | |
| | | 1 | hardware is accessing buffer 1 | | |
| 31:30 | RESERVED | | Reserved | 0 | RO |

#### 15.4.2.8 USB Endpoint Buffer Configuration

**Table 363. USB Endpoint Buffer Configuration (address offset = 0x1C)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 1:0 | RESERVED | 0 | Reserved. Fixed to zero because the control endpoint zero is fixed to single buffering for each physical endpoint. | 0 | RO |
| 29:2 | BUF_SB | | Buffer usage – This bit has one bit per physical endpoint. If the bit is set to single buffer („0“), it will not toggle the corresponding “USB EP Buffer in use” bit when it clears the active bit.<br>If the bit is set to double buffer („1“), hardware will toggle the “USB EP Buffer in use” bit when it clears the Active bit for the buffer. | 0 | R/W |
| | | 0 | Single buffer | | |
| | | 1 | Double buffer | | |
| 31:30 | RESERVED | | Reserved | 0 | RO |

#### 15.4.2.9 USB Interrupt status register

**Table 364. USB Interrupt status register (address offset = 0x20)**

*Reset value: 0x00000000*

| Bit | Symbol | Description | Reset Value | Access |
|---|---|---|---|---|
| 0 | EP0OUT | Interrupt status register bit for the Control EP0 OUT direction. This bit will be set if NBytes transitions to zero or the skip bit is set by firmware or a SETUP packet is successfully received for the control EP0. If the IntOnNAK_CO is set, this bit will also be set when a NAK is transmitted for the Control EP0 OUT direction. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| 1 | EP0IN | Interrupt status register bit for the Control EP0 IN direction. This bit will be set if NBytes transitions to zero or the skip bit is set by firmware. If the IntOnNAK_CI is set, this bit will also be set when a NAK is transmitted for the Control EP0 IN direction. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| 2 | EP1OUT | Interrupt status register bit for the EP1 OUT direction. This bit will be set if the corresponding Active bit is cleared by hardware. This is done in case the programmed NBytes transitions to zero or the skip bit is set by firmware. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP1 OUT direction. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| 3 | EP1IN | Interrupt status register bit for the EP1 IN direction. This bit will be set if the corresponding Active bit is cleared by hardware. This is done in case the programmed NBytes transitions to zero or the skip bit is set by firmware. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP1 IN direction. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| … | | | | |
| 28 | EP14OUT | Interrupt status register bit for the EP14 OUT direction. This bit will be set if the corresponding Active bit is cleared by hardware. This is done in case the programmed NBytes transitions to zero or the corresponding skip bit is set by firmware. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP14 OUT direction. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| 29 | EP14IN | Interrupt status register bit for the EP14 IN direction. This bit will be set if the corresponding Active bit is cleared by hardware. This is done in case the programmed NBytes transitions to zero or the corresponding skip bit is set by firmware. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP14 IN direction. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| 30 | FRAME_INT | Frame interrupt. This bit is set to one every millisecond when the VbusDebounced bit and the DCON bit are set. This bit can be used by software when handling the isochronous endpoints. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |
| 31 | DEV_INT | Device status interrupt. This bit is set by hardware when one of the device status change bits is set. Firmware can clear this bit by writing a one to it. | 0 | R/W/C |

#### 15.4.2.10 USB Interrupt enable register

**Table 365. USB Interrupt enable register (address offset = 0x24)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 31:0 | INT_EN | | If this bit is set and the corresponding USB interrupt status bit is set a hardware interrupt is generated on the interrupt line indicated by the corresponding USB interrupt routing bit. | 0 | R/W |

#### 15.4.2.11 USB set interrupt status register

**Table 366. USB set interrupt status register (address offset = 0x28)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 31:0 | SET_EN | | If firmware writes a one to one of these bits, the corresponding USB interrupt status bit is set. When this register is read, the same value as the USB interrupt status register is returned. | 0 | R/W |

#### 15.4.2.12 USB interrupt routing register

**Table 367. USB interrupt routing register (address offset = 0x2C)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 31:0 | ROUTE_INT | | This bit can control on which hardware interrupt line the interrupt will be generated: | 0 | R/W |
| | | 0 | IRQ interrupt line is selected for this interrupt bit | | |
| | | 1 | FIQ interrupt line is selected for this interrupt bit | | |

#### 15.4.2.13 USB configuration

**Table 368. USB configuration (address offset = 0x30)**

*Reset value: see configuration values*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 4:0 | PHYSEP | | Number of physical endpoints implemented in this design (excluding the default control endpoint). E.g. a value of 2 in this field indicates that the design contains the default control endpoint plus one IN endpoint (EP1 – IN) and one OUT endpoint (EP1 – OUT) | | RO |
| 5 | SB | | EP single buffer supported | | RO |
| 6 | DB | | EP double buffer supported | | RO |

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 7 | TREG | | Toggle register available This indicates if the Data Toggle debug register is reserved or not. | | RO |
| 31:8 | RESERVED | | Reserved | | RO |

#### 15.4.2.14 USB endpoint toggle

**Table 369. USB endpoint toggle (address offset = 0x34)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 29:0 | TOGGLE | | Endpoint data toggle: This field indicates the current value of the data toggle for the corresponding endpoint. | | RO |
| 31:30 | RESERVED | | Reserved | | RO |

#### 15.4.2.15 USB internal PLL

**Table 370. USB internal PLL (address offset = 0x38)**

*Reset value: 0x00000000*

| Bit | Symbol | Value | Description | Reset Value | Access |
|---|---|---|---|---|---|
| 0 | SEL_EXT_CLK | | Select external crystal clock. | 0 | R/W |
| | | 0 | Internal PLL and RC circuit is selected as input clock for the USB block | | |
| | | 1 | External clock input pin is selected as input clock for USB block. The input clock must be 48MHz in this case. | | |
| 31:3 | RESERVED | | Reserved | 0 | R/W |

# 16. Abbreviations

Table 371. Abbreviations

| Acronym | Description |
|---|---|
| AL | Application Layer |
| ACK | Acknowledgement |
| AFI | Application family identifiers |
| ATQA | Answer To Request, type A |
| BAL | Bus Abstraction Layer |
| CB | Callback |
| CC | Connection Complete (in LLCP) |
| CRC | Cyclic Redundancy Check |
| DEP | Data Exchange Protocol |
| DID | Device Identifier |
| DISC | Disconnect (in LLCP) |
| DM | Disconnected Mode (in LLCP) |
| EEPROM | Electrically Erasable Programmable Read-Only Memory |
| FRI | Forum reference implementation |
| FRMR | Frame Reject (in LLCP) |
| GPIO | General Purpose Input Output |
| HAL | Hardware Abstraction Layer |
| I | Information (in LLCP) |
| I2C | Inter-Interchanged Circuit |
| IC | Integrated Circuit |
| LLC | Logical Link Control |
| LLCP | Logical Link Control Protocol |
| LRI | Long range interface |
| LTO | Link Timeout (in LLCP) |
| LUT | Lookup Table |
| MAC | Medium Access Control (in LLCP) |
| MCU | MicroController Unit |
| MF | MIFARE |
| MIU | Maximum Information Unit |
| MIUX | Maximum Information Unit Extension |
| NAD | Node ADdress |
| NAK | Negative AcKnowledgement |
| NDEF | NFC Data Exchange Format |
| NFC | Near Field Communication |
| NFCIP | NFC Interface and Protocol |
| OPT | Option link Parameter |

| Acronym | Description |
|---|---|
| PAL | Protocol Abstraction Layer |
| PAX | PArameter eXchange |
| PCD | Proximity Coupling Device |
| PICC | Proximity Integrated Circuit Card |
| PDU | Protocol Data Unit |
| PTYPE | PDU TYPE |
| RNR | Receive Not Ready |
| RR | Receive Ready |
| RW | Receive window size |
| SAM | Secure Access Module |
| SAP | Service Access Point |
| DSAP | Destination Service Access Point |
| SSAP | Source Service Access Point |
| SNEP | Simple NDEF Exchange Protocol |
| SPI | Serial Peripheral Interface |
| SYMM | SYMMetry token |
| UID | Unnumbered Information |
| UID | Unique IDentifier |
| WKS | Well Known Services |

# 17. Legal information

## 17.1 Definitions

**Draft —** The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

## 17.2 Disclaimers

**Limited warranty and liability —** Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

**Right to make changes —** NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

**Suitability for use —** NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

**Applications —** Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.**Export control —** This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

**Evaluation products —** This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

## 17.3 Licenses

**Purchase of NXP ICs with NFC technology**

Purchase of an NXP Semiconductors IC that complies with one of the Near Field Communication (NFC) standards ISO/IEC 18092 and ISO/IEC 21481 does not convey an implied license under any patent right infringed by implementation of any of those standards. Purchase of NXP Semiconductors IC does not include a license to any NXP patent (or other IP right) covering combinations of those products with other products, whether hardware or software.

**Purchase of NXP ICs with ISO 14443 type B functionality**

This NXP Semiconductors IC is ISO/IEC 14443 Type B software enabled and is licensed under Innovatron's Contactless Card patents license for ISO/IEC 14443 B.

The license includes the right to use the IC in systems and/or end-user equipment.

**RATP/Innovatron Technology**

## 17.4 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

**MIFARE —** is a trademark of NXP B.V.

**I²C-bus logo —** is a trademark of NXP B.V

**ICODE** — is trademark of NXP B.V.

# 18. List of figures

# 19. List of tables

# 20. Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

**Date of release: 08 September 2017**
**314512**
**Document identifier: UM10858**