UM11023

QN908x user manual

Rev. 1.0 — 17 March 2017 User manual

Document information

Info	Content
Keywords	ARM Cortex-M4, microcontroller, sensor hub, USB FS device
Abstract	QN908x user manual

Revision history

Rev	Date	Description

Contact information

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

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

The QN908x is a single chip, 10 mW peak power, high-performance Bluetooth Low-Energy SoC wearable platform. It facilitates the development of wearable applications such as activity tracker, sports shoes, sports equipment, biosensor patch, heart rate monitor and smart watch.

QN908x integrates a Bluetooth Low Energy (v4.2) compliant radio, link controller, host stack and profiles, and provides a single-chip solution for Bluetooth Smart applications. The integrated 32-bit ARM Cortex-M4F MCU together with FSP, on-chip flash memory, as well as a mix of analogue and digital peripherals provide the most efficient data fusion engine. These features make QN908x a superior solution for digital health, sports and fitness applications.

Additional system features include fully integrated DC-to-DC converter and LDO, low-power sleep timer, battery monitor, high-resolution ADC, and GPIOs. These features reduce the overall system cost and size.

The QN908x operates with a power supply that ranges from 1.62 V to 3.6 V. The best-in-class active current with ultra-low power sleep modes gives excellent battery life allowing operation directly from a coin cell.

1.1 Features

- True single-chip Bluetooth Low-Energy (v4.2) SoC solution:
  - Integrates Bluetooth LE radio, protocol stack and application profiles
  - Supports central and peripherals roles
  - Supports master/slave concurrency
  - Supports 16 simultaneous links
  - Supports extended packet length
  - Wifi/Bluetooth LE coexistence interface
  - 48-bit unique BD address in flash
- RF design supports:
  - −93 dBm RX sensitivity (DC-to-DC mode)
  - TX output power from −20 dBm to 2 dBm
  - Single-ended RF port with integrated balun
  - Fast and reliable RSSI in 1dB step
  - Compatible with worldwide radio frequency regulations
- Very low power consumption:
  - Single 1.62 V to 3.6 V power supply
  - Integrated DC-to-DC buck converter and LDO
  - 1 µA power-down mode, to wake up by GPIO
  - 2 µA power-down mode, to wake up by 32 kHz sleep timer, RTC and GPIO
  - 3.6 mA RX current with DC-to-DC converter enabled at 3 V supply

- 3.4 mA TX current @0dBm TX power with DC-to-DC converter enabled at 3 V supply
- 25 μA/MHz for MCU execution from RAM
- 30 μA/MHz for MCU execution from flash

- ARM Cortex-M4 core (version r0p1):
  - ARM Cortex-M4 processor, running at a frequency of up to 32 MHz
  - Floating Point Unit (FPU) and Memory Protection Unit (MPU)
  - ARM Cortex-M4 built-in Nested Vectored Interrupt Controller (NVIC)
  - Serial Wire Debug (SWD) with six instruction breakpoints, two literal comparators, and four watch points. SWD includes serial wire output for enhanced debug capabilities
  - System tick timer
- On-chip memory:
  - Up to 512 kB on-chip flash program memory and 2 kB page erase and write
  - Up to 128 kB SRAM
  - Up to 256 kB ROM
- ROM API supports:
  - Flash In-System Programming (ISP)
  - Bluetooth host stack API
- Serial interfaces:
  - Four Flexcomm serial peripherals. The USART protocol is supported by the Flexcomm0, USART and I^{2}C-bus by the Flexcomm1, SPI and I^{2}C-bus by Flexcomm2, and SPI by Flexcomm3. Each Flexcomm includes a FIFO
  - I^{2}C-bus interfaces support fast-mode with multiple address recognition and monitor mode
  - USB 2.0 full-speed device interface
  - Two quadrature decoder
  - SPI Flash Interface (SPIFI) uses an SPI bus superset with four data lines to access off-chip quad SPI flash memory at a higher rate than that using standard SPI or SSP interfaces. SPIFI supports SPI memories with one or four data lines
- Digital peripherals:
  - DMA controller with 20 channels to access memories and DMA-capable peripherals
  - Up to 35 General-Purpose Input Output (GPIO) pins, with configurable pull-up or pull-down resistors
  - GPIO registers are located on the AHB for fast access
  - 32 GPIOs can be selected as Pin INTerrupts (PINT), triggered by rising or falling input edges
  - AES-128 security co-processor
  - Random Number Generator (RNG)
  - CRC engine

- Fusion Signal Processor (FSP) for data fusion and machine learning, much lower power consumption against software processing
- Analog peripherals:
  - 16-bit ADC with eight input channels of sample rates up to 32k sample per second, and with multiple internal and external trigger inputs
  - Integrated temperature sensor, connected to one internal dedicated ADC channel
  - Integrated battery monitor, connected to one internal dedicated ADC channel
  - General purpose 8-bit 1M sample per second DAC
  - Integrated capacitive sense up to eight channels, able to wake up MCU from low-power states
  - Two ultra low-power analog comparators, able to wake up MCU from low-power states
- Timers:
  - Four 32-bit general purpose timers/counters, support captured inputs and compare outputs, PWM mode, and external count input
  - Sleep timer, which can work in power down mode and wake up MCU
  - 32-bit Real-Time Clock (RTC) with 1 second resolution, running in the always-on power domain. It can be used for wake-up from all low-power modes including power-down
  - Watchdog timer
  - SCTimer/PWM
- Clock generation:
  - 32 MHz internal free running oscillator that can be used as a system clock
  - 16/32 MHz crystal oscillator that can be used as a system and RF reference
  - 32 kHz on-chip RC oscillator
  - 32.768 kHz crystal oscillator
- Single power supply ranges from 1.62 V to 3.6 V
- Operating temperature ranges –40 °C to +85 °C
- Available as 6 × 6 HVQFN48 and 3.28 × 3.20 mm WLCSP packages

1.2 Package information

1.2.1 HVQFN48

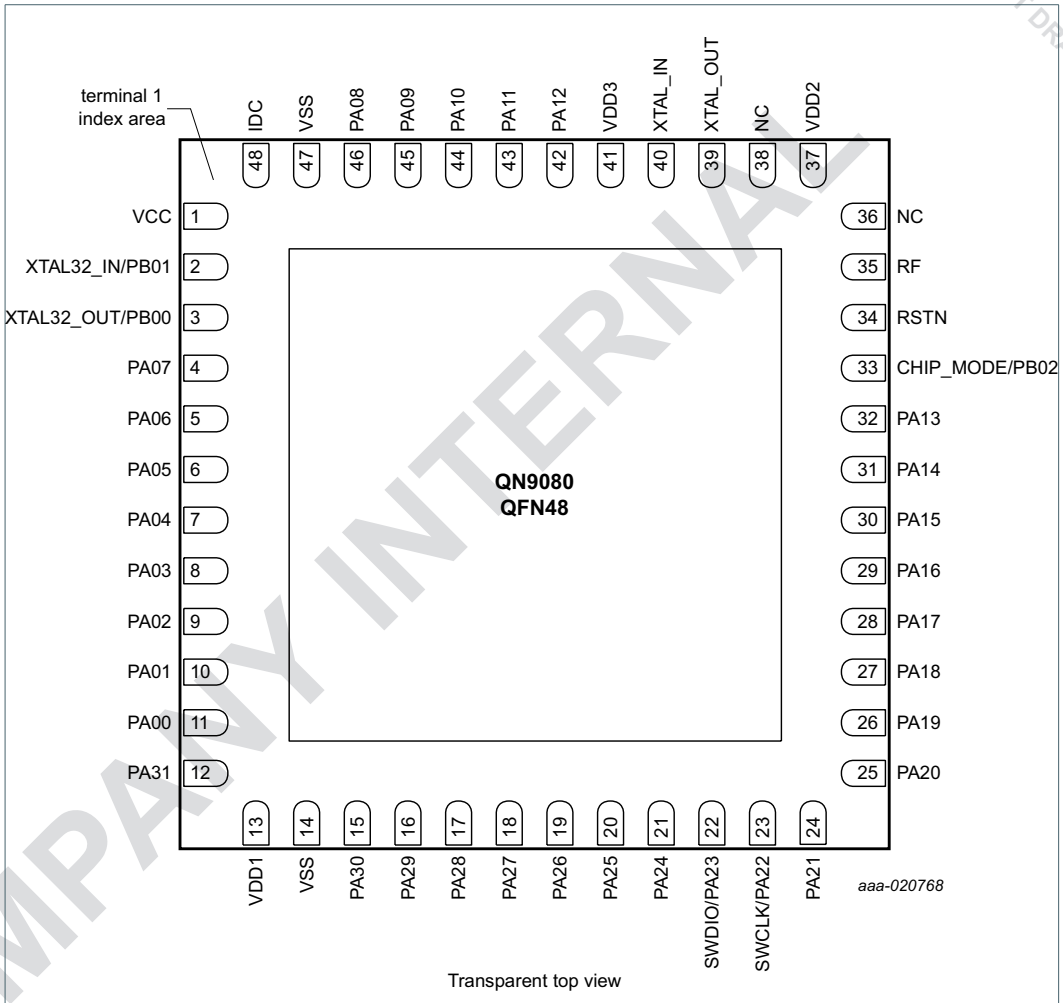


Fig 1. HVQFN48 pin configuration

1.2.2 WLCSP

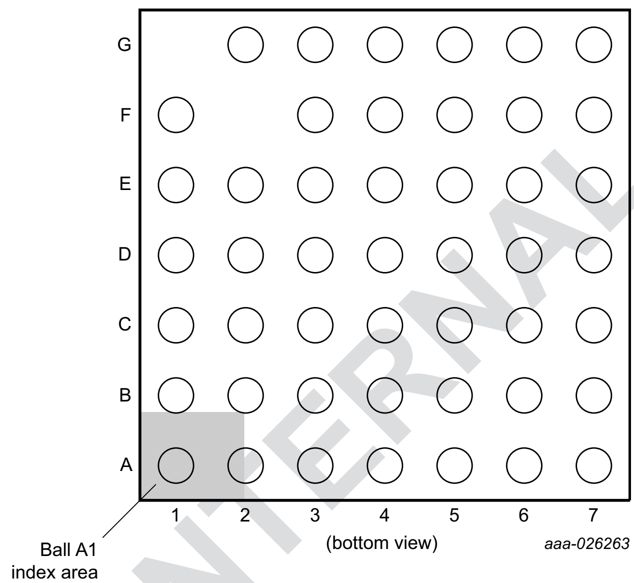


Fig 2. WLCSP pin configuration

2. System architecture and configuration (SYSCON)

2.1 System block diagram

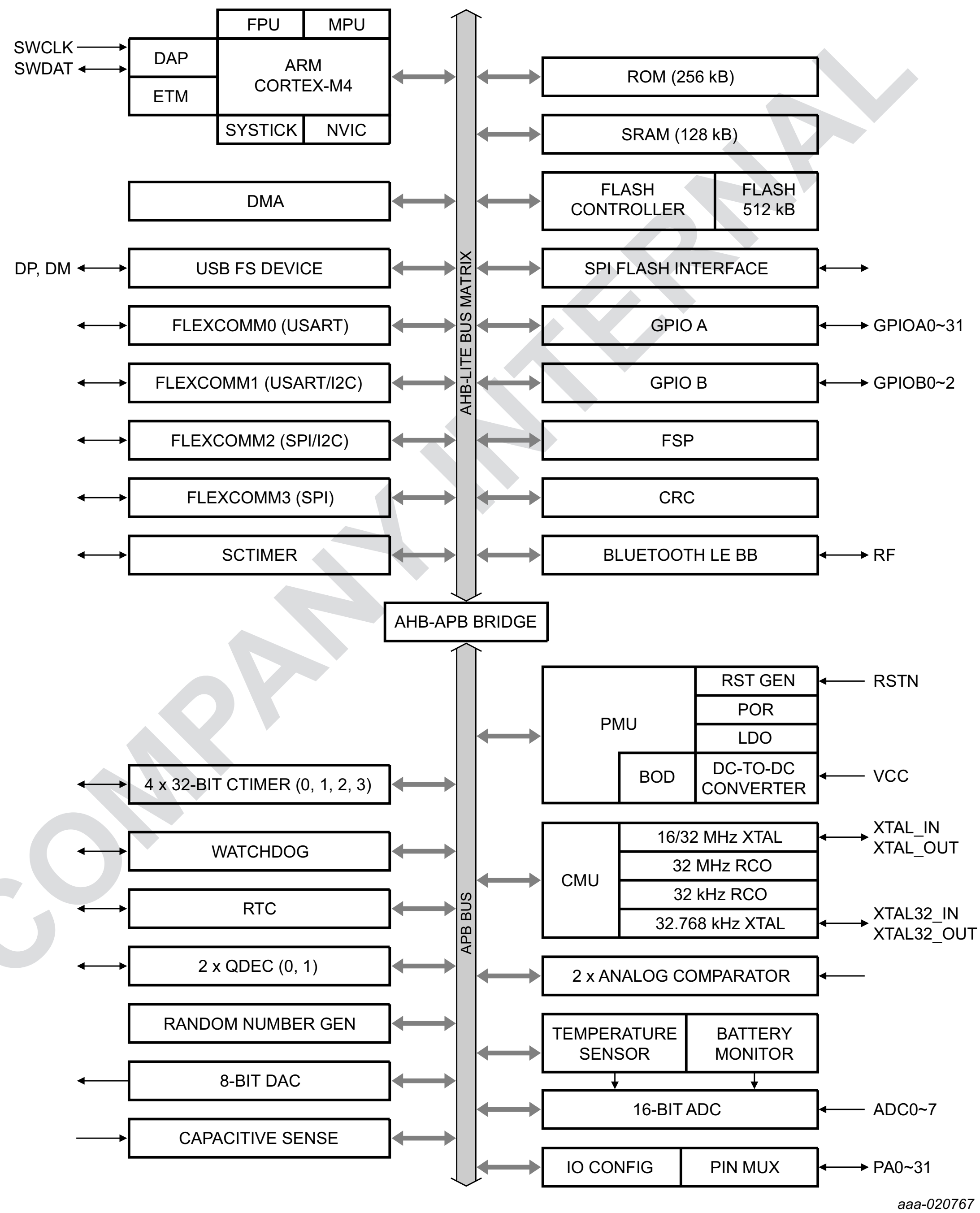


Fig 3. System architecture

2.2 Architectural overview

QN908x contains Bluetooth 4.2 LE RF, core MCU, memory, clock, power management, GPIO and other programmable peripherals with AHB and APB bus system to connect these components together.

The AHB-Lite bus matrix in Cortex-M4 includes three AHB-Lite buses, one system bus and the I-code and D-code buses. One bus is dedicated for instruction fetch (I-code), and one bus is dedicated for data access (D-code). The use of two core buses allow simultaneous operations if concurrent operations target different devices.

A multi-layer AHB matrix connects the bus masters to peripherals. It connects in a flexible manner that optimizes performance by allowing peripherals on different slaves ports of the matrix to be accessed simultaneously by different bus masters. Connections in the multi-layer matrix are shown in Figure 4. Note that while the AHB bus itself supports word, halfword, and byte accesses, not all AHB peripherals require or provide that support.

APB peripherals are connected to the APB bus which is connected to AHB matrix via AHB-APB bridge. Note that APB, by definition, does not directly support byte or halfword accesses.

2.3 ARM Cortex-M4 processor

The Cortex-M4 is a general purpose 32-bit microprocessor, which offers high performance and very low power consumption. The Cortex-M4 offers a Thumb-2 instruction set, low interrupt latency, and interruptible or continuable multiple load and store instructions. It also provides automatic state save and restore for interrupts, tightly integrated interrupt controller, multiple core buses capable of simultaneous accesses, and a floating point unit.

A 3-stage pipeline is employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is decoded, and a third instruction is fetched from memory.

Information about Cortex-M4 configuration options is found in Cortex-M4 TRM.

2.4 Reset Management Unit (RMU)

The Reset Management Unit (RMU) ensures correct reset operation. It connects different reset sources to the reset lines of QN908x.

A correct reset sequence ensures safe and synchronous startup of the QN908x. In the case of error situations such as power supply glitches or software crash, the RMU provides proper reset and startup of the QN908x.

The power-on reset and brownout reset of the QN908x provides power line monitoring with exceptionally low power consumption. The cause of reset may be read from a register, thus providing software with information about the cause of reset.

2.4.1 Reset sources

QN908x has different reset sources belonging to different reset level types. Table 1 shows grouping of reset sources, as per reset level from high to low.

Table 1. Reset source types for QN908x

Reset level type (high to low)	Description	Reset source
global reset	resets all flip-flop and boots CPU again	Power-On Reset (POR)
		Brown-Out Reset (BOR)
		RESETn pin reset
		RTC output reset
system reset	Resets all flip-flops and boots CPU again, except for the following: 1. System reset source cannot reset SWD interface 2. Clock will not disappear while system reset and recover to default settings after system reset	software reboot
		watchdog reset
		core LOCKUP condition
		ARM SW reset
block reset	resets flip-flops that are in one block	software reset for all peripherals in RST_SW_SET register

2.4.1.1 Power-On Reset (POR)

POR ensures that the QN908x does not start up before the supply voltage V_{CC} has reached the threshold voltage. Before the threshold voltage is reached, the QN908x is kept in reset state.

2.4.1.2 Brown-Out Reset (BOR)

QN908x supports BOR up to four levels to monitor the voltage on the pin VCC. If the voltage at pin VCC falls below one of the selected levels, a BOR is generated.

Refer to Section 5.2.2 for more details for BOD and BOR.

2.4.1.3 RESETn pin reset

Forcing the RESETn pin low generates a reset of the QN908x. If external reset source is not required, the RESETn pin is left unconnected.

2.4.1.4 RTC reset

RTC works during both active and sleep modes. When RTC reset is enabled, if the low speed clock is used (enabled), a free running counter in RTC generates a reset when the value of counter equals the threshold. RTC reset is disabled by default. To enable RTC reset, the bits CNT2_RST and CNT2_EN in CNT2_CTRL register are configured to 1. THR_RST is used to set the threshold of RTC reset. This functionality provides recovery from a sleep stalemate.
Note: The high 16-bit write data should be 0x5285 to configure CNT2_CTRL register.

2.4.1.5 Software reset

If the bit SET_REBOOT in RST_SW_SET register is set to 1 through software, a system reboot reset is generated.

2.4.1.6 Watchdog reset

The watchdog circuit is a timer which when enabled, is cleared by software regularly. If software does not clear it, a watchdog reset is activated. This functionality provides recovery from a software stalemate.

2.4.1.7 Lockup reset

A Cortex-M4 lockup indicates incorrect working of kernel software. It is a result of core being locked up due to an unrecoverable exception, following the activation of the built-in system state protection hardware of the processor. Lockup reset is disabled by default. Set the LOCKUP_EN bit of SYS_MODE_CTRL register to 1 to enable lockup reset.

2.4.1.8 System reset request

In Cortex-M4, there is an internal Application Interrupt and Reset Control Register (AIRCR) having the address 0xE000ED0C. In AIRCR, bit 3 is SYSRESETREQ, which when set to 1, asserts a signal to the outer system that requests a reset. It is intended to force a system reset of all major components, except for the debug. The system reset request behavior is similar to software reboot. But the command 'r' of SWD J-LINK and the reset request of KEIL is to configure SYSRESETREQ to 1.

2.4.1.9 Block reset

Block reset is controlled by software. Software configures registers RST_SW_SET and RST_SW_CLR to set or clear block reset. Every bit in RST_SW_SET and RST_SW_CLR is to control different blocks such as DMA, timer, etc.

2.5 System register description

All system control block registers reside on word address boundaries. Details of the registers appear in the description of each function. Base address of system configuration registers is at 0x4000 0000.

All address offsets not shown in the tables are reserved and should not be written to.

Table 2. System register overview (base address 0x4000 0000)

Name	Access	Offset	Description	Reset value	Section
RST_SW_SET	WOO[1]	0x0	block software reset set register	0x7E5FFFFF	2.5.1
RST_SW_CLR	W1C[2]	0x4	block software reset clear register	0x7E5FFFFF	2.5.2
CLK_DIS	W1C	0x8	clock disable register	0xC0001200	2.5.3
CLK_EN	WOO	0xC	clock enable register	0xC0001200	2.5.4
CLK_CTRL	R/W	0x10	system clock source and divider register	0x103C0000	2.5.5
SYS_MODE_CTRL	R/W	0x14	system mode and address remap register	0x0	2.5.6
SYS_STAT	R	0x80	system status register	0x3000	2.5.7
SYS_TICK	R/W	0x100	systick timer control register	0x81000147	2.5.8
SRAM_CTRL	R/W	0x104	exchange memory base address register	0x2400	2.5.9
CHIP_ID	R	0x108	chip id register	0xFC001010	2.5.10
ANA_CTRL0	R/W	0x110	crystal and PA register	0x820007F	2.5.11
XTAL_CTRL	R/W	0x180	crystal control register	0x2020080A	2.5.12
BUCK	R/W	0x184	BUCK control register	0x302B03	2.5.13
FC_FRG	R/W	0x200	flexcomm clock divider register	0xFF00FF00	2.5.14
PIO_PULL_CFG0	R/W	0x800	pad pull control register 0	0xAAAAAAAA	2.5.15
PIO_PULL_CFG1	R/W	0x804	pad pull control register 1	0xAAAAAAAA	2.5.16
PIO_PULL_CFG2	R/W	0x808	pad pull control register 2	0x2A	2.5.17
IO_CAP	WOO	0x80C	I/O status capture register	0x0	2.5.18

Table 2. System register overview (base address 0x4000 0000) *…continued*

Name	Access	Offset	Description	Reset value	Section
PIO_DRV_CFG0	R/W	0x810	pad drive strength register 0	0x0	2.5.19
PIO_DRV_CFG1	R/W	0x814	pad drive strength register 1	0x0	2.5.20
PIO_DRV_CFG2	R/W	0x818	pad drive extra register	0x0	2.5.21
PIO_CFG_MISC	R/W	0x81C	pin misc control register	0x3	2.5.22
PIO_WAKEUP_LVL0	R/W	0x820	pin wakeup polarity register 0	0x0	2.5.23
PIO_WAKEUP_LVL1	R/W	0x824	pin wakeup polarity register 1	0x0	2.5.24
PIO_IE_CFG0	R/W	0x828	pad input enable register 0	0xFFFFFFFF	2.5.25
PIO_IE_CFG1	R/W	0x82C	pad input enable register 1	0x7	2.5.26
PIO_FUNC_CFG0	R/W	0x830	pin MUX control register 0	0x0	2.5.27
PIO_FUNC_CFG1	R/W	0x834	pin MUX control register 1	0x0	2.5.28
PIO_FUNC_CFG2	R/W	0x838	pin MUX control register 2	0x0	2.5.29
PIO_FUNC_CFG3	R/W	0x83C	pin MUX control register 3	0x0	2.5.30
PIO_WAKEUP_EN0	R/W	0x840	pin function selection in sleep mode register 0	0x0	2.5.31
PIO_WAKEUP_EN1	R/W	0x844	pin function selection in sleep mode register 1	0x0	2.5.32
PIO_CAP_OE0	R	0x848	pin output enable status in sleep mode register 0	0x0	2.5.33
PIO_CAP_OE1	R	0x84C	pin output enable status in sleep mode register 1	0x0	2.5.34
PIO_CAP_OUT0	R	0x850	pin output status in sleep mode register 0	0x0	2.5.35
PIO_CAP_OUT1	R	0x854	pin output status in sleep mode register 1	0x0	2.5.36
RST_CAUSE_SRC	R/W	0x858	reset source status register	0x0	2.5.37
PMU_CTRL0	R/W	0x85C	power management unit control register 0	0x80000000	2.5.38
PMU_CTRL1	R/W	0x860	power management unit control register 1	0xC0FF0	2.5.39
ANA_EN	R/W	0x864	analog setting register	0x0	2.5.40
XTAL32K_CTRL	R/W	0x868	crystal 32k control register	0x3023	2.5.41
USB_CFG	R/W	0x86C	USB configuration register	0x32	2.5.42
PMU_CTRL2	R/W	0x880	power management unit control register 2	0x403FF0FF	2.5.43
ANA_CTRL1	R/W	0x884	IVREF and DVREG setting register	0xE4D98000	2.5.44
MISC	R/W	0x890	MISC register	0x3000000	2.5.45

[1] WOO: Write-only one. Write value must be 1.

[2] W1C: Write 1 to clear. If the bit in the written value is a 1, the corresponding bit in the field is set to 0. Otherwise, the field bit is not affected.

2.5.1 Software reset set register

Table 3. RST_SW_SET - Software Reset Set Register (offset = 0x0) bit description

Bit	Symbol	Access	Reset value	Description
0	SET_FC0_RST	WOO	0x1	write 1 to set FLEXCOMM0 reset
1	SET_FC1_RST	WOO	0x1	write 1 to set FLEXCOMM1 reset
2	SET_FC2_RST	WOO	0x1	write 1 to set FLEXCOMM2 reset
3	SET_FC3_RST	WOO	0x1	write 1 to set FLEXCOMM3 reset
4	SET_TIM0_RST	WOO	0x1	write 1 to set CTIMER0 reset
5	SET_TIM1_RST	WOO	0x1	write 1 to set CTIMER1 reset
6	SET_TIM2_RST	WOO	0x1	write 1 to set CTIMER2 reset

Table 3. RST_SW_SET - Software Reset Set Register (offset = 0x0) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
7	SET_TIM3_RST	WOO	0x1	write 1 to set CTIMER3 reset
8	SET_SCT_RST	WOO	0x1	write 1 to set SCT reset
9	SET_WDT_RST	WOO	0x1	write 1 to set watchdog reset
10	SET_USB_RST	WOO	0x1	write 1 to set USB reset
11	SET_GPIO_RST	WOO	0x1	write 1 to set GPIO reset
12	SET_RTC_RST	WOO	0x1	write 1 to set RTC reset
13	SET_ADC_RST	WOO	0x1	write 1 to set ADC reset
14	SET_DAC_RST	WOO	0x1	write 1 to set DAC reset
15	SET_CS_RST	WOO	0x1	write 1 to set capacitive sense reset
16	SET_FSP_RST	WOO	0x1	write 1 to set FSP reset
17	SET_DMA_RST	WOO	0x1	write 1 to set DMA reset
18	RESERVED	-	-	-
19	SET_QDEC0_RST	WOO	0x1	write 1 to set QDEC 0 reset
20	SET_QDEC1_RST	WOO	0x1	write 1 to set QDEC 1 reset
21	RESERVED	-	-	-
22	SET_SPIFI_RST	WOO	0x1	write 1 to set SPIFI reset
25:23	RESERVED	-	-	-
26	SET_CPU_RST	WOO	0x1	write 1 to set CPU reset
27	SET_BLE_RST	WOO	0x1	write 1 to set BLE reset
28	SET_FLASH_RST	WOO	0x1	write 1 to set flash controller reset
29	SET_DP_RST	WOO	0x1	write 1 to set datapath reset
30	SET_REG_RST	WOO	0x1	write 1 to reset retention register
31	SET_REBOOT	WOO	0x0	write 1 to reboot entire system

2.5.2 Software reset clear register

Table 4. RST_SW_CLR - Software Reset Clear register (offset = 0x4) bit description

Bit	Symbol	Access	Reset value	Description
0	CLR_FC0_RST	W1C	0x1	write 1 to clear FLEXCOMM0 reset
1	CLR_FC1_RST	W1C	0x1	write 1 to clear FLEXCOMM1 reset
2	CLR_FC2_RST	W1C	0x1	write 1 to clear FLEXCOMM2 reset
3	CLR_FC3_RST	W1C	0x1	write 1 to clear FLEXCOMM3 reset
4	CLR_TIM0_RST	W1C	0x1	write 1 to clear timer 0 reset
5	CLR_TIM1_RST	W1C	0x1	write 1 to clear timer 1 reset
6	CLR_TIM2_RST	W1C	0x1	write 1 to clear timer 2 reset
7	CLR_TIM3_RST	W1C	0x1	write 1 to clear timer 3 reset
8	CLR_SCT_RST	W1C	0x1	write 1 to clear PWM reset
9	CLR_WDT_RST	W1C	0x1	write 1 to clear watchdog reset
10	CLR_USB_RST	W1C	0x1	write 1 to clear USB reset
11	CLR_GPIO_RST	W1C	0x1	write 1 to clear GPIO reset
12	CLR_RTC_RST	W1C	0x1	write 1 to clear RTC reset
13	CLR_ADC_RST	W1C	0x1	write 1 to clear ADC reset

Table 4. RST_SW_CLR - Software Reset Clear register (offset = 0x4) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
14	CLR_DAC_RST	W1C	0x1	write 1 to clear DAC reset
15	CLR_CS_RST	W1C	0x1	write 1 to clear capacitive sense reset
16	CLR_FSP_RST	W1C	0x1	write 1 to clear FSP reset
17	CLR_DMA_RST	W1C	0x1	write 1 to clear DMA reset
18	RESERVED	-	-	-
19	CLR_QDEC0_RST	W1C	0x1	write 1 to clear QDEC 0 reset
20	CLR_QDEC1_RST	W1C	0x1	write 1 to clear QDEC 1 reset
21	RESERVED	-	-	-
22	CLR_SPIFI_RST	W1C	0x1	write 1 to clear SPIFI reset
25:23	RESERVED	-	-	-
26	CLR_CPU_RST	W1C	0x1	write 1 to clear CPU reset
27	CLR_BLE_RST	W1C	0x1	write 1 to clear BLE reset
28	CLR_FLASH_RST	W1C	0x1	write 1 to clear flash controller reset
29	CLR_DP_RST	W1C	0x1	write 1 to clear datapath reset
30	CLR_REG_RST	W1C	0x1	write 1 to clear retention register reset
31	RESERVED	-	-	-

2.5.3 Clock disable register

Table 5. CLK_DIS - Clock Disable register (offset = 0x8) bit description

Bit	Symbol	Access	Reset value	Description
0	CLK_FC0_DIS	W1C	0x0	write 1 to disable FLEXCOMM0 clock, no effect when write 0
1	CLK_FC1_DIS	W1C	0x0	write 1 to disable FLEXCOMM1 clock, no effect when write 0
2	CLK_FC2_DIS	W1C	0x0	write 1 to disable FLEXCOMM2 clock, no effect when write 0
3	CLK_FC3_DIS	W1C	0x0	write 1 to disable FLEXCOMM3 clock, no effect when write 0
4	CLK_TIM0_DIS	W1C	0x0	write 1 to disable timer 0 clock, no effect when write 0
5	CLK_TIM1_DIS	W1C	0x0	write 1 to disable timer 1 clock, no effect when write 0
6	CLK_TIM2_DIS	W1C	0x0	write 1 to disable timer 2 clock, no effect when write 0
7	CLK_TIM3_DIS	W1C	0x0	write 1 to disable timer 3 clock, no effect when write 0
8	CLK_SCT_DIS	W1C	0x0	write 1 to disable PWM clock, no effect when write 0
9	CLK_WDT_DIS	W1C	0x1	write 1 to disable watchdog clock, no effect when write 0
10	CLK_USB_DIS	W1C	0x0	write 1 to disable USB clock, no effect when write 0
11	CLK_GPIO_DIS	W1C	0x0	write 1 to disable GPIO clock, no effect when write 0

Table 5. CLK_DIS - Clock Disable register (offset = 0x8) bit description

Bit	Symbol	Access	Reset value	Description
12	CLK_BIV_DIS	W1C	0x1	write 1 to disable BIV APB clock include RTC BIV register, no effect when write 0
13	CLK_ADC_DIS	W1C	0x0	write 1 to disable ADC clock, no effect when write 0
14	CLK_DAC_DIS	W1C	0x0	write 1 to disable DAC clock, no effect when write 0
15	CLK_CS_DIS	W1C	0x0	write 1 to disable capacitive sense clock, no effect when write 0
16	CLK_FSP_DIS	W1C	0x0	write 1 to disable FSP clock, no effect when write 0
17	CLK_DMA_DIS	W1C	0x0	write 1 to disable DMA clock, no effect when write 0
18	RESERVED	-	-	-
19	CLK_QDEC0_DIS	W1C	0x0	write 1 to disable QDEC0 clock, no effect when write 0
20	CLK_QDEC1_DIS	W1C	0x0	write 1 to disable QDEC1 clock, no effect when write 0
21	CLK_DP_DIS	W1C	0x0	write 1 to disable datapath 16 MHz or 8 MHz clock, no effect when write 0
22	CLK_SPIFI_DIS	W1C	0x0	write 1 to disable SPIFI clock, no effect when write 0
24:23	RESERVED	-	-	-
25	CLK_CAL_DIS	W1C	0x0	write 1 to disable calibration clock, no effect when write 0
26	RESERVED	-	-	-
27	CLK_BLE_DIS	W1C	0x0	write 1 to disable BLE clock, no effect when write 0
29:28	RESERVED	-	-	-
30	PCLK_DIS	W1C	0x1	write 1 to disable PCLK of some logic, no effect when write 0
31	FCLK_DIS	W1C	0x1	write 1 to disable CPU FCLK, no effect when write 0

2.5.4 Clock enable register

Table 6. CLK_EN - Clock Enable register (offset = 0xC) bit description

Bit	Symbol	Access	Reset value	Description
0	CLK_FC0_EN	WOO	0x0	write 1 to enable FLEXCOMM0 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
1	CLK_FC1_EN	WOO	0x0	write 1 to enable FLEXCOMM1 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
2	CLK_FC2_EN	WOO	0x0	write 1 to enable FLEXCOMM2 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
3	CLK_FC3_EN	WOO	0x0	write 1 to enable FLEXCOMM3 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
4	CLK_TIM0_EN	WOO	0x0	write 1 to enable timer 0 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
5	CLK_TIM1_EN	WOO	0x0	write 1 to enable timer 1 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
6	CLK_TIM2_EN	WOO	0x0	write 1 to enable timer 2 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
7	CLK_TIM3_EN	WOO	0x0	write 1 to enable timer 3 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
8	CLK_SCT_EN	WOO	0x0	write 1 to enable PWM clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled

Table 6. CLK_EN - Clock Enable register (offset = 0xC) bit description

Bit	Symbol	Access	Reset value	Description
9	CLK_WDT_EN	WOO	0x1	write 1 to enable watchdog clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
10	CLK_USB_EN	WOO	0x0	write 1 to enable USB clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
11	CLK_GPIO_EN	WOO	0x0	write 1 to enable GPIO clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
12	CLK_BIV_EN	WOO	0x1	Write 1 to enable BIV APB clock include RTC BiV register, no effect when write 0; Read the register can know the status of clock, 1 to indicate enabled, 0 to indicate disabled.
13	CLK_ADC_EN	WOO	0x0	write 1 to enable ADC clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
14	CLK_DAC_EN	WOO	0x0	write 1 to enable DAC clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
15	CLK_CS_EN	WOO	0x0	write 1 to enable capacitive sense clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
16	CLK_FSP_EN	WOO	0x0	write 1 to enable FSP clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
17	CLK_DMA_EN	WOO	0x0	write 1 to enable DMA clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
18	RESERVED	-	-	-

Table 6. CLK_EN - Clock Enable register (offset = 0xC) bit description

Bit	Symbol	Access	Reset value	Description
19	CLK_QDEC0_EN	WOO	0x0	write 1 to enable QDEC0 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
20	CLK_QDEC1_EN	WOO	0x0	write 1 to enable QDEC1 clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
21	CLK_DP_EN	WOO	0x0	write 1 to enable data path 16/8MHz clock
22	CLK_SPIFI_EN	WOO	0x0	write 1 to enable SPIFI clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
24:23	RESERVED	-	-	-
25	CLK_CAL_EN	WOO	0x0	write 1 to enable calibration clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
26	RESERVED	-	-	-
27	CLK_BLE_EN	WOO	0x0	write 1 to enable BLE clock, no effect when 0 is written read the register to know the status of clock, 1 indicates that clock is enabled, 0 indicates that clock is disabled
31:28	RESERVED	-	-	-

2.5.5 System clock source and divider register

Table 7. CLK_CTRL - System Clock Source and Divider register (offset = 0x10) bit description

Bit	Symbol	Access	Reset value	Description
3:0	APB_DIV	RW	0x0	APB_CLK: AHB_CLK/(APB_DIV+1)
16:4	AHB_DIV	RW	0x0	AHB_CLK: SYS_CLK / (AHB_DIV+1) **Remark**: Before enabling BLE clock (CLK_BLE_EN:1), it is mandatory to set AHB_CLK: 32/16/8 MHz
17	CLK_BLE_SEL	RW	0x0	BLE frequency indicator 0: BLE run at 8 MHz 1: BLE run at 16 MHz
18	CLK_WDT_SEL	RW	0x1	watchdog clock selection 0: watchdog run at 32 kHz 1: watchdog run at APB clock

Table 7. CLK_CTRL - System Clock Source and Divider register (offset = 0x10) *…continued*bit description

Bit	Symbol	Access	Reset value	Description
19	CLK_XTAL_SEL	RW	0x1	crystal clock indicator 0: external crystal is 16 MHz 1: external crystal is 32 MHz
20	CLK_OSC32M_DIV	RW	0x1	digital OSC clock input selection 0: uses original 32 MHz OSC clock 1: divides 32 MHz OSC clock into 16 MHz
21	CLK_32K_SEL	RW	0x1	digital 32 kHz clock source selection 0: digital 32 kHz clock source is external 32 kHz crystal 1: digital 32 kHz clock source is internal 32 kHz RCO
22	CLK_XTAL_OE	RW	0x0	system clock output enable
23	CLK_32K_OE	RW	0x0	32 kHz clock output enable
27:24	XTAL_OUT_DIV	RW	0x0	high frequency crystal clock output divider
28	CGBYPASS	RW	0x1	if it is 0, it can save CPU power in active mode
29	RESERVED	-	-	-
31:30	SYS_CLK_SEL	RW	0x0	SYS_CLK source selection 00: 32 MHz internal clock 01: external 16/32 MHz crystal 10: 32 kHz clock

2.5.6 System mode and address remap register

Table 8. SYS_MODE_CTRL, - System Mode and Address Remap register (offset=0x14) bit description

Bit	Symbol	Access	Reset Value	Description
1:0	REMAP	RW	0x0	software remap system address 0: address 0 is mapped to ROM 1: address 0 is mapped to FLASH 2: address 0 is mapped to RAM
2	LOCKUP_EN	RW	0x0	lock up enable
23:3	RESERVED	-	-	-
24	RESERVED	-	-	-
25	XTAL_RDY	R	0x0	16/32 MHz crystal ready readout
26	XTAL32K_RDY	R	0x0	32 kHz crystal ready readout
27	PLL48M_RDY	R	0x0	48 MHz PLL ready readout
28	OSC32M_RDY	R	0x0	32 MHz oscillator ready readout

Table 8. SYS_MODE_CTRL, - System Mode and Address Remap register (offset=0x14) *…continued*bit description

Bit	Symbol	Access	Reset Value	Description
29	BG_RDY	R	0x0	BG ready readout
30	RESERVED	-	-	-
31	BOOT_MODE	R	0x0	Boot mode pin status 0: Bootloader enters ISP 1: Bootloader jumps to flash without ISP

2.5.7 System status register

Table 9. SYS_STAT - System Status register (offset = 0x80) bit description

Bit	Symbol	Access	Reset Value	Description
7:0	FREQ_WORD	R	0x0	BLE frequency word
8	BLE_FREQ_HOP	R	0x0	BLE frequency word change flag
9	EVENT_IN_PROCESS	R	0x0	BLE event indicator 0: BLE is not in event process 1: BLE is in event process
10	RX_EN	R	0x0	when 1, system is in RX state
11	TX_EN	R	0x0	when 1, system is in TX state
12	OSC_EN	R	0x1	BLE osc_en output
13	RADIO_EN	R	0x1	BLE radio_en output
14	CLK_STATUS	R	0x0	BLE status 0: BLE is active 1: BLE is in sleep mode
31:15	RESERVED	-	-	-

2.5.8 Systick timer control register

Table 10. SYS_TICK - Systick Timer Control register (offset = 0x100) bit description

Bit	Symbol	Access	Reset Value	Description
23:0	TENMS	RW	0x000147	system tick timer calibration value
24	SKEW	RW	0x1	whether THE TENMS value will generate a precise 10 millisecond time or an approximation 0:TENMS is considered to be precise 1:TENMS is not considered to be precise
25	NOREF	RW	0x0	whether an external reference clock is available 0:external reference clock is available 1:external reference clock is not available
30:26	RESERVED	-	-	-
31	EN_STCLKEN	RW	0x1	1 is enable systick

2.5.9 Exchange memory base address register

Table 11. SRAM_CTRL - Exchange memory base address register (offset = 0x104) bit description

Bit	Symbol	Access	Reset Value	Description
14:0	EM_BASE_ADDR	RW	0x2400	exchange memory base address in system memory; default value is 9k word.
31:15	RESERVED	-	-	-

2.5.10 Chip ID register

Table 12. CHIP_ID - Chip ID register (offset = 0x108) bit description

Bit	Symbol	Access	Reset Value	Description
2:0	CID0	R	0x0	CHIP ID for manufacture fab
5:3	CID1	R	0x2	CHIP ID for product family
7:6	CID2	R	0x0	with CID4 to indicate CHIP ID for chip revision 0x0: Revision A 0x1: Revision B 0x2: Revision C 0x3: Revision D 0x4: Revision E 0x5: Revision F 0x6: Revision G 0x7: Revision H .....
13:8	CID3	R	0x10	CHIP ID for product ID 000000b: QN9020 001xxxb: QN903x 010xxxb: QN908x
15:14	CID4	R	0x0	with CID2 to indicate CHIP ID for chip revision 0x0: Revision A 0x1: Revision B 0x2: Revision C 0x3: Revision D 0x4: Revision E 0x5: Revision F 0x6: Revision G 0x7: Revision H .....
25:16	RESERVED	-	-	-
26	MEM_OPTION	R	0x1	memory option indicator 0: chip memory size is 64k 1: chip memory size is 128k
27	ADC_OPTION	R	0x1	ADC option resolution indicator 0: ADC low resolution 1: ADC high resolution

Table 12. CHIP_ID - Chip ID register (offset = 0x108) *...continued*bit description

Bit	Symbol	Access	Reset Value	Description
28	FLASH_OPTION	R	0x1	Flash option indicator 0: Chip flash size is 256k 1: Chip flash size is 512k
29	FPU_OPTION	R	0x1	FPU option indicator 0: FPU does not exist 1: FPU exists
30	USB_OPTION	R	0x1	USB option indicator 0: Chip does not have USB 1: Chip has USB
31	FSP_OPTION	R	0x1	FSP option indicator 0: Chip does not have FSP 1: Chip has FSP

2.5.11 Crystal and PA register

Table 13. ANA_CTRL0 - Crystal and PA register (offset = 0x110) bit description

Bit	Symbol	Access	Reset value	Description
7:0	PA_POWER	RW	0x7F	PA power control Power: PA_POWER[7:0] 2 dbm: 0xFF 0 dbm: 0x89 –2 dbm: 0x52 –4 dbm: 0x3a –6 dbm: 0x2a –8 dbm: 0x1F –10 dbm: 0x18 –12 dbm: 0x12 –14 dbm: 0x0E –16 dbm: 0x0B –18 dbm: 0x09 –20 dbm: 0x07
19:8	RESERVED	-	-	-
21:20	XTAL_AMP	RW	0x2	crystal amplitude set register
27:22	XTAL_LOAD_CAP	RW	0x20	register-controlled load cap of the XTAL in normal mode. XTAL load cap =5pF + 0.35pF × XTAL_LOAD_CAP + 5pF × XTAL_EXTRA_CAP
28	XTAL_EXTRA_CAP	RW	0x0	add extra 16/32 MHz XTAL load cap
29	RESERVED	-	-	-
31:30	XTAL_MODE	RW	0x0	injection mode of the XTAL 0: high frequency crystal oscillator 1: inject digital clock 2: inject single-end sine wave signal 3: İnject differential sine wave signal

2.5.12 Crystal control register

Table 14. XTAL_CTRL - Crystal Control register (offset = 0x180) bit description

Bit	Symbol	Access	Reset value	Description
4:0	RESERVED	-	-	-
5	XTAL_XCUR_BOOST_REG	RW	0x0	crystal current boost regulator 1: increase 16/32 MHz crystal current
6	XTAL_BPXDLY	RW	0x0	bypass the power up delay in the crystal core
7	XTAL_BP_HYSRES_REG	RW	0x0	reduces the hysteresis voltage bypassing the degeneration resistor 1: bypass the degeneration resistor in order to reduce the hysteresis voltage
8	XTAL_XSMT_EN_REG	RW	0x0	hysteresis buffer regulator 1: enables hysteresis buffer
9	XTAL_XRDY_REG	RW	0x0	1: enables crystal ready signal by register
10	XTAL_XOUT_DIS_REG	RW	0x0	1: disable16/32 MHz XTAL clock out 0: enable 16/32 MHz XTAL clock out
11	DIV_DIFF_CLK_DIG_DIS	RW	0x1	disables differential clock of digital
15:12	RESERVED	-	-	-
21:16	XTAL_SU_CB_REG	RW	0x20	register controlled load cap of the XTAL_B in speed up mode CB = 2pF + 0.35 pF × SU_CB + 5 pF × XADD_C
23:22	RESERVED	-	-	-
29:24	XTAL_SU_CA_REG	RW	0x20	register controlled load cap of the XTAL_A in speed-up mode CA = 2 pF + 0.35 pF × SU_CA + 5pF × XADD_C
30	XTAL_INV	RW	0x0	inverse crystal clock
31	XTAL_DIV	RW	0x0	divide crystal clock when external crystal is 32 MHz 1: the SYS_CLK is 16 MHz 0: the SYS_CLK is 32 MHz this bit has no effect if external crystal is 16 MHz

2.5.13 Buck control register

Table 15. BUCK - Buck control register (offset = 0x184) bit description

Bit	Symbol	Access	Reset value	Description
0	BUCK_DRIVER_PART_EN	RW	0x1	1 to short external inductor
1	BUCK_IND_USE_EN	RW	0x1	1 to turn on buck output stage gradually
7:2	RESERVED	-	-	-

Table 15. BUCK - Buck control register (offset = 0x184) *...continued*bit description

Bit	Symbol	Access	Reset value	Description
9:8	BUCK_ISEL	RW	0x03	buck current bias control 11b: 125 % 10b: 100 % 01b: 75 % 00b: 50 %
11:10	BUCK_VREF_SEL	RW	0x02	FC current setting 00b: max current 11b: min current
13:12	BUCK_VBG_SEL	RW	0x02	Buck reference setting 11b: 1.225 V 10b: 1.2 V 01b: 1.15 V 00b: 1.05 V
15:14	RESERVED	-	-	-
20:16	BUCK_TMOS	RW	0x10	constant on time control 11111b: minimum constant on time 00000b: maximum constant on time
21	BUCK_IC	RW	0x1	1: frequency compensation versus BVDD variation 0: no frequency compensation
31:22	RESERVED	-	-	-

2.5.14 Flexcomm clock divider register

Table 16. FC_FRG - Flexcomm Clock Divider register (offset = 0x200) bit description

Bit	Symbol	Access	Reset value	Description
7:0	FRG_MULT0	RW	0xFF	flexcomm0 clock generator, Numerator of the fractional divider. MULT is equal to the programmed value
15:8	FRG_DIV0	RW	0x0	flexcomm0 clock generator, Denominator of the fractional divider. DIV is equal to the programmed value +1. Always set to 0xFF to use with the fractional baud rate generator
23:16	FRG_MULT1	RW	0xFF	flexcomm1 clock generator, Numerator of the fractional divider. MULT is equal to the programmed value
31:24	FRG_DIV1	RW	0x0	flexcomm1 clock generator, Denominator of the fractional divider. DIV is equal to the programmed value +1. Always set to 0xFF to use with the fractional baud rate generator

2.5.15 Pad pull control 0 register

Table 17. PIO_PULL_CFG0 - Pad Pull Control 0 register (offset = 0x800) bit description

Bit	Symbol	Access	Reset value	Description
1:0	PA00_PULL	RW	0x02	PA00 pull control register 00: high-Z 01: pull down 10: pull up
3:2	PA01_PULL	RW	0x02	PA01 pull control register
5:4	PA02_PULL	RW	0x02	PA02 pull control register
7:6	PA03_PULL	RW	0x02	PA03 pull control register
9:8	PA04_PULL	RW	0x02	PA04 pull control register
11:10	PA05_PULL	RW	0x02	PA05 pull control register
13:12	PA06_PULL	RW	0x02	PA06 pull control register
15:14	PA07_PULL	RW	0x02	PA07 pull control register
17:16	PA08_PULL	RW	0x02	PA08 pull control register
19:18	PA09_PULL	RW	0x02	PA09 pull control register
21:20	PA10_PULL	RW	0x02	PA10 pull control register
23:20	PA11_PULL	RW	0x02	PA11 pull control register
25:24	PA12_PULL	RW	0x02	PA12 pull control register
27:26	PA13_PULL	RW	0x02	PA13 pull control register
29:28	PA14_PULL	RW	0x02	PA14 pull control register
31:30	PA15_PULL	RW	0x02	PA15 pull control register

2.5.16 Pad pull control 1 register

Table 18. PIO_PULL_CFG1 - Pad Pull Control 1 register (offset = 0x804) bit description

Bit	Symbol	Access	Reset value	Description
1:0	PA16_PULL	RW	0x02	PA16 pull control register
3:2	PA17_PULL	RW	0x02	PA17 pull control register
5:4	PA18_PULL	RW	0x02	PA18 pull control register
7:6	PA19_PULL	RW	0x02	PA19 pull control register
9:8	PA20_PULL	RW	0x02	PA20 pull control register
11:10	PA21_PULL	RW	0x02	PA21 pull control register
13:12	PA22_PULL	RW	0x02	PA22 pull control register
15:14	PA23_PULL	RW	0x02	PA23 pull control register
17:16	PA24_PULL	RW	0x02	PA24 pull control register
19:18	PA25_PULL	RW	0x02	PA25 pull control register
21:20	PA26_PULL	RW	0x02	PA26 pull control register
23:22	PA27_PULL	RW	0x02	PA27 pull control register
25:24	PA28_PULL	RW	0x02	PA28 pull control register
27:26	PA29_PULL	RW	0x02	PA29 pull control register
29:28	PA30_PULL	RW	0x02	PA30 pull control register
31:30	PA31_PULL	RW	0x02	PA31 pull control register

2.5.17 Pad pull control 2 register

Table 19. PIO_PULL_CFG2 - Pad Pull Control 2 register (offset = 0x808) bit description

Bit	Symbol	Access	Reset value	Description
1:0	PB00_PULL	RW	0x2	PB00 pull control register
3:2	PB01_PULL	RW	0x2	PB01 pull control register
5:4	PB02_PULL	RW	0x2	PB02 pull control register
31:6	RESERVED	-	-	-

2.5.18 IO status capture register

Table 20. IO_CAP - IO Status Capture register (offset = 0x80C) bit description

Bit	Symbol	Access	Reset Value	Description
0	PIN_RETENTION	WOO	0x0	write 1 to capture pad output and output enable and the status will be saved in PIO_CAP_OE0, PIO_CAP_OE1, PIO_CAP_OUT0 and PIO_CAP_OUT1
31:1	RESERVED	-	-	-

2.5.19 Pad drive strength 0 register

Table 21. PIO_DRV_CFG0 - Pad Drive Strength 0 register (offset = 0x810) bit description

Bit	Symbol	Access	Reset value	Description
0	PA00_DRV	RW	0x0	PA00 drive strength register
1	PA01_DRV	RW	0x0	PA01 drive strength register
2	PA02_DRV	RW	0x0	PA02 drive strength register
3	PA03_DRV	RW	0x0	PA03 drive strength register
4	PA04_DRV	RW	0x0	PA04 drive strength register
5	PA05_DRV	RW	0x0	PA05 drive strength register
6	PA06_DRV	RW	0x0	PA06 drive strength register
7	PA07_DRV	RW	0x0	PA07 drive strength register
8	PA08_DRV	RW	0x0	PA08 drive strength register
9	PA09_DRV	RW	0x0	PA09 drive strength register
10	PA10_DRV	RW	0x0	PA10 drive strength register
11	PA11_DRV	RW	0x0	PA11 drive strength register
12	PA12_DRV	RW	0x0	PA12 drive strength register
13	PA13_DRV	RW	0x0	PA13 drive strength register
14	PA14_DRV	RW	0x0	PA14 drive strength register
15	PA15_DRV	RW	0x0	PA15 drive strength register
16	PA16_DRV	RW	0x0	PA16 drive strength register
17	PA17_DRV	RW	0x0	PA17 drive strength register
18	PA18_DRV	RW	0x0	PA18 drive strength register
19	PA19_DRV	RW	0x0	PA19 drive strength register
20	PA20_DRV	RW	0x0	PA20 drive strength register
21	PA21_DRV	RW	0x0	PA21 drive strength register
22	PA22_DRV	RW	0x0	PA22 drive strength register

Table 21. PIO_DRV_CFG0 - Pad Drive Strength 0 register (offset = 0x810) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
23	PA23_DRV	RW	0x0	PA23 drive strength register
24	PA24_DRV	RW	0x0	PA24 drive strength register
25	PA25_DRV	RW	0x0	PA25 drive strength register
26	PA26_DRV	RW	0x0	PA26 drive strength register
27	PA27_DRV	RW	0x0	PA27 drive strength register
28	PA28_DRV	RW	0x0	PA28 drive strength register
29	PA29_DRV	RW	0x0	PA29 drive strength register
30	PA30_DRV	RW	0x0	PA30 drive strength register
31	PA31_DRV	RW	0x0	PA31 drive strength register

2.5.20 Pad drive strength 1 register

Table 22. PIO_DRV_CFG1 - Pad Drive Strength 1 register (offset = 0x814) bit description

Bit	Symbol	Access	Reset Value	Description
0	PB00_DRV	RW	0x0	PB00 drive strength register
1	PB01_DRV	RW	0x0	PB01 drive strengh register
2	PB02_DRV	RW	0x0	PB02 drive strengh register
31:3	RESERVED	-	-	-

2.5.21 Pad drive extra register

Table 23. PIO_DRV_CFG2 - Pad Drive Extra register (offset = 0x818) bit description

Bit	Symbol	Access	Reset value	Description
5:0	RESERVED	-	-	-
6	PA06_DRV_EXTRA	RW	0x0	write 1 to enable extra driven on PA06
10:7	RESERVED	-	-	-
11	PA11_DRV_EXTRA	RW	0x0	write 1 to enable extra driven on PA11
18:12	RESERVED	-	-	-
19	PA19_DRV_EXTRA	RW	0x0	write 1 to enable extra driven on PA19
25:20	RESERVED	-	-	-
26	PA26_DRV_EXTRA	RW	0x0	write 1 to enable extra driven on PA26
27	PA27_DRV_EXTRA	RW	0x0	write 1 to enable extra driven on PA27
31:28	RESERVED	-	-	-

2.5.22 Pad misc control register

Table 24. PIO_CFG_MISC - Pin Misc Control register (offset = 0x81C) bit description

Bit	Symbol	Access	Reset value	Description
0	PB00_AE	RW	0x1	enable PB00 analog function
1	PB01_AE	RW	0x1	enable PB01 analog function
14:2	RESERVED	-	-	-
15	PSYNC	RW	0x0	when 1, bypass first stage of synchronization of DMA pin trigger

Table 24. PIO_CFG_MISC - Pin Misc Control register (offset = 0x81C) *…continued*bit description

Bit	Symbol	Access	Reset value	Description
16	PB02_MODE	RW	0x0	chip mode pin function select 0: PB02 is used as chip mode input 1: PB02 is used as antenna output
18	TRX_EN_INV	RW	0x0	inverse TX_EN & RX_EN pin MUX output polarity
19	RFE_INV	RW	0x0	inverse RFE polarity
31:20	RESERVED	-	-	-

2.5.23 Pin wake-up polarity register 0

Table 25. PIO_WAKEUP_LVL0 - Pin Wake-up Polarity register 0(offset = 0x820) bit description

Bit	Symbol	Access	Reset value	Description
0	PA00_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA00 in sleep mode 0: High-level wake up 1: Low-level wake up
1	PA01_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA01 in sleep mode
2	PA02_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA02 in sleep mode
3	PA03_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA03 in sleep mode
4	PA04_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA04 in sleep mode
5	PA05_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA05 in sleep mode
6	PA06_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA06 in sleep mode
7	PA07_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA07 in sleep mode
8	PA08_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA08 in sleep mode
9	PA09_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA09 in sleep mode
10	PA10_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA10 in sleep mode
11	PA11_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA11 in sleep mode
12	PA12_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA12 in sleep mode
13	PA13_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA13 in sleep mode
14	PA14_WAKEUP_LVL	RW	0x0	Control the wake up polarity of PA14 in sleep mode.
15	PA15_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA15 in sleep mode

Table 25. PIO_WAKEUP_LVL0 - Pin Wake-up Polarity register 0(offset = 0x820) bit description *...continued*

Bit	Symbol	Access	Reset value	Description
16	PA16_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA16 in sleep mode
17	PA17_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA17 in sleep mode
18	PA18_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA18 in sleep mode
19	PA19_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA19 in sleep mode
20	PA20_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA20 in sleep mode
21	PA21_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA21 in sleep mode
22	PA22_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA22 in sleep mode
23	PA23_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA23 in sleep mode
24	PA24_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA24 in sleep mode
25	PA25_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA25 in sleep mode
26	PA26_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA26 in sleep mode
27	PA27_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA27 in sleep mode
28	PA28_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA28 in sleep mode
29	PA29_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA29 in sleep mode
30	PA30_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA30 in sleep mode
31	PA31_WAKEUP_LVL	RW	0x0	control the wake up polarity of PA31 in sleep mode

2.5.24 Pin wake-up polarity register 1

Table 26. PIO_WAKEUP_LVL1 - Pin Wake-up Polarity register 1(offset = 0x824) bit description

Bit	Symbol	Access	Reset Value	Description
0	PB00_WAKEUP_LVL	RW	0x0	control the wake up polarity of PB01 in sleep mode
1	PB01_WAKEUP_LVL	RW	0x0	control the wake up polarity of PB02 in sleep mode
2	PB02_WAKEUP_LVL	RW	0x0	control the wake up polarity of PB03 in sleep mode
31:3	RESERVED	-	-	-

2.5.25 Pad input enable register 0

Table 27. PIO_IE_CFG0 - Pad Input Enable register 0(offset = 0x828) bit description

Bit	Symbol	Access	Reset value	Description
0	PA00_IE	RW	0x1	PA00 digital input enable
1	PA01_IE	RW	0x1	PA01 digital input enable
2	PA02_IE	RW	0x1	PA02 digital input enable
3	PA03_IE	RW	0x1	PA03 digital input enable
4	PA04_IE	RW	0x1	PA04 digital input enable
5	PA05_IE	RW	0x1	PA05 digital input enable
6	PA06_IE	RW	0x1	PA06 digital input enable
7	PA07_IE	RW	0x1	PA07 digital input enable
8	PA08_IE	RW	0x1	PA08 digital input enable
9	PA09_IE	RW	0x1	PA09 digital input enable
10	PA10_IE	RW	0x1	PA10 digital input enable
11	PA11_IE	RW	0x1	PA11 digital input enable
12	PA12_IE	RW	0x1	PA12 digital input enable
13	PA13_IE	RW	0x1	PA13 digital input enable
14	PA14_IE	RW	0x1	PA14 digital input enable
15	PA15_IE	RW	0x1	PA15 digital input enable
16	PA16_IE	RW	0x1	PA16 digital input enable
17	PA17_IE	RW	0x1	PA17 digital input enable
18	PA18_IE	RW	0x1	PA18 digital input enable
19	PA19_IE	RW	0x1	PA19 digital input enable
20	PA20_IE	RW	0x1	PA20 digital input enable
21	PA21_IE	RW	0x1	PA21 digital input enable
22	PA22_IE	RW	0x1	PA22 digital input enable
23	PA23_IE	RW	0x1	PA23 digital input enable
24	PA24_IE	RW	0x1	PA24 digital input enable
25	PA25_IE	RW	0x1	PA25 digital input enable
26	PA26_IE	RW	0x1	PA26 digital input enable
27	PA27_IE	RW	0x1	PA27 digital input enable
28	PA28_IE	RW	0x1	PA28 digital input enable
29	PA29_IE	RW	0x1	PA29 digital input enable
30	PA30_IE	RW	0x1	PA30 digital input enable
31	PA31_IE	RW	0x1	PA31 digital input enable

2.5.26 Pad input enable register 1

Table 28. PIO_IEN_CFG1 - Pad Input Enable register 1(offset = 0x82C) bit description

Bit	Symbol	Access	Reset value	Description
0	PB00_IE	RW	0x1	PB00 digital input enable

Table 28. PIO_IEN_CFG1 - Pad Input Enable register 1(*…continued*offset = 0x82C) bit description

Bit	Symbol	Access	Reset value	Description
1	PB01_IE	RW	0x1	PB01 digital input enable
2	BOOT_MODE_IE	RW	0x1	PB02 input enable
31:3	RESERVED	-	-	-

2.5.27 Pin MUX control register 0

Table 29. PIO_FUNC_CFG0 - Pin MUX Control register 0(offset = 0x830) bit description

Bit	Symbol	Access	Reset value	Description[1]
2:0	PA00_FUNC	RW	0x0	PA00 function control register
3	RESERVED	-	-	-
6:4	PA01_FUNC	RW	0x0	PA01 function control register
7	RESERVED	-	-	-
10:8	PA02_FUNC	RW	0x0	PA02 function control register
11	RESERVED	-	-	-
14:12	PA03_FUNC	RW	0x0	PA03 function control register
15	RESERVED	-	-	-
18:16	PA04_FUNC	RW	0x0	PA04 function control register
19	RESERVED	-	-	-
22:20	PA05_FUNC	RW	0x0	PA05 function control register
23	RESERVED	-	-	-
26:24	PA06_FUNC	RW	0x0	PA06 function control register
27	RESERVED	-	-	-
30:28	PA07_FUNC	RW	0x0	PA07 function control register
31	RESERVED	-	-	-

[1] For available functions on PA01~07, see Table 95.

2.5.28 Pin MUX control register 1

Table 30. PIO_FUNC_CFG1 - Pin MUX Control register 1(offset = 0x834) bit description

Bit	Symbol	Access	Reset value	Description[1]
2:0	PA08_FUNC	RW	0x0	PA08 function control register
3	RESERVED	-	-	-
6:4	PA09_FUNC	RW	0x0	PA09 function control register
7	RESERVED	-	-	-
10:8	PA10_FUNC	RW	0x0	PA10 function control register
11	RESERVED	-	-	-
14:12	PA11_FUNC	RW	0x0	PA11 function control register
15	RESERVED	-	-	-
18:16	PA12_FUNC	RW	0x0	PA12 function control register
19	RESERVED	-	-	-
22:20	PA13_FUNC	RW	0x0	PA13 function control register
23	RESERVED	-	-	-
26:24	PA14_FUNC	RW	0x0	PA14 function control register

Table 30. PIO_FUNC_CFG1 - Pin MUX Control register 1(*…continued*offset = 0x834) bit description

Bit	Symbol	Access	Reset value	Description[1]
27	RESERVED	-	-	-
30:28	PA15_FUNC	RW	0x0	PA15 function control register
31	RESERVED	-	-	-

[1] For available functions on PA08~15, see Table 95.

2.5.29 Pin MUX control register 2

Table 31. PIO_FUNC_CFG2 - Pin MUX Control register 2(offset = 0x838) bit description

Bit	Symbol	Access	Reset value	Description[1]
2:0	PA16_FUNC	RW	0x0	PA16 function control register
3	RESERVED	-	-	-
6:4	PA17_FUNC	RW	0x0	PA17 function control register
7	RESERVED	-	-	-
10:8	PA18_FUNC	RW	0x0	PA18 function control register
11	RESERVED	-	-	-
14:12	PA19_FUNC	RW	0x0	PA19 function control register
15	RESERVED	-	-	-
18:16	PA20_FUNC	RW	0x0	PA20 function control register
19	RESERVED	-	-	-
22:20	PA21_FUNC	RW	0x0	PA21 function control register
23	RESERVED	-	-	-
26:24	PA22_FUNC	RW	0x0	PA22 function control register
27	RESERVED	-	-	-
30:28	PA23_FUNC	RW	0x0	PA23 function control register
31	RESERVED	-	-	-

[1] For available functions on PA16~23, see Table 95.

2.5.30 Pin MUX control register 3

Table 32. PIO_FUNC_CFG3 - Pin MUX Control register 3(offset = 0x83C) bit description

Bit	Symbol	Access	Reset Value	Description[1]
2:0	PA24_FUNC	RW	0x0	PA24 function control register
3	RESERVED	-	-	-
6:4	PA25_FUNC	RW	0x0	PA25 function control register
7	RESERVED	-	-	-
10:8	PA26_FUNC	RW	0x0	PA26 function control register
11	RESERVED	-	-	-
14:12	PA27_FUNC	RW	0x0	PA27 function control register
15	RESERVED	-	-	-
18:16	PA28_FUNC	RW	0x0	PA28 function control register
19	RESERVED	-	-	-
22:20	PA29_FUNC	RW	0x0	PA29 function control register

Table 32. PIO_FUNC_CFG3 - Pin MUX Control register 3(*...continued*offset = 0x83C) bit description

Bit	Symbol	Access	Reset Value	Description[1]
23	RESERVED	-	-	-
26:24	PA30_FUNC	RW	0x0	PA30 function control register
27	RESERVED	-	-	-
30:28	PA31_FUNC	RW	0x0	PA31 function control register
31	RESERVED	-	-	-

[1] For available functions on PA24~31, see Table 95.

2.5.31 Pin function selection in sleep mode register 0

Table 33. PIO_WAKEUP_EN0 - Pin Function Selection in Sleep Mode register 0(offset = 0x840) bit description

Bit	Symbol	Access	Reset value	Description
0	PA00_WAKEUP_EN	RW	0x0	control GPIOA[0] as wakeup source
1	PA01_WAKEUP_EN	RW	0x0	control GPIOA[1] as wakeup source
2	PA02_WAKEUP_EN	RW	0x0	control GPIOA[2] as wakeup source
3	PA03_WAKEUP_EN	RW	0x0	control GPIOA[3] as wakeup source
4	PA04_WAKEUP_EN	RW	0x0	control GPIOA[4] as wakeup source
5	PA05_WAKEUP_EN	RW	0x0	control GPIOA[5] as wakeup source
6	PA06_WAKEUP_EN	RW	0x0	control GPIOA[6] as wakeup source
7	PA07_WAKEUP_EN	RW	0x0	control GPIOA[7] as wakeup source
8	PA08_WAKEUP_EN	RW	0x0	control GPIOA[8] as wakeup source
9	PA09_WAKEUP_EN	RW	0x0	control GPIOA[9] as wakeup source
10	PA10_WAKEUP_EN	RW	0x0	control GPIOA[10] as wakeup source
11	PA11_WAKEUP_EN	RW	0x0	control GPIOA[11] as wakeup source
12	PA12_WAKEUP_EN	RW	0x0	control GPIOA[12] as wakeup source
13	PA13_WAKEUP_EN	RW	0x0	control GPIOA[13] as wakeup source
14	PA14_WAKEUP_EN	RW	0x0	control GPIOA[14] as wakeup source
15	PA15_WAKEUP_EN	RW	0x0	control GPIOA[15] as wakeup source
16	PA16_WAKEUP_EN	RW	0x0	control GPIOA[16] as wakeup source
17	PA17_WAKEUP_EN	RW	0x0	control GPIOA[17] as wakeup source
18	PA18_WAKEUP_EN	RW	0x0	control GPIOA[18] as wakeup source
19	PA19_WAKEUP_EN	RW	0x0	control GPIOA[19] as wakeup source
20	PA20_WAKEUP_EN	RW	0x0	control GPIOA[20] as wakeup source
21	PA21_WAKEUP_EN	RW	0x0	control GPIOA[21] as wakeup source
22	PA22_WAKEUP_EN	RW	0x0	control GPIOA[22] as wakeup source
23	PA23_WAKEUP_EN	RW	0x0	control GPIOA[23] as wakeup source
24	PA24_WAKEUP_EN	RW	0x0	control GPIOA[24] as wakeup source
25	PA25_WAKEUP_EN	RW	0x0	control GPIOA[25] as wakeup source
26	PA26_WAKEUP_EN	RW	0x0	control GPIOA[26] as wakeup source
27	PA27_WAKEUP_EN	RW	0x0	control GPIOA[27] as wakeup source
28	PA28_WAKEUP_EN	RW	0x0	control GPIOA[28] as wakeup source

Table 33. PIO_WAKEUP_EN0 - Pin Function Selection in Sleep Mode register 0(*…continued*offset = 0x840) bit

Bit	Symbol	Access	Reset value	Description
29	PA29_WAKEUP_EN	RW	0x0	control GPIOA[29] as wakeup source
30	PA30_WAKEUP_EN	RW	0x0	control GPIOA[30] as wakeup source
31	PA31_WAKEUP_EN	RW	0x0	control GPIOA[31] as wakeup source

2.5.32 Pin function selection in sleep mode register 1

Table 34. PIO_WAKEUP_EN1 - Pin Function Selection in Sleep Mode register 1(offset = 0x844) bit description

Bit	Symbol	Access	Reset value	Description
0	PB00_WAKEUP_EN	RW	0x0	control PB00 in GPIOB as wake-up source
1	PB01_WAKEUP_EN	RW	0x0	control PB01 in GPIOB as wake-up source
2	PB02_WAKEUP_EN	RW	0x0	control PB02 in GPIOB as wake-up source
3	RESERVED	-	-	-
4	PA04_32K_OE	RW	0x0	32k clock output enable. When this bit is set to 1, PA04 will output 32k clock. At this time, PIN_CTRL register is not effective to control this IO’s function.
5	PA05_XTAL_OE	RW	0x0	XTAL clock output enable. When this bit is set to 1, PA05 will output XTAL clock. At this time, PIN_CTRL register is not effective to control this IO’s function.
9:6	RESERVED	-	-	-
10	PA10_32K_OE	RW	0x0	32K clock output enable. When this bit is set to 1, PA10 (GPIO10) will output 32k clock. At this time, PIN_CTRL register is not effective to control this IO’s function.
11	PA11_XTAL_OE	RW	0x0	XTAL clock output enable. When this bit is set to 1, PA11 will output XTAL clock. At this time, PIN_CTRL register is not effective to control this IO’s function.
17:12	-	-	-	Reserved
18	PA18_32K_OE	RW	0x0	32K clock output enable. When this bit is set to 1, PA18 will output 32k clock. At this time, PIN_CTRL register is not effective to control this IO’s function.
19	PA19_XTAL_OE	RW	0x0	XTAL clock output enable. When this bit is set to 1, PA19 will output XTAL clock. At this time, PIN_CTRL register is not effective to control this IO’s function.
23:20	RESERVED	-	-	-
24	PA24_32K_OE	RW	0x0	32K clock output enable. When this bit is set to 1, PA24 will output 32k clock. At this time, PIN_CTRL register is not effective to control this IO’s function

Table 34. PIO_WAKEUP_EN1 - Pin Function Selection in Sleep Mode register 1(offset = 0x844) bit description

Bit	Symbol	Access	Reset value	Description
25	PA25_XTAL_OE	RW	0x0	XTAL clock output enable. When this bit is set to 1, PA25 will output XTAL clock. At this time, PIN_CTRL register is not effective to control this IO's function
30:26	RESERVED	-	-	-
31	PDM_IO_SEL	RW	0x0	Pin status selection in power down mode;1: IO status include output and output enable use the ones that captured by writing 1 to IO_CAP. 0: IO status include output and output enable depends on PIO_CFGx register

2.5.33 Pin captured output enable status register 0

Table 35. PIO_CAP_OE0 - Pin Captured Output Enable Status register 0 (offset = 0x848) bit description

Bit	Symbol	Access	Reset Value	Description
0	PA00_CAP_OE	R	0x0	PA00 output enable status
1	PA01_CAP_OE	R	0x0	PA01 output enable status
2	PA02_CAP_OE	R	0x0	PA02 output enable status
3	PA03_CAP_OE	R	0x0	PA03 output enable status
4	PA04_CAP_OE	R	0x0	PA04 output enable status
5	PA05_CAP_OE	R	0x0	PA05 output enable status
6	PA06_CAP_OE	R	0x0	PA06 output enable status
7	PA07_CAP_OE	R	0x0	PA07 output enable status
8	PA08_CAP_OE	R	0x0	PA08 output enable status
9	PA09_CAP_OE	R	0x0	PA09 output enable status
10	PA10_CAP_OE	R	0x0	PA10 output enable status
11	PA11_CAP_OE	R	0x0	PA11 output enable status
12	PA12_CAP_OE	R	0x0	PA12 output enable status
13	PA13_CAP_OE	R	0x0	PA13 output enable status
14	PA14_CAP_OE	R	0x0	PA14 output enable status
15	PA15_CAP_OE	R	0x0	PA15 output enable status
16	PA16_CAP_OE	R	0x0	PA16 output enable status
17	PA17_CAP_OE	R	0x0	PA17 output enable status
18	PA18_CAP_OE	R	0x0	PA18 output enable status
19	PA19_CAP_OE	R	0x0	PA19 output enable status
20	PA20_CAP_OE	R	0x0	PA20 output enable status
21	PA21_CAP_OE	R	0x0	PA21 output enable status
22	PA22_CAP_OE	R	0x0	PA22 output enable status
23	PA23_CAP_OE	R	0x0	PA23 output enable status
24	PA24_CAP_OE	R	0x0	PA24 output enable status
25	PA25_CAP_OE	R	0x0	PA25 output enable status
26	PA26_CAP_OE	R	0x0	PA26 output enable status
27	PA27_CAP_OE	R	0x0	PA27 output enable status

Table 35. PIO_CAP_OE0 - Pin Captured Output Enable Status register 0 (offset = 0x848) bit description *...continued*

Bit	Symbol	Access	Reset Value	Description
28	PA28_CAP_OE	R	0x0	PA28 output enable status
29	PA29_CAP_OE	R	0x0	PA29 output enable status
30	PA30_CAP_OE	R	0x0	PA30 output enable status
31	PA31_CAP_OE	R	0x0	PA31 output enable status

2.5.34 Pin output enable status register 1

Table 36. PIO_CAP_OE1 - Pin Captured Output Enable Status register 1(offset = 0x84C) bit description

Bit	Symbol	Access	Reset Value	Description
0	PB00_CAP_OE	R	0x0	PB00 output enable status
1	PB01_CAP_OE	R	0x0	PB01 output enable status
2	PB02_CAP_OE	R	0x0	PB02 output enable status
31:3	RESERVED	-	-	-

2.5.35 Pin captured status register 0

Table 37. PIO_CAP_OUT0 - Pin Captured Output Status register 0(offset = 0x850) bit description

Bit	Symbol	Access	Reset value	Description
0	PA00_CAP_OUT	R	0x0	PA00 output status
1	PA01_CAP_OUT	R	0x0	PA01 output status
2	PA02_CAP_OUT	R	0x0	PA02 output status
3	PA03_CAP_OUT	R	0x0	PA03 output status
4	PA04_CAP_OUT	R	0x0	PA04 output status
5	PA05_CAP_OUT	R	0x0	PA05 output status
6	PA06_CAP_OUT	R	0x0	PA06 output status
7	PA07_CAP_OUT	R	0x0	PA07 output status
8	PA08_CAP_OUT	R	0x0	PA08 output status
9	PA09_CAP_OUT	R	0x0	PA09 output status
10	PA10_CAP_OUT	R	0x0	PA10 output status
11	PA11_CAP_OUT	R	0x0	PA11 output status
12	PA12_CAP_OUT	R	0x0	PA12 output status
13	PA13_CAP_OUT	R	0x0	PA13 output status
14	PA14_CAP_OUT	R	0x0	PA14 output status
15	PA15_CAP_OUT	R	0x0	PA15 output status
16	PA16_CAP_OUT	R	0x0	PA16 output status
17	PA17_CAP_OUT	R	0x0	PA17 output status
18	PA18_CAP_OUT	R	0x0	PA18 output status
19	PA19_CAP_OUT	R	0x0	PA19 output status
20	PA20_CAP_OUT	R	0x0	PA20 output status
21	PA21_CAP_OUT	R	0x0	PA21 output status
22	PA22_CAP_OUT	R	0x0	PA22 output status
23	PA23_CAP_OUT	R	0x0	PA23 output status
24	PA24_CAP_OUT	R	0x0	PA24 output status

Table 37. PIO_CAP_OUT0 - Pin Captured Output Status register 0(offset = 0x850) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
25	PA25_CAP_OUT	R	0x0	PA25 output status
26	PA26_CAP_OUT	R	0x0	PA26 output status
27	PA27_CAP_OUT	R	0x0	PA27 output status
28	PA28_CAP_OUT	R	0x0	PA28 output status
29	PA29_CAP_OUT	R	0x0	PA29 output status
30	PA30_CAP_OUT	R	0x0	PA30 output status
31	PA31_CAP_OUT	R	0x0	PA31 output status

2.5.36 Pin captured output register 1

Table 38. PIO_CAP_OUT1 - Pin Captured Output Status register 1(offset = 0x854) bit description

Bit	Symbol	Access	Reset value	Description
0	PB00_CAP_OUT	R	0x0	PB00 output status
1	PB01_CAP_OUT	R	0x0	PB01 output status
2	PB02_CAP_OUT	R	0x0	PB02 output status
31:3	RESERVED	-	-	-

2.5.37 Reset source status register

Table 39. RST_CAUSE_SRC - Reset Source Status register (offset = 0x858) bit description

Bit	Symbol	Access	Reset value	Description
8:0	RESET_CAUSE	R	0x0	reset source indicator. xxxxxxxx1b: Power-on Reset; xxxxxxx1xb: Brown-out Reset; xxxxxx1xxb: External pin Reset; xxxxx1xxxb: Watchdog Reset; xxxx1xxxxb: Lock Up Reset; xxx1xxxxxb: Reboot Reset; xx1000000b: CPU system Reset requirement; x10000000b: Wake Up reset 10000000b: CPU software Reset;
30:9	RESERVED	-	-	-
31	RST_CAUSE_CLR	WOO	0x0	Write'1' clear RESET_CAUSE bits;

2.5.38 Power management unit control register 0

Table 40. PMU_CTRL0 - Power Management Unit Control register 0 (offset = 0x85C) bit description

Bit	Symbol	Access	Reset value	Description
0	MEM0_DIS	RW	0x0	power control for SRAM memory block 0 1: power down 0: power on
1	MEM1_DIS	RW	0x0	power control for SRAM memory block 1
2	MEM2_DIS	RW	0x0	power control for SRAM memory block 2

Table 40. PMU_CTRL0 - Power Management Unit Control register 0 (offset = 0x85C) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
3	MEM3_DIS	RW	0x0	power control for SRAM memory block 3
4	MEM4_DIS	RW	0x0	power control for SRAM memory block 4
5	MEM5_DIS	RW	0x0	power control for SRAM memory block 5
6	MEM6_DIS	RW	0x0	power control for SRAM memory block 6
7	MEM7_DIS	RW	0x0	power control for SRAM memory block 7
8	MEM8_DIS	RW	0x0	power control for SRAM memory block 8
9	MEM9_DIS	RW	0x0	power control for SRAM memory block 9
10	RESERVED	-	-	-
16	BLE_DIS	RW	0x0	1: power down BLE 0: power on BLE
17	FIR_DIS	RW	0x0	1: power down FIR buffer 0: power on FIR buffer
18	FSP_DIS	RW	0x0	1: power down FSP,USB,CRC and SPIFI 0: power on SP,USB,CRC and SPIFI
19	RESERVED	-	-	-
20	MCU_MODE	RW	0x0	1: power down BG, V2I, VREG_A, VREG_D 0: power on BG, V2I, VREG_A, VREG_D
25:21	RESERVED	-	-	-
26	OSC_INT_EN	RW	0x0	1 to enable OSC_EN as interrupt and wake up source
27	RTC_SEC_WAKEUP_EN	RW	0x0	1 to enable RTC interrupt as wakeup source
28	WAKEUP_EN	RW	0x0	1 to enable sleep wake up source
29	PMU_EN	RW	0x0	1 to enable chip power down mode
30	RETENTION_EN	RW	0x0	1 to enable all CPU registers to be retained in power down mode
31	BOND_EN	RW	0x1	1 to enable FSP option

2.5.39 Power management unit control register 1

Table 41. PMU_CTRL1 - Power Management Unit Control register 1(offset = 0x860) bit description

Bit	Symbol	Access	Reset value	Description
0	RCO32K_DIS	RW	0x0	1: power down 32k RCO 0: power on 32k RCO
1	XTAL32K_DIS	RW	0x0	1: power down 32 kHz XTAL 0: power on 32 kHz XTAL
2	XTAL_DIS	RW	0x0	1: power down 16/32 MHz XTAL 0: power on 16/32 MHz XTAL
3	OSC32M_DIS	RW	0x0	1: power down OSC 0: OSC power is controlled by analog circuits
4	USBPLL_DIS	RW	0x1	1: power down USB 48M PLL 0: power on USB 48M PLL
5	ADC_BUF_DIS	RW	0x1	1: power down buffer in SD ADC 0: power on buffer in SD ADC
6	ADC_BG_DIS	RW	0x1	1: power down bandgap in SD ADC 0: power on bandgap in SD ADC
7	ADC_DIS	RW	0x1	1: power down SD ADC 0: power on SD ADC
8	ADC_VCM_DIS	RW	0x1	1: power down VCM DRV in SD ADC 0: power on VCM DRV in SD ADC
9	ADC_VREF_DIS	RW	0x1	1: power down VREF DRV in SD ADC 0: power on VREF DRV in SD ADC
10	DAC_DIS	RW	0x1	1: power down DAC 0: power on DAC
11	CAP_SEN_DIS	RW	0x1	1: power down cap sensor 0: power on cap sensor
15:12	RESERVED	-	-	-
19:16	BUCK_CTRL	RW	0x0	BUCK power control 0x0: power on BUCK 0xF: power down BUCK
29:20	RESERVED	-	-	-
30	RCO32K_PDM_DIS	RW	0x0	in sleep mode, this bit ORs with RCO32K_DIS to control the RCO 32K power 1: power down RCO 0: power on RCO
31	XTAL32K_PDM_DIS	RW	0x0	in sleep mode, this bit ORs with XTAL32K_DIS to control the XTAL32K power 1: power down XTAL32 0: power on XTAL32

2.5.40 Analog setting register

Table 42. ANA_EN - Analog Setting register (offset = 0x864) bit description

Bit	Symbol	Access	Reset value	Description
0	BOD_AMP_EN	RW	0x0	enable the AMP of brown-out detector
1	BOD_EN	RW	0x0	enable brown-out detector
2	BAT_MON_EN	RW	0x0	enable battery monitor
3	ACMP0_EN	RW	0x0	enable comparator 0
4	ACMP1_EN	RW	0x0	enable comparator 1
5	BOR_AMP_EN	RW	0x0	enable the AMP of brown-down reset detector
6	BOR_EN	RW	0x0	enable brown-down reset detector
7	RESERVED	-	-	-
11:8	ACMP0_REF	RW	0x0	acmp0 reference voltage selection; 0000b: Select external reference voltage; other values: vref0=Acmp_vref × ACMP0_REF/16
15:12	ACMP1_REF	RW	0x0	acmp1 reference voltage selection; 0000b: Select external reference voltage; other values: vref1=Acmp_vref × ACMP1_REF/16
16	ACMP0_HYST_EN	RW	0x0	hysteresis enable of ACMP0 when 1
17	ACMP1_HYST_EN	RW	0x0	hysteresis enable of ACMP1 when 1
18	ACMP_VREF_SEL	RW	0x0	Acmp_vref selection 0: internal band-gap voltage (VBG) 1: external VCC
20:19	BOD_THR	RW	0x0	browned-out detector threshold voltages 00b: 2.06 V 01b: 2.45 V 10b: 2.72 V 11b: 3.04 V
22:21	BOR_THR	RW	0x0	browned-out reset threshold voltages 00b: 1.5 V 01b: 1.85 V 10b: 2 V 11b: 3 V
23	RESERVED	-	-	-
24	ACMP0_VALUE	R	0x0	comparator 0 output
25	ACMP1_VALUE	R	0x0	comparator 1 output
27:26	ACMP0_EDGE_SEL	RW	0x0	ACMP0 interrupt edge selection 00b: positive edge 01b: negative edge 10b: both edges

Table 42. ANA_EN - Analog Setting register (offset = 0x864) *…continued***bit description**

Bit	Symbol	Access	Reset value	Description
29:28	ACMP1_EDGE_SEL	RW	0x0	ACMP1 interrupt edge selection
30	ACMP0_INTEN	RW	0x0	1 to enable ACMP0 interrupt
31	ACMP1_INTEN	RW	0x0	1 to enable ACMP1 interrupt

2.5.41 32k crystal control register

Table 43. XTAL32K_CTRL - 32k Crystal Control (offset = 0x868) bit description

Bit	Symbol	Access	Reset value	Description
5:0	XTAL32K_ICTRL	RW	0x23	XTAL 32 gm cell current bias Y
7:6	XTAL32K_INJ	RW	0x0	XTAL 32 kHz clock injection mode 00b: on-chip oscillator 01b: external digital clock 1xb: external sine wave clock
13:8	XTAL32K_LOAD_CAP	RW	0x30	load cap selection of XTAL32K; XTAL32K load cap = 3.6 pF + 0.4 pF × XTAL32K_LOAD_CAP + 6.4pF × XTAL32K_EXTRA_CAP
14	XTAL32K_EXTRA_CAP	RW	0x0	add extra XTAL32K load cap or not.
31:15	RESERVED	-	-	-

2.5.42 USB configuration register

Table 44. USB_CFG - USB Configuration register (offset = 0x86C) bit description

Bit	Symbol	Access	Reset Value	Description
0	DPPUEN_B_PHY_POL	RW	0x0	drive high to inverse the polarity of the connection; see Table 287
1	DPPUEN_B_PHY_SEL	RW	0x1	control source selection for pull-up resister (connect or disconnect) 0 = connection controlled by register 1 = connection controlled by USB controller See Table 287
2	RESERVED	-	-	-
3	USB_VBUS	RW	0x0	USB voltage connection; see Table 287
4	USB_PHYSTDBY	RW	0x1	When USB_PHYSTDBY_WEN is 1, this bit controls USB PHY power 0 = USB PHY power is turned on 1 = USB PHY power is turned off
5	USB_PHYSTDBY_WEN	RW	0x1	0 = use USB controller to control USB PHY power 1 = use USB PHYSTDBY register to control USB PHY power
31:6	RESERVED	-	-	-

2.5.43 Power management unit control register 2

Table 45. PMU_CTRL2 - Power Management Unit Control register 2 (offset = 0x880) bit description

Bit	Symbol	Access	Reset value	Description
0	BG_PDM_DIS	RW	0x1	1 to power down bandgap in power down mode
1	V2I_PDM_DIS	RW	0x1	1 to power down V2I in power down mode
2	VREG_A_PDM_DIS	RW	0x1	1 to power down VREG_A in power down mode
3	VREG_D_PDM_DIS	RW	0x1	1 to power down VREG_D in power down mode
4	XTAL_PDM_DIS	RW	0x1	1 to power down XTAL in power down mode
5	OSC32M_PDM_DIS	RW	0x1	1 to power down 32M oscillator in power down mode
6	RFAGC_ON	RW	0x1	1 to enable RFAGC
7	RX_EN_SEL	RW	0x1	RX_EN width selection
8	BG_DIS	RW	0x0	1 to switch off bandgap power
9	V2I_DIS	RW	0x0	1 to switch off V2I power
10	VREG_A_DIS	RW	0x0	1 to switch off VREG_A power
11	VREG_D_DIS	RW	0x0	1 to switch off VREG_D power
12	LO_DIS	RW	0x1	1 to switch off LO power
13	VCO_DIS	RW	0x1	1 to switch off VCO power
14	PA_PK_DIS	RW	0x1	1 to switch off PA peek detector power
15	PA_DIS	RW	0x1	1 to switch off PA power
16	LNA_DIS	RW	0x1	1 to switch off LNA power
16	MIXER_DIS	RW	0x1	1 to switch off MIXER power
18	PKDET_DIS	RW	0x1	1 to switch off RRF and PPF peek detector power
19	PPF_DIS	RW	0x1	1 to switch off PPF power
20	SAR_DIS	RW	0x1	1 to switch off SAR ADC power
21	RC_CAL_DIS	RW	0x1	1 to switch off RCCAL power
28:22	RESERVED	-	0x0	-
29	FLSH_DIS	RW	0x0	1 to switch off flash power
30	FLSH_PDM_DIS	RW	0x1	1 to power down flash VDD25 in power down mode
31	SEL_PD	RW	0x0	power control selection 0 = HW control power 1 = SW control power

2.5.44 IVREF and DVREG setting register

Table 46. ANA_CTRL1 - IVREF and DVREG setting register (offset = 0x884) bit description

Bit	Symbol	Access	Reset value	Description
1:0	VDD_PMU_SET_PDM	RW	0x0	Vdd_pmu while in power down
3:2	VDD_PMU_SET	RW	0x0	Vdd_pmu while wakeup
5:4	VDD_MEM_SET_PDM	RW	0x0	Vdd_mem while in power down mode
7:6	VDD_MEM_SET	RW	0x0	Vdd_mem while wakeup
8	VDD_PMU_SET_EXTRA	RW	0x0	extra high setting for vdd_pmu
9	VDD_MEM_SET_EXTRA	RW	0x0	extra high setting for vdd_mem

Table 46. ANA_CTRL1 - IVREF and DVREG setting register (offset = 0x884) *...continued*bit description

Bit	Symbol	Access	Reset value	Description
10	VDD_PMU_SET_ULTRA_LOW	RW	0x0	ultra low setting for vdd_pmu
11	VDD_PMU_MEM_SW	RW	0x0	1 to close the switch between vdd_omu and vdd_mem
15:12	IV_BG_SEL	RW	0x8	VBG voltage select
16	PDM_DIS_BUCK	RW	0x1	1 to power off buck in power down mode
17	BUCK_PD_CCM	RW	0x0	0 buck in CCM mode
18	BUCK_PD_DCM	RW	0x0	0 buck in DCM mode
20:19	IV_IREF_SEL	RW	0x3	Reference current select
23:21	IV_VREG11_SET	RW	0x4	VREG11 setting
24	XTAL32K_FORCE_RDY	RW	0x0	XTAL32K ready from register
25	X32_SMT_EN	RW	0x0	1 to enable schmidt trigger in XTAL32K
27:26	BM_X32BUF	RW	0x3	XTAL32K buffer current bias
30:28	DVREG11_SET_DIG	RW	0x1	Vregd set
31	BUCK_DPD	RW	0x0	ZC control select

2.5.45 MISC register

Table 47. MISC - MISC register (offset = 0x890) bit description

Bit	Symbol	Access	Reset value	Description
1:0	RCO_PWR_MODE	RW	0x0	RCO VDD selection 00b = RCO VDD is set as 1.1 V, 630 nA, 250 ppm 10b = RCO VDD is set as 0.95 V, 350 nA, 340 ppm 10b,11b = RCO VDD is set as 0.82 V, 200 nA, 500 ppm
17:2	RESERVED	-	-	-
18	DIS_USB_PULLUP[1]	RW	0x1	1 = USB pull up resistor is disconnected 0 = USB pull up resistor state depends on other three USB control registers; see Table 287
23:19	RESERVED	-	-	-
24	DPPU_OPT_SEL	RW	0x1	pull-up strength (1.2 kΩ or 2.3 kΩ) source selection 0 = pull up strength controlled by register 1 = pull up strength controlled by USB controller
25	DPPU_OPT_POL	RW	0x1	drive high to swap pull up strength value; see Table 288
31:26	RESERVED	-	-	-

[1] Before USB controller power is turned down, DIS_USB_PULLUP should be configured to 1. Before USB controller is used, DIS_USB_PULLUP should be configured to 0.

3. Memory map

3.1 General description

QN908x has several distinct memory regions. Figure 4 shows the overall map of the entire address space from a user program viewpoint following reset.

The APB peripheral area (see Figure 5) is divided into fixed 4k slots to simplify addressing.

The CPU registers such as NVIC and SysTick, and sleep mode control are located on the private peripheral bus.

For more information on memory size of all products in QN908x series, refer to QN908x data sheet.

3.1.1 ROM

A 256 kB ROM is implemented in QN908x that saves bootloader and BLE stack. The default address of the ROM is 0x00. The chip boots up from ROM every time, and jumps to flash, where user application code is located.

3.1.2 SRAM

The QN908x has on-chip 128 kB SRAM with contiguous memory. This 128 kB is divided up to 10 different SRAM blocks to allow more control over power usage when less SRAM is required. The PMU_CTRL0 register in SYSCON contains a separate power switch for each SRAM block.

Table 48. QN9080 SRAM blocks

Name	Size (kB)	Type	Address
RAM0	4	single port RAM	0x20000000~0x20000FFF
RAM1	4	single port RAM	0x20001000~0x20001FFF
RAM2	8	single port RAM	0x20002000~0x20003FFF
RAM3	16	single port RAM	0x20004000~0x20007FFF
RAM4	16	single port RAM	0x20008000~0x2000BFFF
RAM5	16	single port RAM	0x2000C000~0x2000FFFF
RAM6	16	single port RAM	0x20010000~0x20013FFF
RAM7	16	single port RAM	0x20014000~0x20017FFF
RAM8	16	single port RAM	0x20018000~0x2001BFFF
RAM9	16	single port RAM	0x2001C000~0x2001FFFF

3.1.3 Flash

The QN908x supports up to 512 kB of on-chip flash memory to store code and data. The MCU accesses the flash using the flash controller; see Section 28. Each flash page can be erased individually and has a size of 2 kB.

3.1.4 SPI Flash Interface (SPIFI)

The SPI flash interface allows low-cost serial flash memories to be connected to the CPU, with little performance penalty compared to parallel flash devices with higher pin count. A driver API handles setup, programming, and erasure. After initializing the SPIFI, the flash content is accessible using byte, halfword, and word accesses by the processor and/or DMA. See Section 25 for more information.

Remark: the SPIFI implemented on QN908x devices is for data access. The cache size is reduced and therefore direct execution from off-chip SPI flash is not recommended.

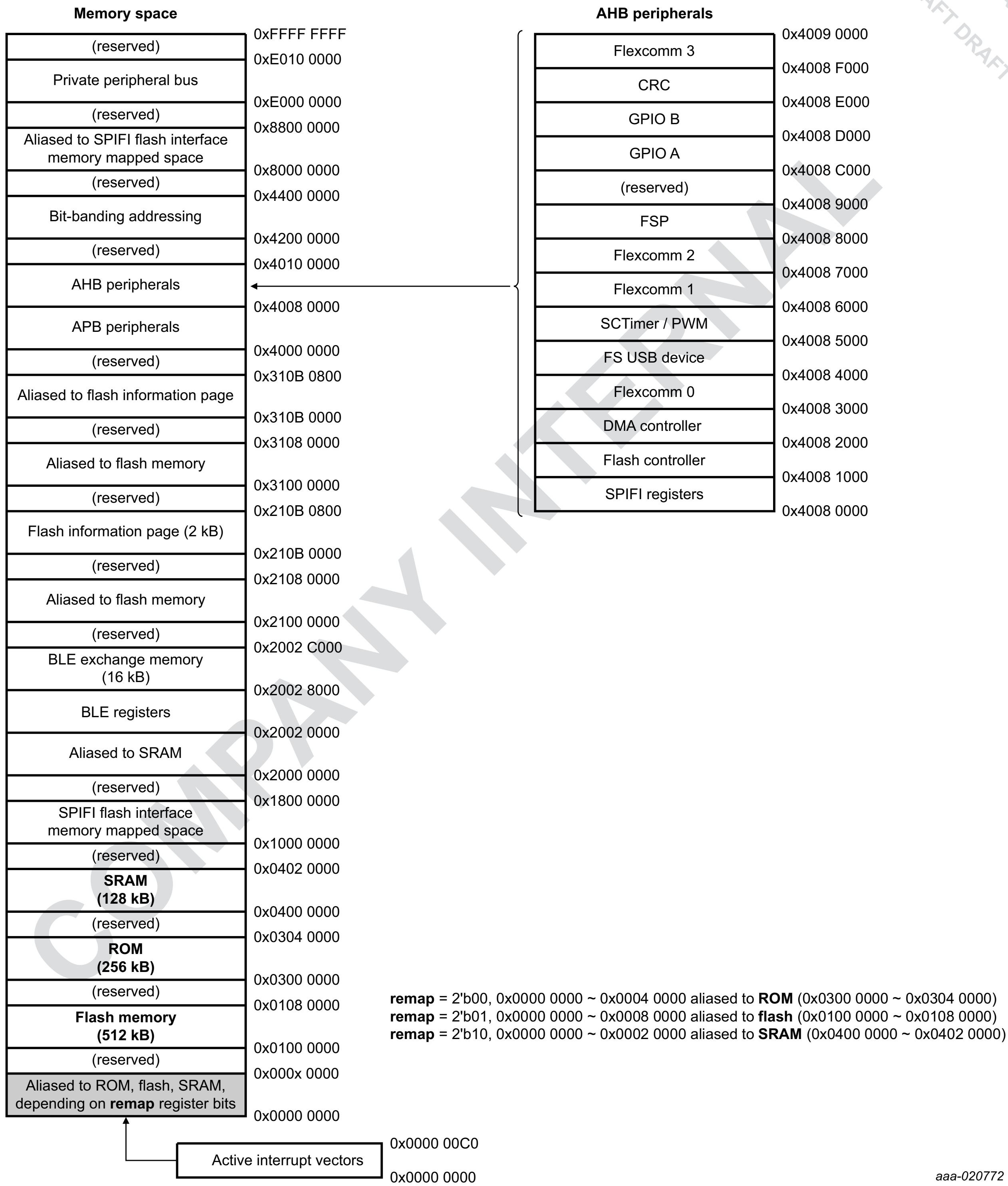
3.1.5 Bit-band addressing

The ARM Cortex-M4 CPU provides a bit-band addressing feature. It offers efficient bit accesses. Bits in the bit-band region (0x2000 0000 to 0x2010 0000 and 0x4000 0000 to 0x40100000) can be accessed in the range 0x2200 0000 and 0x4200 0000, called the alias region. Reads return the respective bit from the bit-band region. Writes perform an atomic read-modify-write on the respective bit of the bit-band region. For details, see the ARM Cortex-M4 technical reference manual.

Remark: Because bit-band operations are implemented as read-modify-write operations, and appear on the AHB bus in that manner, some uses of bit-banding may not work as expected. For example, if a peripheral register contains several write-one-to-clear status flags, clearing one such flag using bit-banding will clear all such flags that read as a one in that register.

3.1.6 Memory mapping

The overall memory map is shown in Figure 4. Details of APB peripheral mapping are shown in Figure 5.



Remark: BLE registers and BLE exchange memory area are only accessible for BLE API

Fig 4. Main memory map

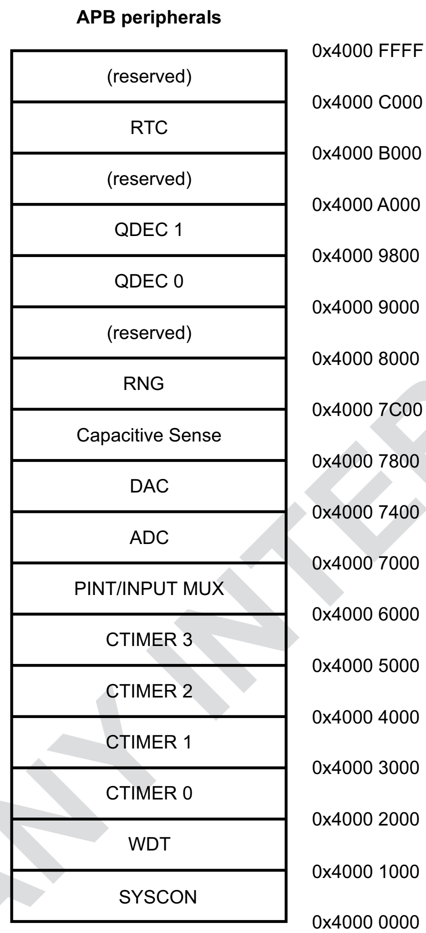


Fig 5. APB memory map

3.1.7 Memory remap

After a power cycle or reset, the MCU will boot from the internal ROM at address 0x0. To make the chip more flexible, address 0x0 can be remapped to either the internal flash or SRAM. The REMAP bit field in the SYS_MODE_CTRL register indicates which memory is mapped to address 0x0.

3.1.8 Memory Protection Unit (MPU)

The Cortex-M4 processor has a Memory Protection Unit (MPU) that provides fine grain memory control, enabling applications to implement security privilege levels, separating code, data and stack on a task-by-task basis. Such requirements are critical in many embedded applications.

The MPU register interface is located on the private peripheral bus and is described in detail in Ref. 1 “Cortex-M4 TRM”.

4. Nested Vectored Interrupt Controller (NVIC)

4.1 Introduction

Available interrupt sources may vary with specific QN908x device type.

4.2 Features

- Nested vectored interrupt controller that is an integral part of each CPU
- Tightly coupled interrupt controller provides low interrupt latency
- Controls system exceptions and peripheral interrupts
- The NVIC of the Cortex-M4 supports:
  - 55 vectored interrupt slots, some are reserved
  - Eight programmable interrupt priority levels with hardware priority level masking
  - Vector Table Offset Register (VTOR)
  - Software interrupt generation

4.3 General description

The tight coupling of NVIC to the CPU allows for low interrupt latency and efficient processing of late arriving interrupts.

4.3.1 Interrupt sources

Table 49 lists the interrupt sources for each peripheral function. Each peripheral device may have one or more interrupt lines to the vectored interrupt controller. Each line may represent more than one interrupt source. The interrupt number does not imply any interrupt priority.

See Ref. 1 "Cortex-M4 TRM" for descriptions of NVIC and the NVIC registers.

Table 49. Collection of interrupt sources to the NVIC

Interrupt	Name	Interrupt description	Flags
0	EXTIO_WAKEUP	external GPIO wake up	IO wakeup
1	BLE_WAKEUP	oscillator	BLE oscillator wakeup
2	ACMP0_INT	analog comparator 0 match	same as comparator
3	ACMP1_INT	analog comparator 1 match	same as comparator
4	RESERVED	-	-
5	RTC_SEC_INT	RTC second match	see RTC
6	RTC_FR_INT	RTC free running match	see RTC
7	CS_WAKEUP_INT	capacitive sense sleep wake up	see capacitive sense
8	CS_INT	capacitive sense detection	see capacitive sense
9	GPIOA_INT	GPIOA	enabled pin interrupts
10	GPIOB_INT	GPIOB	enabled pin interrupts
11	DMA_INT	DMA controller	see DMA
12	PIN_INT0	pin 0 or pattern match engine slice 0	PSTAT: pin interrupt status

Table 49. Collection of interrupt sources to the NVIC *...continued*

Interrupt	Name	Interrupt description	Flags
13	PIN_INT1	pin 1 or pattern match engine slice 1	PSTAT: pin interrupt status
14	PIN_INT2	pin 2 or pattern match engine slice 2	PSTAT: pin interrupt status
15	PIN_INT3	pin 3 or pattern match engine slice 3	PSTAT – pin interrupt status
16	BLE_SLEEP_STATUS	inverse of BLE_WAKEUP	see BLE_WAKEUP
17	USB_INT	USB	see USB
18	FLEXCOMM0_INT	flexcomm 0 (USART)	see flexcomm
19	FLEXCOMM1_INT	flexcomm 1 (USART, I2C)	see flexcomm
20	FLEXCOMM2_INT	flexcomm 2 (SPI, I2C)	see flexcomm
21	FLEXCOMM3_INT	flexcomm 3 (SPI)	see flexcomm
22	BLE_INT	BLE all	all BLE interrupt
23	FSP_INT	FSP	see FSP
24	QDEC0_INT	QDEC0	see QDEC
25	QDEC1_INT	QDEC1	see QDEC
26	CTIMER0_INT	standard counter/timer CTIMER0	match and capture interrupts
27	CTIMER1_INT	standard counter/timer CTIMER1	match and capture interrupts
28	CTIMER2_INT	standard counter/timer CTIMER2	match and capture interrupts
29	CTIMER3_INT	standard counter/timer CTIMER3	match and capture interrupts
30	WDT_INT	watchdog	see WDT
31	ADC_INT	ADC	ADC
32	DAC_INT	DAC	DAC
33	XTAL_RDY_INT	high frequency crystal ready	high to indicate external crystal ready
34	FLASH_INT	embedded flash controller	see Flash
35	SPIFI_INT	SPI flash interface controller	see SPIFI
36	SCT_INT	SCTimer/PWM	see PWM
37	RESERVED	-	-
38	RNG_INT	random number generator	see RNG
39	RESERVED	-	-
40	RESERVED	-	-
41	RESERVED	-	-
42	BLE_TX_IRQn	BLE TX interrupt	BLE TX
43	BLE_RX_IRQn	BLE RX interrupt	BLE RX
44	BLE_FREQ_HOP_IRQn	BLE frequency hop interrupt	BLE frequency hop
45	RESERVED	-	-
46	RESERVED	-	-
47	RESERVED	-	-
48	RESERVED	-	-

Table 49. Collection of interrupt sources to the NVIC *...continued*

Interrupt	Name	Interrupt description	Flags
49	RESERVED	-	-
50	RESERVED	-	-
51	BOD_INT	BOD output	BOD interrupt

4.4 Register description

The NVIC registers are located on the ARM private peripheral bus.

Table 50. Register overview: NVIC (base address 0xE000 E000)

Name	Access	Offset	Description	Reset value	Section
ISER0	R/W	0x100	Interrupt Set Enable Register 0. This register allows enabling interrupts and reading back the interrupt enables for peripheral functions.	0x0	4.4.1
ISER1	R/W	0x104	Interrupt Set Enable Register 1. See ISER0 description.	0x0	4.4.2
ICER0	R/W	0x180	Interrupt Clear Enable Register 0. This register allows disabling interrupts and reading back the interrupt enables for peripheral functions.	0x0	4.4.3
ICER1	R/W	0x184	Interrupt Clear Enable Register 1. See ISER0 description.	0x0	4.4.4
ISPR0	R/W	0x200	Interrupt Set Pending Register 0. This register allows changing the interrupt state to pending and reading back the interrupt pending state for peripheral functions.	0x0	4.4.5
ISPR1	R/W	0x204	Interrupt Set Pending Register 1. See ISPR0 description.	0x0	4.4.6
ICPR0	R/W	0x280	Interrupt Clear Pending Register 0. This register allows changing the interrupt state to not pending and reading back the interrupt pending state for peripheral functions.	0x0	4.4.7
ICPR1	R/W	0x284	Interrupt Clear Pending Register 1. See ICPR0 description.	0x0	4.4.8
IABR0	RO	0x300	Interrupt Active Bit Register 0. This register allows reading the current interrupt active state for specific peripheral functions.	0x0	4.4.9
IABR1	RO	0x304	Interrupt Active Bit Register 1. See IABR0 description.	0x0	4.4.10
IPR0	R/W	0x400	Interrupt Priority Register 0. This register contains the priority fields for interrupts 0 to 3. Three bits for the Cortex-M4.	0x0	4.4.11
IPR1	R/W	0x404	Interrupt Priority Register 1. This register contains the 3-bit priority fields for interrupts 4 to 7. Three bits for the Cortex-M4.	0x0	4.4.12
IPR2	R/W	0x408	Interrupt Priority Register 2. This register contains the 3-bit priority fields for interrupts 8 to 11. Three bits for the Cortex-M4.	0x0	4.4.13
IPR3	R/W	0x40C	Interrupt Priority Register 3. This register contains the 3-bit priority fields for interrupts 12 to 15. Three bits for the Cortex-M4.	0x0	4.4.14
IPR4	R/W	0x410	Interrupt Priority Register 4. This register contains the 3-bit priority fields for interrupts 16 to 19. Three bits for the Cortex-M4.	0x0	4.4.15
IPR5	R/W	0x414	Interrupt Priority Register 5. This register contains the 3-bit priority fields for interrupts 20 to 23. Three bits for the Cortex-M4.	0x0	4.4.16
IPR6	R/W	0x418	Interrupt Priority Register 6. This register contains the 3-bit priority fields for interrupts 24 to 27. Three bits for the Cortex-M4.	0x0	4.4.17
IPR7	R/W	0x41C	Interrupt Priority Register 7. This register contains the 3-bit priority fields for interrupts 28 to 31. Three bits for the Cortex-M4.	0x0	4.4.18
IPR8	R/W	0x420	Interrupt Priority Register 8. This register contains the 3-bit priority fields for interrupts 32 to 35. Three bits for the Cortex-M4.	0x0	4.4.19

Table 50. Register overview: NVIC (base address 0xE000 E000) *…continued*

Name	Access	Offset	Description	Reset value	Section
IPR9	R/W	0x424	Interrupt Priority Register 9. This register contains the 3-bit priority fields for interrupts 36 to 39. Three bits for the Cortex-M4.	0x0	4.4.20
IPR10	R/W	0x428	Interrupt Priority Register 10. This register contains the 3-bit priority fields for interrupts 40 to 43. Three bits for the Cortex-M4.	0x0	4.4.21
IPR11	R/W	0x42C	Interrupt Priority Register 11. This register contains the 3-bit priority fields for interrupts 44 to 47. Three bits for the Cortex-M4.	0x0	4.4.22
IPR12	R/W	0x430	Interrupt Priority Register 12. This register contains the 3-bit priority fields for interrupts 48 to 51. Three bits for the Cortex-M4.	0x0	4.4.23
STIR	WO	0xF00	Software Trigger Interrupt Register, allows software to generate interrupts.	-	4.4.24

4.4.1 Interrupt Set-Enable Register 0 (ISER0)

The ISER0 register allows enabling the first 32 peripheral interrupts, or for reading the enabled state of those interrupts. Bit numbers match Table 49. Unused bits are reserved. The remaining interrupts are enabled via the ISER1 register (Section 4.4.2). Disabling interrupts is done through the ICER0 and ICER1 registers (Section 4.4.3 and Section 4.4.4).

Table 51. ISER0- Interrupt Set-Enable Register 0 (offset = 0x100) bit description

Bit	Name	Function
0	ISE_EXTIO_WAKEUP	external GPIO interrupt enable
1	ISE_OSC	BLE oscillator wake up interrupt enable
2	ISE_ACMP0	analog comparator 0 interrupt enable
3	ISE_ACMP1	analog comparator 1 interrupt enable
4	RESERVED	read value is undefined, only zero should be written.
5	ISE_RTC_SEC	RTC second match interrupt enable
6	ISE_RTC_FR	RTC free running match interrupt enable
7	ISE_CS_WAKEUP	capacitive sense wake up interrupt enable
8	ISE_CS	capacitive Sense detection interrupt enable
9	ISE_GPIOA	GPIOA interrupt enable
10	ISE_GPIOB	GPIOB interrupt enable
11	ISE_DMA	DMA interrupt enable
12	ISE_PIN_INT0	pin interrupt/pattern match engine slice 0 interrupt enable
13	ISE_PIN_INT1	pin interrupt/pattern match engine slice 1 interrupt enable
14	ISE_PIN_INT2	pin interrupt/pattern match engine slice 2 interrupt enable
15	ISE_PIN_INT3	pin interrupt/pattern match engine slice 3 interrupt enable
16	ISE_OSC_INT_LOW	BLE sleep interrupt enable
17	ISE_USB	USB device enable
18	ISE_FC0	flexcomm interface 0 interrupt enable
19	ISE_FC1	flexcomm interface 1 interrupt enable
20	ISE_FC2	flexcomm interface 2 interrupt enable
21	ISE_FC3	flexcomm interface 3 interrupt enable
22	ISE_BLE	BLE interrupt enable
23	ISE_FSP	FSP interrupt enable
24	ISE_QDEC0	quadrature decoder QDEC0 interrupt enable
25	ISE_QDEC1	quadrature decoder QDEC1 interrupt enable
26	ISE_CTIMER0	standard counter/timer CTIMER0 interrupt enable
27	ISE_CTIMER1	standard counter/timer CTIMER1 interrupt enable
28	ISE_CTIMER2	standard counter/timer CTIMER2 interrupt enable
29	ISE_CTIMER3	standard counter/timer CTIMER3 interrupt enable
30	ISE_WDT	watchdog timer interrupt enable
31	ISE_ADC	ADC interrupt enable

For each bit in Table 51,

Write: writing 0 has no effect, writing 1 enables the interrupt.

Read: 0 indicates that the interrupt is disabled, 1 indicates that the interrupt is enabled.

4.4.2 Interrupt Set-Enable Register 1 (ISER1)

The ISER1 register allows enabling the second group of peripheral interrupts, or for reading the enabled state of those interrupts. Disabling interrupts is done through the ICER0 and ICER1 registers (Section 4.4.3 and Section 4.4.4).

Table 52. ISER1 - Interrupt Set-Enable Register 1 (offset = 0x104) bit description

Bit	Name	Function
0	ISE_DAC	DAC interrupt enable
1	ISE_XTAL_READY	high frequency crystal ready interrupt enable
2	ISE_FLASH	flash controller interrupt enable
3	ISE_SPIFI	SPI flash controller interrupt enable
4	ISE_SCT	SCTimer/PWM interrupt enable
5	-	reserved; read value is undefined, only zero should be written
6	ISE_RNG	random number generator interrupt enable
30:7	-	reserved; read value is undefined, only zero should be written
31	ISE_BOD	BOD interrupt enable write: writing 0 has no effect, writing 1 disables the interrupt read: 0 indicates that the interrupt is disabled, 1 indicates that the interrupt is enabled

For each bit in Table 52,

Write: writing 0 has no effect, writing 1 enables the interrupt.

Read: 0 indicates that the interrupt is disabled, 1 indicates that the interrupt is enabled.

4.4.3 Interrupt Clear-Enable Register 0 (ICER0)

The ICER0 register allows disabling the first 32 peripheral interrupts, or for reading the enabled state of those interrupts. The remaining interrupts are disabled via the ICER1 register (Section 4.4.4). Enabling interrupts is done through the ISER0 and ISER1 registers (Section 4.4.1 and Section 4.4.2).

Table 53. ICER0 - Interrupt Clear-Enable Register 0 (offset = 0x180) bit description

Bit	Name	Function
31:0	ICE_...	Peripheral interrupt disables. Bit numbers match ISER0 registers (Table 51). Unused bits are reserved. Write: writing 0 has no effect, writing 1 disables the interrupt. Read: 0 indicates that the interrupt is disabled, 1 indicates that the interrupt is enabled.

4.4.4 Interrupt Clear-Enable Register 1 (ICER1)

The ICER1 register allows disabling the second group of peripheral interrupts, or for reading the enabled state of those interrupts. Enabling interrupts is done through the ISER0 and ISER1 registers (Section 4.4.1 and Section 4.4.2).

Table 54. ICER1 - Interrupt Clear-Enable Register 1 (offset = 0x184) bit description

Bit	Name	Function
31:0	ICE_...	Peripheral interrupt disables. Bit numbers match ISER1 registers (Table 52). Unused bits are reserved. Write: writing 0 has no effect, writing 1 disables the interrupt. Read: 0 indicates that the interrupt is disabled, 1 indicates that the interrupt is enabled.

4.4.5 Interrupt Set-Pending Register 0 (ISPR0)

The ISPR0 register allows setting the pending state of the first 32 peripheral interrupts, or for reading the pending state of those interrupts. The remaining interrupts can have their pending state set via the ISPR1 register (Section 4.4.6). Clearing the pending state of interrupts is done through the ICPR0 and ICPR1 registers (Section 4.4.7 and Section 4.4.8).

Table 55. ISPR0 - Interrupt Set-Pending Register 0 (offset = 0x200) bit description

Bit	Name	Function
31:0	ISP_...	Peripheral interrupt pending set. Bit numbers match ISER0 registers (Table 51). Unused bits are reserved. Write: writing 0 has no effect, writing 1 changes the interrupt state to pending. Read: 0 indicates that the interrupt is not pending, 1 indicates that the interrupt is pending.

4.4.6 Interrupt Set-Pending Register 1 (ISPR1)

The ISPR1 register allows setting the pending state of the second group of peripheral interrupts, or for reading the pending state of those interrupts. Clearing the pending state of interrupts is done through the ICPR0 and ICPR1 registers (Section 4.4.7 and Section 4.4.8).

Table 56. ISPR1 - Interrupt Set-Pending Register 1 (offset = 0x204) bit description

Bit	Name	Function
31:0	ISP_...	Peripheral interrupt pending set. Bit numbers match ISER1 registers (Table 52). Unused bits are reserved. Write: writing 0 has no effect, writing 1 changes the interrupt state to pending. Read: 0 indicates that the interrupt is not pending, 1 indicates that the interrupt is pending.

4.4.7 Interrupt Clear-Pending Register 0 (ICPR0)

The ICPR0 register allows clearing the pending state of the first 32 peripheral interrupts, or for reading the pending state of those interrupts. The remaining interrupts can have their pending state cleared via the ICPR1 register (Section 4.4.8). Setting the pending state of interrupts is done through the ISPR0 and ISPR1 registers (Section 4.4.5 and Section 4.4.6).

Table 57. ICPR0 - Interrupt Clear-Pending Register 0 (offset = 0x280) bit description

Bit	Name	Function
31:0	ICP_...	Peripheral interrupt pending clear. Bit numbers match ISER0 registers (Table 51). Unused bits are reserved. Write: writing 0 has no effect, writing 1 changes the interrupt state to not pending. Read: 0 indicates that the interrupt is not pending, 1 indicates that the interrupt is pending.

4.4.8 Interrupt Clear-Pending Register 1 (ICPR1)

The ICPR1 register allows clearing the pending state of the second group of peripheral interrupts, or for reading the pending state of those interrupts. Setting the pending state of interrupts is done through the ISPR0 and ISPR1 registers (Section 4.4.5 and Section 4.4.6).

Table 58. ICPR1 - Interrupt Clear-Pending Register 1 (offset = 0x284) bit description

Bit	Name	Function
31:0	ICP_...	Peripheral interrupt pending clear. Bit numbers match ISER1 registers (Table 52). Unused bits are reserved. Write: writing 0 has no effect, writing 1 changes the interrupt state to not pending. Read: 0 indicates that the interrupt is not pending, 1 indicates that the interrupt is pending.

4.4.9 Interrupt Active Bit Register 0 (IABR0)

The IABR0 register is a read-only register that allows reading the active state of the first 32 peripheral interrupts. Bits in IABR are set while the corresponding interrupt service routines are in progress. Additional interrupts can have their active state read via the IABR1 register (Section 4.4.10).

Table 59. IABR0 - Interrupt Active Bit Register 0 (offset = 0x300) bit description

Bit	Name	Function
31:0	IAB_...	Peripheral interrupt active. Bit numbers match ISER0 registers (Table 51). Unused bits are reserved. Read: 0 indicates that the interrupt is not active, 1 indicates that the interrupt is active.

4.4.10 Interrupt Active Bit Register 1 (IABR1)

The IABR1 register is a read-only register that allows reading the active state of the second group of peripheral interrupts. Bits in IABR are set while the corresponding interrupt service routines are in progress.

Table 60. IABR1 - Interrupt Active Bit Register 1 (offset = 0x304) bit description

Bit	Name	Function
31:0	IAB_...	Peripheral interrupt active. Bit numbers match ISER1 registers (Table 52). Unused bits are reserved. Read: 0 indicates that the interrupt is not active, 1 indicates that the interrupt is active.

4.4.11 Interrupt Priority Register 0 (IPR0)

The IPR0 register controls the priority of the first 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 61. IPR0 - Interrupt Priority Register 0 (offset = 0x400) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_EXTIO_WAKEUP	External IO wake up interrupt priority; 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_OSC	BLE oscillator interrupt priority; 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_ACMP0	ACMP0 interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_ACMP1	ACMP1 interrupt priority; 0: highest priority, 7: lowest priority

4.4.12 Interrupt Priority Register 1 (IPR1)

The IPR1 register controls the priority of the second group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 62. IPR1 - Interrupt Priority Register 1 (offset = 0x404) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	RESERVED	-
12:8	RESERVED	-
15:13	IP_RTC_SEC	RTC second interrupt priority; 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_RTC_FR	RTC free running interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_CS_WAKEUP	Cap sense sleep wake up interrupt priority; 0: highest priority, 7: lowest priority

4.4.13 Interrupt Priority Register 2 (IPR2)

The IPR2 register controls the priority of the third group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 63. IPR2 - Interrupt Priority Register 2 (offset = 0x408) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_CS	Cap Sense interrupt priority; 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_GPIOA	GPIOA interrupt priority; 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_GPIOB	GPIOB interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_DMA	DMA interrupt priority; 0: highest priority, 7: lowest priority

4.4.14 Interrupt Priority Register 3 (IPR3)

The IPR3 register controls the priority of the fourth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 64. IPR3 - Interrupt Priority Register 3 (offset = 0x40C) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_PIN_INT0	Pin interrupt / pattern match engine slice 0 priority 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_PIN_INT1	Pin interrupt / pattern match engine slice 1 priority 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_PIN_INT2	Pin interrupt / pattern match engine slice 2 priority 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_PIN_INT3	Pin interrupt / pattern match engine slice 3 priority 0: highest priority, 7: lowest priority

4.4.15 Interrupt Priority Register 4 (IPR4)

The IPR4 register controls the priority of the fifth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 65. IPR4 - Interrupt Priority Register 4 (offset = 0x410) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_OSC_INT_L OW	interrupt priority of inverse of BLE_WAKEUP; 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_USB	USB interrupt priority; 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_FC0	Flexcomm Interface 0 interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_FC1	Flexcomm Interface 1 interrupt priority; 0: highest priority, 7: lowest priority

4.4.16 Interrupt Priority Register 5 (IPR5)

The IPR5 register controls the priority of the sixth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 66. IPR5 - Interrupt Priority Register 5 (offset = 0x414) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_FC2	Flexcomm Interface 2 interrupt priority; 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_FC3	Flexcomm Interface 3 interrupt priority; 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_BLE	all BLE interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_FSP	FSP interrupt priority; 0: highest priority, 7: lowest priority

4.4.17 Interrupt Priority Register 6 (IPR6)

The IPR6 register controls the priority of the seventh group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 67. IPR6 - Interrupt Priority Register 6 (offset=0x418) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_QDEC0	QDEC0 interrupt priority; 0: highest priority, 7: lowest priority
12:8	-	Reserved
15:13	IP_QDEC1	QDEC1 interrupt priority; 0: highest priority, 7: lowest priority
20:16	-	Reserved
23:21	IP_CTIMER0	Standard counter/timer CTIMER0 interrupt priority; 0: highest priority, 7: lowest priority
28:24	-	Reserved
31:29	IP_CTIMER1	Standard counter/timer CTIMER1 interrupt priority; 0: highest priority, 7: lowest priority

4.4.18 Interrupt Priority Register 7 (IPR7)

The IPR7 register controls the priority of the eighth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4. 0 is the highest priority.

Table 68. IPR7 - Interrupt Priority Register 7 (offset = 0x41C) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_CTIMER2	Standard counter/timer CTIMER2 interrupt Priority; 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_CTIMER3	Standard counter/timer CTIMER3 interrupt priority; 0: highest priority, 7: lowest priority
20:16	RESERVED	-
23:21	IP_WDT	WDT interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_ADC	ADC interrupt priority; 0: highest priority, 7: lowest priority

4.4.19 Interrupt Priority Register 8 (IPR8)

The IPR8 register controls the priority of the ninth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4, where 0 is the highest priority.

Table 69. IPR8 - Interrupt Priority Register 8 (offset = 0x420) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_DAC	DAC interrupt priority; 0: highest priority, 7: lowest priority
12:8	RESERVED	-
15:13	IP_XTAL_READY	High frequency crystal ready interrupt priority 0: highest priority, 7: lowest priority.
20:16	RESERVED	-
23:21	IP_FLASH	Embedded flash interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_SPIFI	SPI flash interface interrupt priority; 0: highest priority, 7: lowest priority

4.4.20 Interrupt Priority Register 9 (IPR9)

The IPR9 register controls the priority of the tenth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4, where 0 is the highest priority.

Table 70. IPR9 - Interrupt Priority Register 9 (offset = 0x424) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_SCT	SCT interrupt priority; 0: highest priority, 7: lowest priority
20:8	RESERVED	-
23:21	IP_RNG	RNG interrupt priority; 0: highest priority, 7: lowest priority
31:24	RESERVED	-

4.4.21 Interrupt Priority Register 10 (IPR10)

The IPR10 register is reserved.

Table 71. IPR10 - Interrupt Priority Register 10 (offset = 0x428) bit description

Bit	Name	Function
20:0	RESERVED	-
23:21	IP_BLE_TX	BLE TX interrupt priority; 0: highest priority, 7: lowest priority
28:24	RESERVED	-
31:29	IP_BLE_RX	BLE RX interrupt priority; 0: highest priority, 7: lowest priority

4.4.22 Interrupt Priority Register 11 (IPR11)

The IPR11 register is reserved.

Table 72. IPR11 - Interrupt Priority Register 11 (offset = 0x42C) bit description

Bit	Name	Function
4:0	RESERVED	-
7:5	IP_BLE_FREQ_HOP	BLE frequency interrupt priority; 0: highest priority, 7: lowest priority
31:8	RESERVED	-

4.4.23 Interrupt Priority Register 12 (IPR12)

The IPR12 register controls the priority of the ninth group of 4 peripheral interrupts. Each interrupt can have one of the eight priorities for the Cortex-M4, where 0 is the highest priority.

Table 73. IPR12 - Interrupt Priority Register 12 (offset = 0x42C) bit description

Bit	Name	Function
28:0	RESERVED	-
31:29	IP_BOD	BOD interrupt priority; 0: highest priority, 7: lowest priority

4.4.24 Software Trigger Interrupt Register (STIR)

The STIR register provides an alternate way for software to generate an interrupt, in addition to using the ISPR registers. This mechanism can only be used to generate peripheral interrupts, not system exceptions.

By default, only privileged software can write to the STIR register. Unprivileged software can be given this ability if privileged software sets the USERSETMPEND bit in the CCR register.

The interrupt number to be programmed in this register is listed in Table 49.

Table 74. STIR - Software Trigger Interrupt Register (offset = 0xF00) bit description

Bit	Symbol	Description
8:0	INTID	writing a value to this field generates an interrupt for the specified the interrupt number
31:9	RESERVED	read value is undefined, only zero should be written

5. Power Management Unit (PMU)

5.1 Introduction

This chapter provides an overview of power related information about QN908x devices. These devices include a variety of adjustable regulators, power switches, and clock switches to allow fine tuning power usage to match requirements at different performance levels and reduced power modes.

To turn analog components on or off in active and sleep modes, use PMU_CTRL0 and PMU_CTRL1 register.

5.2 General description

QN908x integrates highly efficient regulators to generate all internal supply voltages from a single external supply voltage. Optional integrated DC-to-DC buck convertor can be utilized to convert the external V_{CC} to lower internal voltage to further save the power.

5.2.1 Power supply

QN908x is supplied with battery from VCC. A DC-to-DC converter converting the external V_{CC} to lower voltage is integrated to save power. The following figure is for QN908x power supply.

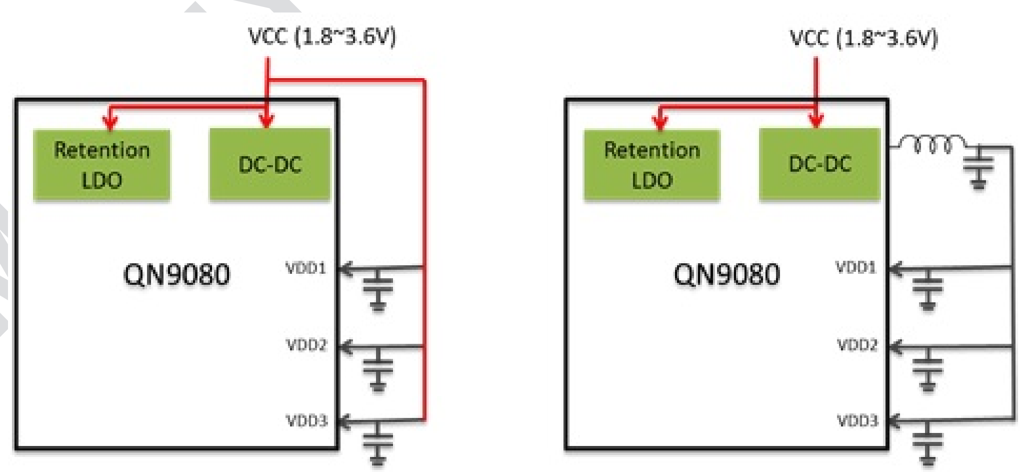


Fig 6. LDO and DC-to-DC converter

Please check BUCK_CTRL in PMU_CTRL1 register(Section 2.5.39), set it to 0x0 to enable DC-to-DC converter, 0xF to disable DC-to-DC converter

5.2.2 Power supervisor (BOD/BOR)

QN908x includes up to four levels for monitoring the voltage on the pin VCC. If this voltage falls below one of the selected levels, the BOD asserts an interrupt signal to the NVIC. The interrupt signal can be enabled for interrupt in the interrupt enable register in the NVIC (see Table 49), in order to cause a CPU interrupt.
If the BOD interrupt is enabled in the STARTER0 register and in the NVIC, the BOD interrupt can wake up the chip from sleep mode.

QN908x also supports BOR with up to four levels. If the voltage of V_{CC} falls below one of the selected levels, BOR generates a reset of QN908x. The logic of BOD and BOR are almost same as shown in Figure 7.

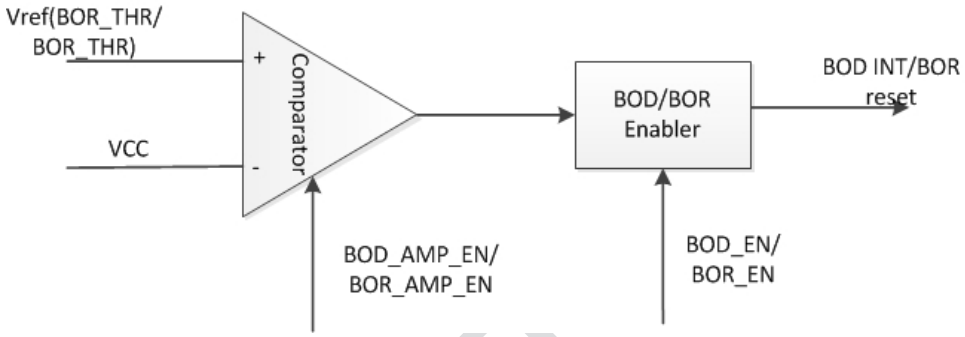


Fig 7. BOR/BOR block diagram

Both BOR and BOD are disabled by default. As in analog, there is a comparator to monitor the supplied voltage, to enable BOD or BOR by configuring both of BOD_EN/BOR_EN and BOD_AMP_EN/BOR_AMP_EN in ANA_EN register (see Section 2.5.40).

BOD_AMP_EN/BOR_AMP_EN should be set a earlier (100 μs) than BOD_EN/BOR_EN.

BOD threshold (BOD_THR) can be set from four levels (2.06 V,2.45 V, 2.72 V and 3.04 V).

BOR threshold (BOR_THR) can be set from four levels (1.5 V,1.85 V, 2 V and 3 V).

5.2.3 Power mode

QN908x supports a variety of power control features. In active mode, when the chip is running, power and clocks to selected peripherals can be optimized for power consumption. In addition, there are two special modes of processor power reduction with different peripherals running: sleep mode, power-down mode. Upon power-up or reset, the device enters into active mode. After processing is done, the software can put the chip into sleep mode or power-down mode, to save power consumption, from which the device can be woken up by related wake-up resources.

1. Active mode

   In active mode, the CPU, memories, and peripherals are clocked by the AHB/APB clock. The chip is in active mode after a Power-On Reset (POR) and when it is fully powered and operational after booting.

2. Sleep mode

   In sleep mode, the system clock to the CPU is stopped and execution of instructions is suspended until either a reset or an interrupt occurs. Peripheral functions, if selected to be clocked, can continue operation during sleep mode and may generate interrupts to cause the processor to resume execution. Sleep mode eliminates dynamic power used by the processor itself, memory systems and related controllers, internal buses, and unused peripherals. The processor state and registers, peripheral registers, and internal SRAM values are maintained, and the logic levels of the pins remain static.

3. Power-down mode

In power-down mode, power is shut off to the entire chip except for PMU domain, 32.768 kHz crystal oscillator, 32 kHz RC oscillator, sleep timer, RTC, and the RSTN pin. Configuration for switching off the power of 32 kHz RCO and 32.768 kHz crystal oscillator is illustrated in Table 75 and Table 76.

All peripheral clocks and all clock sources are switched off with the option of keeping the 32 kHz clock running. In addition, all analog blocks and the flash are shut down. In power-down mode, the application can keep the analog comparator circuit running to monitor external voltage input, which can wake up the device if the external voltage is higher than the reference voltage. In power down mode, the processor state and registers, and SRAM values can be maintained, and the logic levels of the pins remain static. This mode reduces power consumption more compared to sleep mode, at the expense of longer wake-up time.

Table 75. Switched off power of RCO32K

Power mode	RCO32K_PDM_DIS[1]	RCO32K_DIS[1]
active	x	1
sleep	x	1
power down	1	x
	x	1

[1] 'x' stands for "not applicable"; the value of this bit is not effective.

Table 76. Switched off power of 32.768 kHz crystal

Power mode	XTAL32K_PDM_DIS[1]	XTAL32K_DIS[1]
active	x	1
sleep	x	1
power down	1	x
	x	1

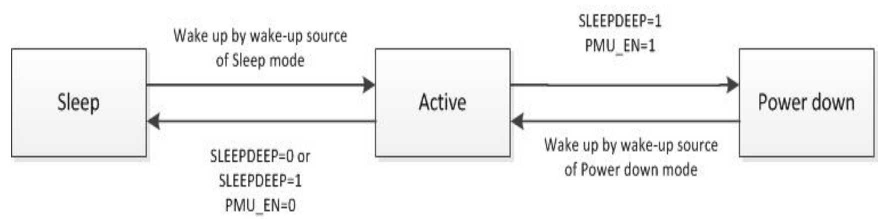
[1] 'x' stands for "not applicable"; the value of this bit is not effective.

Table 77. Peripherals configuration in reduced power modes

<table>
<tr><th>Peripheral</th><th>Active mode</th><th>Sleep mode</th><th>Power-down mode</th></tr>
<tr><td>ACMP0/1</td><td colspan="3">SW configured</td></tr>
<tr><td>32 kHz crystal</td><td colspan="3">SW configured</td></tr>
<tr><td>32 kHz RCO</td><td colspan="3">SW configured</td></tr>
<tr><td>CapSense</td><td colspan="3">SW configured</td></tr>
<tr><td>other analog peripherals</td><td colspan="2">SW configured</td><td>OFF</td></tr>
<tr><td>SRAM blocks</td><td colspan="3">SW configured</td></tr>
<tr><td>RTC</td><td colspan="3">ON</td></tr>
<tr><td>sleep timer</td><td colspan="3">ON</td></tr>
<tr><td>FSP,USB,CRC,SPIFI</td><td colspan="2">SW configured</td><td>OFF</td></tr>
<tr><td>BLE</td><td colspan="2">SW configured</td><td>OFF</td></tr>
<tr><td>other digital peripherals</td><td colspan="2">ON</td><td>OFF</td></tr>
</table>

5.2.4 Power mode state machine

Transition between power modes are illustrated as Figure. For detailed operation information for power modes transition, refer to QN908x SW developer guide.



(1) SLEEPDEEP is an ARM internal register accessible by user

(2) PMU_EN is a bit in PMU_CTRL0 register described in SYSCON

Fig 8. Power mode state machine

5.2.5 Wake-up process

The part always wakes up to the active mode. To wake up from the reduced power modes, the wake-up source is configured. Each reduced power mode supports its own wake-up sources and are configured as shown in Table 78.

Table 78. Wake-up sources for reduced power modes

Power mode	Wake-up source	Conditions
sleep	any interrupt	enable interrupt in NVIC
power down	sleep timer	enable OSC_INT interrupt in NVIC enable 32 kHz crystal or RCO sleep timer register enable WAKEUP_EN bit in PMU_CTRL0 register
	external IO interrupt(PA00~PA31)	enable EXTIO_WAKEUP interrupt in NVIC Configure PIO_WAKEUP_EN0 and PIO_WAKEUP_LVL0 register Enable WAKEUP_EN bit in PMU_CTRL0 register
	ACMP0/1	configure PIO_FUNC_CFGx to enable comparator input configure ANALOG_ENABLE register to enable comparator enable WAKEUP_EN bit in PMU_CTRL0 register
	CapSense interrupt	enable CS_WAKEUP_INT interrupt in NVIC enable 32 kHz crystal or RCO configure CapSense register enable WAKEUP_EN bit in PMU_CTRL0 register
	RTC second interrupt	enable RTC_SEC_INT interrupt in NVIC enable 32 kHz crystal or RCO enable RTC_SEC_WAKEUP_EN bit in PMU_CTRL0 register enable WAKEUP_EN bit in PMU_CTRL0 register
	RTC free running interrupt	enable RTC_FR_INT interrupt in NVIC enable 32 kHz crystal or RCO configure free running register in RTC enable WAKEUP_EN bit in PMU_CTRL0 register

6. Clock management unit

6.1 Introduction

QN908x supports external high-speed and low-speed crystal clock and integrated high and low RCO clock. This chapter is to illustrate the clock selection and controlling for system and blocks.

6.2 Basic configuration

6.2.1 System clock configuration

The system clock can be sourced from the internal 32 MHz oscillator, directly from the 16/32 MHz external crystal oscillator, or from 32 kHz (32.768 kHz) oscillator, selected by configuring register SYS_CLK_SEL; see Section 2.5.5.

6.2.2 Clock gating

Each clock gating for peripheral can be disabled or enabled independently by setting CLK_EN; see Section 2.5.4 or CLK_DIS; see Section 2.5.3. The user can disable peripheral by disabling the corresponding clock.

6.3 Pin description

Table 79. Clock pin description

Function	Type	Pin	Description	Reference
XTAL_OUT	O	XTAL_OUT	external clock output	Section 8
XTAL_IN	I	XTAL_IN	external clock input	Section 8
XTAL32_OUT	O	PB00	external 32K clock output	Section 8
XTAL32_IN	I	PB01	external 32K clock input	Section 8
32 kHz clock output	O	-	connected to PA04, PA10, PA18, or PA24 by configuring PIO_WAKEUP_EN1	Section 8
high frequency clock output	O	-	connected to PA05, PA11, PA19, PA25 by configuring PIO_WAKEUP_EN1	Section 8
CLK_AHB	O	PA28	AHB clock output	Section 8
RTC_CAP	I	PA28, PA31	RTC capture input	Section 8

6.4 General description

The Clock Management Unit (CMU) is responsible for controlling oscillators and clocks. The CMU provides capability to selectively turn on and off the clocks to peripherals, in addition to enabling/disabling and configuring of all available oscillators. The high degree of flexibility enables software to minimize the energy consumption in any specific application by not wasting power on the peripherals and the oscillators that are inactive.

6.4.1 Clock source

QN908x supports the following clock sources.

- 16/32 MHz high frequency crystal oscillator
- 32.768 kHz low frequency crystal oscillator
- 32 MHz high frequency internal free running oscillator
- 32 kHz low frequency internal RCO

6.4.1.1 16/32 MHz crystal oscillator (XTAL)

The high frequency crystal oscillator of 16/32 MHz external crystal with ±50 ppm accuracy provides the reference frequency for the radio transceiver.The load capacitance is integrated to reduce BOM cost, configurable by changing XTAL_LOAD_CAP and XTAL_EXTRA_CAP registers as per the following expression:

XTAL load cap = 5 pF + 0.35pF × XTAL_LOAD_CAP + 5 pF × XTAL_EXTRA_CAP

6.4.1.2 32.768 kHz crystal oscillator (XTAL32K)

Crystal oscillator with frequency 32.768 kHz is used to replace the RCO32K where accurate timing is needed. The 32 kHz clock with higher accuracy can reduce power consumption of Bluetooth Low Energy operation.

The load capacitance of XTAL32K is arrived as per the following expression:

XTAL32K load cap = 3.6 pF + 0.4pF × XTAL32K_LOAD_CAP + 6.4 pF × XTAL32K_EXTRA_CAP

6.4.1.3 32 MHz free running oscillator (OSC32M)

The internal 32 MHz oscillator can be used as a clock that drives the CPU. Upon power-up, or any chip reset, the QN908x uses the internal 32 MHz oscillator as the clock source. Software switches to one of the available clock sources later.

6.4.1.4 32 kHz free running oscillator (RCO32K)

The 32 kHz low-frequency RC oscillator is a low-power internal oscillator. The RCO32K can be used to provide a clock to the sleep timer, RTC and to the entire chip. The ±500 ppm accuracy can be achieved with calibration.

6.4.2 Clock generation

The following figure illustrates an overview of potential clock options in QN908x.

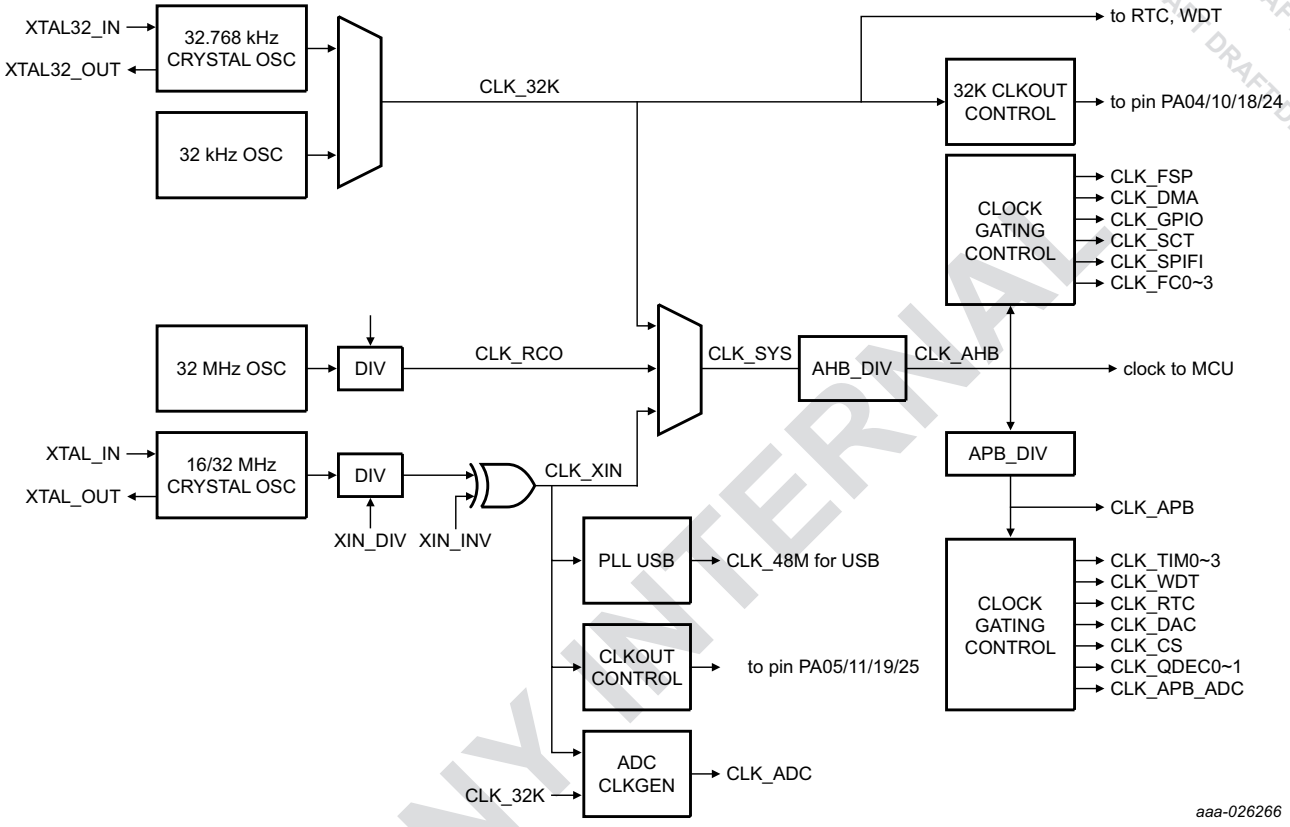


Fig 9. Clock generation

6.5 Register description

All clock related registers are system registers with base address 0x4000 0000. The bit description of the registers are specified in Section 2.

All address offsets not shown in the tables are reserved and should not be written to.

Table 80. Register overview: Main system configuration (base address 0x4000 0000)

Name	Access	Offset	Description	Reset value [1]	Section
CLK_DIS	R/W	0x08	clock disable register	0xC0001200	Section 2.5.3
CLK_EN	R/W	0x0C	clock enable register	0xC0001200	Section 2.5.4
CLK_CTRL	R/W	0x10	system clock source and divider register	0x103C0000	Section 2.5.5
SYS_TICK_CTRL	R/W	0x100	systick timer control register	0x81000147	Section 2.5.8
XTAL1_PA	R/W	0x110	crystal and PA register	0x820007F	Section 2.5.11
XTAL_CTRL	R/W	0x180	crystal control register	0x2020080A	Section 2.5.12
XTAL32K_CTRL	R/W	0x868	crystal 32 kHz control register	0x3023	Section 2.5.11

[1] Reset value reflects the data stored in defined bits only, reserved bits assumed to be 0.

6.6 Functional description

6.6.1 Start-up behavior

Upon power-up, or any chip reset, the QN908x uses the internal 32 MHz HFRCO as the clock source. Software switches to one of the available clock sources later.

6.6.2 Configure BLE clock

BLE clock is derived from the AHB clock. They are used for BLE RF block and can be configured by the user with register CLK_BLE_SEL. The BLE clock can only run at 8 MHz or 16 MHz.

6.6.3 Configure AHB and APB clock

The AHB clock is derived from the system clock and serves as a clock source for CPU, FSP, SCT, Quad-SPI, Flexcomm, GPIO, BLE_AHB and DMA.

As AHB clock should not be less than BLE clock.

if BLE clock is 8 MHz, AHB clock can only be 8 MHz or 16 MHz or 32 MHz.

if BLE clock is 16 MHz, AHB clock can only be 16 MHz or 32 MHz.

While BLE clock is disabled, if it is required to make the system work in lower speed, the AHB clock can be derived by configuring AHB divider (AHB_DIV in CLK_CTRL register; see Section 2.5.5) and calculated as expression AHB clock: System clock / (AHB_DIV + 1).

The APB clock is derived from the AHB clock and serves as the clock source for lots of peripherals. The required APB clock can be selected by configuring APB divider (APB_DIV in CLK_CTRL register; see Section 2.5.5) and calculated using the following expression:

APB clock: AHB clock / (APB_DIV + 1).

6.6.4 Clock output

QN908x provides 32k clock and high-frequency crystal clock output at eight different pins, out of which four pins are for 32 kHz and four are for high-frequency clock.

6.6.4.1 32K clock output

The GPIO pins PA04, PA10, PA18 & PA24 can output 32K clock.
The bit field values of PA04_32K_OE, PA10_32K_OE, PA18_32K_OE and PA24_32K_OE in PIO_WAKEUP_EN1 register decides the output from pins PA04, PA10, PA18 and PA24 irrespective of PIN_MUX_CTRL[x] register configuration.
The 32K output logic can be shown as follows:

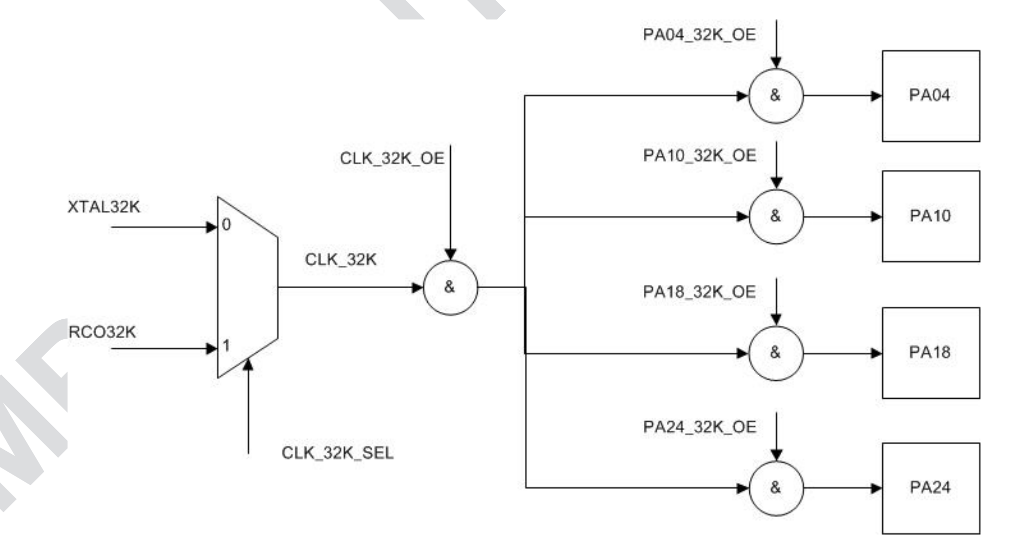


Fig 10. 32K clock output

Configuration for PA10, PA18 and PA24 to output 32K clock is same.

The flow of output 32K clock is:

- Power on 32 K XTAL (by write XTAL32K_DIS bit in PMU_CTRL1 register into 0) or 32K RCO (by write RCO32K_DIS bit in PMU_CTRL1 register into 0)
- Select the 32 K clock source by configure CLK_32K_SEL bit in CLK_CTRL register, where 1 for 32K RCO and 0 for XTAL32K
- Write CLK_32K_OE into 1 in CLK_CTRL
- Write PA<xx>_32K_OE bit in PIO_WAKEUP_EN1 register into 1

6.6.4.2 High-frequency clock output

The GPIO pins PA05, PA11, PA19 and PA25 can be configured to output high-frequency clock derived from XTAL. The output enable register is also located in PIO_WAKEUP_EN1 register named PA<xx>_XTAL_OE. Different from 32 kHz clock output, the frequency of these pins is configurable by XTAL_OUT_DIV bit in CLK_CTRL register. The formula of output clock frequency is:
F(clk_output): F(CLK_XIN) / (2 × XTAL_OUT_DIV) while XTAL_DIV is not equal to 0.

F(clk_output): F(CLK_XIN) while XTAL_OUT_DIV is equal to 0.

The logic can be shown as below.

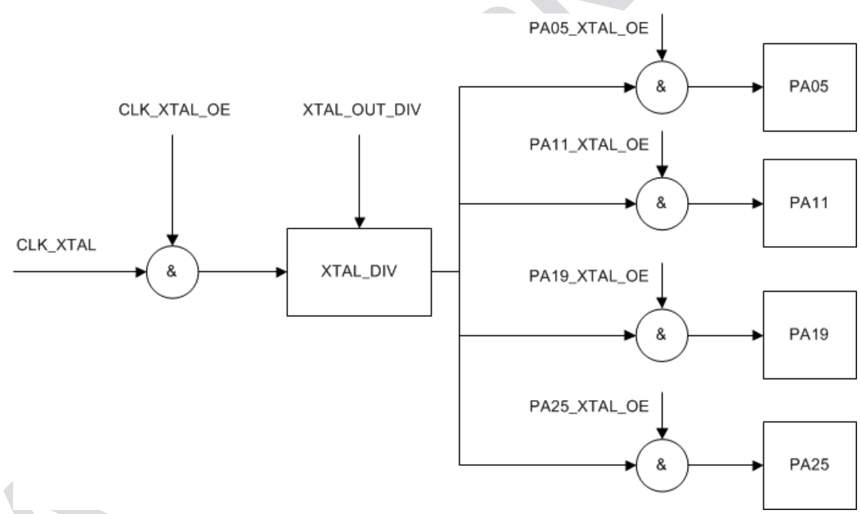


Fig 11. High-frequency clock output

The flow of output high frequency clock is:

- Power on XTAL
- Configure output clock frequency
- Write CLK_XTAL_OE into 1 in CLK_CTRL
- Write PA<xx>_XTAL_OE into 1

7. Boot process

7.1 Introduction

The QN908x on-chip ROM contains a boot loader which provides In-System Programming (ISP). The ISP supports to program internal flash via UART/SPI/USB. The content in the flash contains application program, user information (NVDS), user data, system configuration (lock bits & protection bit) and boot information. A connection must be made with the MCU immediately after entering ISP mode. Depending on the crystal frequency, ISP mode will time out after up to 200 ms.

7.2 Pin description

The QN908x supports ISP via USART, SPI and USB. The ISP mode is determined by the state of the CHIP_MODE pin at power-up or external reset. When the CHIP_MODE pin is low, ISP commands can be received from the USART(Flexcomm 0), SPI(Flexcomm 3) or USB.

The USART ISP interface is implemented on the following pins:

- PA16 for transfer
- PA17 for receive

The SPI ISP interface is implemented on the following pins:

- PA14 for SPI SSEL (select) signal
- PA15 for SPI CLK (clock)
- PA16 for SPI MOSI (master out, slave in) data signal
- PA17 for SPI MISO (master in, slave out) data signal

The USB interface is implemented on the following pins:

- PA26 for positive differential data
- PA27 for negative differential data

7.3 Boot loader

There are two modes in the boot loader.

- Fast boot mode: The purpose of Fast Boot Mode is to execute application image in the flash directly.
- ISP mode: ISP mode is to start the procedure for programming an application image in flash correctly over either USART/SPI or USB.

Remark: If no valid image is detected in the flash, the boot loader will enter into infinite loop.

7.3.1 Flash layout

There are two available flash space, main pages (512/256k bytes) and information page (2k bytes) could be used by software. This section describes the arrangement of these two flash spaces.

7.3.1.1 Flash Information Page

The information page keeps factory information which cannot be modified by users. Its base address is 0x210B 0000, see Figure 4

Table 81: Flash information page description

Offset	Length (Bytes)	Description
0x0	256	Reserved for factory test
0x100	1024	Reserved
0x500	256	Reserved for factory test
0x600	256	Reserved
0x700	4	CRC-32 (CRC value from offset 0x704 to 0x7FF) If CRC check failed, the default value will be used in boot loader instead of the settings in flash information page.
0x704	4	0x87654321: Boot loader version marker
0x708	4	0xFFFFF0FF: 512K Flash 0xFFFFE0FF: 256K Flash
0x70C	228	Reserved
0x7F0	4	Temperature sensor calibration
0x7F4	4	Bandgap voltage for ADC reference calibration
0x7F8	2	Reserved
0x7FA	6	Default Bluetooth address If there is Bluetooth address in the NVDS, the protocol stack will use address stored in the NVDS. Otherwise the protocol stack will use this address

7.3.1.2 Main Pages

The main pages are used for storing application code and user data.The 512KB of flash is partitioned as shown below.

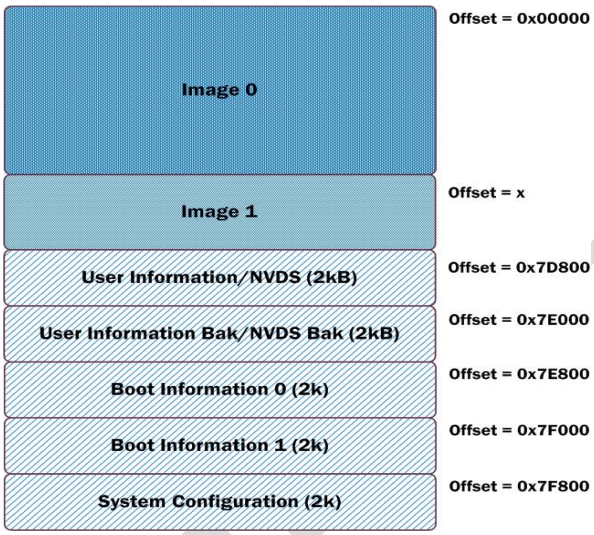


Fig 12. Flash layout

The boot loader supports 2 types of images: legacy image and single enhanced image. The selection of the image type is based on several fields defined in the application itself.

- Legacy image: the legacy image is located from sector 0 and does not contain an enhanced image marker or an image header structure. The only requirement for a legacy image is that the checksum of the first 7 32-bit words of the image must add to 0x00000000.
- Single enhanced image: The single enhanced image is located from sector 0. For an image to be single enhanced, it needs to have the single enhanced image marker value (0xEDDC9494) at offset 0x24 in vector table (see Table 83). It must also have a valid image header in the image pointed to at offset 0x28. The image header is explained in Image Boot Block Header, see Table 85.

Table 82: Legacy and Single enhanced image

Image type	Advantage	Disadvantage	Best use
Legacy	Simple to create, work with all tool chains without special processing	No image CRC checking	When debugging an application
Single enhanced	Support optional CRC checking	More complex to build than legacy image, CRC generation can be complex, may increase boot time	Production systems that benefit from image validation

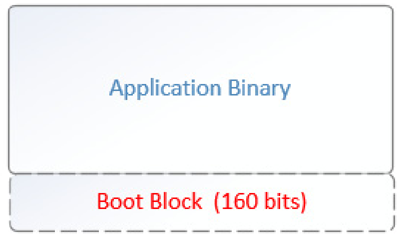


Fig 13. Image layout

Image layout is shown as Figure 13. Note:

- For legacy image, the boot block is not needed.
- Boot block for single enhanced image can be added by a tool named Enhanced Image Generator.

Image vector

The vector table contains the standard ARM vector as shown in Table 83.

Table 83: Image vector table

Offset	Description
0x0~0x1B	Vector 1~7(standard ARM vectors)
0x1C	Vector check sum
0x20	Code Read Protection (CRP)
0x24	Image type (Legacy or Single Enhanced) 0xEDDC9494: Single enhanced 0x0FFEB6B6: Reserved for Dual enhanced Others: Legacy
0x28	Boot Block pointer (Address offset in the flash)
0x2C~0x10F	Interrupt vector table
0x110	[7:6] SMART_EN 10: SMART write enabled to ISP Others: Disabled [5:4] USB_DIS 10: USB disabled to ISP Others: Enabled [3:2] UART_DIS 10: UART disabled to ISP Others: Enabled [1:0] SPI_DIS 10: SPI disabled to ISP Others: Enabled

Image CRP

Table 84: CRP bit description

Offset	Description
7:0	The number of Pages protected against Erase/Write during ISP, which is equal to (255 - N). N is defined by these 8 bits. 0xFF: N=255, the number of pages protected is 0, which indicates none of the pages is protected against Erase/Write during ISP 0xFE: N=254, the number of pages protected is 1, 1 page (page 0) is protected … 0x0: 255 pages are protected Note: The last page is always protected if page 0 is protected.
9:8	Reserved
11:10	Mass erase is allowed or not. 10: Mass erase is allowed to ISP 01: Mass erase is not allowed to ISP 10: Reserved 11: Reserved
13:12	Page erase/write is allowed or not. 10: Page erase/write is allowed to ISP, but protection is applicable 01: Page erase/Write is not allowed to ISP 10: Reserved 11: Reserved
15:14	Flash read is allowed or not. 10: Flash read is allowed to ISP 01: Flash read is not allow to ISP 10: Reserved 11: Reserved
17:16	ISP entry is allowed or not. 10: ISP entry is allowed via CHIP_MODE pin 01: ISP entry is not allow via CHIP_MODE pin 10: Reserved 11: Reserved
19:18	External access is allowed or not. 10: Enable external access to the chip. This option enables SWD. 01: Disable external access to chip. This option disable SWD. 10: Reserved 11: Reserved
31:20	Reserved

Image -Book Block

The Boot Block must be placed at the end of the image and is composed as shown in Table 85.

The QN908x boot loader supports up to two flash images, as shown in Figure 12. Each flash image has a boot information section to store data about the flash image. The boot info is defined like a structure, with the 32-bit elements shown below.

Table 85: Boot Block description

Offset	Description
0x0	Header marker (0xFEEDA5A5)
0x4	0: Check CRC 1: Do not check CRC (All other values are invalid, will not boot)
0x8	Image length For secure image the length should be the full length of the image except the signature block at the end. For non-secure image this can be any non-zero value for which the user decides to calculate the CRC. Applications requiring fast boot time can have partial CRC check by altering this length field. Note, the CRC value field should be calculated for the length of image specified in this field.
0x0C	CRC value [1](applicable to non-secure image)
0x10	Image version

[1] This CRC value is computed with CRC32 polynomial as follows: 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 = 0xFFFFFFFF

7.3.1.3 Image

When the ROM boots an image from the flash, the flash image needs to configure the REMAP bit field in the SYS_MODE_CTRL register to remap the flash to address 0x0. If the flash image does not start at address 0x0 the VTOR register will also need to be changed in order to correctly point to the vector table in flash image.

7.3.1.4 User information (NVDS)

The QN908x reserves a page of the flash for user information, called Nonvolatile Data Storage (NVDS). Each entry in the NVDS contains a tag ID and data value. When a tag's value needs to be updated, the old tag is marked as invalid and a new tag with same tag ID is appended after the last valid entry. When the NVDS space is exhausted, it should be deleted to release all tags marked invalid. In order to prevent NVDS data loss in the event of power cut suddenly during the operation, the NVDS data should be backed up in the NVDS backup page.

Although the user information's address and size is configurable and indicated in the boot information, the fixed location and size of 2 KB is recommended.

7.3.1.5 System configuration

The system configuration occupies the last page of flash with memory lock and protection configuration. Please refer to Section 28.3.2 for details.

Table 86: System configuration description

Offset	Description
0x7F800	Lock bits, page 0~31
0x7F804	Lock bits, page 32~63
0x7F808	Lock bits, page 64~95
0x7F80C	Lock bits, page 96~127
0x7F810	Lock bits, page 128~159
0x7F814	Lock bits, page 160~191
0x7F818	Lock bits, page 192~223
0x7F81C	Lock bits, page 223~255
0x7F820	Lock bits, page 192~223
0x7F824	Mass erase lock and memory protection bits
0x7F828	NVDS size

7.4 Boot process

7.4.1 Startup flow chart

The following figures show the startup process of the QN908x

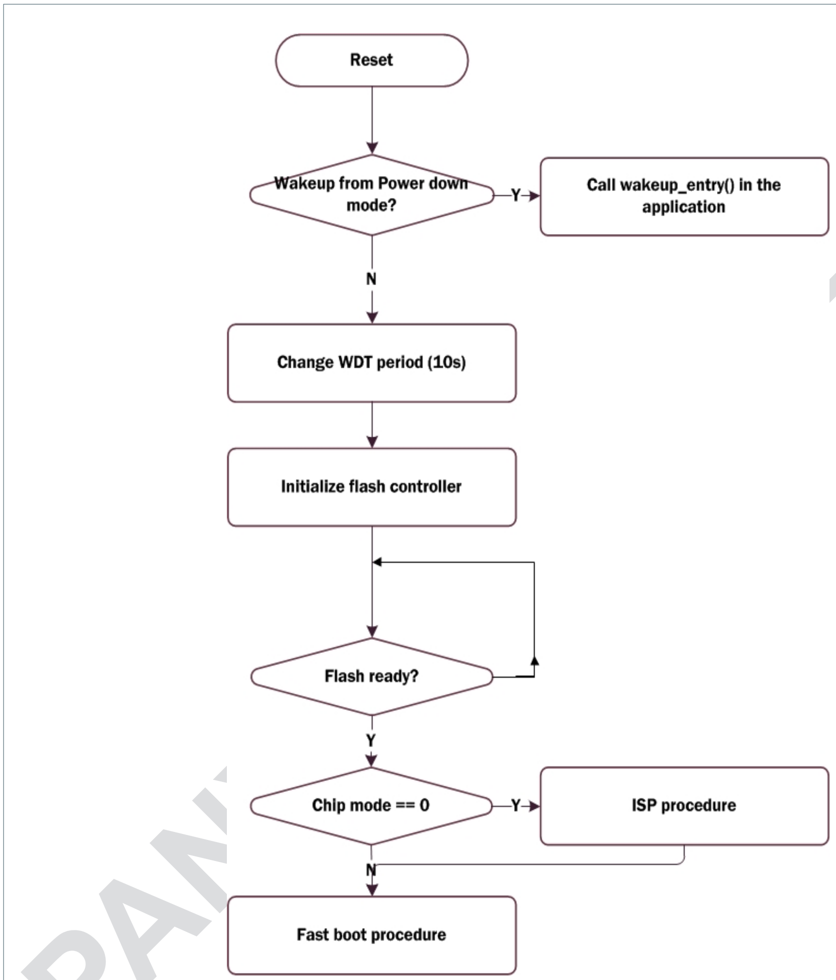


Fig 14. Startup flowchart

Remark: For details of INI_RD_DONE and FSH_STA, see Table 431

7.4.2 Fast boot flowchart

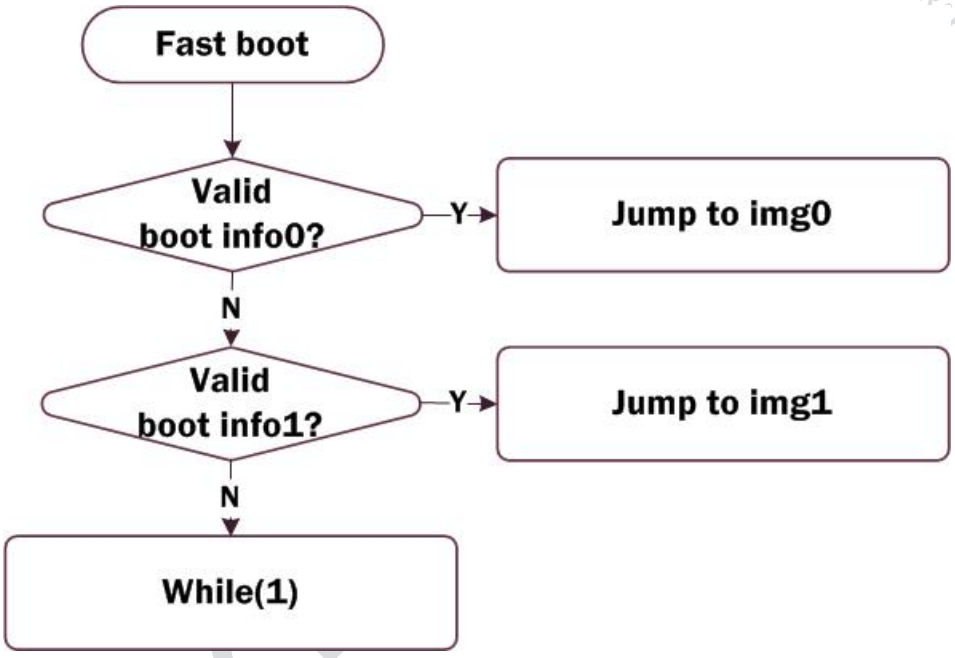


Fig 15. Fast boot flowchart

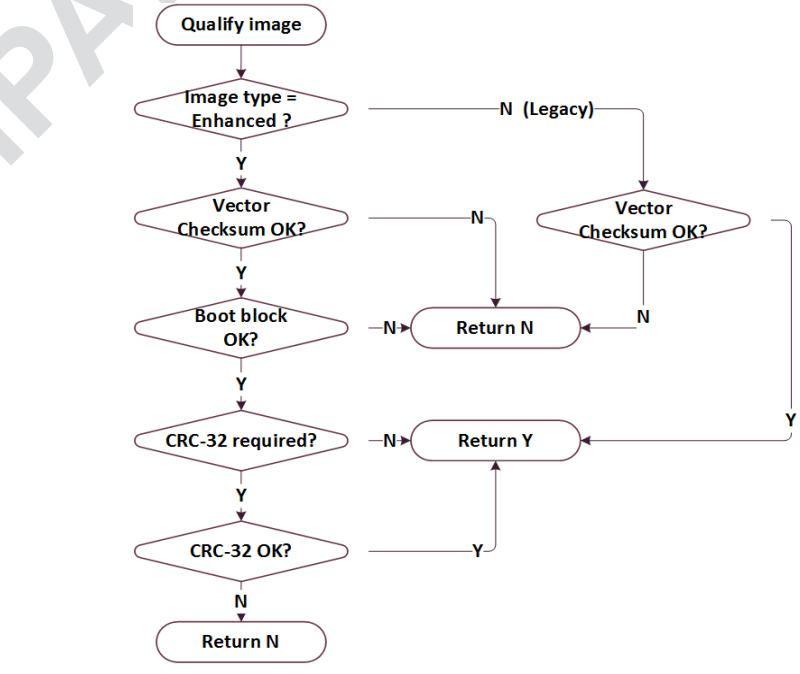


Fig 16. Image qualify flowchart

7.5 UART/SPI ISP

7.5.1 ISP transportation layer

The bootloader supports UART and SPI as the protocol transportation layer for interacting ISP protocol PDU to implement image programming.

Flexcomm0 UART and Flexcomm3 SPI are used and the pad is from PA14 to PA17.

The QN908x is able to lock specific pages in the flash to prevent unintentional modification. Please see Section 7.4.1.4 Boot information and Section 28.3.2 for more details.

The transportation layer frame format as below:

- UART Frame Format: 1bit start + 8bit data + 1bit stop
- SPI Frame Format: 8 bit data width, LSB transmitting first, Mode 0 (CPOL: 0, CPHA: 0)

7.5.2 ISP flowchart

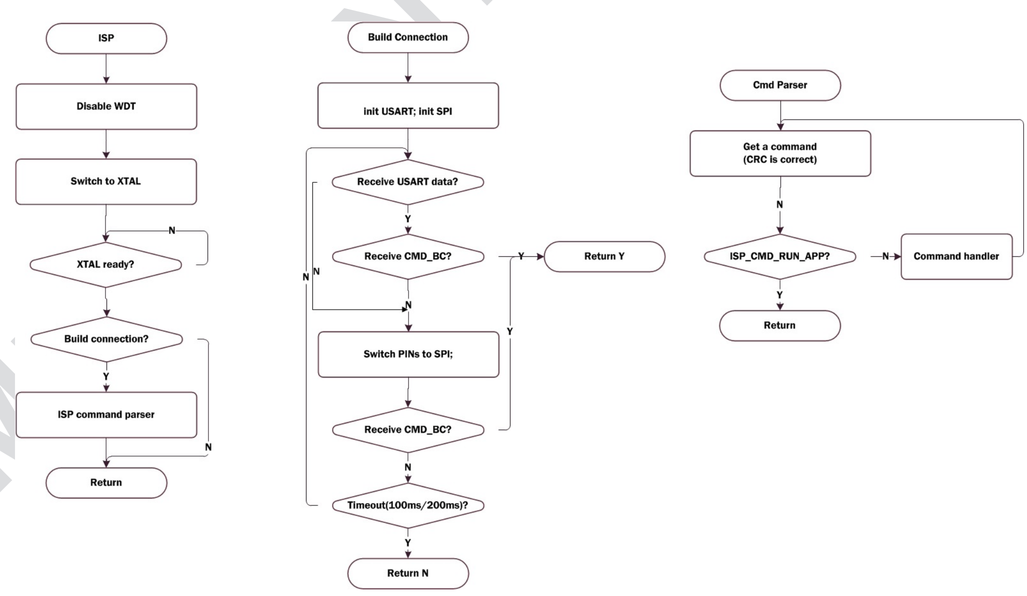


Fig 17. ISP flowchart

7.5.3 ISP Protocol Data Unit (PDU) format

All two interfaces use the same ISP PDU format when downloading application. The format of PDU is defined as below:

Fig 18. ISP PDU format

- Head Code field is the first byte of a PDU, which is transmitted first. The Head Code is 0x71 if there is payload in the PDU. The Head Code is 0x72 if there is no payload in the PDU.
- Command field contains the value for the requested ISP command.
- Data Length field indicates the number of bytes in the Data field. Do not include the CRC field in the length value.
- Data field contains the payload of the PDU. The maximum data length is 2048.
- CRC field contains the combined 16-bit checksum value for the Command field, Data Length field and Data field. The CRC16 polynomial is X^16+X^12+X^5+1.
- Confirm message is sent by boot loader to acknowledge whether the PDU has been received correctly.
- Response field (0x10) is sent by boot loader when the command needs data feedback.

The return values and error codes in the Confirm message are shown in the table below.

Table 87: BLE confirm and value

Confirm	Description	Value
ISP_CFM_OK	No error.	0x11
ISP_CFM_ERR_LEN	The PDU Data Length field contains an invalid value.	0x12
ISP_CFM_ERR_CRC	The calculated PDU CRC value does not match the value stored in the CRC field.	0x13
ISP_CFM_ERR_UNSUPPORTED_CMD	The PDU Command field contains an unknown value.	0x14
ISP_CFM_ERR_INVALID_PARAM	The flash page read, write, or erase command contains an invalid address or length.	0x15
ISP_CFM_ERR_PROTECTION	The flash read or write command failed due to the flash protection bit being active.	0x16
ISP_CFM_ERR_VERIFY	The flash image CRC value in the boot information does not match the calculated flash image CRC value.	0x17
ISP_CFM_ERR_FLH_CE	The flash chip erase command failed.	0x18
ISP_CFM_ERR_FLH_PE	The flash page erase command failed.	0x19
ISP_CFM_ERR_FLH_WR	The flash write command failed.	0x1A
ISP_CFM_ERR_BOOT_INFO	The APP_STATUS value in the boot information does not equal 0x55aaaa55, indicating that the flash image is invalid.	0x1B

7.5.4 ISP commands

The following table details the different ISP commands available. See Section 7.5.3 for the packet format for ISP command.

Table 88: ISP commands

Code	Command	Description
0x33	ISP_CMD_B_C	Build connection with boot loader.
0x50	ISP_CMD_SET_CLK_BR	Set system clock and USART0 baud rate.
0x51	ISP_CMD_CE_FLASH_MAIN	Erase Flash's main memory by chip. If this command is executed, the flash will be unprotected.
0x52	ISP_CMD_PE_FLASH_MAIN	Erase Flash's main memory by page.
0x53	ISP_CMD_RD_FLASH	Read a specific region in the flash.
0x54	ISP_CMD_WR_FLASH	Write to a specific region in the flash.
0x55	ISP_CMD_VERIFY_APP	Verify the app_status and CRC fields in the boot information.
0x56	Reserved	-
0x57	ISP_CMD_VERSION	Read hardware and firmware version.
0x58	ISP_CMD_FLASH_CFG	Set the flash configuration. See Section 28 for more information.
0x59	ISP_CMD_FLASH_INI	Initiate the flash by setting the INI_RD_EN bit in the flash initial read enable register. See Section 28.4.1 for more information.

7.5.5 ISP PDU interaction

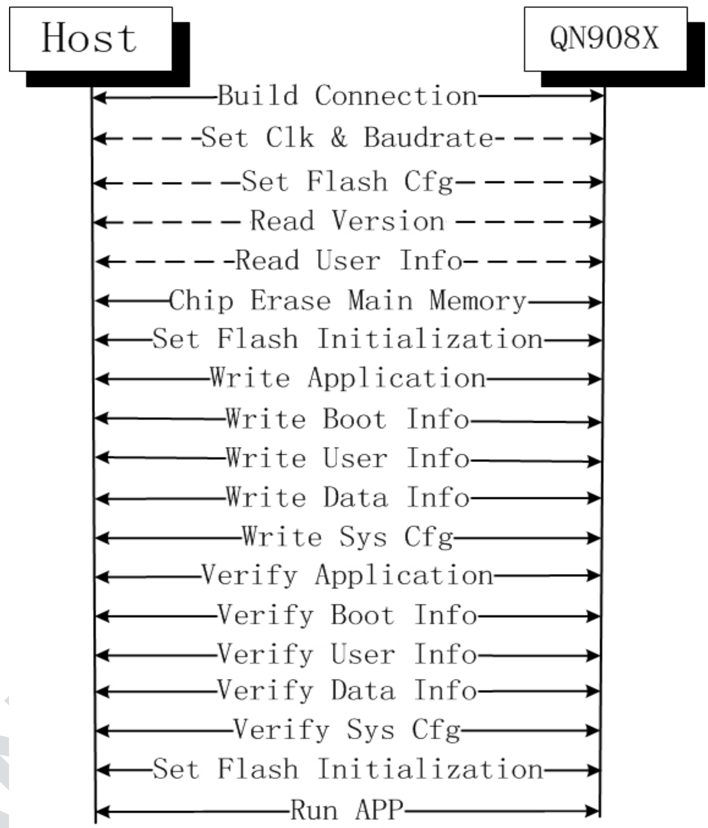


Fig 19. ISP PDU interaction

- Select USART or SPI to send the ISP_CMD_B_C to initiate the ISP connection.
- Configure the system clock and USART baud rate with the ISP_CMD_SET_CLK_BR command.
- If desired, set a custom flash configuration with the ISP_CMD_FLASH_CFG command. It is recommended to use the default values.
- Read the hardware and firmware version if needed with the ISP_CMD_ VERSION command.
- ISP could read user information back if do not want to change it.
- Erase whole main flash with the ISP_CMD_CE_FLASH_MAIN command.
- Initiate the flash to load lock information with the ISP_CMD_FLASH_INI command.
- Write the application, boot information, user information, data information, OTA and system configuration using the ISP_CMD_WR_FLASH command.
- Check application integrity with the ISP_CMD_VERIFY_APP command.
- Read back user information to check integrity.
- Read back data information to check integrity.
- Read back OTA to check integrity.

- Read back the system configuration to check integrity.

7.5.6 ISP PDU description

The procedure of building connection with bootloader.

This format of ISP_CMD_B_C is different from the other ISP commands. There is only one byte in this command, no head code and length, payload and CRC. The host continuously sends connection command until it receives the confirmation which represents that the bootloader has selected interface and entered into ISP command parser or timeout.

The procedure of setting UART/SPI baud rate.

The payload in ISP_CMD_SET_CLK_BR is two-word data which represents the value of system clock control register and USART0 baud rate register. After ISP_CMD_SET_CLK_BR is confirmed, the new baud rate will be used for subsequent PDU exchange.

Baud rate format: 4bytes for 3 parameters:

- uint16_t brgval;
- uint8_t osrval;
- uint8_t frac_mult;

The procedure of erasing flash main memory by chip erase

The procedure of erasing flash main memory by page erase

The procedure of reading flash

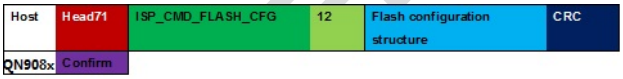
or

The procedure of writing flash

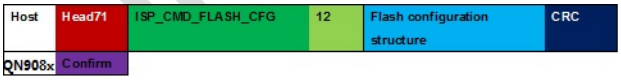
The procedure of verifying application

The procedure of running application

The format of version reading command



The format of flash configuring command



7.5.7 ISP Security

The code security is decided by the image itself, there is a 32bit configuration named CRP (Code Read Protection) to define the security levels, here list some typical security levels.

Table 89: Typical cases of CRP

CRP level	SWD	ISP	Read	Write/ Erase	Mass Erase	Number of Protected Pages	Description	CRP data
No CRP	10(√)	10(√)	10(√)	10(√)	10(√)	0xFF	By default (No image)	0x000AA8FF
CRP1	01(X)	10(√)	01(X)	10(√)	01(X)	XX	At least Page 0 and Page 255 is protected	0x000664FE
CRP2	01(X)	10(√)	01(X)	10(√)	10(√)	<= 0xFE	At least Page 0 and Page 255 is protected, but support chip erase	0x000668FE
CRP3	01(X)	01(X)	XX(X)	XX(X)	XX(X)	XX	No ISP and No SWD	0x00050000 0x00000000 0xFFFFFFFF
No ISP	10(√)	01(X)	XX(X)	XX(X)	XX(X)	XX	Only support SWD	0x00090000

Table 90: CRP levels for USB ISP

CRP level	Volume label	Description
No CRP	CRP DISABLD	The user flash can be read or written
CRP1	CRP ENABLED	The user flash content cannot be read but can be updated, The flash memory sectors are updated depending on the new firmware image.
CRP2	CRP ENABLED	The user flash content cannot be read but can be updated. The entire user flash memory will be erased before writing the new firmware image.
CPR3	--	The user flash content cannot be read or updated, The boot loader always executes the user application if valid.

7.6 USB ISP

7.6.1 USB firmware structure

A typical structure of the firmware.bin for USB ISP shows as bellow:

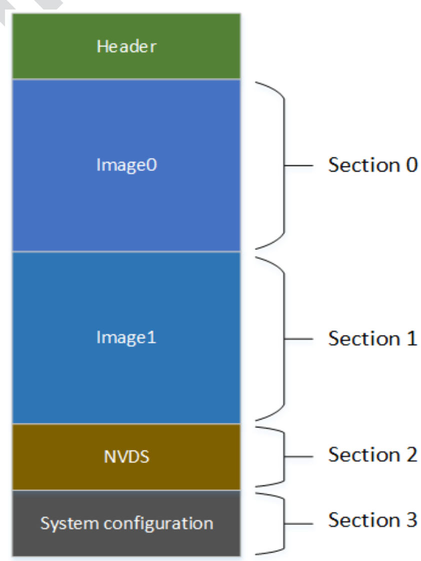


Fig 20. USB ISP structure

The format of header is shown as below as table bellow:

Table 91: USB ISP header

Offset	Description
0x00	Header Marker (0xFEEDA5A5)
0x04	[31:24] Smart max erase times [23:16] Smart write times [15:0] Program cycle
0x08	Erase time base

Table 91: USB ISP header

Offset	Description
0x0C	Write time base
0x10	System clock
0x14	Section count, the max value is 6
0x18	Section0 start offset
0x1C	Section0 length
0x20	Section0 CRC[1]
0x24	Section1 start offset
0x28	Section1 length
0x2C	Section1 CRC[1]
0x30	Section2 start offset
0x34	Section2 length
0x38	Section2 CRC[1]
...	...

[1] This CRC value is computed with CRC32 polynomial as follows: 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 = 0xFFFFFFFF

The length of image header is fixed, which is 96 (24 + 12 * 6)

7.6.2 USB ISP flowchart

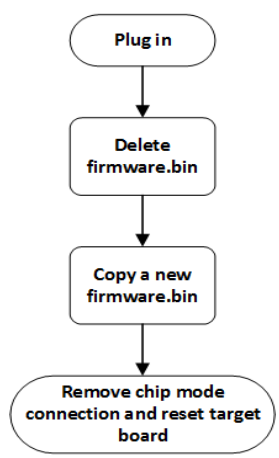


Fig 21. USB ISP general process

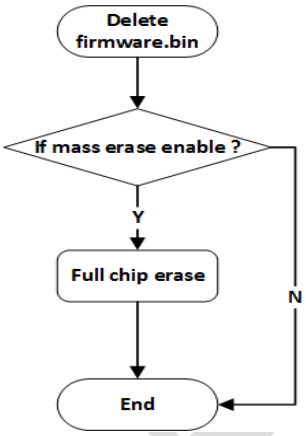


Fig 22. USB ISP delete process

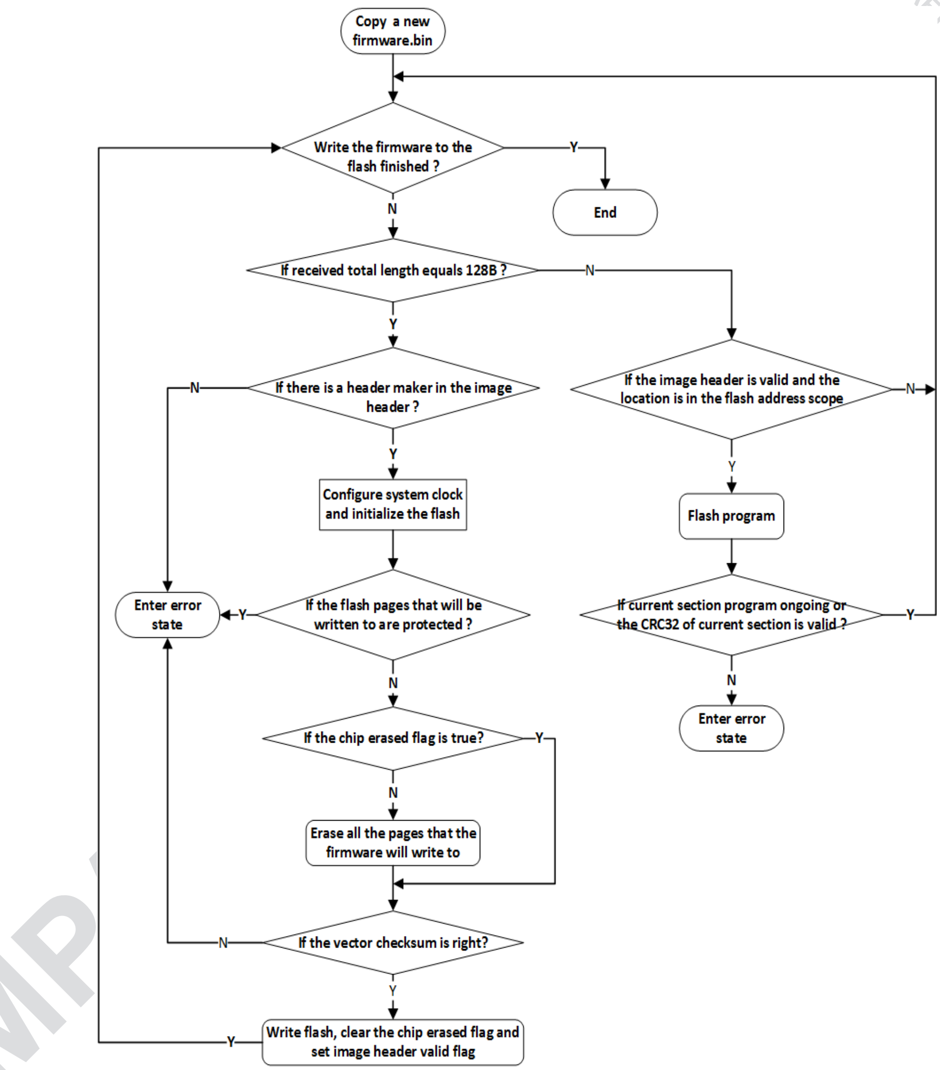


Fig 23. USB ISP image copy process

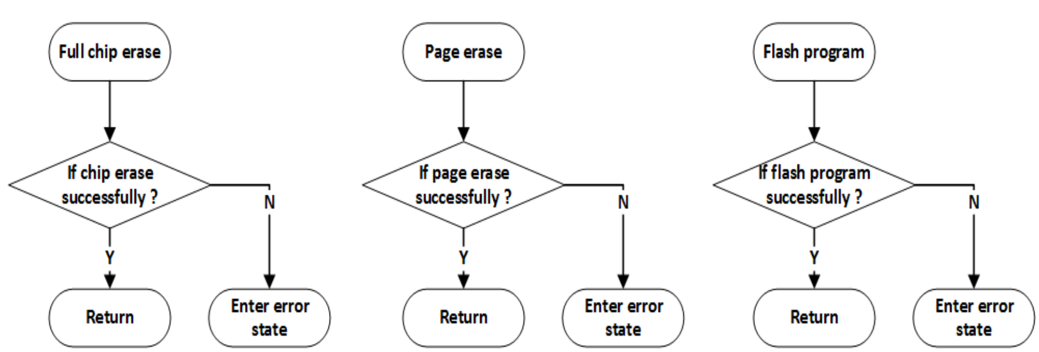


Fig 24. Flash operation process

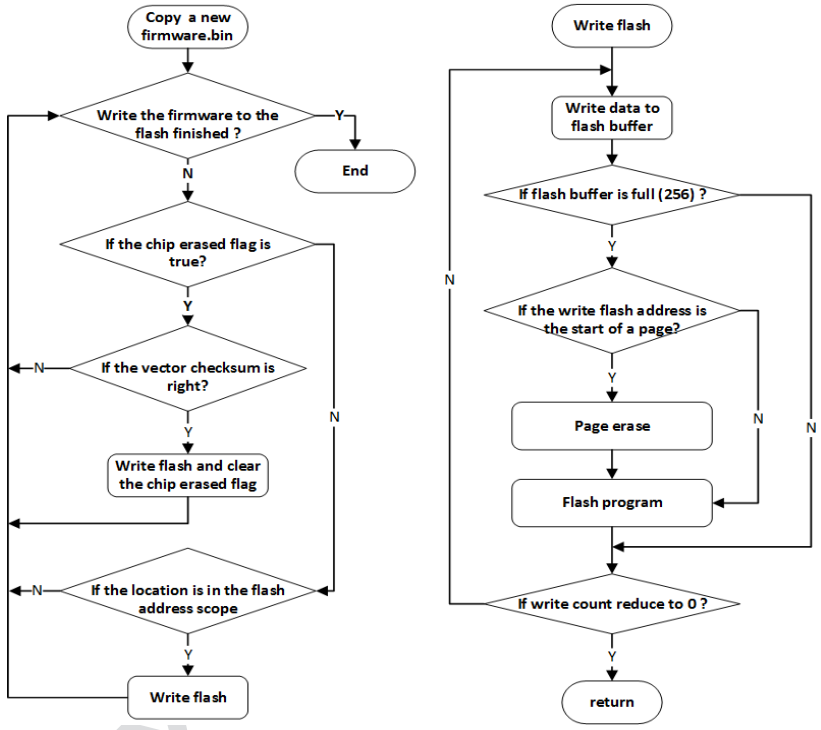


Fig 25. Legacy copy a new firmware binary

8. I/O configuration and pin MUX

8.1 Introduction

The I/O pin configuration and pin MUX are available in QN908x family.

Registers for pins that are not available on a specific package are reserved.

Table 92. Available pins and configuration registers

Package	USB	Total GPIOs	GPIO A	GPIO B
47-bumps (WLCSP)	yes	28	PA00 to PA31(no PA02, PA03, PA12, PA13, PA20, PA21, PA28)	PB00 to PB02
48-pins (HVQFN)	yes	35	PA00 to PA31	PB00 to PB02

8.2 Features

The following electrical properties are configurable for standard port pins:

- Pull-up/pull-down resistor
- Drive strength
- Flexible configuration of pin function
  - A 7-to-1 multiplexing for pin function

– Output 32 kHz or high-speed crystal clock in sleep mode
– Cortex-M4 debug trace port
– BLE and WLAN combo pin
– External tuner control pin output for which polarity is configurable

8.3 General description

8.3.1 Pin configuration

To be different with other PAD, the PAD has extra logic to control analog input and output, and the control signal is named as aen. When aen is effective, the PAD is connected to analog module, and digital input and output are switched off.

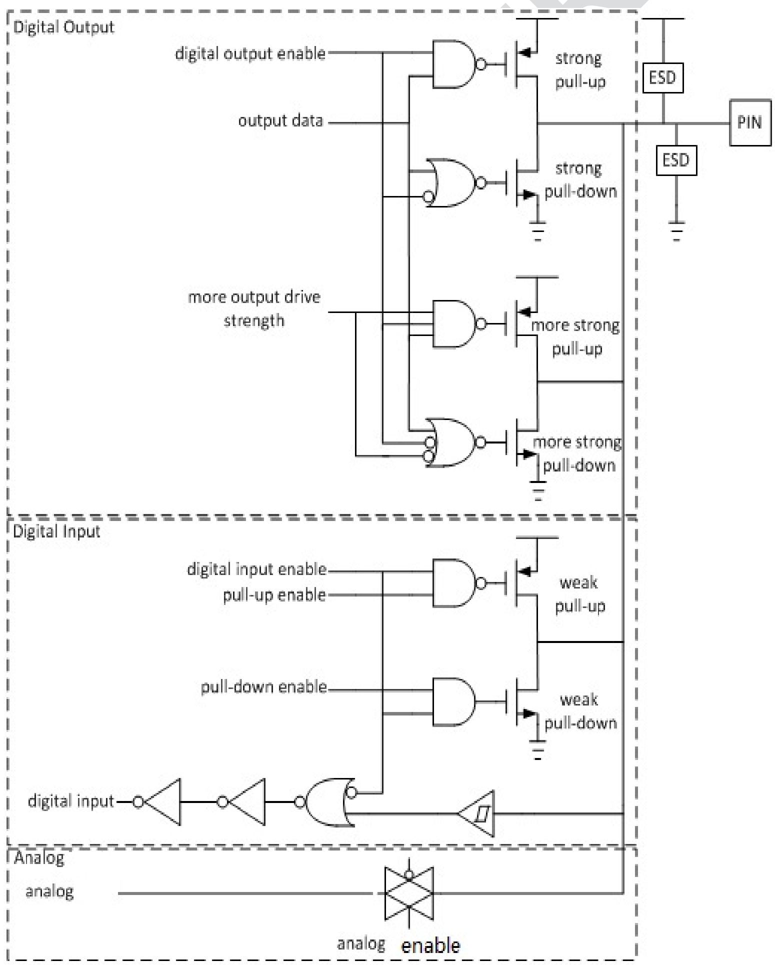


Fig 26. Pin configuration

8.3.1.1 Pull control

Our chip provides flexible pull control. Each PIN has three kinds of status:

- Pull up
- Pull down

- High-Z

While pin is in input mode, the pin status is defined by PIO_PULL_CFG0, PIO_PULL_CFG1 and PIO_PULL_CFG2 in SYSCON.
While pin is used as analog function or digital output mode, this pull control register will not take effect.

Each pin has two bits to control the status:

Table 93. Pin pull control status definition

Pull control register of each pin	Status
00b	high-Z
01b	pull-down
10b	pull-high
11b	reserved

8.3.1.2 Input enable

Input enable of pin is also configured through register. When it is enabled, input signal can be input into internal circuit, otherwise not.
The input enable register is located at address 0x40000828 and 0x4000082C, named PIO_IE_CFG0 & PIO_IE_CFG1.

8.3.1.3 Drive strength

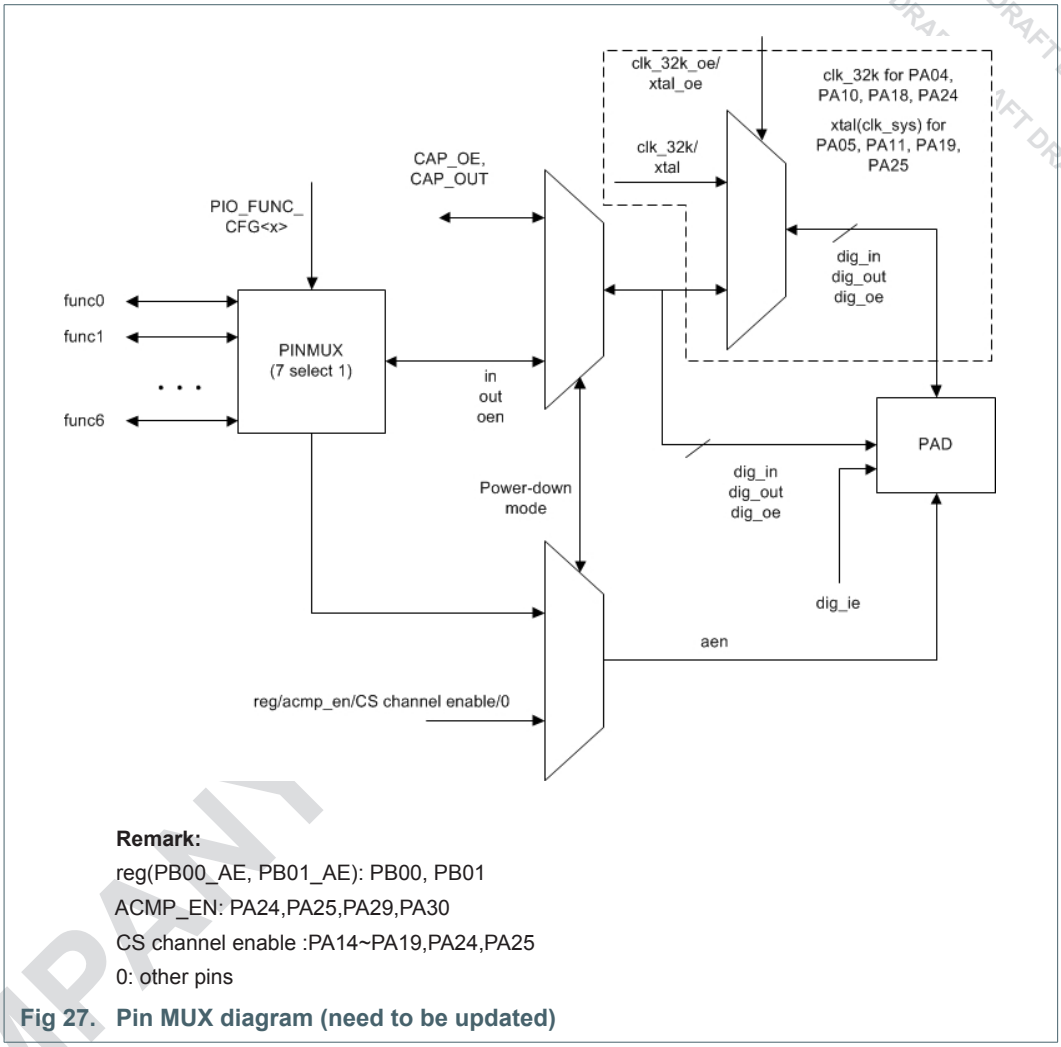
Ordinary pin has two levels of pin strength, except PA6, PA11, PA19, PA26 and PA27, which have extra driven strength compared to ordinary pins.
The driven strength registers are PIO_DRV_CFG0 & PIO_DRV_CFG1, while extra driven strength register is PIO_DRV_CFG2. These registers are described in Section 2. The drive current can be listed as given in the following table:

Table 94. Drive current (replace the number with defined levels?)

Current (mA)	DRV_CTRL[x]	DRV_EXTRA
1.6	0	0
6.4	1	0
14.6	0	1
19.4	1	1

8.3.2 Pin MUX

Pin MUX is a design for multiplexing multiple functions in one pin by programming corresponding Pin MUX control registers.
The total pin MUX block diagram can be shown as below:



Remark:

reg(PB00_AE, PB01_AE): PB00, PB01

ACMP_EN: PA24,PA25,PA29,PA30

CS channel enable :PA14~PA19,PA24,PA25

0: other pins

Fig 27. Pin MUX diagram (need to be updated)

8.3.2.1 Pin Mux table

Table 95. Pin MUX table (please refer to PIO_FUNC_CFG0~3 for pin function configuration)

Name	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
PA00	GPIOA0(I/O)	ADC0(AI)	SCT0_OUT0(O)	CTIMER0_CAP0(I)	FC0_RTS(O)	FC2_SSEL3(I/O)	WLAN_TX(I)
PA01	GPIOA1(I/O)	ADC1(AI)	SCT0_OUT1(O)	CTIMER0_CAP1(I)	FC0_CTS(I)	FC2_SSEL2(I/O)	WLAN_RX(I)
PA02	GPIOA2(I/O)	QDEC0_A(I)	SCT0_OUT2(O)	CTIMER0_MAT0(O)	Reserved	FC2_SCL_SSEL1(I/O)	RFE_RX_EN(O)
PA03	GPIOA3(I/O)	QDEC0_B(I)	SCT0_OUT3(O)	CTIMER0_MAT1(O)	Reserved	FC2_SDA_SSEL0(I/O)	RFE_TX_EN(O)
PA04	GPIOA4(I/O)	ADC2(AI)	SCT0_OUT4(O)	CTIMER0_MAT0(O)	FC0_TXD(O)	FC2_SDA_MOSI(I/O)	SPIFI_IO0(I/O)
PA05	GPIOA5(I/O)	ADC3(AI)	SCT0_OUT5(O)	CTIMER0_MAT1(O)	FC0_RXD(I)	FC2_SCL_MISO(I/O)	SPIFI_IO1(I/O)
PA06	GPIOA6(I/O)	Reserved	SCT0_OUT3(O)	CTIMER0_MAT2(O)	FC1_RTS_SCL(I/O)	BLE_PTI0(O)	SPIFI_CLK(O)

Table 95. Pin MUX table (please refer to *…continued*PIO_FUNC_CFG0~3 for pin function configuration)

Name	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
PA07	GPIOA7(I/O)	ADC_VREFI(AI)	SCT0_OUT2(O)	CTIMER1_CAP0(I)	FC1_CTS_SDA(I/O)	BLE_PTI1(O)	SPIFI_CSN(O)
PA08	GPIOA8(I/O)	ADC4(AI)	SCT0_IN0(I)	CTIMER1_CAP1(I)	FC1_TXD_SCL(I/O)	BLE_PTI2(O)	SPIFI_IO2(I/O)
PA09	GPIOA9(I/O)	ADC5(AI)	SCT0_IN1(I)	CTIMER1_MAT0(O)	FC1_RXD_SDA(I/O)	BLE_PTI3(O)	SPIFI_IO3(I/O)
PA10	GPIOA10(I/O)	ADC6(AI)	SCT0_IN2(I)	CTIMER1_MAT1(O)	FC1_SCK(I/O)	ACMP0_OUT(O)	BLE_TX(O)
PA11	GPIOA11(I/O)	ADC7(AI)	SCT0_IN3(I)	CTIMER1_MAT2(O)	FC2_SSEL2(I/O)	ACMP1_OUT(O)	BLE_RX(O)
PA12	GPIOA12(I/O)	Reserved	SCT0_OUT5(O)	ACMP0_OUT(O)	FC1_TXD_SCL(I/O)	SD_DAC(O)	ANT_SW(O)
PA13	GPIOA13(I/O)	Reserved	SCT0_OUT4(O)	ACMP1_OUT(O)	FC1_RXD_SDA(I/O)	FC3_SSEL1(I/O)	RFE_EN(O)
PA14	GPIOA14(I/O)	CS0(AI)	ANT_SW(O)	CTIMER2_CAP0(I)	FC0_RTS(O)	FC3_SSEL0(I/O)	QDEC1_A(I)
PA15	GPIOA15(I/O)	CS1(AI)	SCT0_OUT0(O)	CTIMER2_CAP1(I)	FC0_CTS(I)	FC3_SCK(I/O)	QDEC1_B(I)
PA16	GPIOA16(I/O)	CS2(AI)	SCT0_OUT1(O)	CTIMER2_MAT0(O)	FC0_TXD(O)	FC3_MOSI(I/O)	QDEC0_A(I)
PA17	GPIOA17(I/O)	CS3(AI)	SD_DAC(O)	CTIMER2_MAT1(O)	FC0_RXD(I)	FC3_MISO(I/O)	QDEC0_B(I)
PA18	GPIOA18(I/O)	CS4(AI)	SCT0_OUT3(O)	CTIMER2_MAT2(O)	FC0_SCK(I/O)	FC3_SSEL2(I/O)	BLE_SYNC(O)
PA19	GPIOA19(I/O)	CS5(AI)	SCT0_OUT2(O)	RFE_EN(O)	FC0_SCK(I/O)	FC3_SSEL3(I/O)	BLE_IN_PROC(O)
PA20	GPIOA20(I/O)	QDEC1_A(I)	SCT0_OUT1(O)	CTIMER2_MAT0(O)	SWO(I/O)	FC1_RTS_SCL(I/O)	SPIFI_CLK(O)
PA21	GPIOA21(I/O)	QDEC1_B(I)	SCT0_OUT0(O)	CTIMER2_MAT1(O)	FC2_SSEL3(I/O)	FC1_CTS_SDA(I/O)	SPIFI_CSN(O)
PA22	SWCLK(I/O)	GPIOA22(I/O)	SCT0_IN2(I)	CTIMER3_MAT0(O)	FC2_SDA_SSEL0(I/O)	FC3_SSEL3(I/O)	QDEC1_A(I)
PA23	SWDIO(I/O)	GPIOA23(I/O)	SCT0_IN3(I)	CTIMER3_MAT1(O)	FC2_SCL_SSEL1(I/O)	FC3_SSEL2(I/O)	QDEC1_B(I)
PA24	GPIOA24(I/O)	ACMP0N/CS6(AI)	ETM_TRACEDAT0(O)	CTIMER3_CAP0(I)	RFE_RX_EN(O)	FC3_SSEL1(I/O)	SPIFI_IO0(I/O)
PA25	GPIOA25(I/O)	ACMP0P/CS7(AI)	ETM_TRACEDAT1(O)	CTIMER3_CAP1(I)	RFE_TX_EN(O)	FC3_SSEL0(I/O)	SPIFI_IO1(I/O)
PA26	GPIOA26(I/O)	USB_DP(A)	SCT0_IN0(I)	CTIMER1_MAT0(O)	FC2_SDA_MOSI(I/O)	QDEC0_A(I)	BLE_SYNC(O)
PA27	GPIOA27(I/O)	USB_DM(A)	SCT0_IN1(I)	CTIMER1_MAT2(O)	FC2_SCL_MISO(I/O)	QDEC0_B(I)	BLE_IN_PROC(O)
PA28	GPIOA28(I/O)	CLK_AHB(O)	ETM_TRACECLK(O)	RTC_CAP(I)	FC1_SCK(I/O)	SD_DAC(O)	SPIFI_CSN(O)
PA29	GPIOA29(I/O)	ACMP1N(AI)	ETM_TRACEDAT2(O)	CTIMER3_MAT0(O)	FC2_SCK(I/O)	FC3_MISO(I/O)	SPIFI_IO2(I/O)

Table 95. Pin MUX table (please refer to *...continued*PIO_FUNC_CFG0~3 for pin function configuration)

Name	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
PA30	GPIOA30(I/O)	ACMP1P(AI)	ETM_TRACEDAT3(O)	CTIMER3_MAT1(O)	FC2_SCK(I/O)	FC3_MOSI(I/O)	SPIFI_IO3(I/O)
PA31	GPIOA31(I/O)	DAC(AO)	RTC_CAP(I)	CTIMER3_MAT2(O)	SWO(I/O)	FC3_SCK(I/O)	SPIFI_CLK(O)
PB00	GPIOB00(I/O)	XTAL32_OUT					
PB01	GPIOB01(I/O)	XTAL32_IN					
CHIP_MODE/PB02	GPIOB2(I/O)	ANT_SW(O)					

Remark: In CSP package, there is no PA02, PA03, PA12, PA13, PA20, PA21, PA28

8.3.2.2 Analog enable

1. PA00~PA31: Analog enable signal can be generate in pin MUX logic in ordinary mode
2. PB00, PB01: as XTAL32K input, so the analog enable should be 1 to let the 32k XTAL clock input to analog module, which are defined by PIO_CFG_MISC register bit 1 and bit 0 which is described in Section 2

8.3.2.3 Chip mode function

PB02 is an ordinary GPIO pin used to indicate if fast boot can be performed.

When PB02_MODE(bit 16 of PIO_CFG_MISC register) is 0, PB02 pin is used as chip boot mode input. If PB02 is 0, bootloader enters ISP process. If PB02 is 1, bootloader jumps to flash to execute application without ISP process.

When PB02_MODE is1, PB02 is used as antenna output.

8.3.3 I/O status under power down

In active mode, I/O status is defined by user configuration. While before entering power down mode, user need to capture the I/O status, so that it will hold in power down mode. The flow is:

- Write 1 to IO_CAP register described in Section 2, to capture the IO status
- Write PDM_IO_SEL bit to 1 in PIO_WAKEUP_EN1 register described in Section 2. After that, IO status is decided by the value captured in the registers
- Enter WFI to sleep
- Wakeup, and write PDM_IO_SEL bit to 0 in PIO_WAKEUP_EN1 register to recover the IO status from power down mode. After that, I/O status is user configurable

8.4 Register description

Please refer to SYSCON chapter for all PIO_xx_xx registers.

9. Input multiplexing (input MUX)

9.1 Introduction

Input multiplexing is present on all QN908x devices. Based on the package, all inputs from external pins may not be available.

9.2 Features

- Configures the inputs to the pin interrupt block and pattern match engine
- Configures the inputs to the DMA triggers

9.3 Basic configuration

CPU can always write to or read from the input MUX registers. The peripheral clock is not required.

9.4 Pin description

The input multiplexer does not have dedicated pins. However, all digital pins can be selected as inputs to the pin interrupts. Multiplexer inputs from external pins work independently of any other function assigned to the pin, as long as no analog function is enabled.

Table 96. INPUT MUX pin description

Pins	Peripheral	Section
any pin from PA00~PA31	pin interrupts 0 to 3	Section 9.6.1

9.5 General description

The inputs to the DMA triggers, and to the four pin interrupts are multiplexed to multiple input sources. The sources can be external pins, interrupts, or output signals of other peripherals.

The input multiplexing makes it possible to design event-driven processes without CPU intervention, by connecting peripherals like the ADC.

The DMA can use trigger input multiplexing to sequence DMA transactions without the use of interrupt service routines.

9.5.1 Pin interrupt input multiplexing

The input MUX for the pin interrupts and pattern match engine multiplexes all existing pins.

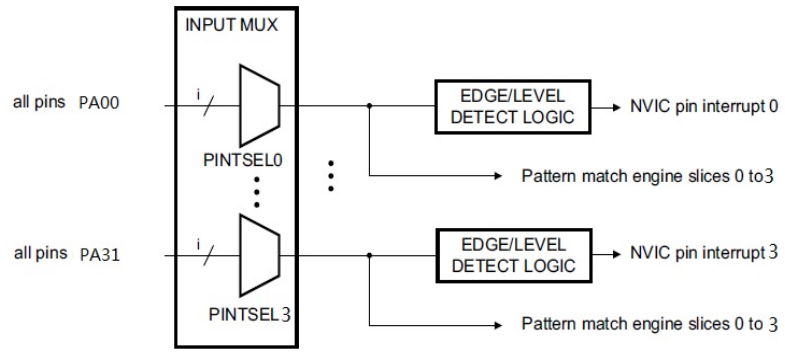


Fig 28. Pin interrupt multiplexing

9.5.2 DMA trigger input multiplexing

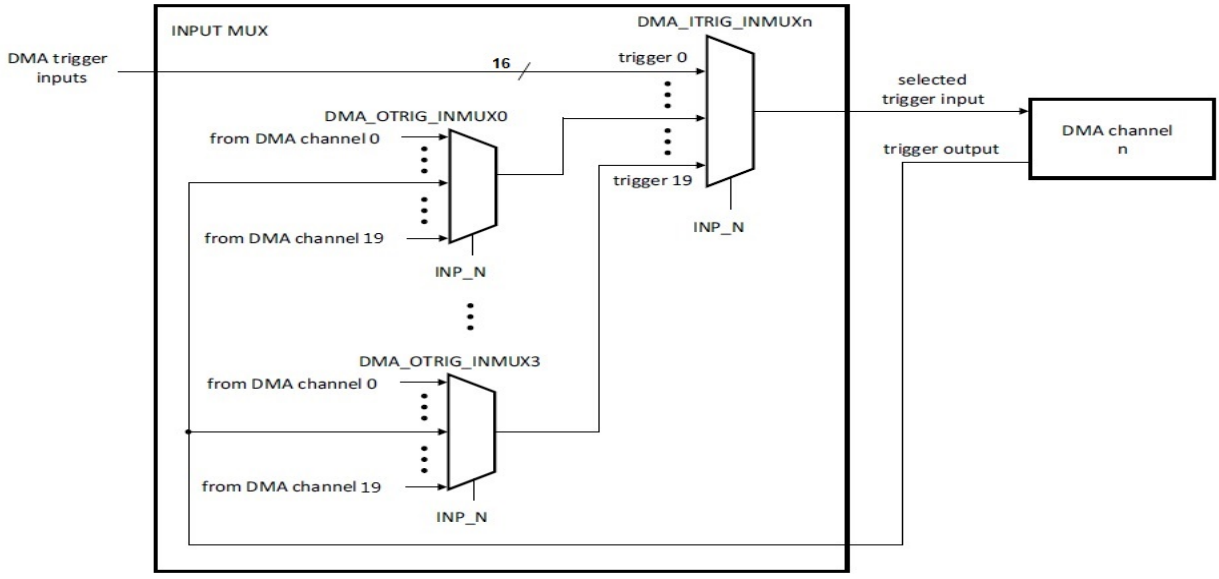


Fig 29. DMA trigger multiplexing

9.6 Register description

All input MUX registers reside on word address boundaries. Details of the registers appear in the description of each function.

All address offsets not shown in Table 97 are reserved and should not be written to.

Table 97. Register overview: Input multiplexing (base address 0x4000 6000)

Name	Access	Offset	Description	Reset value	Section
PINTSEL0	R/W	0x200	pin interrupt select register 0	0x0	Section 9.6.1
PINTSEL1	R/W	0x204	pin interrupt select register 1	0x0	Section 9.6.1
PINTSEL2	R/W	0x208	pin interrupt select register 2	0x0	Section 9.6.1
PINTSEL3	R/W	0x20C	pin interrupt select register 3	0x0	Section 9.6.1
DMA_ITRIG_INMUX0	R/W	0x400	trigger select register for DMA channel 0	0x1F	Section 9.6.2
DMA_ITRIG_INMUX1	R/W	0x404	trigger select register for DMA channel 1	0x1F	Section 9.6.2
DMA_ITRIG_INMUX2	R/W	0x408	trigger select register for DMA channel 2	0x1F	Section 9.6.2
DMA_ITRIG_INMUX3	R/W	0x40C	trigger select register for DMA channel 3	0x1F	Section 9.6.2
DMA_ITRIG_INMUX4	R/W	0x410	trigger select register for DMA channel 4	0x1F	Section 9.6.2
DMA_ITRIG_INMUX5	R/W	0x414	trigger select register for DMA channel 5	0x1F	Section 9.6.2
DMA_ITRIG_INMUX6	R/W	0x418	trigger select register for DMA channel 6	0x1F	Section 9.6.2
DMA_ITRIG_INMUX7	R/W	0x41C	trigger select register for DMA channel 7	0x1F	Section 9.6.2
DMA_ITRIG_INMUX8	R/W	0x600	trigger select register for DMA channel 8	0x1F	Section 9.6.2
DMA_ITRIG_INMUX9	R/W	0x604	trigger select register for DMA channel 9	0x1F	Section 9.6.2
DMA_ITRIG_INMUX10	R/W	0x608	trigger select register for DMA channel 10	0x1F	Section 9.6.2
DMA_ITRIG_INMUX11	R/W	0x60C	trigger select register for DMA channel 11	0x1F	Section 9.6.2
DMA_ITRIG_INMUX12	R/W	0x610	trigger select register for DMA channel 12	0x1F	Section 9.6.2
DMA_ITRIG_INMUX13	R/W	0x614	trigger select register for DMA channel 13	0x1F	Section 9.6.2
DMA_ITRIG_ INMUX14	R/W	0x618	trigger select register for DMA channel 14	0x1F	Section 9.6.2
DMA_ITRIG_INMUX15	R/W	0x61C	trigger select register for DMA channel 15	0x1F	Section 9.6.2
DMA_ITRIG_INMUX16	R/W	0x800	trigger select register for DMA channel 16	0x1F	Section 9.6.2
DMA_ITRIG_INMUX17	R/W	0x804	trigger select register for DMA channel 17	0x1F	Section 9.6.2
DMA_ITRIG_INMUX18	R/W	0x808	trigger select register for DMA channel 18	0x1F	Section 9.6.2
DMA_ITRIG_INMUX19	R/W	0x80C	trigger select register for DMA channel 19	0x1F	Section 9.6.2
DMA_OTRIG_INMUX0	R/W	0xA00	DMA output trigger selection to become DMA trigger 16	0x1F	Section 9.6.3
DMA_OTRIG_INMUX1	R/W	0xA04	DMA output trigger selection to become DMA trigger 17	0x1F	Section 9.6.3
DMA_OTRIG_INMUX2	R/W	0xA08	DMA output trigger selection to become DMA trigger 18	0x1F	Section 9.6.3
DMA_OTRIG_INMUX3	R/W	0xA0C	DMA output trigger selection to become DMA trigger 19	0x1F	Section 9.6.3

9.6.1 Pin interrupt select registers

Each of these four registers select one pin as the source of a pin interrupt, or as the input to the pattern match engine. To select a pin for any of the four pin interrupts or pattern match engine inputs, write the GPIO port pin number as 0 to 31 for pins PA00 to PA31 to

the INTPIN bits. For example, setting INTPIN to 0x5 in PINTSEL0 selects pin PA05 for pin interrupt 0. To determine the GPIO port pin number for a given device package, see the pin description table in the data sheet.

Each of the pin interrupts must be enabled in the NVIC (see Table 49) before it becomes active.

To use the selected pins for pin interrupts or the pattern match engine, see Section 11.5.2.

Table 98. Pin interrupt select registers (PINTSEL[0:3], offsets [0x200:0x20C]) bit description

Bit	Symbol	Description	Reset value
4:0	INTPIN	Pin number select for pin interrupt or pattern match engine input. (PA00 to PA31)	0x0
31:5	-	Reserved	-

9.6.2 DMA trigger input MUX registers 0 to 19

With the DMA trigger input MUX registers, one trigger input can be selected for each of the DMA channels from the potential internal sources. By default, none of the triggers are selected.

Table 99. DMA trigger input MUX registers (DMA_ITRIG_INMUX[0:19], offsets [0x400:0x80C]) bit description

Bit	Symbol	Description	Reset value
4:0	INP	trigger input number (decimal value) for DMA channel n (n: 0 to 19) 0: FSP DMA request 1: RTC second trigger 2: SCT0 DMA request 0 3: SCT0 DMA request 1 4: CTIMER0 match 0 5: CTIMER0 match 1 6: CTIMER1 match 0 7: CTIMER1 match 1 8: CTIMER2 match 0 9: CTIMER2 match 1 10: CTIMER3 match 0 11: CTIMER3 match 1 12: pin interrupt 0 13: pin interrupt 1 14: pin interrupt 2 15: pin interrupt 3 16: DMA output trigger MUX 0 17: DMA output trigger MUX 1 18: DMA output trigger MUX 2 19: DMA output trigger MUX 3	0x1F
31:5	RESERVED	-	-

9.6.3 DMA output trigger feedback MUX registers 0 to 3

This register provides a multiplexer for inputs 16 to 19 of each DMA trigger input MUX register DMA_ITRIG_INMUX. These inputs can be selected from among the trigger outputs generated by the each DMA channel. By default, none of the triggers are selected.

Table 100. DMA output trigger feedback MUX registers (DMA_OTRIG_INMUX[0:3], offset [0xA00:0xA0C]) bit description

Bit	Symbol	Description	Reset value
4:0	INP	DMA trigger output number (decimal value) for DMA channel n (n: 0 to 19)	0x1F
31:5	RESERVED	-	-

10. General purpose I/O

10.1 Introduction

QN908x supports up to 35 GPIO pins. Depending on the device and package type, a subset of those pins are available, and the unused bits in GPIO registers are reserved (see Table 92).

10.2 Basic configuration

Enable the clock to each GPIO pin by configuring bit CLK_GPIO_EN in the CLK_EN register(Section 2.5.4).

10.3 Features

- Accelerated GPIO functions:
  - GPIO registers are located on the AHB so that the fastest possible I/O timing can be achieved
  - All GPIO registers are byte and half-word addressable
  - Entire port value can be written in one instruction
- Direction control of individual bits
- All I/O default to inputs after reset
- All GPIO pins can be selected to create an edge or level-sensitive GPIO interrupt request
- Support word, half-word and byte access
- One GPIO group interrupt can be triggered by a combination of any pin or pins
- Inputs are sampled using a double flip-flop to avoid meta-stability issues
- As output, the GPIOs can be individually cleared or set

10.4 General description

The GPIO pins can be used in several ways to set pins as inputs or outputs and use the inputs as combinations of level and edge sensitive interrupts.
The GPIOs can be used as external interrupts together with the pin interrupt, see Section 11.
The GPIO port registers configure each GPIO pin as input or output and read the state of each pin if the pin is configured as input or set the state of each pin if the pin is configured as output.

10.5 Register description

Note: In all GPIO registers, bits that are not shown are **reserved**.

GPIO port addresses can be read and written as bytes, halfwords, or words.

Remark: A reset value noted as “ext” in this table and subsequent tables indicates that the data read after reset depends on the state of the pin, which in turn may depend on an external source.

Table 101. GPIO register overview (base address 0x4008 C000 for GPIO A, 0x4008 D000 for GPIO B)

Name	Access	Offset	Description	Reset value	Section
DATA	RO	0x00	GPIO value register	ext	Section 10.5.1
DATAOUT	RW	0x04	GPIO output status register	0x0	Section 10.5.2
OUTENSET	WOO[1]	0x10	GPIO output enable register	0x0	Section 10.5.3
OUTENCLR	W1C[2]	0x14	GPIO output clear register	0x0	Section 10.5.4
INTENSET	WOO	0x20	interrupt enable set register	0x0	Section 10.5.5
INTENCLR	W1C	0x24	interrupt enable clear register	0x0	Section 10.5.6
INTTYPESET	WOO	0x28	interrupt type set register	0x0	Section 10.5.7
INTTYPECLR	W1C	0x2C	interrupt type clear register	0x0	Section 10.5.8
INTPOLSET	WOO	0x30	interrupt polarity set register	0x0	Section 10.5.9
INTPOLCLR	W1C	0x34	interrupt polarity clear register	0x0	Section 10.5.10
INTSTATUS	W1C	0x38	interrupt status register	0x0	Section 10.5.11

[1] WOO: Write-only one. Write value must be 1.

[2] W1C: Write 1 to clear. If the bit in the written value is a 1, the corresponding bit in the field is set to 0. Otherwise, the field bit is not affected.

10.5.1 DATA register

Table 102. GPIO DATA register (DATA, offset: 0x00) bit description

Bit	Symbol	Description	Reset value	Access
31:0	DATA	data value on GPIO port 0: Pin logic level is logic zero 1: Pin logic level is logic one	ext	RO

10.5.2 DATAOUT register

Table 103. DATAOUT - DATAOUT register (offset: 0x04) bit description

Bit	Symbol	Description	Reset value	Access
31:0	DATAOUT	data output register value	0x0	R/W

10.5.3 GPIO port output enable set register

Table 104. OUTENSET - GPIO port Output Enable Set register (offset = 0x10) bit description

Bit	Symbol	Description	Reset value	Access
31:0	OUTENSET	Output enable set Write 1 to configure the pin as GPIO output, no effect when write 0 Read the register can know the GPIO input/output status: 1: Pin is configured as output 0: Pin is configured as input	0x0	WOO

10.5.4 GPIO port output enable clear register

Table 105. OUTENCLR - GPIO Output Enable Clear register (offset = 0x14) bit description

Bit	Symbol	Description	Reset value	Access
31:0	OUTENCLR	output enable clear	0x0	W1C

10.5.5 GPIO port interrupt enable set registers

Table 106. INTENSET - GPIO port Interrupt Enable Set register (offset = 0x20) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTENSET	Interrupt enable set Write 1 to enable GPIO interrupt, no effect when write 0 Read the register can know the interrupt enable/disable status: 1: GPIO interrupt is enabled 0: GPIO interrupt is disabled	0x0	WOO

10.5.6 GPIO port interrupt enable clear registers

Table 107. INTENCLR - GPIO port Interrupt Enable Clear register (offset = 0x24) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTENCLR	interrupt enable clear	0x0	W1C

10.5.7 GPIO interrupt type set registers

Table 108. INTTYPESET - GPIO Interrupt Type Set register (offset = 0x28) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTTYPESET	interrupt type set	NA	WOO

10.5.8 GPIO interrupt type clear registers

Table 109. INTTYPECLR - GPIO Interrupt Type Clear register (offset = 0x2C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTTYPECLR	interrupt type clear	0x0	W1C

10.5.9 GPIO interrupt polarity set registers

Table 110. INTPOLSET - GPIO Interrupt Polarity Set register (offset = 0x30) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTPOLSET	polarity-level, edge IRQ configuration	0x0	WOO

10.5.10 GPIO interrupt polarity clear registers

Table 111. INTPOLCLR - GPIO Interrupt Polarity Clear register (offset = 0x34) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTPOLCLR	interrupt polarity clear	0x0	W1C

10.5.11 GPIO interrupt status registers

Table 112. INTSTATUS - GPIO Interrupt Status register (offset = 0x38) bit description

Bit	Symbol	Description	Reset value	Access
31:0	INTSTATUS	write one to clear interrupt request	0x0	W1C

10.6 Functional description

10.6.1 General-purpose I/O

During and immediately after reset, the alternate functions are not active and the I/O ports are configured in the GPIO digital input mode.

All GPIOs can be configured as inputs or outputs by setting pin output configuration registers OUTENSET and OUTENCLR. If the pin is configured as an output, its values can be set by writing to the data output register DATAOUT. If the pin is configured as an input, the programmed output state re-occurs when the pin is reconfigured as an output.

Software can read the state of all GPIO pins except those selected as analog input or output in the I/O configuration logic. A pin does not have to be selected as GPIO in I/O configuration in order to read its state.

- The input/output state of a single pin can be read OUTENSET or OUTENCLR register
- If the signal direction is input, read DATA register to get the sample at the pin
- If the signal direction is output, read DATAOUT to get the data in the output register

10.6.2 External interrupt/wake-up lines

The AHB GPIO provides programmable interrupt generation features. Three registers control this, and each register has separate set and clear addresses. You can configure each bit of the I/O pins to generate interrupts based on these three registers as given in the following table.

Table 113. Interrupt feature

Interrupt enable[n]	Interrupt polarity[n]	Interrupt type[n]	Interrupt feature
0	-	-	disabled
1	0	0	low level
1	0	1	falling edge
1	1	0	high level
1	1	1	rising edge

After an interrupt is triggered, the corresponding bit in the INTSTATUS register is set. We can clear the interrupt status using an interrupt handler that writes 1 to the corresponding bit of the INTENCLR register, the same address as the INTSTATUS register.

11. Pin INTerrupt (PINT) and pattern match

11.1 Introduction

The pin interrupt generator and pattern match engine are available on all QN908x parts.

11.2 Features

- Pin interrupts
  - Up to four pins can be selected from 32 GPIO pins (from PA01 to PA31) as edge- or level-sensitive interrupt requests. Each request creates a separate interrupt in the NVIC.
  - Edge-sensitive interrupt pins can interrupt on rising or falling edges or both.
  - Level-sensitive interrupt pins can be HIGH- or LOW-active.
- Pattern match engine
  - Up to four pins can be selected from 32 GPIO pins in group A to contribute to a boolean expression. The boolean expression consists of specified levels and/or transitions on various combinations of these pins.
  - Each bit slice minterm (product term) comprising of the specified boolean expression can generate its own, dedicated interrupt request.
  - Pattern match can be used along with software, to create complex state machines based on pin inputs.

11.3 Basic configuration

- Pin interrupts
  - Select up to four external interrupt pins from all digital port pins on group A in the input MUX block (Table 98). The pin selection process is the same for pin interrupts and the pattern match engine. The two features are mutually exclusive.

Enable the clock to the pin interrupt register block with bit CLK_DMA_EN in the CLK_EN register (Table 6).

  - Each selected pin interrupt is assigned to one interrupt in the NVIC.
- Pattern match engine
  - Select up to four external pins from 32 digital port pins on group A in the Input MUX block (Table 98). The pin selection process is the same for pin interrupts and the pattern engine. The two features are mutually exclusive.

Enable the clock to the pin interrupt register block in the CLK_EN register (Table 6).

  - Each bit slice of the pattern match engine is assigned to one interrupt in the NVIC.

11.3.1 Configure pins as pin interrupts or as inputs to the pattern match engine

Follow these steps to configure pins as pin interrupts:

1. Determine the pins that serve as pin interrupts on the QN908x package. See the data sheet for determining the GPIO port pin number associated with the package pin.
2. For each pin interrupt, program the GPIO port pin number from GPIO group A into one of the four PINTSEL registers in the input MUX block.

Remark: The port pin number serves to identify the pin to the PINTSEL register. Any function, including GPIO, can be assigned to this pin via IOCON.

3. Enable each pin interrupt in the NVIC.

Once the pin interrupts or pattern match inputs are configured, the pin interrupt detection levels or the pattern match boolean expression can set up.

See Section 9.6.1 in the input MUX block for the PINTSEL registers.

Remark: The inputs to the Pin interrupt select registers bypass the IOCON function selection. They do not have to be selected as GPIO in IOCON. Make sure that no analog function is selected on pins that are input to the pin interrupts.

11.4 Pin description

The inputs to the pin interrupt and pattern match engine are determined by the pin interrupt select registers in the input MUX; see Section 9.6.1.

11.5 General description

Pins with configurable functions can serve as external interrupts or inputs to the pattern match engine. Up to four pins can be configured using the PINTSEL registers in the input MUX block for these features.

11.5.1 Pin interrupts

- From all 32 GPIO pins in GPIO group A, up to four pins can be selected in the system control block to serve as external interrupt pins (see Table 98). The external interrupt pins are connected to four individual interrupts in the NVIC and are created based on rising or falling edges or on the input level on the pin.

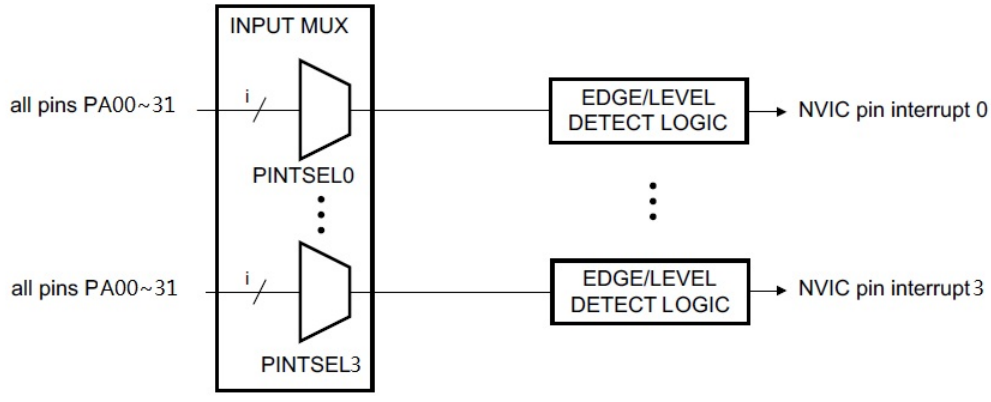


Fig 30. Pin interrupt connections

11.5.2 Pattern match engine

The pattern match feature allows complex boolean expressions to be constructed from the same set of four GPIO pins that were selected for the GPIO pin interrupts. Each term in the boolean expression is implemented as one slice of the pattern match engine. A slice consists of an input selector and a detect logic. The detect logic monitors the selected

input continuously and creates a HIGH output if the input qualifies as detected (input evaluates as true). Several terms can be combined to a minterm and a pin interrupt is asserted when the minterm evaluates as true.

The detect logic of each slice can detect the following events on the selected input:

- Edge with memory (sticky): A rising edge, a falling edge, or a rising or falling edge that is detected at any time after the edge-detection mechanism has been cleared. The input qualifies as detected (the detect logic output remains HIGH) until the pattern match engine detect logic is cleared again.
- Event (non-sticky): Every time an edge (rising or falling) is detected, the detect logic output for this pin goes HIGH. This bit is cleared after one clock cycle, and the detect logic can detect another edge.
- Level: A HIGH or LOW level on the selected input.

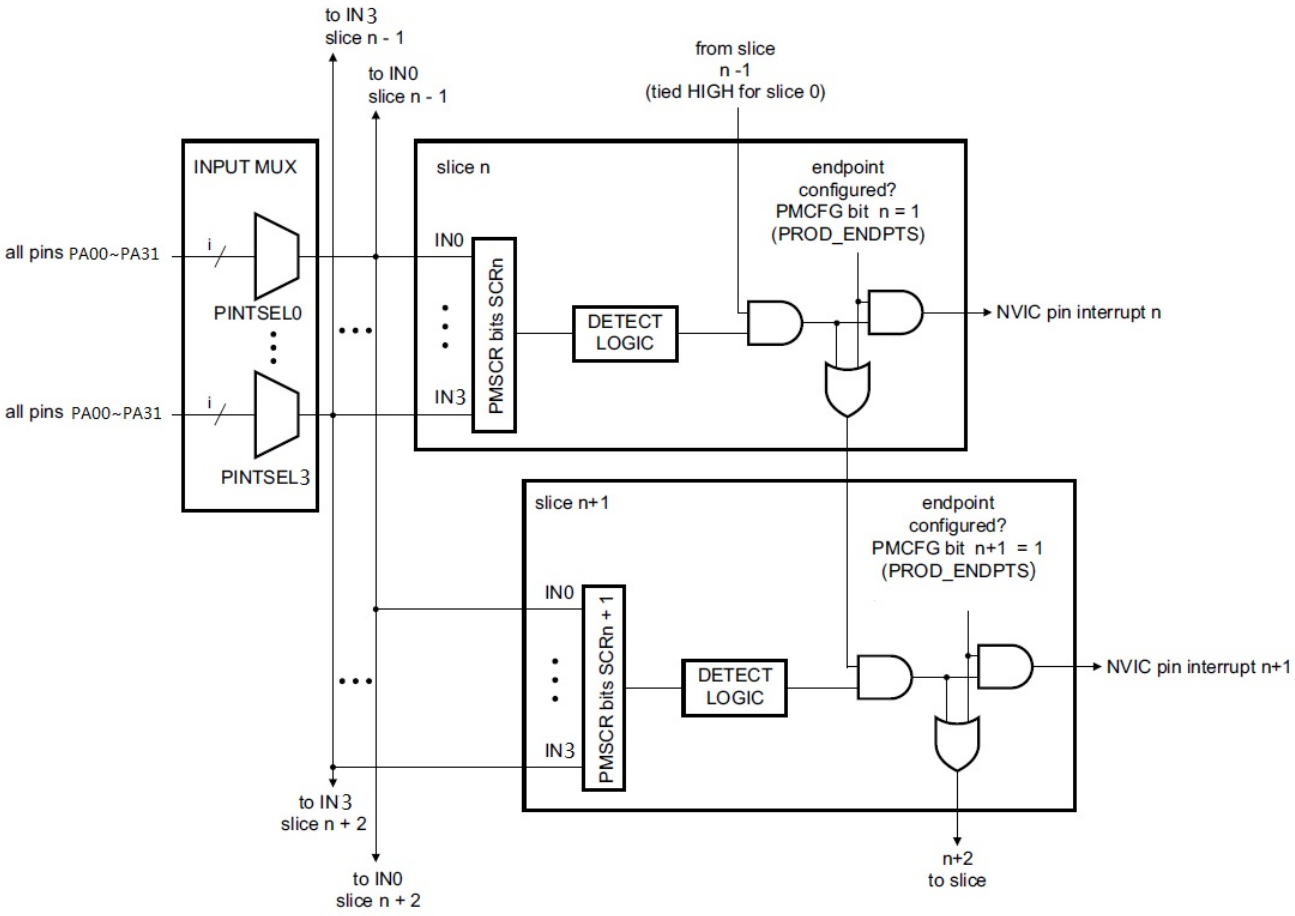
Figure 32 shows the details of the edge detection logic for each slice.

Whenever a rising or falling edge occurs after a qualifying edge event, sticky events can be combined with non-sticky events to create a pin interrupt.

A time window can be created during which rising or falling edges can create a pin interrupt by combining a level detect with an event detect; see Section 11.7.3 for details.

The connections between the pins and the pattern match engine are shown in Figure 31. All pins that are inputs to the pattern match engine are selected in the Syscon block and can be GPIO port pins or other pin function depending on the I/O pin configuration and pin MUX configuration.

Remark: The pattern match feature requires clocks to operate and does not generate an interrupt or wake up the device during reduced power modes below sleep mode.



See Figure 32 for the detect logic block.

Fig 31. Pattern match engine connections

The pattern match logic continuously monitors the four inputs and generates interrupts when any one or more minterms (product terms) of the specified boolean expression is matched. A separate interrupt request is generated for each individual minterm.

The pattern match function utilizes the same four interrupt request lines as the pin interrupts. So, these two features are mutually exclusive as far as interrupt generation is concerned. A control bit (SEL_PMATCH) is provided to select whether interrupt requests are generated in response to the standard pin interrupts or to pattern matches.

Remark: Pattern matching cannot be used to wake the part up from power-down mode. Pin interrupts must be selected for using the GPIO for wake-up.

The pattern match module has four bit slice elements. Each bit slice is programmed to represent one component of one minterm (product term) within the boolean expression. The interrupt request associated with the last bit slice for a particular minterm is asserted whenever that minterm is matched; see bit slice drawing Figure 32.

The pattern match capability can be used to create complex software state machines. Each minterm (and its corresponding individual interrupt) represents a different transition event to a new state. Software can then establish the new set of conditions (that is a new boolean expression) that will cause a transition out of the current state.

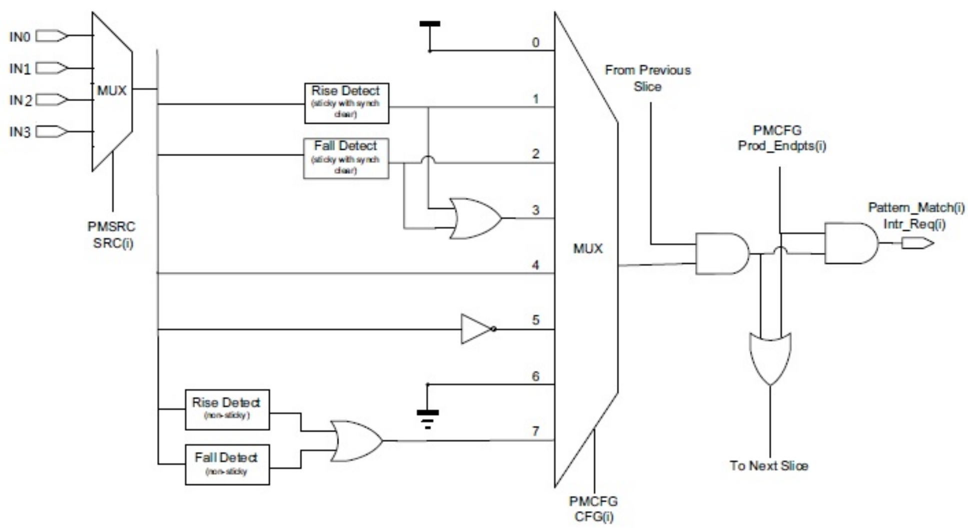


Fig 32. Pattern match bit slice with detect logic

11.5.2.1 Example

Assume the expression: (IN0)~(IN1)(IN3)^ + (IN1) is specified through the registers PMSRC (Table 126) and PMCFG (Table 127). Each term in the boolean expression, (IN0), ~(IN1), (IN3)^, etc., represents one bit slice of the pattern match engine.

- In the first minterm (IN0)~(IN1)(IN3)^, bit slice 0 monitors for a high level on input (IN0), bit slice 1 monitors for a low level on input (IN1) and bit slice 2 monitors for a rising edge on input (IN3). If this combination is detected (all three terms are true), the interrupt associated with bit slice 2 is asserted.
- In the second minterm (IN1), bit slice 3 monitors input (IN1) for a high level. If detected, the interrupt associated with bit slice 3 is asserted.

Related links: Section 11.7.2

11.6 Register description

Table 114. Register overview: Pin interrupts/pattern match engine (base address 0x4000 6000)

Name	Access	Offset	Description	Reset value	Section
ISEL	R/W	0x00	pin interrupt mode register	0x0	Section 11.6.1
IENR	R/W	0x04	pin interrupt level or rising edge interrupt enable register	0x0	Section 11.6.2
SIENR	WO	0x08	pin interrupt level or rising edge interrupt set register	NA	Section 11.6.3
CIENR	WO	0x0C	pin interrupt level (rising edge interrupt) clear register	NA	Section 11.6.4
IENF	R/W	0x10	pin interrupt active level or falling edge interrupt enable register	0x0	Section 11.6.5
SIENF	WO	0x14	pin interrupt active level or falling edge interrupt set register	NA	Section 11.6.6
CIENF	WO	0x18	pin interrupt active level or falling edge interrupt clear register	NA	Section 11.6.7
RISE	R/W	0x1C	pin interrupt rising edge register	0x0	Section 11.6.8
FALL	R/W	0x20	pin interrupt falling edge register	0x0	Section 11.6.9
IST	R/W	0x24	pin interrupt status register	0x0	Section 11.6.10
PMCTRL	R/W	0x28	pattern match interrupt control register	0x0	Section 11.6.11
PMSRC	R/W	0x2C	pattern match interrupt bit slice source register	0x0	Section 11.6.12
PMCFG	R/W	0x30	pattern match interrupt bit slice configuration register	0x0	Section 11.6.13

11.6.1 Pin interrupt mode register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the ISEL register determines whether the interrupt is edge or level sensitive.

Table 115. ISEL - pin interrupt mode register (offset = 0x00) bit description

Bit	Symbol	Description	Reset value	Access
3:0	PMODE	selects the interrupt mode for each pin interrupt; bit n configures the pin interrupt selected in PINTSELn 0: edge sensitive 1: level sensitive	0x0	R/W
31:4	RESERVED	read value is undefined, only zero should be written	-	-

11.6.2 Pin interrupt level or rising edge interrupt enable register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the IENR register enables the interrupt depending on the pin interrupt mode configured in the ISEL register.

- If the pin interrupt mode is edge sensitive (PMODE = 0), the rising edge interrupt is enabled
- If the pin interrupt mode is level sensitive (PMODE = 1), the level interrupt is enabled. The IENF register configures the active level (HIGH or LOW) for this interrupt

Table 116. IENR - pin interrupt level or rising edge interrupt enable register (offset = 0x04) bit description

Bit	Symbol	Description	Reset value	Access
3:0	ENRL	enables the rising edge or level interrupt for each pin interrupt; bit n configures the pin interrupt selected in PINTSELn 0: disable rising edge or level interrupt 1: enable rising edge or level interrupt	0x0	R/W
31:4	RESEER VED	read value is undefined, only zero should be written	-	-

11.6.3 Pin interrupt level or rising edge interrupt set register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the SIENR register sets the corresponding bit in the IENR register depending on the pin interrupt mode configured in the ISEL register.

- If the pin interrupt mode is edge sensitive (PMODE = 0), the rising edge interrupt is set
- If the pin interrupt mode is level sensitive (PMODE = 1), the level interrupt is set

Table 117. SIENR - pin interrupt level or rising edge interrupt set register (offset = 0x08) bit description

Bit	Symbol	Description	Reset value	Access
3:0	SETENRL	Ones written to this address set bits in the IENR, thus enabling interrupts; bit n sets bit n in the IENR register 0: no operation 1: enable rising edge or level interrupt	NA	WO
31:4	RESERVE D	-	-	-

11.6.4 Pin interrupt level or rising edge interrupt clear register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the CIENR register clears the corresponding bit in the IENR register depending on the pin interrupt mode configured in the ISEL register.

- If the pin interrupt mode is edge sensitive (PMODE = 0), the rising edge interrupt is cleared
- If the pin interrupt mode is level sensitive (PMODE = 1), the level interrupt is cleared

Table 118. CIENR - pin interrupt level or rising edge interrupt clear register (offset = 0x0C) bit description

Bit	Symbol	Description	Reset value	Access
3:0	CENRL	ones written to this address clear bits in the IENR, thus disabling the interrupts; bit n clears bit n in the IENR register 0: no operation 1: disable rising edge or level interrupt	NA	WO
31:4	RESERV ED	-	-	-

11.6.5 Pin interrupt active level or falling edge interrupt enable register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the IENF register enables the falling edge interrupt or the configures the level sensitivity depending on the pin interrupt mode configured in the ISEL register.

- If the pin interrupt mode is edge sensitive (PMODE = 0), the falling edge interrupt is enabled
- If the pin interrupt mode is level sensitive (PMODE = 1), the active level of the level interrupt (HIGH or LOW) is configured

Table 119. IENF - pin interrupt active level or falling edge interrupt enable register (offset = 0x10) bit description

Bit	Symbol	Description	Reset value	Access
3:0	ENAF	enables the falling edge or configures the active level interrupt for each pin interrupt; bit n configures the pin interrupt selected in PINTSELn 0: disable falling edge interrupt or set active interrupt level LOW 1: enable falling edge interrupt enabled or set active interrupt level HIGH	0x0	R/W
31:4	RESERVED	-	-	-

11.6.6 Pin interrupt active level or falling edge interrupt set register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the SIENF register sets the corresponding bit in the IENF register depending on the pin interrupt mode configured in the ISEL register.

- If the pin interrupt mode is edge sensitive (PMODE = 0), the falling edge interrupt is set
- If the pin interrupt mode is level sensitive (PMODE = 1), the HIGH-active interrupt is selected

Table 120. SIENF - pin interrupt active level or falling edge interrupt set register (offset = 0x14) bit description

Bit	Symbol	Description	Reset value	Access
3:0	SETENAF	ones written to this address set bits in the IENF, thus enabling interrupts; bit n sets bit n in the IENF register 0: no operation 1: select HIGH-active interrupt or enable falling edge interrupt	NA	WO
31:4	RESERVED	-	-	-

11.6.7 Pin interrupt active level or falling edge interrupt clear register

For each of the four pin interrupts selected in the PINTSELn registers (see Table 98), one bit in the CIENF register sets the corresponding bit in the IENF register depending on the pin interrupt mode configured in the ISEL register.

- If the pin interrupt mode is edge sensitive (PMODE = 0), the falling edge interrupt is cleared
- If the pin interrupt mode is level sensitive (PMODE = 1), the LOW-active interrupt is selected

Table 121. CIENF - pin interrupt active level or falling edge interrupt clear register (offset = 0x18) bit description

Bit	Symbol	Description	Reset value	Access
3:0	CENAF	ones written to this address clears bits in the IENF, thus disabling interrupts; bit n clears bit n in the IENF register 0: no operation 1: LOW-active interrupt selected or falling edge interrupt disabled	NA	WO
31:4	RESERVED	-	-	-

11.6.8 Pin interrupt rising edge register

This register contains ones for pin interrupts selected in the PINTSELn registers (see Table 98) on which a rising edge has been detected. Writing ones to this register clears rising edge detection. Ones in this register assert an interrupt request for pins that are enabled for rising-edge interrupts. All edges are detected for all pins selected by the PINTSELn registers, regardless of whether they are interrupt-enabled.

Table 122. RISE - pin interrupt rising edge register (offset = 0x1C) bit description

Bit	Symbol	Description	Reset value	Access
3:0	RDET	rising edge detect; bit n detects the rising edge of the pin selected in PINTSELn read 0: No rising edge has been detected on this pin since reset or the last time a one was written to this bit write 0: no operation read 1: a rising edge has been detected since reset or the last time a one was written to this bit write 1: clear rising edge detection for this pin	0x0	R/W
31:4	RESERVED	-	-	-

11.6.9 Pin interrupt falling edge register

This register contains ones for pin interrupts selected in the PINTSELn registers (see Table 98) on which a falling edge has been detected. Writing ones to this register clears falling edge detection. Ones in this register assert an interrupt request for pins that are enabled for falling-edge interrupts. All edges are detected for all pins selected by the PINTSELn registers, regardless of whether they are interrupt-enabled.

Table 123. FALL - pin interrupt falling edge register (offset = 0x20) bit description

Bit	Symbol	Description	Reset value	Access
3:0	FDET	falling edge detect; bit n detects the falling edge of the pin selected in PINTSELn read 0: no falling edge has been detected on this pin since reset or the last time a one was written to this bit write 0: no operation read 1: a falling edge has been detected since reset or the last time a one was written to this bit write 1: clear falling edge detection for this pin	0x0	R/W
31:4	RESERVED	-	-	-

11.6.10 Pin interrupt status register

Reading this register returns ones for pin interrupts that are requesting an interrupt. For pins identified as edge-sensitive in the interrupt select register, writing ones to this register clears both rising- and falling-edge detection for the pin. For level-sensitive pins, writing ones invert the corresponding bit in the active level register, thus switching the active level on the pin.

Table 124. IST - pin interrupt status register (offset = 0x24) bit description

Bit	Symbol	Description	Reset value	Access
3:0	PSTAT	pin interrupt status; bit n returns the status, clears the edge interrupt, or inverts the active level of the pin selected in PINTSELn read 0: interrupt is not being requested for this interrupt pin write 0: no operation read 1: interrupt is being requested for this interrupt pin write 1, when pin is in edge-sensitive mode: clear rising- and falling-edge detection for this pin write 1, when pin is in level-sensitive mode: switch the active level for this pin (in the IENF register)	0x0	R/W
31:4	RESERVED	-	-	-

11.6.11 Pattern match interrupt control register

The pattern match control register contains one bit to select pattern-match interrupt generation (as opposed to pin interrupts which share the same interrupt request lines). This register also allows the current state of any pattern matches to be read.

If the pattern match feature is not used (either for interrupt generation), bits SEL_PMATCH of this register should be 0 to conserve power.

Remark: Set up the pattern-match configuration in the PMSRC and PMCFG registers before writing to this register to enable (or re-enable) the pattern-match functionality. This eliminates the possibility of spurious interrupts as the feature is being enabled.

Remark: The pattern match feature requires clocks for operating and does not generate an interrupt or wake up the device during reduced power modes below sleep mode.

Table 125. PMCTRL - pattern match interrupt control register (offset = 0x28) bit description

Bit	Symbol	Value	Description	Reset value
0	SEL_PMATCH		specifies whether the four pin interrupts are controlled by the pin interrupt function or by the pattern match function	0x0
		0	pin interrupt; interrupts are driven in response to the standard pin interrupt function	
		1	pattern match; interrupts are driven in response to pattern matches	
23:1	RESERVED	-	do not write 1s to unused bits	0x0
27:24	PMAT	-	displays the current state of pattern matches; a 1 in any bit of this field indicates that the corresponding product term is matched by the current state of the appropriate inputs	0x0
31:28	RESERVED	-	do not write 1s to unused bits	0x0

11.6.12 Pattern match interrupt bit slice source register

The bit slice source register specifies the input source for each of the four pattern match bit slices.

Each of the possible four inputs is selected in the pin interrupt select registers in the input MUX block; see Section 9.6.1. Input 0 corresponds to the pin selected in the PINTSEL0 register; input 1 corresponds to the pin selected in the PINTSEL1 register, and so forth.

Remark: Writing any value to either the PMCFG register or the PMSRC register, or disabling the pattern match feature (by clearing both the SEL_PMATCH bit in the PMCTRL register to zeros) erases all edge detect history.

Table 126. PMSRC - pattern match bit slice source register (offset = 0x2C) bit description

Bit	Symbol	Value	Description	Reset value
7:0	RESERVED	-	software should not write 1s to unused bits	0x0
10:8	SRC0		selects the input source for bit slice 0	0x0
		0x0	input 0; selects the pin selected in the PINTSEL0 register as the source to bit slice 0	
		0x1	input; selects the pin selected in the PINTSEL1 register as the source to bit slice 0	
		0x2	input 2; selects the pin selected in the PINTSEL2 register as the source to bit slice 0	
		0x3	input 3; selects the pin selected in the PINTSEL3 register as the source to bit slice 0	
13:11	SRC1		selects the input source for bit slice 1	0x0
		0x0	input 0; selects the pin selected in the PINTSEL0 register as the source to bit slice 1	
		0x1	input 1; selects the pin selected in the PINTSEL1 register as the source to bit slice 1	
		0x2	input 2; selects the pin selected in the PINTSEL2 register as the source to bit slice 1	
		0x3	input 3; selects the pin selected in the PINTSEL3 register as the source to bit slice 1	
16:14	SRC2		selects the input source for bit slice 2	0x0
		0x0	input 0; selects the pin selected in the PINTSEL0 register as the source to bit slice 2	
		0x1	input 1; selects the pin selected in the PINTSEL1 register as the source to bit slice 2	
		0x2	input 2; selects the pin selected in the PINTSEL2 register as the source to bit slice 2	
		0x3	input 3; selects the pin selected in the PINTSEL3 register as the source to bit slice 2	
19:17	SRC3		selects the input source for bit slice 3	0x0
		0x0	input 0; selects the pin selected in the PINTSEL0 register as the source to bit slice 3	
		0x1	input 1; selects the pin selected in the PINTSEL1 register as the source to bit slice 3	
		0x2	input 2; selects the pin selected in the PINTSEL2 register as the source to bit slice 3	
		0x3	input 3; selects the pin selected in the PINTSEL3 register as the source to bit slice 3	
31:20	RESERVED	-	software should not write 1s to unused bits	0x0

11.6.13 Pattern match interrupt bit slice configuration register

The bit slice configuration register configures the detect logic and contains bits to select from eight alternative conditions for each bit slice that cause that bit slice to contribute to a pattern match. The four LSBs of this register specify which bit slices are the end points of product terms in the boolean expression (that is, where OR terms are to be inserted in the expression).

Two types of edge detection on each input are possible:

- Sticky: A rising edge, a falling edge, or a rising or falling edge that is detected at any time after the edge detection mechanism is cleared. The input qualifies as detected (the detect logic output remains HIGH) until the pattern match engine detect logic is cleared again.
- Non-sticky: Every time an edge (rising or falling) is detected, the detect logic output for this pin goes HIGH. This bit is cleared after one clock cycle, and the edge detect logic can detect another edge.

Remark: To clear the pattern match engine detect logic, write any value to either the PMCFG register or the PMSRC register, or disable the pattern-match feature (by clearing both the SEL_PMATCH bit in the PMCTRL register to zeros). This will erase all edge detect history.

To select whether a slice marks the final component in a minterm of the boolean expression, write a 1 in the corresponding PROD_ENPTSn bit. Setting a term as the final component has two effects:

1. The interrupt request associated with this bit slice will be asserted whenever a match to that product term is detected
2. The next bit slice will start a new, independent product term in the boolean expression (i.e. an OR will be inserted in the boolean expression following the element controlled by this bit slice)

Table 127. PMCFG - pattern match bit slice configuration register (offset = 0x30) bit description

Bit	Symbol	Value	Description	Reset value
0	PROD_ENDPTS0		determines whether slice 0 is an endpoint	0x0
		0	no effect; slice 0 is not an endpoint	
		1	endpoint; slice 0 is the endpoint of a product term (minterm); if the minterm evaluates as true, pin interrupt 0 in the NVIC is raised	
1	PROD_ENDPTS1		determines whether slice 1 is an endpoint	0x0
		0	no effect; slice 1 is not an endpoint	
		1	endpoint; slice 1 is the endpoint of a product term (minterm); if the minterm evaluates as true, pin interrupt 1 in the NVIC is raised	
2	PROD_ENDPTS2		determines whether slice 2 is an endpoint	0x0
		0	no effect; slice 2 is not an endpoint	
		1	endpoint; slice 2 is the endpoint of a product term (minterm); if the minterm evaluates as true, pin interrupt 2 in the NVIC is raised	

Table 127. PMCFG - pattern match bit slice configuration register (offset = 0x30) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
3	PROD_ENDPTS3		determines whether slice 3 is an endpoint	0x0
		0	no effect; slice 3 is not an endpoint	
		1	endpoint; slice 3 is the endpoint of a product term (minterm); if the minterm evaluates as true, pin interrupt 3 in the NVIC is raised	
7:4	RESERVED	-	do not access these bits	0x0
10:8	CFG0		specifies the match contribution condition for bit slice 0	0x0
		0x0	Constant HIGH. This bit slice always contributes to a product term match.	
		0x1	Sticky rising edge. Match occurs if a rising edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x2	Sticky falling edge. Match occurs if a falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x3	Sticky rising or falling edge. Match occurs if either a rising or falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x4	High level. Match (for this bit slice) occurs when there is a high level on the input specified for this bit slice in the PMSRC register.	
		0x5	Low level. Match occurs when there is a low level on the specified input.	
		0x6	Constant 0. This bit slice never contributes to a match (should be used to disable any unused bit slices).	
		0x7	Event. Non-sticky rising or falling edge. Match occurs on an event - i.e. when either a rising or falling edge is first detected on the specified input (this is a non-sticky version of value 0x3). This bit is cleared after one clock cycle.	
13:11	CFG1		specifies the match contribution condition for bit slice 1	0x0
		0x0	Constant HIGH. This bit slice always contributes to a product term match.	
		0x1	Sticky rising edge. Match occurs if a rising edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x2	Sticky falling edge. Match occurs if a falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x3	Sticky rising or falling edge. Match occurs if either a rising or falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x4	High level. Match (for this bit slice) occurs when there is a high level on the input specified for this bit slice in the PMSRC register.	
		0x5	Low level. Match occurs when there is a low level on the specified input.	
		0x6	Constant 0. This bit slice never contributes to a match (should be used to disable any unused bit slices).	
		0x7	Event. Non-sticky rising or falling edge. Match occurs on an event - i.e. when either a rising or falling edge is first detected on the specified input (this is a non-sticky version of value 0x3). This bit is cleared after one clock cycle.	

Table 127. PMCFG - pattern match bit slice configuration register (offset = 0x30) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
16:14	CFG2		specifies the match contribution condition for bit slice 2	0x0
		0x0	Constant HIGH. This bit slice always contributes to a product term match.	
		0x1	Sticky rising edge. Match occurs if a rising edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x2	Sticky falling edge. Match occurs if a falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x3	Sticky rising or falling edge. Match occurs if either a rising or falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x4	High level. Match (for this bit slice) occurs when there is a high level on the input specified for this bit slice in the PMSRC register.	
		0x5	Low level. Match occurs when there is a low level on the specified input.	
		0x6	Constant 0. This bit slice never contributes to a match (should be used to disable any unused bit slices).	
		0x7	Event. Non-sticky rising or falling edge. Match occurs on an event - i.e. when either a rising or falling edge is first detected on the specified input (this is a non-sticky version of value 0x3). This bit is cleared after one clock cycle.	
19:17	CFG3		specifies the match contribution condition for bit slice 3	0x0
		0x0	Constant HIGH. This bit slice always contributes to a product term match.	
		0x1	Sticky rising edge. Match occurs if a rising edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x2	Sticky falling edge. Match occurs if a falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x3	Sticky rising or falling edge. Match occurs if either a rising or falling edge on the specified input has occurred since the last time the edge detection for this bit slice was cleared. This bit is only cleared when the PMCFG or the PMSRC registers are written to.	
		0x4	High level. Match (for this bit slice) occurs when there is a high level on the input specified for this bit slice in the PMSRC register.	
		0x5	Low level. Match occurs when there is a low level on the specified input.	
		0x6	Constant 0. This bit slice never contributes to a match (should be used to disable any unused bit slices).	
		0x7	Event. Non-sticky rising or falling edge. Match occurs on an event - i.e. when either a rising or falling edge is first detected on the specified input (this is a non-sticky version of value 0x3). This bit is cleared after one clock cycle.	
31:20	RESERVED	-	do not access these bits	

11.7 Functional description

11.7.1 Pin interrupts

In this interrupt facility, up to four pins are identified as interrupt sources by the Pin Interrupt Select (PINTSEL0-3) registers. All registers in the pin interrupt block contain four bits, corresponding to the pins called out by the PINTSEL0-3 registers. The ISEL register defines whether each interrupt pin is edge- or level-sensitive. The RISE and FALL registers detect edges on each interrupt pin, and can be written to clear (and set) edge detection. The IST register indicates whether each interrupt pin is currently requesting an interrupt, and this register can also be written to clear interrupts.

The other pin interrupt registers play different roles for edge-sensitive and level-sensitive pins; see Table 128.

Table 128. Pin interrupt registers for edge- and level-sensitive pins

Name	Edge sensitive function	Level sensitive function
IENR	enables rising edge interrupts.	enables level interrupts
SIENR	write to enable rising-edge interrupts	write to enable level interrupts
CIENR	write to disable rising-edge interrupts	write to disable level interrupts
IENF	enables falling edge interrupts	selects active level
SIENF	write to enable falling edge interrupts	write to select high-active
CIENF	write to disable falling edge interrupts	Write to select low-active

11.7.2 Pattern match engine example

Suppose the desired boolean pattern to be matched is:
(IN1*~IN0) + (IN3re * IN2fe)

with:

IN3re: (sticky) rising edge on input 3

IN2fe: (sticky) falling edge on input 2

Each individual term in the expression shown above is controlled by one bit slice. To specify this expression, program the pattern match bit slice source and configuration register fields as follows:

- PMSRC register (Table 126):
  - since bit slice 2 and 3 will be used to detect a sticky event, a 1 can be written to the SRC2 and SRC3 bits to clear any pre-existing edge detects on bit slice 2 and 3.
  - SRC0: 001 - select input 1 for bit slice 0
  - SRC1: 000 - select input 0 for bit slice 1
  - SRC2: 011 - select input 3 for bit slice 2
  - SRC3: 010 - select input 2 for bit slice 3
- PMCFG register (Table 127):
  - PROD_ENDPTS1: 1
  - PROD_ENDPTS3: 1
  - all other slices are not product term endpoints and their PROD_ENDPTS bits are 0

- CFG0: 100 - high level on the selected input (input 1) for bit slice 0
- CFG1: 101 - low level on the selected input (input 0) for bit slice 1
- CFG2: 001 - (sticky) rising edge on the selected input (input 3) for bit slice 2
- CFG3: 010 - (sticky) falling edge on the selected input (input 2) for bit slice 3

• PMCTRL register (Table 125):
  - Bit 0: Setting this bit will select pattern matches to generate the pin interrupts in place of the normal pin interrupt mechanism
  - For this example, pin interrupt 1 is asserted when a match is detected on the first product term (which in this case, is a high level on input 1 and a low level on input 0)
  - pin interrupt 3 will be asserted in response to a match on the second product term.
  - Bit 27:24: At any given time, if the corresponding product terms are currently matching, bits 1, 3 may be high
  - The remaining bits will always be low

11.7.3 Pattern match engine edge detect examples

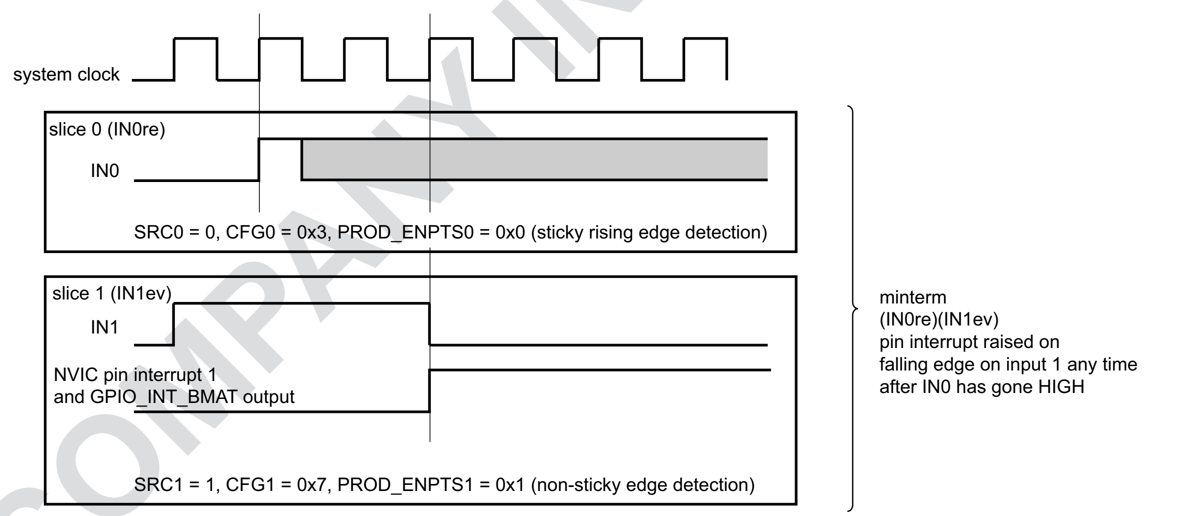


Figure shows pattern match functionality only and accurate timing is not implied. Inputs (INn) are shown synchronized to the system clock for simplicity.

Fig 33. Pattern match engine examples: sticky edge detect

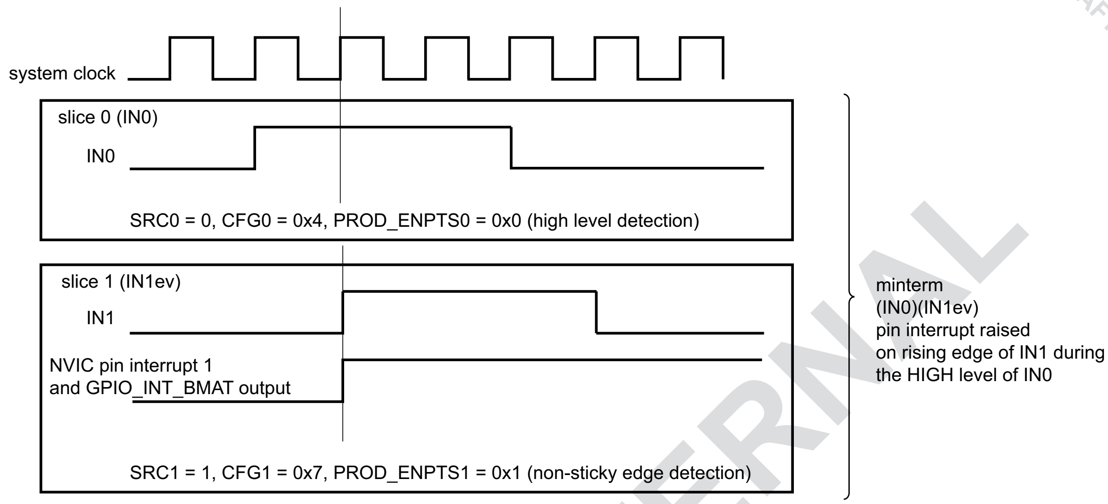


Figure shows pattern match functionality only and accurate timing is not implied. Inputs (INn) are shown synchronized to the system clock for simplicity.

Fig 34. Pattern match engine examples: Windowed non-sticky edge detect evaluates as true

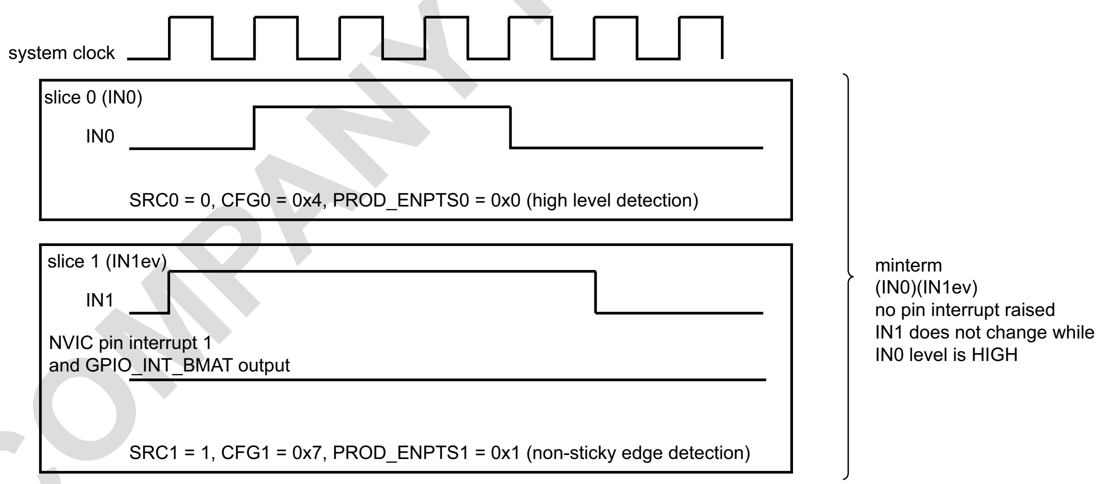


Figure shows pattern match functionality only and accurate timing is not implied. Inputs (INn) are shown synchronized to the system clock for simplicity.

Fig 35. Pattern match engine examples: windowed non-sticky edge detect evaluates as false

12. DMA controller

12.1 Introduction

The DMA controller is available on all QN908x devices.

12.2 Features

- 20 channels, 15 of which are connected to peripheral DMA requests. They come from the Flexcomm interface (USART, SPI, I^{2}C-bus), ADC, and SPIFI interfaces. Four spare channels have no DMA request connected, and can be used for functions such as memory-to-memory transfers. Any unused channel can also be used for other purposes
- DMA operations can be triggered by on- or off-chip events. Each DMA channel can select one trigger input from 20 sources. Trigger sources include FSP interrupts, timer interrupts, pin interrupts, and the SCT DMA request lines
- Priority is user-selectable for each channel (up to four priority levels)
- Continuous priority arbitration
- Address cache with four entries (each entry is a pair of transfer addresses)
- Efficient use of data bus
- Supports single transfers up to 1024 words
- Address increment options allow packing and/or unpacking data

12.3 Basic configuration

Configure the DMA as follows:

- Enable the DMA clock by writing CLK_DMA_EN bit (in CLK_EN register) into 1
- Clear the DMA peripheral reset by writing 1 into CLR_DMA_RST bit (in RST_SW_CLR register)
- The DMA controller provides an interrupt to the NVIC, see Table 49
- Each DMA channel has one DMA request line associated and can also select one of the 20 input triggers, through the input MUX registers DMA_ITRIG_INMUX[0:19]

  Trigger outputs are connected to DMA_INMUX_INMUX[0:3] as inputs to DMA triggers.

For details on the trigger input and output multiplexing; see Section 9.5.2.

12.4 Pin description

The DMA controller has no direct pin connections. However, some DMA triggers can be associated with pin functions; see Section 12.5.1.2.

12.5 General description

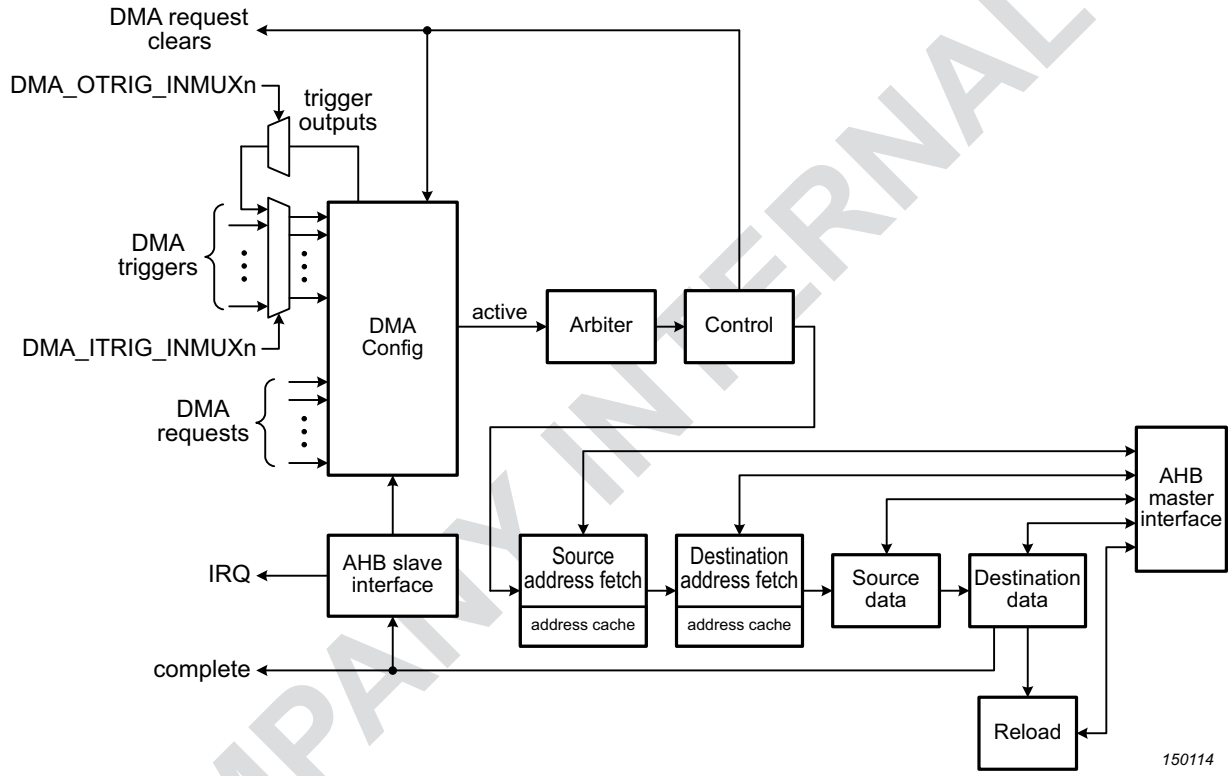


Fig 36. DMA block diagram

12.5.1 DMA requests and triggers

An operation on a DMA channel can be initiated by either a DMA request or a trigger event. DMA requests come from peripherals and specifically indicate when a peripheral either needs input data to be read from it, or that output data may be sent to it. DMA requests are created by the USART, SPI, and I^2C-bus peripherals.

A trigger initiates a DMA operation and can be a signal from an unrelated peripheral. Peripherals that generate triggers are SCT and ADC. In addition, the DMA triggers also creates a trigger output that can trigger DMA transactions on another channel. Triggers can be used to send a character or a string to a USART or other serial output at a fixed time interval or when an event occurs.

A DMA channel using a trigger can respond by moving data from any memory address to any other memory address. This includes fixed peripheral data registers, or incrementing through RAM buffers. The size of data moved by a single trigger event can range from a single transfer to many transfers. A transfer that is started by a trigger can still be paced using the DMA request of the channel. This allows sending a string to a serial peripheral, for instance, without overrunning the transmit buffer of the peripheral.

Each DMA channel has an output that can be used as a trigger input to another channel. The trigger outputs appear in the trigger source list for each channel and can be selected through the DMA_INMUX registers as inputs to other channels.

12.5.1.1 DMA requests

DMA requests are directly connected to the peripherals. Each channel supports one DMA request line and one trigger input which is multiplexed to many possible input sources; see Table 129.

Table 129. DMA requests and trigger MUXes

DMA channel #	Request input	DMA trigger MUX
0	flexcomm 0 RX	DMA_ITRIG_INMUX0
1	flexcomm 0 TX	DMA_ITRIG_INMUX1
2	flexcomm 1 RX/I2C-bus slave [1]	DMA_ITRIG_INMUX2
3	flexcomm 1 TX/I2C-bus master [1]	DMA_ITRIG_INMUX3
4	flexcomm 2 RX/I2C-bus slave [1]	DMA_ITRIG_INMUX4
5	flexcomm 2 TX/I2C-bus master [1]	DMA_ITRIG_INMUX5
6	flexcomm 3 RX	DMA_ITRIG_INMUX6
7	flexcomm 3 TX	DMA_ITRIG_INMUX7
8	SPIFI	DMA_ITRIG_INMUX8
9	prop RX	DMA_ITRIG_INMUX9
10	prop TX	DMA_ITRIG_INMUX10
11	low 256k flash TX	DMA_ITRIG_INMUX11
12	high 256k flash TX	DMA_ITRIG_INMUX12
13	DAC TX	DMA_ITRIG_INMUX13
14	ADC RX	DMA_ITRIG_INMUX14
15	capacitive sense DMA request	DMA_ITRIG_INMUX15
16	no DMA request	DMA_ITRIG_INMUX16
17	no DMA request	DMA_ITRIG_INMUX17
18	no DMA request	DMA_ITRIG_INMUX18
19	no DMA request	DMA_ITRIG_INMUX19

[1] See "DMA with I2C-bus monitor mode" for information about DMA for the I2C-bus monitor function.

DMA with I2C-bus monitor mode: If one of the channels related to the same flexcomm interface is available, the I2C-bus monitor function may be used with DMA.

Table 130. DMA with the I2C-bus monitor function

I2C-bus master DMA	I2C-bus slave DMA	I2C-bus monitor DMA
not enabled	-	if I2C-bus monitor DMA is enabled, it will use the DMA channel for the master function of the same flexcomm interface
enabled	not enabled	if I2C-bus monitor is DMA enabled, it will use the DMA channel for the Slave function of the same flexcomm interface
enabled	enabled	I2C-bus monitor function cannot use DMA

12.5.1.2 Hardware triggers

Each DMA channel can use one trigger, that is independent of the request input for this channel. The trigger input is selected in the DMA_ITRIG_INMUX registers. There are 20 possible internal trigger sources for each DMA channel. In addition, the DMA trigger output can be routed to the trigger input of another channel through trigger input multiplexing; see Table 129 and Section 9.5.2.

Table 131. DMA trigger sources

DMA trigger #	Trigger input	Software trigger
0	FSP DMA request	yes
1	RTC Second interrupt	yes
2	SCT0 DMA request 0	yes
3	SCT0 DMA request 1	yes
4	timer CTIMER0 Match 0 DMA request	yes
5	timer CTIMER0 Match 1 DMA request	yes
6	timer CTIMER1 Match 0 DMA request	yes
7	timer CTIMER1 Match 1 DMA request	yes
8	timer CTIMER2 Match 0 DMA request	yes
9	timer CTIMER2 Match 1 DMA request	yes
10	timer CTIMER3 Match 0 DMA request	yes
11	timer CTIMER3 Match 1 DMA request	yes
12	pin interrupt 0	yes
13	pin interrupt 1	yes
14	pin interrupt 2	yes
15	pin interrupt 3	yes
16	DMA output trigger 0	yes
17	DMA output trigger 1	yes
18	DMA output trigger 2	yes
19	DMA output trigger 3	yes

12.5.1.3 Trigger operation detail

A trigger is needed to start a transfer on a DMA channel. It can be a hardware or software trigger, and can be used in several ways.

If a channel is configured with the SWTRIG bit equal to 0, the channel can be later triggered either by hardware or software. Software triggering is accomplished by writing a 1 to the appropriate bit in the SETTRIG register. Hardware triggering requires setup of the HWTRIGEN, TRIGPOL, TRIGTYPE, and TRIGBURST fields in the CFG register for the related channel. When a channel is initially set up, the SWTRIG bit in the XFERCFG register can be set, causing the transfer to begin immediately.

Once triggered, if the PERIPHREQEN bit in the related CFG register is set, transfer on a channel will be paced by DMA requests. Otherwise, the transfer will proceed at full speed.

The TRIG bit in the CTLSTAT register can be cleared at the end of a transfer, determined by the value CLRTRIG (bit 0) in the XFERCFG register. When a 1 is found in CLRTRIG and the descriptor is exhausted, the trigger is cleared.

12.5.1.4 Trigger output detail

Each channel of the DMA controller provides a trigger output. This allows the possibility of using the trigger outputs as a trigger source to a different channel in order to support complex transfers on selected peripherals. This kind of transfer can, for example, use more than one peripheral DMA request. An example would be to input data to a holding buffer from one peripheral, and then output the data to another peripheral, with both transfers being paced by the appropriate peripheral DMA request. This kind of operation is called “chained operation” or “channel chaining”.

12.5.2 DMA modes

The DMA controller does not have separate operating modes, but there are ways of using the DMA controller that have commonly used terminology in the industry.

Once the DMA controller is setup for operation, using any specific DMA channel requires initializing the registers associated with that channel (see Table 129), and supplying at least the channel descriptor. The channel descriptor is located in memory, typically in on-chip SRAM (see Section 12.6.3). The channel descriptor is shown in Table 132.

Table 132: Channel descriptor

Offset	Description
+ 0x0	reserved
+ 0x4	source data end address
+ 0x8	destination end address
+ 0xC	link to next descriptor

The source and destination end addresses, as well as the link to the next descriptor are just memory addresses that can point to any valid address on the device. The starting address for both source and destination data is the specified end address minus the transfer length (XFERCOUNT * the address increment as defined by SRCINC and DSTINC). The link to the next descriptor is used only if it is a linked transfer.

After the channel has a sufficient number of DMA requests and/or triggers, based on its configuration, the initial descriptor will be exhausted. At that point, if the transfer configuration directs it, the channel descriptor will be reloaded with data from memory pointed to by the “link to next descriptor” entry of the initial channel descriptor. Descriptors loaded in this manner look slightly different than the channel descriptor; see Table 133. The difference is that a new transfer configuration is specified in the reload descriptor instead of being written to the XFERCFG register for that channel.

This process repeats as each descriptor is exhausted and as long as reload is selected in the transfer configuration for each new descriptor.

Table 133: Reload descriptors

Offset	Description
+ 0x0	transfer configuration
+ 0x4	source end address: If the address is incremented, it points to the address of the last entry of the source address range. The address to be used in the transfer is calculated from the end address, data width, and transfer size
+ 0x8	destination end address: If the address is incremented, it points to the address of the last entry of the destination address range. The address to be used in the transfer is calculated from the end address, data width, and transfer size
+ 0xC	link to next descriptor: If used, this address must be aligned to a multiple of 16 bytes, which is the size of a descriptor

12.5.3 Single buffer

It generally applies to memory to memory transfers, and peripheral DMA that occurs only occasionally and is set up for each transfer. For this kind of operation, only the initial channel descriptor shown in Table 134 is needed.

Table 134: Channel descriptor for a single transfer

Offset	Description
+ 0x0	reserved
+ 0x4	source data end address
+ 0x8	destination data end address
+ 0xC	not used

This case is identified by the reload bit in the XFERCFG register. When the DMA channel receives a DMA request or trigger (depending on how it is configured), it performs one or more transfers as configured and stops. Once the channel descriptor is exhausted, additional DMA requests or triggers will have no effect until the channel configuration is updated by software.

12.5.4 Ping-pong

Ping-pong is a special case of linked transfer. It is described separately because it is frequently than complicated versions of linked transfers.

A ping-pong transfer uses two buffers alternately. At any time, one buffer is loaded or unloaded by DMA operations. The other buffer has opposite operation, handled by software, readying the buffer for use when the buffer currently being used by the DMA controller is full or empty. Table 135 shows an example of descriptors for ping-pong from a peripheral to two buffers in memory.

Table 135: Example descriptors for ping-pong operation: peripheral to buffer

Channel descriptor		Descriptor B		Descriptor A	
+ 0x0	not used	+ 0x0	buffer B transfer configuration	+ 0x0	buffer A transfer configuration
+ 0x4	peripheral data end address	+ 0x4	peripheral data end address	+ 0x4	peripheral data end address
+ 0x8	buffer A memory end address	+ 0x8	buffer B memory end address	+ 0x8	buffer A memory end address
+ 0xC	address of descriptor B	+ 0xC	address of descriptor A	+ 0xC	address of descriptor B

In this example, the channel descriptor is used first, with a first buffer in memory called buffer A. The configuration of the DMA channel is set to indicate a reload. Similarly, both descriptor A and descriptor B must also specify reload. When the channel descriptor is exhausted, descriptor B is loaded using the link to descriptor B and a transfer interrupt informs the CPU that buffer A is available.

Descriptor B is then used until it is also exhausted, when descriptor A is loaded using the link to descriptor A contained in descriptor B. Then a transfer interrupt informs the CPU that buffer B is available for processing. The process repeats when descriptor A is exhausted, alternately using each of the two memory buffers.

12.5.5 Linked transfers (linked list)

A linked transfer can use any number of descriptors to define a complicated transfer. It can be configured such that a single transfer, portion of a transfer, one whole descriptor, or an entire structure of links can be initiated by a single DMA request or trigger.

An example of a linked transfer is a ping-pong transfer; see Table 135. The difference is that descriptor B would not link back to descriptor A, but would continue on to another different descriptor. This continues as long as it is desired, and can be ended anywhere, or linked back to any point to repeat a sequence of descriptors. Any descriptor which is not currently used can also be altered by software.

12.5.6 Address alignment for data transfers

Transfers of 16-bit width require an address alignment to a multiple of 2 bytes. Transfers of 32-bit width require an address alignment to a multiple of 4 bytes. Transfers of 8-bit width can be at any address.

12.5.7 Channel chaining

Channel chaining is a feature which allows completion of a DMA transfer on channel x to trigger a DMA transfer on channel y. For example, this feature can be used to have DMA channel x reading n bytes from UART to memory, and then have DMA channel y transferring the received bytes to the CRC engine, without any action required from the ARM core.

To use channel chaining, first configure DMA channels x and y as if no channel chaining would be used. Then,

- For channel x:
  - If channel x is configured to auto reload the descriptor on exhausting of the descriptor (bit RELOAD in the transfer configuration of the descriptor is set), then enable 'clear trigger on descriptor exhausted' by setting bit CLRTRIG in the transfer configuration of the channel in the descriptor
- For channel y:
  - Configure the input trigger input MUX register (DMA_ITRIG_INMUX[0:19]) for channel y to use any of the available DMA trigger MUXes (DMA trigger MUX 0/1)
  - Configure the chosen DMA trigger MUX to select DMA channel x
  - Enable hardware triggering by setting bit HWTRIGEN in the channel configuration register
  - Set the trigger type to edge sensitive by clearing bit TRIGTYPE in the channel configuration register

– Configure the trigger edge to falling edge by clearing bit TRIGPOL in the channel configuration register

Note: After completion of channel x, the descriptor may be reloaded (if configured), but remains untriggered. To configure the chain to auto-trigger itself, setup channels x and y for channel chaining as described earlier. In addition to it,

- A ping-pong configuration for both channel x and y is recommended, so that data currently moved by channel y is not altered by channel x.
- For channel x:
  – Configure the input trigger input MUX register (DMA_ITRIG_INMUX[0:19]) for channel y to use the same DMA trigger MUX as chosen for channel y
  – Enable hardware triggering by setting bit HWTRIGEN in the channel configuration register
  – Set the trigger type to edge sensitive by clearing bit TRIGTYPE in the channel configuration register
  – Configure the trigger edge to falling edge by clearing bit TRIGPOL in the channel configuration register

12.5.8 DMA in reduced power modes

DMA in sleep mode

In sleep mode, the DMA operates and accesses all enabled SRAM blocks, without waking the CPU.

DMA in power-down mode

DMA power is switched off in power-down modes. It stops working in both power-down modes for power saving.

12.6 Register description

The DMA registers are grouped into DMA control, interrupt and status registers and DMA channel registers. DMA transfers are controlled by a set of three registers per channel, the CFG[0:19], CTRLSTAT[0:19], and XFERCFG[0:19] registers.

The reset value reflects the data stored in used bits only. It does not include the content of reserved bits.

Table 136. Register overview: DMA controller (base address 0x4008 2000)

Name	Access	Offset	Description	Reset value	Section
global control and status registers					
CTRL	R/W	0x000	DMA control	0x0	Section 12.6.1
INTSTAT	RO	0x004	interrupt status	0x0	Section 12.6.2
SRAMBASE	R/W	0x008	SRAM address of the channel configuration table	0x0	Section 12.6.3
shared registers					
ENABLESET0	R/W	0x020	channel enable read and Set for all DMA channels	0x0	Section 12.6.4
ENABLECLR0	WO	0x028	channel enable Clear for all DMA channels	NA	Section 12.6.5

Table 136. Register overview: DMA controller (base address 0x4008 2000)

Name	Access	Offset	Description	Reset value	Section
ACTIVE0	RO	0x030	channel active status for all DMA channels	0x0	Section 12.6.6
BUSY0	RO	0x038	channel busy status for all DMA channels	0x0	Section 12.6.7
ERRINT0	R/W	0x040	error interrupt status for all DMA channels	0x0	Section 12.6.8
INTENSET0	R/W	0x048	interrupt enable read and set for all DMA channels	0x0	Section 12.6.9
INTENCLR0	WO	0x050	interrupt enable clear for all DMA channels	NA	Section 12.6.10
INTA0	R/W	0x058	interrupt A status for all DMA channels	0x0	Section 12.6.11
INTB0	R/W	0x060	interrupt B status for all DMA channels	0x0	Section 12.6.12
SETVALID0	WO	0x068	set ValidPending control bits for all DMA channels	NA	Section 12.6.13
SETTRIG0	WO	0x070	set trigger control bits for all DMA channels	NA	Section 12.6.14
ABORT0	WO	0x078	channel abort control for all DMA channels	NA	Section 12.6.15
channel 0 registers					
CFG0	R/W	0x400	configuration register for DMA channel 0	0x0	Section 12.6.16
CTLSTAT0	RO	0x404	control and status register for DMA channel 0	0x0	Section 12.6.17
XFERCFG0	R/W	0x408	transfer configuration register for DMA channel 0	0x0	Section 12.6.18
channel 1 registers					
CFG1	R/W	0x410	configuration register for DMA channel 1	0x0	Section 12.6.16
CTLSTAT1	RO	0x414	control and status register for DMA channel 1	0x0	Section 12.6.17
XFERCFG1	R/W	0x418	transfer configuration register for DMA channel 1	0x0	Section 12.6.18
channel 2 registers					
CFG2	R/W	0x420	configuration register for DMA channel 2	0x0	Section 12.6.16
CTLSTAT2	RO	0x424	control and status register for DMA channel 2	0x0	Section 12.6.17
XFERCFG2	R/W	0x428	transfer configuration register for DMA channel 2	0x0	Section 12.6.18
channel 3 registers					
CFG3	R/W	0x430	configuration register for DMA channel 3	0x0	Section 12.6.16
CTLSTAT3	RO	0x434	control and status register for DMA channel 3	0x0	Section 12.6.17
XFERCFG3	R/W	0x438	transfer configuration register for DMA channel 3	0x0	Section 12.6.18
channel 4 registers					
CFG4	R/W	0x440	configuration register for DMA channel 4	0x0	Section 12.6.16
CTLSTAT4	RO	0x444	control and status register for DMA channel 4	0x0	Section 12.6.17
XFERCFG4	R/W	0x448	transfer configuration register for DMA channel 4	0x0	Section 12.6.18
channel 5 registers					
CFG5	R/W	0x450	configuration register for DMA channel 5	0x0	Section 12.6.16
CTLSTAT5	RO	0x454	control and status register for DMA channel 5	0x0	Section 12.6.17
XFERCFG5	R/W	0x458	transfer configuration register for DMA channel 5	0x0	Section 12.6.18

Table 136. Register overview: DMA controller (base address 0x4008 2000)

Name	Access	Offset	Description	Reset value	Section
channel 6 registers					
CFG6	R/W	0x460	configuration register for DMA channel 6	0x0	Section 12.6.16
CTLSTAT6	RO	0x464	control and status register for DMA channel 6	0x0	Section 12.6.17
XFERCFG6	R/W	0x468	transfer configuration register for DMA channel 6	0x0	Section 12.6.18
channel 7 registers					
CFG7	R/W	0x470	configuration register for DMA channel 7	0x0	Section 12.6.16
CTLSTAT7	RO	0x474	control and status register for DMA channel 7	0x0	Section 12.6.17
XFERCFG7	R/W	0x478	transfer configuration register for DMA channel 7	0x0	Section 12.6.18
channel 8 registers					
CFG8	R/W	0x480	configuration register for DMA channel 8	0x0	Section 12.6.16
CTLSTAT8	RO	0x484	control and status register for DMA channel 8	0x0	Section 12.6.17
XFERCFG8	R/W	0x488	transfer configuration register for DMA channel 8	0x0	Section 12.6.18
channel 9 registers					
CFG9	R/W	0x490	configuration register for DMA channel 9	0x0	Section 12.6.16
CTLSTAT9	RO	0x494	control and status register for DMA channel 9	0x0	Section 12.6.17
XFERCFG9	R/W	0x498	transfer configuration register for DMA channel 9	0x0	Section 12.6.18
channel 10 registers					
CFG10	R/W	0x4A0	configuration register for DMA channel 10	0x0	Section 12.6.16
CTLSTAT10	RO	0x4A4	control and status register for DMA channel 10	0x0	Section 12.6.17
XFERCFG10	R/W	0x4A8	transfer configuration register for DMA channel 10	0x0	Section 12.6.18
channel 11 registers					
CFG11	R/W	0x4B0	configuration register for DMA channel 11	0x0	Section 12.6.16
CTLSTAT11	RO	0x4B4	control and status register for DMA channel 11	0x0	Section 12.6.17
XFERCFG11	R/W	0x4B8	transfer configuration register for DMA channel 11	0x0	Section 12.6.18
channel 12 registers					
CFG12	R/W	0x4C0	configuration register for DMA channel 12	0x0	Section 12.6.16
CTLSTAT12	RO	0x4C4	control and status register for DMA channel 12	0x0	Section 12.6.17
XFERCFG12	R/W	0x4C8	transfer configuration register for DMA channel 12	0x0	Section 12.6.18
channel 13 registers					
CFG13	R/W	0x4D0	configuration register for DMA channel 13	0x0	Section 12.6.16
CTLSTAT13	RO	0x4D4	control and status register for DMA channel 13	0x0	Section 12.6.17
XFERCFG13	R/W	0x4D8	transfer configuration register for DMA channel 13	0x0	Section 12.6.18
channel 14 registers					
CFG14	R/W	0x4E0	configuration register for DMA channel 14	0x0	Section 12.6.16
CTLSTAT14	RO	0x4E4	control and status register for DMA channel 14	0x0	Section 12.6.17

Table 136. Register overview: DMA controller (base address 0x4008 2000)

Name	Access	Offset	Description	Reset value	Section
XFERCFG14	R/W	0x4E8	transfer configuration register for DMA channel 14	0x0	Section 12.6.18
channel 15 registers					
CFG15	R/W	0x4F0	configuration register for DMA channel 15	0x0	Section 12.6.16
CTLSTAT15	RO	0x4F4	control and status register for DMA channel 15	0x0	Section 12.6.17
XFERCFG15	R/W	0x4F8	transfer configuration register for DMA channel 15	0x0	Section 12.6.18
channel 16 registers					
CFG16	R/W	0x500	configuration register for DMA channel 16	0x0	Section 12.6.16
CTLSTAT16	RO	0x504	control and status register for DMA channel 16	0x0	Section 12.6.17
XFERCFG16	R/W	0x508	transfer configuration register for DMA channel 16	0x0	Section 12.6.18
channel 17 registers					
CFG17	R/W	0x510	configuration register for DMA channel 17	0x0	Section 12.6.16
CTLSTAT17	RO	0x514	control and status register for DMA channel 17	0x0	Section 12.6.17
XFERCFG17	R/W	0x518	transfer configuration register for DMA channel 17	0x0	Section 12.6.18
channel 18 registers					
CFG18	R/W	0x520	configuration register for DMA channel 18	0x0	Section 12.6.16
CTLSTAT18	RO	0x524	control and status register for DMA channel 18	0x0	Section 12.6.17
XFERCFG18	R/W	0x528	transfer configuration register for DMA channel 18	0x0	Section 12.6.18
channel 19 registers					
CFG19	R/W	0x530	configuration register for DMA channel 19	0x0	Section 12.6.16
CTLSTAT19	RO	0x534	control and status register for DMA channel 19	0x0	Section 12.6.17
XFERCFG19	R/W	0x538	transfer configuration register for DMA channel 19	0x0	Section 12.6.18

12.6.1 Control register

The CTRL register contains global control bit for a enabling the DMA controller.

Table 137. CTRL - control register (offset = 0x000) bit description

Bit	Symbol	Value	Description	Reset value
0	ENABLE		DMA controller master enable	0x0
		0	Disabled. The DMA controller is disabled. It clears any triggers that were asserted at the point when disabled, but does not prevent re-triggering when the DMA controller is re-enabled.	
		1	Enabled. The DMA controller is enabled.	
31:1	RESERVED	-	Read value is undefined, only zero should be written.	NA

12.6.2 Interrupt STATus register (INSTAT)

The read only INTSTAT register provides an overview of DMA status. It allows quick determination of whether any enabled interrupt is pending. Details of channels involved are found in the interrupt type specific registers.

Table 138. INTSTAT - Interrupt Status register (offset = 0x004) bit description

Bit	Symbol	Value	Description	Reset value
0	RESERVED	-	read value is undefined, only zero should be written	NA
1	ACTIVEINT		summarizes whether any enabled interrupts (other than error interrupts) are pending	0x0
		0	not pending; no enabled interrupts are pending	
		1	pending; at least one enabled interrupt is pending	
2	ACTIVEERRINT		summarizes whether any error interrupts are pending	0x0
		0	not pending; no error interrupts are pending	
		1	pending; atleast one error interrupt is pending	
31:3	RESERVED	-	read value is undefined, only zero should be written	NA

12.6.3 SRAM base address register (SRAMBASE)

The SRAMBASE register must be configured with an address (preferably in on-chip SRAM) where DMA descriptors will be stored. Software must set up the descriptors for those DMA channels that will be used in the application.

Table 139. SRAMBASE - SRAM base address register (offset = 0x008) bit description

Bit	Symbol	Description	Reset value
8:0	RESERVED	read value is undefined, only zero should be written	-
31:9	OFFSET	address bits 31:9 of the beginning of the DMA descriptor table; for 18 channels, the table must begin on a 512 byte boundary	0x0

Each DMA channel has an entry for the channel descriptor in the SRAM table. The values for each channel start at the address offsets found in Table 140. Only the descriptors for channels defined at extraction are used. The contents of each channel descriptor are described in Table 132.

Table 140. Channel descriptor map

Table offset	Descriptor
0x000	channel descriptor for DMA channel 0
0x010	channel descriptor for DMA channel 1
0x020	channel descriptor for DMA channel 2
0x030	channel descriptor for DMA channel 3
0x040	channel descriptor for DMA channel 4
0x050	channel descriptor for DMA channel 5
0x060	channel descriptor for DMA channel 6
0x070	channel descriptor for DMA channel 7
0x080	channel descriptor for DMA channel 8
0x090	channel descriptor for DMA channel 9
0x0A0	channel descriptor for DMA channel 10
0x0B0	channel descriptor for DMA channel 11
0x0C0	channel descriptor for DMA channel 12
0x0D0	channel descriptor for DMA channel 13
0x0E0	channel descriptor for DMA channel 14

Table 140. Channel descriptor map

Table offset	Descriptor
0x0F0	channel descriptor for DMA channel 15
0x100	channel descriptor for DMA channel 16
0x110	channel descriptor for DMA channel 17
0x120	channel descriptor for DMA channel 18
0x130	channel descriptor for DMA channel 19

12.6.4 Enable read and set registers (ENABLESET0)

The ENABLESET0 register determines whether each DMA channel is enabled or disabled. Disabling a DMA channel does not reset the channel in any way. A channel can be paused and restarted by clearing, then setting the enable bit for that channel.

Reading ENABLESET0 provides the current state of all of the DMA channels represented by that register. Writing a 1 to a bit position in ENABLESET0 that corresponds to an implemented DMA channel sets the bit, enabling the related DMA channel. Writing a 0 to any bit has no effect. Enables are cleared by writing to ENABLECLR0.

Table 141. ENABLESET0 - Enable read and Set register 0 (offset = 0x020) bit description

Bit	Symbol	Description	Reset value
31:0	ENA	Enable for DMA channels. Bit n enables or disables DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: disabled 1: enabled	0x0

12.6.5 Enable clear register (ENABLECLR0)

The ENABLECLR0 register is used to clear the enable of one or more DMA channels. This register is write-only.

Table 142. ENABLECLR0 - Enable clear register 0 (offset = 0x028) bit description

Bit	Symbol	Description	Reset value
31:0	CLR	Writing ones to this register clears the corresponding bits in ENABLESET0. Bit n clears the channel enable bit n. The number of bits: number of DMA channels in this device. Other bits are reserved	NA

12.6.6 Active status register (ACTIVE0)

The ACTIVE0 register indicates which DMA channels are active at the point when the read occurs. The register is read-only.

A DMA channel is considered active when a DMA operation has been started but not yet fully completed. The active status will persist from a DMA operation being started, until the pipeline is empty after end of the last descriptor (when there is no reload). An active channel is aborted by software by setting the appropriate bit in one of the abort register (see Section 12.6.15).

Table 143. ACTIVE0 - Active status register 0 (offset = 0x030) bit description

Bit	Symbol	Description	Reset value
31:0	ACT	Active flag for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved. 0: not active 1: active	0x0

12.6.7 Busy status register (BUSY0)

The BUSY0 register indicates which DMA channels are busy at the point when the read occurs. This register is read-only.

A DMA channel is considered busy when there is any operation related to that channel in the internal pipeline of the DMA controller. This information can be used after a DMA channel is disabled by software (but still active), allowing confirmation that there are no remaining operations in progress for that channel.

Table 144. BUSY0 - Busy status register 0 (offset = 0x038) bit description

Bit	Symbol	Description	Reset value
31:0	BSY	Busy flag for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: not busy 1: busy	0x0

12.6.8 Error interrupt register (ERRINT0)

The ERRINT0 register contains flags for each error interrupt of the DMA channel. Any pending interrupt flag in the register will be reflected on the DMA interrupt output.

Reading the register provides the current state of all DMA channel error interrupts. Writing a 1 to a bit position in ERRINT0 that corresponds to an implemented DMA channel clears the bit, removing the interrupt for the related DMA channel. Writing a 0 to any bit has no effect.

Table 145. ERRINT0 - Error Interrupt register 0 (offset = 0x040) bit description

Bit	Symbol	Description	Reset value
31:0	ERR	Error Interrupt flag for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: error interrupt is not active 1: error interrupt is active	0x0

12.6.9 Interrupt enable read and set register (INTENSET0)

The INTENSET0 register controls whether the individual interrupt for DMA channels contribute to the DMA interrupt output.

Reading the register provides the current state of all DMA channel interrupt enables. Writing a 1 to a bit position in INTENSET0 that corresponds to an implemented DMA channel sets the bit, enabling the interrupt for the related DMA channel. Writing a 0 to any bit has no effect. Interrupt enables are cleared by writing to INTENCLR0.

Table 146. INTENSET0 - Interrupt enable read and Set register 0 (offset = 0x048) bit description

Bit	Symbol	Description	Reset value
31: 0	INTEN	Interrupt enable read and set for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: interrupt for DMA channel is disabled 1: interrupt for DMA channel is enabled	0x0

12.6.10 Interrupt enable clear register (INTENCLR0)

The INTENCLR0 register is used to clear interrupt enable bits in INTENSET0. The register is write-only.

Table 147. INTENCLR0 - Interrupt enable clear register 0 (offset = 0x050) bit description

Bit	Symbol	Description	Reset value
31:0	CLR	Writing ones to this register clears corresponding bits in the INTENSET0. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved	NA

12.6.11 Interrupt A register (INTA0)

The INTA0 register contains the interrupt A status for each DMA channel. The status is set when the SETINTA bit is 1 in the transfer configuration for a channel, when the descriptor becomes exhausted. Writing a 1 to a bit in this register clears the related INTA flag. Writing 0 has no effect. Any interrupt pending status in the registers will be reflected on the DMA interrupt output if it is enabled in the related INTENSET register.

Table 148. INTA0 - Interrupt A register 0 (offset = 0x058) bit description

Bit	Symbol	Description	Reset value
31:0	IA	Interrupt A status for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: DMA channel interrupt A is not active 1: DMA channel interrupt A is active	0x0

12.6.12 Interrupt B register (INTB0)

The INTB0 register contains the interrupt B status for each DMA channel. The status is set when the SETINTB bit is 1 in the transfer configuration for a channel, when the descriptor becomes exhausted. Writing a 1 to a bit in the register clears the related INTB flag. Writing 0 has no effect. Any interrupt pending status in this register will be reflected on the DMA interrupt output if it is enabled in the INTENSET register.

Table 149. INTB0 - Interrupt B register 0 (offset = 0x060) bit description

Bit	Symbol	Description	Reset value
31:0	IB	Interrupt B status for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: DMA channel interrupt B is not active 1: DMA channel interrupt B is active	0x0

12.6.13 Set valid register (SETVALID0)

The SETVALID0 register allows setting the valid bit in the CTRLSTAT register for one or more DMA channels. See Section 12.6.17 for a description of the VALID bit. This register is write-only.

The CFGVALID and SV (set valid) bits allow direct DMA block timing control by software. Each channel descriptor in a sequence of descriptors can be validated either by the setting of CFGVALID bit or by setting the SETVALID flag of the channel. Normally, the CFGVALID bit is set. It indicates to the DMA that the channel descriptor is active and can be executed. The DMA continues sequencing through descriptor blocks whose CFGVALID bit are set without further software intervention. Leaving a CFGVALID bit set to 0 allows the DMA sequence to pause at the descriptor until software triggers the continuation. If, during DMA transmission, a channel descriptor is found with CFGVALID set to 0, the DMA checks for a previously buffered SETVALID0 setting for the channel. If found, the DMA will set the descriptor valid, clear the SV setting, and resume processing the descriptor. Otherwise, the DMA pauses until the channels SETVALID0 bit is set.

Table 150. SETVALID0 - Set valid 0 register (offset = 0x068) bit description

Bit	Symbol	Description	Reset value
31:0	SV	SETVALID control for DMA channel n. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: no effect 1: sets the VALIDPENDING control bit for DMA channel n	NA

12.6.14 Set trigger register

The SETTRIG0 register allows setting the TRIG bit in the CTRLSTAT register for one or more DMA channel. See Section 12.6.17 for a description of the TRIG bit, and Section 12.5.1 for a general description of triggering. This register is write-only.

Table 151. SETTRIG0 - Set trigger 0 register (offset = 0x070) bit description

Bit	Symbol	Description	Reset value
31:0	TRIG	Set trigger control bit for DMA channel 0. Bit n corresponds to DMA channel n. The number of bits: number of DMA channels in this device. Other bits are reserved 0: no effect 1: sets the TRIG bit for DMA channel n	NA

12.6.15 Abort register (ABORT0)

The ABORT0 register allows aborting operation of a DMA channel if required. To abort a selected channel, the channel should first be disabled by clearing the corresponding enable bit by writing a 1 to the proper bit ENABLECLR. Then wait until the channel is no longer busy by checking the corresponding bit in BUSY. Finally, write a 1 to the proper bit of ABORT. It prevents the channel from restarting an incomplete operation when it is enabled again. This register is write-only.

Table 152. ABORT0 - Abort 0 register (offset = 0x078) bit description

Bit	Symbol	Description	Reset value
31:0	ABORTCTRL	Abort control for DMA channel 0. Bit n corresponds to DMA channel n 0: no effect 1: aborts DMA operations on channel n	NA

12.6.16 Channel configuration registers (CFGn)

The CFGn register contains various configuration options for DMA channel n.

See Table 154 for a summary of trigger options.

Table 153. Channel configuration registers bit description

Bit	Symbol	Value	Description	Reset value
0	PERIPHREQEN		Peripheral request Enable. If a DMA channel is used to perform a memory-to-memory move, any peripheral DMA request associated with that channel can be disabled to prevent any interaction between the peripheral and the DMA controller	0x0
		0	Disabled. Peripheral DMA requests are disabled	
		1	Enabled. Peripheral DMA requests are enabled	
1	HWTRIGEN		Hardware Triggering Enable for this channel	0x0
		0	Disabled. Hardware triggering is not used	
		1	Enabled. Use hardware triggering	
3:2	RESERVED	-	Read value is undefined, only zero should be written	NA
4	TRIGPOL		Trigger Polarity. Selects the polarity of a hardware trigger for this channel.	0x0
		0	Active low - falling edge. Hardware trigger is active low or falling edge triggered, based on TRIGTYPE	
		1	Active high - rising edge. Hardware trigger is active high or rising edge triggered, based on TRIGTYPE	
5	TRIGTYPE		Trigger Type. Selects hardware trigger as edge triggered or level triggered	0x0
		0	Edge. Hardware trigger is edge triggered. Transfers will be initiated and completed, as specified for a single trigger	
		1	Level. Hardware trigger is level triggered. Note that when level triggering without burst (BURSTPOWER: 0) is selected, only hardware triggers should be used on that channel Transfers continue as long as the trigger level is asserted. Once the trigger is deasserted, the transfer will be paused until the trigger is, again, asserted. However, the transfer will not be paused until any remaining transfers within the current BURSTPOWER length are completed	
6	TRIGBURST		Trigger burst. Selects whether hardware triggers cause a single or burst transfer	0x0
		0	Single transfer. Hardware trigger causes a single transfer	
		1	Burst transfer. When the trigger for this channel is set to edge triggered, a hardware trigger causes a burst transfer, as defined by BURSTPOWER When the trigger for this channel is set to level triggered, a hardware trigger causes transfers to continue as long as the trigger is asserted, unless the transfer is complete	
7	RESERVED	-	Read value is undefined, only zero should be written	NA

Table 153. Channel configuration registers bit description

Bit	Symbol	Value	Description	Reset value
11:8	BURSTPOWER		Burst Power is used in two ways. It always selects the address wrap size when SRCBURSTWRAP and/or DSTBURSTWRAP modes are selected (see descriptions elsewhere in this register) When the TRIGBURST field elsewhere in this register: 1, Burst Power selects how many transfers are performed for each DMA trigger. This can be used, for example, with peripherals that contain a FIFO that can initiate a DMA operation when the FIFO reaches a certain level 0000: Burst size: 1 (2^0) 0001: Burst size: 2 (2^1) 0010: Burst size: 4 (2^2) ... 1010: Burst size: 1024 (2^{10}). This corresponds to the maximum supported transfer count others: not supported The total transfer length as defined in the XFERCOUNT bits in the XFERCFG register must be an even multiple of the burst size. Note that the total number of bytes transferred is: (XFERCOUNT + 1) x data width (as defined by the WIDTH field)	0x0
13:12	RESERVED	-	Read value is undefined, only zero should be written	NA
14	SRCBURSTWRAP		Source Burst Wrap. When enabled, the source data address for the DMA is “wrapped”, meaning that the source address range for each burst will be the same. As an example, this could be used to read several sequential registers from a peripheral for each DMA burst, reading the same registers again for each burst	0x0
		0	Disabled. Source burst wrapping is not enabled for this DMA channel	
		1	Enabled. Source burst wrapping is enabled for this DMA channel	
15	DSTBURSTWRAP		Destination burst wrap. When enabled, the destination data address for the DMA is “wrapped”, meaning that the destination address range for each burst will be the same. As an example, this could be used to write several sequential registers to a peripheral for each DMA burst, writing the same registers again for each burst	0x0
		0	Disabled. Destination burst wrapping is not enabled for this DMA channel	
		1	Enabled. Destination burst wrapping is enabled for this DMA channel	
18:16	CHPRIORITY		Priority of this channel when multiple DMA requests are pending four priority levels are supported: 0x0: highest priority 0x7: lowest priority	0x0
31:19	RESERVED	-	Read value is undefined, only zero should be written	NA

Table 154. Trigger setting summary

TrigBurst	TrigType	TrigPol	Description
0	0	0	Hardware DMA trigger is falling edge sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap
0	0	1	Hardware DMA trigger is rising edge sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap
0	1	0	Hardware DMA trigger is low level sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap

Table 154. Trigger setting summary

TrigBurst	TrigType	TrigPol	Description
0	1	1	Hardware DMA trigger is high level sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap
1	0	0	Hardware DMA trigger is falling edge sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap, and also determines how much data is transferred for each trigger
1	0	1	Hardware DMA trigger is rising edge sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap, and also determines how much data is transferred for each trigger
1	1	0	Hardware DMA trigger is low level sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap, and also determines how much data is transferred for each trigger
1	1	1	Hardware DMA trigger is high level sensitive. The BURSTPOWER field controls address wrapping if enabled via SrcBurstWrap and/or DstBurstWrap, and also determines how much data is transferred for each trigger

12.6.17 Channel control and status registers (CTLSTATn)

The CTLSTATn register provides status flags specific to DMA channel n. These registers are read-only.

Table 155. Channel control and Status registers bit description

Bit	Symbol	Value	Description	Reset value
0	VALIDPENDING		Valid pending flag for this channel. This bit is set when a 1 is written to the corresponding bit in the related SETVALID register when CFGVALID: 1 for the same channel	0x0
		0	No effect. No effect on DMA operation	
		1	Valid pending	
1	RESERVED	-	Read value is undefined, only zero should be written	NA
2	TRIG		Trigger flag. Indicates that the trigger for this channel is currently set. This bit is cleared at the end of an entire transfer or upon reload when CLRTRIG: 1	0x0
		0	Not triggered. The trigger for this DMA channel is not set. DMA operations will not be carried out	
		1	Triggered. The trigger for this DMA channel is set. DMA operations will be carried out	
31:3	RESERVED	-	Read value is undefined, only zero should be written	NA

12.6.18 Channel transfer configuration registers (XFERCFGn)

The XFERCFGn register contains transfer related configuration information for DMA channel n. Using the reload bit, this register can optionally be automatically reloaded when the current settings are exhausted (the full transfer count has been completed), allowing linked transfers with more than one descriptor to be performed.

See Section 12.5.1.3 for details on trigger operation.

Table 156. Channel transfer configuration registers bit description

Bit	Symbol	Value	Description	Reset value
0	CFGVALID		Configuration valid flag. This bit indicates whether the current channel descriptor is valid and can potentially be acted upon, if all other activation criteria are fulfilled.	0x0
		0	Not valid. The channel descriptor is not considered valid until validated by an associated SETVALID0 setting.	
		1	Valid. The current channel descriptor is considered valid.	
1	RELOAD		Indicates whether the control structure of the channel will be reloaded when the current descriptor is exhausted. Reloading allows ping-pong and linked transfers.	0x0
		0	Disabled. Do not reload the channels' control structure when the current descriptor is exhausted	
		1	Enabled. Reload the control structure of the channel when the current descriptor is exhausted	
2	SWTRIG		Software trigger	0x0
		0	Not set. When written by software, the trigger for this channel is not set. A new trigger, as defined by the HWTRIGEN, TRIGPOL, and TRIGTYPE will be needed to start the channel.	
		1	Set. When written by software, the trigger for this channel is set immediately. This feature should not be used with level triggering when TRIGBURST: 0.	
3	CLRTRIG		Clear trigger	0x0
		0	Not cleared. The trigger is not cleared when this descriptor is exhausted. If there is a reload, the next descriptor will be started.	
		1	Cleared. The trigger is cleared when this descriptor is exhausted	
4	SETINTA		Set Interrupt flag A for this channel. There is no hardware distinction between interrupt A and B. They can be used by software to assist with more complex descriptor usage. By convention, interrupt A may be used when only one interrupt flag is needed.	0x0
		0	No effect	
		1	Set. The INTA flag for this channel will be set when the current descriptor is exhausted	
5	SETINTB		Set Interrupt flag B for this channel. There is no hardware distinction between interrupt A and B. They can be used by software to assist with more complex descriptor usage. By convention, interrupt A may be used when only one interrupt flag is needed.	0x0
		0	No effect	
		1	Set. The INTB flag for this channel will be set when the current descriptor is exhausted	
7:6	-	-	Reserved. Read value is undefined, only zero should be written	NA
9:8	WIDTH		Transfer width used for this DMA channel	0x0
		0x0	8-bit. 8-bit transfers are performed (8-bit source reads and destination writes)	
		0x1	16-bit. 6-bit transfers are performed (16-bit source reads and destination writes)	
		0x2	32-bit. 32-bit transfers are performed (32-bit source reads and destination writes)	
		0x3	Reserved. Reserved setting, do not use	
11:10	-	-	Reserved. Read value is undefined, only zero should be written	NA

Table 156. Channel transfer configuration registers bit description

Bit	Symbol	Value	Description	Reset value
13:12	SRCINC		Determines whether the source address is incremented for each DMA transfer	0x0
		0x0	No increment. The source address is not incremented for each transfer. This is the usual case when the source is a peripheral device	
		0x1	1 × width. The source address is incremented by the amount specified by width for each transfer. This is the usual case when the source is memory	
		0x2	2 × width. The source address is incremented by 2 times the amount specified by width for each transfer	
		0x3	4 × width. The source address is incremented by 4 times the amount specified by width for each transfer	
15:14	DSTINC		Determines whether the destination address is incremented for each DMA transfer	0x0
		0x0	No increment. The destination address is not incremented for each transfer. This is the usual case when the destination is a peripheral device	
		0x1	1 × width. The destination address is incremented by the amount specified by width for each transfer. This is the usual case when the destination is memory	
		0x2	2 × width. The destination address is incremented by 2 times the amount specified by width for each transfer	
		0x3	4 × width. The destination address is incremented by 4 times the amount specified by width for each transfer	
25:16	XFERCOUNT		Total number of transfers to be performed, minus 1 encoded **Remark:** The DMA controller uses this bit field during transfer to count down. Hence, it cannot be used by software to read back the size of the transfer, for instance, in an interrupt handler 0x0: a total of 1 transfer will be performed 0x1: a total of 2 transfers will be performed ... 0x3FF: a total of 1,024 transfers will be performed	0x0
31:26	RESERVED	-	Read value is undefined, only zero should be written	NA

13. SCTimer/PWM (SCT)

13.1 How to read this chapter

The SCTimer/PWM is available on all QN908x devices.

Remark: For a detailed description of SCTimer/PWM applications and code examples, see Ref. 3 “AN11538”.

13.2 Features

- The SCTimer/PWM supports:
  - Eight inputs
  - Ten outputs
  - Ten match/capture registers
  - Ten events
  - Ten states
- Counter/timer features:
  - Each SCTimer is configurable as two 16-bit counters or one 32-bit counter
  - Counters clocked by system clock or selected input
  - Configurable as up counters or up-down counters
  - Configurable number of match and capture registers. Up to 10 match and capture registers total
  - Upon match and/or an input or output transition create the following events: interrupt; stop, limit, halt the timer or change counting direction; toggle outputs; change the state
  - Counter value can be loaded into capture register triggered by a match or input/output toggle
- PWM features:
  - Counters can be used in conjunction with match registers to toggle outputs and create time-proportioned PWM signals. PWM waveforms can change based on the current state
  - Up to 8 single-edge or 4 dual-edge PWM outputs with independent duty cycle and common PWM cycle length
- Event creation features:
  - In bidirectional mode, events can be enabled based on the count direction
  - The following conditions define an event: a counter match condition, an input (or output) condition such as an rising or falling edge or level, a combination of match and/or input/output condition
  - Selected events can limit, halt, start, or stop a counter or change its direction
  - Events trigger state changes, output transitions, timer captures, interrupts, and DMA transactions
  - Match register 0 can be used as an automatic limit
  - In bidirectional mode, events can be enabled based on the count direction

– Match events can be held until another qualifying event occurs

- State control features:
  – A state is defined by events that can happen in the state while the counter is running
  – A state changes into another state as a result of an event
  – Each event can be assigned to one or more states
  – State variable allows sequencing across multiple counter cycles

13.3 Basic configuration

Configure the SCT as follows:

- Enable the clock to the SCTimer/PWM (SCT) by setting CLK_SCT_EN (bit 8 of CLK_EN register in Section 2) to 1, to enable the register interface and the peripheral clock
- The SCT provides an interrupt to the NVIC, see Table 49
- Configure the SCT outputs or inputs to external pins by setting PIN_MUX_CTRL<x> according to pin mux table Table 95
- The SCT DMA request lines are connected to the DMA trigger inputs via the DMA_ITRIG_PINMUX registers

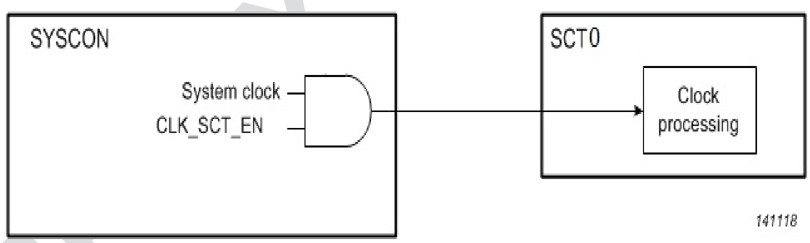


Fig 37. SCT clocking (AHBCLKCTRL1[2] should be replaced with CLK_PWM_EN

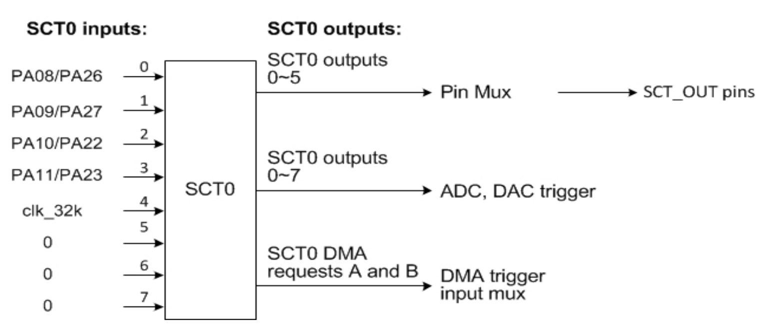


Fig 38. SCT connections

13.4 Pin description

See Section 8 to assign the SCT functions to external pins.

SCT input signals are predefined. The signals from external pins and internal signals are connected directly to the SCT inputs and not routed through IOCON.

SCT outputs can be routed to multiple places and can be connected to both a pin and an of ADC trigger at the same time.

Table 157. SCT0 pin description (inputs)

Function	Type	Connect to	Reference
SCT0_IN[0:7]	external from pin	4 GPIO pins (PA08 to PA11, or P26 P27 P22 P23)	Figure 38
	internal	Cau_clk_32k	Figure 38

Table 158. SCT0 pin description (outputs)

Type	Function	Connect to	Use register	Reference
external to pin	SCT0_OUT0	PA00, PA15, PA21	PIN_MUX_CTRL<x> register for the related	See Section 2
	SCT0_OUT1	PA01, PA16,PA20		
	SCT0_OUT2	PA02, PA19,PA07		
	SCT0_OUT3	PA03, PA18,PA06		
	SCT0_OUT4	PA04, PA13		
	SCT0_OUT5	PA05, PA12		
internal	-	ADC/DAC trigger	SEQ_A, SEQ_B	

13.5 General description

The SCTimer/PWM is a powerful and flexible timer module capable of creating complex PWM waveforms and performing other advanced timing and control operations with minimal or no CPU intervention.

The SCT can operate as a single 32-bit counter or as two independent, 16-bit counters in unidirectional or bidirectional mode. As with most timers, the SCT supports a selection of match registers against which the count value can be compared, and capture registers where the current count value can be recorded when some pre-defined condition is detected.

An additional feature contributing to the versatility of the SCT is the concept of “events”. The SCT module supports multiple separate events that can be defined by the user based on some combination of parameters including a match on one of the match registers, and/or a transition on one of the SCT inputs or outputs, the direction of count, and other factors.

Every action that the SCT block can perform occurs in direct response to one of these user-defined events without any software overhead. Any event can be enabled to:

- Start, stop, or halt the counter
- Limit the counter, which means to clear the counter in unidirectional mode or change its direction in bidirectional mode
- Set, clear, or toggle any SCT output
- Force a capture of the count value into any capture register

- Generate an interrupt of DMA request

The SCT allows the user to group and filter events, thereby selecting some events to be enabled together while others are disabled in a given context. A group of enabled and disabled events can be described as a state, and several states with different sets of enabled and disabled events are allowed. Changing from one state to another is event driven as well and can therefore happen without software intervention. By defining these states, the SCTimer/PWM provides the means to run entire state machines in hardware with any desired level of complexity to accomplish complex waveform and timing tasks.

In a simple system such as a classical timer/counter with capture and match capabilities, all events that could cause the timer to capture the timer value or toggle a match output are enabled. They are enabled at all times, while the counter is running. In this case, no events are filtered and the system is described by one state that does not change. This is the default configuration of the SCT.

In a more complex system, two states could be set up, that allow some events in one state and not in the other. An event itself, enabled in both states, can then be used, to move from one state to the other and back while filtering out events in either state. In such a two-state system different waveforms at the SCT output can be created depending on the event history. Changing between states is event-driven and happens without any intervention by the CPU.

Formally, the SCTimer/PWM can be programmed as state machine generator. The ability to perform switching between groups of events provides the SCT the unique capability to be utilized as a highly complex state machine engine. Events identify the occurrence of conditions that warrant state changes and determine the next state to move to. This provides an extremely powerful control tool - particularly when the SCT inputs and outputs are connected to other on-chip resources (such as ADC triggers, other timers etc.) in addition to general-purpose I/O.

In addition to events and states, the SCTimer/PWM provides other enhanced features:

- Four alternative clocking modes including a fully asynchronous mode
- Selection of any SCT input as a clock source or a clock gate
- Capability of selecting a “greater-than-or-equal-to” match condition for the purpose of event generation

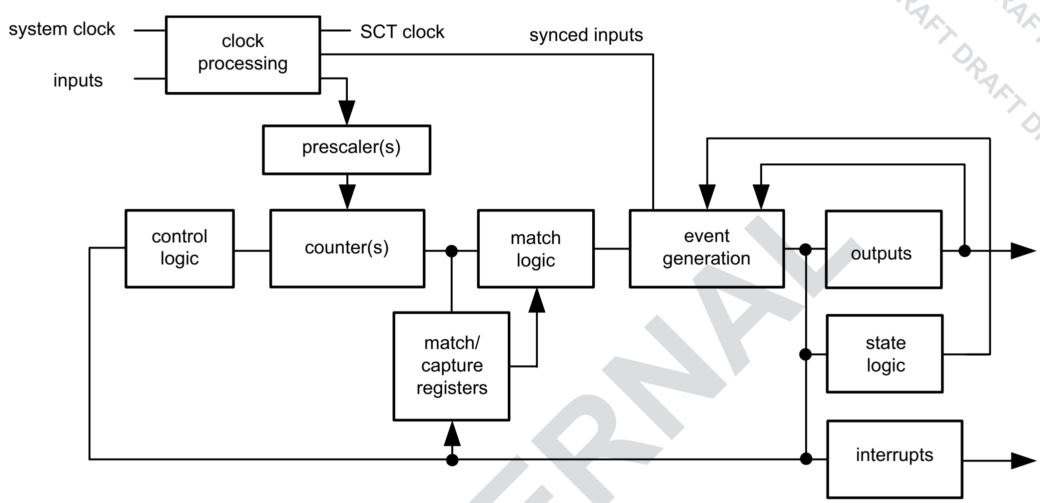


Fig 39. SCTimer/PWM block diagram

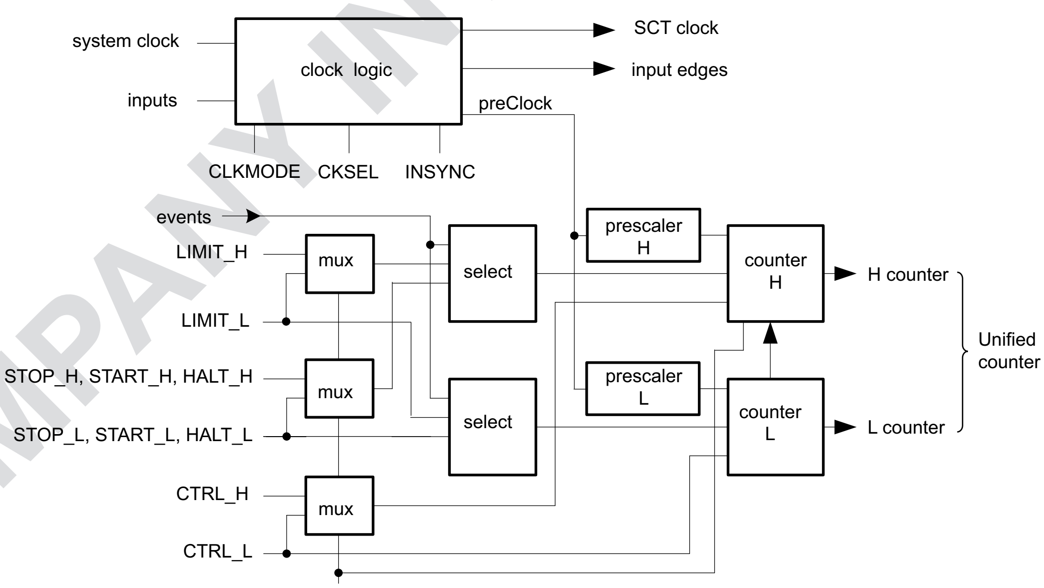


Fig 40. SCTimer/PWM counter and select logic

Remark: In this chapter, the term bus error indicates an SCT response that makes the processor take an exception.

13.6 Register description

The register addresses of the SCTimer/PWM are shown in Table 159. For most of the SCT registers, the register function depends on the setting of certain other register bits:

1. The UNIFY bit in the CONFIG register determines whether the SCT is used as one 32-bit register (for operation as one 32-bit counter/timer) or as two 16-bit counter/timers named L and H. The setting of the UNIFY bit is reflected in the register map:

– UNIFY: 1: Only one register is used (for operation as one 32-bit counter/timer)

– UNIFY: 0: Access the L and H registers by a 32-bit read or write operation or can be read or written to individually (for operation as two 16-bit counter/timers)

Typically, the UNIFY bit is configured by writing to the CONFIG register before any other registers are accessed.

2. The REGMODEn bits in the REGMODE register determine whether each set of Match/Capture registers uses the match or capture functionality:

– REGMODEn: 0: Registers operate as match and reload registers

– REGMODEn: 1: Registers operate as capture and capture control registers

Table 159. Register overview: SCTimer/PWM (base address 0x4008 5000)

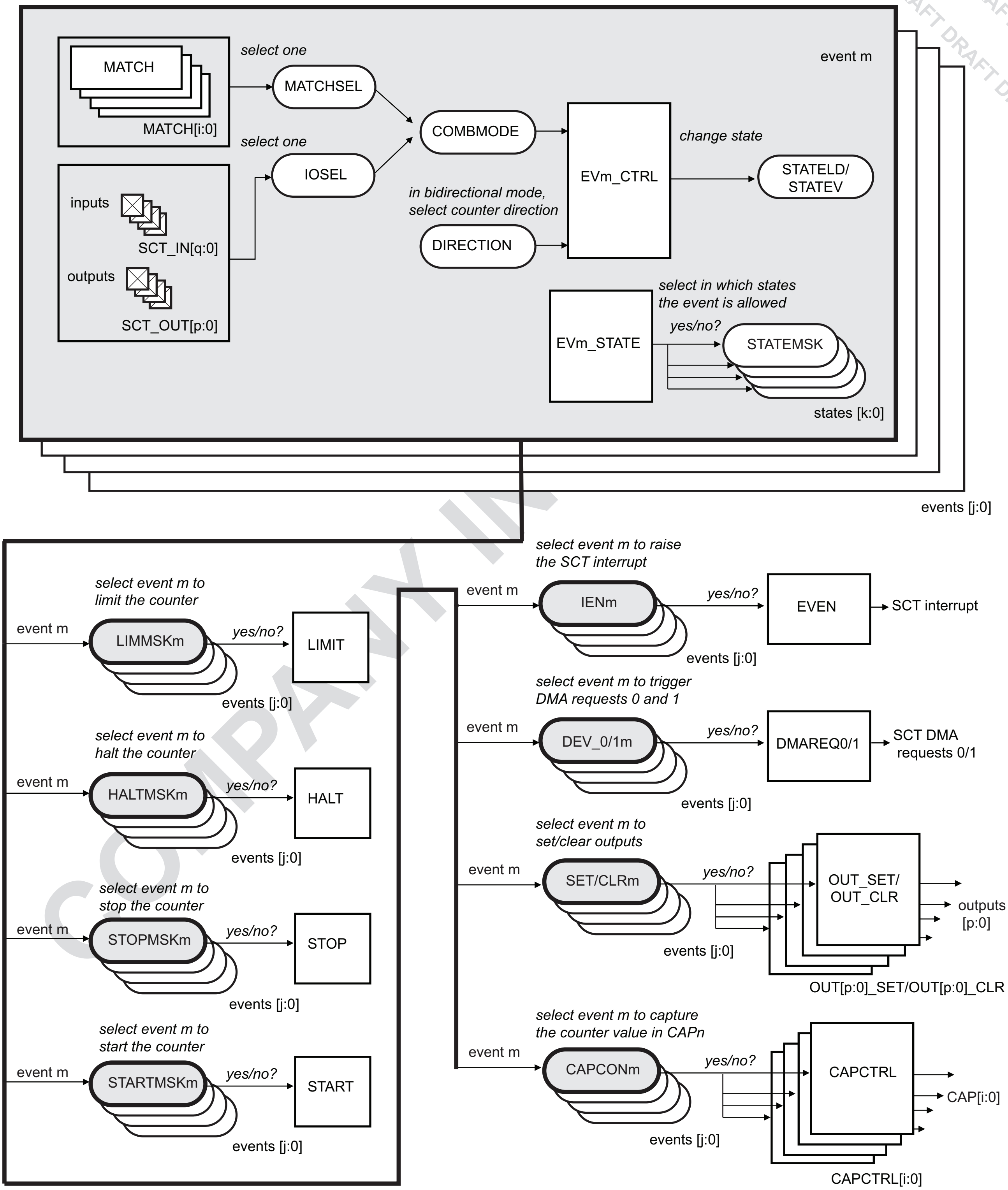
Name	Access	Offset	Description	Reset value	Section
CONFIG	R/W	0x000	SCT configuration register	0x0000 1E00	Section 13.6.2
CTRL	R/W	0x004	SCT control register	0x0004 0004	Section 13.6.3
LIMIT	R/W	0x008	SCT limit event select register	0x0	Section 13.6.4
HALT	R/W	0x00C	SCT halt event select register	0x0	Section 13.6.5
STOP	R/W	0x010	SCT stop event select register	0x0	Section 13.6.6
START	R/W	0x014	SCT start event select register	0x0	Section 13.6.7
COUNT	R/W	0x040	SCT counter register	0x0	Section 13.6.8
STATE	R/W	0x044	SCT state register	0x0	Section 13.6.9
INPUT	RO	0x048	SCT input register	0x0	Section 13.6.10
REGMODE	R/W	0x04C	SCT match/capture mode register	0x0	Section 13.6.11
OUTPUT	R/W	0x050	SCT output register	0x0	Section 13.6.12
OUTPUTDIRCTRL	R/W	0x054	SCT output counter direction control register	0x0	Section 13.6.13
RES	R/W	0x058	SCT conflict resolution register	0x0	Section 13.6.14
DMAREQUEST0	R/W	0x05C	SCT DMA request 0 register	0x0	Section 13.6.15
DMAREQUEST1	R/W	0x060	SCT DMA request 1 register	0x0	Section 13.6.15
EVEN	R/W	0x0F0	SCT event interrupt enable register	0x0	Section 13.6.16
EVFLAG	R/W	0x0F4	SCT event flag register	0x0	Section 13.6.17
CONEN	R/W	0x0F8	SCT conflict interrupt enable register	0x0	Section 13.6.18
CONFLAG	R/W	0x0FC	SCT conflict flag register	0x0	Section 13.6.19
MATCH0 to MATCH9	R/W	0x100 to 0x124	SCT match value register of match channels 0 to 9; REGMODE0 to REGMODE9: 0	0x0	Section 13.6.20
CAP0 to CAP9	R/W	0x100 to 0x124	SCT capture register of capture channel 0 to 9; REGMODE0 to REGMODE9: 1	0x0	Section 13.6.21
MATCHREL0 to MATCHREL9	R/W	0x200 to 0x224	SCT match reload value register 0 to 9; REGMODE0: 0 to REGMODE9: 0	0x0	Section 13.6.22
CAPCTRL0 to CAPCTRL9	R/W	0x200 to 0x224	SCT capture control register 0 to 9; REGMODE0: 1 to REGMODE9: 1	0x0	Section 13.6.23
EV0_STATE	R/W	0x300	SCT event state register 0	0x0	Section 13.6.24
EV0_CTRL	R/W	0x304	SCT event control register 0	0x0	Section 13.6.25
EV1_STATE	R/W	0x308	SCT event state register 1	0x0	Section 13.6.24
EV1_CTRL	R/W	0x30C	SCT event control register 1	0x0	Section 13.6.25
EV2_STATE	R/W	0x310	SCT event state register 2	0x0	Section 13.6.24

Table 159. Register overview: SCTimer/PWM (base address 0x4008 5000) *…continued*

Name	Access	Offset	Description	Reset value	Section
EV2_CTRL	R/W	0x314	SCT event control register 2	0x0	Section 13.6.25
EV3_STATE	R/W	0x318	SCT event state register 3	0x0	Section 13.6.24
EV3_CTRL	R/W	0x31C	SCT event control register 3	0x0	Section 13.6.25
EV4_STATE	R/W	0x320	SCT event state register 4	0x0	Section 13.6.24
EV4_CTRL	R/W	0x324	SCT event control register4	0x0	Section 13.6.25
EV5_STATE	R/W	0x328	SCT event state register 5	0x0	Section 13.6.24
EV5_CTRL	R/W	0x32C	SCT event control register 5	0x0	Section 13.6.25
EV6_STATE	R/W	0x330	SCT event state register 6	0x0	Section 13.6.24
EV6_CTRL	R/W	0x334	SCT event control register 6	0x0	Section 13.6.25
EV7_STATE	R/W	0x338	SCT event state register 7	0x0	Section 13.6.24
EV7_CTRL	R/W	0x33C	SCT event control register 7	0x0	Section 13.6.25
EV8_STATE	R/W	0x340	SCT event state register 8	0x0	Section 13.6.24
EV8_CTRL	R/W	0x344	SCT event control register 8	0x0	Section 13.6.25
EV9_STATE	R/W	0x348	SCT event state register 9	0x0	Section 13.6.24
EV9_CTRL	R/W	0x34C	SCT event control register 9	0x0	Section 13.6.25
OUT0_SET	R/W	0x500	SCT output 0 set register	0x0	Section 13.6.26
OUT0_CLR	R/W	0x504	SCT output 0 clear register	0x0	Section 13.6.27
OUT1_SET	R/W	0x508	SCT output 1 set register	0x0	Section 13.6.26
OUT1_CLR	R/W	0x50C	SCT output 1 clear register	0x0	Section 13.6.27
OUT2_SET	R/W	0x510	SCT output 2 set register	0x0	Section 13.6.26
OUT2_CLR	R/W	0x514	SCT output 2 clear register	0x0	Section 13.6.27
OUT3_SET	R/W	0x518	SCT output 3 set register	0x0	Section 13.6.26
OUT3_CLR	R/W	0x51C	SCT output 3 clear register	0x0	Section 13.6.27
OUT4_SET	R/W	0x520	SCT output 4 set register	0x0	Section 13.6.26
OUT4_CLR	R/W	0x524	SCT output 4 clear register	0x0	Section 13.6.27
OUT5_SET	R/W	0x528	SCT output 5 set register	0x0	Section 13.6.26
OUT5_CLR	R/W	0x52C	SCT output 5 clear register	0x0	Section 13.6.27
OUT6_SET	R/W	0x530	SCT output 6 set register	0x0	Section 13.6.26
OUT6_CLR	R/W	0x534	SCT output 6 clear register	0x0	Section 13.6.27
OUT7_SET	R/W	0x538	SCT output 7 set register	0x0	Section 13.6.26
OUT7_CLR	R/W	0x53C	SCT output 7 clear register	0x0	Section 13.6.27

13.6.1 Register functional grouping

Most SCT registers either configure an event or select an event for a specific action of the counter (or counters) and outputs. Figure 41 shows the registers and register bits that need to be configured for each event.



Note: in this figure, letters are used to represent the maximum quantity of certain SCTimer/PWM features as noted below.

i: match/captures, j: events, k: states, p: outputs, q: inputs

Fig 41. SCT event configuration and selection registers

13.6.1.1 Counter configuration and control registers

The SCT contains two registers for configuring the SCT and monitor and control its operation by software.

- The configuration register (CONFIG) configures the SCT in single, 32-bit counter mode or in dual, 16-bit counter mode. It configures the clocking and clock synchronization, and also configures automatic limits and the use of reload registers
- The control register (CTRL) allows to monitor and set the counter direction, and to clear, start, stop, or halt the 32-bit counter or each individual 16-bit counter if in dual-counter mode

13.6.1.2 Event configuration registers

Each event is associated with two registers:

- One EVn_CTRL register per event to define what triggers the event
- One EVn_STATE register per event to enable the event

13.6.1.3 Match and capture registers

The SCT includes a set of registers to store the SCT match or capture values. Each match register is associated with a match reload register which automatically reloads the match register at the beginning of each counter cycle. This register group includes the following registers:

- One REGMODE register per match/capture register to configure each match/capture register for either storing a match value or a capture value
- A set of match/capture registers with each register, depending on the setting of REGMODE, either storing a match value or a counter value
- One reload register for each match register

13.6.1.4 Event select registers for the counter operations

This group contains the registers that select events which affect the counter. Counter actions are limit, halt, and start or stop and apply to the unified counter or to the two 16-bit counters. Also included is the counter register with the counter value, or values in the dual-counter set-up. This register group includes the following registers:

- LIMIT selects the events that limit the counter
- START and STOP select events that start or stop the counter
- HALT selects events that halt the counter: HALT
- COUNT contains the counter value

The LIMIT, START, STOP, and HALT registers each contain one bit per event that selects for each event whether the event limits, stops, starts, or halts the counter, or counters in dual counter mode.

In the dual counter mode, the events can be selected independently for each counter.

13.6.1.5 Event select registers for setting or clearing the outputs

This group contains registers that select the events affecting the level of each SCT output. Also included are registers to manage conflicts that occur when events try to set or clear the same output. This register group includes the following registers:

- One OUTn_SET register for each output to select the events which set the output
- One OUTn_CLR register for each output to select the events which clear the output
- The conflict resolution register which defines an action when more than one event try to control an output at the same time
- The conflict flag and conflict interrupt enable registers that monitor interrupts arising from output set and clear conflicts
- The output direction control register that interchanges the set and clear output operation caused by an event in bidirectional mode

The OUTn_SET and OUTn_CLR registers each contain one bit per event that selects whether the event changes the state a given output n.

In the dual counter mode, the events can be selected independently for each output.

13.6.1.6 Event select registers for capturing a counter value

This group contains registers that select events which capture the counter value and store it in one of the CAP registers. Each capture register m has one associated CAPCTRLm register which in turn selects the events to capture the counter value.

13.6.1.7 Event select register for initiating DMA transfers

One register is provided for each of the two DMA requests to select the events that can trigger a DMA request.

The DMAREQn register contain one bit for each event that selects whether this event triggers a DMA request. An additional bit enables the DMA trigger when the match registers are reloaded.

13.6.1.8 Interrupt handling registers

The following registers provide flags that are set by events and select the events that when they occur request an interrupt:

- The event flag register which provides one flag for each event that is set when the event occurs
- The event flag interrupt enable register which provides one bit for each event to be enabled for the SCT interrupt.

13.6.1.9 Registers for controlling SCT inputs and outputs by software

Two registers are provided that allow software (as opposed to events) to set input and outputs of the SCT:

- The SCT input register to read the state of any of the SCT inputs
- The SCT output register to set or clear any of the SCT outputs or to read the state of the outputs

13.6.2 SCT configuration register

This register configures the overall operation of the SCT. Write to this register before any other registers. Only word writes are permitted to this register. Attempting to write a halfword value results in a bus error.

Table 160. CONFIG - SCT configuration register (offset = 0x000) bit description

Bit	Symbol	Access	Reset value	Description
0	UNIFY	-	0x0	SCT operation
		0		SCT operates as two 16-bit counters named COUNTER_L and COUNTER_H
		1		SCT operates as a unified 32-bit counter
2:1	CLKMODE		0x0	SCT clock mode
		0x0		System clock mode. The system clock clocks the entire SCT module including the counter(s) and counter prescalers
		0x1		Sampled system clock mode. The system clock clocks the SCT module, but the counter and prescalers are only enabled to count when the designated edge is detected on the input selected by the CKSEL field. The minimum pulse width on the selected clock-gate input is 1 bus clock period. This mode is the high-performance, sampled-clock mode
		0x2		SCT input clock mode. The input/edge selected by the CKSEL field clocks the SCT module, including the counters and prescalers, after first being synchronized to the system clock. The minimum width of the positive and negative phases of the clock input must each be greater than one full period of the bus/system clock
		0x3		Asynchronous mode. The entire SCT module is clocked directly by the input/edge selected by the CKSEL field. In this mode, the SCT outputs are switched synchronously to the SCT input clock - not the system clock. The input clock rate must be at least half the system clock rate and can be the same or faster than the system clock
6:3	CKSEL	-	0x0	SCT clock select. The specific functionality of the designated input/edge is dependent on the CLKMODE bit selection in this register
		0x0		Rising edges on input 0
		0x1		Falling edges on input 0
		0x2		Rising edges on input 1
		0x3		Falling edges on input 1
		0x4		Rising edges on input 2
		0x5		Falling edges on input 2
		0x6		Rising edges on input 3
		0x7		Falling edges on input 3
		0x8		Rising edges on input 4
		0x9		Falling edges on input 4
		0xA		Rising edges on input 5
		0xB		Falling edges on input 5
		0xC		Rising edges on input 6
		0xD		Falling edges on input 6
		0xE		Rising edges on input 7
		0XF		Falling edges on input 7
7	NORELAOD_L	-	0x0	1 in this bit prevents the lower match registers from being reloaded from their respective reload registers. Setting this bit eliminates the need to write to the reload registers MATCHREL if the match values are fixed. Software can write to set or clear this bit at any time. This bit applies to both the higher and lower registers when the UNIFY bit is set

Table 160. CONFIG - SCT configuration register (offset = 0x000) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
8	NORELOAD_H	-	0x0	1 in this bit prevents the higher match registers from being reloaded from their respective reload registers. Setting this bit eliminates the need to write to the reload registers MATCHREL if the match values are fixed. Software can write to set or clear this bit at any time. This bit is not used when the UNIFY bit is set
12:9	INSYNC	-	0xF	Synchronization for input N (bit 9: input 0, bit 10: input 1,..., bit 12: input 3); all other bits are reserved. A 1 in one of these bits subjects the corresponding input to synchronization to the SCT clock, before it is used to create an event. This synchronization injects a two SCT-clock delay in the input path. Clearing this bit bypasses synchronization on the corresponding input. This bit may be cleared for faster input response time if both of the following conditions are met (for all clock modes): The corresponding input is already synchronous to the SCT clock The SCT clock frequency does not exceed 100 MHz Note: The SCT clock is the bus/system clock for CKMODE 0-2 or the selected, asynchronous input clock for CKMODE3 Alternatively, for CKMODE2 only, it is also allowable to bypass synchronization if both of the following conditions are met: The corresponding input is synchronous to the designated CKMODE2 input clock The CKMODE2 input clock frequency is less than one-third the frequency of the bus/system clock
16:13	RESERVED	-	-	-
17	AUTOLIMIT_L	-	0x0	1 in this bit causes a match on match register 0 to be treated as a de-facto LIMIT condition without the need to define an associated event As with any LIMIT event, this automatic limit causes the counter to be cleared to zero in unidirectional mode or to change the direction of count in bidirectional mode Software can write to set or clear this bit at any time. This bit applies to both the higher and lower registers when the UNIFY bit is set
18	AUTOLIMIT_H	-	0x0	1 in this bit will cause a match on match register 0 to be treated as a de-facto LIMIT condition without the need to define an associated event As with any LIMIT event, this automatic limit causes the counter to be cleared to zero in unidirectional mode or to change the direction of count in bidirectional mode Software can write to set or clear this bit at any time. This bit is not used when the UNIFY bit is set
31:19	RESERVED	-	-	-

13.6.3 SCT control register

If bit UNIFY: 1 in the CONFIG register, only the _L bits are used.

If bit UNIFY: 0 in the CONFIG register, this register can be written to as two registers CTRL_L and CTRL_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

All bits in this register can be written to when the counter is stopped or halted. When the counter is running, the only bits that can be written are STOP or HALT. (Other bits can be written in a subsequent write after HALT is set to 1).

Remark: If CLKMODE: 0x3 is selected, wait at least 12 system clock cycles between a write access to the H, L or unified version of this register and the next write access. This restriction does not apply when writing to the HALT bit or bits and then writing to the CTRL register again to restart the counters - for example because software must update the MATCH register, which is only allowed when the counters are halted.

Remark: If the SCTimer/PWM is operating as two 16-bit counters, events can only modify the state of the outputs when neither counter is halted. This is true regardless of what triggered the event.

Table 161. CTRL - SCT control register (offset = 0x004) bit description

Bit	Symbol	Access	Reset value	Description
0	DOWN_L	-	0x0	This bit is 1 when the L or unified counter is counting down. Hardware sets this bit when the counter is counting up, counter limit occurs, and BIDIR: 1.Hardware clears this bit when the counter is counting down and a limit condition occurs or when the counter reaches 0
1	STOP_L	-	0x0	When this bit is 1 and HALT is 0, the L or unified counter does not run, but I/O events related to the counter can occur. If a designated start event occurs, this bit is cleared and counting resumes
2	HALT_L	-	0x1	When this bit is 1, the L or unified counter does not run and no events can occur. A reset sets this bit. When the HALT_L bit is one, the STOP_L bit is cleared. It is possible to remove the halt condition while keeping the SCT in the stop condition (not running) with a single write to this register to simultaneously clear the HALT bit and set the STOP bit **Remark:** Once set, only software can clear this bit to restore counter operation. This bit is set on reset
3	CLRCTR_L	-	0x0	Writing a 1 to this bit clears the L or unified counter. This bit always reads as 0
4	BIDIR_L	-	0x0	L or unified counter direction select
		0		Up. The counter counts up to a limit condition, then is cleared to zero
		1		Up-down. The counter counts up to a limit, then counts down to a limit condition or to 0
12:5	PRE_L	-	0x0	Specifies the factor by which the SCT clock is prescaled to produce the L or unified counter clock. The counter clock is clocked at the rate of the SCT clock divided by PRE_L+1 **Remark:** Clear the counter (by writing a 1 to the CLRCTR bit) whenever changing the PRE value
15:13	RESERVED	-	-	-
16	DOWN_H	-	0x0	This bit is 1 when the H counter is counting down. Hardware sets this bit when the counter is counting, a counter limit condition occurs, and BIDIR is 1. Hardware clears this bit when the counter is counting down and a limit condition occurs or when the counter reaches 0
17	STOP_H	-	0x0	When this bit is 1 and HALT is 0, the H counter does not, run but I/O events related to the counter can occur. If such an event matches the mask in the Start register, this bit is cleared and counting resumes
18	HALT_H	-	0x1	When this bit is 1, the H counter does not run and no events can occur. A reset sets this bit. When the HALT_H bit is one, the STOP_H bit is cleared It is possible to remove the halt condition while keeping the SCT in the stop condition (not running) with a single write to this register to simultaneously clear the HALT bit and set the STOP bit **Remark:** Once set, this bit can only be cleared by software to restore counter operation. This bit is set on reset

Table 161. CTRL - SCT control register (offset = 0x004) bit description *...continued*

Bit	Symbol	Access	Reset value	Description
19	CLRCTR_H	-	0x0	Writing a 1 to this bit clears the H counter. This bit always reads as 0
20	BIDIR_H	-	0x0	Direction select
		0		The H counter counts up to its limit condition, then is cleared to zero
		1		The H counter counts up to its limit, then counts down to a limit condition or to 0
28:21	PRE_H	-	0x0	Specifies the factor by which the SCT clock is prescaled to produce the H counter clock. The counter clock is clocked at the rate of the SCT clock divided by PRELH+1 **Remark:** Clear the counter (by writing a 1 to the CLRCTR bit) whenever changing the PRE value
31:29	RESERVED	-	-	-

13.6.4 SCT limit event select register

The running counter can be limited by an event. When any of the events selected in this register occur, the counter is cleared to zero from its current value or changes counting direction if in bidirectional mode.

Each bit of the register is associated with a different event (bit 0 with event 0, etc.). Setting a bit causes its associated event to serve as a LIMIT event. When any limit event occurs, the counter is reset to zero in unidirectional mode or changes its direction of count in bidirectional mode and keeps running.To define the actual limiting event (a match, an I/O pin toggle, etc.), see the EVn_CTRL register.

Remark: Counting up to all ones or counting down to zero is equivalent to a limit event occurring.

Note that in addition to using this register to specify events that serve as limits, it is also possible to automatically cause a limit condition whenever a match register 0 match occurs. This eliminates the need to define an event for the sole purpose of creating a limit. The AUTOLIMITL and AUTOLIMITH bits in the configuration register enable/disable this feature (see Table 160).

If UNIFY: 1 in the CONFIG register, only the _L bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers LIMIT_L and LIMIT_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Table 162. LIMIT - SCT limit event select register (offset = 0x008) bit description

Bit	Symbol	Access	Reset value	Description
15:0	LIMMSK_L	-	0x0	If bit n is one, event n is used as a counter limit for the L or unified counter (event 0: bit 0, event 1: bit 1, ...). The number of bits: number of events supported by this SCT
31:16	LIMMSK_H	-	0x0	If bit n is one, event n is used as a counter limit for the H counter (event 0: bit 16, event 1: bit 17, ...). The number of bits: number of events supported by this SCT

13.6.5 SCT halt event select register

The running counter can be disabled (halted) by an event. When any of the events selected in this register occur, the counter stops running and all further events are disabled.

Each bit of the register is associated with a different event (bit 0 with event 0, etc.). Setting a bit will cause its associated event to serve as a HALT event. To define the actual events that cause the counter to halt (a match, an I/O pin toggle, etc.), see the EVn_CTRL registers.

Remark: A HALT condition can only be removed when software clears the HALT bit in the CTRL register (Table 161).

If UNIFY: 1 in the CONFIG register, only the L bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers HALT_L and HALT_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Table 163. SCT halt event select register (HALT, offset 0x00C) bit description

Bit	Symbol	Access	Reset value	Description
15:0	HALTMSK_L	-	0x0	If bit n is one, event n sets the HALT_L bit in the CTRL register (event 0: bit 0, event 1: bit 1, ...). The number of bits: number of events supported by this SCT
31:16	HALTMSK_H	-	0x0	If bit n is one, event n sets the HALT_H bit in the CTRL register (event 0: bit 16, event 1: bit 17, ...). The number of bits: number of events supported by this SCT

13.6.6 SCT stop event select register

The running counter can be stopped by an event. When any of the events selected in this register occur, counting is suspended, that is, the counter stops running and remains at its current value. Event generation remains enabled, and any event selected in the START register such as an I/O event or an event generated by the other counter can restart the counter.

This register specifies which events stop the counter. Each bit of the register is associated with a different event (bit 0 with event 0, etc.). Setting a bit will cause its associated event to serve as a STOP event. To define the actual event that causes the counter to stop (a match, an I/O pin toggle, etc.), see the EVn_CTRL register.

Remark: Software can stop and restart the counter by writing to the CTRL register.

If UNIFY: 1 in the CONFIG register, only the _L bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers STOPT_L and STOP_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Table 164. STOP - SCT stop event select register (offset = 0x010) bit description

Bit	Symbol	Access	Reset value	Description
15:0	STOPMSK_L	-	0x0	If bit n is one, event n sets the STOP_L bit in the CTRL register (event 0: bit 0, event 1: bit 1, …). The number of bits: number of events supported by this SCT
31:16	STOPMSK_H	-	0x0	If bit n is one, event n sets the STOP_H bit in the CTRL register (event 0: bit 16, event 1: bit 17, …). The number of bits: number of events supported by this SCT

13.6.7 SCT start event select register

The stopped counter can be re-started by an event. When any of the events selected in this register occur, counting is restarted from the current counter value.

Each bit of the register is associated with a different event (bit 0 with event 0, etc.). Setting a bit will cause its associated event to serve as a START event. When any START event occurs, hardware will clear the STOP bit in the control register CTRL. Note that a START event has no effect on the HALT bit. Only software can remove a HALT condition. To define the actual event that starts the counter (an I/O pin toggle or an event generated by the other running counter in dual-counter mode), see the EVn_CTRL register.

If UNIFY: 1 in the CONFIG register, only the _L bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers START_L and START_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Table 165. START - SCT start event select register (offset = 0x014) bit description

Bit	Symbol	Access	Reset value	Description
15:0	STARTMSK_L	-	0x0	If bit n is one, event n clears the STOP_L bit in the CTRL register (event 0: bit 0, event 1: bit 1, …). The number of bits: number of events supported by this SCT
31:16	STARTMSK_H	-	0x0	If bit n is one, event n clears the STOP_H bit in the CTRL register (event 0: bit 16, event 1: bit 17, …). The number of bits: number of events supported by this SCT

13.6.8 SCT counter register

If UNIFY: 1 in the CONFIG register, the counter is a unified 32-bit register and both the _L and _H bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers COUNT_L and COUNT_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation. In this case, the L and H registers count independently under the control of the other registers.

Writing to the COUNT_L, COUNT_H, or unified register is only allowed when the corresponding counter is halted (HALT bits are set to 1 in the CTRL register). Attempting to write to the counter when it is not halted causes a bus error. Software can read the counter registers at any time.

Table 166. COUNT - SCT counter register (offset = 0x040) bit description

Bit	Symbol	Access	Reset value	Description
15:0	CTR_L	-	0x0	When UNIFY: 0, read or write the 16-bit L counter value. When UNIFY: 1, read or write the lower 16 bits of the 32-bit unified counter
31:16	CTR_H	-	0x0	When UNIFY: 0, read or write the 16-bit H counter value. When UNIFY: 1, read or write the upper 16 bits of the 32-bit unified counter

13.6.9 SCT state register

Each group of enabled and disabled events is assigned a number called the state variable. For example, a state variable with a value of 0 could have events 0, 2, and 3 enabled and all other events disabled. A state variable with the value of 1 could have events 1, 4, and 5 enabled and all others disabled.

Remark: The EVm_STATE registers define which event is enabled in each group.

Software can read the state associated with a counter at any time. Writing to the STATE_L, STATE_H, or unified register is only allowed when the corresponding counter is halted (HALT bits are set to 1 in the CTRL register).

The state variable is the main feature that distinguishes the SCTimer/PWM from other counter/timer/ PWM blocks. Events can be made to occur only in certain states. Events, in turn, can perform the following actions:

- set and clear outputs
- limit, stop, and start the counter
- cause interrupts and DMA requests
- modify the state variable

The value of a state variable is completely under the control of the application. If an application does not use states, the value of the state variable remains zero, which is the default value.

A state variable can be used to track and control multiple cycles of the associated counter in any desired operational sequence. The state variable is logically associated with a state machine diagram which represents the SCT configuration. See Section 13.6.24 and Section 13.6.25 for more about the relationship between states and events.

The STATELD/STADEV fields in the event control registers of all defined events set all possible values for the state variable. The change of the state variable during multiple counter cycles reflects how the associated state machine moves from one state to the next.

If UNIFY: 1 in the CONFIG register, only the _L bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers STATE_L and STATE_H. Both L and H registers can be read or written individually or in a single 32-bit read or write operation.

Table 167. SCT state register (STATE, offset 0x044) bit description

Bit	Symbol	Access	Reset value	Description
4:0	STATE_L	-	0x0	State variable
15:5	RESERVED	-	-	-
20:16	STATE_H	-	0x0	State variable
31:21	RESERVED	-	-	-

13.6.10 SCT input register

Software can read the state of the SCT inputs in this read-only register in slightly different forms.

- The AIN bit displays the state of the input captured on each rising edge of the SCT clock. This corresponds to a nearly direct read-out of the input but can cause spurious fluctuations in case of an asynchronous input signal.
- The SIN bit displays the form of the input as it is used for event detection. This may include additional stages of synchronization, depending on what is specified for that input in the INSYNC field in the CONFIG register:
  - If the INSYNC bit is set for the input, the input is triple-synchronized to the SCT clock resulting in a stable signal that is delayed by three SCT clock cycles
  - If the INSYNC bit is not set, the SIN bit value is identical to the AIN bit value

Table 168. SCT input register (INPUT, offset 0x048) bit description

Bit	Symbol	Access	Reset value	Description
0	AIN0	-	-	Input 0 state. Input 0 state on the last SCT clock edge
1	AIN1	-	-	Input 1 state. Input 1 state on the last SCT clock edge
2	AIN2	-	-	Input 2 state. Input 2 state on the last SCT clock edge
3	AIN3	-	-	Input 3 state. Input 3 state on the last SCT clock edge
15:4	AIN…	-	-	Input state for the remainder of inputs implemented in this SCT
16	SIN0	-	-	Input 0 state. Input 0 state following the synchronization specified by INSYNC0
17	SIN1	-	-	Input 1 state. Input 1 state following the synchronization specified by INSYNC0
18	SIN2	-	-	Input 2 state. Input 2 state following the synchronization specified by INSYNC0
19	SIN3	-	-	Input 3 state. Input 3 state following the synchronization specified by INSYNC0
31:20	SIN…	-	-	Input state for the remainder of states implemented in this SCT

13.6.11 SCT match/capture mode register

If UNIFY: 1 in the CONFIG register, only the _L bits of this register are used. In this case, REGMODE_H is not used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers REGMODE_L and REGMODE_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation. The _L bits/registers control the L match/capture registers, and the _H bits/registers control the H match/capture registers.

The SCT contains multiple match/capture registers. The register mode register selects whether each register acts as a match register (see Section 13.6.20) or as a capture register (see Section 13.6.21). Each match/capture register has an accompanying

register which functions as a reload register when the primary register is used as a match register (Section 13.6.22) or as a capture-control register when the register is used as a capture register (Section 13.6.23). REGMODE_H is used only when the UNIFY bit is 0.

Table 169. REGMODE - SCT match/capture mode register (offset = 0x04C) bit description

Bit	Symbol	Access	Reset value	Description
15:0	REGMOD_L	-	0x0	Each bit controls one match/capture register (register 0: bit 0, register 1: bit 1, …). The number of bits: number of match/captures supported by this SCT
		0		register operates as match registers
		1		register operates as capture register
31:16	REGMOD_H	-	0x0	Each bit controls one match/capture register (register 0: bit 16, register 1: bit 17, …). The number of bits: number of match/captures supported by this SCT
		0		register operates as match registers
		1		register operates as capture register

13.6.12 SCT output register

Each SCT output has a corresponding bit in this register to allow software to control the output state directly or read its current state.

While the counter is running, outputs are set, cleared, or toggled only by events. However, using this register, software can write to any of the output registers when both counters are halted to control the outputs directly. Writing to the OUT register is only allowed when all counters (L-counter, H-counter, or unified counter) are halted (HALT bits are set to 1 in the CTRL register).

Software can read this register at any time to sense the state of the outputs.

Table 170. OUTPUT - SCT output register (offset = 0x050) bit description

Bit	Symbol	Access	Reset value	Description
15:0	OUT	-	0x0	Writing a 1 to bit n forces the corresponding output HIGH. Writing a 0 forces the corresponding output LOW (output 0: bit 0, output 1: bit 1, …). The number of bits: number of outputs supported by this SCT
31:16	RESERVED	-	-	-

13.6.13 SCT bidirectional output control register

For bidirectional mode, this register specifies (for each output) the impact of counting direction on the meaning of set and clear operations on output; see Section 13.6.26 and Section 13.6.27. The purpose of this register is to facilitate the creation of center-aligned output waveforms without the need to define additional events.

Table 171. OUTPUTDIRCTRL - SCT bidirectional output control register (offset = 0x054) bit description

Bit	Symbol	Access	Reset value	Description
1:0	SETCLR0	-	0x0	set/clear operation on output 0
		0x0		set and clear do not depend on the direction of any counter
		0x1		set and clear are reversed when counter L or the unified counter is counting down
		0x2		set and clear are reversed when counter H is counting down; do not use if UNIFY: 1
		0x3		value 0x3 is reserved; do not program this value
3:2	SETCLR1	-	0	set/clear operation on output 1. Value 0x3 is reserved; do not program this value
		0x0		set and clear do not depend on the direction of any counter
		0x1		set and clear are reversed when counter L or the unified counter is counting down
		0x2		set and clear are reversed when counter H is counting down; do not use if UNIFY: 1
5:4	SETCLR2	-	0	set/clear operation on output 2; value 0x3 is reserved; do not program this value
		0x0		set and clear do not depend on the direction of any counter
		0x1		set and clear are reversed when counter L or the unified counter is counting down
		0x2		set and clear are reversed when counter H is counting down; do not use if UNIFY: 1
7:6	SETCLR3	-	0	set/clear operation on output 3; value 0x3 is reserved; do not program this value
		0x0		set and clear do not depend on the direction of any counter
		0x1		set and clear are reversed when counter L or the unified counter is counting down
		0x2		set and clear are reversed when counter H is counting down; do not use if UNIFY: 1
9:8	SETCLR4	-	0	set/clear operation on output 4; value 0x3 is reserved; do not program this value
		0x0		set and clear do not depend on the direction of any counter
		0x1		set and clear are reversed when counter L or the unified counter is counting down
		0x2		set and clear are reversed when counter H is counting down; do not use if UNIFY: 1
31:10	SETCLR...	-	0	set/clear operation controls for the remainder of outputs on this SCT

[1] For as many outputs as are supported by the specific SCTimer/PWM.

13.6.14 SCT conflict resolution register

The output conflict resolution register specifies what action should be taken if multiple events (or even the same event) dictate that a given output should be both set and cleared at the same time.

To enable an event to toggle an output each time the event occurs, set the bits for that event in both the OUTn_SET and OUTn_CLR registers and set the On_RES value to 0x3 in this register.

Table 172. RES - SCT conflict resolution register (offset = 0x058) bit description

Bit	Symbol	Access	Reset value	Description
1:0	O0RES	-	0x0	effect of simultaneous set and clear on output 0
		0x0		no change
		0x1		set output (or clear based on the SETCLR0 field in the OUTPUTDIRCTRL register)
		0x2		clear output (or set based on the SETCLR0 field)
		0x3		toggle output
3:2	O1RES	-	0	effect of simultaneous set and clear on output 1
		0x0		no change
		0x1		set output (or clear based on the SETCLR1 field in the OUTPUTDIRCTRL register)
		0x2		clear output (or set based on the SETCLR1 field)
		0x3		toggle output
5:4	O2RES	-	0	effect of simultaneous set and clear on output 2
		0x0		no change
		0x1		set output (or clear based on the SETCLR2 field in the OUTPUTDIRCTRL register)
		0x2		clear output n (or set based on the SETCLR2 field)
		0x3		toggle output
7:6	O3RES	-	0	effect of simultaneous set and clear on output 3
		0x0		no change
		0x1		set output (or clear based on the SETCLR3 field in the OUTPUTDIRCTRL register)
		0x2		clear output (or set based on the SETCLR3 field)
		0x3		toggle output
9:8	O4RES	-	0	effect of simultaneous set and clear on output 4
		0x0		no change
		0x1		set output (or clear based on the SETCLR4 field in the OUTPUTDIRCTRL register)
		0x2		clear output (or set based on the SETCLR4 field)
		0x3		toggle output
31:10	O...RES	-	0	resolution controls for the remainder of outputs on this SCT

[1] For as many outputs as are supported by the specific SCTimer/PWM.

13.6.15 SCT DMA request 0 and 1 registers

The SCT includes two DMA request outputs. These registers enable the DMA requests to be triggered when a particular event occurs or when counter match registers are loaded from its reload registers. The DMA request registers are word-write only. Attempting to write a half-word value to these registers result in a bus error.

Event-triggered DMA requests are particularly useful for launching DMA activity to or from other peripherals under the control of the SCT.

Table 173. DMAREQUEST0 - SCT DMA 0 request register (offset = 0x05C) bit description

Bit	Symbol	Access	Reset value	Description
15:0	DEV_0	-	0x0	If bit n is one, event n triggers DMA request 0 (event 0: bit 0, event 1: bit 1, ...). The number of bits: number of events supported by this SCT
29:16	RESERVED	-	-	-
30	DRL0	-	0x0	A 1 in this bit triggers DMA request 0 when it loads the MATCH_L/Unified registers from the RELOAD_L/Unified registers
31	DRQ0	-	0x0	This read-only bit indicates the state of DMA Request 0 Note that if the related DMA channel is enabled and properly set up, it is unlikely that software will see this flag, it will be cleared rapidly by the DMA service. The flag remaining set could point to an issue with DMA setup

Table 174. SCT DMA 1 request register (DMAREQUEST1, offset 0x060) bit description

Bit	Symbol	Access	Reset value	Description
15:0	DEV_1	-	0x0	If bit n is one, event n triggers DMA request 1 (event 0: bit 0, event 1: bit 1, ...). The number of bits: number of events supported by this SCT
29:16	RESERVED	-	-	-
30	DRL1	-	0x0	A 1 in this bit triggers DMA request 1 when it loads the Match L/Unified registers from the Reload L/Unified registers
31	DRQ1	-	0x0	This read-only bit indicates the state of DMA Request 1 Note that if the related DMA channel is enabled and properly set up, it is unlikely that software will see this flag, it will be cleared rapidly by the DMA service. The flag remaining set could point to an issue with DMA setup

13.6.16 SCT event interrupt enable register

If the FLAGn bit in the SCT event flag register (Section 13.6.17) is also set, this register enables flags to request an interrupt.

Table 175. SCT event interrupt enable register (EVEN, offset 0x0F0) bit description

Bit	Symbol	Access	Reset value	Description
15:0	IEN	-	0x0	The SCT requests an interrupt when bit n of this register and the event flag register are both one (event 0: bit 0, event 1: bit 1, ...). The number of bits: number of events supported by this SCT
31:16	RESERVED	-	-	-

13.6.17 SCT event flag register

This register records events. Writing ones to this register clears the corresponding flags and negates the SCT interrupt request if all enabled flag register bits are zero.

Table 176. EVFLAG - SCT event flag register (offset = 0x0F4) bit description

Bit	Symbol	Access	Reset value	Description
15:0	FLAG	-	0x0	Bit n is one if event n has occurred since reset or a 1 was last written to this bit (event 0: bit 0, event 1: bit 1, ...). The number of bits: number of events supported by this SCT.
31:16	RESERVED	-	-	-

13.6.18 SCT conflict interrupt enable register

This register enables the no-change conflict events specified in the SCT conflict resolution register to generate an interrupt request.

Table 177. CONEN - SCT conflict interrupt enable register (offset = 0x0F8) bit description

Bit	Symbol	Access	Reset value	Description
15:0	NCEN	-	0x0	The SCT requests an interrupt when bit n of this register and the SCT conflict flag register are both one (output 0: bit 0, output 1: bit 1, ...). The number of bits: number of outputs supported by this SCT
31:16	RESERVED	-	-	-

13.6.19 SCT conflict flag register

This register records a no-change conflict occurrence and provides details of a bus error. Writing ones to the NCFLAG bits clears the corresponding read bits and negates the SCT interrupt request if all enabled flag bits are zero.

Table 178. CONFLAG - SCT conflict flag register (offset = 0x0FC) bit description

Bit	Symbol	Access	Reset value	Description
15:0	NCFLAG	-	0x0	Bit n is one if a no-change conflict event occurred on output n since reset or a 1 was last written to this bit (output 0: bit 0, output 1: bit 1, ...). The number of bits: number of outputs supported by this SCT
29:16	RESERVED	-	-	-
30	BUSERRL	-	0x0	The most recent bus error from this SCT involved writing CTR L/Unified, STATE L/Unified, MATCH L/Unified, or the Output register when the L/U counter was not halted. A word write to certain L and H registers can be half successful and half unsuccessful
31	BUSERRH	-	0x0	The most recent bus error from this SCT involved writing CTR H, STATE H, MATCH H, or the Output register when the H counter was not halted

13.6.20 SCT match registers 0 to 9 (REGMODEn bit: 0)

Match registers are compared to the counters to help create events. When the UNIFY bit is 0, the L and H registers are independently compared to the L and H counters. When UNIFY is 1, the combined L and H registers hold a 32-bit value that is compared to the unified counter. A match can only occur in a clock in which the counter is running (STOP and HALT are both 0).

Match registers can be read at any time. Writing to the MATCH_L, MATCH_H, or unified register is only allowed when the corresponding counter is halted (HALT bits are set to 1 in the CTRL register). Match events occur in the SCT clock in which the counter is (or would be) incremented to the next value. When a match event limits its counter as described in Section 13.6.4, the value in the match register is the last value of the counter before it is cleared to zero (or decremented if BIDIR is 1).

There is no “write-through” from reload registers to match registers. Before starting a counter, software can write one value to the match register used in the first cycle of the counter and a different value to the corresponding match reload register used in the second cycle.

Table 179. SCT match registers 0 to 9 (MATCH[0:9], offset 0x100 (MATCH0) to 0x124 (MATCH9)) bit description (REGMODEn bit: 0)

Bit	Symbol	Access	Reset value	Description
15:0	MATCHn_L	-	0x0	When UNIFY: 0, read or write the 16-bit value to be compared to the L counter When UNIFY: 1, read or write the lower 16 bits of the 32-bit value to be compared to the unified counter
31:16	MATCHn_H	-	0x0	When UNIFY: 0, read or write the 16-bit value to be compared to the H counter When UNIFY: 1, read or write the upper 16 bits of the 32-bit value to be compared to the unified counter

13.6.21 SCT capture registers 0 to 9 (REGMODEn bit: 1)

These registers allow software to record the counter values upon occurrence of the events selected by the corresponding capture control registers occurred.

Table 180. SCT capture registers 0 to 9 (CAP[0:9], offset 0x100 (CAP0) to 0x124 (CAP9)) bit description (REGMODEn bit: 1)

Bit	Symbol	Access	Reset value	Description
15:0	CAPn_L	-	0x0	When UNIFY: 0, read the 16-bit counter value at which this register was last captured. When UNIFY: 1, read the lower 16 bits of the 32-bit value at which this register was last captured
31:16	CAPn_H	-	0x0	When UNIFY: 0, read the 16-bit counter value at which this register was last captured. When UNIFY: 1, read the upper 16 bits of the 32-bit value at which this register was last captured

13.6.22 SCT match reload registers 0 to 9 (REGMODEn bit: 0)

A match register (L, H, or unified 32-bit) is loaded from its corresponding reload register at the start of each new counter cycle, that is

- when BIDIR: 0 and the counter is cleared to zero upon reaching it limit condition
- when BIDIR: 1 and the counter counts down to 0, unless the appropriate NORELOAD bit is set in the CFG register

Table 181. SCT match reload registers 0 to 9 (MATCHREL[0:9], offset 0x200 (MATCHREL0) to 0x224 (MATCHREL9)) bit description (REGMODEn bit: 0)

Bit	Symbol	Access	Reset value	Description
15:0	RELOADn_L	-	0x0	When UNIFY: 0, specifies the 16-bit value to be loaded into the MATCHn_L register. When UNIFY: 1, specifies the lower 16 bits of the 32-bit value to be loaded into the MATCHn register
31:16	RELOADn_H	-	0x0	When UNIFY: 0, specifies the 16-bit to be loaded into the MATCHn_H register When UNIFY: 1, specifies the upper 16 bits of the 32-bit value to be loaded into the MATCHn register

13.6.23 SCT capture control registers 0 to 9 (REGMODEn bit: 1)

If UNIFY: 1 in the CONFIG register, only the _L bits are used.

If UNIFY: 0 in the CONFIG register, this register can be written to as two registers CAPCTRLn_L and CAPCTRLn_H. Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Based on a selected event, the capture registers can be loaded with the current counter value when the event occurs.

Each capture control register (L, H, or unified 32-bit) controls which events cause the load of corresponding capture register from the counter.

Table 182. SCT capture control registers 0 to 9 (CAPCTRL[0:9], offset 0x200 (CAPCTRL0) to 0x224 (CAPCTRL9)) bit description (REGMODEn bit: 1)

Bit	Symbol	Access	Reset value	Description
15:0	CAPCONn_L	-	0x0	If bit m is one, event m causes the CAPn_L (UNIFY: 0) or the CAPn (UNIFY: 1) register to be loaded (event 0: bit 0, event 1: bit 1, …). The number of bits: number of match/captures supported by this SCT
31:16	CAPCONn_H	-	0x0	If bit m is one, event m causes the CAPn_H (UNIFY: 0) register to be loaded (event 0: bit 16, event 1: bit 17, …). The number of bits: number of match/captures supported by this SCT

13.6.24 SCT event enable registers 0 to 9

Each event can be enabled in some contexts (or states) and disabled in others. Each event defined in the EV_CTRL register has one associated event enable register that can enable or disable the event for each available state.

Each event has one associated SCT event state mask register that allow this event to happen in one or more states of the counter selected by the HEVENT bit in the corresponding EVn_CTRL register.

An event n is disabled when its EVn_STATE register contains all zeros, since it is masked regardless of the current state.

In simple applications that do not use states, write 0x01 to this register to enable each event in exactly one state. Since the state doesn't change (that is, the state variable always remains at its reset value of 0), writing 0x01 permanently enables this event.

Table 183. SCT event state mask registers 0 to 9 (EV[0:9]_STATE, offsets 0x300 (EV0_STATE) to 0x348 (EV9_STATE)) bit description

Bit	Symbol	Access	Reset value	Description
15:0	STATEMSKn	-	0	If bit m is one, event n happens in state m of the counter selected by the HEVENT bit (n: event number, m: state number; state 0: bit 0, state 1= bit 1, …). The number of bits: number of states supported by this SCT.
31:16	RESERVED	-	-	-

13.6.25 SCT event control registers 0 to 9

This register defines the conditions for an event to occur based on the counter values or input and output states.Once the event is configured, it can be selected to trigger multiple actions (for example stop the counter and toggle an output) unless the event is blocked in the current state of the SCT or the counter is halted. To block a particular event from occurring, use the EV_STATE register. To block all events for a given counter, set the HALT bit in the CTRL register or select an event to halt the counter.

An event can be programmed to occur based on a selected input or output edge or level and/or based on its counter value matching a selected match register. In bidirectional mode, events can also be enabled based on the direction of count.

When the UNIFY bit is 0, each event is associated with a particular counter by the HEVENT bit in its event control register. An event is permanently disabled when its event state mask register contains all 0s.

Each event can modify its counter STATE value. If more than one event associated with the same counter occurs in a given clock cycle, only the state change specified for the highest numbered event among them takes place. Other actions dictated by any simultaneously occurring events all take place.

Table 184. SCT event control register 0 to 9 (EV[0:9]_CTRL, offset 0x304 (EV0_CTRL) to 0x34C (EV9_CTRL)) bit description

Bit	Symbol	Acess	Reset value	Description
3:0	MATCHSEL	-	0x0	Selects the match register associated with this event (if any). A match can occur only when the counter selected by the HEVENT bit is running
4	HEVENT	-	0x0	Select L/H counter. Do not set this bit if UNIFY: 1
		0		Selects the L state and the L match register selected by MATCHSEL
		1		Selects the H state and the H match register selected by MATCHSEL
5	OUTSEL		0x0	Input/output select
		0		Selects the inputs selected by IOSEL
		1		Selects the outputs selected by IOSEL
9:6	IOSEL	-	0x0	Selects the input or output signal number associated with this event (if any). Do not select an input in this register if CKMODE is 1x. In this case the clock input is an implicit ingredient of every event
11:10	IOCOND	-	0x0	Selects the I/O condition for event n. (The detection of edges on outputs lag the conditions that switch the outputs by one SCT clock). In order to guarantee proper edge/state detection, an input must have a minimum pulse width of at least one SCT clock period
		0x0		LOW
		0x1		Rise
		0x2		Fall
		0x3		HIGH
13:12	COMBMODE	-	0x0	Selects how the specified match and I/O condition are used and combined
		0x0		OR. The event occurs when either the specified match or I/O condition occurs
		0x1		MATCH. Uses the specified match only
		0x2		IO. Uses the specified I/O condition only.
		0x3		AND. The event occurs when the specified match and I/O condition occur simultaneously
14	STATELD	-	0x0	This bit controls how the STATEV value modifies the state selected by HEVENT when this event is the highest-numbered event occurring for that state
		0		STATEV value is added into STATE (the carry-out is ignored)
		1		STATEV value is loaded into STATE
19:15	STATEV	-	0x0	This value is loaded into or added to the state selected by HEVENT, depending on STATELD, when this event is the highest-numbered event occurring for that state. If STATELD and STATEV are both zero, there is no change to the STATE value

Table 184. SCT event control register 0 to 9 (EV[0:9]_CTRL, offset 0x304 (EV0_CTRL) to 0x34C (EV9_CTRL)) bit description *…continued*

Bit	Symbol	Acess	Reset value	Description
20	MATCHMEM	-	0x0	If this bit is one and the COMBMODE field specifies a match component to the triggering of this event, then a match is considered to be active whenever the counter value is greater than or equal to the value specified in the match register when counting up, less than or equal to the match value when counting down If this bit is zero, a match is only be active during the cycle when the counter is equal to the match value
22:21	DIRECTION	-	0x0	Direction qualifier for event generation. This field only applies when the counters are operating in BIDIR mode. If BIDIR: 0, the SCT ignores this field. Value 0x3 is reserved
		0x0		Direction independent. This event is triggered regardless of the count direction
		0x1		Counting up. This event is triggered only during up-counting when BIDIR: 1
		0x2		Counting down. This event is triggered only during down-counting when BIDIR: 1
31:23	RESERVED	-	-	-

13.6.26 SCT output set registers 0 to 7

Based on a selected event, each SCT output can be set.

There is one output set register for each SCT output which selects which events can set that output. Each bit of an output set register is associated with a different event (bit 0 with event 0, etc.). A selected event can set or clear the output depending on the setting of SETCLRn field in the OUTPUTDIRCTRL register. To define the actual event that sets the output (a match, an I/O pin toggle, etc.), see the EVn_CTRL register.

Remark: If the SCTimer/PWM is operating as two 16-bit counters, events can only modify the state of the outputs when neither counter is halted. This is true regardless of what triggered the event.

Table 185. SCT output set register (OUT[0:9]_SET, offset 0x500 (OUT0_SET) to 0x538 (OUT7_SET) bit description

Bit	Symbol	Access	Reset value	Description
15:0	SET	-	0x0	A 1 in bit m selects event m to set output n (or clear it if SETCLRn: 0x1 or 0x2) output 0: bit 0, output 1: bit 1, … The number of bits: number of events supported by this SCT When the counter is used in bi-directional mode, it is possible to reverse the action specified by the output set and clear registers when counting down, See the OUTPUTCTRL register
31:16	RESERVED	-	-	-

13.6.27 SCT output clear registers 0 to 7

Based on a selected event, each SCT output can be cleared.

There is one register for each SCT output which selects which events can clear that output. Each bit of an output clear register is associated with a different event (bit 0 with event 0, etc.). A selected event can clear or set the output depending on the setting of the SETCLRn field in the OUTPUTDIRCTRL register. To define the actual event that clears the output (a match, an I/O pin toggle, etc.), see the EVn_CTRL register.

Remark: If the SCTimer/PWM is operating as two 16-bit counters, events can only modify the state of the outputs when neither counter is halted. This is true regardless of what triggered the event.

Table 186. SCT output clear register (OUT[0:9]_CLR, offset 0x504 (OUT0_CLR) to 0x53C (OUT7_CLR)) bit description

Bit	Symbol	Access	Reset value	Description
15:0	CLR	-	0x0	A 1 in bit m selects event m to clear output n (or set it if SETCLRn: 0x1 or 0x2) event 0: bit 0, event 1: bit 1, … The number of bits: number of events supported by this SCT When the counter is used in bi-directional mode, it is possible to reverse the action specified by the output set and clear registers when counting down, See the OUTPUTCTRL register
31:16	RESERVED	-	-	-

13.7 Functional description

13.7.1 Match logic

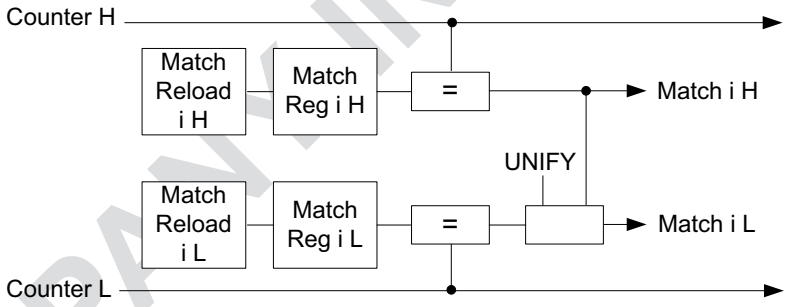


Fig 42. Match logic

13.7.2 Capture logic

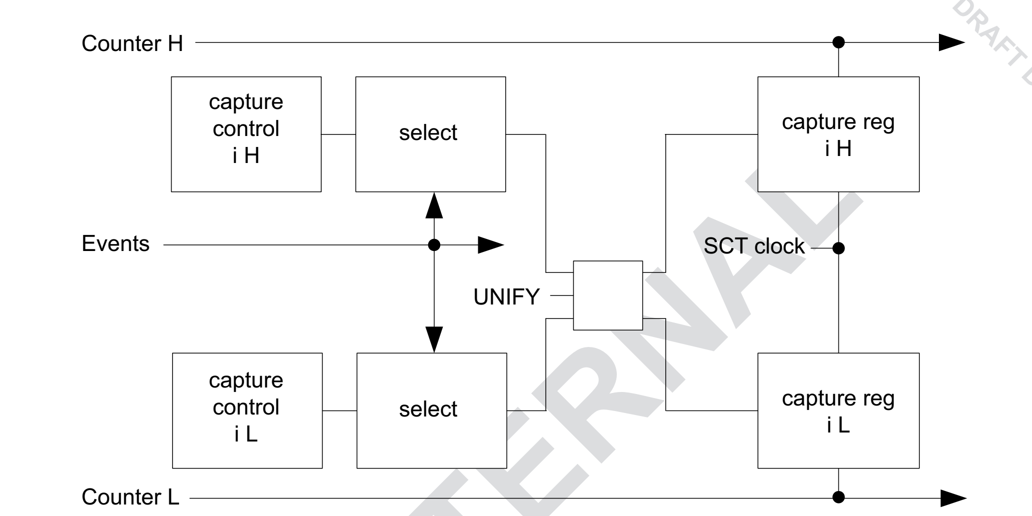


Fig 43. Capture logic

13.7.3 Event selection

State variables allow control of SCT across more than one cycle of the counter. Counter matches, input/output edges, and state values are combined into a set of general-purpose events that can switch outputs, request interrupts, and change state values.

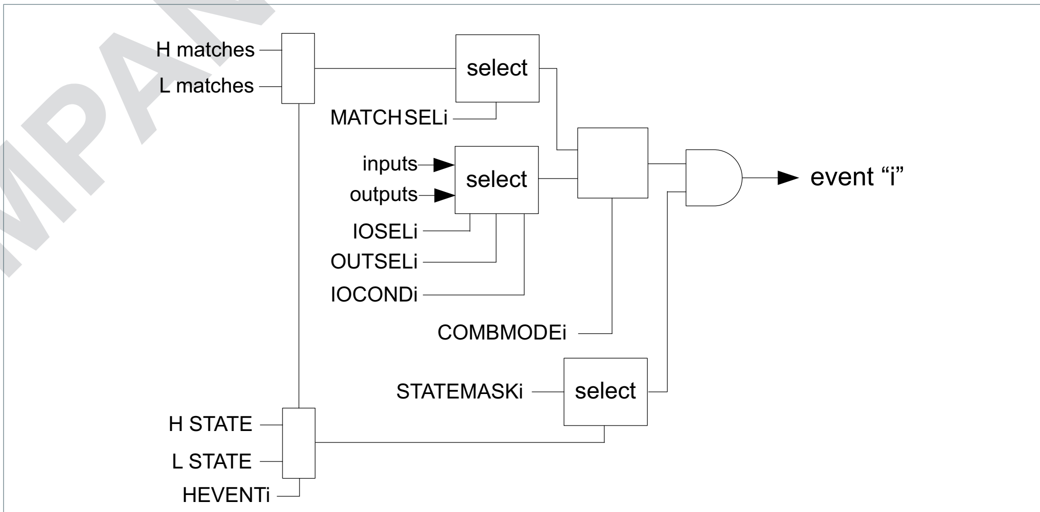


Fig 44. Event selection

13.7.4 Output generation

Figure 45 shows one output slice of the SCT.

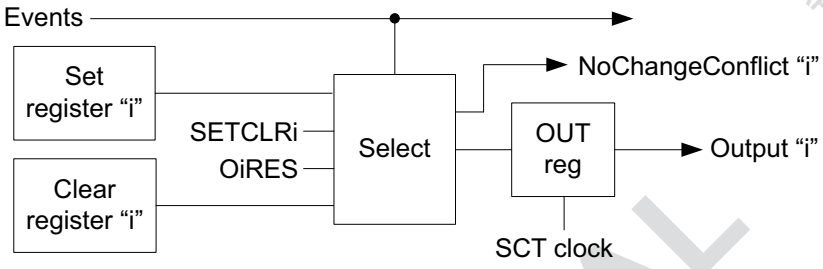


Fig 45. Output slice i

13.7.5 State logic

The SCT can be configured as a timer/counter with multiple programmable states. The states are user-defined through the events that can be captured in each particular state. In a multi-state SCT, the SCT can change from one state to another state when a user-defined event triggers a state change. The state change is triggered through each event’s EV_CTRL register in one of the following ways:

- The event can increment the current state number by a new value
- The event can write a new state value

If an event increments the state number beyond the number of available states, the SCT enters a locked state in which all further events are ignored while the counter is still running. Software must interfere to change out of this state.

Software can capture the counter value (and potentially create an interrupt and write to all outputs) when the event moving the SCT into a locked state occurs.Later, while the SCT is in locked state, software can read counter again to record the time passed since locking event. It can also read the state variable to obtain the current state number.

If the SCT registers an event that forces an abort, putting the SCT in a locked state can be a safe way to record the time that has passed since the abort event while no new events are allowed to occur. Since multiple states (any state number between the maximum implemented state and 31) are locked states, multiple abort or error events can be defined each incrementing the state number by a different value.

13.7.6 Interrupt generation

The SCT generates one interrupt to the NVIC.

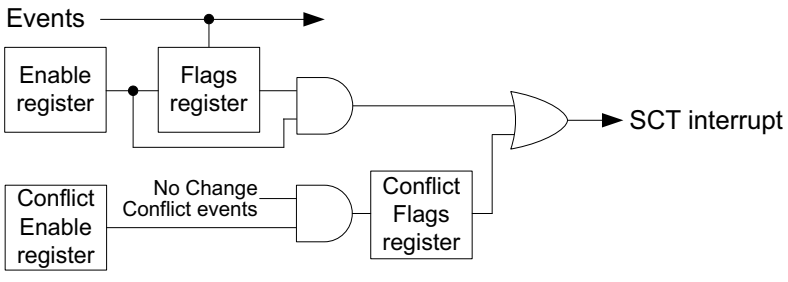


Fig 46. SCT interrupt generation

13.7.7 Clearing the prescaler

When enabled by a non-zero PRE field in the control register, the prescaler acts as a clock divider for the counter, like a fractional part of the counter value. The prescaler is cleared whenever the counter is cleared or loaded for any of the following reasons:

- Hardware reset
- Software writing to the counter register
- Software writing a 1 to the CLRCTR bit in the control register
- An event selected by a 1 in the counter limit register when BIDIR: 0

When BIDIR is 0, a limit event caused by an I/O signal can clear a non-zero prescaler. However, a limit event caused by a Match only clears a non-zero prescaler in one special case as described Section 13.7.8.

A limit event when BIDIR is 1 does not clear the prescaler. Rather it clears the DOWN bit in the control register, and decrements the counter on the same clock if the counter is enabled in that clock.

13.7.8 Match vs. I/O events

Counter operation is complicated by the prescaler and by clock mode 01 in which the SCT clock is the bus clock. However, the prescaler and counter are enabled to count only when a selected edge is detected on a clock input.

- The prescaler is enabled when the clock mode is not 01, or when the input edge selected by the CLKSEL field is detected
- The counter is enabled when the prescaler is enabled, and (PRELIM: 0 or the prescaler is equal to the value in PRELIM)

An I/O component of an event can occur in any SCT clock when its counter HALT bit is 0. In general, a Match component of an event can only occur in a UT clock when its counter HALT and STOP bits are both 0 and the counter is enabled.

Table 187 shows when the various kinds of events can occur.

Table 187. Event conditions

COMBMODE	IOMODE	Event can occur on clock:
IO	any	event can occur whenever HALT: 0 (type A)
MATCH	any	event can occur when HALT: 0 and STOP: 0 and the counter is enabled (type C)
OR	any	From the IO component: Event can occur whenever HALT: 0 (A) From the match component: Event can occur when HALT: 0 and STOP: 0 and the counter is enabled (C)
AND	LOW or HIGH	event can occur when HALT: 0 and STOP: 0 and the counter is enabled (C)
AND	RISE or FALL	event can occur whenever HALT: 0 (A)

13.7.9 SCT operation

In its simplest, single-state configuration, the SCT operates as an event controlled one- or bidirectional counter. Events can be configured to be counter match events, an input or output level, transitions on an input or output pin, or a combination of match and input/output behavior. In response to an event, the SCT output or outputs can transition,

or the SCT can perform other actions such as creating an interrupt or starting, stopping, or resetting the counter. Multiple simultaneous actions are allowed for each event. Furthermore, any number of events can trigger one specific action of the SCT.

An action or multiple actions of the SCT uniquely define an event. A state is defined by which events are enabled to trigger an SCT action or actions in any stage of the counter. Events not selected for this state are ignored.

In a multi-state configuration, states change in response to events. A state change is an additional action that the SCT can perform when the event occurs. When an event is configured to change the state, the new state defines a new set of events resulting in different actions of the SCT. Through multiple cycles of the counter, events can change the state multiple times and thus create a large variety of event controlled transitions on the SCT outputs and/or interrupts.

Once configured, the SCT can run continuously without software intervention and can generate multiple output patterns entirely under the control of events.

- To configure the SCT, see Section 13.7.10
- To start, run, and stop the SCT, see Section 13.7.11
- To configure the SCT as simple event controlled counter/timer, see Section 13.7.12

13.7.10 Configure the SCT

To set up the SCT for multiple events and states, perform the following configuration steps:

13.7.10.1 Configure the counter

- Configure the L and H counters in the CONFIG register by selecting two independent 16-bit counters (L counter and H counter) or one combined 32-bit counter in the UNIFY field
- Select the SCT clock source in the CONFIG register (fields CLKMODE and CLKSEL) from any of the inputs or an internal clock

13.7.10.2 Configure the match and capture registers

- Select how many match and capture registers the application uses (not more than what is available on this device):
  - In the REGMODE register, select for each of the match/capture register pairs whether the register is used as a match register or capture register
- Define match conditions for each match register selected:
  - Each match register MATCH sets one match value, if a 32-bit counter is used, or two match values, if the L and H 16-bit counters are used
  - Each match reload register MATCHRELOAD sets a reload value that is loaded into the match register when the counter reaches a limit condition or the value 0

13.7.10.3 Configure events and event responses

- Define when each event can occur in the following way in the EVn_CTRL registers (up to 6, one register per event):

- Select whether the event occurs on an input or output changing, on an input or output level, a match condition of the counter, or a combination of match and input/output conditions in field COMBMODE
- For a match condition:

  Select the match register that contains the match condition for the event to occur. Enter the number of the selected match register in field MATCHSEL.

  If using L and H counters, define whether the event occurs on matching the L or the H counter in field HEVENT.
- For an SCT input or output level or transition:

  Select the input number or the output number that is associated with this event in fields IOSEL and OUTSEL.

  Define how the selected input or output triggers the event (edge or level sensitive) in field IOCOND.

- Define what the effect of each event is on the SCT outputs in the OUTn_SET or OUTn_CLR registers (up to the maximum number of outputs on this device, one register per output):
  - For each SCT output, select which events set or clear this output. More than one event can change the output, and each event can change multiple outputs.
- Define how each event affects the counter:
  - Set the corresponding event bit in the LIMIT register for the event to set an upper limit for the counter

    When a limit event occurs in unidirectional mode, the counter is cleared to zero and begins counting up on the next clock edge.

    When a limit event occurs in bidirectional mode, the counter begins to count down from the current value on the next clock edge.
  - Set the corresponding event bit in the HALT register for the event to halt the counter. If the counter is halted, it stops counting and no new events can occur. The counter operation can only be restored by clearing the HALT_L and/or the HALT_H bits in the CTRL register
  - Set the corresponding event bit in the STOP register for the event to stop the counter. If the counter is stopped, it stops counting. However, an event that is configured as a transition on an input/output can restart the counter
  - Set the corresponding event bit in the START register for the event to restart the counting. Only events that are defined by an input changing can be used to restart the counter
- Define which events contribute to the SCT interrupt:
  - Set the corresponding event bit in the EVEN and the EVFLAG registers to enable the event to contribute to the SCT interrupt

13.7.10.4 Configure multiple states

- In the EVn_STATE register for each event (up to the maximum number of events on this device, one register per event), select the state or states (up to 2) in which this event is allowed to occur. Each state can be selected for more than one event
- Determine how the event affects the system state:

In the EVn_CTRL registers (up to the maximum number of events on this device, one register per event), set the new state value in the STATEV field for this event. If the event is the highest numbered in the current state, this value is either added to the existing state value or replaces the existing state value, depending on the field STATELD.

Remark: If there are higher numbered events in the current state, this event cannot change the state.

If the STATEV and STATELD values are set to zero, the state does not change.

13.7.10.5 Miscellaneous options

- There are a certain (selectable) number of capture registers. Each capture register can be programmed to capture the counter contents when one or more events occur
- If the counter is in bidirectional mode, the effect of set and clear of an output can be made to depend on whether the counter is counting up or down by writing to the OUTPUTDIRCTRL register

13.7.11 Run the SCT

- Configure the SCT (see Section 13.7.10)
- Write to the STATE register to define the initial state. By default the initial state is state 0
- To start the SCT, write to the CTRL register:
  - Clear the counters
  - Clear or set the STOP_L and/or STOP_H bits

    Remark: The counter starts counting once the STOP bit is cleared as well. If the STOP bit is set, the SCT waits instead for an event to occur that is configured to start the counter.
  - For each counter, select unidirectional or bidirectional counting mode (field BIDIR_L and/or BIDIR_H)
  - Select the prescale factor for the counter clock (CTRL register)
  - Clear the HALT_L and/or HALT_H bit. By default, the counters are halted and no events can occur
- To stop the counters by software at any time, stop or halt the counter (write to STOP_L and/or STOP_H bits or HALT_L and/or HALT_H bits in the CTRL register)
  - When the counters are stopped, both an event configured to clear the STOP bit or software writing a zero to the STOP bit can start the counter again
  - When the counter are halted, only a software write to clear the HALT bit can start the counter again. No events can occur
  - When the counters are halted, software can set any SCT output HIGH or LOW directly by writing to the OUT register

The current state can be read at any time by reading the STATE register.

To change the current state by software (that is independently of any event occurring), set the HALT bit and write to the STATE register to change the state value. Writing to the STATE register is only allowed when the counter is halted (the HALT_L and/or HALT_H bits are set) and no events can occur.

13.7.12 Configure the SCT without using states

The SCT can be used as standard counter/timer with external capture inputs and match outputs without using the state logic. To operate the SCT without states, configure the SCT as follows:

- Write zero to the STATE register (zero is the default)
- Write zero to the STATELD and STATEV fields in the EVCTRL registers for each event
- Write 0x1 to the EVn_STATE register of each event. Writing 0x1 enables the event

  In effect, the event is allowed to occur in a single state which never changes while the counter is running.

13.7.13 SCT PWM Example

Figure 47 shows a simple application of the SCT using two sets of match events (EV0/1 and EV3/4) to set/clear SCT output 0. The timer is automatically reset whenever it reaches the MAT0 match value.

In the initial state 0, match event EV0 sets output 0 to HIGH and match event EV1 clears output 0. The SCT input 0 is monitored: If input0 is found LOW by the next time the timer is reset(EV2), the state is changed to state 1, and EV3/4 are enabled, which create the same output but triggered by different match values. If input 0 is found HIGH by the next time the timer is reset, the associated event (EV5) causes the state to change back to state 0 where the events EV0 and EV1 are enabled.

The example uses the following SCT configuration:

- One input
- One output
- Five match registers
- Six events and match 0 used with auto-limit function
- Two states

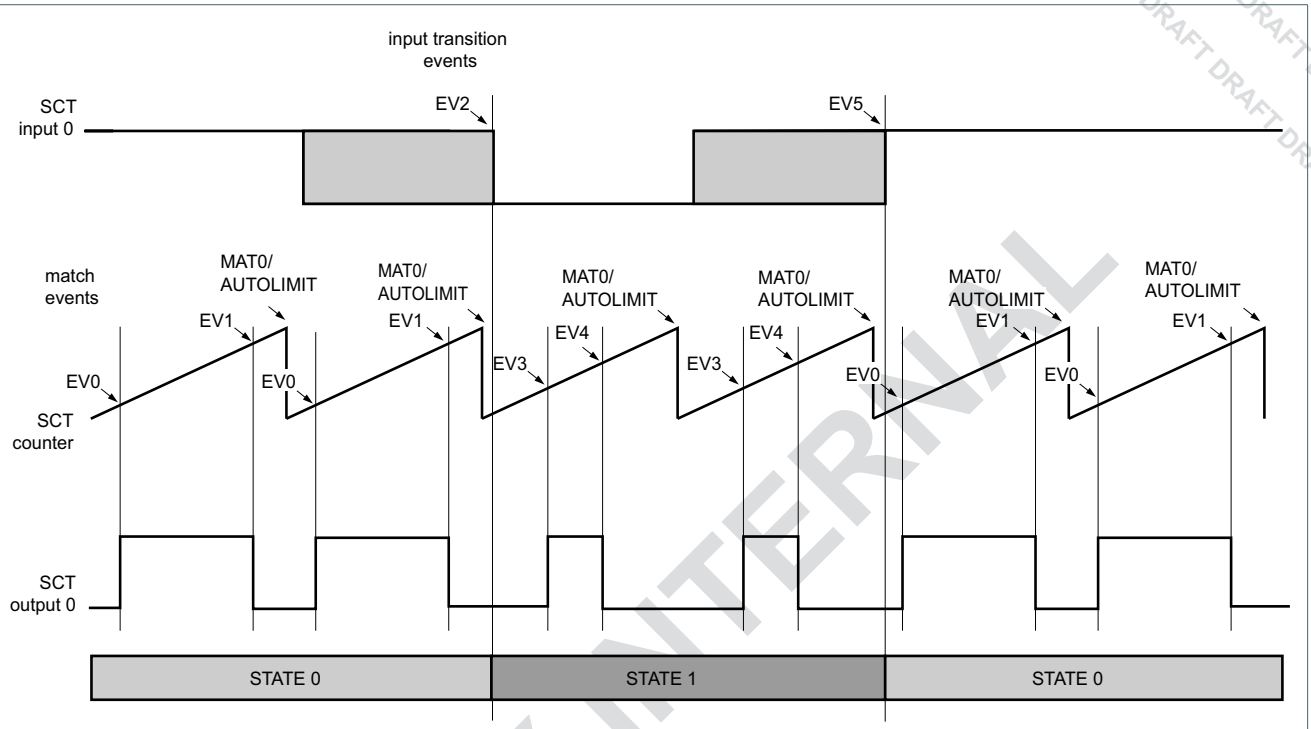


Fig 47. SCT configuration example

This application of the SCT uses the following configuration (all register values not listed in Table 188 are set to their default values):

Table 188. SCT configuration example

Configuration	Registers	Setting
Counter	CONFIG	Uses one counter (UNIFY: 1)
	CONFIG	Enable the autolimit for MAT0. (AUTOLIMIT: 1)
	CTRL	Uses unidirectional counter (BIDIR_L: 0)
Clock base	CONFIG	Uses default values for clock configuration
Match/Capture registers	REGMODE	Configure one match register for each match event by setting REGMODE_L bits 0,1, 2, 3, 4 to 0. This is the default.
Define match values	MATCH 0/1/2/3/4	Set a match value MATCH0/1/2/4/5_L in each register. The match 0 register serves as an automatic limit event that resets the counter. without using an event. To enable the automatic limit, set the AUTOLIMIT bit in the CONFIG register.
Define match reload values	MATCHREL 0/1/2/3/4	Set a match reload value RELOAD0/1/2/3/4_L in each register (same as the match value in this example).
Define when event 0 occurs	EV0_CTRL	• Set COMBMODE: 0x1. Event 0 uses match condition only • Set MATCHSEL: 1. Select match value of match register 1. The match value of MAT1 is associated with event 0
Define when event 1 occurs	EV1_CTRL	• Set COMBMODE: 0x1. Event 1 uses match condition only • Set MATCHSEL: 2 Select match value of match register 2. The match value of MAT2 is associated with event 1

Table 188. SCT configuration example *…continued*

Configuration	Registers	Setting
Define when event 2 occurs	EV2_CTRL	• Set COMBMODE: 0x3. Event 2 uses match condition and I/O condition • Set IOSEL: 0. Select input 0 • Set IOCOND: 0x0. Input 0 is LOW • Set MATCHSEL: 0. Chooses match register 0 to qualify the event
Define how event 2 changes the state	EV2_CTRL	Set STATEV bits to 1 and the STATED bit to 1. Event 2 changes the state to state 1
Define when event 3 occurs	EV3_CTRL	• Set COMBMODE: 0x1. Event 3 uses match condition only • Set MATCHSEL: 0x3. Select match value of match register 3. The match value of MAT3 is associated with event 3
Define when event 4 occurs	EV4_CTRL	• Set COMBMODE: 0x1. Event 4 uses match condition only • Set MATCHSEL: 0x4. Select match value of match register 4.The match value of MAT4 is associated with event 4
Define when event 5 occurs	EV5_CTRL	• Set COMBMODE: 0x3. Event 5 uses match condition and I/O condition • Set IOSEL: 0. Select input 0 • Set IOCOND: 0x3. Input 0 is HIGH • Set MATCHSEL: 0. Chooses match register 0 to qualify the event
Define how event 5 changes the state	EV5_CTRL	Set STATEV bits to 0 and the STATED bit to 1. Event 5 changes the state to state 0
Define by which events output 0 is set	OUT0_SET	Set SET0 bits 0 (for event 0) and 3 (for event 3) to one to set the output when these events 0 and 3 occur
Define by which events output 0 is cleared	OUT0_CLR	Set CLR0 bits 1 (for events 1) and 4 (for event 4) to one to clear the output when events 1 and 4 occur
Configure states in which event 0 is enabled	EV0_STATE	Set STATEMSK0 bit 0 to 1. Set all other bits to 0. Event 0 is enabled in state 0
Configure states in which event 1 is enabled	EV1_STATE	Set STATEMSK1 bit 0 to 1. Set all other bits to 0. Event 1 is enabled in state 0
Configure states in which event 2 is enabled	EV2_STATE	Set STATEMSK2 bit 0 to 1. Set all other bits to 0. Event 2 is enabled in state 0
Configure states in which event 3 is enabled	EV3_STATE	Set STATEMSK3 bit 1 to 1. Set all other bits to 0. Event 3 is enabled in state 1
Configure states in which event 4 is enabled	EV4_STATE	Set STATEMSK4 bit 1 to 1. Set all other bits to 0. Event 4 is enabled in state 1
Configure states in which event 5 is enabled	EV5_STATE	Set STATEMSK5 bit 1 to 1. Set all other bits to 0. Event 5 is enabled in state 1

14. Standard counter/timers (CTIMER 0 to 3)

14.1 Introduction

These four standard timers are available on all QN908x parts.

14.2 Basic configuration

- Set the appropriate bits to enable clocks to timers that will be used: write CLK_TIM0_EN, CLK_TIM1_EN, CLK_TIM2_EN or CLK_TIM3_EN in CLK_EN register to 1 to enable each timer clock
- Reset each timer using RST_SW_SET register by write SET_TIM0_RST, SET_TIM1_RST, SET_TIM2_RST or SET_TIM3_RST to 1
- Clear the timer reset using the RST_SW_CLR register by write CLR_TIM0_RST, CLR_TIM1_RST, CLR_TIM2_RST or CLR_TIM3_RST to 1
- Pins: Select timer pins and pin modes as needed through the relevant IOCON registers (Section 8)
- Interrupts: See register MCR (Table 198) and CCR (Table 200) for match and capture events. Interrupts are enabled in the NVIC using the appropriate Interrupt Set Enable register. The CTIMER0_INT to CTIMER3_INT is at the system interrupt position 26 to 29, see Table 49
- DMA: Some timer match conditions can be used to generate timed DMA requests, see Table 131

14.3 Features

- Each is a 32-bit counter/timer with a programmable 32-bit prescaler. All of the timers include external capture and match pin connections
- Counter or timer operation
- For each timer with pin connections, up to 3 32-bit capture channels that can take a snapshot of the timer value when an input signal transitions. 2 captures can be configured from external pins in the PIO_FUNC_CFG<x>, the other capture is connected to 32 k clock (either 32k crystal or 32k RCO) directly. A capture event may also optionally generate an interrupt
- The timer and prescaler may be configured to be cleared on a designated capture event. This feature permits easy pulse-width measurement by clearing the timer on the leading edge of an input pulse and capturing the timer value on the trailing edge
- Four 32-bit match registers that allow:
  - Continuous operation with optional interrupt generation on match
  - Stop timer on match with optional interrupt generation
  - Reset timer on match with optional interrupt generation
- For each timer with pin connections, up to 4 external outputs corresponding to match registers with the following capabilities:
  - Set LOW on match
  - Set HIGH on match
  - Toggle on match

– Do nothing on match

Remark: For each timer, only three of the four outputs can be output to some pins by configuring PIO_FUNC_CFG<x> registers. The other output can be input as some internal module DAC, ADC, DMA.

- PWM: For each timer with pin connections, up to 3 match outputs can be used as single edge controlled PWM outputs

14.4 General description

Each counter/timer is designed to count cycles of the APB bus clock or an externally supplied clock. It can optionally generate interrupts or perform other actions at specified timer values based on four match registers. Each counter/timer also includes one capture input to trap the timer value when an input signal transitions, optionally generating an interrupt.

In PWM mode, three match registers can be used to provide a single-edge controlled PWM output on the match output pins. One match register is used to control the PWM cycle length.

14.4.1 Capture inputs

The capture signal can be configured to load the capture register with the value in the counter/timer and optionally generate an interrupt. The capture signal is generated by one of the pins with a capture function. Each capture signal is connected to one capture channel of the timer.

The counter/timer block can select a capture signal as a clock source instead of the APB bus clock. For more details see Section 14.6.11.

14.4.2 Match outputs

When a match register equals the Timer Counter (TC), the corresponding match output can either toggle, go LOW, go HIGH, or do nothing. The External Match Register (EMR) and the PWM Control Register (PWMCON) control the functionality of this output.

14.4.3 Applications

- Interval timer for counting internal events
- Pulse width modulator via match outputs
- Pulse width demodulator via capture input
- Free running timer

14.4.4 Architecture

The block diagram for the timers is shown in Figure 48.

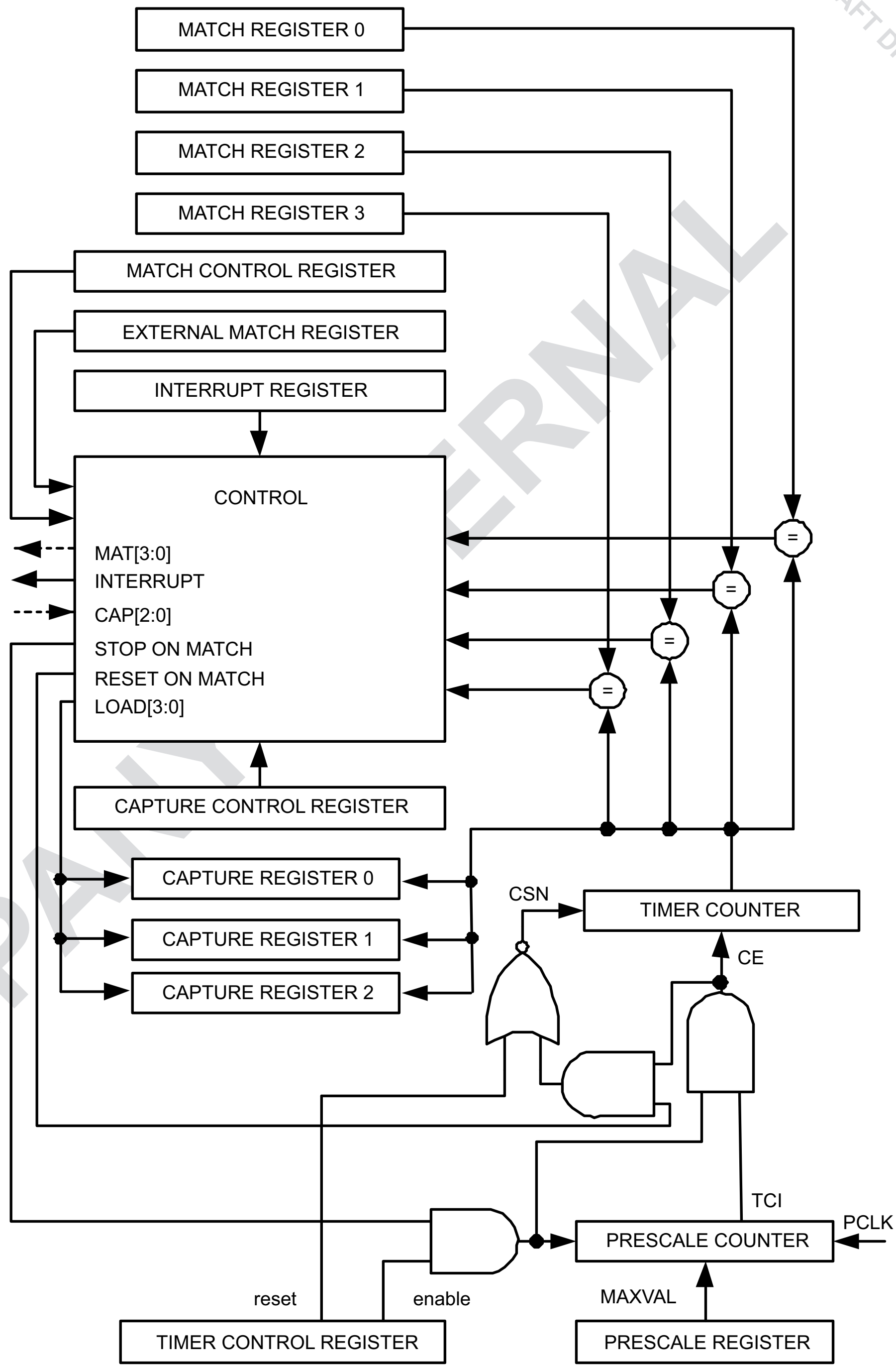


Fig 48. 32-bit counter/timer block diagram

14.5 Pin description

Table 189 gives a brief summary of each of the timer/counter related pins.

Table 189. Timer/Counter pin description

Pin	Type	Description
CTIMER0_CAP2:0 CTIMER1_CAP2:0 CTIMER2_CAP2:0 CTIMER3_CAP2:0	input	Capture Signals- A transition on a capture pin can be configured to load one of the capture registers with the value in the timer counter and optionally generate an interrupt. Capture functionality can be selected from a number of pins. When more than one pin is selected for a capture input on a single timer channel, the pin with the lowest Port number is used Timer/Counter block can select a capture signal as a clock source instead of the APB bus clock. For more details, see Section 14.6.11
CTIMER0_MAT3:0 CTIMER1_MAT3:0 CTIMER2_MAT3:0 CTIMER3_MAT3:0	output	External match output - When a match register (MR3:0) equals the timer counter (TC), this output can either toggle, go low, go high, or do nothing. The External Match Register (EMR) controls the functionality of this output. Match Output functionality can be selected on a number of pins in parallel

14.5.1 Multiple CAP and OUT pins

For each timer, two CAP inputs and three outputs can be configured from/to external pins by configuring PIO_FUNC_CFG<x> registers based on pin MUX table Table 95.

Table 190. CTIMER[0:3] pin description (inputs)

Function	Type	Connect to	Reference
CTIMER0_CAP[0:1]	external from pin	PA00, PA01	Table 95
CTIMER1_CAP[0:1]		PA07, PA08	
CTIMER2_CAP[0:1]		PA14, PA15	
CTIMER3_CAP[0:1]		PA24, PA25	

Table 191. CTIMER[0:3] pin description (outputs)

Function	Type	Connect to	Reference
CTIMER0_MAT0	external to pin	PA02, PA04	Table 95
CTIMER0_MAT1		PA03, PA05	
CTIMER0_MAT2		PA06	
CTIMER1_MAT0		PA09, PA26	
CTIMER1_MAT1		PA10, PA27	
CTIMER1_MAT2		PA11	
CTIMER2_MAT0		PA16, PA20	
CTIMER2_MAT1		PA17, PA21	
CTIMER2_MAT2		PA18	
CTIMER3_MAT0		PA22, PA29	
CTIMER3_MAT1		PA23, PA30	
CTIMER3_MAT2		PA31	

14.6 Register description

Each timer/counter contains the registers shown in Table 192 ("reset value" refers to the data stored in used bits only; it does not include reserved bits content). More detailed descriptions follow.

Table 192. Register overview: CTIMER0/1/2/3 (register base addresses 0x4000 2000 (CTIMER0), 0x4000 3000 (CTIMER1), 0x4000 4000 (CTIMER2), 0x4005 0000 (CTIMER3))

Name	Access	Offset	Description	Reset value[1]	Section
IR	R/W	0x00	Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of four possible interrupt sources are pending.	0x0	Section 14.6.1
TCR	R/W	0x04	Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.	0x0	Section 14.6.2
TC	R/W	0x08	Timer Counter. The 32 bit TC is incremented every PR+1 cycles of the APB bus clock. The TC is controlled through the TCR.	0x0	Section 14.6.3
PR	R/W	0x0C	Prescale Register. When the Prescale Counter (PC) is equal to this value, the next clock increments the TC and clears the PC.	0x0	Section 14.6.4
PC	R/W	0x10	Prescale Counter. The 32 bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented and the PC is cleared. The PC is observable and controllable through the bus interface.	0x0	Section 14.6.5
MCR	R/W	0x14	Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.	0x0	Section 14.6.6
MR0	R/W	0x18	Match Register 0. MR0 can be enabled through the MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt every time MR0 matches the TC.	0x0	Section 14.6.7
MR1	R/W	0x1C	Match Register 1. See MR0 description.	0x0	Section 14.6.7
MR2	R/W	0x20	Match Register 2. See MR0 description.	0x0	Section 14.6.7
MR3	R/W	0x24	Match Register 3. See MR0 description.	0x0	Section 14.6.8
CCR	R/W	0x28	Capture Control Register. The CCR controls which edges of the capture inputs are used to load the Capture Registers and whether or not an interrupt is generated when a capture takes place.	0x0	Section 14.6.8
CR0	RO	0x2C	Capture Register 0. CR0 is loaded with the value of TC when there is an event on the CAPn.0 input.	0x0	Section 14.6.9
CR1	RO	0x30	Capture Register 1. See CR0 description.	0x0	Section 14.6.9
CR2	RO	0x34	Capture Register 2. See CR0 description.	0x0	Section 14.6.9
EMR	R/W	0x3C	External Match Register. The EMR controls the match function and the external match pins.	0x0	Section 14.6.10
CTCR	R/W	0x70	Count Control Register. The CTCR selects between Timer and Counter mode, and in Counter mode selects the signal and edge(s) for counting.	0x0	Section 14.6.11
PWMC	R/W	0x74	PWM Control Register. The PWMCON enables PWM mode for the external match pins.	0x0	Section 14.6.12

[1] Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

14.6.1 Interrupt register

The interrupt register consists of four bits for the match interrupts and four bits for the capture interrupts. If an interrupt is generated, then the corresponding bit in the IR will be high. Otherwise, the bit will be low. Writing a logic one to the corresponding IR bit will reset the interrupt. Writing a zero has no effect. The act of clearing an interrupt for a timer match also clears any corresponding DMA request. Writing a zero has no effect.

Table 193. Interrupt Register (IR, offset 0x000) bit description

Bit	Symbol	Description	Reset value
0	MR0INT	interrupt flag for match channel 0	0x0
1	MR1INT	interrupt flag for match channel 1	0x0
2	MR2INT	interrupt flag for match channel 2	0x0
3	MR3INT	interrupt flag for match channel 3	0x0
4	CR0INT	interrupt flag for capture channel 0 event	0x0
5	CR1INT	interrupt flag for capture channel 1 event	0x0
6	CR2INT	interrupt flag for capture channel 2 event	0x0
31:7	RESERVE D	read value is undefined, only zero should be written	-

14.6.2 Timer Control Register

The Timer Control Register (TCR) is used to control the operation of the Timer/Counter.

Table 194. Timer Control Register (TCR, offset 0x004) bit description

Bit	Symbol	Value	Description	Reset value
0	CEN		Counter enable	0x0
		0	Disabled.The counters are disabled	
		1	Enabled. The Timer Counter and Prescale Counter are enabled	
1	CRST		Counter reset	0x0
		0	Disabled. Do nothing	
		1	Enabled. The Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of the APB bus clock. The counters remain reset until TCR[1] is returned to zero	
31:2	RESERV ED	-	Read value is undefined, only zero should be written	NA

14.6.3 Timer counter registers

The 32-bit timer counter register is incremented when the prescale counter reaches its terminal count. Unless it is reset before reaching its upper limit, the timer counter will count up through the value 0xFFFF FFFF and then wrap back to the value 0x0000 0000. This event does not cause an interrupt, but a match register can be used to detect an overflow, if required.

Table 195. TC - Timer Counter registers (offset = 0x08) bit description

Bit	Symbol	Description	Reset value
31:0	TCVAL	timer counter value	0x0

14.6.4 Prescale register

The 32-bit prescale register specifies the maximum value for the prescale counter.

Table 196. PR - Timer prescale registers (offset = 0x00C) bit description

Bit	Symbol	Description	Reset value
31:0	PRVAL	prescale counter maximum value	0x0

14.6.5 Prescale counter register

The 32-bit prescale counter controls division of the APB bus clock by some constant value before it is applied to the timer counter. It allows control of the relationship of resolution of the timer versus the maximum time before the timer overflows. The prescale counter is incremented on every APB bus clock. When it reaches the value stored in the prescale register, the timer counter is incremented and the prescale counter is reset on the next APB bus clock. It causes the timer counter to increment on every APB bus clock when PR = 0, every 2 APB bus clocks when PR = 1, etc.

Table 197. PC - Timer prescale counter registers (offset = 0x010) bit description

Bit	Symbol	Description	Reset value
31:0	PCVAL	prescale counter value	0x0

14.6.6 Match control register

The match control register is used to control what operations are performed when one of the match registers matches the timer counter.

Table 198. MCR - Match Control Register (offset = 0x014) bit description

Bit	Symbol	Description	Reset Value
0	MR0I	Interrupt on MR0: an interrupt is generated when MR0 matches the value in the TC. 0: disabled. 1: enabled	0x0
1	MR0R	Reset on MR0: the TC will be reset if MR0 matches it. 0: disabled. 1: enabled	0x0
2	MR0S	Stop on MR0: the TC and PC will be stopped and TCR[0] will be set to 0 if MR0 matches the TC. 0: disabled. 1: enabled	0x0
3	MR1I	Interrupt on MR1: an interrupt is generated when MR1 matches the value in the TC. 0: disabled. 1: enabled. 0: disabled. 1: enabled	0x0
4	MR1R	Reset on MR1: the TC will be reset if MR1 matches it. 0: disabled. 1: enabled	0x0
5	MR1S	Stop on MR1: the TC and PC will be stopped and TCR[0] will be set to 0 if MR1 matches the TC. 0: disabled. 1: enabled	0x0
6	MR2I	Interrupt on MR2: an interrupt is generated when MR2 matches the value in the TC. 0: disabled. 1: enabled	0x0
7	MR2R	Reset on MR2: the TC will be reset if MR2 matches it. 0: disabled. 1: enabled	0x0
8	MR2S	Stop on MR2: the TC and PC will be stopped and TCR[0] will be set to 0 if MR2 matches the TC. 0: disabled. 1: enabled	0x0
9	MR3I	Interrupt on MR3: an interrupt is generated when MR3 matches the value in the TC. 0: disabled. 1: enabled	0x0

Table 198. MCR - Match Control Register (offset = 0x014) bit description

Bit	Symbol	Description	Reset Value
10	MR3R	Reset on MR3: the TC will be reset if MR3 matches it 0: disabled. 1: enabled	0x0
11	MR3S	Stop on MR3: the TC and PC will be stopped and TCR[0] will be set to 0 if MR3 matches the TC. 0: disabled. 1: enabled	0x0
31:12	RESERVED	Read value is undefined, only zero should be written	NA

14.6.7 Match registers

The match register values are continuously compared to the timer counter value. When the two values are equal, actions can be triggered automatically. The action possibilities are to generate an interrupt, reset the timer counter, or stop the timer. Actions are controlled by the settings in the MCR register.

Table 199. Timer match registers (MR[0:3], offset [0x018:0x024]) bit description

Bit	Symbol	Description	Reset value
31:0	MATCH	timer counter match value	0x0

14.6.8 Capture control register

The capture control register is used to control whether one of the four capture registers is loaded with the value in the timer counter when the capture event occurs, and whether an interrupt is generated by the capture event. Setting both the rising and falling bits at the same time is a valid configuration, resulting in a capture event for both edges. In the description below, "n" represents the timer number, 0 or 1.

Note: If counter mode is selected for a particular CAP input in the CTCR, the three bits for that input in this register should be programmed as 000, but capture and/or interrupt can be selected for the other three CAP inputs.

Table 200. CCR - Capture Control Register (offset = 0x028) bit description

Bit	Symbol	Description	Reset Value
0	CAP0RE	Rising edge of capture channel 0: a sequence of 0 then 1 causes CR0 to be loaded with the contents of TC. 0: disabled. 1: enabled	0x0
1	CAP0FE	Falling edge of capture channel 0: a sequence of 1 then 0 causes CR0 to be loaded with the contents of TC. 0: disabled. 1: enabled	0x0
2	CAP0I	Generate interrupt on channel 0 capture event: a CR0 load generates an interrupt	0x0
3	CAP1RE	Rising edge of capture channel 1: a sequence of 0 then 1 causes CR1 to be loaded with the contents of TC. 0: disabled. 1: enabled	0x0
4	CAP1FE	Falling edge of capture channel 1: a sequence of 1 then 0 causes CR1 to be loaded with the contents of TC. 0: disabled. 1: enabled	0x0
5	CAP1I	Generate interrupt on channel 1 capture event: a CR1 load generates an interrupt	0x0
6	CAP2RE	Rising edge of capture channel 2: a sequence of 0 then 1 causes CR2 to be loaded with the contents of TC. 0: disabled. 1: enabled	0x0

Table 200. CCR - Capture Control Register (offset = 0x028) bit description

Bit	Symbol	Description	Reset Value
7	CAP2FE	Falling edge of capture channel 2: a sequence of 1 then 0 causes CR2 to be loaded with the contents of TC. 0: disabled. 1: enabled	0x0
8	CAP2I	Generate interrupt on channel 2 capture event: a CR2 load generates an interrupt	0x0
31:9	RESERVED	Read value is undefined, only zero should be written	NA

14.6.9 Capture registers

Each capture register is associated with one capture channel and may be loaded with the counter/timer value when a specified event occurs on the signal defined for that capture channel. The signal could originate from an external pin or from an internal source. The settings in the Capture Control Register register determine whether the capture function is enabled, and whether a capture event happens on the rising edge of the associated signal, the falling edge, or on both edges.

Table 201. Timer capture registers (CR[0:2], offsets [0x02C:0x038]) bit description

Bit	Symbol	Description	Reset value
31:0	CAP	timer counter capture value	0x0

14.6.10 External match register

The external match register provides both control and status of the external match pins. In the descriptions below, “n” represents the timer number, 0 or 1, and “m” represent a match number, 0 through 3.

Match events for match 0 and match 1 in each timer can cause a DMA request; see Section 14.7.2.

If the match outputs are configured as PWM output, the function of the external match registers is determined by the PWM rules (Section 14.7.1).

Table 202. EMR - Timer External Match Registers (offset = 0x03C) bit description

Bit	Symbol	Value	Description	Reset value
0	EM0	-	External Match 0. This bit reflects the state of output MAT0, whether or not this output is connected to a pin. When a match occurs between the TC and MR0, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by EMR[5:4]. This bit is driven to the MAT pins if the match function is selected via IOCON. 0: LOW. 1: HIGH	0x0
1	EM1	-	External Match 1. This bit reflects the state of output MAT1, whether or not this output is connected to a pin. When a match occurs between the TC and MR1, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by EMR[7:6]. This bit is driven to the MAT pins if the match function is selected via IOCON. 0: LOW. 1: HIGH	0x0
2	EM2	-	External Match 2. This bit reflects the state of output MAT2, whether or not this output is connected to a pin. When a match occurs between the TC and MR2, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by EMR[9:8]. This bit is driven to the MAT pins if the match function is selected via IOCON. 0: LOW. 1: HIGH	0x0
3	EM3	-	External Match 3. This bit reflects the state of output MAT3, whether or not this output is connected to a pin. When a match occurs between the TC and MR3, this bit can either toggle, go LOW, go HIGH, or do nothing, as selected by MR[11:10]. This bit is driven to the MAT pins if the match function is selected via IOCON. 0: LOW. 1: HIGH	0x0

Table 202. EMR - Timer External Match Registers (offset = 0x03C) bit description

Bit	Symbol	Value	Description	Reset value
5:4	EMC0		External Match Control 0. Determines the functionality of External Match 0	0x0
		0x0	Do Nothing	
		0x1	Clear. Clear the corresponding External Match bit/output to 0 (MAT0 pin is LOW if pinned out)	
		0x2	Set. Set the corresponding External Match bit/output to 1 (MAT0 pin is HIGH if pinned out)	
		0x3	Toggle. Toggle the corresponding External Match bit/output	
7:6	EMC1		External Match Control 1. Determines the functionality of External Match 1	0x0
		0x0	Do Nothing	
		0x1	Clear. Clear the corresponding External Match bit/output to 0 (MAT1 pin is LOW if pinned out)	
		0x2	Set. Set the corresponding External Match bit/output to 1 (MAT1 pin is HIGH if pinned out)	
		0x3	Toggle. Toggle the corresponding External Match bit/output	
9:8	EMC2		External Match Control 2. Determines the functionality of External Match 2	0x0
		0x0	Do Nothing	
		0x1	Clear. Clear the corresponding External Match bit/output to 0 (MAT2 pin is LOW if pinned out)	
		0x2	Set. Set the corresponding External Match bit/output to 1 (MAT2 pin is HIGH if pinned out)	
		0x3	Toggle. Toggle the corresponding External Match bit/output	
11:10	EMC3		External Match Control 3. Determines the functionality of External Match 3	0x0
		0x0	Do Nothing	
		0x1	Clear. Clear the corresponding External Match bit/output to 0 (MAT3 pin is LOW if pinned out)	
		0x2	Set. Set the corresponding External Match bit/output to 1 (MAT3 pin is HIGH if pinned out)	
		0x3	Toggle. Toggle the corresponding External Match bit/output	
31:12	-	-	Reserved. Read value is undefined, only zero should be written	NA

14.6.11 Count control register

The Count Control Register (CTCR) is used to select between timer and counter mode, and in counter mode to select the pin and edge(s) for counting.

When counter mode is chosen as a mode of operation, the CAP input (selected by the CTCR bits 3:2) is sampled on every rising edge of the APB bus clock. After comparing two consecutive samples of this CAP input, one of the following four events is recognized - rising edge, falling edge, either of edges or no changes in the level of the selected CAP input. Only if the identified event occurs and the event corresponds to the one selected by bits 1:0 in the CTCR register, will the timer counter register be incremented.

Effective processing of the externally supplied clock to the counter has some limitations. Since two successive rising edges of the APB bus clock are used to identify only one edge on the CAP selected input, the frequency of the CAP input cannot exceed one half of the APB bus clock. Consequently, duration of the HIGH/LOWLOW levels on the same CAP input in this case cannot be shorter than 1/APB bus clock.

Bits 7:4 of this register are also used to enable and configure the capture-clears-timer feature. This feature allows for a designated edge on a particular CAP input to reset the timer to all zeros. Using this mechanism to clear the timer on the leading edge of an input pulse and performing a capture on the trailing edge, permits direct pulse-width measurement using a single capture input without the need to perform a subtraction operation in software.

Table 203. CTCR - Count Control Register (offset = 0x070) bit description

Bit	Symbol	Value	Description	Reset value
1:0	CTMODE		Counter/Timer mode This field selects which rising APB bus clock edges can increment timer's Prescale Counter (PC), or clear PC and increment Timer Counter (TC) Timer Mode: the TC is incremented when the prescale counter matches the prescale register	0x0
		0x0	Timer Mode. Incremented every rising APB bus clock edge	
		0x1	Counter Mode rising edge. TC is incremented on rising edges on the CAP input selected by bits 3:2	
		0x2	Counter Mode falling edge. TC is incremented on falling edges on the CAP input selected by bits 3:2	
		0x3	Counter Mode dual edge. TC is incremented on both edges on the CAP input selected by bits 3:2	
3:2	CINSEL		Count Input Select When bits 1:0 in this register are not 0, these bits select which CAP pin is sampled for clocking **Note:** If Counter mode is selected for a particular CAPn input in the CTCR, the 3 bits for that input in the Capture Control Register (CCR) must be programmed as 000. However, capture and/or interrupt can be selected for the other 3 CAPn inputs in the same timer	0x0
		0x0	Channel 0. CAPn.0 for CTIMERn	
		0x1	Channel 1. CAPn.1 for CTIMERn	
		0x2	Channel 2. CAPn.2 for CTIMERn	

Table 203. CTCR - Count Control Register (offset = 0x070) bit description

Bit	Symbol	Value	Description	Reset value
4	ENCC	-	Setting this bit to 1 enables clearing of the timer and the prescaler when the capture-edge event specified in bits 7:5 occurs	0x0
7:5	SELCC		Edge select. When bit 4 is 1, these bits select which capture input edge will cause the timer and prescaler to be cleared. These bits have no effect when bit 4 is low. Values 0x2 to 0x3 and 0x6 to 0x7 are reserved	0x0
		0x0	Channel 0 Rising Edge. Rising edge of the signal on capture channel 0 clears the timer (if bit 4 is set)	
		0x1	Channel 0 Falling Edge. Falling edge of the signal on capture channel 0 clears the timer (if bit 4 is set)	
		0x2	Channel 1 Rising Edge. Rising edge of the signal on capture channel 1 clears the timer (if bit 4 is set)	
		0x3	Channel 1 Falling Edge. Falling edge of the signal on capture channel 1 clears the timer (if bit 4 is set)	
		0x4	Channel 2 Rising Edge. Rising edge of the signal on capture channel 2 clears the timer (if bit 4 is set)	
		0x5	Channel 2 Falling Edge. Falling edge of the signal on capture channel 2 clears the timer (if bit 4 is set)	
31:8	RESERVED	-	Read value is undefined, only zero should be written	NA

14.6.12 PWM control register

The PWM control register is used to configure the match outputs as PWM outputs. Each match output can be independently set to perform either as PWM output or as match output whose function is controlled by the External Match Register (EMR).

For each timer, a maximum of three single edge controlled PWM outputs can be selected on the MATn.2:0 outputs. One additional match register determines the PWM cycle length. When a match occurs in any of the other match registers, the PWM output is set to HIGH. The timer is reset by the match register that is configured to set the PWM cycle length. When the timer is reset to zero, all currently HIGH match outputs configured as PWM outputs are cleared.

Table 204: PWMC - PWM Control register (offset = 0x074)) bit description

Bit	Symbol	Value	Description	Reset value
0	PWMEN0		PWM mode enable for channel 0	0x0
		0	Match. CTIMERn_MAT0 is controlled by EM0	
		1	PWM. PWM mode is enabled for CTIMERn_MAT0	
1	PWMEN1		PWM mode enable for channel 1	0x0
		0	Match. CTIMERn_MAT01 is controlled by EM1	
		1	PWM. PWM mode is enabled for CTIMERn_MAT1	
2	PWMEN2		PWM mode enable for channel2	0x0
		0	Match. CTIMERn_MAT2 is controlled by EM2	
		1	PWM. PWM mode is enabled for CTIMERn_MAT2	

Table 204: PWMC - PWM Control register (offset = 0x074)) bit description

Bit	Symbol	Value	Description	Reset value
3	PWMEN3		PWM mode enable for channel3. **Note:** It is recommended to use match channel 3 to set the PWM cycle	0x0
		0	Match. CTIMERn_MAT3 is controlled by EM3	
		1	PWM. PWM mode is enabled for CT132Bn_MAT3	
31:4	RESERVED	-	Read value is undefined, only zero should be written	NA

14.7 Functional description

Figure 49 shows a timer configured to reset the count and generate an interrupt on match. The prescaler is set to 2 and the match register set to 6. At the end of the timer cycle where the match occurs, the timer count is reset. This gives a full-length cycle to the match value. The interrupt indicating that a match occurred is generated in the next clock after the timer reached the match value.

Figure 50 shows a timer configured to stop and generate an interrupt on match. The prescaler is again set to 2 and the match register set to 6. In the next clock after the timer reaches the match value, the timer enable bit in TCR is cleared, and the interrupt indicating that a match occurred is generated.

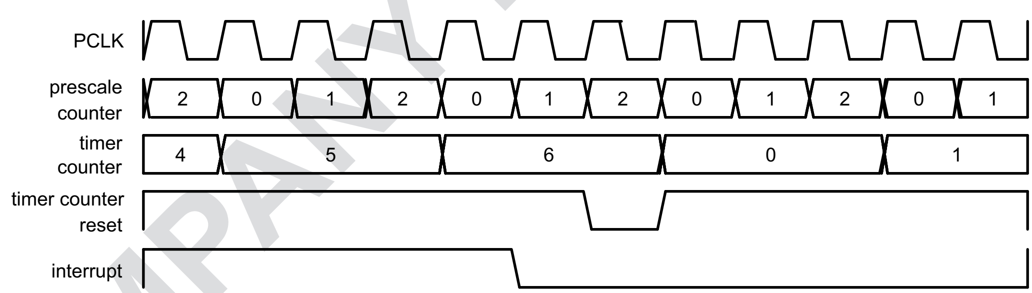


Fig 49. A timer cycle in which PR = 2, MRx = 6, and both interrupt and reset on match are enabled

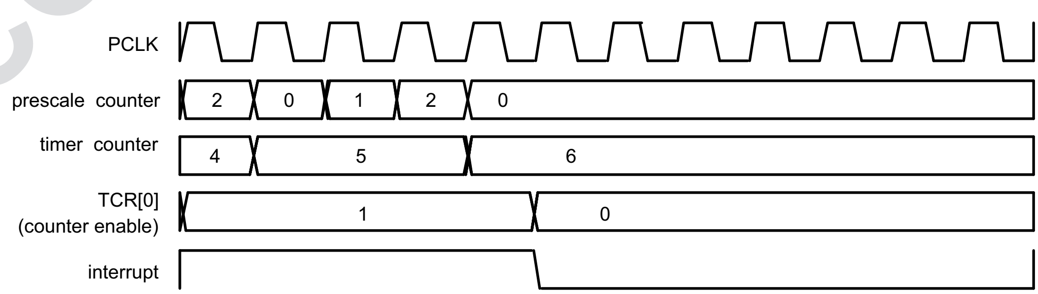


Fig 50. A timer cycle in which PR = 2, MRx = 6, and both interrupt and stop on match are enabled

14.7.1 Rules for single-edge controlled PWM outputs

1. All single-edge controlled PWM outputs go LOW at the beginning of each PWM cycle (timer is set to zero) unless their match value is equal to zero.

2. Each PWM output will go HIGH when its match value is reached. If no match occurs (i.e. the match value is greater than the PWM cycle length), the PWM output remains continuously LOW.
3. If a match value larger than the PWM cycle length is written to the match register, and the PWM signal is HIGH already, then the PWM signal will be cleared with the start of the next PWM cycle.
4. If a match register contains the same value as the timer reset value (the PWM cycle length), then the PWM output will be reset to LOW on the next clock tick after the timer reaches the match value. Therefore, the PWM output will always consist of a one clock tick wide positive pulse with a period determined by the PWM cycle length (i.e. the timer reload value).
5. If a match register is set to zero, then the PWM output will go to HIGH the first time the timer goes back to zero and will stay HIGH continuously.

Note: When the match outputs are selected to perform as PWM outputs, the timer reset (MRnR) and timer stop (MRnS) bits in the Match Control Register MCR must be set to zero except for the match register setting the PWM cycle length. For this register, set the MRnR bit to one to enable the timer reset when the timer value matches the value of the corresponding match register.

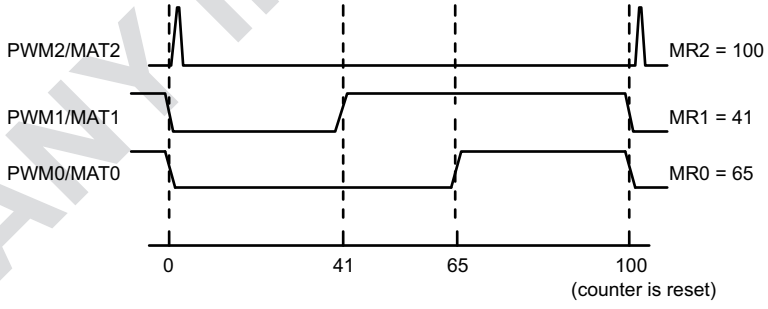


Fig 51. Sample PWM waveforms with a PWM cycle length of 100 (selected by MR3) and MAT3:0 enabled as PWM outputs by the PWCON register

14.7.2 DMA operation

DMA requests are generated by a match of the Timer Counter (TC) register value to either Match Register 0 (MR0) or Match Register 1 (MR1). This is not connected to the operation of the match outputs controlled by the EMR register. Each match sets a DMA request flag, which is connected to the DMA controller. In order to have an effect, the DMA controller must be configured correctly.

When a timer is initially set up to generate a DMA request, the request may already be asserted before a match condition occurs. An initial DMA request may be avoided by having software write a one to the interrupt flag location, as if clearing a timer interrupt. See Section 14.6.1. A DMA request will be cleared automatically when it is acted upon by the DMA controller.

Note: Because timer DMA requests are generated whenever the timer value is equal to the related match register value, DMA requests are always generated when the timer is running, unless the match register value is higher than the upper count limit of the timer. It is important not to select and enable timer DMA requests in the DMA block unless the timer is correctly configured to generate valid DMA requests.

15. WatchDog Timer (WDT)

15.1 Introduction

The watchdog timer is available on all QN908x devices. This module in QN908x is a compliant peripheral developed, tested and licensed by ARM Limited.

15.2 Features

The watchdog module is an AMBA (Advanced Microcontroller Bus Architecture) slave module and connects to the Advanced Peripheral Bus (APB). The features of the module are:

- 32-bit down counter with a programmable timeout interval
- Separate watchdog clock with clock enable for flexible control of the timeout interval.
- Interrupt output generation on timeout
- Reset signal generation on timeout if the interrupt from the previous timeout remains unserviced by software
- Lock register to protect registers from being altered by runaway software.
- WDT stops working in power down mode and holds the status as that before powered down, and can resume to work once waked up.

15.3 Basic configuration

Configuration of the WDT is accomplished as follows:

- Enable WDT clock by writing 1 to bit CLK_WDT_EN in CLK_EN register in syscon.
- Write 0x1ACCEE551 to LOCK register to unlock the WDT register configuration.
- For WDT interrupt feature
  - Enable WDT interrupt in ISER0 register in NVIC
  - Enable WDT interrupt feature by writing INTEN bit of CTRL register to 1
  - Configure LOAD and VALUE register to determine the watchdog count time
- For WDT reset feature
  - Enable WDT reset feature by writing RESEN bit of CTRL register to 1
  - Configure LOAD and VALUE register to determine the Watchdog count time
- Write LOCK register with any value other than 0x1ACCE551 to lock the WDT configuration to protect the WDT from unintentionally modified

15.4 Pin description

The WDT has no external pins.

15.5 General description

15.5.1 Block diagram

The block diagram of the watchdog is shown below in the Figure 52.

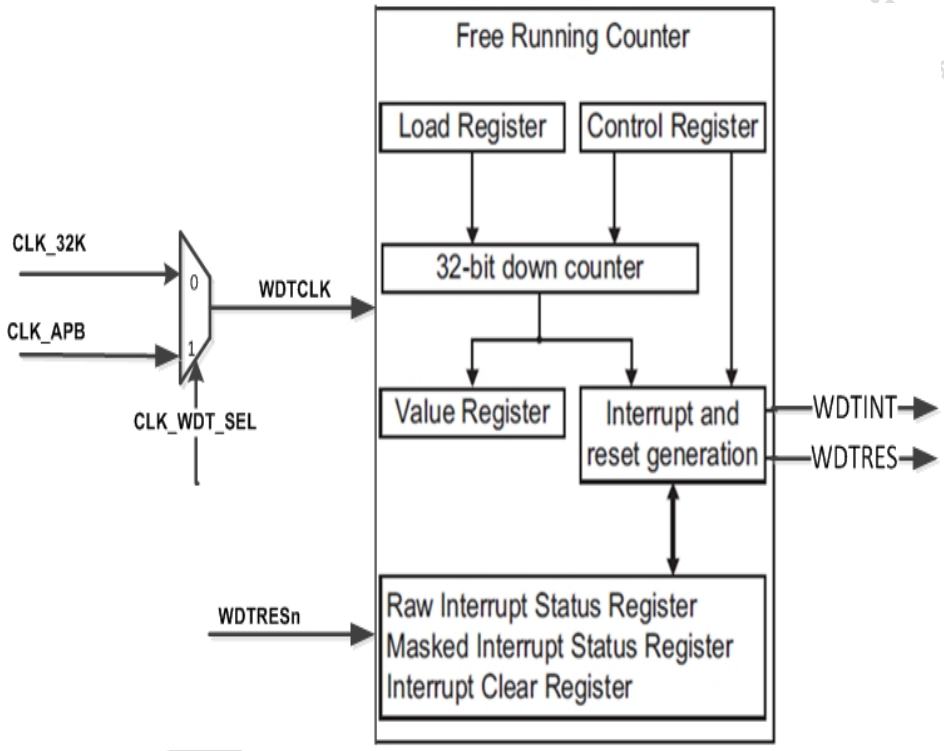


Fig 52. Watchdog timer block diagram

15.5.2 WDT reset

The WDT resets the system when the software fails to clear the WDT within the selected time interval. The WDT can be configured either as a watchdog timer or as a timer for general-purpose use. If the watchdog function is not required in an application, it is possible to configure the watchdog timer to be used as an interval timer that can be used to generate interrupts at selected time intervals. The maximum timeout interval is 1.5 days. The WDT is initialized from the register LOAD. The module generates a regular interrupt, WDOGINT, depending on a programmed value. The counter decrements by one on each positive clock edge of WDOGCLK when the clock enable, CLK_WDT_EN, is HIGH. The watchdog monitors the interrupt and asserts a reset WDOGINT signal, when the counter reaches 0, and the counter is stopped. On the next enabled WDOGCLK clock edge, the counter is reloaded from the LOAD register and the count-down sequence continues. If the interrupt is not cleared by the time that the counter next reaches 0, then the watchdog module reasserts the reset signal.

15.5.3 Using the WDT lock features

To protect the watchdog module registers from being changed unintentionally, the LOCK register WdogLock must be used to disable the write access to the watchdog module registers after registers have been modified. To enable write access to all registers, write 0x1ACCE551 to the LOCK register. After writing to the required watchdog registers, disable write access to all registers except the LOCK register by writing any value other than 0x1ACCE551 to the LOCK register. Reading the LOCK register returns the lock status rather than the 32-bit value written. Therefore, when write accesses are

disabled, reading the LOCK register returns 0x00000001 (locked), otherwise the return value is 0x00000000 (unlocked).

15.5.4 Programming the timeout interval

The watchdog module counter is clocked by the rising edge of WDOGCLK when WDOGCLKEN is HIGH. In the case where WDOGCLKEN is permanently HIGH, the count rate is equal to the WDOGCLK frequency. When WDOGCLKEN is periodically pulsed HIGH for one WDOGCLK rising edge then the count rate is equal to the frequency of the WDOGCLKEN pulses. The frequency of enabled clock edges is referred to as the effective watchdog clock frequency and the period is referred to as the effective watchdog clock period.

The watchdog counter is reloaded from the LOAD register whenever:

- The counter reaches zero
- The interrupt generation is enabled by setting the INTEN bit in the Control Register CTRL, when it was previously disabled
- An interrupt is cleared by writing to the Interrupt Clear register INT_CLR
- A new value is written to the load register LOAD

The time interval between the counter load occurring, and the counter reaching zero and generating an interrupt is given by the following expression:

Interrupt interval: (LOAD + 1) $\times$ effective watchdog clock period

The initial reset value for LOAD is 0xFFFFFFFF and for an example effective

watchdog frequency of 1 MHz (period of 1 μs), the interrupt interval is 4295 seconds.

The minimum valid value for LOAD is 0x00000001. If LOAD is set to 0x00000000, an interrupt is always generated immediately.

15.6 Register description

The watchdog timer contains the registers shown in Table 205.

The reset value reflects the data stored in used bits only. It does not include the content of reserved bits.

Table 205. Register overview:Watchdog timer (base address 0x4000 1000)

Name	Access	Offset	Description	Reset value	Section
LOAD	R/W	0x000	watchdog counter start value register	0xFFFF	15.6.1
VALUE	R/W	0x004	watchdog counter value register	0x1FFFFFFF	15.6.2
CTRL	R/W	0x008	watchdog control register	0x0	15.6.3
INT_CLR	R/W	0x00C	interrupt clear register	0x0	15.6.4
INT_RAW	R	0x010	raw interrupt status register	0x0	15.6.5
MIS	R/W	0x014	interrupt mask register	0x0	15.6.6
LOCK	R/W	0x020	watchdog lock register	0x0	15.6.7

15.6.1 WDT LOAD register

Table 206. LOAD - WDT LOAD register (offset = 0x00) bit description

Bit	Symbol	Access	Reset value	Description
31:0	LOAD	rw	0xFF FF	Contains the value from which the counter is to decrement. When this register is written to the count is immediately restarted from the new value. The minimum valid value is 1

15.6.2 WDT VALUE register

Table 207. VALUE - WDT VALUE register (offset = 0x04) bit description

Bit	Symbol	Access	Reset value	Description
31:0	VALUE	rw	0x1F FFFF FF	The current value of the decrementing counter

15.6.3 WDT CTRL register

Table 208. WDT CONTROL register (CTRL, offset: 0x08) bit description

Bit	Symbol	Access	Reset value	Description
0	INTEN	rw	0	Enable the interrupt event WDOGINT. Set HIGH to enable the counter and the interrupt, and set LOW to disable the counter and interrupt
1	RESEN	rw	0	Enable watchdog reset output WDOGRES. Acts as a mask for the reset output. Set HIGH to enable the reset and LOW to disable the reset
31:2	RESERVED	-	-	-

15.6.4 WDT interrupt clear register

Table 209. INT_CLR - WDT Interrupt Clear register (offset = 0x0C) bit description

Bit	Symbol	Access	Reset value	Description
0	INTCLR	rw	0	A write of any value to the Register clears theWatchdog interrupt and reloads the counter from the value in WDOGLOAD.
31:1	RESERVED	-	-	-

15.6.5 WDT RAW interrupt status register

Table 210. INT_RAW - WDT Raw Interrupt Status register (offset = 0x10) bit description

Bit	Symbol	Access	Reset value	Description
0	RAWINTSTAT	r	0	raw interrupt status from the counter
31:1	RESERVED	-	-	-

15.6.6 WDT masked interrupt status register

Table 211. MIS - WDT Masked Interrupt Status register (offset = 0x14) bit description

Bit	Symbol	Access	Reset value	Description
0	MASKINTSTAT	rw	0	Enabled interrupt status from the counter
31:1	RESERVED	-	-	-

15.6.7 WDT LOCK register

Table 212. LOCK - WDT LOCK register (offset = 0x20) bit description

Bit	Symbol	Access	Reset value	Description
31:0	LOCK_31_0	rw	0	Writing 0x1ACCE551 to this register enables write access to all other registers

16. Real-Time Clock (RTC)

16.1 Introduction

The RTC is available on all QN908x devices.

16.2 Features

The RTC provides real time counter based on low-power 32KHz clock.The Features of RTC are listed as below

- Operate on external 32.768KHz crystal or internal 32KHz oscillator clock
- 15-bit counter running of 32KHz clock, to generate second with calibration function
- 32-bit second counter to count seconds generated by the 15-bit counter
  – Support configuration of second register on the fly
- 32-bit free running counter and associated match register:
  – Operate on 32 kHz clock.
  – Match register for interrupt generation.
  – Match register to generate reset, to be used asWatchdog function.
- RTC can work in power-down 0 mode.
- RTC can only be reset by power-on reset or by setting SET_RTC_RST to 1. While chip is powered on, the counter starts running.
- Maskable interrupts
  – Programmable second interrupt
  – Programmable Input capture interrupt with configurable capture edge
  – Free-running interrupt

16.3 Basic configuration

Configure the RTC as follows:

- Write CLK_BIV_EN bit into 1 in CLK_EN register to enable RTC clock.
- Supply power to XTAL32K (writing XTAL32K_DIS bit into 0 of PMU_CTRL1 register) or RCO 32K (writing RCO32K_DIS bit into 0 of PMU_CTRL1 register), according to application.
- Configure CLK_32K_SEL bit in CLK_CTRL register to select the source of 32KHz clock.
- For enabling SEC_INT interrupt, write SEC_INT_EN bit into 1 of CTRL register (Section 16.6.1) to enable SEC_INT and the corresponding NVIC interrupt number is 5. Then, SEC_INT will be generated every second.
- To enable free-running function (FR_INT), write the following bits of CNT2_CTRL register:
  – CNT2_EN bit as 1 to enable the free-running counter.
  – CNT2_INT_EN bit as 1 to enable free-running interrupt.
  – CNT2_WAKEUP bit as1 to wake up the MCU.

Remark: The higher 16 bit of CNT2_CTRL should be a magic number 0x5285 that protects the register written by any unexpected run-away program. The corresponding NVIC interrupt number is 6.

The clock of RTC can be shown as below:

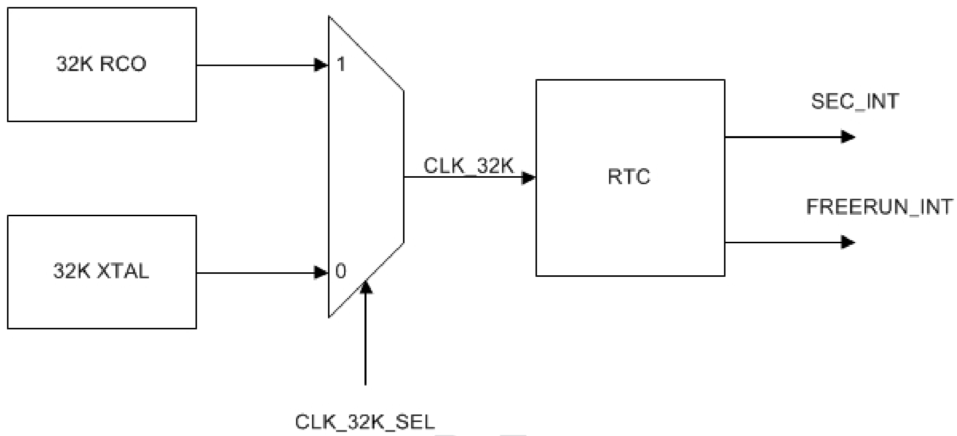


Fig 53. RTC clocking

16.4 Pin Description

If XTAL32K is used as the clock source of RTC, XTAL32K_IN and XTAL32K_OUT pins should be connected to a 32.768KHz crystal. During this time, PB00_AE & PB01_AE bit in PIO_CFG_MISC register(Section 2.5.22) should be configured into 1.

16.5 General description

The RTC block diagram can be shown as below:

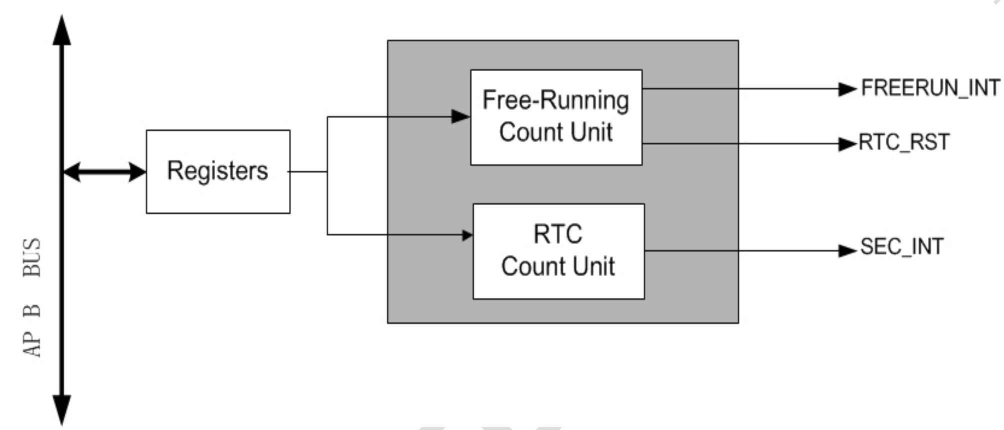


Fig 54. RTC Block Diagram

There are mainly two count units in RTC:

- RTC count unit: CNT0 and CNT1 work together to generate second interrupt
- Free running counter unit: CNT 2 generates free running interrupt and reset

16.5.1 RTC Count Unit

The following figure illustrates RTC counter unit.

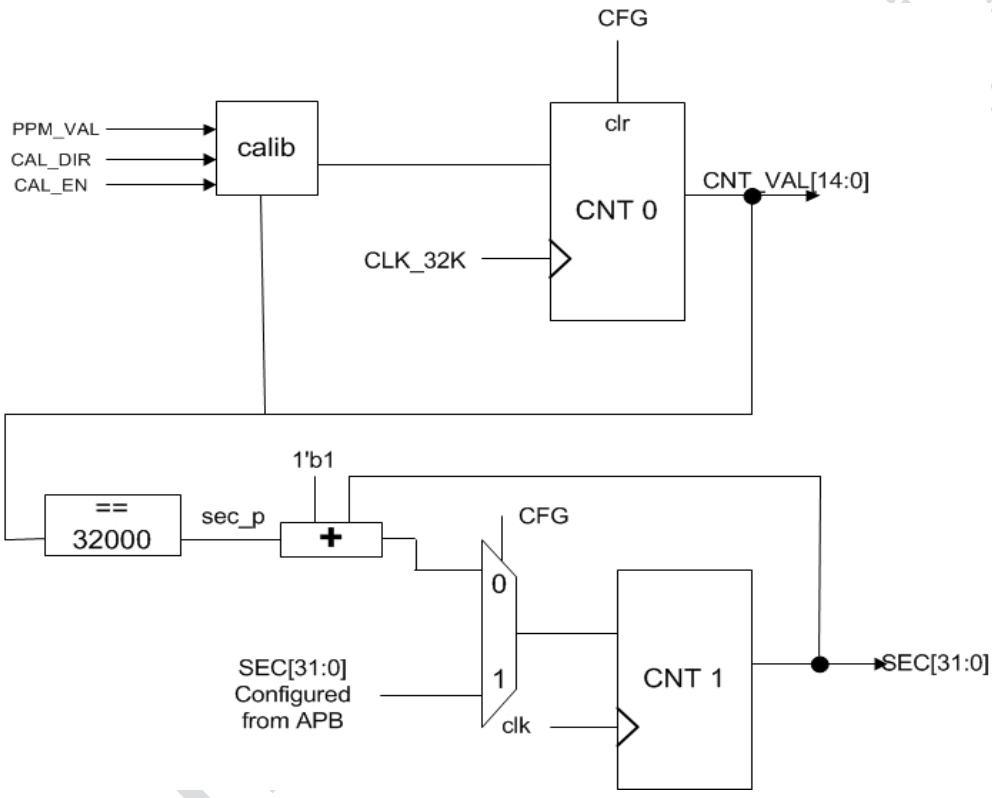


Fig 55. RTC Counter Unit

Two counters work together for appropriate functioning of RTC. The function of CNT 0 is to generate the second pulse with calibration and the function of CNT 1 is to provide the time information in seconds. The bit width of counter 1 is 32 for which the time range is about 136 years.

16.5.1.1 CNT 0

CNT 0 is 15-bit width and counts up to 32000. The counter begins to count from zero to 32000 when the 32KHz clock is enabled. When it reaches 32000, a pulse is generated to indicate the second. This counter is clear to zero if the CFG bit of the CTRL (Section 16.6.1) register is written as 1.

CNT 0 also supports calibration function to calibrate the clock jitter. The calibration function is enabled by configuration of CAL_EN in CTRL (Section 16.6.1) register. The PPM value of calibration is set by PPM_VAL in the register CALIB (Section 16.6.4), and the direction of calibration is controlled by CAL_DIR bit of CALIB. The precision of calibration is 1 ppm.

The ppm definition is calculated as per the following equation. where f_0: 32000.

$$ppm = (f_0 / f_1 - 1) \times 2^{20} \quad (1)$$

If the ppm value is positive, the calibration direction bit (CAL_DIR) is set to 0, otherwise is this bit is set to 1.

16.5.1.2 CNT 1

As shown in Figure 55, the bit width of counter 1 is 32. The time range it can provide is about 136 years.

For configuring the initial value to count from, the software needs to set the SEC register (Section 16.6.7) and writes 1 to the configure bit CFG in the RTC_CTRL register.

16.5.2 Free-Running counter unit

Free-Running counter unit contains one counter (CNT2), which is a 32-bit counter with 32 KHz clock. When it is enabled, it adds one at every clock rising edge. When it is disabled, it will be 0.

When the value of the counter is equal to the threshold of interrupt (THR_INT, Section 16.6.7), a free-running counter interrupt is generated if the interrupt is enabled (Register CNT2_INT_EN). If both CNT2_INT_EN and CNT2_WAKEUP are enabled, the interrupt can be the wake-up source of system. By CNT2_INT_EN bit is set to 0, it can clear the free running interrupt.

When the value of counter is equal to the threshold of reset (THR_RST, Section 16.6.8), a system reset is generated if the CNT2_RST is enabled.

Remark: if both CNT2_RST and CNT2_INT_EN are enabled, the value of THR_RST must be larger than THR_INT.

16.5.2.1 Magic Number

While configuring CNT2_CTRL register, magic number is implemented to protect the register from errors written by any unexpected run-away program. The magic number is 0x5285. If you want to write this register, the higher 16 bit of the writing data should be 0x5285, otherwise the data can not be written into register.

16.6 Register description

Reset Values pertain to initial power-up of the device or when an RTC software reset is applied (except where noted). This block is not initialized by any other system reset.

Table 213. Register Overview: RTC (base address 0x4000 B000)

Name	Access	Offset	Description	Reset value	Section
CTRL	RW	0x00	RTC control register	0x100	16.6.1
STATUS	RW	0x04	RTC status register	0x0	16.6.2
SEC	RW	0x08	RTC second register	0x0	16.6.3
CAL	RW	0x10	RTC calibration register	0x0	16.6.4
CNT_VAL	RW	0x14	RTC count value register	0x0	16.6.5
CNT2_CTRL	RW	0x20	Free running control register	0x0	16.6.6
THR_INT	RW	0x24	interrupt threshold of free running counter register	0x0f000000	16.6.7
THR_RST	RW	0x28	reset threshold of free running counter register	0xff000000	16.6.8
CNT2	RW	0x2C	free running count value	0x0	16.6.9

16.6.1 RTC CTRL register

Table 214. RTC control register (CTRL, offset 0x00) bit description

Bit	Symbol	Access	Reset value	Description
0	SEC_INT_EN	RW	0x0	RTC second interrupt enable
1	-	-	-	Reserved
2	CFG	RW	0x0	RTC second configuration control. This bit is self-cleared after synchronization
7:3	-	-	-	Reserved
8	CAL_EN	RW	0x1	Calibration enable
31:9	-	-	-	Reserved

16.6.2 RTC Status register

Table 215. RTC status register (STATUS, offset 0x04) bit description

Bit	Symbol	Access	Reset value	Description
0	SEC_INT	W1C	0x0	Second interrupt flag
7:1	-	-	-	Reserved
8	CTRL_SYNC	R	0x0	Control Register synchronization busy indicator
9	STATUS_SYNC	R	0x0	Status Register synchronization busy indicator
10	SEC_SYNC	R	0x0	Second configuration Register synchronization busy indicator
11	-	-	-	Reserved
12	CALIB_SYNC	R	0x0	Calibration Register synchronization busy indicator
15:13	-	-	-	Reserved
16	FREE_SYNC	R	0x0	Free running counter control Register synchronization busy indicator
17	THR_INT_SYNC	R	0x0	Free running counter interrupt Threshold Register synchronization busy indicator
18	THR_RST_SYNC	R	0x0	Free running counter Reset Threshold Register synchronization busy indicator
30:19	-	-	-	Reserved
31	FREE_RUNNING_INT	R	0x0	Free running interrupt status.

16.6.3 RTC Second register

Table 216. RTC second register (SEC, offset 0x08) bit description

Bit	Symbol	Access	Reset value	Description
31:0	SEC	rw	0x0	Second configuration register.

16.6.4 RTC Calibration register

Table 217. RTC second register (CAL, offset 0x10) bit description

Bit	Symbol	Access	Reset value	Description
15:0	PPM	rw	0x0	RTC calibration ppm value the precision is 1 ppm.
16	DIR	rw	0x0	RTC calibration direction indicator; 0:forward calibrate; 1:backward calibrate;
31:17	-	-	-	Reserved

16.6.5 RTC Count Value register

Table 218. RTC Count Value register (CNT_VAL, offset 0x14) bit description

Bit	Symbol	Access	Reset value	Description
0:14	CNT	r	0	RTC counter current value read only.
31:15	-	-	-	Reserved

16.6.6 RTC Free running control register

Table 219. RTC free running control register (CNT2_CTRL, offset 0x20) bit description

Bit	Symbol	Access	Reset value	Description
0	CNT2_EN	rw	0	1 to enable free running counter
1	CNT2_INT_EN	rw	0	1 to enable free running interrupt
2	CNT2_WAKEUP	rw	0	1 to enable free running wakeup
3	CNT2_RST	rw	0	1 to enable free running reset
31:4	-	-	-	Reserved

16.6.7 RTC Interrupt Threshold register

Table 220. RTC Count Value register (THR_INT, offset 0x24) bit description

Bit	Symbol	Access	Reset value	Description
31:0	THR_INT	rw	0x0f00 0000	The Threshold of free running counter is to generate free running interrupt.

16.6.8 RTC Reset Threshold register

Table 221. RTC FR Counter Reset Threshold register (THR_RST, offset 0x28) bit description

Bit	Symbol	Access	Reset value	Description
31:0	THR_RST	rw	0xff00 0000	The Threshold of free running counter is to generate free running reset.

16.6.9 RTC Freerunning Count Value register

Table 222. RTC Free Running count value(CNT2, offset 0x2C) bit description

Bit	Symbol	Access	Reset value	Description
31:0	CNT2	R	0	The current value of free running counter

17. CPU system tick timer (SYSTICK)

17.1 Introduction

The system tick timer (SysTick timer) is available on all QN908x devices in the ARM Cortex-M4.

17.2 Basic configuration

Configuration of the system tick timer is accomplished as follows:

- Program the SYST_RVR register with the reload value calculated to obtain the desired time interval.
- 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
- Configure SYST_CSR register to select clock source and enable the system tick timer.

17.3 Features

- Simple 24-bit timer.
- Uses dedicated exception vector.
- Clocked by CLK_AHB or 32KHz clock.

17.4 General description

Block diagram of the SysTick timer for each CPU is shown in Figure 56.

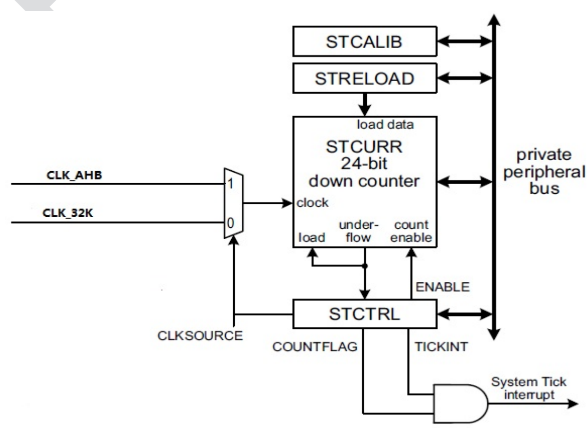


Fig 56. System tick timer block diagram

The SysTick timer is an integral part of the Cortex-M4. The SysTick timer is intended to generate a fixed 10 ms interrupt for use by an operating system or other system management software.

Since the SysTick timer is a part of the CPU, it facilitates porting of software by providing a standard timer that is available on ARM Cortex-based devices. The SysTick timer can be used for:

- A RTOS tick timer which fires at a programmable rate (for example 100 Hz) and invokes a SysTick routine.
- A high-speed alarm timer using the core clock.
- A simple counter. Software can use this to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNTFLAG bit-field in the control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

Refer to the appropriate ARM Cortex User Guide for details.

17.5 Register description

The systick timer registers are located on the private peripheral bus of each CPU.

Table 223. Register overview: SysTick timer (base address 0xE000 E000)

Name	Access	Offset	Description	Reset value[1]	Section
SYST_CSR	R/W	0x010	System Timer Control and status register	0x0	17.5.1
SYST_RVR	R/W	0x014	System Timer Reload value register	0x0	17.5.2
SYST_CVR	R/W	0x018	System Timer Current value register	0x0	17.5.3
SYST_CALIB	RO	0x01C	System Timer Calibration value register	0x0	17.5.4

[1] Reset Value reflects the data stored in used bits only. It does not include content of reserved bits.

17.5.1 System Timer Control and status register

The SYST_CSR register contains control information for the SysTick timer and provides a status flag. This register is part of the CPU.

This register determines the clock source for the system tick timer.

Table 224. SysTick Timer Control and status register (SYST_CSR, offset 0x010) bit description

Bit	Symbol	Description	Reset value
0	ENABLE	System Tick counter enable. 1= the counter is enabled. 0: the counter is disabled.	0x0
1	TICKINT	System Tick interrupt enable. 1: the System Tick interrupt is enabled. 0: the System Tick interrupt is disabled. When enabled, the interrupt is generated when the System Tick counter counts down to 0.	0x0
2	CLKSOURCE	System Tick clock source selection. 1: the CLK_AHB is selected. 0: the 32 kHz clock is selected as the reference clock.	0x0
15:3	-	Reserved. Read value is undefined, only zero should be written.	-
16	COUNTFLAG	Returns 1 if the SysTick timer counted to 0 since the last read of this register.	0x0
31:17	-	Reserved. Read value is undefined, only zero should be written.	-

17.5.2 System Timer Reload value register

The SYST_RVR register is set to the value that will be loaded into the SysTick timer whenever it counts down to zero. This register is loaded by software as part of timer initialization. The SYST_CALIB register can be read and used as the value for SYST_RVR register if the CPU is running at the frequency intended for use with the SYST_CALIB value.

Table 225. System Timer Reload value register (SYST_RVR, offset 0x014) bit description

Bit	Symbol	Description	Reset value
23:0	RELOAD	This is the value that is loaded into the System Tick counter when it counts down to 0.	0x0
31:24	-	Reserved. Read value is undefined, only zero should be written.	-

17.5.3 System Timer Current value register

The SYST_CVR register returns the current count from the System Tick counter when it is read by software.

Table 226. System Timer Current value register (SYST_CVR, offset 0x018) bit description

Bit	Symbol	Description	Reset value
23:0	CURRENT	Reading this register returns the current value of the System Tick counter. Writing any value clears the System Tick counter and the COUNTFLAG bit in SYST_CSR.	0x0
31:24	-	Reserved. Read value is undefined, only zero should be written.	-

17.5.4 System Timer Calibration value register

The value of the SYST_CALIB register is read-only and is provided by the value of the SYS_TICK register in the system configuration block (see Table 10).

Table 227. System Timer Calibration value register (SYST_CALIB, offset 0x01C) bit description

Bit	Symbol	Description	Reset value
23:0	TENMS	Reload value from the SYS_TICK register in the SYSCON block. This field is loaded from the SYS_TICK register in Syscon.	0x0
29:24	-	Reserved. Read value is undefined, only zero should be written.	-
30	SKEW	Indicates whether the TENMS value will generate a precise 10 millisecond time, or an approximation. This bit is loaded from the SYS_TICK register in Syscon. 0: the value of TENMS is considered to be precise. 1: the value of TENMS is not considered to be precise.	0x0
31	NOREF	Indicates whether an external reference clock is available. This bit is loaded from the SYS_TICK register in Syscon. 0: 32 KHz clock is available. 1: 32 KHz clock is not available.	0x0

17.6 Functional description

The SysTick timer is a 24-bit timer that counts down to zero and generates an interrupt. The intent is to provide a fixed 10 ms time interval between interrupts. The SysTick timer is clocked from CLK_AHB (see Figure 9) or from the 32kHz reference 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.

17.7 Example timer calculations

To use the system tick timer, do the following:

1. Program the SYST_RVR register with the reload value calculated as shown n the following examples in this section to obtain the desired time interval.
2. 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 enabled.

The following examples illustrate selecting SysTick timer reload values for different system configurations. All of the examples calculate an interrupt interval of 10 ms, as the SysTick timer is intended to be used, and there are no rounding errors.

CLK_AHB: 32 MHz

Program the SYST_CSR register with the value 0x7 which selects CLK_AHB as the clock source and enables the SysTick timer and the SysTick timer interrupt.

Write 0 to bit[24] in SYS_TICK register to indicate a precise TENMS value.

SYST_RVR: (CLK_AHB * 10 ms) -1: (32 MHz * 10 ms) -1: 320000 -1: 319999: 0x0004 E1FF

Use 32KHz RCO as reference clock

Program the SYST_CSR register with the value 0x3 which selects the 32kHz RCO clock as the clock source and enables the SysTick timer and the SysTick timer interrupt..

Write 0 to bit[24] in SYS_TICK register to indicate a precise TENMS value.

Write 0 to bit[25] in SYS_TICK register to indicate the external reference clock available.

SYST_RVR: (32 *10 ms) -1: (32 kHz * 10 ms) -1: 320 -1:319: 0x0000 013F

Use 32.768KHz crystal as reference clock

Program the SYST_CSR register with the value 0x3 which selects the 32.768kHz clock as the clock source and enables the SysTick timer and the SysTick timer interrupt.

Write 1 to bit[24] in SYS_TICK register to indicate a not precise TENMS value.

Write 0 to bit[25] in SYS_TICK register to indicate the external reference clock available.

SYST_RVR: (32.768 kHz   10 ms) 1: (32.768 kHz   10 ms) 1: 328 1:327: 0x0000 0147

CLK_AHB: 16 MHz

Program the SYST_CSR register with the value 0x7 which selects CLK_AHB as the clock source and enables the SysTick timer and the SysTick timer interrupt.
Write 0 to bit[24] in SYS_TICK register to indicate a precise TENMS value.
SYST_RVR: (system clock frequency * 10 ms) -1: (16 MHz *10 ms) -1: 160000 -1: 159999: 0x0002 70FF.

18. Fusion Signal Processing (FSP)

18.1 Introduction

Fusion Signal Processor (FSP) is a hardware module that enables faster computation of some of the frequently used Math and Signal Processing operations, including:

- Transcendental functions
- Matrix computations
- FFT/IFFT/DCT/IDCT
- Statistics function
- Correlation
- Multi-channel FIR filter

The FSP functions can be accessed through an API which is compatible with ARM CMSIS library. This enables a user to easily port an existing software application that uses CMSIS lib to the much faster FSP lib.

18.2 Features

- Round-robin arbitration when FSP read & write data from/to SRAM.
- Support Q31 (<s,0,31>, one bit sign bit, 0 bit integer bit, and 31 bit faction bit) and single precision floating point data, with internal data type conversion circuit.
- Matrix operation
  - Matrix inversion based on LU decomposition with pivot, support size from 2x2 to 9x9
  - Matrix multiplier with size MxN and NxK (M,N,K ⩽ 9)
  - Matrix dot multiplier with size MxN (M, N ⩽ 9)
  - Matrix transpose with size MxN (M, N ⩽ 9)
  - Matrix addition and subtraction with size M x N
  - Linear combination of two matrices: a * X + b * Y, where are a and b are single precision floating point numbers and X and Y are M x N float/q31 matrices
- Transform Engine
  - Support 64, 128, 256 points FFT, DCT, real or complex data input, complex data output
  - Support 64, 128, 256 points IFFT, IDCT using the same circuit as FFT/DCT., real or complex data input, complex data output
- Correlation
  - Length of 2 sequence is programmable.
  - Sequence correlation start point is configurable, when calculation reaches the end, wraps back to the beginning of sequence.
- Statistic engine, which can be used to compute
  - Min/Max values and their corresponding index of a given array

  - Configurable to select either the smallest or the largest index of the sub-set of indices that are equal to Min/Max
  - Sum of all elements of an array
  - Sum of squares an array
- FIR filter
  - Supports up to 9 channel
  - Channel tap length programmable up to 16.
  - Each channel buffer is clearable.
- Cordic Engine
  - sin/cos calculation in a single function call
  - ln/sqrt (natural log/square root) calculation in a single function call
  - native mode support to enable computation of other transcendental functions

18.3 Basic configuration

Configure FSP as follows:

- Enable FSP clock by setting CLK_FSP_EN bit to '1' in CLK_EN register (see Section 2.5.4).
- Power on FSP by setting FSP_DIS bit to '0; in PMU_CTRL0 register (see Section 2.5.38).
- For MOU (Matrix Operation Unit)
  - Configure MOU_DONE_INTEN & FSP_INT_EN bit in INTEN register (offset 0xC) to '1' in order to enable MOU interrupt.
  - Write NVIC register to enable FSP interrupt.
  - Configure MA_SRC_BASE, MB_SRC_BASE (if needed), MO_SRC_BASE register to their corresponding matrices' base address
  - Configure MOU_SCALEA and MOU_SCALEB register if needed
  - Configure MOU_CTRL, writing corresponding parameters of calculation to this register to start the MOU operation
  - Wait for FSP interrupt, and read the result data from base address configured above.
- For TE (Transform Engine)
  - Configure TE_DONE_INTEN & FSP_INT_EN bit in INTEN (offset 0xC) register to '1' in order to enable TE interrupt.
  - Write NVIC register to enable FSP interrupt.
  - Configure TE_SRC_BASE & TE_DST_BASE to corresponding data array base address.
  - Configure TE_CTRL, writing corresponding parameters of calculation to this register to start the TE operation.
  - Wait for FSP interrupt, and read the result data from base address configured above.
- For FIR

    - Set DIS_FIR bit to '0' in PMU_CTRL0 register (address: 0x4000085C) to power on FIR buffer
    - Write '1' to FIR_BUF_CLR_ALL bit in FIR_CFG_CH0 register to clear all the FIR buffers. If not all channels are used then write '1' to FIR_CH<x>_BUF_CLR bit in FIR_CFG_CH<x> register to clear the buffer corresponding to that specific channel
    - Wait for the clear operation to be complete, by reading FIR_BUF_CLR_ALL or FIR_CH<x>_BUF_CLR bit to check if they are '0'.
    - Write channel configuration data that includes channel tap length, coefficient base address to FIR_CFG_CH<x> register
    - Write fixed point data to FIR_DAT<x>_FX register one by one or write floating point data to FIR_DAT<x>_FL register one by one to start the calculation
    - After writing one data sample to each register above, read the result from the same address to get the result
- For COR (correlation)
    - Configure COR_DONE_INTEN & FSP_INT_EN bit in INTEN (offset: 0xC) register to '1' in order to enable COR interrupt
    - Write NVIC register to enable FSP interrupt
    - Write CX_SRC_BASE, CY_SRC_BASE, CO_DST_BASE register with input and output arrays' base address
    - Write COR_OFFSET register with two source sequences' offset.
    - Write COR_CTRL register with corresponding settings, which includes the length of the sequence, fixed or floating point input/output, to start the calculation
    - Wait for FSP interrupt, and read the result data from base address configured above
- For SE (statistic engine)
    - Write SE_DONE_INTEN & FSP_INT_EN bit in INTEN (offset: 0xC) register to 1 in order to enable SE interrupt
    - Write NVIC register to enable FSP interrupt
    - Write SE_SRC_BASE register with source array address
    - Write SE_CTRL register with corresponding settings to start the calculation
    - Wait for FSP interrupt, and read the result from SE_IDX, SE_SUM and SE_PWR registers
- For Cordic Engine
    - Write the source data and output data word address to SIN_COS_<x> register, to start sine and cosine calculation, and read the data from output data address. The input and output data format is decoded by register address. Where x is IXOX, IXOL, ILOX or ILOL.
    - Write the source data and output data word address to LN_SQRT_<x> register, to start square root and natural logarithm calculation, and read the data from output data address. The input and output data format is decoded by register address. Where x is IXOX, IXOL, ILOX or ILOL.
    - Write the source data and output data word address to CORDIC<x> register to start native cordic calculation, and read the data from output data address. The parameter and data format are decoded by register address. Where x is

T0U0_IXOX, T0U0_IXOL, T0U0_ILOX, T0U0_ILOL, T0U1_IXOX, T0U1_IXOL, T0U1_ILOX, T0U1_ILOL, T1U0_IXOX, T1U0_IXOL, T1U0_ILOX, T1U0_ILOL, T1U1_IXOX, T1U1_IXOL, T1U1_ILOX or T1U1_ILOL.

18.4 Pin description

There is no pin for FSP.

18.5 General description

The FSP has six computational modules which can be conceptually viewed as three hardware blocks:

- Cordic
- Matrix Operation Unit (MOU)& Transform Engine (TE)
- Statistic Engine (SE), Correlation & FIR

The three hardware blocks can operate in parallel, but within each hardware block only one module can be active at any given time. For example, the cordic module and the TE can operate in parallel, but the MOU cannot be active when the TE is also active.

Users need to configure registers to start FSP calculation, and the engines themselves fetch data from SRAM automatically.

FSP reads source data from system memory, and writes back the computed data to system memory. The memory address that FSP needs is only the offset of system memory, which is independent with system memory remap. Besides, it is assumed that all the data is stored in system memory, and that the ROM or flash will not be used to store the data.

18.5.1 Cordic

The Cordic module supports both fixed point (Q31) and floating point (single precision) operations. The core of the Cordic engine is implemented using fixed point operations. Floating point operations are enabled by performing floating to fixed point conversion at the input to the cordic engine and then converting to fixed point output to floating point format. The format of the input or output data is decoded by the register address. See Section 18.5.1.3 for additional details on Cordic configuration.

The Cordic module is used for transcendental functions. It can be used in one of the three following modes:

- Sin/Cos - Computes sine and cosine of the input argument
- Ln/Sqrt - Computes natural logarithm (ln) and square root of the input argument
- Native - Can be programmed to compute other transcendental functions

To choose these operation, address decoder is also implemented here to distinguish them.

CPU writes source data (word) address (SA) and output destination data (word) address (DA) into certain pre-defined address to start Cordic. The SA occupies the lower 16 bits and the DA occupies the higher 16 bits:

Table 228. Address allocation

31-16	15-0
DA	SA

To support larger memory space, DA and SA only saves the word address of data. Both the source address and the destination address needs to be shifted to the right by 2 bits to generate the word address. Note that the address here is the offset address of data in system memory, not the absolute address. Here is a pseudo code to get the offset address:

Address: ((&(src_data)>> 2) & 0xFFFF) | ((&(dst_data) << 14) & 0xFFFF0000)

After writing the address to the register, Cordic module will the read source data from system memory (SRAM), and begin computations, then the output data will be written into destination address configured in the register.

Common Cordic equations are given below:

$$\begin{cases} x^{(i+1)} = (x^{(i)} - \mu d_i (2^{-i} y^{(i)})) \\ y^{(i+1)} = (y^{(i)} + d_i (2^{-i} x^{(i)})) \\ z^{(i+1)} = z^{(i)} - d_i e^{(i)} \end{cases}$$

The parameters μ, d_1, $e^{(i)}$ are different in different mode. The detailed information of cordic algorithm can be found in Section 18.6.1.3.

To make the description more clearly, here defines:

- X0, Y0, Z0: initial state of X, Y and Z in CORDIC engine.
- X(n), Y(n), Z(n): X, Y, and Z value after n step of iteration.

18.5.1.1 Program flow

To perform the Cordic computations, follow these steps::

- Enable FSP clock.
- Save source data in system memory, which means source data is saved in address SA (word address).
- Declare an array to save cordic output (two depth array), which means destination data are saved in address DA (word address).
- Gather the source address and destination address ((DA << 14) & 0xFFFF0000 + SA & 0xFFFF) and write to Cordic register (offset address in 0x140 to 0x190).
- Read the result from DA.

All the operation are based on the same flow above, and different mode are decoded out by register address.

18.5.1.2 Data address

Both the input and the output data are saved in system memory, and they are arranged according to the procedure described below.

For source data, SIN/COS & LN/SQRT mode, there is only one input data, so it is saved in SA address. And for native mode, the data address arrangement can be shown as below:

Table 229. Data address arrangement

mode	native
SA	X0
SA+1	Y0
SA+2	Z0

The description of X0, Y0 and Z0 can be found in Section 18.5.1 above.

For output result, the 3 modes output may not the same, which is listed as below:

Table 230. Output result

mode	Sin/Cos	Ln/Sqrt	Native
DA	Cos	Sqrt	X(n)
DA+1	Sin	Ln	Y(n) (t:= 0) /Z(n) (t:= 1)

The description of X(n), Y(n) and Z(n) can be found in Section 18.5.1 above.

For example, when software writes source/destination address to SIN_COS_IXOX register, the output cosine data is saved in DA, and sine data is saved in DA+1.

18.5.1.3 Address decode

All the modes and parameters or cordic are decoded out by register address. These include:

- Input format
- Output format
- T: 0/1 (native mode only)
- U: 1/-1 (native mode only)

As a whole, the following table shows the address decoder content:

Table 231. Address decoder content

AHB Address	Register Name	input format	output format	t	u
140h	SIN_COS_IXOX	fix	fix		
144h	SIN_COS_IXOL	fix	float		
148h	SIN_COS_ILOX	float	fix		
14Ch	SIN_COS_ILOL	float	float		
150h	LN_SQRT_IXOX	fix	fix		
154h	LN_SQRT_IXOL	fix	float		
158h	LN_SQRT_ILOX	float	fix		
15Ch	LN_SQRT_ILOL	float	float		
160h	CORDIC_T0UN_IXOX	fix	fix	0	-1
164h	CORDIC_T0UN_IXOL	fix	float	0	-1
168h	CORDIC_T0UN_ILOX	float	fix	0	-1
16Ch	CORDIC_T0UN_ILOL	float	float	0	-1
170h	CORDIC_T0UP_IXOX	fix	fix	0	1
174h	CORDIC_T0UP_IXOL	fix	float	0	1
178h	CORDIC_T0UP_ILOX	float	fix	0	1
17Ch	CORDIC_T0UP_ILOL	float	float	0	1
180h	CORDIC_T1UN_IXOX	fix	fix	1	-1
184h	CORDIC_T1UN_IXOL	fix	float	1	-1
188h	CORDIC_T1UN_ILOX	float	fix	1	-1
18Ch	CORDIC_T1UN_ILOL	float	float	1	-1

Table 231. Address decoder content

AHB Address	Register Name	input format	output format	t	u
190h	CORDIC_T1UP_IXOX	fix	fix	1	1
194h	CORDIC_T1UP_IXOL	fix	float	1	1
198h	CORDIC_T1UP_ILOX	float	fix	1	1
19Ch	CORDIC_T1UP_ILOL	float	float	1	1

For example, writing CORDIC_T1UN_IXOX register means:

- Input source data format is fix point
- Output destination data format is fix point
- T=1
- U=-1(negetive)

18.5.1.4 Error handling

The Cordic engine raises an error flag for certain combinations of the input values and the functions. The table below summarizes these error conditions:

Table 232. Error handling

	s	e	m	value	sin_cos	native	ln	sqrt
special numbers	1	e=0	m=0	-0.0	Sin=-0.0	a=0	error	error
					Cos=1			
	0	e=0	m=0	0.0	Sin=+0.0	a=0	error	Sqrt=+0.0
					Cos=1			
	1	e=all 1	m=0	$-\infty$	error	error	error	error
	0	e=all 1	m=0	∞	error	error	error	error
		e=all 1	m!=0	NaN(not a number)	error	error	error	error
subnormal numbers	1	e=0		A: $(-1)^s$ x 0.m x 2^{1-bias}	Sin=-0.0	a=0	error	error
					Cos=1			
	0	e=0		A: $(-1)^s$ x 0.m x 2^{1-bias}	Sin=+0.0 Cos=1	a=0	A=mm*2^N mm=(-1)s *0.m,N=-126	Sqrt=+0.0
normal numbers	x	e!=0 && e!=all 1		A: $(-1)^s$ x 0.m x 2^{1-bias}	A=mm*2^N When s=1, m=0, mm=(-1)s*1.0, N=-e-bias Otherwise, mm=(-1)s*1.m/2,N=-e-bias+1 When N<=-31, A=0		When s=1, error. A=mm*2^N When e is even number, mm=(-1)s*1.m/2,N=-e-bias+1 When e is odd number, mm=(-1)s*1.m/4,N=-e-bias+2 N is even number	
					Wrap to [-0.5,0.5]	A ∈ [-1.0,1.0] N>=1, error.		

Where s represents the sign bit of data, e represents the exponent value of a floating point data, m represents the mantissa part of a floating point data.

18.5.2 MOU

All the Matrix Operation Unit parameters are in registers, so MCU can configure registers to select the operation mode. The MOU source data address and output destination address are configured in registers, when MOU is started, MOU reads data from system memory and starts computation. After calculation, all the output data will be written into system memory at the address configured in register. Intermediate data of the matrix calculations are saved in FSP local memory.

MOU and FFT shares the same arithmetic unit (FPU), so they can not work at the same time.

When MOU is working, Cordic and SE/COR/FIR can also work at the same time. So it is possible that they are both accessing system memory. So FSP internal round-robin arbiter is implemented to deal with this scenario.

The MOU engine expects the matrix data to be stored in row order in the system memory for both read and write operations. The figure below shows the address map for a 6 x 6 matrix A.

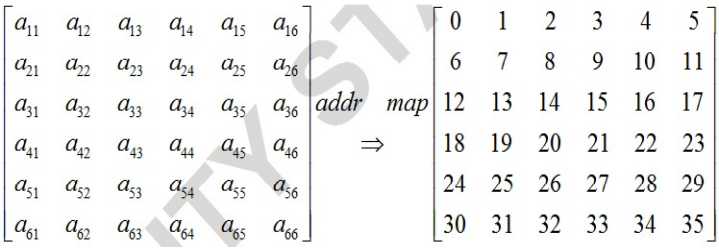


Fig 57.

The MOU engine will read & write the data through the sequence described above.

The MOU is a floating point engine. If the input data is fixed point then the data is converted to floating point by the FP conversion circuit, before being input to the MOU engine. On the output side, the floating point data is optionally converted to fixed point before being written to the system memory.

18.5.2.1 Program Flow

To perform matrix operations using MOU engine, follow these steps:

- Enable FSP clock.
- Configure MA_SRC_BASE, MB_SRC_BASE, MO_DST_BASE with input and output matrix array addresses.
- Configure MOU_SCALEA and MOU_SCALEB to matrix A and matrix B scale factor if needed.
- Configure MOU_CTRL register to start the MOU engine.
- Wait for MOU_DONE interrupt.
- Read the output and clear the interrupt.

18.5.2.2 Error Handling

There is a local floating point calculation unit (FPU1) in MOU (shared with TE), so while calculation, the validity of the data is checked and a corresponding error interrupt is generated. Besides, FSP supports fixed point and floating point data input and output, so data type conversion unit is implemented in the module. During the conversion, some error may possibly happen.

There are several types of error interrupts in MOU:

- FPU1_CALC_IN_ERR: FPU1 input data error, means internal multiplier and adder inputs are INF or NAN.
- FPU1_CALC_OUT_ERR: FPU1 output data error, means internal multiplier and adder outputs are overflow.
- FPU1_DIN_OV: only happens in TE. When TE input is floating point data, and it need to be converted to fix point data (Q31), because the TE itself is a fix point engine. This error means that the conversion process has resulted in an overflow.
- FPU1_DOUT_OV: only happens in MOU. MOU itself is a floating point engine, and its output is floating point data, when user asks to output fixed point data to system memory, conversion happens here. This error means that the conversion process has resulted in an overflow.
- FINV_DIN_ERR: only happens in MOU. Internal invertor input data error. This error implies that the input of invertor is INF or NAN.
- FINV_DOUT_ERR: only happens in MOU. Internal invertor output data error. This error implies that there is an overflow in the output.
- FINV_ZERO: only happens in MOU. Internal invertor input is zero.
- SINGULAR: only happens in MOU. This error implies that the input matrix is singular.

18.5.3 TE

software can access system memory to write the initial data and read the final results.

FSP memory only holds the intermediate data in calculation.

Data flow between TE and system memory is

- TE fetches original data from system memory
- TE keeps computing and stores or fetches the intermediate data to/from FSP mem
- TE writes back the final results to system mem for the usage of software

The TE is a fixed point engine with 20 bits of precision. So a FP conversion module is implemented here to convert floating point data to fixed point if the input is floating point. The FP conversion module is shared with MOU FP conversion module:

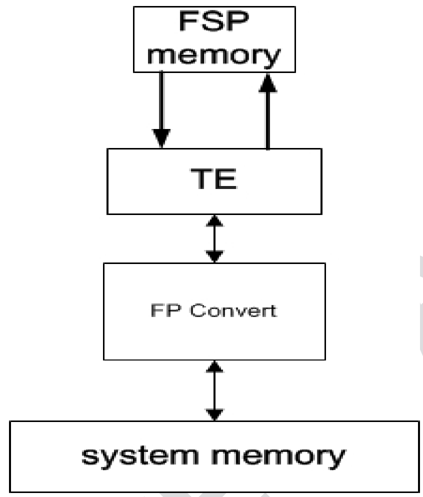


Fig 58. FP conversion module

The engine support real and complex calculation. In real number mode, the data input and output is saved sequentially in system memory. However, in complex number mode, the data in system memory is arranged repeatedly one real part number followed by one image part number.

18.5.3.1 Program Flow

To perform transform operations using TE, follow these steps::

- Enable FSP clock.
- Enable interrupt.
- Configure TE_SRC_BASE, TE_DST_BASE with input and output array addresses.
- Configure TE_CTRL register to start the TE engine.
- Wait TE_DONE interrupt.
- Read the output and clear the interrupt.

18.5.3.2 Error Handling

TE shares the internal calculation unit and data converting unit with MOU. So the error interrupt is similar to MOU.

18.5.4 Statistics Engine (SE)

The statistics Engine module is used to compute the sum, sum of squares, maximum and/or minimum of a given array. The maximum length of the array is limited to 256 elements. The module can process both floating point and fixed point arrays.

When SE is started, it reads data from system memory directly and calculates the sum, power, and determines the maximum and minimum values of an array along with their indices.

Each of the statistics operations can be enabled or disabled independently by register configuration, which can help save some power while calculation.

The SE is a floating point engine, so data type conversion unit is also implemented here to support Q31 operations.

18.5.4.1 Program Flow

To perform statistics operations using SE, follow these steps:

- Enable FSP clock.
- Configure SE_SRC_BASE with source address of an array.
- Configure SE_CTRL register to start the SE engine.
- Wait for SE_DONE interrupt.
- Read the result in SE_IDX, SE_SUM and SE_PWR registers, and clear interrupt.

18.5.4.2 Error Handling

SE, COR and FIR share the internal computation unit and data type conversion unit and in/out, so the error interrupt are similar:

- FPU0_CALC_IN_ERR: FPU0 input data error. This error implies that the internal multiplier and adder inputs are INF or NAN.
- FPU0_CALC_OUT_ERR: FPU0 output data error. This error implies that an overflow has resulted due to internal multiplier and adder operations
- FPU0_DIN_OV: This bit is always 0, since the engines themselves are floating point module, and input data will never overflow.
- FPU0_DOUT_OV: The engines themselves are floating point engine, and the output is floating point data, when user asked to output fixed point data to system memory, conversion happens here. This error means that an overflow has occurred during the conversion process.

18.5.5 Correlation Module (COR)

The address generator block in COR generates the read or write system memory address. While COR is in operation, the engine reads data from system memory, and after calculation writes back the result to system memory.

The COR is a floating point engine, so FP conversion module is implemented here to support different data types.

18.5.5.1 Program Flow

To perform correlation using the FSP, follow these steps:

- Enable FSP clock.
- Configure CX_SRC_BASE and CY_SRC_BASE with source address of 2 input arrays. And configure CO_DST_BASE with destination address of the result array.
- Configure COR_OFFSET with the offset of the two arrays.
- Configure COR_CTRL register to start the COR engine.
- Wait for COR_DONE interrupt.
- Read the result in memory and clear interrupt.

18.5.5.2 Error Handling

The error handling of the COR module is similar to SE.

18.5.6 Multi-channel Finite Impulse Response (FIR) Filter

The FIR filter module implements up to nine independent channel filters using the same computational engine. The number of taps for each of the filters can range from 1 to 16 and can be individually programmed. An internal FIR buffer is used to retain the data register content. The core of the FIR module is implemented using floating point arithmetic. CPU writes input sequence data to register, and the data format is decoded by the address decoder, which generates a signal to the FP conversion unit to convert the input data to floating point type. The register addresses for the data for the different channels is listed below:

Table 233. registers for data of different channels

Name	address	data type	channel
FIR_DAT0_FX	0x0D0	fix	0
FIR_DAT1_FX	0x0D4	fix	1
FIR_DAT2_FX	0x0D8	fix	2
FIR_DAT3_FX	0x0DC	fix	3
FIR_DAT4_FX	0x0E0	fix	4
FIR_DAT5_FX	0x0E4	fix	5
FIR_DAT6_FX	0x0E8	fix	6
FIR_DAT7_FX	0x0EC	fix	7
FIR_DAT8_FX	0x0F0	fix	8
FIR_DAT0_FL	0x100	float	0
FIR_DAT1_FL	0x104	float	1
FIR_DAT2_FL	0x108	float	2
FIR_DAT3_FL	0x10C	float	3
FIR_DAT4_FL	0x110	float	4
FIR_DAT5_FL	0x114	float	5
FIR_DAT6_FL	0x118	float	6
FIR_DAT7_FL	0x11C	float	7
FIR_DAT8_FL	0x120	float	8

The same address in the AHB is used to store the input and output data. This implies that the user writes the input data to FIR_DAT<x>_F<y>, waits until the computations are complete and then reads the output data from the same address, where x indicates the channel number and can range from 0 to 8, and y indicates the data type with F representing fixed point data and L representing floating point data.

While FIR engine gets one input data, it fetches FIR coefficients from the system memory and starts the computation. In the FSP, only the floating point data type is supported for the coefficients. A local buffer is implemented to hold the intermediate data used in the computations. If the filter does not need to maintain the state from one call to another then the buffer content should be cleared before the FIR computations. The output for each channel will be ready and will be held in output register after certain number of cycles depending on the filter length.

In order to speed up the computations within in each channel, FIR module uses ping-pong logic. The MCU should read the data from the output register once it is available and write to system memory before the next output is prepared by the engine.

18.5.6.1 Program Flow

To perform multi-channel FIR filtering, follow these steps:

- Enable FSP clock.
- Clear FIR channel buffer by setting FIR_CH<x>_BUF_CLR bit to '1' in FIR_CFG_CH<x> register (x: 0 to 8) for all the channels that will be used by the application
- Read FIR_READY bit in STATUS register. When this bit changes to '1' then that indicates that the clear operation is complete
- Configure FIR_CFG_CH<x> (x: 0 to 8) with FIR coefficient address and the tap length
- Continually write two FIR input data to FIR_DAT<x>_FX when the input data format is fixed point or FIR_DAT<x>_FL when the input data format is floating point
- Read the FIR output data from FIR_DAT<x>_FX or FIR_DAT<x>_FL register
- Write new data to FIR input data to FIR_DAT<x>_FX or FIR_DAT<x>_FL register. If this is not the last data, loop back to previous step, otherwise go to next step
- Read two FIR output data from FIR_DAT<x>_FX or FIR_DAT<x>_FL

18.5.6.2 Error Handling

The error handling in multi-channel FIR is similar to SE.

18.6 Function Description

18.6.1 CORDIC

The CORDIC module enables computation of different transcendental functions with 23 bits of accuracy. Some of the commonly used functions are implemented using custom modes, while the less frequently used functions are implemented using the native mode where some additional pre-processing and post-processing is required. Only one of these modes can be used at any given time. The three modes are given as below.

- SIN/COS
- LN/SQRT
- Native mode

18.6.1.1 SIN/COS mode

This custom mode of the CORDIC is used to compute the sine and cosine of the given input. The input data and output data should be stored into system memory, and should be word aligned.

The SIN_COS_xxxx register is used to configure the source address and the destination address. The cosine of the input data is saved in the destination address (DA) and the sine of the input data is saved at DA + 1 (word address) in system memory. The input data should be in radians and shall be divided by π prior to being input to the module.

The SIN/COS module supports Q31 and single precision floating point (IEEE 754) formats. However, the internal engine precision is 23 bits, so rightmost 9 bits of Q31 input will be ignored in calculations.

Depending on the input and the output data types, one of the following four SIN_COS registers are used:

- If both the input and the output data are in Q31 format then SIN_COS_IXOX is used.
- If the input data is in Q31 format and the output data is in floating point format then SIN_COS_IXOL is used.
- If the input data is in floating point format and the output data is in Q31 format, SIN_COS_ILOX is used.
- If both the input and the output data are in floating point format then SIN_COS_ILOL is used.

18.6.1.2 LN/SQRT mode

This custom mode of the CORDIC is used to compute the natural logarithm and the square root of the given input. The input data and output data should be stored into system memory, and should be word aligned.

The LN_SQRT_xxxx register is used to configure the source address and the destination address. The natural logarithm of the input data is saved in the destination address (DA) and the square root of the input data is saved at DA + 1 (word address) in system memory.

The LN/SQRT module supports Q31 and single precision floating point (IEEE 754) formats. However, the internal engine precision is 23 bits, so rightmost 9 bits of Q31 input will be ignored in calculations.

Depending on the input and the output data types, one of the following four LN_SQRT registers are used:

- If both the input and the output data are in Q31 format then LN_SQRT_IXOX is used.
- If the input data is in Q31 format and the output data is in floating point format then LN_SQRT_IXOL is used.
- If the input data is in floating point format and the output data is in Q31 format, LN_SQRT_ILOX is used.
- If both the input and the output data are in floating point format then LN_SQRT_ILOL is used

18.6.1.3 Native mode

In the native CORDIC mode, direct access to the core engine is enabled. The software can use this to compute the different transcendental functions using the standard CORDIC equations as described below.

$$\begin{cases} x^{(i+1)} = (x^{(i)} - \mu d_i (2^{-i} y^{(i)})) \\ y^{(i+1)} = (y^{(i)} + d_i (2^{-i} x^{(i)})) \\ z^{(i+1)} = z^{(i)} - d_i e^{(i)} \end{cases}$$

The parameter μ depends on the coordinate system and has a value of 1 for Circular coordinate system and a value of -1 for Hyperbolic coordinate system. The value of $e^{(i)}$ is given by the following equation:

$$\begin{cases} \text{Circular coordinate}: \ \mu = 1, \quad e^{(i)} = a\tan(2^{-i})/2/pi \\ \text{Hyperbolic coordinate}: \mu = -1, \quad e^{(i)} = a\tanh(2^{-i}) \end{cases}$$

And

$$\begin{cases} \text{Rotation Mode(t = 0)}: \ \mathrm{d_i} = sign(z^{(i)}) \\ \text{Vectoring Mode(t = 1)}: \mathrm{d_i} = -sign(x^{(i)}y^{(i)}) \end{cases}$$

The value of i ranges from 0 (or 1 when μ: -1) to n - 1, where n is the number of iterations. There are 5 input

x_0, y_0, z_0, μ, t

in the above cordic equation. The input parameters μ and t are decoded out by register. And x, y and z are saved in memory sequentially, which means x is stored at SA, y at SA + 1 and z at SA + 2 (here SA is the word address of source data).

After n iterations, the output can be read out from the x, y and/or z address. The following table shows the inputs and outputs for all the combinations of μ and t. The other transcendental functions can be computed by post-processing the CORDIC core output.

Table 234. Cordic equation in native mode

	t = 0	t = 1
μ = 1	$\begin{cases} x^{(n)} = K_n(x_0\cos(z_0) - y_0\sin(z_0)) \\ y^{(n)} = K_n(y_0\cos(z_0) + x_0\sin(z_0)) \\ z^{(n)} = 0 \end{cases}$	$\begin{cases} x^{(n)} = K_n(\sqrt{x_0^2 + y_0^2}) \\ y^{(n)} = 0 \\ z^{(n)} = z_0 + \tan^{-1}(\frac{y_0}{x_0}) \end{cases}$
μ = -1	$\begin{cases} x^{(n)} = L_n(x_0\cosh(z_0) + y_0\sinh(z_0)) \\ y^{(n)} = L_n(y_0\cosh(z_0) + x_0\sinh(z_0)) \\ z^{(n)} = 0 \end{cases}$	$\begin{cases} x^{(n)} = L_n(\sqrt{x_0^2 - y_0^2}) \\ y^{(n)} = 0 \\ z^{(n)} = z_0 + \tanh^{-1}(\frac{y_0}{x_0}) \end{cases}$

In the above table

$$K_n = \prod_{i=0}^{n} \frac{1}{(\cos\theta)^{(i)}} = \prod_{i=0}^{n} \sqrt{1+2^{(-2i)}}$$

When n → ∞, K_n → 1.6476, and $1/K_n$ → 0.6073.

When the number of iterations (n) is a fixed value, Kn is also a fixed value.

In FSP, the value of Kn should be 1.6467602581f, and 1/Kn should be 0.607252935009249f.

$$L_n = \prod_{i=0}^{n} \sqrt{1-2^{(-2i)}}$$

When n → ∞, L_n → 0.82816, and $1/L_n$ → 1.20750.

When the number of iterations (n) is a fixed value, Ln is also a fixed value.

In FSP, the value of Ln should be 0.82816f, and 1/Ln should be 1.20750.

Note: FSP supports either fix point or float point data, so it is optional to write fix point or float point value of Ln & Kn into system memory in native mode. If in fix point mode, Ln & Kn here should be converted to fix point data (q31) and written into system memory.

When native mode, input X0,Y0,Z0 must in range of [-1,1), otherwise, an error flag is raised. This implies that some additional pre-processing may be needed on the input data before being processed by the CORDIC core.

In native mode, output results x(n), and y(n)/z(n) can be read from DA and DA+1 word aligned addresses.

Table 235. output in native mode

mode	Native
DA	x(n)
DA+1	y(n) (t: 0) /z(n) (t: 1)

18.6.2 Matrix Operations Unit (MOU)

MOU supports different matrix operations for fixed point and floating point types. The specific operation mode to be used is defined by the OP_MODE field (4-bits) in MOU_CTRL register (offset: 0x40):

Table 236. Matrix operation mode

OP_MODE	Operation mode
0000b	INV- Matrix inverse
0001b	MULT- Matrix multiplication
0010b	TRANS- Matrix transpose
0011b	LINEAR- Linear combining
0100b	DOT MULT - Matrix dot product
otherwise	reserved

Since the input & output data format is flexible, 2 register bit MOU_DIN_FP_SEL & MOU_DOUT_FP_SEL in MOU_CTRL register are used to select the data format:

Table 237. MOU data format

MOU_DIN_FP_SEL/MOU_DOUT_FP_SEL	description
0	input/output is float point
1	input/output is fix point

18.6.2.1 Matrix inversion

Matrix inversion is supported for square matrices with size M × M. The range of M is from 2 to 9. LU decomposition is used for matrix inversion.
In inversion operation, it is possible that very small numbers might appear in calculation. To make the operation more flexible, a DIV_EPSILON bit in MOU_CTRL register is implemented to define the minimum value. If the data is smaller than this value, error interrupt will trigger. The table below maps the DIV_EPSILON field to the minimum exponent value:

Table 238. Defined minimum value in matrix inversion

DIV_EPSILON	data exponent small than
00b	0
01b	32
10b	64
11b	reserved

MOU also triggers an error interrupt if the input matrix is singular.

18.6.2.2 Matrix multiplication

Multiplication of two matrices A and B (C: AB), is only valid when the number of columns of matrix A is equal to the number of rows of matrix B. So, if the size of matrix A size is M x N, and the size of matrix B is N x K, then the matrix multiplication is supported. The size of the output matrix C is M x K. The range of M, N and K is from 1 to 9.
Matrix multiplication is given by the following equation:

$$c_{ij} = \sum_{t=1}^{n} a_{it} b_{tj}$$

(i: 1,2,…,M; j: 1,2,…,K)

Where M, N, K are configured as MAT_K, MAT_M and MAT_N fields in MOU_CTRL register. All the references to M, N, and K in the below sub-sections are also defined by these fields values.

18.6.2.3 Matrix Transpose

When matrix A with a size of M x N is transposed then the result is a matrix with size N × M. The range of M and N is from 1 to 9.
Matrix transposition is given by the following equation:

$$c_{ij} = a_{ji}$$

(i: 1,2,…,N; j: 1,2,…,M)

18.6.2.4 Matrix Linear Combination

Linear combination of two matrices A and B is supported when both the matrices have the same size (M x N). The range of M, N is from 1 to 9.
Linear combination is given by the following equation:

$$c_{ij} = \alpha \times a_{ij} + \beta \times b_{ij}$$

(i: 1,2,…,M; j: 1,2,…,N)

Here the scale factors are defined in MOU_SCALEA & MOU_SCALEB registers and they should be in floating point format.

18.6.2.5 Matrix Dot Multiply

Matrix dot multiplication represents element by element multiplication of two matrices. The size of the two input matrices and the output matrix should be the same.
The equation to compute the dot product of Matrix A (M x N) with Matrix B (M x N) is given as:

$$c_{ij} = a_{i\,j} * b_{ij}$$

where i belongs to 0~M-1, and j belongs to 0~Ñ-1.

18.6.3 Transform Engine (TE)

The TE supports different types of transform operations. The specific mode to be used is controlled by the TE_MODE field in TE_CTRL register (offset: 0x20):

Table 239. TE mode

TE_MODE	Description
00b	FFT - Fast Fourier Transform
01b	IFFT - Inverse Fast Fourier Transform
10b	DCT - Discrete Cosine Transform
11b	IDCT - Inverse Discrete Cosine Transform

The input and output mode is also configurable, which is determined by the TE_IO_MODE field in TE_CTRL register:

Table 240. TE IO mode

TE_IO_MODE	Description
00b	real input, complex output
01b	complex input, complex output
10b	real input, real output
11b	complex input, real output

The number of points in the transform is also configurable, which is determined by the TE_PTS field in TE_CTRL register:

Table 241. Points of transform

TE_PTS	Description
00b	64 point

Table 241. Points of transform

TE_PTS	Description
01b	128 point
10b	256 point
11b	reserved

TE input and output data type is also selectable, which is determined by TE_DIN_FP_SEL & TE_DOUT_FP_SEL in TE_CTRL:

Table 242.

TE_DIN_FP_SEL/TE_DOUT_FP_SEL	Description
0	input/output is fix point
1	input/output is float point

Since TE treats input data as a <s, 0, 19> data, and if the input data is float point data, it may exceed this range, Therefore, a scale function is implemented in this module to scale the input data to a reasonable range, and after computation the output data will be scaled back. The scale factor is determined by the TE_SCALE field in TE_CTRL register.

The scaled data is derived from the input data and the scale factor using the following equation:

Data_scaled: data_in * 2 ^ TE_SCALE.

TE_SCALE is a signed value. So, it can scale up the input data if it is too small or scale down the input data if it is too big.

18.6.3.1 FFT

Assume FFT size is N, input x(n) and output X(k) are of length N, X(k) can be expressed as,

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} e^{-j\frac{2\pi nk}{N}} x(n)$$

Where k: 0, 1, 2, …, N-1. In the FSP implementation, FFT size can be 64, 128 or 256 points. Radix-4 butterfly is adopted.

Remark: In order to make sure that the FFT result range is in [-1,1)□the FFT expression includes a 1/N factor. This is different from the conventional definition of FFT where this factor is only included in IFFT definition.

18.6.3.2 IFFT

IFFT is the inverse transformation of FFT, assume IFFT size is N, input X(k) and output x(n) are of length N, x(n) can be expressed as,

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} e^{j\frac{2\pi nk}{N}} X(k)$$

where k: 0,1,2, …, N-1. In FSP implementation, IFFT size can be 64, 128 or 256 points. Radix-4 butterfly is adopted here. The implementation of IFFT is same as FFT except for the sign of the exponential term of

$$e^{j\frac{2\pi nk}{N}}$$

.

18.6.3.3 DCT

DCT transformation in FSP is based on the DCT-II form. It is implemented using FFT. Assume DCT size is N, input x(n) and output X(k) are of length of N, the k-th bin of DCT X(k) can be expressed as,

$$X(k) = \sum_{n=0}^{N-1} x(n) \cos\left[\frac{\pi}{N} k(n + \frac{1}{2})\right]$$

where k: 0,1,2, …, N-1.

The algorithm of DCT includes following steps.

- Re-sort input sequence as, x'=[x_0,x_2,…$x_{N/2-1}$,x_{N-2},x_{N-4},…,x_1];
- Perform N/2 point FFT on x' and obtain X'=FFT(x');
- Multiply X' by rotation factor,
  $$w = [1, e^{-j\frac{\pi}{2N}}, ..., e^{-j\frac{(N-2)\pi}{2N}}, e^{-j\frac{(N-1)\pi}{2N}}]$$
  , so the final output of DCT is
  X(k)=FFT_(x') (k)*e^(-j kπ/2N)

18.6.3.4 IDCT

IDCT transformation is the inverse of DCT and its implementation in our FSP is the IDCT-II form, which is most commonly used in engineering applications. Assuming IDCT has size of N, the input X(k) and output x(n) are both vectors of length N, the n-th output of IDCT, x(n) is expressed as,

$$x(n) = \sum_{k=0}^{N-1} X(k) \cos\left(\frac{\pi}{N} k(n + \frac{1}{2})\right) \qquad (n = 0,1,..., N-1)$$

The implementation of IDCT includes 3 steps which are the inverse operations of the ones described in DCT from step c to step a.

Please note that the IDCT(DCT(x)) is not equal to x, since we omitted the normalization.

18.6.4 Statistics Engine (SE)

SE supports the following operations:

- Sum of an array
- Power (squared sum) of an array
- Max/min value of an array as well as the corresponding index

To enable each function, PWR_EN, SUM_EN, MAX_IDX_EN & MIN_IDX_EN bit in SE_CTRL register should be set to 1.

While searching for the index of the maximum or minimum value of an array, it is possible that the maximum/minimum value is not unique. In those instances, the SE can output the smallest index or the largest index of the subset of the array elements which are equal to the maximum or the minimum value. This is determined by the MAX_SEL or MIN_SEL bit in SE_CTRL register. The table below maps the value of these fields to their definition:

Table 243. SE index of max/min

MAX_SEL/MIN_SEL	Description
0	Smallest index of the subset
1	Largest index of the subset

The input array length is configurable and is defined by the SE_LEN field in TE_CTRL register. The SE_LEN field is 8 bits long and can range from 0 to 255 which translates to an array length ranging from 1 to 256 elements.

SE input & output data type is also selectable, which is decided by SE_DIN_FP_SEL & SE_DOUT_FP_SEL in TSE_CTRL.

Table 244. SE data type

SE_DIN_FP_SEL/SE_DOUT_FP_SEL	Description
0	input/output is floating point
1	input/output is fixed point

When the output data type is fixed point, a Q31 data can not hold sum or power of an array. Therefore the SE module will hold the actual value scaled by 2^-8 in the SUM and PWR registers.

18.6.4.1 Sum of an array (Sum)

The sum of an array is given by the following equation

$$c = \sum_{i=0}^{N-1} a_i$$

where N is the length of the array.

The mean value of an array can be computed by dividing the sum by the N in software.

18.6.4.2 Sum of squares of an array (Power)

The sum of squares of an array is given by the following equation.

$$c = \sum_{i=0}^{N-1} a_i \times a_i$$

where N is the length of the array.

18.6.4.3 The index of minimum and maximum data

The maximum and minimum of an array is given by the following equation

$$a_j = \max(a_i), \quad a_t = \min(a_i) \quad (i = 0,1,2,\ldots,N-1)$$

$$c = j, \quad d = t$$

Where c is the index of maximum value, d is the index of minimum value. The maximum and minimum values are determined over the signed values.

18.6.5 COR

Correlation of two sequences X and Y as implemented in FSP is given by the following equation.

$$Z(i) = \sum_{j=0}^{N_y-1} X(i+j)Y(j)$$

where i: $0,1,2,\ldots,N_x-N_y+1$

Nx and Ny are the length of the sequences X and Y, with Nx >= Ny . The correlation values are computed in the overlap part of sequences X, and Y. So the number of correlation points are Nx - Ny + 1.

The length of x and y are defined by COR_X_LEN & COR_Y_LEN fields in COR_CTRL register.

The data format is also selectable, which are defined by COR_DOUT_FP_SEL & COR_DIN_FP_SEL in COR_CTRL register:

Table 245. COR data format

COR_DIN_FP_SEL/COR_DOUT_FP_SEL	Description
0	input/output is float point
1	input/output is fix point

18.6.6 Finite Impulse Response (FIR) Filter

The equation to compute the output of an FIR function is given below:

$$y(n) = \sum_{m=0}^{L-1} h(m)x(n-m)$$

Where L is the number of taps, which is defined by FIR_CH<x>_TAP_LEN field in FIR_CFG_CH<x> register, where x is from 0 to 8 and h(m) are the filter coefficients, whose base address is defined by FIR_CH<x>_COEF_BASE in FIR_CFG_CH<x> register.

To clear the FIR buffer, set FIR_CH<x>_BUF_CLR in FIR_CFG_CH<x> register to '1'. After the clear is done, this bit will be set to '0', otherwise it will remain '1'.

An FIR_BUF_CLR_ALL bit is also provided in FIR_CFG_CH0 register to clear all of the channels' data registers in buffer.

18.7 Register description

Table 246. FSP Register overview (base address 0x4008 8000)

Name	Access	Offset	Description	Reset value	Section
SYS_CTRL	WOO	0x000	FSP system control register	0x0	18.7.1
STATUS	r	0x004	FSP status register	0x0	18.7.2
INT	W1C	0x008	FSP interrupt register	0x0	18.7.3
INTEN	rw	0x00C	FSP interrupt enable register	0x0	18.7.4
TE_CTRL	rw	0x020	Transform engine control register	0x0	18.7.5

Table 246. FSP Register overview (base address 0x4008 8000)

Name	Access	Offset	Description	Reset value	Section
TE_SRC_BASE	rw	0x024	Transform Engine source data memory base register	0x0	18.7.6
TE_DST_BASE	rw	0x028	Transform Engine destination data memory base register	0x0	18.7.7
MOU_CTRL	rw	0x040	matrix operation unit control register	0x1110000	18.7.8
MA_SRC_BASE	rw	0x044	matrix A source data memory base register	0x0	18.7.9
MB_SRC_BASE	rw	0x048	matrix B source data memory base register	0x0	18.7.9
MO_DST_BASE	rw	0x04C	matrix output data memory base register	0x0	18.7.10
MOU_SCALEA	rw	0x050	scale coefficient A register	0x0	18.7.11
MOU_SCALEB	rw	0x054	scale coefficient B register	0x0	18.7.11
SE_CTRL	rw	0x060	static engine control register	0x1c0	18.7.12
SE_SRC_BASE	rw	0x064	statistic engine source data base register	0x0	18.7.13
SE_IDX	r	0x068	max or min data index register	0x0	18.7.14
SE_SUM	r	0x06C	array summary result register	0x0	18.7.15
SE_PWR	r	0x070	array power result register	0x0	18.7.16
COR_CTRL	rw	0x080	correlation control register	0x300	18.7.17
CX_SRC_BASE	rw	0x084	correlation x sequence base register	0x0	18.7.18
CY_SRC_BASE	rw	0x088	correlation y sequence base register	0x0	18.7.18
CO_DST_BASE	rw	0x08C	correlation output sequence base register	0x0	18.7.19
COR_OFFSET	rw	0x090	correlation offset register	0x0	18.7.20
FIR_CFG_CH0	rw	0x0A0	FIR channel 0 configuration register	0x0	18.7.21
FIR_CFG_CH1	rw	0x0A4	FIR channel 1 configuration register	0x0	18.7.22
FIR_CFG_CH2	rw	0x0A8	FIR channel 2 configuration register	0x0	18.7.22
FIR_CFG_CH3	rw	0x0AC	FIR channel 3 configuration register	0x0	18.7.22
FIR_CFG_CH4	rw	0x0B0	FIR channel 4 configuration register	0x0	18.7.22

Table 246. FSP Register overview (base address 0x4008 8000)

Name	Access	Offset	Description	Reset value	Section
FIR_CFG_CH5	rw	0x0B4	FIR channel 5 configuration register	0x0	18.7.22
FIR_CFG_CH6	rw	0x0B8	FIR channel 6 configuration register	0x0	18.7.22
FIR_CFG_CH7	rw	0x0BC	FIR channel 7 configuration register	0x0	18.7.22
FIR_CFG_CH8	rw	0x0C0	FIR channel 8 configuration register	0x0	18.7.22
FIR_DAT0_FX	rw	0x0D0	FIR channel 0 fix point data input & output register	0x0	18.7.23
FIR_DAT1_FX	rw	0x0D4	FIR channel 1 fix point data input & output register	0x0	18.7.23
FIR_DAT2_FX	rw	0x0D8	FIR channel 2 fix point data input & output register	0x0	18.7.23
FIR_DAT3_FX	rw	0x0DC	FIR channel 3 fix point data input & output register	0x0	18.7.23
FIR_DAT4_FX	rw	0x0E0	FIR channel 4 fix point data input & output register	0x0	18.7.23
FIR_DAT5_FX	rw	0x0E4	FIR channel 5 fix point data input & output register	0x0	18.7.23
FIR_DAT6_FX	rw	0x0E8	FIR channel 6 fix point data input & output register	0x0	18.7.23
FIR_DAT7_FX	rw	0x0EC	FIR channel 7 fix point data input & output register	0x0	18.7.23
FIR_DAT8_FX	rw	0x0F0	FIR channel 8 fix point data input & output register	0x0	18.7.23
FIR_DAT0_FL	rw	0x100	FIR channel 0 float point data input & output register	0x0	18.7.24
FIR_DAT1_FL	rw	0x104	FIR channel 1 float point data input & output register	0x0	18.7.24
FIR_DAT2_FL	rw	0x108	FIR channel 2 float point data input & output register	0x0	18.7.24
FIR_DAT3_FL	rw	0x10C	FIR channel 3 float point data input & output register	0x0	18.7.24
FIR_DAT4_FL	rw	0x110	FIR channel 4 float point data input & output register	0x0	18.7.24
FIR_DAT5_FL	rw	0x114	FIR channel 5 float point data input & output register	0x0	18.7.24
FIR_DAT6_FL	rw	0x118	FIR channel 6 float point data input & output register	0x0	18.7.24
FIR_DAT7_FL	rw	0x11C	FIR channel 7 float point data input & output register	0x0	18.7.24
FIR_DAT8_FL	rw	0x120	FIR channel 8 float point data input & output register	0x0	18.7.24
SIN_COS_IXOX	w	0x140	sin & cos input fix output fix mode data address register	0x0	18.7.25

Table 246. FSP Register overview (base address 0x4008 8000)

Name	Access	Offset	Description	Reset value	Section
SIN_COS_IXOL	w	0x144	sin & cos input fix output float mode data address register	0x0	18.7.26
SIN_COS_ILOX	w	0x148	sin & cos input float output fix mode data address register	0x0	18.7.27
SIN_COS_ILOL	w	0x14C	sin & cos input float output float mode data address register	0x0	18.7.28
LN_SQRT_IXOX	w	0x150	LN & sqrt input fix output fix mode data address register	0x0	18.7.29
LN_SQRT_IXOL	w	0x154	LN & sqrt input fix output float mode data address register	0x0	18.7.30
LN_SQRT_ILOX	w	0x158	LN & sqrt input float output fix mode data address register	0x0	18.7.31
LN_SQRT_ILOL	w	0x15C	LN & sqrt input float output float mode data address register	0x0	18.7.32
CORDIC_T0UP_IXOX	w	0x160	native cordic input fix output fix, t=0, u=1 mode data address register	0x0	18.7.33
CORDIC_T0UP_IXOL	w	0x164	native cordic input fix output float, t=0, u=1 mode data address register	0x0	18.7.34
CORDIC_T0UP_ILOX	w	0x168	native cordic input float output fix, t=0, u=1 mode data address register	0x0	18.7.35
CORDIC_T0UP_ILOL	w	0x16C	native cordic input float output float, t=0, u=1 mode data address register	0x0	18.7.36
CORDIC_T0UN_IXOX	w	0x170	native cordic input fix output fix, t=0, u=-1 mode data address register	0x0	18.7.37
CORDIC_T0UN_IXOL	w	0x174	native cordic input fix output float, t=0, u=-1 mode data address register	0x0	18.7.38
CORDIC_T0UN_ILOX	w	0x178	native cordic input float output fix, t=0, u=-1 mode data address register	0x0	18.7.39
CORDIC_T0UN_ILOL	w	0x17C	native cordic input float output float, t=0, u=-1 mode data address register	0x0	18.7.40
CORDIC_T1UP_IXOX	w	0x180	native cordic input fix output fix, t=1, u=1 mode data address register	0x0	18.7.33
CORDIC_T1UP_IXOL	w	0x184	native cordic input fix output float, t=1, u=1 mode data address register	0x0	18.7.34
CORDIC_T1UP_ILOX	w	0x188	native cordic input float output fix, t=1, u=1 mode data address register	0x0	18.7.35

Table 246. FSP Register overview (base address 0x4008 8000)

Name	Access	Offset	Description	Reset value	Section
CORDIC_T1UP_ILOL	w	0x18C	native cordic input float output float, t=1, u=1 mode data address register	0x0	18.7.36
CORDIC_T1UN_IXOX	w	0x190	native cordic input fix output fix, t=1, u=-1 mode data address register	0x0	18.7.37
CORDIC_T1UN_IXOL	w	0x194	native cordic input fix output float, t=1, u=-1 mode data address register	0x0	18.7.38
CORDIC_T1UN_ILOX	w	0x198	native cordic input float output fix, t=1, u=-1 mode data address register	0x0	18.7.39
CORDIC_T1UN_ILOL	w	0x19C	native cordic input float output float, t=1, u=-1 mode data address register	0x0	18.7.40

18.7.1 FSP system control register

Table 247. FSP system control register (SYS_CTRL, offset: 0x000) bit description

Bit	Symbol	Description	Reset Value	Access
0:0	TE_ABORT	Transform Engine abort; write 1 to abort	0	WOO
1:1	MOU_ABORT	Matrix Operation Unit abort; write 1 to abort	0	WOO
2:2	SCF_ABORT	SE COR FIR abort; write 1 to abort	0	WOO
31:3	-	Reserved	-	-

18.7.2 FSP status register

Table 248. FSP status register (STATUS, offset: 0x004) bit description

Bit	Symbol	Description	Reset Value	Access
0	FPU0_BUSY	SE COR FIR is in processing busy	0	r
1	FPU1_BUSY	TE MOU is in processing busy	0	r
2	FIR_READY	FIR output buffer is not empty which is valid for read	0	r
31:3	-	Reserved	-	-

18.7.3 FSP interrupt register

Table 249. FSP interrupt register (INT, offset: 0x008) bit description

Bit	Symbol	Description	Reset Value	Access
0	TE_DONE_INT	Transform engine done interrupt	0	W1C
1	MOU_DONE_INT	Matrix operation unit done interrupt	0	W1C
2	SE_DONE_INT	Statistic engine done interrupt	0	W1C
3	COR_DONE_INT	Correlation done interrupt	0	W1C
7:4	-	Reserved	-	-
8	FPU0_CALC_IN_ERR_INT	SE COR FIR calculation input data error interrupt	0	W1C
9	FPU0_CALC_OUT_ERR_INT	SE COR FIR calculation output data error interrupt	0	W1C
10	FPU0_DIN_OV_INT	SE COR FIR input data overflow interrupt (always 0)	0	W1C
11	FPU0_DOUT_OV_INT	SE COR FIR output data overflow interrupt	0	W1C
12	SINGULAR_INT	MOU singular interrupt	0	W1C
15:13	-	Reserved	-	-
16	FPU1_CALC_IN_ERR_INT	MOU TE calculation input data error interrupt	0	W1C
17	FPU1_CALC_OUT_ERR_INT	MOU TE calculation output data error interrupt	0	W1C
18	FPU1_DIN_OV_INT	MOU TE input data overflow interrupt	0	W1C
19	FPU1_DOUT_OV_INT	MOU TE output data overflow interrupt	0	W1C
20	FINV_DIN_ERR_INT	FINV input data is inf or nan	0	W1C
21	FINV_DOUT_OV_INT	FINV output data overflow	0	W1C
22	FINV_ZERO_INT	FINV input data is zero	0	W1C

Table 249. FSP interrupt register (INT, offset: 0x008) bit description *…continued*

Bit	Symbol	Description	Reset Value	Access
23	-	Reserved	-	-
24	TALU_DIN_ERR_INT	TALU input data error interrupt	0	W1C
25	TALU_DOUT_ERR_INT	TALU output data error interrupt	0	W1C
26	TALU_CALC_ERR_INT	TALU calculation error interrupt	0	W1C
30:27	-	Reserved	-	-
31	FSP_INT	Or signal of all FSP function interrupt in this register	0	r

18.7.4 FSP interrupt enable register

Table 250. FSP interrupt enable register (INTEN, offset: 0x00C) bit description

Bit	Symbol	Description	Reset Value	Access
0	TE_DONE_INTEN	Transform engine done interrupt enable	0	rw
1	MOU_DONE_INTEN	Matrix operation unit done interrupt enable	0	rw
2	SE_DONE_INTEN	Statistic engine done interrupt enable	0	rw
3	COR_DONE_INTEN	Correlation done interrupt enable	0	rw
7:4	-	Reserved	-	-
8	FPU0_CALC_IN_ERR_INTEN	SE COR FIR calculation input data error interrupt enable	0	rw
9	FPU0_CALC_OUT_ERR_INTEN	SE COR FIR calculation output data error interrupt enable	0	rw
10	FPU0_DIN_OV_INTEN	SE COR FIR input data overflow interrupt enable	0	rw
11	FPU0_DOUT_OV_INTEN	SE COR FIR output data overflow interrupt enable	0	rw
12	SINGULAR_INTEN	MOU singular interrupt enable	0	rw
15:13	-	Reserved	-	-

Table 250. FSP interrupt enable register (INTEN, offset: 0x00C) bit description ...continued

Bit	Symbol	Description	Reset Value	Access
16	FPU1_CALC_IN_ERR_INTEN	MOU TE calculation input data error interrupt enable	0	rw
17	FPU1_CALC_OUT_ERR_INTEN	MOU TE calculation output data error interrupt enable	0	rw
18	FPU1_DIN_OV_INTEN	MOU TE input data overflow interrupt enable	0	rw
19	FPU1_DOUT_OV_INTEN	MOU TE output data overflow interrupt enable	0	rw
20	FINV_DIN_ERR_INTEN	FINV data input is inf or nan interrupt enable	0	rw
21	FINV_DOUT_OV_INTEN	FINV data output overflow interrupt enable	0	rw
22	FINV_ZERO_INTEN	FINV input is zero interrupt enable	0	rw
23	-	Reserved	-	-
24	TALU_DIN_ERR_INTEN	TALU input data error interrupt enable	0	rw
25	TALU_DOUT_ERR_INTEN	TALU output data error interrupt enable	0	rw
26	TALU_CALC_ERR_INTEN	TALU calculation error interrupt enable	0	rw
30:27	-	Reserved	-	-
31	FSP_INTEN	Or signal of all FSP function interrupt in this register enable	0	rw

18.7.5 FSP Transform Engine control register

Table 251. FSP Transform Engine control register (TE_CTRL, offset: 0x020) bit description

Bit	Symbol	Description	Reset Value	Access
1:0	TE_MODE	TE compute mode; 0:FFT; 1:IFFT; 2:DCT; 3:IDCT;	0	rw
3:2	TE_IO_MODE	TE input & output mode select; 0:real input, complex output; 1:complex input, complex output; 2:real input, real output; 3:complex input, real output;	0	rw
5:4	TE_PTS	TE compute point; 0:64 points; 1:128 points; 2:256 points;	0	rw
6	TE_DIN_FP_SEL	TE input data format select; 0:fix; 1:float;	0	rw
7	TE_DOUT_FP_SEL	TE output data format select; 0:fix; 1:float;	0	rw
15:8	TE_SCALE	TE scale	0	rw
23:16	-	Reserved	-	-
26:24	TE_PAUSE_LVL	Transform Engine stop level for debug use only.	0	rw
31:27	-	Reserved	-	-

18.7.6 TE source data memory base register

Table 252. TE source data memory base register (TE_SRC_BASE, offset: 0x024) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	TE_SRC_BASE	TE source data memory base address	0	rw
31:17	-	Reserved	-	-

18.7.7 TE destination data memory base register

Table 253. TE destination data memory base register (TE_DST_BASE, offset: 0x028) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	TE_DST_BASE	TE destination data memory base address	0	rw
31:17	-	Reserved	-	-

18.7.8 FSP MOU control register

Table 254. FSP MOU control register (MOU_CTRL, offset: 0x040) bit description

Bit	Symbol	Description	Reset Value	Access
3:0	OP_MODE	MOU operation mode; 0:inversion; 1:matrix multiply; 2:transposition; 3:linear operation; 4:dot multiply;	0	rw
7:4	-	Reserved	-	-
8	MOU_DIN_FP_SEL	MOU data input format select£[a] 0:fix; 1:float;	0	rw
9	MOU_DOUT_FP_SEL	MOU data output format select£[a] 0:fix; 1:float;	0	rw
15:10	-	Reserved	-	-
19:16	MAT_M	MOU Matrix column	1	rw
23:20	MAT_N	MOU Matrix row only valid when matrix's column is not equal to row	1	rw
27:24	MAT_K	MOU Matrix row only valid when matrix mult operation	1	rw
29:28	DIV_EPSILON	When the data exponent is small than DIV_EPSILON the inverse operation will output a error signal.	0	rw
30	LU_STOP	Stop at LU	0	rw
31	UINV_STOP	stop at U-Matrix inverse	0	rw

18.7.9 Matrix A/B source data memory base register

Table 255. matrix A/B source data memory base register (MA_SRC_BASE/MB_SRC_BASE, offset: 0x048) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	MA_SRC_BASE	Matrix A source data memory base address	0	rw
31:17	-	Reserved	-	-

18.7.10 Matrix output data memory base register

Table 256. matrix output data memory base register (MO_DST_BASE, offset: 0x04C) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	MB_SRC_BASE	Matrix Operation Unit output data destination memory base address	0	rw
31:17	-	Reserved	-	-

18.7.11 MOU scale coefficient A/B register

Table 257. MOU scale coefficient A/B register (MOU_SCALEA/B, offset: 0x054) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	MOU_SCALEA/B	MOU scale coefficient A/B	0	rw

18.7.12 FSP SE control register

Table 258. FSP SE control register (SE_CTRL, offset: 0x060) bit description

Bit	Symbol	Description	Reset Value	Access
0	MIN_SEL	Minimum value selection£[a] 0:first one; 1:last one;	0	rw
1	MAX_SEL	Maximum value selection0 the first one1 the last one	0	rw
2	MIN_IDX_EN	Minimum value index calculation enable	0	rw
3	MAX_IDX_EN	Maximum value index calculation enable	0	rw
4	SUM_EN	Summary calculation enable	0	rw
5	PWR_EN	Power calculation enable	0	rw

Table 258. FSP SE control register (SE_CTRL, offset: 0x060) bit description *…continued*

Bit	Symbol	Description	Reset Value	Access
6	SE_DIN_FP_SEL	SE data input format select; 0:fix; 1:float;	1	rw
7	SE_DOUT_FP_SEL	SE data output format select	1	rw
15:8	-	Reserved	-	-
23:16	SE_LEN	Statistic engine length	0	rw
31:24	-	Reserved	-	-

18.7.13 SE source data memory base register

Table 259. SE source data memory base register (SE_SRC_BASE, offset: 0x064) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	SE_SRC_BASE	Statistic engine source data base address	0	rw
31:17	-	Reserved	-	-

18.7.14 SE max or min data index register

Table 260. Max or Min dta index register (SE_IDX_BASE, offset: 0x068) bit description

Bit	Symbol	Description	Reset Value	Access
7:0	SE_MIN_IDX	Minimum data index of an array	0	r
15:8	-	Reserved	-	-
23:16	SE_MAX_IDX	Maximum data index of an array	0	r
31:24	-	Reserved	-	-

18.7.15 SE array summary result register

Table 261. SE array summary result register (SE_SUM, offset: 0x06C) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	SE_SUM	Summary of an array	0	r

18.7.16 SE array power result register

Table 262. SE array power result register (SE_PWR, offset: 0x07C) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	SE_PWR	Power value of an array	0	r

18.7.17 FSP COR control register

Table 263. FSP COR control register (COR_CTRL, offset: 0x080) bit description

Bit	Symbol	Description	Reset Value	Access
7:0	-	Reserved	-	-
8	COR_DIN_FP_SEL	COR input data format select; 0:fix; 1:float;	1	rw
9	COR_DOUT_FP_SEL	COR output data format select	1	rw
15:10	-	Reserved	-	-
23:16	COR_X_LEN	The length of X sequence to be Correlator 0-255	0	rw
31:24	COR_Y_LEN	The length of Y sequence to be Correlator 0-255	0	rw

18.7.18 Correlation X/Y sequence base register

Table 264. correlation X/Y sequence base register (CX/Y_SRC_BASE, offset: 0x084/0x088) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	COR_X/Y_ADDR	The base address of X sequence to be Correlator	0	rw
31:17	-	Reserved	-	-

18.7.19 Correlation output sequence base register

Table 265. correlation output sequence base register (CX/Y_SRC_BASE, offset: 0x08C) bit description

Bit	Symbol	Description	Reset Value	Access
16:0	COR_DST_BASE	correlation output data destination address base	0	rw
31:17	-	Reserved	-	-

18.7.20 Correlation offset register

Table 266. correlation offset register (COR_OFFSET, offset: 0x090) bit description

Bit	Symbol	Description	Reset Value	Access
7:0	COR_X_OFFSET	COR input X SEQ offset 0-255	0	rw
15:8	COR_Y_OFFSET	COR input Y SEQ offset 0-255	0	rw
31:16	-	Reserved	-	-

18.7.21 FIR channel 0 configuration register

Table 267. FIR channel 0 configuration register (FIR_CFG_CH0, offset: 0x0A0) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	FIR_CH0_COEF_BASE	FIR channel 0 coefficient base address	0	rw
19:16	FIR_CH0_TAP_LEN	FIR channel 0 tap length the register value equals to real tap length minus 1.	0	rw
29:20	-	Reserved	-	-
30	FIR_BUF_CLR_ALL	clear all FIR buffer	0	WOO
31	FIR_CH0_BUF_CLR	FIR channel 0 buffer clear	0	WOO

18.7.22 FIR channel [1,8]configuration register

Table 268. FIR channel [1,8]configuration register (FIR_CFG_CH[1,8], offset: 0x0A4/0x0A8/0x0AC/0x0B0/0x0B4/0x0B8/0x0BC/0x0C0) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	FIR_CHx_COEF_BASE	FIR channel x coefficient base address; x=[1,8]	0	rw
19:16	FIR_CHx_TAP_LEN	FIR channel x tap length;x=[1,8]	0	rw
30:20	-	Reserved	-	-
31	FIR_CHx_BUF_CLR	FIR channel x buffer clear;x=[1,8]	0	WOO

18.7.23 FIR channel x fix-point data register

Table 269. FIR channel x fix-point data register (FIR_DATx_FX, offset: 0x0D0) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	FIR_DATx_FX	FIR channel x fix data, x=[0,8]	0	rw

18.7.24 FIR channel x float data register

Table 270. FIR channel x float data register (FIR_DATx_FL, offset: 0x100) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	FIR_DATx_FL	FIR channel x float data, x=[0,8]	0	rw

18.7.25 Sin & cos IXOX mode data address register

Table 271. Sin & cos IXOX mode data address register (SIN_COS_IXOX, offset: 0x140) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	SIN_COS_IXOX_SRC	SIN_COS input data source address. Input fix output fix	0	w
31:16	SIN_COS_IXOX_DST	SIN_COS output data destination address. Input fix output fix.	0	w

18.7.26 Sin & cos IXOL mode data address register

Table 272. Sin & cos IXOL mode data address register (SIN_COS_IXOL, offset: 0x144) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	SIN_COS_IXOL_SRC	SIN_COS input data source word address. Input fix output float	0	w
31:16	SIN_COS_IXOL_DST	SIN_COS output data destination word address. Input fix output float.	0	w

18.7.27 Sin & cos ILOX mode data address register

Table 273. Sin & cos ILOX mode data address register (SIN_COS_ILOX, offset: 0x148) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	SIN_COS_ILOX_SRC	SIN_COS input data source word address. Input float output fix	0	w
31:16	SIN_COS_ILOX_DST	SIN_COS output data destination word address. Input float output fix.	0	w

18.7.28 Sin & cos ILOL mode data address register

Table 274. Sin & cos ILOL mode data address register (SIN_COS_ILOL, offset: 0x14C) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	SIN_COS_ILOL_SRC	SIN_COS input data source word address. Input float output float	0	w
31:16	SIN_COS_ILOL_DST	SIN_COS output data destination word address. Input float output float	0	w

18.7.29 LN &SQRT IXOX mode data address register

Table 275. LN & SQRT IXOX mode data address register (LN_SQRT_IXOX, offset: 0x150) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	LN_SQRT_IXOX_SRC	LN_SQRT input data source word address. Input fix output fix	0	w
31:16	LN_SQRT_IXOX_DST	LN_SQRT output data destination word address. Input fix output fix.	0	w

18.7.30 LN &SQRT IXOL mode data address register

Table 276. LN & SQRT IXOL mode data address register (LN_SQRT_IXOL, offset: 0x154) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	LN_SQRT_IXOL_SRC	LN_SQRT input data source word address. Input fix output float	0	w
31:16	LN_SQRT_IXOL_DST	LN_SQRT output data destination word address. Input fix output float.	0	w

18.7.31 LN &SQRT ILOX mode data address register

Table 277. LN & SQRT ILOX mode data address register (LN_SQRT_ILOX, offset: 0x158) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	LN_SQRT_ILOX_SRC	LN_SQRT input data source word address. Input float output fix	0	w
31:16	LN_SQRT_ILOX_DST	LN_SQRT output data destination word address. Input float output fix.	0	w

18.7.32 LN &SQRT ILOL mode data address register

Table 278. LN & SQRT ILOL mode data address register (LN_SQRT_ILOL, offset: 0x15C) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	LN_SQRT_ILOL_SRC	LN_SQRT input data source word address. Input float output float	0	w
31:16	LN_SQRT_ILOL_DST	LN_SQRT output data destination word address. Input float output float	0	w

18.7.33 Cordic T0/1UP IXOX mode data address register

Table 279. Cordic T0/1 UP IXOX mode data address register (CORDIC_T0/1UP_IXOX, offset = 0x160/0x180) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0/1UP_IXOX_SRC	CORDIC_T0/1UP_IXOX input data source word address. Input fix output fix	0	w
31:16	CORDIC_T0/1UP_IXOX_DST	CORDIC_T0/1UP_IXOX output data destination word address. X is saved at here Y is saved at word address CORDIC_T0/1UP_IXOX_DST+1. Input fix output fix	0	w

18.7.34 Cordic T0/1UP IXOL mode data address register

Table 280. Cordic T0/1 UP IXOL mode data address register (CORDIC_T0/1UP_IXOL, offset = 0x164/0x184) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UP_IXOL_SRC	CORDIC_T0UP_IXOL input data source word address. Input fix output float	0	w
31:16	CORDIC_T0UP_IXOL_DST	CORDIC_T0UP_IXOL output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UP_IXOL_DST+1. Input fix output float	0	w

18.7.35 Cordic T0/1UP ILOX mode data address register

Table 281. Cordic T0/1 UP ILOX mode data address register (CORDIC_T0/1UP_ILOX, offset = 0x168/0x188) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UP_ILOX_SRC	CORDIC_T0UP_ILOX input data source word address. Input float output fix	0	w
31:16	CORDIC_T0UP_ILOX_DST	CORDIC_T0UP_ILOX output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UP_ILOX_DST+1. Input float output fix	0	w

18.7.36 Cordic T0/1UP ILOL mode data address register

Table 282. Cordic T0/1 UP ILOL mode data address register (CORDIC_T0/1UP_ILOL, offset = 0x17C/0x18C) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UP_ILOL_SRC	CORDIC_T0UP_ILOL input data source word address. Input float output float	0	w
31:16	CORDIC_T0UP_ILOL_DST	CORDIC_T0UP_ILOL output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UP_ILOL_DST+1. Input float output float	0	w

18.7.37 Cordic T0/1UN IXOX mode data address register

Table 283. Cordic T0/1 UN IXOX mode data address register (CORDIC_T0/1UN_IXOX, offset = 0x170/0x190) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UN_IXOX_SRC	CORDIC_T0UN_IXOX input data source word address. Input fix output fix	0	w
31:16	CORDIC_T0UN_IXOX_DST	CORDIC_T0UN_IXOX output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UN_IXOX_DST+1. Input fix output fix	0	w

18.7.38 Cordic T0/1UN IXOL mode data address register

Table 284. Cordic T0/1 UN IXOL mode data address register (CORDIC_T0/1UN_IXOL, offset = 0x174/0x194) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UN_IXOL_SRC	CORDIC_T0UN_IXOL input data source word address. Input fix output float	0	w
31:16	CORDIC_T0UN_IXOL_DST	CORDIC_T0UN_IXOL output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UN_IXOL_DST+1. Input fix output float	0	w

18.7.39 Cordic T0/1UN ILOX mode data address register

Table 285. Cordic T0/1 UN ILOX mode data address register (CORDIC_T0/1UN_ILOX, offset = 0x178/0x198) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UN_ILOX_SRC	CORDIC_T0UN_ILOX input data source word address. Input float output fix	0	w
31:16	CORDIC_T0UN_ILOX_DST	CORDIC_T0UN_ILOX output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UN_ILOX_DST+1. Input float output fix	0	w

18.7.40 Cordic T0/1UN ILOL mode data address register

Table 286. Cordic T0/1 UN ILOL mode data address register (CORDIC_T0/1UN_ILOL, offset = 0x17C/0x19C) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	CORDIC_T0UN_ILOL_SRC	CORDIC_T0UN_ILOL input data source word address. Input float output float	0	w
31:16	CORDIC_T0UN_ILOL_DST	CORDIC_T0UN_ILOL output data destination word address. X is saved at here Y is saved at word address CORDIC_T0UN_ILOL_DST+1. Input float output float	0	w

19. USB 2.0 device controller

19.1 Introduction

The USB block is available on all QN908x devices.

19.2 Basic configuration

Initial configuration of the USB is accomplished as follows:

- Pins: PA26 and PA27 as USB_DP/USB_DM functions in PIO_FUNC_CFG3 register (see Section 2.5.29), for more information, please refer to pin mux table Table Table 95.
- Clock control: enable USB AHB clock by writing a 1 to (bit 10 of CLK_EN register, see Section 2.5.4); enable 48M PLL by configuring USBPLL_DIS to 0 (bit 4 of the PMU_CTRL1, see Section 2.5.39). 48M oscillator should be calibrated, please refer to the USB example project.
- Power control: enable the power of USB PHY by writing a 0 to USB_PHYSTDBY_WEN(bit 5 of the USB_CFG register, see Section 2.5.42), power on USB controller by writing a 0 to FSP_DIS (bit 18 of the PMU_CTRL0 register, see Section 2.5.38).
- Pull-up resistor connection control: configure DIS_USB_PULLUP to 0(bit 18 of the FUTURE_USE register, see Section-xxx); configure DPPUEN_B_PHY_POL to 1, configure DPPUEN_B_PHY_SEL to 1, configure USB_VBUS to 1 (bit 0,1,3 of the USB_CFG register, see Section 2.5.42). Let the connection controlled by USB controller register DCON (bit 16 of the DEVCMDSTAT register, see Section-xxx). See detail in Table-1.
- Pull-up resistor strength control: configure DPPU_OTP_POL to 1, configure DPPU_OTP_SEL to 1 (bit 24,25 of the FUTURE_USE register, see Section-xxx). Let USB controller controls the pull-up resistor strength. See detail in Table-2.
- Read CHIP_ID (Section 2.5.10) register to make sure USB_OPTION is set.
- Configure the USB wake-up signal (see Section 19.7.6) if needed.

Table 287. USB pull-up resister connection control

Input					Output
DIS_USB_PULLUP	DPPUEN_B_PHY_POL	DPPUEN_B_PHY_SEL	USB_VBUS	DCON	
0	0	0	x	x	connect
0	0	x	0	x	connect
0	0	x	x	0	connect
0	0	1	1	1	disconnect
0	1	0	x	x	disconnect
0	1	x	0	x	disconnect
0	1	x	x	0	disconnect
0	1	1	1	1	connect
1	x	x	x	x	disconnect
1	0	1	0	0	disconnect

DCON is a USB controller register, while the others are system registers.

The last line is the reset value.

Table 288. USB pull-up resistor strength control

Input			Output
DPPU_OPT_SETL	DPPU_OPT_POL	USB_STATUS	
0	0	x	1.2K ohm
0	1	x	2.4K ohm
1	0	0	1.2K ohm
1	0	1	2.4K ohm
1	1	0	2.4K ohm
1	1	1	1.2K ohm

USB_STATUS is not a register, it is a USB controller status.

The last line is the reset value.

19.3 Features

- USB2.0 full-speed device controller
- Supports 16 physical (8 logical) endpoints including two control endpoints if physical, one control endpoint if logical
- Single and double-buffering supported
- Each non-control endpoint supports bulk, interrupt, or isochronous endpoint types
- Supports remote wake-up
- Supports SoftConnect internally
- Link Power Management (LPM) supported

19.4 General description

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.

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. Each device can have a maximum of 16 logical or 32 physical endpoints. The device controller supports up to 10 physical 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.

For more information on the Universal Serial Bus, see the USB Implementers Forum website.

The USB device controller enables full-speed (12 Mb/s) data exchange with a USB host controller.

Figure 59 shows the block diagram of the USB device controller.

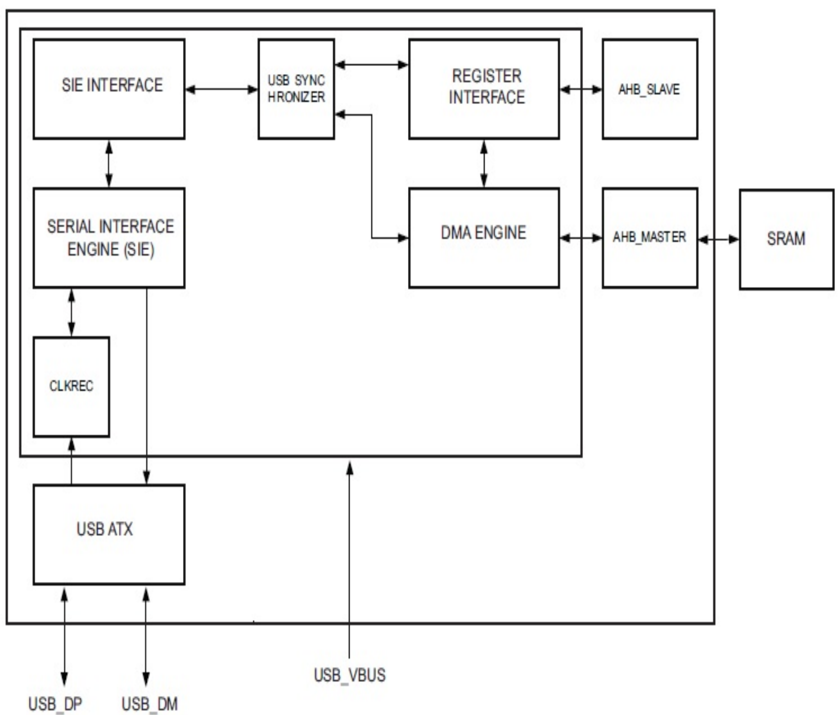


Fig 59. USB block diagram

The USB Device Controller has a built-in analog transceiver (ATX). The USB ATX sends/receives the bi-directional USB_DP and USB_DM signals of the USB bus.

The SIE implements the full USB protocol layer. It is completely hard-wired for speed and needs no software intervention. It handles transfer of data between the endpoint buffers in USB RAM and the USB bus. The functions of this block include: synchronization pattern recognition, parallel/serial conversion, bit stuffing/de-stuffing, CRC checking/generation, PID verification/generation, address recognition, and handshake evaluation/generation.

19.4.1 USB software interface

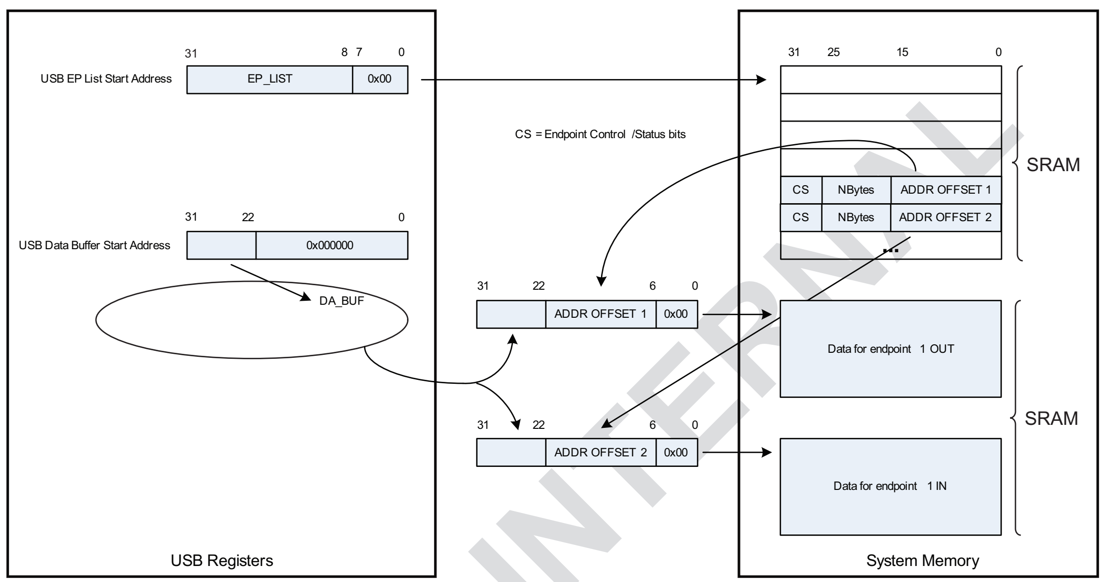


Fig 60. USB software interface

19.4.2 Fixed endpoint configuration

Table 289 shows the supported endpoint configurations. The packet size is configurable up to the maximum value shown in Table 289 for each type of end point.

Table 289. Fixed endpoint configuration

Logical endpoint	Physical endpoint	Endpoint type	Direction	Max packet size (byte)	Double buffer
0	0	Control	Out	64	No
0	1	Control	In	64	No
1	2	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
1	3	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes
2	4	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
2	5	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes
3	6	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
3	7	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes
4	8	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
4	9	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes
5	10	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
5	11	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes
6	12	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
6	13	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes
7	14	Interrupt/Bulk/Isochronous	Out	64/64/1023	Yes
7	15	Interrupt/Bulk/Isochronous	In	64/64/1023	Yes

19.4.3 SoftConnect

The softConnect signal is implemented internally. An external pull-up resistor between USB_DP and VDD is not necessary.

Software can control the USB_CONNECT signal by setting the DCON bit in the DEVCMDSTAT register. If the DCON bit is set to 1, the USB_DP line is pulled up to VDD through an internal 1.5 KΩ pull-up resistor.

The purpose of the soft connect feature using USB_CONNECT is to control when the device connects to the bus. When the device detects a USB_VBUS signal on the bus, it can finish processing if necessary, and then under software control indicate its presence to the host by pulling the USB_DP line HIGH. In a similar way, software can re-initialize a USB connection without the necessity to unplug the USB cable.

19.4.4 Interrupts

The USB controller has two interrupt lines, a general USB interrupt and a USB activity wake-up interrupt (USB_NEEDCLK). An general interrupt is generated by the hardware if both the interrupt status bit and the corresponding interrupt enable bit are set. The interrupt status bit is set by hardware if the interrupt condition occurs (irrespective of the interrupt enable bit setting).

19.4.5 Suspend and resume

The USB protocol insists on 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 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 a maximum of 500 μA.

A device will go into the L2 suspend state 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 controller also supports software initiated remote wake-up. To initiate remote wake-up, 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 USB controller supports Link Power Management (LPM). Link Power Management defines an additional link power management state L1 that supplements the existing L2 state by utilizing most of the existing suspend/resume infrastructure but provides much faster transitional latencies between L1 and L0 (On).

The assertion of USB suspend 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 USB NEEDCLK signal will go low. This indicates that the USB main clock can be switched off.

When activity is detected on the USB bus, the USB suspend signal is deactivated and USB NEEDCLK signal is activated. This process is fully combinatorial and hence no USB main clock is required to activate the USB NEEDCLK signal.

19.4.6 Clocking

The USB device controller has the following clock connections:

- USB main clock: The USB main clock is a 48 MHz clock used for USB functions.
- AHB clock: This is the AHB system bus clock. The minimum frequency of the AHB clock is 6 MHz when the USB device controller is receiving or transmitting USB packets.

19.5 Pin description

The device controller can access one USB port.

Table 290. USB device pin description

Name	Direction	Description
USB_DP	I/O	Positive differential data.
USB_DM	I/O	Negative differential data.

[1] Note that the VBUS is controlled by USB_CFG register in SYSCON chapter.

19.6 Register description

Table 291. Register overview: USB (base address 0x4008 4000)

Name	Access	Offset	Description	Reset value	Section
DEVCMDSTAT	R/W	0x000	USB Device Command/Status register	0x800	19.6.1
INFO	R/W	0x004	USB Info register	0x0	19.6.2
EPLISTSTART	R/W	0x008	USB EP Command/Status List start address	0x0	19.6.3
DATABUFSTART	R/W	0x00C	USB Data buffer start address	0x0	19.6.4
LPM	R/W	0x010	USB Link Power Management register	0x0	19.6.5
EPSKIP	R/W	0x014	USB Endpoint skip	0x0	19.6.6
EPINUSE	R/W	0x018	USB Endpoint Buffer in use	0x0	19.6.7
EPBUFCFG	R/W	0x01C	USB Endpoint Buffer Configuration register	0x0	19.6.8
INTSTAT	R/W	0x020	USB interrupt status register	0x0	19.6.9
INTEN	R/W	0x024	USB interrupt enable register	0x0	19.6.10
INTSETSTAT	R/W	0x028	USB set interrupt status register	0x0	19.6.11
EPTOGGLE	RO	0x034	USB Endpoint toggle register	0x0	19.6.12

19.6.1 USB device command/status register

Table 292. USB Device Command/Status register (DEVCMDSTAT, offset 0x00) bit description

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, software must program the new address before completing the status phase of the SetAddress Control Request.	0x0	R/W
7	DEV_EN	-	USB device enable. If this bit is set, the HW will start responding on packets for function address DEV_ADDR.	0x0	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 HW. SW must clear this bit by writing a one. If this bit is zero, HW will handle the tokens to the CTRL EP0 as indicated by the CTRL EP0 IN and OUT data information programmed by SW.	0x0	R/W1
9	FORCE_ NEEDCLK		Forces the NEEDCLK output to always be on:	0x0	R/W
		0	USB_NEEDCLK has normal function.		
		1	USB_NEEDCLK always 1. Clock will not be stopped in case of suspend.		
10	-	-	Reserved.	0x0	RO
11	LPM_SUP		LPM Supported:	0x1	R/W
		0	LPM not supported.		
		1	LPM supported.		
12	INTONNAK_AO		Interrupt on NAK for interrupt and bulk OUT EP	0x0	R/W
		0	Only acknowledged packets generate an interrupt		
		1	Both acknowledged and NAKed packets generate interrupts.		

Table 292. USB Device Command/Status register (DEVCMDSTAT, offset 0x00) bit description

Bit	Symbol	Value	Description	Reset value	Access
13	INTONNAK_AI		Interrupt on NAK for interrupt and bulk IN EP	0x0	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	0x0	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	0x0	R/W
		0	Only acknowledged packets generate an interrupt		
		1	Both acknowledged and NAKed packets generate interrupts.		
16	DCON	-	Device status - connect. The connect bit must be set by SW to indicate that the device must signal a connect. The pull-up resistor on USB_DP will be enabled when this bit is set and the VBUSDEBOUNCED bit is one.	0x0	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 DSUS: 1) and the software writes a 0 to it, the device will generate a remote wake-up. 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 never has an effect.	0x0	R/W
18	-	-	Reserved.	0x0	RO
19	LPM_SUS	-	Device status - LPM Suspend. This bit represents the current LPM suspend state. It is set to 1 by HW 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 software writes a zero to this bit, the device will generate a remote wake-up. Software can only write a zero to this bit when the LPM_REWP bit is set to 1. HW resets this bit when it receives a host initiated resume. HW only updates the LPM_SUS bit when the LPM_SUPP bit is equal to one.	0x0	R/W
20	LPM_REWP	-	LPM Remote Wake-up Enabled by USB host. HW sets this bit to one when the bRemoteWake bit in the LPM extended token is set to 1. HW will reset this bit to 0 when it receives the host initiated LPM resume, when a remote wake-up is sent by the device or when a USB bus reset is received. Software can use this bit to check if the remote wake-up feature is enabled by the host for the LPM transaction.	0x0	RO
23:21	-	-	Reserved.	0x0	RO
24	DCON_C	-	Device status - connect change. The Connect Change bit is set when the device's pull-up resistor is disconnected because VBus disappeared. The bit is reset by writing a one to it.	0x0	R/W1

Table 292. USB Device Command/Status register (DEVCMDSTAT, offset 0x00) bit description

Bit	Symbol	Value	Description	Reset value	Access
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: - The device goes in the suspended state - The device is disconnected - The device receives resume signaling on its upstream port. The bit is reset by writing a one to it.	0x0	R/W1
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 writing a one to it.	0x0	R/W1
27	-	-	Reserved.	0x0	RO
28	VBUS DEBOUNCED	-	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 HW will enable the pull-up resistor to signal a connect.	0x0	RO
31:29	-	-	Reserved.	0x0	RO

19.6.2 USB info register

Table 293. USB info register (INFO, offset 0x04) bit description

Bit	Symbol	Value	Description	Reset value	Access
10:0	FRAME_NR	-	Frame number. This 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.	0x0	RO

Table 293. USB info register (INFO, offset 0x04) bit description

Bit	Symbol	Value	Description	Reset value	Access
14:11	ERR_CODE		The error code which last occurred:	0x0	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/Received NAK		
		0xA	Sent Stall		
		0xB	Overrun		
		0xC	Sent empty packet		
		0xD	Bitstuff error		
		0xE	Sync error		
		0xF	Wrong data toggle		
15	-	-	Reserved.	-	-
31:16	-	-	Reserved	-	-

19.6.3 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 294. USB EP Command/Status List start address (EPLISTSTART, offset 0x08) bit description

Bit	Symbol	Description	Reset value	Access
7:0	-	Reserved	-	-
31:8	EP_LIST	Start address of the USB EP Command/Status List.	0x0	R/W

19.6.4 USB data buffer start address

This register indicates the page of the AHB address where the endpoint data can be located.

Table 295. USB Data buffer start address (DATABUFSTART, offset 0x0C) bit description

Bit	Symbol	Description	Reset value	Access
21:0	-	Reserved	-	-
31:22	DA_BUF	Start address of the buffer pointer page where all endpoint data buffers are located.	0x0	R/W

19.6.5 USB link power management register

Table 296. Link Power Management register (LPM, offset 0x10) bit description

Bit	Symbol	Description	Reset value	Access
3:0	HIRD_HW	Host Initiated Resume Duration - HW. This is the HIRD value from the last received LPM token	0x0	RO
7:4	HIRD_SW	Host Initiated Resume Duration - SW. 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.	0x0	R/W
8	DATA_PENDING	As long as this bit is set to one and LPM supported bit is set to one, HW will return a NYET handshake on every LPM token it receives. If LPM supported bit is set to one and this bit is zero, HW 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.	0x0	R/W
31:9	-	Reserved	-	-

19.6.6 USB endpoint skip

Table 297. USB Endpoint skip (EPSKIP, offset 0x14) bit description

Bit	Symbol	Description	Reset value	Access
29:0	SKIP	Endpoint skip: Writing 1 to one of these bits, will indicate to HW that it must deactivate the buffer assigned to this endpoint and return control back to software. When HW has deactivated the endpoint, it will clear this bit, but it will not modify the EPINUSE bit. An interrupt will be generated when the Active bit goes from 1 to 0. Note: In case of double-buffering, HW will only clear the Active bit of the buffer indicated by the EPINUSE bit.	0x0	R/W
31:30	-	Reserved	-	-

19.6.7 USB endpoint buffer in use

Table 298. USB Endpoint Buffer in use (EPINUSE, offset 0x18) bit description

Bit	Symbol	Description	Reset value	Access
1:0	-	Reserved. Fixed to zero because the control endpoint zero is fixed to single-buffering for each physical endpoint.	-	-
9:2	BUF	Buffer in use: This register has one bit per physical endpoint. 0: HW is accessing buffer 0. 1: HW is accessing buffer 1.	0x0	R/W
31:10	-	Reserved	-	-

19.6.8 USB endpoint buffer configuration

Table 299. USB Endpoint Buffer Configuration (EPBUFCFG, offset 0x1C) bit description

Bit	Symbol	Description	Reset value	Access
1:0	-	Reserved. Fixed to zero because the control endpoint zero is fixed to single-buffering for each physical endpoint.	-	-
9:2	BUF_SB	Buffer usage: This register has one bit per physical endpoint. 0: Single-buffer. 1: Double-buffer. If the bit is set to single-buffer (0), it will not toggle the corresponding EPINUSE bit when it clears the active bit. If the bit is set to double-buffer (1), HW will toggle the EPINUSE bit when it clears the Active bit for the buffer.	0x0	R/W
31:10	-	Reserved	-	-

19.6.9 USB interrupt status register

Table 300. USB interrupt status register (INTSTAT, offset 0x20) bit description

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 software 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. Software can clear this bit by writing a one to it.	0x0	R/W1
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 software. If the IntOnNAK_CI is set, this bit will also be set when a NAK is transmitted for the Control EP0 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
2	EP1OUT	Interrupt status register bit for the EP1 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP1 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1
3	EP1IN	Interrupt status register bit for the EP1 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP1 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
4	EP2OUT	Interrupt status register bit for the EP2 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP2 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1
5	EP2IN	Interrupt status register bit for the EP2 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP2 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1

Table 300. USB interrupt status register (INTSTAT, offset 0x20) bit description

Bit	Symbol	Description	Reset value	Access
6	EP3OUT	Interrupt status register bit for the EP3 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP3 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1
7	EP3IN	Interrupt status register bit for the EP3 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP3 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
8	EP4OUT	Interrupt status register bit for the EP4 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP4 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1
9	EP4IN	Interrupt status register bit for the EP4 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP4 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
10	EP5OUT	Interrupt status register bit for the EP5 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP5 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1
11	EP5IN	Interrupt status register bit for the EP5 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP5 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
12	EP6OUT	Interrupt status register bit for the EP6 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP6 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1

Table 300. USB interrupt status register (INTSTAT, offset 0x20) bit description

Bit	Symbol	Description	Reset value	Access
13	EP6IN	Interrupt status register bit for the EP6 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP6 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
14	EP7OUT	Interrupt status register bit for the EP7 OUT direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AO is set, this bit will also be set when a NAK is transmitted for the EP7 OUT direction. Software can clear this bit by writing a one to it.	0x0	R/W1
15	EP7IN	Interrupt status register bit for the EP7 IN direction. This bit will be set if the corresponding Active bit is cleared by HW. This is done in case the programmed NBytes transitions to zero or the skip bit is set by software. If the IntOnNAK_AI is set, this bit will also be set when a NAK is transmitted for the EP7 IN direction. Software can clear this bit by writing a one to it.	0x0	R/W1
29:16	-	Reserved	-	-
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 isochronous endpoints. Software can clear this bit by writing a one to it.	0x0	R/W1
31	DEV_INT	Device status interrupt. This bit is set by HW when one of the bits in the Device Status Change register are set. Software can clear this bit by writing a one to it.	0x0	R/W1

19.6.10 USB interrupt enable register

Table 301. USB interrupt enable register (INTEN, offset 0x24) bit description

Bit	Symbol	Description	Reset value	Access
15:0	EP_INT_EN	If this bit is set and the corresponding USB interrupt status bit is set, a HW interrupt is generated on the interrupt line indicated by the corresponding USB interrupt routing bit.	0x0	R/W
29:16	-	Reserved	-	-
30	FRAME_INT_EN	If this bit is set and the corresponding USB interrupt status bit is set, a HW interrupt is generated on the interrupt line indicated by the corresponding USB interrupt routing bit.	0x0	R/W
31	DEV_INT_EN	If this bit is set and the corresponding USB interrupt status bit is set, a HW interrupt is generated on the interrupt line indicated by the corresponding USB interrupt routing bit.	0x0	R/W

19.6.11 USB set interrupt status register

Table 302. USB set interrupt status register (INTSETSTAT, offset 0x28) bit description

Bit	Symbol	Description	Reset value	Access
15:0	EP_SET_INT	If software 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.	0x0	R/W
29:16	-	Reserved	-	-
30	FRAME_SET_INT	If software 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.	0x0	R/W
31	DEV_SET_INT	If software 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.	0x0	R/W

19.6.12 USB endpoint toggle

Table 303. USB Endpoint toggle (EPTOGGLE, offset 0x34) bit description

Bit	Symbol	Description	Reset value	Access
15:0	TOGGLE	Endpoint data toggle: This field indicates the current value of the data toggle for the corresponding endpoint.	0x0	RO
31:10	-	Reserved	-	-

19.7 Functional description

19.7.1 Endpoint command/status list

Figure 61 gives an overview on how the Endpoint List is organized in memory. The USB EP Command/Status List start register points to the start of the list that contains all the endpoint information in memory. The order of the endpoints is fixed as shown in the picture.

USB EP Command/Status FIFO start

31	30	29	28	27	26	25–16	15–0	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	EP7 OUT Buffer 0 NBytes	EP7 OUT Buffer 0 Address Offset	0x70
A	D	S	TR	RF TV	T	EP7 OUT Buffer 1 NBytes	EP7 OUT Buffer 1 Address Offset	0x74
A	D	S	TR	RF TV	T	EP7 IN Buffer 0 NBytes	EP7 IN Buffer 0 Address Offset	0x78
A	D	S	TR	RF TV	T	EP7 IN Buffer 1 NBytes	EP7 IN Buffer 1 Address Offset	0x7C

Fig 61. Endpoint command/status list (see also Table 304)(Lines should be added in the figure)

Table 304. Endpoint commands

Symbol	Access	Description
A	R/W	Active The buffer is enabled. HW can use the buffer to store received OUT data or to transmit data on the IN endpoint. Software can only set this bit to '1'. As long as this bit is set to one, software is not allowed to update any of the values in this 32-bit word. In case software 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 software has written a one to the "skip" bit.
D	R/W	Disabled 0: The selected endpoint is enabled. 1: The selected endpoint is disabled. If a USB token is received for an endpoint that has the disabled bit set, hardware will ignore the token and not return any data or handshake. When a bus reset is received, software must set the disable bit of all endpoints to 1. Software can only modify this bit when the active bit is zero.
S	R/W	Stall 0: The selected endpoint is not stalled. 1: The selected endpoint is stalled. 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 and the stall bit is one. Software can only modify this bit when the active bit is zero.

Table 304. Endpoint commands

Symbol	Access	Description
TR	R/W	Toggle Reset When software sets this bit to one, the HW will set the toggle value equal to the value indicated in the "toggle value" (TV) bit. 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. 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 For bulk endpoints and isochronous endpoints this bit is reserved and must be set to zero. 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. When the endpoint is used as an interrupt endpoint, it can be set to the following values. 0: Interrupt endpoint in 'toggle mode'. 1: Interrupt endpoint in 'rate feedback mode'. This means that the data toggle is fixed to zero for all data packets. When the interrupt endpoint is in 'rate feedback mode', the TR bit must always be set to zero.
T	R/W	Endpoint Type 0: Generic endpoint. The endpoint is configured as a bulk or interrupt endpoint. 1: Isochronous endpoint
NBytes	R/W	For OUT endpoints this is the number of bytes that can be received in this buffer. For IN endpoints this is the number of bytes that must be transmitted. HW decrements this value with the packet size every time when a packet is successfully transferred. 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. Software calculates the received number of bytes by subtracting the remaining NBytes from the programmed value.
Address Offset	R/W	Bits 21 to 6 of the buffer start address. The address offset is updated by hardware after each successful reception/transmission of a packet. Hardware increments the original value with the integer value when the packet size is divided by 64. Examples: • If an isochronous packet of 200 bytes is successfully received, the address offset is incremented by 3. • If a packet of 64 bytes is successfully received, the address offset is incremented by 1. • If a packet of less than 64 bytes is received, the address offset is not incremented.

Remark: When receiving a SETUP token for endpoint zero, the HW will only read the SETUP bytes Buffer Address offset to know where it has to store the received SETUP bytes. The 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.

19.7.2 Control endpoint 0

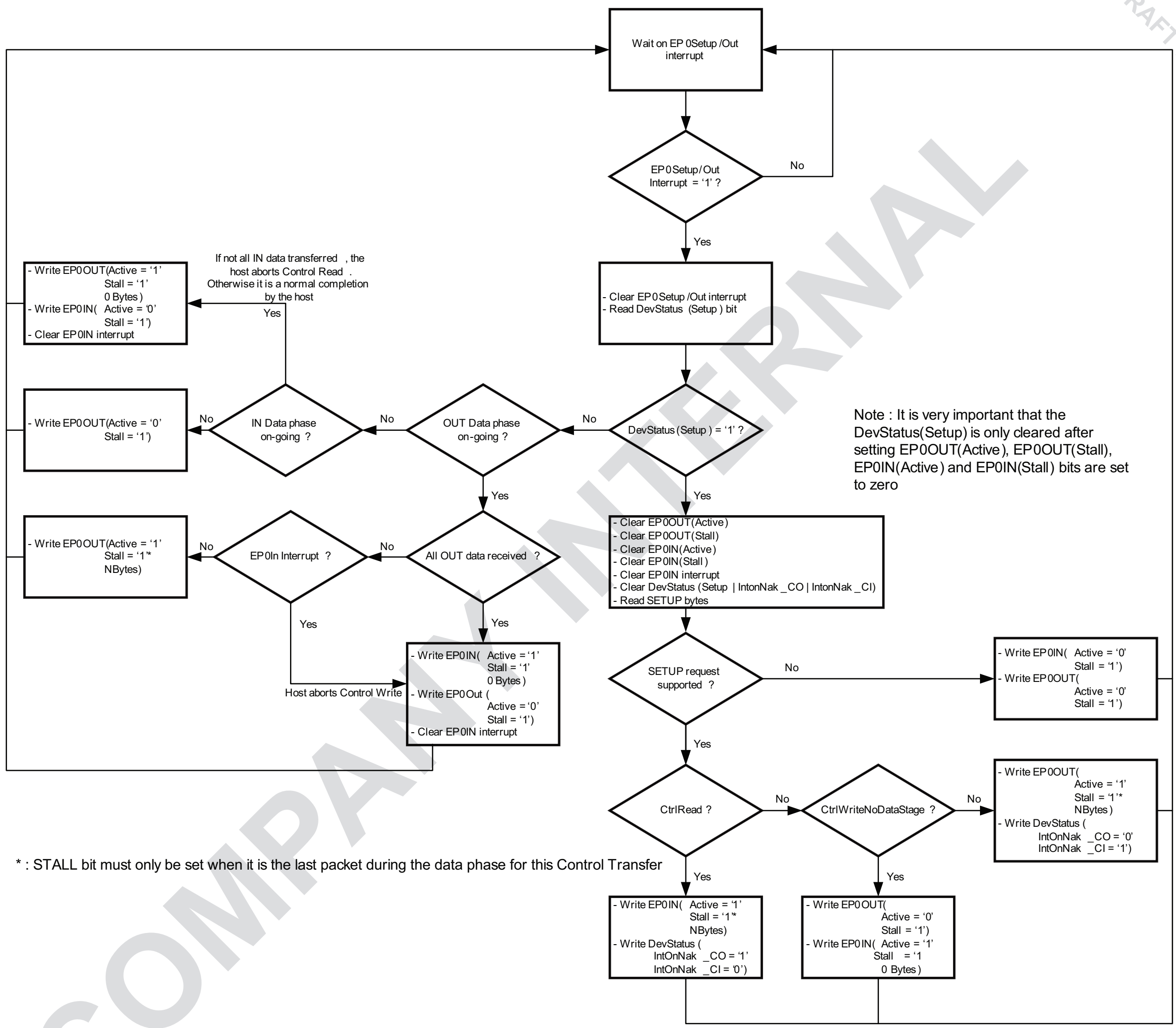
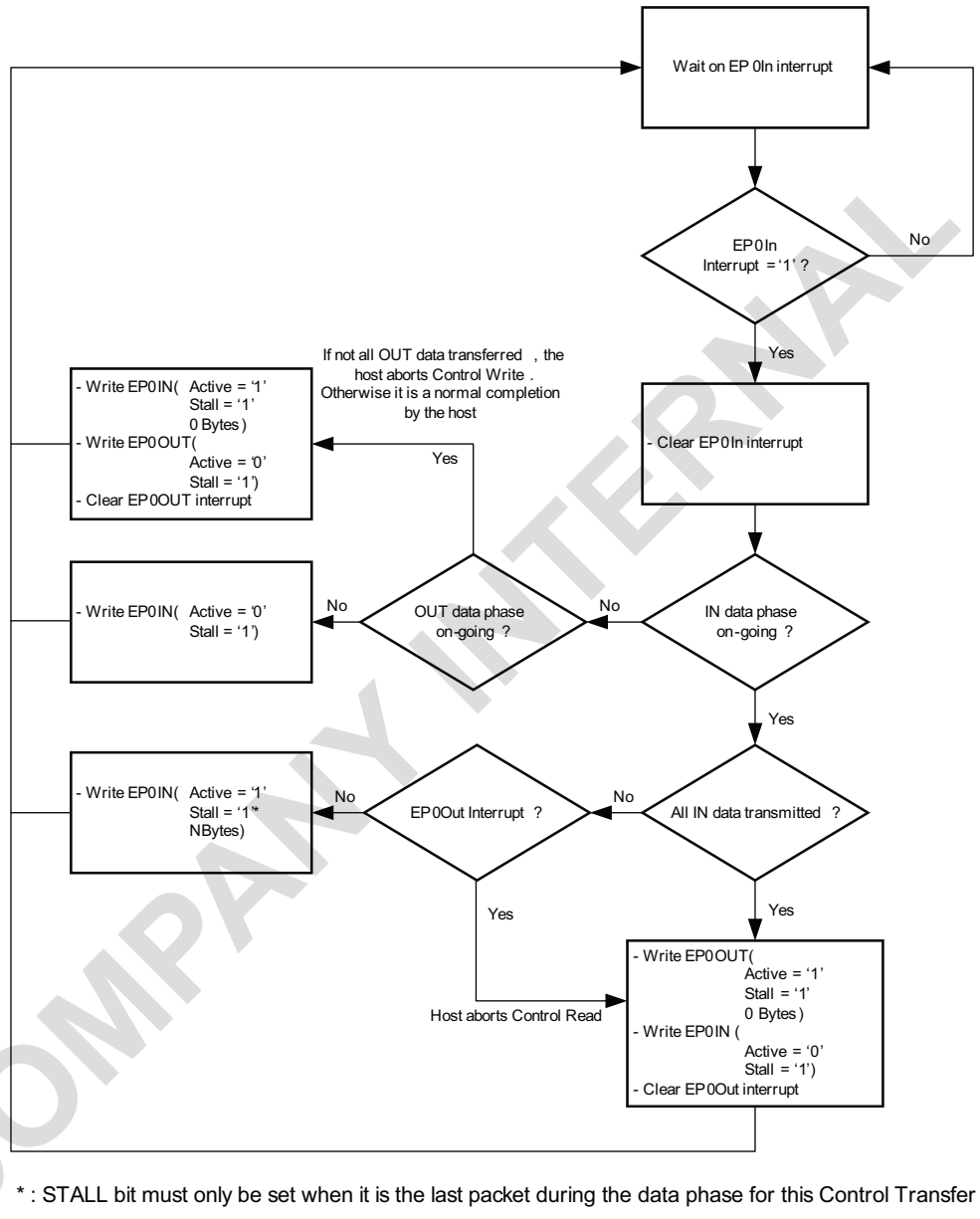


Fig 62. Flowchart of control endpoint 0 - OUT direction



* : STALL bit must only be set when it is the last packet during the data phase for this Control Transfer

Fig 63. Flowchart of control endpoint 0 - IN direction

19.7.3 Generic endpoint: single-buffering

To enable single-buffering, software must set the corresponding "USB EP Buffer Config" bit to zero. In the "USB EP Buffer in use" register, software can indicate which buffer is used in this case.

When software wants to transfer data, it programs the different bits in the Endpoint command/status entry and sets the active bits. The 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.

Software must wait until hardware has cleared the Active bit to change some of the command/status bits. This prevents hardware from overwriting a new value programmed by software with some old values that were still cached.

If software wants to disable the active bit before the hardware has finished handling the complete buffer, it can do this by setting the corresponding endpoint skip bit in USB endpoint skip register.

19.7.4 Generic endpoint: double-buffering

To enable double-buffering, software 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 HW when the next token is received.

When HW clears the active bit of the current buffer in use, it will switch the buffer in use. Software can also force HW to use a certain buffer by writing to the "USB EP Buffer in use" bit.

19.7.5 Special cases

19.7.5.1 Use of the Active bit

The use of the Active bit is a bit different between OUT and IN endpoints.

When data must be received for the OUT endpoint, the software 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 software sets the Active bit to one and programs the NBytes field to the number of bytes that must be transmitted.

19.7.5.2 Generation of a STALL handshake

Special care must be taken when 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 0. (No packet needs to be received/transmitted for that endpoint).
- The stall bit of the endpoint is set to one.

19.7.5.3 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 un-stalled and the toggle bit for that endpoint must be reset back to zero. In order to do that the software 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 0.
- Set toggle reset bit (TR) to 1 and set toggle value bit (TV) to 0.

If the endpoint is used in double-buffer mode, program the following:

- Set the STALL bit of both 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.

19.7.5.4 Set configuration

When a Set configuration request is received with a configuration value different from zero, the device software 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 Section 19.7.5.3 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.

19.7.6 USB wake-up

19.7.6.1 Remote wake-up

To issue a remote wake-up when the USB activity is suspended, complete the following steps:

- Set bit FORCE_NEEDCLK in the DEVCMDSTAT register to 0 (Section 19.6.1, default) to enable automatic control of the USB NEEDCLK signal.
- When it is time to issue a remote wake-up, turn on the USB clock and enable the USB clock source.
- Power up the USB PHY by writing a 1 to bit USB_PHYSTDBY_WEN and a 0 to bit USB_PHYSTDBY in the USB_CFG system register.
- Force the USB clock on by writing a 1 to bit FORCE_NEEDCLK (Section 19.6.1) in the DEVCMDSTAT register.
- Write a 0 to the DSUS bit in the DSVCMD_STAT register.
- Wait until the USB leaves the suspend state by polling the DSUS bit in the DSVCMD_STAT register (DSUS:0).
- Clear the FORCE_NEEDCLK bit (Section 19.6.1, bit 9) in the DEVCMDSTAT to enable automatic USB clock control.
- Clear bit USB_PHYSTDBY_WEN in the USB_CFG system register to enable automatic USB PHY control.

20. Flexcomm interface serial communication

20.1 Introduction

Multiple Flexcomm Interfaces are available on all QN908x parts.

Each Flexcomm Interface provides one peripheral function from a choice of several, chosen by the user. This chapter describes the overall Flexcomm Interface and how to choose peripheral functions. Details of the different peripherals are found in separate chapters for each type.

20.2 Features

Generally, each Flexcomm Interface provides a choice of peripheral functions, one of which must be chosen by the user before the function can be configured and used.

- USART with asynchronous operation or synchronous master or slave operation.
- SPI master or slave, with up to 4 slave selects.
- I^2C, including separate master, slave, and monitor functions.
- Data for USART and SPI traffic uses the Flexcomm Interface FIFO. The I^2C function does not use the FIFO.

Flexcomm Interfaces and functions included in QN908x devices were as shown in Table 305. Flexcomm0 and Flexcomm3 are fixed for USART and SPI interface. Specific part numbers and package variations may include a subset of this list.

Table 305: Flexcomm Interface base addresses and functions

Flexcomm number	Base address	USART (Section 21)	SPI (Section 22)	I^2C (Section 23)
0	0x4008 3000	Yes	No	No
1	0x4008 6000	Yes	No	Yes
2	0x4008 7000	No	Yes	Yes
3	0x4008 F000	No	Yes	No

20.3 Basic configuration

The Flexcomm Interface is configured as follows:

- If needed, configure RST_SW_SET and RST_SW_CLR registers to reset the related Flexcomm..
- Configure CLK_EN register to enable clock for the related Flexcomm.
- Select the desired Flexcomm function by writing to the PSELID register of the related Flexcomm (Section 20.6.2).
- See specific peripheral chapters for information on configuring those peripherals: UART (Section 21), SPI (Section 22), I2C (Section 23).

20.4 Architecture

The overall structure of one Flexcomm Interface is shown in Figure 64.

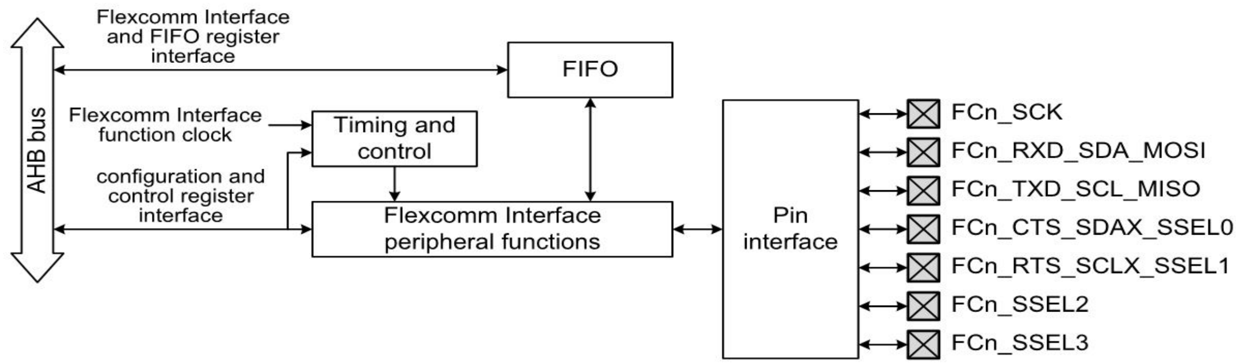


Fig 64. Flexcomm Interface block diagram

20.4.1 Choosing a peripheral function

A specific peripheral function, from among those supported by a particular Flexcomm Interface, is selected by software writing to the PSELID register. Reading the PSELID register provides information on which peripheral functions are available on that Flexcomm Interface.

Once a specific peripheral function has been selected, the PID register will supply an identifier for the selected peripheral. Software may use this information to confirm the selection before proceeding.

20.4.2 FIFO usage

Refer to the chapter for a specific peripheral function for information on how the FIFO is used (see Table 305).

20.4.3 DMA

The Flexcomm Interface generates DMA requests if desired, based on a selectable FIFO level. Refer to the chapter for a specific peripheral function for information on how the FIFO is used (see Table 305).

20.4.4 AHB bus access

Generally, the bus interface to the registers contained in the Flexcomm Interface (including its serial peripheral functions) support only word writes. Byte and halfword writes should not be used. An exception is that the FIFOWR register, when the Flexcomm Interface is configured for use as an SPI interface, it also allows byte and halfword writes. This allows support for control information embedded in DMA buffers, for example. See Section 22.6.14 for more information.

20.5 Pin description

Each Flexcomm Interface allows up to 7 pin connections. Specific uses of a Flexcomm Interface typically do not use all of these, and some Flexcomm Interface instances may not provide a means to connect all functions to device pins. Pin usage for a specific peripheral function is described in the chapter for that peripheral.

Table 306: Flexcomm Interface Pin Description

Pin	Type	Description
SCK	I/O	Clock input or output for the USART function in synchronous modes.
	I/O	Clock input or output for the SPI function.
	I/O	Clock input or output for the I^2S function (if present).
RXD_SDA_MOSI	Input	Receive data input for the USART function.
	I/O	SDA (data) input/output for the I^2C function.
	I/O	Master data output/slave data input for the SPI function.
	I/O	Data input or output for the I^2S function (if present).
TXD_SCL_MISO	Output	Transmit data output for the USART function.
	I/O	SCL input/output for the I^2C function.
	I/O	Master data input/slave data output for the SPI function.
	I/O	WS (also known as LRCLK) input or output for the I^2S function (if present).
CTS_SDA_SSEL0	Input	Clear To Send input for the USART function.
	I/O	SDA (data) input/output for the I^2C function.
	I/O	Slave Select 0 input or output for the SPI function.
RTS_SCL_SSEL1	Output	Request To Send output for the USART function.
	I/O	SCL (clock) input/output for the I^2C function.
	I/O	Slave Select 1 input or output for the SPI function.
SSEL2	I/O	Slave Select 2 input or output for the SPI function.
SSEL3	I/O	Slave Select 3 input or output for the SPI function.

20.6 Register description

Each Flexcomm Interface contains registers that are related to configuring the Flexcomm Interface to do a specific peripheral function and other registers related to peripheral FIFOs and data access. The latter depend somewhat on the chosen peripheral functions and are described in the chapters for each specific function (USART, SPI and I2C if present in a specific Flexcomm Interface). The Flexcomm Interface registers that identify and configure the Flexcomm Interface are shown in Table 307.

The base addresses of all Flexcomm Interfaces may be found in Table 305.

Table 307: Register map for the first channel pair within one Flexcomm Interface

Name	Access	Offset	Description	Reset Value [1]	Section
IOMODE	R/W	0xF00	IO mode register	0x0	20.6.1
PSELID	R/W	0xFF8	Peripheral Select and Flexcomm Interface ID register.	0x0	20.6.2
PID	RO	0xFFC	Peripheral identification register.	0x00101000	20.6.3

[1] Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

20.6.1 IO Mode register

The IOMODE register indicates share pin or not and shared pin output enabled or not while PERSEL is configured as USART and SPI. Note: if PERSEL is configured as I2C, the IOMODE register should be configured as default value.

Table 308. IO mode register (IOMODE - offset 0xF00) bit description

Bit	Symbol	Description	Reset Value
0	DIO_MODE	IO mode register, SPI share MISO/MOSI at MOSI, USART share TXD/RXD at RXD. If PERSEL=USART, it indicate whether 7816 mode is used: 0: Normal USART 1: 7816 mode, TXD/RXD will share on pin FCn_RXD_SDA_MOSI; If PERSEL=SPI, it indicates whether 3-wire mode is used: 0: Normal SPI 1: 3-wire SPI, MOSI/MISO will share on pin FCn_RXD_SDA_MOSI	0x0
1	DIO_OEN	This bit indicates IO output enable. If PERSEL=USART: 0 - FCn_RXD_SDA_MOSI is an input pin to receive data from another 7816 device. 1 - FCn_RXD_SDA_MOSI is an output pin to transmit data to another 7816 device. If PERSEL=SPI: 0 - FCn_RXD_SDA_MOSI is an input pin to receive data from another SPI device. 1 - FCn_RXD_SDA_MOSI is an output pin to transmit data to another SPI device.	0x0
31:2	-	Reserved. Read value is undefined, only zero should be written.	-

20.6.2 Peripheral Select and Flexcomm Interface ID register

The PSELID register identifies the Flexcomm Interface and provides information about which peripheral functions are supported by each Flexcomm Interface. It also provides the means to select one peripheral function for each Flexcomm Interface.

Table 309. Peripheral Select and Flexcomm Interface ID register (PSELID - offset 0xFF8) bit description

Bit	Symbol	Value	Description	Reset Value
2:0	PERSEL		Peripheral Select. This field is writable by software.	0x0
		0x0	No peripheral selected.	
		0x1	USART function selected.	
		0x2	SPI function selected.	
		0x3	I^2C function selected.	
		0x4	Reserved	
		0x5	Reserved	
		0x6	Reserved	
		0x7	Reserved	
3	LOCK		Lock the peripheral select. This field is writable by software.	0x0
		0	Peripheral select can be changed by software.	
		1	Peripheral select is locked and cannot be changed until this Flexcomm Interface or the entire device is reset.	
4	USARTPRESENT		USART present indicator. This field is Read-only.	0x0
		0	This Flexcomm Interface does not include the USART function.	
		1	This Flexcomm Interface includes the USART function.	
5	SPIPRESENT		SPI present indicator. This field is Read-only.	0x0
		0	This Flexcomm Interface does not include the SPI function.	
		1	This Flexcomm Interface includes the SPI function.	
6	I2CPRESENT		I^2C present indicator. This field is Read-only.	0x0
		0	This Flexcomm Interface does not include the I^2C function.	
		1	This Flexcomm Interface includes the I^2C function.	
7	-	-	Reserved.	NA
8	SC3W	-	Smart card/SPI 3 wire mode feature indicator. This field is Read-only; 0:This Flexcomm does not support smart card/SPI 3 wire mode feature; 1:This Flexcomm support smart card/SPI 3 wire mode feature;	0x0
11:9	-	-	Reserved. Read value is undefined, only zero should be written.	NA
31:12	ID		Flexcomm Interface ID.	0x00101

20.6.3 Peripheral identification register

This register is read-only and will read as 0 until a specific Flexcomm Interface function is selected via the PID register. Once the Flexcomm Interface is configured for a function, this register confirms the selection by returning the module ID for that function, and identifies the revision of that function. A software driver could make use of this information register to implement module type or revision specific behavior.

Table 310. Peripheral identification register (PID - offset 0xFFC) bit description

Bit	Symbol	Description	Reset Value
7:0	-	-	0x0
11:8	Minor_Rev	Minor revision of module implementation.	See specific device chapter
15:12	Major_Rev	Major revision of module implementation.	See specific device chapter
31:16	ID	Module identifier for the selected function.	See specific device chapter

21. USART

21.1 Introduction

USART functions are available on all QN908x devices as a selectable function in Flexcomm 1 and as a fixed function in Flexcomm 0.

21.2 Features

- 7, 8, or 9 data bits and 1 or 2 stop bits
- Synchronous mode with master or slave operation. Includes data phase selection and continuous clock option
- Multiprocessor/multidrop (9-bit) mode with software address compare
- RS-485 transceiver output enable
- Parity generation and checking: odd, even, or none
- Software selectable oversampling from 5 to 16 clocks in asynchronous mode
- One transmit and one receive data buffer
- The USART function supports separate transmit and receive FIFO with 16 entries each
- RTS/CTS for hardware signaling for automatic flow control. Software flow control can be performed using Delta CTS detect, Transmit Disable control, and any GPIO as an RTS output
- Break generation and detection
- Receive data is 2 of 3 sample “voting”. Status flag set when one sample differs
- Built-in baud rate generator
- Auto-baud mode for automatic baud rate detection
- 2 fractional rate dividers for Flexcomm0 and Flexcomm1
- Interrupts available for FIFO receive level reached, FIFO transmit level reached, FIFO overflow or underflow, Transmitter Idle, change in receiver break detect, Framing error, Parity error, Delta CTS detect, and receiver sample noise detected (among others)
- USART transmit and receive functions can operated with the system DMA controller
- Loopback mode for testing of data and flow control
- The USART function supports 7816 protocol by sharing TXD and RXD pin
- Special operating mode allows operation at up to 9600 baud using the 32 kHz clock as the USART clock. This mode can be used while the device is in sleep mode and can wake-up the device when a character is received

21.3 Basic configuration

Initial configuration of a USART peripheral is accomplished as follows:

- If needed, configure RST_SW_SET and RST_SW_CLR registers to reset the Flexcomm that is about to have a specific peripheral function selected.
- Configure the related Flexcomm pin functions via IOCON, see Section 8.

- Configure USART for receiving and transmitting data:
  - Configure CLK_EN register to enable clock for the related Flexcomm.
  - Select the desired Flexcomm function by writing to the PSELID register of the related Flexcomm (Section 20.6.2).
  - Enable or disable the related Flexcomm interrupt in the NVIC (Table 49).
  - Configure the FIFOs for operation.
  - Configure the Flexcomm clock and USART baudrate, See Section 21.3.1.
- Configure the USART to wake up the part from sleep mode. See Section 21.3.2.

21.3.1 Configure the Flexcomm Interface clock and USART baud rate

Both Flexcomm0 and Flexcomm1 have a separate fractional divider. The function clock and the fractional divider for the baud rate can be configured as follows:

- If a fractional value is needed to obtain a particular baud rate, program the fractional rate divider (FRG, controlled by Syscon register FC_FRG). The fractional divider value is the fraction of MULT/DIV. The MULT and DIV values are programmed in the FC_FRG register. The DIV value must be programmed with the fixed value of 256.

  Flexcomm Interface clock: (FRG input clock) / (1+(MULT / DIV))

  The following rules apply for MULT and DIV:
  - Always set DIV to 256 by programming the FRGCTRL register with the value of 0xFF.
  - Set the MULT to any value between 0 and 255.

  See Section 2.5.14 for more information on the FRG.
- In asynchronous mode: configure the baud rate divider BRGVAL in the BRG register. The baud rate divider divides the Flexcomm Interface function clock (FCLK) to create the clock needed to produce the desired baud rate.

  Generally: baud rate: [FCLK / oversample rate] / BRG divide

  With specific register values: baud rate: [FCLK / (OSRVAL+1)] / (BRGVAL + 1)

  Generally: BRG divide: [FCLK / oversample rate] / baud rate

  With specific register values: BRGVAL: [[FCLK / (OSRVAL + 1)] / baud rate] - 1

  See Section 21.6.6.
- In synchronous master mode: The serial clock is Un_SCLK: FCLK / (BRGVAL+1).

The USART can also be clocked by the 32 kHz clock. Set the MODE32K bit to enable this 32 kHz mode. See also Section 21.7.2.3.

For details on the clock configuration see: Section 21.7.2

21.3.2 Configure the USART for wake-up

A USART can wake up the system from sleep mode in asynchronous or synchronous mode on any enabled USART interrupt.

Steps to wake-up from sleep mode:

- Configure the USART in either asynchronous mode or synchronous mode. See Table 314.

- Enable the USART interrupt in the NVIC.
- Any enabled USART interrupt wakes up the part from sleep mode.

21.4 Pin description

The USART receive, transmit, and control signals are movable Flexcomm Interface functions and are assigned to external pins through via IOCON. See the IOCON description (Section 8) to assign the USART functions to pins on the device package.

Table 311. USART pin description

Pin	Type	Name used in Pin Configuration chapter	Description
TXD	O	FCn_TXD_SCL_MISO	Transmitter output for USART on Flexcomm Interface n. Serial transmit data.
RXD	I	FCn_RXD_SDA_MOSI	Receiver input for USART on Flexcomm Interface n. Serial receive data.
$\overline{RTS}$	O	FCn_RTS_SCL_SSEL1	Request To Send output for USART on Flexcomm Interface n. This signal supports inter-processor communication through the use of hardware flow control. This signal can also be configured to act as an output enable for an external RS-485 transceiver. RTS is active when the USART RTS signal is configured to appear on a device pin.
$\overline{CTS}$	I	FCn_CTS_SDA_SSEL0	Clear To Send input for USART on Flexcomm Interface n. Active low signal indicates that the external device that is in communication with the USART is ready to accept data. This feature is active when enabled by the CTSEn bit in CFG register and when configured to appear on a device pin. When deasserted (high) by the external device, the USART will complete transmitting any character already in progress, then stop until CTS is again asserted (low).
SCLK	I/O	FCn_SCK	Serial clock input/output for USART on Flexcomm Interface n in synchronous mode. Clock input or output in synchronous mode. **Remark:** when the USART is configured as a master, such that SCK is an output, it must actually be connected to a pin in order for the USART to work properly.

Recommended IOCON settings are shown in Table 312. See Section 8 for definitions of pin types.

Table 312: USART pin configuration example

Pin	Type	Name used in pin configuration chapter	PAD name	PINMUX register and configure value
TXD	O	FC0_TXD_SCL_MISO	PA04	PIO_FUNC_CFG0[18:16]: 0x4 (FUNC4)
RXD	I	FC0_RXD_SDA_MOSI	PA05	PIO_FUNC_CFG0[[22:20]: 0x4 (FUNC4)
RTS	O	FC0_RTS_SCL_SSEL1	PA00	PIO_FUNC_CFG0[[2:0]: 0x4 (FUNC4)
CTS	I	FC0_RTS_SCL_SSEL0	PA01	PIO_FUNC_CFG0[[6:4]: 0x4 (FUNC4)
SCLK	I/O	FC0_CLK	PA18	PIO_FUNC_CFG0[[10:8]: 0x4 (FUNC4)

21.5 General description

The USART receiver block monitors the serial input line, Un_RXD, for valid input. The receiver shift register assembles characters as they are received, after which they are passed to the receiver FIFO to await access by the CPU or DMA controller. When RTS is

configured as an RS-485 output enable, it is asserted at the beginning of an transmitted character, and deasserted either at the end of the character, or after a one character delay (selected by software).

The USART transmitter block accepts data written by the CPU or DMA controller to the transmit FIFO. When the transmitter is available, the transmit shift register takes that data, formats it, and serializes it to the serial output, Un_TXD.

The Baud Rate Generator block divides the incoming clock to create an oversample clock (typically 16x) in the standard asynchronous operating mode. The BRG clock input source is the shared Fractional Rate Generator that runs from the USART function clock. The 32 kHz operating mode generates a specially timed internal clock.

In synchronous slave mode, data is transmitted and received using the serial clock directly. In synchronous master mode, data is transmitted and received using the baud rate clock without division.

Status information from the transmitter and receiver is provided via the STAT register. Many of the status flags are able to generate interrupts, as selected by software. The INTSTAT register provides a view of all interrupts that are both enabled and pending.

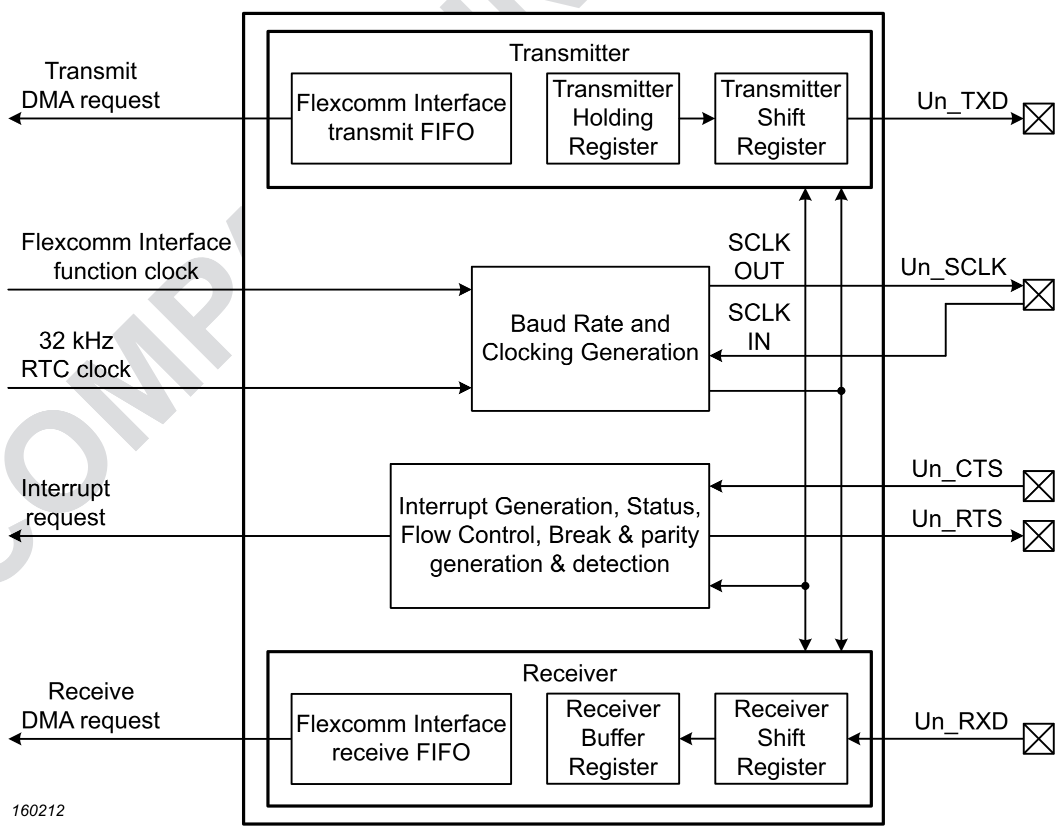


Fig 65. USART block diagram

21.6 Register description

The reset value reflects the data stored in used bits only. It does not include the content of reserved bits. Address offsets are within the related Flexcomm Interface address space **after** the USART function has been selected for that Flexcomm Interface (see Table 305 for a summary of Flexcomm Interface addresses).

In register description, the following notation is used: RO: Read-only, W1C: write 1 to clear.

Table 313: USART register overview

Name	Access	Offset	Description	Reset value	Section
Registers for the USART function:					
CFG	R/W	0x000	USART Configuration register. Basic USART configuration settings that typically are not changed during operation.	0x0	21.6.1
CTL	R/W	0x004	USART Control register. USART control settings that are more likely to change during operation.	0x0	21.6.2
STAT	R/W	0x008	USART Status register. The complete status value can be read here. Writing ones clears some bits in the register. Some bits can be cleared by writing a 1 to them.	0x0A	21.6.3
INTENSET	R/W	0x00C	Interrupt Enable read and Set register for USART (not FIFO) status. Contains individual interrupt enable bits for each potential USART interrupt. A complete value may be read from this register. Writing a 1 to any implemented bit position causes that bit to be set.	0x0	21.6.4
INTENCLR	WO	0x010	Interrupt Enable Clear register. Allows clearing any combination of bits in the INTENSET register. Writing a 1 to any implemented bit position causes the corresponding bit to be cleared.	-	21.6.5
BRG	R/W	0x020	Baud Rate Generator register. 16-bit integer baud rate divisor value.	0x0	21.6.6
INTSTAT	RO	0x024	Interrupt status register. Reflects interrupts that are currently enabled.	0x0	21.6.7
OSR	R/W	0x028	Oversample selection register for asynchronous communication.	0xF	21.6.8
ADDR	R/W	0x02C	Address register for automatic address matching.	0x0	21.6.9
Registers for FIFO control and data access:					
FIFOCFG	R/W	0xE00	FIFO configuration and enable register.	0x0	21.6.10
FIFOSTAT	R/W	0xE04	FIFO status register.	0x30	21.6.11
FIFOTRIG	R/W	0xE08	FIFO trigger level settings for interrupt and DMA request.	0x0	21.6.12
FIFOINTENSET	R/W1S	0xE10	FIFO interrupt enable set (enable) and read register.	0x0	21.6.13
FIFOINTENCLR	R/W1C	0xE14	FIFO interrupt enable clear (disable) and read register.	0x0	21.6.14
FIFOINTSTAT	RO	0xE18	FIFO interrupt status register.	0x0	21.6.15
FIFOWR	WO	0xE20	FIFO write data.	NA	21.6.16
FIFORD	RO	0xE30	FIFO read data.	NA	21.6.17
FIFORDNOPOP	RO	0xE40	FIFO data read with no FIFO pop.	NA	21.6.18
ID register:					
ID	RO	0xFFC	USART module Identification. This value appears in the shared Flexcomm Interface peripheral ID register when USART is selected.	0xE010 0000	21.6.19

21.6.1 USART Configuration register

The CFG register contains communication and mode settings for aspects of the USART that would normally be configured once in an application.

Remark: Only the CFG register can be written when the ENABLE bit: 0. CFG can be set up by software with ENABLE: 1, then the rest of the USART can be configured.

Remark: If software needs to change configuration values, the following sequence should be used: 1) Make sure the USART is not currently sending or receiving data. 2) Disable the USART by writing a 0 to the Enable bit (0 may be written to the entire register). 3) Write the new configuration value, with the ENABLE bit set to 1.

Table 314. USART Configuration register (CFG, offset 0x000) bit description

Bit	Symbol	Value	Description	Reset Value
0	ENABLE		USART Enable.	0x0
		0	Disabled. The USART is disabled and the internal state machine and counters are reset. While Enable: 0, all USART interrupts and DMA transfers are disabled. When Enable is set again, CFG and most other control bits remain unchanged. When re-enabled, the USART will immediately be ready to transmit because the transmitter has been reset and is therefore available.	
		1	Enabled. The USART is enabled for operation.	
1	-	-	Reserved. Read value is undefined, only zero should be written.	NA
3:2	DATALEN		Selects the data size for the USART.	0x0
		0x0	7 bit Data length.	
		0x1	8 bit Data length.	
		0x2	9 bit data length. The 9th bit is commonly used for addressing in multidrop mode. See the ADDRDET bit in the CTL register.	
		0x3	Reserved.	
5:4	PARITYSEL		Selects what type of parity is used by the USART.	0x0
		0x0	No parity.	
		0x1	Reserved.	
		0x2	Even parity. Adds a bit to each character such that the number of 1s in a transmitted character is even, and the number of 1s in a received character is expected to be even.	
		0x3	Odd parity. Adds a bit to each character such that the number of 1s in a transmitted character is odd, and the number of 1s in a received character is expected to be odd.	
6	STOPLEN		Number of stop bits appended to transmitted data. Only a single stop bit is required for received data.	0x0
		0	1 stop bit.	
		1	2 stop bits. This setting should only be used for asynchronous communication.	
7	MODE32K		Selects standard or 32 kHz clocking mode.	0x0
		0	Disabled. USART uses standard clocking.	
		1	Enabled. USART uses the 32 kHz clock as the clock source to the BRG, and uses a special bit clocking scheme.	
8	LINMODE		LIN break mode enable.	0x0
		0	Disabled. Break detect and generate is configured for normal operation.	
		1	Enabled. Break detect and generate is configured for LIN bus operation.	

Table 314. USART Configuration register (CFG, offset 0x000) bit description *…continued*

Bit	Symbol	Value	Description	Reset Value
9	CTSEN		CTS Enable. Determines whether CTS is used for flow control. CTS can be from the input pin, or from the USART's own RTS if loopback mode is enabled.	0x0
		0	No flow control. The transmitter does not receive any automatic flow control signal.	
		1	Flow control enabled. The transmitter uses the CTS input (or RTS output in loopback mode) for flow control purposes.	
10	-	-	Reserved. Read value is undefined, only zero should be written.	NA
11	SYNCEN		Selects synchronous or asynchronous operation.	0x0
		0	Asynchronous mode.	
		1	Synchronous mode.	
12	CLKPOL		Selects the clock polarity and sampling edge of received data in synchronous mode.	0x0
		0	Falling edge. Un_RXD is sampled on the falling edge of SCLK.	
		1	Rising edge. Un_RXD is sampled on the rising edge of SCLK.	
13	-	-	Reserved. Read value is undefined, only zero should be written.	NA
14	SYNCMST		Synchronous mode Master select.	0x0
		0	Slave. When synchronous mode is enabled, the USART is a slave.	
		1	Master. When synchronous mode is enabled, the USART is a master.	
15	LOOP		Selects data loopback mode.	0x0
		0	Normal operation.	
		1	Loopback mode. This provides a mechanism to perform diagnostic loopback testing for USART data. Serial data from the transmitter (Un_TXD) is connected internally to serial input of the receive (Un_RXD). Un_TXD and Un_RTS activity will also appear on external pins if these functions are configured to appear on device pins. The receiver RTS signal is also looped back to CTS and performs flow control if enabled by CTSEN.	
17:16	-	-	Reserved. Read value is undefined, only zero should be written.	NA
18	OETA		Output Enable Turnaround time enable for RS-485 operation.	0x0
		0	Disabled. If selected by OESEL, the Output Enable signal deasserted at the end of the last stop bit of a transmission.	
		1	Enabled. If selected by OESEL, the Output Enable signal remains asserted for one character time after the end of the last stop bit of a transmission. OE will also remain asserted if another transmit begins before it is deasserted.	
19	AUTOADDR		Automatic Address matching enable.	0x0
		0	Disabled. When addressing is enabled by ADDRDET, address matching is done by software. This provides the possibility of versatile addressing (e.g. respond to more than one address).	
		1	Enabled. When addressing is enabled by ADDRDET, address matching is done by hardware, using the value in the ADDR register as the address to match.	
20	OESEL		Output Enable Select.	0x0
		0	Standard. The RTS signal is used as the standard flow control function.	
		1	RS-485. The RTS signal configured to provide an output enable signal to control an RS-485 transceiver.	
21	OEPOL		Output Enable Polarity.	0x0
		0	Low. If selected by OESEL, the output enable is active low.	
		1	High. If selected by OESEL, the output enable is active high.	

Table 314. USART Configuration register (CFG, offset 0x000) bit description *…continued*

Bit	Symbol	Value	Description	Reset Value
22	RXPOL		Receive data polarity.	0x0
		0	Standard. The RX signal is used as it arrives from the pin. This means that the RX rest value is 1, start bit is 0, data is not inverted, and the stop bit is 1.	
		1	Inverted. The RX signal is inverted before being used by the USART. This means that the RX rest value is 0, start bit is 1, data is inverted, and the stop bit is 0.	
23	TXPOL		Transmit data polarity.	0x0
		0	Standard. The TX signal is sent out without change. This means that the TX rest value is 1, start bit is 0, data is not inverted, and the stop bit is 1.	
		1	Inverted. The TX signal is inverted by the USART before being sent out. This means that the TX rest value is 0, start bit is 1, data is inverted, and the stop bit is 0.	
31:24	-	-	Reserved. Read value is undefined, only zero should be written.	NA

21.6.2 USART Control register

The CTL register controls aspects of USART operation that are more likely to change during operation.

Table 315. USART Control register (CTL, offset 0x004) bit description

Bit	Symbol	Value	Description	Reset Value
0	-	-	Reserved. Read value is undefined, only zero should be written.	NA
1	TXBRKEN		Break Enable.	0x0
		0	Normal operation.	
		1	Continuous break. Continuous break is sent immediately when this bit is set, and remains until this bit is cleared. A break may be sent without danger of corrupting any currently transmitting character if the transmitter is first disabled (TXDIS in CTL is set) and then waiting for the transmitter to be disabled (TXDISINT in STAT: 1) before writing 1 to TXBRKEN.	
2	ADDRDET		Enable address detect mode.	0x0
		0	Disabled. The USART presents all incoming data.	
		1	Enabled. The USART receiver ignores incoming data that does not have the most significant bit of the data (typically the 9th bit): 1. When the data MSB bit: 1, the receiver treats the incoming data normally, generating a received data interrupt. Software can then check the data to see if this is an address that should be handled. If it is, the ADDRDET bit is cleared by software and further incoming data is handled normally.	
5:3	-	-	Reserved. Read value is undefined, only zero should be written.	NA
6	TXDIS		Transmit Disable.	0x0
		0	Not disabled. USART transmitter is not disabled.	
		1	Disabled. USART transmitter is disabled after any character currently being transmitted is complete. This feature can be used to facilitate software flow control.	
7	-	-	Reserved. Read value is undefined, only zero should be written.	NA

Table 315. USART Control register (CTL, offset 0x004) bit description *…continued*

Bit	Symbol	Value	Description	Reset Value
8	CC		Continuous Clock generation. By default, SCLK is only output while data is being transmitted in synchronous mode.	0x0
		0	Clock on character. In synchronous mode, SCLK cycles only when characters are being sent on Un_TXD or to complete a character that is being received.	
		1	Continuous clock. SCLK runs continuously in synchronous mode, allowing characters to be received on Un_RxD independently from transmission on Un_TXD).	
9	CLRCCONRX		Clear Continuous Clock.	0x0
		0	No effect. No effect on the CC bit.	
		1	Auto-clear. The CC bit is automatically cleared when a complete character has been received. This bit is cleared at the same time.	
15:10	-	-	Reserved. Read value is undefined, only zero should be written.	NA
16	AUTOBAUD		Auto-baud enable.	0x0
		0	Disabled. USART is in normal operating mode.	
		1	Enabled. USART is in auto-baud mode. This bit should only be set when the USART receiver is idle. The first start bit of RX is measured and used the update the BRG register to match the received data rate. AUTOBAUD is cleared once this process is complete, or if there is an AERR.	
31:17	-	-	Reserved. Read value is undefined, only zero should be written.	NA

21.6.3 USART status register

The STAT register primarily provides a set of USART status flags (not including FIFO status) for software to read. Flags other than read-only flags may be cleared by writing ones to corresponding bits of STAT. Interrupt status flags that are read-only and cannot be cleared by software, can be masked using the INTENCLR register (see Table 318).

The error flags for received noise, parity error, and framing error are set immediately upon detection and remain set until cleared by software action in STAT.

Table 316. USART status register (STAT, offset 0x008) bit description

Bit	Symbol	Description	Reset value	Access [1]
0	-	Reserved. Read value is undefined, only zero should be written.	NA	NA
1	RXIDLE	Receiver Idle. When 0, indicates that the receiver is currently in the process of receiving data. When 1, indicates that the receiver is not currently in the process of receiving data.	0x1	RO
2	-	Reserved. Read value is undefined, only zero should be written.	NA	NA
3	TXIDLE	Transmitter Idle. When 0, indicates that the transmitter is currently in the process of sending data.When 1, indicate that the transmitter is not currently in the process of sending data.	0x1	RO
4	CTS	This bit reflects the current state of the CTS signal, regardless of the setting of the CTSEN bit in the CFG register. This will be the value of the CTS input pin unless loopback mode is enabled.	NA	RO
5	DELTACTS	This bit is set when a change in the state is detected for the CTS flag above. This bit is cleared by software.	0x0	W1

Table 316. USART status register (STAT, offset 0x008) bit description *…continued*

Bit	Symbol	Description	Reset value	Access [1]
6	TXDISSTAT	Transmitter Disabled Status flag. When 1, this bit indicates that the USART transmitter is fully idle after being disabled via the TXDIS bit in the CFG register (TXDIS: 1).	0x0	RO
9:7	-	Reserved. Read value is undefined, only zero should be written.	NA	NA
10	RXBRK	Received Break. This bit reflects the current state of the receiver break detection logic. It is set when the Un_RXD pin remains low for 16 bit times. Note that FRAMERRINT will also be set when this condition occurs because the stop bit(s) for the character would be missing. RXBRK is cleared when the Un_RXD pin goes high.	0x0	RO
11	DELTARXBRK	This bit is set when a change in the state of receiver break detection occurs. Cleared by software.	0x0	R/W1C
12	START	This bit is set when a start is detected on the receiver input. Its purpose is primarily to allow wake-up from deep-sleep mode immediately when a start is detected. Cleared by software.	0x0	R/W1C
13	FRAMERRINT	Framing Error interrupt flag. This flag is set when a character is received with a missing stop bit at the expected location. This could be an indication of a baud rate or configuration mismatch with the transmitting source.	0x0	R/W1C
14	PARITYERRINT	Parity Error interrupt flag. This flag is set when a parity error is detected in a received character.	0x0	R/W1C
15	RXNOISEINT	Received Noise interrupt flag. Three samples of received data are taken in order to determine the value of each received data bit, except in synchronous mode. This acts as a noise filter if one sample disagrees. This flag is set when a received data bit contains one disagreeing sample. This could indicate line noise, a baud rate or character format mismatch, or loss of synchronization during data reception.	0x0	R/W1C
16	ABERR	Auto baud Error. An auto baud error can occur if the BRG counts to its limit before the end of the start bit that is being measured, essentially an auto baud time-out.	0x0	R/W1C
31:17	-	Reserved. Read value is undefined, only zero should be written.	NA	NA

[1] RO: Read-only, W1C: write 1 to clear.

21.6.4 USART interrupt enable read and set register

The INTENSET register is used to enable various USART interrupt sources (not including FIFO interrupts). Enable bits in INTENSET are mapped in locations that correspond to the flags in the STAT register. Interrupt enables may also be read back from this register. Writing ones to implemented bits in this register causes those bits to be set. The INTENCLR register is used to clear bits in this register.

Table 317. USART interrupt enable read and set register (INTENSET, offset 0x00C) bit description

Bit	Symbol	Description	Reset Value
2:0	-	Reserved. Read value is undefined, only zero should be written.	NA
3	TXIDLEEN	When 1, enables an interrupt when the transmitter becomes idle (TXIDLE: 1).	0x0
4	-	Reserved. Read value is undefined, only zero should be written.	NA
5	DELTACTSEN	When 1, enables an interrupt when there is a change in the state of the CTS input.	0x0
6	TXDISEN	When 1, enables an interrupt when the transmitter is fully disabled as indicated by the TXDISINT flag in STAT. See description of the TXDISINT bit for details.	0x0

Table 317. USART interrupt enable read and set register (INTENSET, offset 0x00C) bit description *…continued*

Bit	Symbol	Description	Reset Value
10:7	-	Reserved. Read value is undefined, only zero should be written.	-
11	DELTARXBRKEN	When 1, enables an interrupt when a change of state has occurred in the detection of a received break condition (break condition asserted or deasserted).	0x0
12	STARTEN	When 1, enables an interrupt when a received start bit has been detected.	0x0
13	FRAMERREN	When 1, enables an interrupt when a framing error has been detected.	0x0
14	PARITYERREN	When 1, enables an interrupt when a parity error has been detected.	0x0
15	RXNOISEEN	When 1, enables an interrupt when noise is detected. See description of the RXNOISEINT bit in Table 316.	0x0
16	ABERREN	When 1, enables an interrupt when an auto baud error occurs.	0x0
31:17	-	Reserved. Read value is undefined, only zero should be written.	-

21.6.5 USART interrupt enable clear register

The INTENCLR register is used to clear bits in the INTENSET register.

Table 318. USART interrupt enable clear register (INTENCLR, offset 0x010) bit description

Bit	Symbol	Description	Reset value
2:0	-	Reserved. Read value is undefined, only zero should be written.	-
3	TXIDLECLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
4	-	Reserved. Read value is undefined, only zero should be written.	-
5	DELTACTSCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
6	TXDISCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
10:7	-	Reserved. Read value is undefined, only zero should be written.	-
11	DELTARXBRKCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
12	STARTCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
13	FRAMERRCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
14	PARITYERRCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
15	RXNOISECLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
16	ABERRCLR	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
31:17	-	Reserved. Read value is undefined, only zero should be written.	-

21.6.6 USART baud rate generator register

The baud rate generator is a simple 16-bit integer divider controlled by the BRG register. The BRG register contains the value used to divide the Flexcomm Interface clock (FCLK) in order to produce the clock used for USART internal operations.

A 16-bit value allows producing standard baud rates from 300 baud and lower at the highest frequency of the device, up to 921,600 baud from a base clock as low as 14.7456 MHz.

Typically, the baud rate clock is 16 times the actual baud rate. This overclocking allows for centering the data sampling time within a bit cell, and for noise reduction and detection by taking three samples of incoming data.

Note that in 32 kHz mode, the baud rate generator is still used and must be set to 0 if 9600 baud is required.

For more information on USART clocking, see Section 21.7.2 and Section 21.3.1.

Remark: In order to change a baud rate after a USART is running, the following sequence should be used:

1. Make sure the USART is not currently sending or receiving data.
2. Disable the USART by writing a 0 to the Enable bit (0 may be written to the entire register).
3. Write the new BRGVAL.
4. Write to the CFG register to set the Enable bit to 1.

Table 319. USART baud rate generator register (BRG, offset 0x020) bit description

Bit	Symbol	Description	Reset value
15:0	BRGVAL	This value is used to divide the USART input clock to determine the baud rate, based on the input clock from the FRG. 0: FCLK is used directly by the USART function. 1: FCLK is divided by 2 before use by the USART function. 2: FCLK is divided by 3 before use by the USART function. ... 0xFFFF: FCLK is divided by 65,536 before use by the USART function.	0x0
31:16	-	Reserved. Read value is undefined, only zero should be written.	-

21.6.7 USART interrupt status register

The read-only INTSTAT register provides a view of those interrupt flags that are currently enabled. This can simplify software handling of interrupts. See Table 316 for detailed descriptions of the interrupt flags.

Table 320. USART interrupt status register (INTSTAT, offset 0x024) bit description

Bit	Symbol	Description	Reset value
2:0	-	Reserved. Read value is undefined, only zero should be written.	-
3	TXIDLE	Transmitter Idle status.	0x0
4	-	Reserved. Read value is undefined, only zero should be written.	-
5	DELTACTS	This bit is set when a change in the state of the CTS input is detected.	0x0
6	TXDISINT	Transmitter Disabled Interrupt flag.	0x0

Table 320. USART interrupt status register (INTSTAT, offset 0x024) bit description *…continued*

Bit	Symbol	Description	Reset value
10:7	-	Reserved. Read value is undefined, only zero should be written.	-
11	DELTARXBRK	This bit is set when a change in the state of receiver break detection occurs.	0x0
12	START	This bit is set when a start is detected on the receiver input.	0x0
13	FRAMERRINT	Framing Error interrupt flag.	0x0
14	PARITYERRINT	Parity Error interrupt flag.	0x0
15	RXNOISEINT	Received Noise interrupt flag.	0x0
16	ABERRINT	Auto baud Error Interrupt flag.	0x0
31:17	-	Reserved. Read value is undefined, only zero should be written.	-

21.6.8 Oversample selection register

The OSR register allows selection of oversampling in asynchronous modes. The oversample value is the number of BRG clocks used to receive one data bit. The default is industry standard 16x oversampling.

Changing the oversampling can sometimes allow better matching of baud rates in cases where the function clock rate is not a multiple of 16 times the expected maximum baud rate. For all modes where the OSR setting is used, the USART receiver takes three consecutive samples of input data in the approximate middle of the bit time. Smaller values of OSR can make the sampling position within a data bit less accurate and may potentially cause more noise errors or incorrect data.

Table 321. Oversample selection register (OSR, offset 0x028) bit description

Bit	Symbol	Description	Reset value
3:0	OSRVAL	Oversample Selection Value. 0 to 3: not supported 0x4: 5 function clocks are used to transmit and receive each data bit. 0x5: 6 function clocks are used to transmit and receive each data bit. ... 0xF= 16 function clocks are used to transmit and receive each data bit.	0xF
31:4	-	Reserved, the value read from a reserved bit is not defined.	-

21.6.9 Address register

The ADDR register holds the address for hardware address matching in address detect mode with automatic address matching enabled.

Table 322. Address register (ADDR, offset 0x02C) bit description

Bit	Symbol	Description	Reset value
7:0	ADDRESS	8-bit address used with automatic address matching. Used when address detection is enabled (ADDRDET in CTL: 1) and automatic address matching is enabled (AUTOADDR in CFG: 1).	0x0
31:8	-	Reserved, the value read from a reserved bit is not defined.	-

21.6.10 FIFO configuration register

This register configures FIFO usage. A peripheral function within the Flexcomm Interface must be selected prior to configuring the FIFO.

Table 323. FIFO configuration register (FIFOCFG - offset 0xE00) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ENABLETX		Enable the transmit FIFO.	0x0	R/W
		0	The transmit FIFO is not enabled.		
		1	The transmit FIFO is enabled.		
1	ENABLERX		Enable the receive FIFO.	0x0	R/W
		0	The receive FIFO is not enabled.		
		1	The receive FIFO is enabled.		
3:2	-	-	Reserved. Read value is undefined, only zero should be written.	-	-
5:4	SIZE		FIFO size configuration. This is a read-only field. 0x0: FIFO is configured as 16 entries of 8 bits. 0x1, 0x2, 0x3: not applicable to USART.	NA	RO
11:6	-	-	Reserved. Read value is undefined, only zero should be written.	-	-
12	DMATX		DMA configuration for transmit.	0x0	R/W
		0	DMA is not used for the transmit function.		
		1	Generate a DMA request for the transmit function if the FIFO is not full. Generally, data interrupts would be disabled if DMA is enabled.		
13	DMARX		DMA configuration for receive.	0x0	R/W
		0	DMA is not used for the receive function.		
		1	Generate a DMA request for the receive function if the FIFO is not empty. Generally, data interrupts would be disabled if DMA is enabled.		
15:14	-	-	Reserved.	NA	NA
16	EMPTYTX	-	Empty command for the transmit FIFO. When a 1 is written to this bit, the TX FIFO is emptied.	NA	WO
17	EMPTYRX	-	Empty command for the receive FIFO. When a 1 is written to this bit, the RX FIFO is emptied.	NA	WO
31:18	-	-	Reserved. Read value is undefined, only zero should be written.	-	-

21.6.11 FIFO status register

This register provides status information for the FIFO and also indicates an interrupt from the peripheral function.

Table 324. FIFO status register (FIFOSTAT - offset 0xE04) bit description

Bit	Symbol	Description	Reset value	Access
0	TXERR	TX FIFO error. Will be set if a transmit FIFO error occurs. This could be an overflow caused by pushing data into a full FIFO, or by an underflow if the FIFO is empty when data is needed. Cleared by writing a 1 to this bit.	0x0	R/W1C
1	RXERR	RX FIFO error. Will be set if a receive FIFO overflow occurs, caused by software or DMA not emptying the FIFO fast enough. Cleared by writing a 1 to this bit.	0x0	R/W1C
2	-	Reserved. Read value is undefined, only zero should be written.	-	-

Table 324. FIFO status register (FIFOSTAT - offset 0xE04) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
3	PERINT	Peripheral interrupt. When 1, this indicates that the peripheral function has asserted an interrupt. The details can be found by reading the peripheral's STAT register.	0x0	RO
4	TXEMPTY	Transmit FIFO empty. When 1, the transmit FIFO is empty. The peripheral may still be processing the last piece of data.	0x1	RO
5	TXNOTFULL	Transmit FIFO not full. When 1, the transmit FIFO is not full, so more data can be written. When 0, the transmit FIFO is full and another write would cause it to overflow.	0x1	RO
6	RXNOTEMPTY	Receive FIFO not empty. When 1, the receive FIFO is not empty, so data can be read. When 0, the receive FIFO is empty.	0x0	RO
7	RXFULL	Receive FIFO full. When 1, the receive FIFO is full. Data needs to be read out to prevent the peripheral from causing an overflow.	0x0	RO
12:8	TXLVL	Transmit FIFO current level. A 0 means the TX FIFO is currently empty, and the TXEMPTY and TXNOTFULL flags will be 1. Other values tell how much data is actually in the TX FIFO at the point where the read occurs. If the TX FIFO is full, the TXEMPTY and TXNOTFULL flags will be 0.	0x0	RO
15:13	-	Reserved. Read value is undefined, only zero should be written.	-	-
20:16	RXLVL	Receive FIFO current level. A 0 means the RX FIFO is currently empty, and the RXFULL and RXNOTEMPTY flags will be 0. Other values tell how much data is actually in the RX FIFO at the point where the read occurs. If the RX FIFO is full, the RXFULL and RXNOTEMPTY flags will be 1.	0x0	RO
31:21	-	Reserved. Read value is undefined, only zero should be written.	-	-

21.6.12 FIFO trigger level settings register

This register allows selecting when FIFO-level related interrupts occur.

Table 325. FIFO trigger level settings register (FIFOTRIG - offset 0xE08) bit description

Bit	Symbol	Value	Description	Reset value
0	TXLVLENA		Transmit FIFO level trigger enable. The FIFO level trigger will cause an interrupt if enabled in FIFOINTENSET. This field is not used for DMA requests (see DMATX in FIFOCFG).	0x0
		0	Transmit FIFO level does not generate a FIFO level trigger.	
		1	An interrupt will be generated if the transmit FIFO level reaches the value specified by the TXLVL field in this register.	
1	RXLVLENA		Receive FIFO level trigger enable. This trigger will become an interrupt if enabled in FIFOINTENSET. This field is not used for DMA requests (see DMARX in FIFOCFG).	0x0
		0	Receive FIFO level does not generate a FIFO level trigger.	
		1	An interrupt will be generated if the receive FIFO level reaches the value specified by the RXLVL field in this register.	
10:2	-	-	Reserved. Read value is undefined, only zero should be written.	-

Table 325. FIFO trigger level settings register (FIFOTRIG - offset 0xE08) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
11:8	TXLVL		Transmit FIFO level trigger point. This field is used only when TXLVLENA: 1. If enabled to do so, the FIFO level can wake up the device just enough to perform DMA, then return to the reduced power mode. 0: generate an interrupt when the TX FIFO becomes empty. 1: generate an interrupt when the TX FIFO level decreases to one entry. ... 15: generate an interrupt when the TX FIFO level decreases to 15 entries (is no longer full).	0x0
15:12	-	-	Reserved. Read value is undefined, only zero should be written.	-
19:16	RXLVL		Receive FIFO level trigger point. The RX FIFO level is checked when a new piece of data is received. This field is used only when RXLVLENA: 1. If enabled to do so, the FIFO level can wake up the device just enough to perform DMA, then return to the reduced power mode. 0: generate an interrupt when the RX FIFO has one entry (is no longer empty). 1: generate an interrupt when the RX FIFO has two entries. ... 15: generate an interrupt when the RX FIFO increases to 16 entries (has become full).	0x0
31:20	-	-	Reserved. Read value is undefined, only zero should be written.	-

21.6.13 FIFO interrupt enable set and read

This register is used to enable various interrupt sources. The complete set of interrupt enables may be read from this register. Writing ones to implemented bits in this register causes those bits to be set. The FIFOINTENCLR register is used to clear bits in this register.

Table 326. FIFO interrupt enable set and read register (FIFOINTENSET - offset 0xE10) bit description

Bit	Symbol	Value	Description	Reset value
0	TXERR		Determines whether an interrupt occurs when a transmit error occurs, based on the TXERR flag in the FIFOSTAT register.	0x0
		0	No interrupt will be generated for a transmit error.	
		1	An interrupt will be generated when a transmit error occurs.	
1	RXERR		Determines whether an interrupt occurs when a receive error occurs, based on the RXERR flag in the FIFOSTAT register.	0x0
		0	No interrupt will be generated for a receive error.	
		1	An interrupt will be generated when a receive error occurs.	
2	TXLVL		Determines whether an interrupt occurs when a the transmit FIFO reaches the level specified by the TXLVL field in the FIFOTRIG register.	0x0
		0	No interrupt will be generated based on the TX FIFO level.	
		1	If TXLVLENA in the FIFOTRIG register: 1, an interrupt will be generated when the TX FIFO level decreases to the level specified by TXLVL in the FIFOTRIG register.	

Table 326. FIFO interrupt enable set and read register (FIFOINTENSET - offset 0xE10) bit description …continued

Bit	Symbol	Value	Description	Reset value
3	RXLVL		Determines whether an interrupt occurs when a the receive FIFO reaches the level specified by the TXLVL field in the FIFOTRIG register.	0x0
		0	No interrupt will be generated based on the RX FIFO level.	
		1	If RXLVLENA in the FIFOTRIG register: 1, an interrupt will be generated when the when the RX FIFO level increases to the level specified by RXLVL in the FIFOTRIG register.	
31:4	-	-	Reserved. Read value is undefined, only zero should be written.	-

21.6.14 FIFO interrupt enable clear and read

The FIFOINTENCLR register is used to clear interrupt enable bits in FIFOINTENSET. The complete set of interrupt enables may also be read from this register as well as FIFOINTENSET.

Table 327. FIFO interrupt enable clear and read (FIFOINTENCLR - offset 0xE14) bit description

Bit	Symbol	Description	Reset value
0	TXERR	Writing a one to this bit disables the TXERR interrupt.	0x0
1	RXERR	Writing a one to this bit disables the RXERR interrupt.	0x0
2	TXLVL	Writing a one to this bit disables the interrupt caused by the transmit FIFO reaching the level specified by the TXLVL field in the FIFOTRIG register.	0x0
3	RXLVL	Writing a one to this bit disables the interrupt caused by the receive FIFO reaching the level specified by the RXLVL field in the FIFOTRIG register.	0x0
31:4		Reserved. Read value is undefined, only zero should be written.	-

21.6.15 FIFO interrupt status register

The read-only FIFOINTSTAT register provides a view of those interrupt flags that are both pending and currently enabled. This can simplify software handling of interrupts. Refer to the descriptions of interrupts in Section 21.6.11 and Section 21.6.12 for details.

Table 328. FIFO interrupt status register (FIFOINTSTAT - offset 0xE18) bit description

Bit	Symbol	Description	Reset value
0	TXERR	TX FIFO error	0x0
1	RXERR	RX FIFO error	0x0
2	TXLVL	Transmit FIFO level interrupt	0x0
3	RXLVL	Receive FIFO level interrupt	0x0
4	PERINT	Peripheral interrupt	0x0
31:5	-	Reserved. Read value is undefined, only zero should be written	-

21.6.16 FIFO write data register

The FIFOWR register is used to write values to be transmitted to the FIFO.

Table 329. FIFO write data register (FIFOWR - offset 0xE20) bit description

Bit	Symbol	Description	Reset value
8:0	TXDATA	Transmit data to the FIFO	NA
31:9	-	Reserved	-

21.6.17 FIFO read data register

The FIFORD register is used to read values that have been received by the FIFO.

Table 330. FIFO read data register (FIFORD - offset 0xE30) bit description

Bit	Symbol	Description	Reset value
8:0	RXDATA	Received data from the FIFO. The number of bits used depends on the DATALEN and PARITYSEL settings.	NA
12:9	-	Reserved, the value read from a reserved bit is not defined.	-
13	FRAMERR	Framing Error status flag. This bit reflects the status for the data it is read along with from the FIFO, and indicates that the character was received with a missing stop bit at the expected location. This could be an indication of a baud rate or configuration mismatch with the transmitting source.	NA
14	PARITYERR	Parity Error status flag. This bit reflects the status for the data it is read along with from the FIFO. This bit will be set when a parity error is detected in a received character.	NA
15	RXNOISE	Received Noise flag. See description of the RxNoiseInt bit in Table 316.	NA
31:16	-	Reserved, the value read from a reserved bit is not defined.	-

21.6.18 FIFO data read with no FIFO pop

This register acts in exactly the same way as FIFORD, except that it supplies data from the top of the FIFO without popping the FIFO (i.e. leaving the FIFO state unchanged). This could be used to allow system software to observe incoming data without interfering with the peripheral driver.

Table 331. FIFO data read with no FIFO pop (FIFORDNOPOP - offset 0xE40) bit description

Bit	Symbol	Description	Reset value
8:0	RXDATA	Received data from the FIFO	NA
12:9	-	Reserved, the value read from a reserved bit is not defined	-
13	FRAMERR	Framing error status flag	NA
14	PARITYERR	Parity error status flag	NA
15	RXNOISE	Received noise flag	NA
31:16	-	Reserved, the value read from a reserved bit is not defined	-

21.6.19 Module identification register

The ID register identifies the type and revision of the USART module. A generic SW driver can make use of this information register to implement module type or revision specific behavior.

Table 332. Module identification register (ID - offset 0xFFC) bit description

Bit	Symbol	Description	Reset Value
7:0	APERTURE	Aperture: encoded as (aperture size/4K) -1, so 0x00 means a 4K aperture.	0x0
11:8	MINOR_REV	Minor revision of module implementation, starting at 0. Minor revision of module implementation, starting at 0. Software compatibility is expected between minor revisions.	-
15:12	MAJOR_REV	Major revision of module implementation, starting at 0. There may not be software compatibility between major revisions.	-
31:16	ID	Unique module identifier for this IP block.	0xE010

21.7 Functional description

21.7.1 AHB bus access

The bus interface to the USART registers contained in the Flexcomm Interface support only word writes. Byte and halfword writes are not supported in conjunction with the USART function.

21.7.2 Clocking and baud rates

In order to use the USART, clocking details must be defined such as setting up the clock source selection, the BRG, and setting up the FRG if it is the selected clock source.

Also see Section 21.3.1.

21.7.2.1 Fractional Rate Generator (FRG)

The Fractional Rate Generator can be used to obtain more precise baud rates when the function clock is not a good multiple of standard (or otherwise desirable) baud rates.

The FRG is typically set up to produce an integer multiple of the highest required baud rate, or a very close approximation. The BRG is then used to obtain the actual baud rate needed.

The FRG register controls the Fractional Rate Generator, which provides the base clock that may be used by any Flexcomm Interface. The Fractional Rate Generator creates a lower rate output clock by suppressing selected input clocks. When not needed, the value of 0 can be set for the FRG, which will then not divide the input clock.

The FRG output clock is defined as the input clock divided by 1 + (MULT / 256), where MULT is in the range of 1 to 255. This allows producing an output clock that ranges from the input clock divided by 1+1/256 to 1+255/256 (just more than 1 to just less than 2). Any further division can be done specific to each USART block by the integer BRG divider contained in each USART.

The base clock produced by the FRG cannot be perfectly symmetrical, so the FRG distributes the output clocks as evenly as is practical. Since USARTs normally uses 16x overclocking, the jitter in the fractional rate clock in these cases tends to disappear in the ultimate USART output.

For setting up the fractional divider, see Section 2.5.14.

21.7.2.2 Baud Rate Generator (BRG)

The Baud Rate Generator (see Section 21.6.6) is used to divide the base clock to produce a rate 16 times the desired baud rate. Typically, standard baud rates can be generated by integer divides of higher baud rates.

21.7.2.3 32 kHz mode

In order to use a 32 kHz clock to operate a USART at any reasonable speed, a number of adaptations need to be made. First, 16x overclocking has to be abandoned. Otherwise, the maximum data rate would be very low. For the same reason, multiple samples of each data bit must be reduced to one. Finally, special clocking has to be used for individual bit times because 32 kHz is not particularly close to an even multiple of any standard baud rate.

When 32 kHz mode is enabled, clocking comes from the selected 32kHz clock. The FRG is bypassed, and the BRG can be used to divide down the default 9600 baud to lower rates. Other adaptations required to make the USART work for rates up to 9600 baud are done internally. Rate error will be less than one half percent in this mode.

21.7.3 DMA

A DMA request is provided for each USART direction, and can be used in lieu of interrupts for transferring data by configuring the DMA controller and FIFO level triggering appropriately. The DMA controller provides an acknowledgement signal that clears the related request when it completes handling a that request. The transmitter DMA request is asserted when the transmitter can accept more data. The receiver DMA request is asserted when received data is available to be read.

When DMA is used to perform USART data transfers, other mechanisms can be used to generate interrupts when needed. For instance, completion of the configured DMA transfer can generate an interrupt from the DMA controller. Also, interrupts for special conditions, such as a received break, can still generate useful interrupts.

21.7.4 Synchronous mode

In synchronous mode, a master generates a clock as defined by the clock selection and BRG, which is used to transmit and receive data. As a slave, the external clock is used to transmit and receive data. There is no overclocking in either case.

21.7.5 Flow control

The USART supports both hardware and software flow control.

21.7.5.1 Hardware flow control

The USART supports hardware flow control using RTS and/or CTS signalling. If RTS is configured to appear on a device pin so that it can be sent to an external device, it indicates to an external device the ability of the receiver to receive more data. It can also be used internally to throttle the transmitter from the receiver, which can be especially useful if loopback mode is enabled.

If connected to a pin, and if enabled to do so, the CTS input can allow an external device to throttle the USART transmitter. Both internal and external CTS can be used separately or together.

Figure 66 shows an overview of RTS and CTS within the USART.

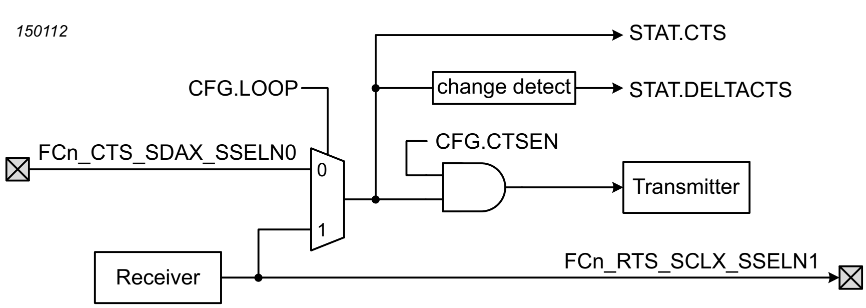


Fig 66. Hardware flow control using RTS and CTS

21.7.5.2 Software flow control

Software flow control could include XON / XOFF flow control, or other mechanisms. these are supported by the ability to check the current state of the CTS input, and/or have an interrupt when CTS changes state (via the CTS and DELTACTS bits, respectively, in the STAT register), and by the ability of software to gracefully turn off the transmitter (via the TXDIS bit in the CTL register).

21.7.6 Auto-baud function

The auto-baud functions attempts to measure the start bit time of the next received character. For this to work, the measured character must have a 1 in the least significant bit position, so that the start bit is bounded by a falling and rising edge. Before an auto-baud operation is requested, the BRG value must be set to 0. The measurement is made using the current clocking settings, including the oversampling configuration. The result is that a value is stored in the BRG register that is as close as possible to the correct setting for the sampled character and the current clocking settings. The sampled character is provided in the RXDAT and RXDATSTAT registers, allowing software to double check for the expected character.

Auto-baud includes a time-out that is flagged by ABERR if no character is received at the expected time. It is recommended that auto-baud only be enabled when the USART receiver is idle. Once enabled, either data will become available in the FIFO or ABERR will be asserted at some point, at which time software should turn off auto-baud.

Auto-baud has no meaning and should not be enabled when the USART is in synchronous mode.

21.7.7 RS-485 support

RS-485 support requires some form of address recognition and data direction control.

This USART has provisions for hardware address recognition (see the AUTOADDR bit in the CFG register in Section 21.6.1 and the ADDR register in Section 21.6.9), as well as software address recognition (see the ADDRDET bit in the CTL register in Section 21.6.2).

Automatic data direction control with the RTS pin can be set up using the OESEL, OEPOL, and OETA bits in the CFG register (Section 21.6.1). Data direction control can also be implemented in software using a GPIO pin.

21.7.8 Oversampling

Typical industry standard USARTs use a 16x oversample clock to transmit and receive asynchronous data. This is the number of BRG clocks used for one data bit. The Oversample Select Register (OSR) allows this USART to use a 16x down to a 5x oversample clock. There is no oversampling in synchronous modes.

Reducing the oversampling can sometimes help in getting better baud rate matching when the baud rate is very high, or the function clock is very low. For example, the closest actual rate near 115,200 baud with a 12 MHz function clock and 16x oversampling is 107,143 baud, giving a rate error of 7%. Changing the oversampling to 15x gets the actual rate to 114,286 baud, a rate error of 0.8%. Reducing the oversampling to 13x gets the actual rate to 115,385 baud, a rate error of only 0.16%.

There is a cost for altering the oversampling. In asynchronous modes, the USART takes three samples of incoming data on consecutive oversample clocks, as close to the center of a bit time as can be done. When the oversample rate is reduced, the three samples spread out and occupy a larger proportion of a bit time. For example, with 5x oversampling, there is one oversample clock, then three data samples taken, then one more oversample clock before the end of the bit time. Since the oversample clock is running asynchronously from the input data, skew of the input data relative to the expected timing has little room for error. At 16x oversampling, there are several oversample clocks before actual data sampling is done, making the sampling more robust. Generally speaking, it is recommended to use the highest oversampling where the rate error is acceptable in the system.

21.7.9 Break generation and detection

A line break may be sent at any time, regardless of other USART activity. Received break is also detected at any time, including during reception of a character. Received break is signaled when the RX input remains low for 16 bit times. Both the beginning and end of a received break are noted by the DELTARXBRK status flag, which can be used as an interrupt. See Section 21.7.10 for details of LIN mode break.

In order to avoid corrupting any character currently being transmitted, it is recommended that the USART transmitter be disabled by setting the TXDIS bit in the CTL register, then waiting for the TXDISSTAT flag to be set prior to sending a break. Then a 1 may be written to the TXBRKEN bit in the CTL register. This sends a break until TXBRKEN is cleared, allowing any length break to be sent.

21.7.10 LIN bus

The only difference between standard operation and LIN mode is that LIN mode alters the way that break generation and detection is performed (see Section 21.7.9 for details of the standard break). When a break is requested by setting the TXBRKEN bit in the CTL register, then sending a dummy character, a 13 bit time break is sent. A received break is flagged when the RX input remains low for 11 bit times. As for non-LIN mode, a received character is also flagged, and accompanied by a framing error status.

As a LIN slave, the auto-baud feature can be used to synchronize to a LIN sync byte, and will return the value of the sync byte as confirmation of success.

Wake-up for LIN can potentially be handled in a number of ways, depending on the system, and what clocks may be running in a slave device. For instance, as long as the USART is receiving internal clocks allowing it to function, it can be set to wake up the CPU for any interrupt, including a received start bit. If there are no clocks running, the GPIO function of the USART RX pin can be programmed to wake up the device.

21.7.11 7816 Protocol Support

7816 is a half-duplex protocol, it can be accomplished by enabling DIO_MODE bit in IOMODE register. The DIO_OEN bit in the same register determines whether it works as a transmitter or a receiver.

22. Serial Peripheral Interfaces (SPI)

22.1 Introduction

SPI functions are available on all QN908x devices as a selectable function in Flexcomm2 and Flexcomm3 peripheral.

22.2 Features

- Master and slave operation.
- Data transmits of 4 to 16 bits supported directly. Larger frames supported by software.
- The SPI function supports separate transmit and receive FIFOs with 8 entries each
- Supports DMA transfers: SPIn transmit and receive functions can be operated with the system DMA controller.
- Data can be transmitted to a slave without the need to read incoming data. This can be useful while setting up an SPI memory.
- Up to four Slave Select input/outputs with selectable polarity and flexible usage.
- The SPI function supports 3-wire mode (SCK, SSEL, DATA).

Remark: Texas Instruments SSI and National Microwire modes are not supported.

22.3 Basic configuration

Initial configuration of an SPI peripheral is accomplished as follows:

- If needed, configure RST_SW_SET and RST_SW_CLR registers to reset the Flexcomm that is about to have a specific peripheral function selected.
- Configure the required Flexcomm pin functions through IOCON (see Section 22.4).
- Configure the SPI for receiving and transmitting data:
  - Configure CLK_EN register to enable clock for the related Flexcomm.
  - Select the desired Flexcomm function by writing to the PSELID register of the related Flexcomm (see Section 20.6.2).
  - Enable or disable the related Flexcomm interrupts in the NVIC (see Table 49).
  - Configure the FIFOs for operation.
  - Configure the Flexcomm Interface clock and SPI data rate (see Section 21.7.2).
  - Set the RXIGNORE bit to only transmit data and not read the incoming data. Otherwise, the transmit halts when the FIFORD buffer is full.
- Configure the SPI function to wake up the part from sleep mode (see Section 21.3.2).

22.3.1 Configure the SPI for wake-up

In sleep mode, any signal that triggers an SPI interrupt can wake up the part, provided that the interrupt is enabled in the INTENSET register and the NVIC. As long as the SPI clock is configured to be active in sleep mode, the SPI can wake up the part independently of whether the SPI block is configured in master or slave mode.

22.3.1.1 Wake-up from sleep mode

- Configure the SPI in either master or slave mode. See Table 336.
- Enable the SPI interrupt in the NVIC.
- Any enabled SPI interrupt wakes up the part from sleep mode.

22.4 Pin description

The SPI signals are movable Flexcomm functions and are assigned to external pins via PIO_FUNC_CFGn registers. See Section 2.5.27~Section 2.5.30.

Table 333: SPI Pin Description

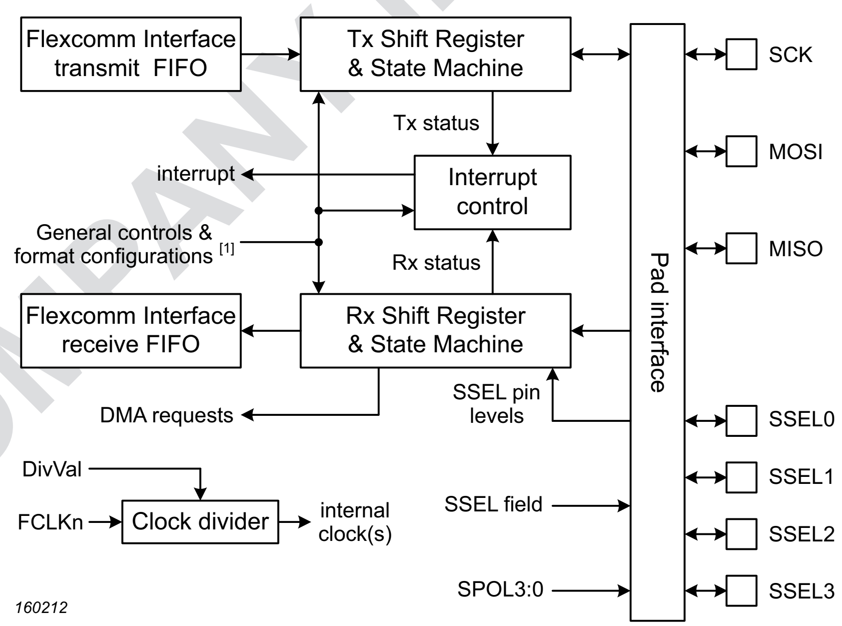
Function	Type	Pin name used in Pin Description chapter	Description
SCK	I/O	FCn_SCK	Serial Clock for SPI on Flexcomm Interface n. SCK is a clock signal used to synchronize the transfer of data. It is driven by the master and received by the slave. When the SPI interface is used, the clock is programmable to be active-high or active-low. SCK only switches during a data transfer. It is driven whenever the Master bit in CFG equals 1, regardless of the state of the Enable bit.
MOSI	I/O	FCn_RXD_SDA_MOSI or FCn_RXD_SDA_MOSI_DATA	Master Out Slave In for SPI on Flexcomm Interface n. The MOSI signal transfers serial data from the master to the slave. When the SPI is a master, it outputs serial data on this signal. When the SPI is a slave, it clocks in serial data from this signal. MOSI is driven whenever the Master bit in SPInCfg equals 1, regardless of the state of the Enable bit.
MISO	I/O	FCn_TXD_SCL_MISO or FCn_TXD_SCL_MISO_WS	Master In Slave Out for SPI on Flexcomm Interface n. The MISO signal transfers serial data from the slave to the master. When the SPI is a master, serial data is input from this signal. When the SPI is a slave, serial data is output to this signal. MISO is driven when the SPI block is enabled, the Master bit in CFG equals 0, and when the slave is selected by one or more SSEL signals.
SSEL0	I/O	FCn_CTS_SDA_SSEL0	Slave Select 0 for SPI on Flexcomm Interface n. When the SPI interface is a master, it will drive the SSEL signals to an active state before the start of serial data and then release them to an inactive state after the serial data has been sent. By default, this signal is active low but can be selected to operate as active high. When the SPI is a slave, any SSEL in an active state indicates that this slave is being addressed. The SSEL pin is driven whenever the Master bit in the CFG register equals 1, regardless of the state of the Enable bit.
SSEL1	I/O	FCn_RTS_SCL_SSEL1	Slave Select 1 for SPI on Flexcomm Interface n.
SSEL2	I/O	FCn_SSEL2	Slave Select 2 for SPI on Flexcomm Interface n.
SSEL3	I/O	FCn_SSEL3	Slave Select 3 for SPI on Flexcomm Interface n.

Recommended IOCON settings are shown in below table.

Table 334: SPI pin configuration example

Pin	Type	Name used in pin configuration chapter	PAD name	PINMUX register and configure value
SCK	I/O	FC2_SCK	PA29	PIO_FUNC_CFG3[22:20]: 0x4 (FUNC4)
MOSI	I/O	FC2_RXD_SDA_MOSI	PA26	PIO_FUNC_CFG3[10:8]: 0x4 (FUNC4)
MISO	I/O	FC2_TXD_SCL_MISO	PA27	PIO_FUNC_CFG3[14:12]: 0x4 (FUNC4)
SSEL0	I/O	FC2_CTS_SDA_SSEL0	PA22	PIO_FUNC_CFG2[26:24]: 0x4 (FUNC4)
SSEL1	I/O	FC0_RTS_SCL_SSEL1	PA23	PIO_FUNC_CFG2[30:28]: 0x4 (FUNC4)
SSEL2	I/O	FC2_SSEL2	PA11	PIO_FUNC_CFG1[14:12]: 0x4 (FUNC4)
SSEL3	I/O	FC2_SSEL3	PA21	PIO_FUNC_CFG2[22:20]: 0x4 (FUNC4)

22.5 General description



[1] Includes CPOL, CPHA, LSBF, LEN, master, enable, transfer_delay, frame_delay, pre_delay, post_delay, SOT, EOT, EOF, RXIGNORE, individual interrupt enables.

Fig 67. SPI block diagram

22.6 Register description

The base addresses of all Flexcomm interfaces can be found in Table 305.

Address offsets are within the address space of the related Flexcomm Interface. The Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

Table 335. SPI register overview

Name	Access	Offset	Description	Reset value	Section
Registers for the SPI function:					
CFG	R/W	0x400	SPI Configuration register.	0x0	22.6.1
DLY	R/W	0x404	SPI Delay register.	0x0	22.6.2
STAT	R/W	0x408	SPI Status. Some status flags can be cleared by writing a 1 to that bit position.	0x100	22.6.3
INTENSET	R/W	0x40C	SPI Interrupt Enable read and Set. A complete value may be read from this register. Writing a 1 to any implemented bit position causes that bit to be set.	0x0	22.6.4
INTENCLR	WO	0x410	SPI Interrupt Enable Clear. Writing a 1 to any implemented bit position causes the corresponding bit in INTENSET to be cleared.	NA	22.6.5
DIV	R/W	0x424	SPI clock Divider.	0x0	22.6.6
INTSTAT	RO	0x428	SPI Interrupt Status.	0x0	22.6.7
Registers for FIFO control and data access:					
FIFOCFG	R/W	0xE00	FIFO configuration and enable register.	0x0	22.6.8
FIFOSTAT	R/W	0xE04	FIFO status register.	0x30	22.6.9
FIFOTRIG	R/W	0xE08	FIFO trigger level settings for interrupt and DMA request.	0x0	22.6.10
FIFOINTENSET	R/W1C	0xE10	FIFO interrupt enable set (enable) and read register.	0x0	22.6.11
FIFOINTENCLR	R/W1C	0xE14	FIFO interrupt enable clear (disable) and read register.	0x0	22.6.12
FIFOINTSTAT	RO	0xE18	FIFO interrupt status register.	0x0	22.6.13
FIFOWR	WO	0xE20	FIFO write data.	NA	22.6.14
FIFORD	RO	0xE30	FIFO read data.	NA	22.6.15
FIFORDNOPOP	RO	0xE40	FIFO data read with no FIFO pop.	NA	22.6.16
ID register:					
ID	RO	0xFFC	SPI module Identification. This value appears in the shared Flexcomm Interface peripheral ID register when SPI is selected.	see table	22.6.17

22.6.1 SPI Configuration register

The CFG register contains information for the general configuration of the SPI. Typically, this information is not changed during operation. See the description of the master idle status (MSTIDLE in Table 338) for more information.

Remark: A setup sequence is recommended for initial SPI setup (after the SPI function has been selected (see Section 20), and when changes need to be made to settings in the CFG register after the interface has been in use. See the list below. In the case of changing existing settings, the interface should first be disabled by clearing the ENABLE bit once the interface is fully idle. See the description of the master idle status (MSTIDLE in Table 338) for more information.

- Disable the FIFO by clearing the ENABLETX and ENABLERX bits in FIFOCFG
- Setup the SPI interface in the CFG register, leaving ENABLE: 0.
- Enable the FIFO by setting the ENABLETX and/or ENABLERX bits in FIFOCFG
- Enable the SPI by setting the ENABLE bit in CFG.

Table 336. SPI Configuration register (CFG, offset 0x400) bit description

Bit	Symbol	Value	Description	Reset value
0	ENABLE		SPI enable.	0x0
		0	Disabled. The SPI is disabled and the internal state machine and counters are reset.	
		1	Enabled. The SPI is enabled for operation.	
1	-	-	Reserved. Read value is undefined, only zero should be written.	-
2	MASTER		Master mode select.	0x0
		0	Slave mode. The SPI will operate in slave mode. SCK, MOSI, and the SSEL signals are inputs, MISO is an output.	
		1	Master mode. The SPI will operate in master mode. SCK, MOSI, and the SSEL signals are outputs, MISO is an input.	
3	LSBF		LSB First mode enable.	0x0
		0	Standard. Data is transmitted and received in standard MSB first order.	
		1	Reverse. Data is transmitted and received in reverse order (LSB first).	
4	CPHA		Clock Phase select.	0x0
		0	Change. The SPI captures serial data on the first clock transition of the transfer (when the clock changes away from the rest state). Data is changed on the following edge.	
		1	Capture. The SPI changes serial data on the first clock transition of the transfer (when the clock changes away from the rest state). Data is captured on the following edge.	
5	CPOL		Clock Polarity select.	0x0
		0	Low. The rest state of the clock (between transfers) is low.	
		1	High. The rest state of the clock (between transfers) is high.	
6	-	-	Reserved. Read value is undefined, only zero should be written.	-
7	LOOP		Loopback mode enable. Loopback mode applies only to Master mode, and connects transmit and receive data connected together to allow simple software testing.	0x0
		0	Disabled.	
		1	Enabled.	

Table 336. SPI Configuration register (CFG, offset 0x400) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
8	SPOL0		SSEL0 Polarity select.	0x0
		0	Low. The SSEL0 pin is active low.	
		1	High. The SSEL0 pin is active high.	
9	SPOL1		SSEL1 Polarity select.	0x0
		0	Low. The SSEL1 pin is active low.	
		1	High. The SSEL1 pin is active high.	
10	SPOL2		SSEL2 Polarity select.	0x0
		0	Low. The SSEL2 pin is active low.	
		1	High. The SSEL2 pin is active high.	
11	SPOL3		SSEL3 Polarity select.	0x0
		0	Low. The SSEL3 pin is active low.	
		1	High. The SSEL3 pin is active high.	
31:12	-	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.2 SPI Delay register

The DLY register controls several programmable delays related to SPI signalling. These delays apply only to master mode, and are all stated in SPI clocks.

Timing details are shown in Section 22.7.3.1, Section 22.7.3.2, and Section 22.7.3.3.

Table 337. SPI Delay register (DLY, offset 0x404) bit description

Bit	Symbol	Description	Reset value
3:0	PRE_ DELAY	Controls the amount of time between SSEL assertion and the beginning of a data transfer. There is always one SPI clock time between SSEL assertion and the first clock edge. This is not considered part of the pre-delay. 0x0: No additional time is inserted. 0x1: 1 SPI clock time is inserted. 0x2: 2 SPI clock times are inserted. ... 0xF: 15 SPI clock times are inserted.	0x0
7:4	POST_ DELAY	Controls the amount of time between the end of a data transfer and SSEL deassertion. 0x0: No additional time is inserted. 0x1: 1 SPI clock time is inserted. 0x2: 2 SPI clock times are inserted. ... 0xF: 15 SPI clock times are inserted.	0x0

Table 337. SPI Delay register (DLY, offset 0x404) bit description

Bit	Symbol	Description	Reset value
11:8	FRAME_ DELAY	If the EOF flag is set, controls the minimum amount of time between the current frame and the next frame (or SSEL deassertion if EOT). 0x0: No additional time is inserted. 0x1: 1 SPI clock time is inserted. 0x2: 2 SPI clock times are inserted. ... 0xF: 15 SPI clock times are inserted.	0x0
15:12	TRANSFER _ DELAY	Controls the minimum amount of time that the SSEL is deasserted between transfers. 0x0: The minimum time that SSEL is deasserted is 1 SPI clock time. (Zero added time.) 0x1: The minimum time that SSEL is deasserted is 2 SPI clock times. 0x2: The minimum time that SSEL is deasserted is 3 SPI clock times. ... 0xF: The minimum time that SSEL is deasserted is 16 SPI clock times.	0x0
31:16	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.3 SPI Status register

The STAT register provides SPI status flags for software to read, and a control bit for forcing an end of transfer. Flags other than read-only flags may be cleared by writing ones to corresponding bits of STAT.

In this register, the following notation is used: RO: Read-only, W1C: write 1 to clear.

Table 338. SPI Status register (STAT, offset 0x408) bit description

Bit	Symbol	Description	Reset value	Access
3:0	-	Reserved. Read value is undefined, only zero should be written.	-	-
4	SSA	Slave Select Assert. This flag is set whenever any slave select transitions from deasserted to asserted, in both master and slave modes. This allows determining when the SPI transmit/receive functions become busy, and allows waking up the device from reduced power modes when a slave mode access begins. This flag is cleared by software.	0x0	W1C
5	SSD	Slave Select Deassert. This flag is set whenever any asserted slave selects transition to deasserted, in both master and slave modes. This allows determining when the SPI transmit/receive functions become idle. This flag is cleared by software.	0x0	W1C
6	STALLED	Stalled status flag. This indicates whether the SPI is currently in a stall condition.	0x0	RO
7	END TRANSFER	End Transfer control bit. Software can set this bit to force an end to the current transfer when the transmitter finishes any activity already in progress, as if the EOT flag had been set prior to the last transmission. This capability is included to support cases where it is not known when transmit data is written that it will be the end of a transfer. The bit is cleared when the transmitter becomes idle as the transfer comes to an end. Forcing an end of transfer in this manner causes any specified FRAME_DELAY and TRANSFER_DELAY to be inserted.	0x0	RO/ W1C
8	MSTIDLE	Master idle status flag. This bit is 1 whenever the SPI master function is fully idle. This means that the transmit holding register is empty and the transmitter is not in the process of sending data.	0x1	RO
31:9	-	Reserved. Read value is undefined, only zero should be written.	-	-

22.6.4 SPI Interrupt Enable read and Set register

The INTENSET register is used to enable various SPI interrupt sources. Enable bits in INTENSET are mapped in locations that correspond to the flags in the STAT register. The complete set of interrupt enables may be read from this register. Writing ones to implemented bits in this register causes those bits to be set. The INTENCLR register is used to clear bits in this register. See Table 338 for details of the interrupts.

Table 339. SPI Interrupt Enable read and Set register (INTENSET, offset 0x40C) bit description

Bit	Symbol	Value	Description	Reset value
3:0	-	-	Reserved. Read value is undefined, only zero should be written.	-
4	SSAEN		Slave select assert interrupt enable. Determines whether an interrupt occurs when the Slave Select is asserted.	0x0
		0	Disabled. No interrupt will be generated when any Slave Select transitions from deasserted to asserted.	
		1	Enabled. An interrupt will be generated when any Slave Select transitions from deasserted to asserted.	
5	SSDEN		Slave select deassert interrupt enable. Determines whether an interrupt occurs when the Slave Select is deasserted.	0x0
		0	Disabled. No interrupt will be generated when all asserted Slave Selects transition to deasserted.	
		1	Enabled. An interrupt will be generated when all asserted Slave Selects transition to deasserted.	
7:6	-	-	Reserved. Read value is undefined, only zero should be written.	-
8	MSTIDLEEN		Master idle interrupt enable.	0x0
		0	No interrupt will be generated when the SPI master function is idle.	
		1	An interrupt will be generated when the SPI master function is fully idle.	
31:9	-	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.5 SPI Interrupt Enable Clear register

The INTENCLR register is used to clear interrupt enable bits in the INTENSET register.

Table 340. SPI Interrupt Enable clear register (INTENCLR, offset 0x410) bit description

Bit	Symbol	Description	Reset value
3:0	-	Reserved. Read value is undefined, only zero should be written.	-
4	SSAEN	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
5	SSDEN	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
7:6	-	Reserved. Read value is undefined, only zero should be written.	-
8	MSTIDLE	Writing 1 clears the corresponding bit in the INTENSET register.	0x0
31:9	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.6 SPI Divider register

The DIV register determines the clock used by the SPI in master mode.

For details on clocking, see Section 21.7.2.

Table 341. SPI Divider register (DIV, offset 0x424) bit description

Bit	Symbol	Description	Reset Value
15:0	DIVVAL	Rate divider value. Specifies how the Flexcomm Interface clock (FCLK) is divided to produce the SPI clock rate in master mode. DIVVAL is -1 encoded such that the value 0 results in FCLK/1, the value 1 results in FCLK/2, up to the maximum possible divide value of 0xFFFF, which results in FCLK/65536.	0x0
31:16	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.7 SPI Interrupt Status register

The read-only INTSTAT register provides a view of those interrupt flags that are currently enabled. This can simplify software handling of interrupts. See Table 338 for detailed descriptions of the interrupt flags.

Table 342. SPI Interrupt Status register (INTSTAT, offset 0x428) bit description

Bit	Symbol	Description	Reset value
3:0	-	Reserved. Read value is undefined, only zero should be written.	-
4	SSA	Slave Select Assert.	0x0
5	SSD	Slave Select Deassert.	0x0
7:6	-	Reserved. Read value is undefined, only zero should be written.	-
8	MSTIDLE	Master Idle status flag.	0x0
31:9	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.8 FIFO Configuration register

This register configures FIFO usage. A peripheral function within the Flexcomm Interface must be selected prior to configuring the FIFO.

Table 343. FIFO Configuration register (FIFOCFG - offset 0xE00) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ENABLETX		Enable the transmit FIFO.	0x0	R/W
		0	The transmit FIFO is not enabled.		
		1	The transmit FIFO is enabled.		
1	ENABLERX		Enable the receive FIFO.	0x0	R/W
		0	The transmit FIFO is not enabled.		
		1	The transmit FIFO is enabled.		
3:2	-	-	Reserved. Read value is undefined, only zero should be written.	-	-
5:4	SIZE		FIFO size configuration. This is a read-only field. 0x1: FIFO is configured as 8 entries of 16 bits. 0x0, 0x2, 0x3: not applicable to SPI.	NA	RO
11:6	-	-	Reserved. Read value is undefined, only zero should be written.	-	-
12	DMATX		DMA configuration for transmit.	0x0	R/W
		0	DMA is not used for the transmit function.		
		1	Generate a DMA request for the transmit function if the FIFO is not full. Generally, data interrupts would be disabled if DMA is enabled.		
13	DMARX		DMA configuration for receive.	0x0	R/W
		0	DMA is not used for the receive function.		
		1	Generate a DMA request for the receive function if the FIFO is not empty. Generally, data interrupts would be disabled if DMA is enabled.		
15:14	-	-	Reserved.	-	-
16	EMPTYTX	-	Empty command for the transmit FIFO. When a 1 is written to this bit, the TX FIFO is emptied.	NA	WO
17	EMPTYRX	-	Empty command for the receive FIFO. When a 1 is written to this bit, the RX FIFO is emptied.	NA	WO
31:18	-	-	Reserved. Read value is undefined, only zero should be written.	-	-

22.6.9 FIFO status register

This register provides status information for the FIFO and also indicates an interrupt from the peripheral function.

Table 344. FIFO status register (FIFOSTAT - offset 0xE04) bit description

Bit	Symbol	Description	Reset value	Access
0	TXERR	TX FIFO error. Will be set if a transmit FIFO error occurs. This could be an overflow caused by pushing data into a full FIFO, or by an underflow if the FIFO is empty when data is needed. Cleared by writing a 1 to this bit.	0x0	R/W1C
1	RXERR	RX FIFO error. Will be set if a receive FIFO overflow occurs, caused by software or DMA not emptying the FIFO fast enough. Cleared by writing a 1 to this bit.	0x0	R/W1C
2	-	Reserved. Read value is undefined, only zero should be written.	-	-

Table 344. FIFO status register (FIFOSTAT - offset 0xE04) bit description

Bit	Symbol	Description	Reset value	Access
3	PERINT	Peripheral interrupt. When 1, this indicates that the peripheral function has asserted an interrupt. The details can be found by reading the peripheral's STAT register.	0x0	RO
4	TXEMPTY	Transmit FIFO empty. When 1, the transmit FIFO is empty. The peripheral may still be processing the last piece of data.	0x1	RO
5	TXNOTFULL	Transmit FIFO not full. When 1, the transmit FIFO is not full, so more data can be written. When 0, the transmit FIFO is full and another write would cause it to overflow.	0x1	RO
6	RXNOTEMPTY	Receive FIFO not empty. When 1, the receive FIFO is not empty, so data can be read. When 0, the receive FIFO is empty.	0x0	RO
7	RXFULL	Receive FIFO full. When 1, the receive FIFO is full. Data needs to be read out to prevent the peripheral from causing an overflow.	0x0	RO
12:8	TXLVL	Transmit FIFO current level. A 0 means the TX FIFO is currently empty, and the TXEMPTY and TXNOTFULL flags will be 1. Other values tell how much data is actually in the TX FIFO at the point where the read occurs. If the TX FIFO is full, the TXEMPTY and TXNOTFULL flags will be 0.	0x0	RO
15:13	-	Reserved. Read value is undefined, only zero should be written.	-	-
20:16	RXLVL	Receive FIFO current level. A 0 means the RX FIFO is currently empty, and the RXFULL and RXNOTEMPTY flags will be 0. Other values tell how much data is actually in the RX FIFO at the point where the read occurs. If the RX FIFO is full, the RXFULL and RXNOTEMPTY flags will be 1.	0x0	RO
31:21	-	Reserved. Read value is undefined, only zero should be written.	-	-

22.6.10 FIFO trigger settings register

This register allows selecting when FIFO-level related interrupts occur.

Table 345. FIFO trigger settings register (FIFOTRIG - offset 0xE08) bit description

Bit	Symbol	Value	Description	Reset value
0	TXLVLENA		Transmit FIFO level trigger enable. The FIFO level trigger will cause an interrupt if enabled in FIFOINTENSET. This field is not used for DMA requests (see DMATX in FIFOCFG).	0x0
		0	Transmit FIFO level does not generate a FIFO level trigger.	
		1	An interrupt will be generated if the transmit FIFO level reaches the value specified by the TXLVL field in this register.	
1	RXLVLENA		Receive FIFO level trigger enable. This trigger will become an interrupt if enabled in FIFOINTENSET. This field is not used for DMA requests (see DMARX in FIFOCFG).	0x0
		0	Receive FIFO level does not generate a FIFO level trigger.	
		1	An interrupt will be generated if the receive FIFO level reaches the value specified by the RXLVL field in this register.	
7:2	-	-	Reserved. Read value is undefined, only zero should be written.	-

Table 345. FIFO trigger settings register (FIFOTRIG - offset 0xE08) bit description

Bit	Symbol	Value	Description	Reset value
11:8	TXLVL		Transmit FIFO level trigger point. This field is used only when TXLVLENA: 1. If enabled to do so, the FIFO level can wake up the device just enough to perform DMA, then return to the reduced power mode. 0: generate an interrupt when the TX FIFO becomes empty. 1: generate an interrupt when the TX FIFO level decreases to one entry. ... 7: generate an interrupt when the TX FIFO level decreases to 7 entries (is no longer full).	0x0
15:12	-	-	Reserved. Read value is undefined, only zero should be written.	-
19:16	RXLVL		Receive FIFO level trigger point. The RX FIFO level is checked when a new piece of data is received. This field is used only when RXLVLENA: 1. If enabled to do so, the FIFO level can wake up the device just enough to perform DMA, then return to the reduced power mode. 0: generate an interrupt when the RX FIFO has one entry (is no longer empty). 1: generate an interrupt when the RX FIFO has two entries. ... 7: generate an interrupt when the RX FIFO increases to 8 entries (has become full).	0x0
31:20	-	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.11 FIFO interrupt enable set and read

This register is used to enable various interrupt sources. The complete set of interrupt enables may be read from this register. Writing ones to implemented bits in this register causes those bits to be set. The FIFOINTENCLR register is used to clear bits in this register.

Table 346. FIFO interrupt enable set and read register (FIFOINTENSET - offset 0xE10) bit description

Bit	Symbol	Value	Description	Reset value
0	TXERR		Determines whether an interrupt occurs when a transmit error occurs, based on the TXERR flag in the FIFOSTAT register.	0x0
		0	No interrupt will be generated for a transmit error.	
		1	An interrupt will be generated when a transmit error occurs.	
1	RXERR		Determines whether an interrupt occurs when a receive error occurs, based on the RXERR flag in the FIFOSTAT register.	0x0
		0	No interrupt will be generated for a receive error.	
		1	An interrupt will be generated when a receive error occurs.	
2	TXLVL		Determines whether an interrupt occurs when a the transmit FIFO reaches the level specified by the TXLVL field in the FIFOTRIG register.	0x0
		0	No interrupt will be generated based on the TX FIFO level.	
		1	If TXLVLENA in the FIFOTRIG register: 1, an interrupt will be generated when the TX FIFO level decreases to the level specified by TXLVL in the FIFOTRIG register.	
3	RXLVL		Determines whether an interrupt occurs when a the receive FIFO reaches the level specified by the TXLVL field in the FIFOTRIG register.	0x0
		0	No interrupt will be generated based on the RX FIFO level.	
		1	If RXLVLENA in the FIFOTRIG register: 1, an interrupt will be generated when the when the RX FIFO level increases to the level specified by RXLVL in the FIFOTRIG register.	
31:4	-	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.12 FIFO interrupt enable clear and read

The FIFOINTENCLR register is used to clear interrupt enable bits in FIFOINTENSET. The complete set of interrupt enables may also be read from this register as well as FIFOINTENSET.

Table 347. FIFO interrupt enable clear and read (FIFOINTENCLR - offset 0xE14) bit description

Bit	Symbol	Description	Reset value
0	TXERR	Writing a one to this bit disables the TXERR interrupt.	0x0
1	RXERR	Writing a one to this bit disables the RXERR interrupt.	0x0
2	TXLVL	Writing a one to this bit disables the interrupt caused by the transmit FIFO reaching the level specified by the TXLVL field in the FIFOTRIG register.	0x0
3	RXLVL	Writing a one to this bit disables the interrupt caused by the receive FIFO reaching the level specified by the RXLVL field in the FIFOTRIG register.	0x0
31:4	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.13 FIFO interrupt status register

The read-only FIFOINTSTAT register provides a view of those interrupt flags that are both pending and currently enabled. This can simplify software handling of interrupts. Refer to the descriptions of interrupts in Section 22.6.9 and Section 22.6.10 for details.

Table 348. FIFO interrupt status register (FIFOINTSTAT - offset 0xE18) bit description

Bit	Symbol	Description	Reset value
0	TXERR	TX FIFO error.	0x0
1	RXERR	RX FIFO error.	0x0
2	TXLVL	Transmit FIFO level interrupt.	0x0
3	RXLVL	Receive FIFO level interrupt.	0x0
4	PERINT	Peripheral interrupt.	0x0
31:5	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.14 FIFO write data register

The FIFOWR register is used to write values to be transmitted to the FIFO.

FIFOWR provides the possibility of altering some SPI controls at the same time as sending new data. For example, this can allow a series of SPI transactions involving multiple slaves to be stored in a DMA buffer and sent automatically.These added fields are described for bits 16 through 27 below.

Each FIFO entry holds data and associated control bits. Before data and control bits are pushed into the FIFO, the control bit settings can be modified. Halfword writes to just the control bits (offset 0xE22) doesn't push anything into the FIFO. A zero written to the upper halfword will not modify the control settings. Non-zero writes to it will modify all the control bits. Note that this is a write only register. Do not read-modify-write the register.

Byte, halfword or word writes to FIFOWR will push the data and control bits into the FIFO. Word writes with the upper halfword of zero, byte writes or halfword writes to FIFOWR will push the data and the current control bits, into the FIFO. Word writes with a non-zero upper halfword will modify the control bits before pushing them onto the stack.

Table 349. FIFO write data register (FIFOWR - offset 0xE20) bit description

Bit	Symbol	Value	Description	Reset value
15:0	TXDATA	-	Transmit data to the FIFO.	NA
16	TXSSEL0_N		Transmit Slave Select. This field asserts SSEL0 in master mode. The output on the pin is active LOW by default. Remark: The active state of the SSEL0 pin is configured by bits in the CFG register.	NA
		0	SSEL0 asserted.	
		1	SSEL0 not asserted.	
17	TXSSEL1_N		Transmit Slave Select. This field asserts SSEL1 in master mode. The output on the pin is active LOW by default. Remark: The active state of the SSEL1 pin is configured by bits in the CFG register.	NA
		0	SSEL1 asserted.	
		1	SSEL1 not asserted.	
18	TXSSEL2_N		Transmit Slave Select. This field asserts SSEL2 in master mode. The output on the pin is active LOW by default. Remark: The active state of the SSEL2 pin is configured by bits in the CFG register.	NA
		0	SSEL2 asserted.	
		1	SSEL2 not asserted.	
19	TXSSEL3_N		Transmit Slave Select. This field asserts SSEL3 in master mode. The output on the pin is active LOW by default. Remark: The active state of the SSEL3 pin is configured by bits in the CFG register.	NA
		0	SSEL3 asserted.	
		1	SSEL3 not asserted.	
20	EOT		End of Transfer. The asserted SSEL will be deasserted at the end of a transfer, and remain so for at least the time specified by the Transfer_delay value in the DLY register.	NA
		0	SSEL not deasserted. This piece of data is not treated as the end of a transfer. SSEL will not be deasserted at the end of this data.	
		1	SSEL deasserted. This piece of data is treated as the end of a transfer. SSEL will be deasserted at the end of this piece of data.	
21	EOF		End of Frame. Between frames, a delay may be inserted, as defined by the FRAME_DELAY value in the DLY register. The end of a frame may not be particularly meaningful if the FRAME_DELAY value: 0. This control can be used as part of the support for frame lengths greater than 16 bits.	NA
		0	Data not EOF. This piece of data transmitted is not treated as the end of a frame.	
		1	Data EOF. This piece of data is treated as the end of a frame, causing the FRAME_DELAY time to be inserted before subsequent data is transmitted.	
22	RXIGNORE		Receive Ignore. This allows data to be transmitted using the SPI without the need to read unneeded data from the receiver. Setting this bit simplifies the transmit process and can be used with the DMA.	NA
		0	Read received data. Received data must be read in order to allow transmission to progress. SPI transmit will halt when the receive data FIFO is full. In slave mode, an overrun error will occur if received data is not read before new data is received.	
		1	Ignore received data. Received data is ignored, allowing transmission without reading unneeded received data. No receiver flags are generated.	

Table 349. FIFO write data register (FIFOWR - offset 0xE20) bit description

Bit	Symbol	Value	Description	Reset value
23	-	-	Reserved. Read value is undefined, only zero should be written.	-
27:24	LEN		Data Length. Specifies the data length from 4 to 16 bits. Note that transfer lengths greater than 16 bits are supported by implementing multiple sequential transmits. 0x0-2: Reserved 0x3: Data transfer is 4 bits in length. 0x4: Data transfer is 5 bits in length. ... 0xF: Data transfer is 16 bits in length.	NA
31:28	-	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.15 FIFO read data register

The FIFORD register is used to read values that have been received by the FIFO.

Table 350. FIFO read data register (FIFORD - offset 0xE30) bit description

Bit	Symbol	Description	Reset value
15:0	RXDATA	Received data from the FIFO.	NA
16	RXSSEL0_N	Slave Select for receive. This field allows the state of the SSEL0 pin to be saved along with received data. The value will reflect the SSEL0 pin for both master and slave operation. A zero indicates that a slave select is active. The actual polarity of each slave select pin is configured by the related SPOL bit in CFG.	NA
17	RXSSEL1_N	Slave Select for receive. This field allows the state of the SSEL1 pin to be saved along with received data. The value will reflect the SSEL1 pin for both master and slave operation. A zero indicates that a slave select is active. The actual polarity of each slave select pin is configured by the related SPOL bit in CFG.	NA
18	RXSSEL2_N	Slave Select for receive. This field allows the state of the SSEL2 pin to be saved along with received data. The value will reflect the SSEL2 pin for both master and slave operation. A zero indicates that a slave select is active. The actual polarity of each slave select pin is configured by the related SPOL bit in CFG.	NA
19	RXSSEL3_N	Slave Select for receive. This field allows the state of the SSEL3 pin to be saved along with received data. The value will reflect the SSEL3 pin for both master and slave operation. A zero indicates that a slave select is active. The actual polarity of each slave select pin is configured by the related SPOL bit in CFG.	NA
20	SOT	Start of Transfer flag. This flag will be 1 if this is the first data after the SSELs went from deasserted to asserted (i.e., any previous transfer has ended). This information can be used to identify the first piece of data in cases where the transfer length is greater than 16 bits.	NA
31:21	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.16 FIFO data read with no FIFO pop

This register acts in exactly the same way as FIFORD, except that it supplies data from the top of the FIFO without popping the FIFO (i.e. leaving the FIFO state unchanged). This could be used to allow system software to observe incoming data without interfering with the peripheral driver.

Table 351. FIFO data read with no FIFO pop (FIFORDNOPOP - offset 0xE40) bit description

Bit	Symbol	Description	Reset value
15:0	RXDATA	Received data from the FIFO.	NA
16	RXSSEL0_N	Slave Select for receive.	NA
17	RXSSEL1_N	Slave Select for receive.	NA
18	RXSSEL2_N	Slave Select for receive.	NA
19	RXSSEL3_N	Slave Select for receive.	NA
20	SOT	Start of Transfer flag.	NA
31:21	-	Reserved. Read value is undefined, only zero should be written.	-

22.6.17 Module identification register

The ID register identifies the type and revision of the SPI module. A generic SW driver can make use of this information register to implement module type or revision specific behavior.

Table 352. Module identification register (ID - offset 0xFFC) bit description

Bit	Symbol	Description	Reset Value
7:0	APERTURE	Aperture: encoded as (aperture size/4K) -1, so 0x00 means a 4K aperture.	0x0
11:8	MINOR_REV	Minor revision of module implementation, starting at 0. Minor revision of module implementation, starting at 0. Software compatibility is expected between minor revisions.	-
15:12	MAJOR_REV	Major revision of module implementation, starting at 0. There may not be software compatibility between major revisions.	-
31:16	ID	Unique module identifier for this IP block.	0xE020

22.7 Functional description

22.7.1 AHB bus access

With the exception of the FIFOWR register, the bus interface to the SPI registers contained in the Flexcomm Interface support only word writes. Byte and halfword writes are not supported in conjunction with the SPI function for those registers.

The FIFOWR register also supports byte and halfword (data only) writes in order to allow writing FIFO data without affecting the SPI control fields above bit 15 (see Section 22.6.14).

22.7.2 Operating modes: clock and phase selection

SPI interfaces typically allow configuration of clock phase and polarity. These are sometimes referred to as numbered SPI modes, as described in Table 353 and shown in Figure 68. CPOL and CPHA are configured by bits in the CFG register (Section 22.6.1).

Table 353: SPI mode summary

CPOL	CPHA	SPI Mode	Description	SCK rest state	SCK data change edge	SCK data sample edge
0	0	0	The SPI captures serial data on the first clock transition of the transfer (when the clock changes away from the rest state). Data is changed on the following edge.	low	falling	rising
0	1	1	The SPI changes serial data on the first clock transition of the transfer (when the clock changes away from the rest state). Data is captured on the following edge.	low	rising	falling
1	0	2	Same as mode 0 with SCK inverted.	high	rising	falling
1	1	3	Same as mode 1 with SCK inverted.	high	falling	rising

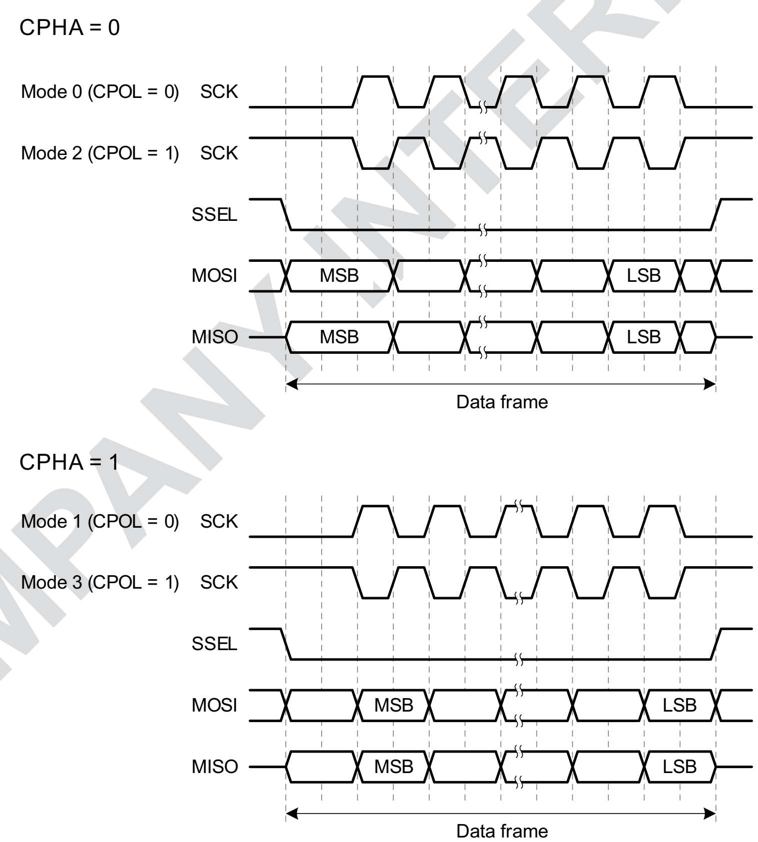


Fig 68. Basic SPI operating modes

22.7.3 Frame delays

Several delays can be specified for SPI frames. These include:

- Pre_delay: delay after SSEL is asserted before data clocking begins
- Post_delay: delay at the end of a data frame before SSEL is deasserted
- Frame_delay: delay between data frames when SSEL is not deasserted
- Transfer_delay: minimum duration of SSEL in the deasserted state between transfers

22.7.3.1 Pre_delay and Post_delay

Pre_delay and Post_delay are illustrated by the examples in Figure 69. The Pre_delay value controls the amount of time between SSEL being asserted and the beginning of the subsequent data frame. The Post_delay value controls the amount of time between the end of a data frame and the deassertion of SSEL.

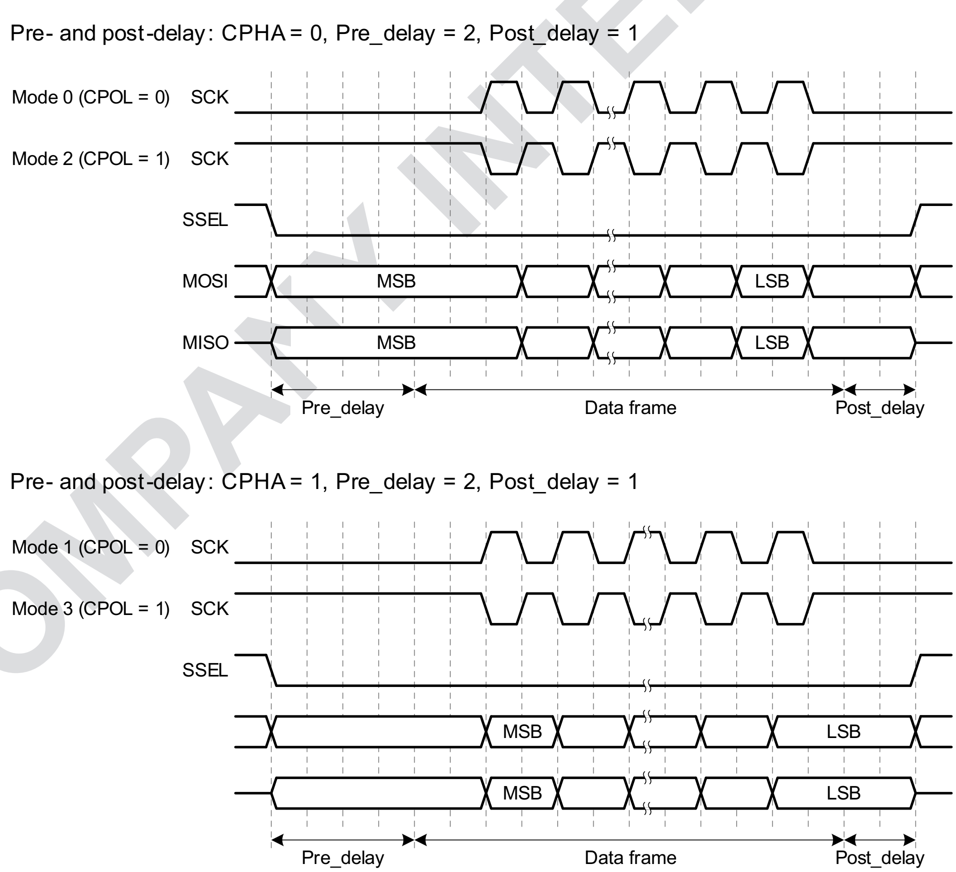


Fig 69. Pre_delay and Post_delay

22.7.3.2 Frame_delay

The Frame_delay value controls the amount of time at the end of each frame. This delay is inserted when the EOF bit: 1. Frame_delay is illustrated by the examples in Figure 70. Note that frame boundaries occur only where specified. This is because frame lengths can be any size, involving multiple data writes. See Section 22.7.7 for more information.

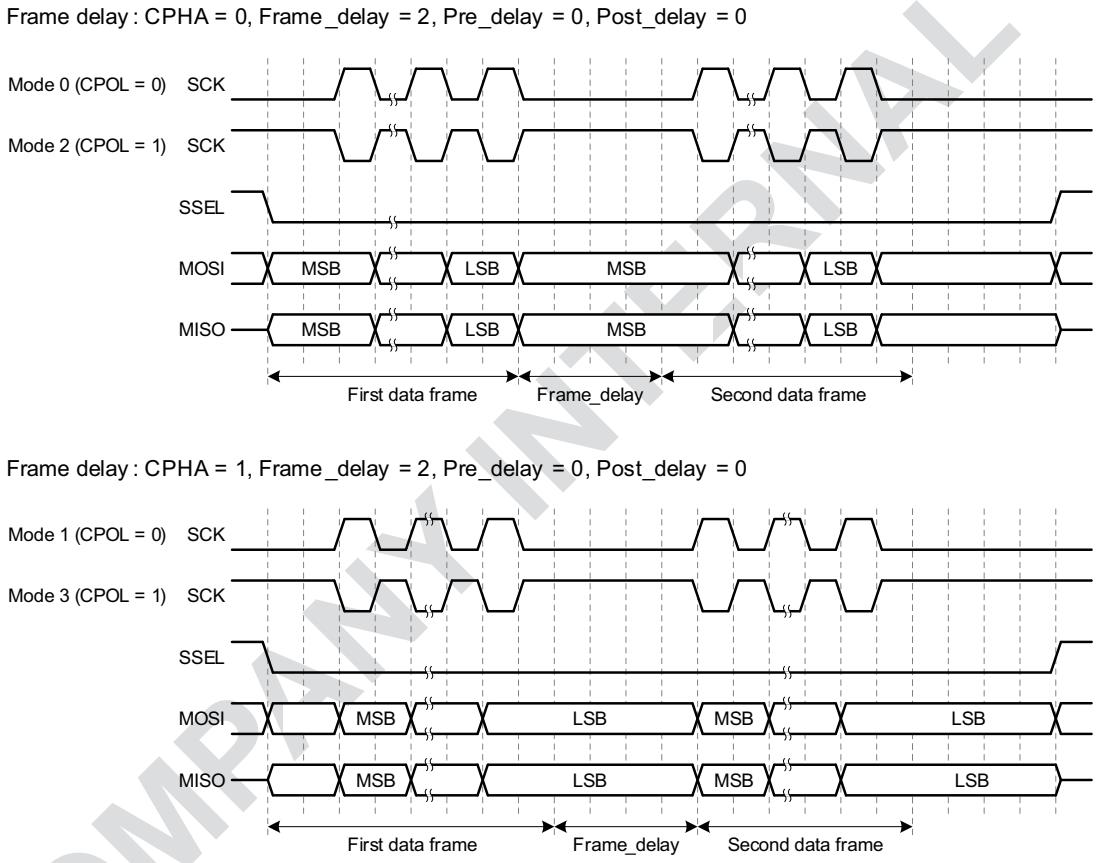


Fig 70. Frame_delay

22.7.3.3 Transfer_delay

The Transfer_delay value controls the minimum amount of time that SSEL is deasserted between transfers, because the EOT bit: 1. When Transfer_delay: 0, SSEL may be deasserted for a minimum of one SPI clock time. Transfer_delay is illustrated by the examples in Figure 71.

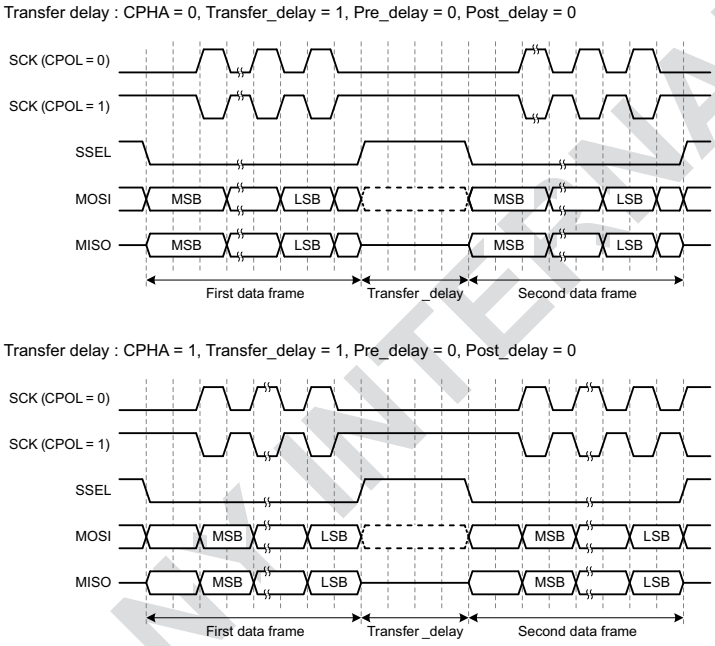


Fig 71. Transfer_delay

22.7.4 Clocking and data rates

In order to use the SPI, clocking details must be defined. This includes configuring the system clock and selection of the clock divider value in DIV. See Figure 9.

22.7.4.1 Data rate calculations

The SPI interface is designed to operate asynchronously from any on-chip clocks, and without the need for overclocking.

In slave mode, this means that the SCK from the external master is used directly to run the transmit and receive shift registers and other logic.

In master mode, the SPI rate clock produced by the SPI clock divider is used directly as the outgoing SCK.

The SPI clock divider is an integer divider. The SPI in master mode can be set to run at the same speed as the selected FCLK, or at lower integer divide rates. The SPI rate will be: FCLK of Flexcomm Interface n / DIVVAL.

In slave mode, the clock is taken from the SCK input and the SPI clock divider is not used.

22.7.5 Slave select

The SPI block provides for four Slave Select inputs in slave mode or outputs in master mode. Each SSEL can be set for normal polarity (active low), or can be inverted (active high). Representation of the 4 SSELs in a register is always active low. If an SSEL is inverted, this is done as the signal leaves/enters the SPI block.

In slave mode, **any** asserted SSEL that is connected to a pin will activate the SPI. In master mode, all SSELs that are connected to a pin will be output as defined in the SPI registers. In the latter case, the SSELs could potentially be decoded externally in order to address more than four slave devices. Note that at least one SSEL is asserted when data is transferred in master mode.

In master mode, Slave Selects come from the TXSSEL bits in the FIFOWR register. In slave mode, the state of all four SSELs is saved along with received data in the RXSSEL_N field of the FIFORD register.

Remark: In slave mode, unused SSELs should not be configured as SPI pin function in Pin mux. Related SPOLx bits in CFG register (see Table 336) should be set as Table 354 to ensure the unused SSELs not to activate the SPI. For the used SSELs pin, its related SPOLx bit should be configured as the master's requirement. If the master requires active low, configure SPOLx to 0, if the master requires active high, configure it to 1.

Table 354. SPIOLx configuration for used SSELs

Flexcomm	used SSEL pin	SPOLx bit
Flexcomm2	SSEL0	0
	SSEL1	0
	SSEL2	1
	SSEL3	1

Table 354. SPIOLx configuration for used SSELs

Flexcomm	used SSEL pin	SPOLx bit
Flexcomm3	SSEL0	1
	SSEL1	1
	SSEL2	1
	SSEL3	1

22.7.6 DMA operation

A DMA request is provided for each SPI direction, and can be used in lieu of interrupts for transferring data by configuring the DMA controller appropriately. The DMA controller provides an acknowledgement signal that clears the related request when it completes handling that request.

The transmitter DMA request is asserted when Tx DMA is enabled and the transmitter can accept more data.

The receiver DMA request is asserted when Rx DMA is enabled and received data is available to be read.

22.7.6.1 DMA master mode End-Of-Transfer

When using polled or interrupt mode to transfer data in master mode, the transition to end-of-transfer status (drive SSEL inactive) is straightforward. The EOT bit of the FIFOWR control bits would be set just before or along with the writing of the last data to be sent.

When using the DMA in master mode, the end-of-transfer status (drive SSEL inactive) can be generated in a number of ways:

1. Using DMA interrupt and a second DMA transfer:

   To use only 8 or 16 bit wide DMA transfers for all the data, a second DMA transfer can be used to terminate the transfer (drive SSEL inactive).

   The transfer would be started by setting the control bits and then initiating the DMA transfer of all but the last byte/halfword of data. The DMA completion interrupt function must modify the control bits to set EOT and then set-up DMA to send the last data.

2. Using DMA and SPI interrupts (or background SPI status polling):

   To use only one 8 or 16 bit wide DMA transfer for all the data, two interrupts would be required to properly terminate the transfer (drive SSEL inactive).

   The SPI Tx DMA completion interrupt function sets the TXLVL field in the SPI FIFOTRIG register to 0 and sets the TXLVL interrupt enable bit in the FIFOINTENSET register.

   The interrupt function handling the SPI TXLVL would set the SPI STAT register "END TRANSFER" bit, to force termination after all data output is complete.

3. Using DMA linked descriptor:

   The DMA controller provides for a linked list of DMA transfer control descriptors. The initial descriptor(s) can be used to transfer all but the last data byte/halfword. These data transfers can be done as 8 or 16 bit wide DMA operations. A final DMA

descriptor, linked to the first DMA descriptor, can be used to send the last data along with control bits to the FIFOWR register. The control bits would include the setting of the EOT bit.

Note: The DMA interrupt function cannot set the SPI Status register (STAT) END TRANSFER control bit. This may terminate the transfer while the FIFO still has data to send.

4. Using 32 bit wide DMA:

   Write both data and control bits to FIFOWR for all data. The control bits for the last entry would include the setting of the EOT bit. This also allows a series of SPI transactions involving multiple slaves with one DMA operation, by changing the TXSSELn_N bits.

22.7.7 Data lengths greater than 16 bits

The SPI interface handles data frame sizes from 4 to 16 bits directly. Larger sizes can be handled by splitting data up into groups of 16 bits or less. For example, 24 bits can be supported as 2 groups of 16 bits and 8 bits or 2 groups of 12 bits, among others. Frames of any size, including greater than 32 bits, can supported in the same way.

Details of how to handle larger data widths depend somewhat on other SPI configuration options. For instance, if it is intended for Slave Selects to be deasserted between frames, then this must be suppressed when a larger frame is split into more than one part. Sending 2 groups of 12 bits with SSEL deasserted between 24-bit increments, for instance, would require changing the value of the EOF bit on alternate 12-bit frames.

22.7.8 Data stalls

A stall for Master transmit data can happen in modes 0 and 2 when SCK cannot be returned to the rest state until the MSB of the next data frame can be driven on MOSI. In this case, the stall happens just before the final clock edge of data if the next piece of data is not yet available.

A stall for Master receive can happen when a receiver overrun would otherwise occur if the transmitter was not stalled. In modes 0 and 2, this occurs if the previously received data is not read before the end of the next piece of data is received. This stall happens one clock edge earlier than the transmitter stall.

In modes 1 and 3, the same kind of receiver stall can occur, but just before the final clock edge of the received data. Also, a transmitter stall will not happen in modes 1 and 3 because the transmitted data is complete at the point where a stall would otherwise occur, so it is not needed.

Stalls are reflected in the STAT register by the Stalled status flag, which indicates the current SPI status. The transmitter will be stalled until data is read from the receive FIFO. Use the RXIGNORE control bit setting to avoid the need to read the received data.

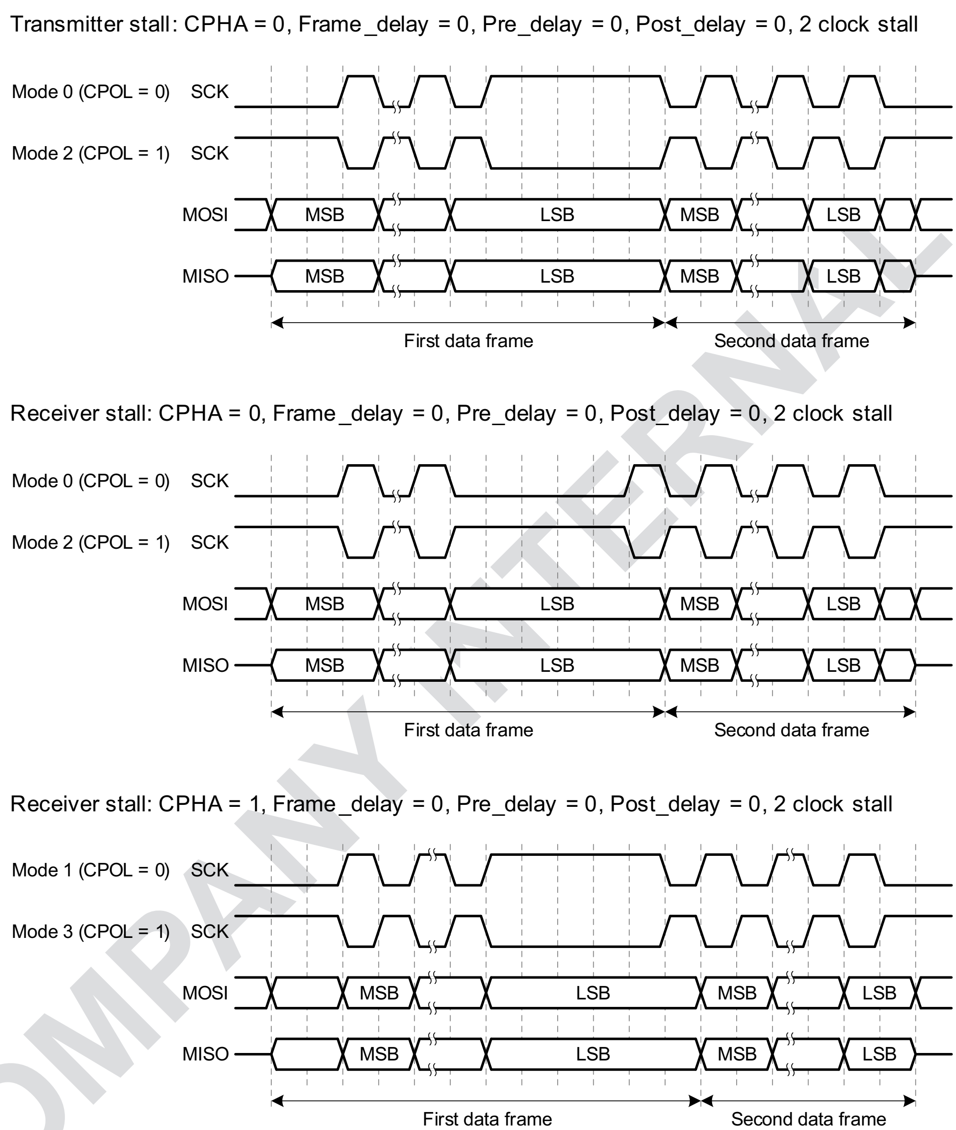


Fig 72. Examples of data stalls

23. I²C-bus interfaces

23.1 Introduction

I²C-bus functions are available on all QN908x devices as a selectable function in Flexcomm1 and Flexcomm2 peripheral.

23.2 Features

- Independent Master, slave, and monitor functions.
- Bus speeds supported up to 400 kbits/s
- Supports both multi-master and multi-master with slave functions.
- Multiple I²C slave addresses supported in hardware.
- One slave address can be selectively qualified with a bit mask or an address range in order to respond to multiple I²C bus addresses.
- 10-bit addressing supported with software assist.
- Separate DMA requests for master, slave, and monitor functions.
- No chip clocks are required in order to receive and compare an address as a slave, so this event can wake up the device from sleep mode.
- Automatic modes optionally allow less software overhead for some use cases.

23.3 Pin description

The I2C signals are movable Flexcomm functions and are assigned to external pins via PIO_FUNC_CFG<x> registers. The pins that are assigned to should be configured to enable pull-up resistor via PIO_PULL_CFG<x> registers. An example of PIO_PULL_CFG<x> settings are shown in Table 353.

Table 355. I²C-bus pin description

Function	Type	Pin name used in data sheet Pin Description	Description
SCL	I/O	FCn_TXD_SCL_MISO, FCn_RTS_SCL_SSEL1	I²C serial clock.
SDA	I/O	FCn_RXD_SDA_MOSI, FCn_CTS_SDA_SSEL0	I²C serial data.

Table 356: USART pin configuration example

Pin	Type	Name used in pin configuration chapter	PAD name	PINMUX register and configure value
SCL	I/O	FC1_RTS_SCL_SSEL1	PA29	PIO_FUNC_CFG0[26:24]: 0x4 (FUNC4)
SDA	I/O	FC1_CTS_SDA_SSEL0	PA26	PIO_FUNC_CFG0[30:28]: 0x4 (FUNC4)

23.4 Basic configuration

Configure the I²C and related clocks as follows:

- If needed, configure RST_SW_SET and RST_SW_CLR registers to reset the Flexcomm that is about to have a specific peripheral function selected.

- Configure the related Flexcomm pin functions via IOCON, see Section 8.
- Configure the I^{2}C for the desired functions:
  - Configure CLK_EN register to enable clock for the related Flexcomm.
  - Select the desired Flexcomm function by writing to the PSELID register of the related Flexcomm (Section 20.6.2).
  - Enable or disable the related Flexcomm Interface interrupt in the NVIC (see Table 49).
  - Configure the I^{2}C clock and data rate. This includes the CLKDIV register for both master and slave modes, and MSTTIME for master mode. Also see Section 23.6.6 and Section 23.7.2.

Remark: while the I^{2}C function is incorporated into the Flexcomm Interface, it does not make use of the Flexcomm Interface FIFO.

23.4.1 I^{2}C transmit/receive in master mode

In this example, Flexcomm Interface 1 is configured as an I^{2}C master. The master sends 8 bits to the slave and then receives 8 bits from the slave.

The transmission of the address and data bits is controlled by the state of the MSTPENDING status bit. Whenever the status is Master pending, the master can read or write to the MSTDAT register and go to the next step of the transmission protocol by writing to the MSTCTL register.

Configure the I^{2}C bit rate:

- Select a source for the Flexcomm Interface 1 clock that will allow for the desired I^{2}C-bus rate. Divide the clock as needed, see Table 369.
- Further divide the source clock if needed using the CLKDIV register (Section 23.6.6).
- Set the SCL high and low times to complete the bus rate setup. See Section 23.6.9.

23.4.1.1 Master write to slave

Configure Flexcomm Interface 1 as I^{2}C interface, see Section 20.

Configure the I^{2}C as a master: set the MSTEN bit to 1 in the CFG register. See Table 362.

Write data to the slave:

- Write the slave address with the $\overline{RW}$ bit set to 0 to the Master data register MSTDAT. See Table 373.
- Start the transmission by setting the MSTSTART bit to 1 in the Master control register. See Table 371. The following happens:
  - The pending status is cleared and the I^{2}C-bus is busy.
  - The I^{2}C master sends the start bit and address with the $\overline{RW}$ bit to the slave.
- Wait for the pending status to be set (MSTPENDING = 1) by polling the STAT register.
- Write 8 bits of data to the MSTDAT register.
- Continue with the transmission of data by setting the MSTCONT bit to 1 in the Master control register. See Table 371. The following happens:
  - The pending status is cleared and the I^{2}C-bus is busy.

- The I²C master sends the data bits to the slave address.

- Wait for the pending status to be set (MSTPENDING = 1) by polling the STAT register.
- Stop the transmission by setting the MSTSTOP bit to 1 in the Master control register. See Table 371.

Table 357. Code Example

Master write to slave:

```
//Master write 1 byte to slave. Address 0x23, Data 0xdd. Polling mode.
I2C->CFG = I2C_CFG_MSTEN;
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_IDLE) abort();
I2C->MSTDAT = (0x23 << 1) | 0; // address and 0 for RWn bit
I2C->MSTCTL = 2C_MSTCTL_MSTSTART; // send start
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_TX) abort();
I2C->MSTDAT = 0xdd; // send data
I2C->MSTCTL = I2C_MSTCTL_MSTCONTINUE; // continue transaction
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_TX) abort();
I2C->MSTCTL = I2C_MSTCTL_MSTSTOP; // send stop
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_IDLE) abort();
```

23.4.1.2 Master read from slave

Configure Flexcomm Interface 1 as I²C interface, see Section 20.

Configure the I²C as a master: set the MSTEN bit to 1 in the CFG register. See Table 362.

Read data from the slave:

1. Write the slave address with the $\overline{RW}$ bit set to 1 to the Master data register MSTDAT. See Table 373.
2. Start the transmission by setting the MSTSTART bit to 1 in the Master control register. See Table 371. The following happens:
   - The pending status is cleared and the I²C-bus is busy.
   - The I²C master sends the start bit and address with the $\overline{RW}$ bit to the slave.
   - The slave sends 8 bit of data.
3. Wait for the pending status to be set (MSTPENDING = 1) by polling the STAT register.
4. Read 8 bits of data from the MSTDAT register.
5. Stop the transmission by setting the MSTSTOP bit to 1 in the Master control register. See Table 371.

Table 358. Code Example

Master read from slave

```
// Master read 1 byte from slave. Address 0x23. Polling mode. No error checking.
uint8_t data;
I2C->CFG = I2C_CFG_MSTEN;
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_IDLE) abort();
I2C->MSTDAT = (0x23 << 1) | 1; // address and 1 for RWn bit
I2C->MSTCTL = I2C_MSTCTL_MSTSTART; // send start
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_RX) abort();
data=I2C->MSTDAT; // read data
if(data != 0xdd) abort();
I2C->MSTCTL = I2C_MSTCTL_MSTSTOP; // send stop
while(!(I2C->STAT & I2C_STAT_MSTPENDING));
if((I2C->STAT & I2C_STAT_MSTSTATE) != I2C_STAT_MSTST_IDLE) abort();
```


23.4.2 I2C receive/transmit in slave mode

In this example, Flexcomm Interface 1 is configured as an I2C slave. The slave receives 8 bits from the master and then sends eight bits to the master.

The transmission of the address and data bits is controlled by the state of the SLVPENDING status bit. Whenever the status is Slave pending, the slave can acknowledge (“ack”) or send or receive an address and data. The received data or the data to be sent to the master are available in the SLVDAT register. After sending and receiving data, continue to the next step of the transmission protocol by writing to the SLVCTL register.

23.4.2.1 Slave read from master

Configure Flexcomm Interface 1 as I2C interface, see Section 20.

Configure the I2C as a slave with address x:

- Set the SLVEN bit to 1 in the CFG register. See Table 362.
- Write the slave address x to the address 0 match register. See Table 376.

Read data from the master:

1. Wait for the pending status to be set (SLVPENDING = 1) by polling the STAT register.
2. Acknowledge (“ack”) the address by setting SLVCONTINUE =1 in the slave control register. See Table 374.
3. Wait for the pending status to be set (SLVPENDING = 1) by polling the STAT register.
4. Read 8 bits of data from the SLVDAT register. See Table 375.
5. Acknowledge (“ack”) the data by setting SLVCONTINUE: 1 in the slave control register. See Table 374.

Table 359. Code Example

Slave read from Master

```
//Slave read 1 byte from master. Address 0x23. Polling mode.
uint8_t data;
I2C->SLVADR0 = 0x23 << 1; // put address in address 0 register
I2C->CFG = I2C_CFG_SLVEN;
I2C->CFG;
while(!(I2C->STAT & I2C_STAT_SLVPENDING));
if((I2C->STAT & I2C_STAT_SLVSTATE) != I2C_STAT_SLVST_ADDR) abort();
I2C->SLVCTL = I2C_SLVCTL_SLVCONTINUE; // ack address
while(!(I2C->STAT & I2C_STAT_SLVPENDING));
if((I2C->STAT & I2C_STAT_SLVSTATE) != I2C_STAT_SLVST_RX) abort();
data: I2C->SLVDAT; // read data
if(data != 0xdd) abort();
I2C->SLVCTL = I2C_SLVCTL_SLVCONTINUE; // ack data
```

23.4.2.2 Slave write to master

Configure Flexcomm Interface 1 as I^2C interface, see Section 20.

Configure the I^2C as a slave with address x:

- Set the SLVEN bit to 1 in the CFG register. See Table 362.
- Write the slave address x to the address 0 match register. See Table 376.

Write data to the master:

1. Wait for the pending status to be set (SLVPENDING: 1) by polling the STAT register.
2. ACK the address by setting SLVCONTINUE: 1 in the slave control register. See Table 374.
3. Wait for the pending status to be set (SLVPENDING: 1) by polling the STAT register.
4. Write 8 bits of data to SLVDAT register. See Table 375.
5. Continue the transaction by setting SLVCONTINUE: 1 in the slave control register. See Table 374.

Table 360. Code Example

Slave write to master

```
//Slave write 1 byte to master. Address 0x23, Data 0xdd. Polling mode.
I2C->SLVADR0: 0x23 << 1; // put address in address 0 register
I2C->CFG = I2C_CFG_SLVEN;
I2C->CFG;
while(!(I2C->STAT & I2C_STAT_SLVPENDING));
if((I2C->STAT & I2C_STAT_SLVSTATE) != I2C_STAT_SLVST_ADDR) abort();
I2C->SLVCTL =I2C_SLVCTL_SLVCONTINUE; // ack address
while(!(I2C->STAT & I2C_STAT_SLVPENDING));
if((I2C->STAT & I2C_STAT_SLVSTATE) != I2C_STAT_SLVST_TX) abort();
I2C->SLVDAT = 0xdd; // write data
```

23.4.3 Configure the I^2C for wake-up

In sleep mode, any activity on the I^2C-bus that triggers an I^2C interrupt can wake up the part, provided that the interrupt is enabled in the INTENSET register and the NVIC. As long as the Flexcomm Interface clock remains active in sleep mode, the I^2C can wake up the part independently of whether the I^2C interface is configured in master or slave mode.

To wak up from sleep mode:

- Enable the I^2C interrupt in the NVIC.
- Enable the I^2C wake-up event in the INTENSET register. Wake-up on any enabled interrupts is supported (see the INTENSET register). Examples are the following events:
  - Master pending
  - Change to idle state
  - Start/stop error
  - Slave pending
  - Address match (in slave mode)
  - Data available/ready

23.5 General description

The architecture of the I²C-bus interface is shown in Figure 73.

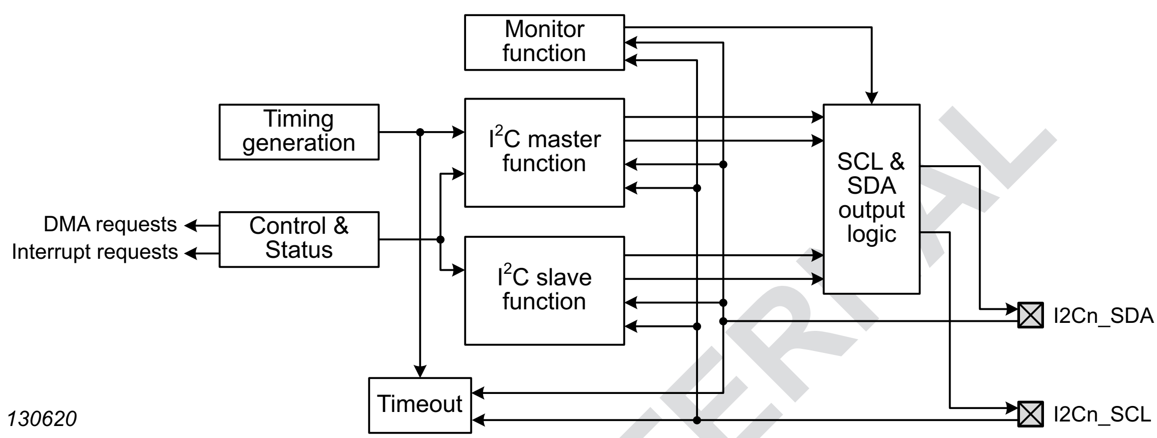


Fig 73. I²C block diagram

23.6 Register description

Address offsets are within the address space of the related Flexcomm Interface. The Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

Table 361. I²C register overview

Name	Access	Offset	Description	Reset value	Section
Shared I²C registers:					
CFG	R/W	0x800	Configuration for shared functions.	0x0	23.6.1
STAT	R/W	0x804	Status register for Master, Slave, and Monitor functions.	0x0801	23.6.2
INTENSET	R/W	0x808	Interrupt Enable Set and read register.	0x0	23.6.3
INTENCLR	WO	0x80C	Interrupt Enable Clear register.	NA	23.6.4
TIMEOUT	R/W	0x810	Time-out value register.	0xFFFF	23.6.5
CLKDIV	R/W	0x814	Clock pre-divider for the entire I²C interface. This determines what time increments are used for the MSTTIME register, and controls some timing of the Slave function.	0x0	23.6.6
INTSTAT	RO	0x818	Interrupt Status register for Master, Slave, and Monitor functions.	0x0	23.6.7
Master function registers:					
MSTCTL	R/W	0x820	Master control register.	0x0	23.6.8
MSTTIME	R/W	0x824	Master timing configuration.	0x77	23.6.9
MSTDAT	R/W	0x828	Combined Master receiver and transmitter data register.	NA	23.6.10
Slave function registers:					
SLVCTL	R/W	0x840	Slave control register.	0x0	23.6.11
SLVDAT	R/W	0x844	Combined Slave receiver and transmitter data register.	NA	23.6.12
SLVADR0	R/W	0x848	Slave address 0.	0x01	23.6.13
SLVADR1	R/W	0x84C	Slave address 1.	0x01	23.6.14
SLVADR2	R/W	0x850	Slave address 2.	0x01	23.6.14

Table 361. I^{2}C register overview

Name	Access	Offset	Description	Reset value	Section
SLVADR3	R/W	0x854	Slave address 3.	0x01	23.6.14
SLVQUAL0	R/W	0x858	Slave Qualification for address 0.	0x0	23.6.15
Monitor function registers:					
MONRXDAT	RO	0x880	Monitor receiver data register.	0x0	23.6.16
ID register:					
ID	RO	0xFFC	I2C module Identification. This value appears in the shared Flexcomm Interface peripheral ID register when I2C is selected.	see table	23.6.17

23.6.1 I2C Configuration register

The CFG register contains mode settings that apply to master, slave, and monitor functions.

Table 362. I^{2}C Configuration register (CFG, offset 0x800) bit description

Bit	Symbol	Value	Description	Reset Value
0	MSTEN		Master Enable. When disabled, configurations settings for the Master function are not changed, but the Master function is internally reset.	0x0
		0	Disabled. The I^{2}C Master function is disabled.	
		1	Enabled. The I^{2}C Master function is enabled.	
1	SLVEN		Slave Enable. When disabled, configurations settings for the Slave function are not changed, but the Slave function is internally reset.	0x0
		0	Disabled. The I^{2}C slave function is disabled.	
		1	Enabled. The I^{2}C slave function is enabled.	
2	MONEN		Monitor Enable. When disabled, configurations settings for the Monitor function are not changed, but the Monitor function is internally reset.	0x0
		0	Disabled. The I^{2}C Monitor function is disabled.	
		1	Enabled. The I^{2}C Monitor function is enabled.	
3	TIMEOUTEN		I^{2}C bus Time-out Enable. When disabled, the time-out function is internally reset.	0x0
		0	Disabled. Time-out function is disabled.	
		1	Enabled. Time-out function is enabled. Both types of time-out flags will be generated and will cause interrupts if they are enabled. Typically, only one time-out will be used in a system.	
4	MONCLKSTR		Monitor function Clock Stretching.	0x0
		0	Disabled. The Monitor function will not perform clock stretching. Software or DMA may not always be able to read data provided by the Monitor function before it is overwritten. This mode may be used when non-invasive monitoring is critical.	
		1	Enabled. The Monitor function will perform clock stretching in order to ensure that software or DMA can read all incoming data supplied by the Monitor function.	
31:5	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.2 I2C Status register

The STAT register provides status flags and state information about all of the functions of the I2C interface. Access to bits in this register varies. RO: Read-only, W1C: write 1 to clear.

Details of the master and slave states described in the MSTSTATE and SLVSTATE bits in this register are listed in Table 364 and Table 365.

Table 363. I2C Status register (STAT, offset 0x804) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	MST PENDING		Master Pending. Indicates that the Master is waiting to continue communication on the I2C-bus (pending) or is idle. When the master is pending, the MSTSTATE bits indicate what type of software service if any the master expects. This flag will cause an interrupt when set if, enabled via the INTENSET register. The MSTPENDING flag is not set when the DMA is handling an event (if the MSTDMA bit in the MSTCTL register is set). If the master is in the idle state, and no communication is needed, mask this interrupt.	0x1	RO
		0	In progress. Communication is in progress and the Master function is busy and cannot currently accept a command.		
		1	Pending. The Master function needs software service or is in the idle state. If the master is not in the idle state, it is waiting to receive or transmit data or the NACK bit.		
3:1	MSTSTATE		Master State code. The master state code reflects the master state when the MSTPENDING bit is set, that is the master is pending or in the idle state. Each value of this field indicates a specific required service for the Master function. All other values are reserved. See Table 364 for details of state values and appropriate responses.	0x0	RO
		0x0	Idle. The Master function is available to be used for a new transaction.		
		0x1	Receive ready. Received data available (Master Receiver mode). Address plus Read was previously sent and Acknowledged by slave.		
		0x2	Transmit ready. Data can be transmitted (Master Transmitter mode). Address plus Write was previously sent and Acknowledged by slave.		
		0x3	NACK Address. Slave NACKed address.		
		0x4	NACK Data. Slave NACKed transmitted data.		
4	MST ARBLOSS		Master Arbitration Loss flag. This flag can be cleared by software writing a 1 to this bit. It is also cleared automatically when a 1 is written to MSTCONTINUE.	0x0	W1C
		0	No Arbitration Loss has occurred.		
		1	Arbitration loss. The Master function has experienced an Arbitration Loss. At this point, the Master function has already stopped driving the bus and gone to an idle state. Software can respond by doing nothing, or by sending a Start in order to attempt to gain control of the bus when it next becomes idle.		
5	-	-	Reserved. Read value is undefined, only zero should be written.	-	-

Table 363. I^{2}C Status register (STAT, offset 0x804) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
6	MST STSTPERR		Master Start/Stop Error flag. This flag can be cleared by software writing a 1 to this bit. It is also cleared automatically when a 1 is written to MSTCONTINUE.	0x0	W1C
		0	No Start/Stop Error has occurred.		
		1	The Master function has experienced a Start/Stop Error. A Start or Stop was detected at a time when it is not allowed by the I^{2}C specification. The Master interface has stopped driving the bus and gone to an idle state, no action is required. A request for a Start could be made, or software could attempt to insure that the bus has not stalled.		
7	-	-	Reserved. Read value is undefined, only zero should be written.	-	-
8	SLV PENDING		Slave Pending. Indicates that the Slave function is waiting to continue communication on the I^{2}C-bus and needs software service. This flag will cause an interrupt when set if enabled via INTENSET. The SLVPENDING flag is not set when the DMA is handling an event (if the SLVDMA bit in the SLVCTL register is set). The SLVPENDING flag is read-only and is automatically cleared when a 1 is written to the SLVCONTINUE bit in the SLVCTL register. The point in time when SlvPending is set depends on whether the I^{2}C interface is in HSCAPABLE mode. See "Clock stretching". When the I^{2}C interface is configured to be HSCAPABLE, HS master codes are detected automatically. Due to the requirements of the HS I^{2}C specification, slave addresses must also be detected automatically, since the address must be acknowledged before the clock can be stretched.	0x0	RO
		0	In progress. The Slave function does not currently need service.		
		1	Pending. The Slave function needs service. Information on what is needed can be found in the adjacent SLVSTATE field.		
10:9	SLVSTATE		Slave State code. Each value of this field indicates a specific required service for the Slave function. All other values are reserved. See Table 365 for state values and actions. **Remark:** note that the occurrence of some states and how they are handled are affected by DMA mode and Automatic Operation modes.	0x0	RO
		0x0	Slave address. Address plus R/W received. At least one of the four slave addresses has been matched by hardware.		
		0x1	Slave receive. Received data is available (Slave Receiver mode).		
		0x2	Slave transmit. Data can be transmitted (Slave Transmitter mode).		
11	SLV NOTSTR		Slave Not Stretching. Indicates when the slave function is stretching the I^{2}C clock. This is needed in order to gracefully invoke deep-sleep mode during slave operation. This read-only flag reflects the slave function status in real time.	0x1	RO
		0	Stretching. The slave function is currently stretching the I^{2}C bus clock. deep-sleep mode cannot be entered at this time.		
		1	Not stretching. The slave function is not currently stretching the I^{2}C bus clock. deep-sleep mode could be entered at this time.		

Table 363. I²C Status register (STAT, offset 0x804) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
13:12	SLVIDX		Slave address match Index. This field is valid when the I²C slave function has been selected by receiving an address that matches one of the slave addresses defined by any enabled slave address registers, and provides an identification of the address that was matched. It is possible that more than one address could be matched, but only one match can be reported here.	0x0	RO
		0x0	Address 0. Slave address 0 was matched.		
		0x1	Address 1. Slave address 1 was matched.		
		0x2	Address 2. Slave address 2 was matched.		
		0x3	Address 3. Slave address 3 was matched.		
14	SLVSEL		Slave selected flag. SLVSEL is set after an address match when software tells the Slave function to acknowledge the address, or when the address has been automatically acknowledged. It is cleared when another address cycle presents an address that does not match an enabled address on the Slave function, when slave software decides to NACK a matched address, when there is a Stop detected on the bus, when the master NACKs slave data, and in some combinations of Automatic Operation. SLVSEL is not cleared if software NACKs data.	0x0	RO
		0	Not selected. The Slave function is not currently selected.		
		1	Selected. The Slave function is currently selected.		
15	SLVDESEL		Slave Deselected flag. This flag will cause an interrupt when set if enabled via INTENSET. This flag can be cleared by writing a 1 to this bit.	0x0	W1C
		0	Not deselected. The Slave function has not become deselected. This does not mean that it is currently selected. That information can be found in the SLVSEL flag.		
		1	Deselected. The Slave function has become deselected. This is specifically caused by the SLVSEL flag changing from 1 to 0. See the description of SLVSEL for details on when that event occurs.		
16	MONRDY		Monitor Ready. This flag is cleared when the MONRXDAT register is read.	0x0	RO
		0	No data. The Monitor function does not currently have data available.		
		1	Data waiting. The Monitor function has data waiting to be read.		
17	MONOV		Monitor Overflow flag.	0x0	W1C
		0	No overrun. Monitor data has not overrun.		
		1	Overrun. A Monitor data overrun has occurred. This can only happen when Monitor clock stretching not enabled via the MONCLKSTR bit in the CFG register. Writing 1 to this bit clears the flag.		
18	MON ACTIVE		Monitor Active flag. Indicates when the Monitor function considers the I²C bus to be active. Active is defined here as when some Master is on the bus: a bus Start has occurred more recently than a bus Stop.	0x0	RO
		0	Inactive. The Monitor function considers the I²C bus to be inactive.		
		1	Active. The Monitor function considers the I²C bus to be active.		

Table 363. I^{2}C Status register (STAT, offset 0x804) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
19	MONIDLE		Monitor Idle flag. This flag is set when the Monitor function sees the I^{2}C bus change from active to inactive. This can be used by software to decide when to process data accumulated by the Monitor function. This flag will cause an interrupt when set if enabled via the INTENSET register. The flag can be cleared by writing a 1 to this bit.	0x0	W1C
		0	Not idle. The I^{2}C bus is not idle, or this flag has been cleared by software.		
		1	Idle. The I^{2}C bus has gone idle at least once since the last time this flag was cleared by software.		
23:20	-	-	Reserved. Read value is undefined, only zero should be written.	-	-
24	EVENT TIMEOUT		Event Time-out Interrupt flag. Indicates when the time between events has been longer than the time specified by the TIMEOUT register. Events include Start, Stop, and clock edges. The flag is cleared by writing a 1 to this bit. No time-out is created when the I^{2}C-bus is idle.	0x0	W1C
		0	No time-out. I^{2}C bus events have not caused a time-out.		
		1	Event time-out. The time between I^{2}C bus events has been longer than the time specified by the TIMEOUT register.		
25	SCL TIMEOUT		SCL Time-out Interrupt flag. Indicates when SCL has remained low longer than the time specific by the TIMEOUT register. The flag is cleared by writing a 1 to this bit.	0x0	W1C
		0	No time-out. SCL low time has not caused a time-out.		
		1	Time-out. SCL low time has caused a time-out.		
31:26	-	-	Reserved. Read value is undefined, only zero should be written.	-	-

Table 364. Master function state codes (MSTSTATE)

MST STATE	Description	Actions	DMA allowed
0x0	**Idle.** The Master function is available to be used for a new transaction.	Send a Start or disable MSTPENDING interrupt if the Master function is not needed currently.	No
0x1	**Received data is available (Master Receiver mode).** Address plus Read was previously sent and Acknowledged by slave.	Read data and either continue, send a Stop, or send a Repeated Start.	Yes
0x2	**Data can be transmitted (Master Transmitter mode).** Address plus Write was previously sent and Acknowledged by slave.	Send data and continue, or send a Stop or Repeated Start.	Yes
0x3	**Slave NACKed address.**	Send a Stop or Repeated Start.	No
0x4	**Slave NACKed transmitted data.**	Send a Stop or Repeated Start.	No

Table 365. Slave function state codes (SLVSTATE)

SLVSTATE		Description	Actions	DMA allowed
0	SLVST_ ADDR	**Address plus R/W received.** At least one of the 4 slave addresses has been matched by hardware.	Software can further check the address if needed, for instance if a subset of addresses qualified by SLVQUAL0 is to be used. Software can ACK or NACK the address by writing 1 to either SLVCONTINUE or SLVNACK. Also see Section 23.7.4 regarding 10-bit addressing.	No
1	SLVST_RX	**Received data is available (Slave Receiver mode).**	Read data, reply with an ACK or a NACK.	Yes
2	SLVST_TX	**Data can be transmitted (Slave Transmitter mode).**	Send data. Note that when the Master NACKs dat transmitted by the slave, the slave becomes de-selected.	Yes

23.6.3 Interrupt Enable Set and read register

The INTENSET register controls which I^{2}C status flags generate interrupts. Writing a 1 to a bit position in this register enables an interrupt in the corresponding position in the STAT register (Table 363), if an interrupt is supported there. Reading INTENSET indicates which interrupts are currently enabled.

Table 366. Interrupt Enable Set and read register (INTENSET, offset 0x808) bit description

Bit	Symbol	Value	Description	Reset value
0	MSTPENDINGEN		Master Pending interrupt Enable.	0x0
		0	Disabled. The MstPending interrupt is disabled.	
		1	Enabled. The MstPending interrupt is enabled.	
3:1	-	-	Reserved. Read value is undefined, only zero should be written.	-
4	MSTARBLOSSEN		Master Arbitration Loss interrupt Enable.	0x0
		0	Disabled. The MstArbLoss interrupt is disabled.	
		1	Enabled. The MstArbLoss interrupt is enabled.	
5	-	-	Reserved. Read value is undefined, only zero should be written.	-
6	MSTSTSTPERREN		Master Start/Stop Error interrupt Enable.	0x0
		0	Disabled. The MstStStpErr interrupt is disabled.	
		1	Enabled. The MstStStpErr interrupt is enabled.	
7	-	-	Reserved. Read value is undefined, only zero should be written.	-
8	SLVPENDINGEN		Slave Pending interrupt Enable.	0x0
		0	Disabled. The SlvPending interrupt is disabled.	
		1	Enabled. The SlvPending interrupt is enabled.	
10:9	-	-	Reserved. Read value is undefined, only zero should be written.	-
11	SLVNOTSTREN		Slave Not Stretching interrupt Enable.	0x0
		0	Disabled. The SlvNotStr interrupt is disabled.	
		1	Enabled. The SlvNotStr interrupt is enabled.	
14:12	-	-	Reserved. Read value is undefined, only zero should be written.	-
15	SLVDESELEN		Slave Deselect interrupt Enable.	0x0
		0	Disabled. The SlvDeSel interrupt is disabled.	
		1	Enabled. The SlvDeSel interrupt is enabled.	

Table 366. Interrupt Enable Set and read register (INTENSET, offset 0x808) bit description

Bit	Symbol	Value	Description	Reset value
16	MONRDYEN		Monitor data Ready interrupt Enable.	0x0
		0	Disabled. The MonRdy interrupt is disabled.	
		1	Enabled. The MonRdy interrupt is enabled.	
17	MONOVEN		Monitor Overrun interrupt Enable.	0x0
		0	Disabled. The MonOv interrupt is disabled.	
		1	Enabled. The MonOv interrupt is enabled.	
18	-	-	Reserved. Read value is undefined, only zero should be written.	-
19	MONIDLEEN		Monitor Idle interrupt Enable.	0x0
		0	Disabled. The MonIdle interrupt is disabled.	
		1	Enabled. The MonIdle interrupt is enabled.	
23:20	-	-	Reserved. Read value is undefined, only zero should be written.	-
24	EVENTTIMEOUTEN		Event time-out interrupt Enable.	0x0
		0	Disabled. The Event time-out interrupt is disabled.	
		1	Enabled. The Event time-out interrupt is enabled.	
25	SCLTIMEOUTEN		SCL time-out interrupt Enable.	0x0
		0	Disabled. The SCL time-out interrupt is disabled.	
		1	Enabled. The SCL time-out interrupt is enabled.	
31:26	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.4 Interrupt Enable Clear register

Writing a 1 to a bit position in INTENCLR clears the corresponding position in the INTENSET register, disabling that interrupt. INTENCLR is a write-only register.

Bits that do not correspond to defined bits in INTENSET are reserved and only zeroes should be written to them.

Table 367. Interrupt Enable Clear register (INTENCLR, offset 0x80C) bit description

Bit	Symbol	Description	Reset value
0	MSTPENDINGCLR	Master Pending interrupt clear. Writing 1 to this bit clears the corresponding bit in the INTENSET register if implemented.	0x0
3:1	-	Reserved. Read value is undefined, only zero should be written.	-
4	MSTARBLOSSCLR	Master Arbitration Loss interrupt clear.	0x0
5	-	Reserved. Read value is undefined, only zero should be written.	-
6	MSTSTSTPERRCLR	Master Start/Stop Error interrupt clear.	0x0
7	-	Reserved. Read value is undefined, only zero should be written.	-
8	SLVPENDINGCLR	Slave Pending interrupt clear.	0x0
10:9	-	Reserved. Read value is undefined, only zero should be written.	-
11	SLVNOTSTRCLR	Slave Not Stretching interrupt clear.	0x0
14:12	-	Reserved. Read value is undefined, only zero should be written.	-
15	SLVDESELCLR	Slave Deselect interrupt clear.	0x0
16	MONRDYCLR	Monitor data Ready interrupt clear.	0x0
17	MONOVCLR	Monitor Overrun interrupt clear.	0x0
18	-	Reserved. Read value is undefined, only zero should be written.	-

Table 367. Interrupt Enable Clear register (INTENCLR, offset 0x80C) bit description *…continued*

Bit	Symbol	Description	Reset value
19	MONIDLECLR	Monitor Idle interrupt clear.	0x0
23:20	-	Reserved. Read value is undefined, only zero should be written.	-
24	EVENTTIMEOUTCLR	Event time-out interrupt clear.	0x0
25	SCLTIMEOUTCLR	SCL time-out interrupt clear.	0x0
31:26	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.5 Time-out value register

The TIMEOUT register allows setting an upper limit to certain I^{2}C bus times, informing by status flag and/or interrupt when those times are exceeded.

Two time-outs are generated, and software can elect to use either of them.

1. EVENTTIMEOUT checks the time between bus events while the bus is not idle: Start, SCL rising, SCL falling, and Stop. The EVENTTIMEOUT status flag in the STAT register is set if the time between any two events becomes longer than the time configured in the TIMEOUT register. The EVENTTIMEOUT status flag can cause an interrupt if enabled to do so by the EVENTTIMEOUTEN bit in the INTENSET register.
2. SCLTIMEOUT checks only the time that the SCL signal remains low while the bus is not idle. The SCLTIMEOUT status flag in the STAT register is set if SCL remains low longer than the time configured in the TIMEOUT register. The SCLTIMEOUT status flag can cause an interrupt if enabled to do so by the SCLTIMEOUTEN bit in the INTENSET register.

Also see Section 23.7.3.

Table 368. Time-out value register (TIMEOUT, offset 0x810) bit description

Bit	Symbol	Description	Reset value
3:0	TOMIN	Time-out time value, bottom four bits. These are hard-wired to 0xF. This gives a minimum time-out of 16 I^{2}C function clocks and also a time-out resolution of 16 I^{2}C function clocks.	0xF
15:4	TO	Time-out time value. Specifies the time-out interval value in increments of 16 I^{2}C function clocks, as defined by the CLKDIV register. To change this value while I^{2}C is in operation, disable all time-outs, write a new value to TIMEOUT, then re-enable time-outs. 0x000: A time-out will occur after 16 counts of the I^{2}C function clock. 0x001: A time-out will occur after 32 counts of the I^{2}C function clock. ... 0xFFF: A time-out will occur after 65,536 counts of the I^{2}C function clock.	0xFFF
31:16	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.6 Clock Divider register

The CLKDIV register divides down the Flexcomm Interface clock (FCLK) to produce the I^{2}C function clock that is used to time various aspects of the I^{2}C interface. The I^{2}C function clock is used for some internal operations in the I^{2}C interface and to generate the timing required by the I^{2}C bus specification, some of which are user configured in the MSTTIME register for Master operation. Slave operation uses CLKDIV for some timing functions.

See Section 23.7.2.1 for details on bus rate setup.

Table 369. I²C Clock Divider register (CLKDIV, offset 0x814) bit description

Bit	Symbol	Description	Reset value
15:0	DIVVAL	This field controls how the Flexcomm Interface clock (FCLK) is used by the I²C functions that need an internal clock in order to operate. 0x0000: FCLK is used directly by the I²C. 0x0001: FCLK is divided by 2 before use. 0x0002: FCLK is divided by 3 before use. ... 0xFFFF: FCLK is divided by 65,536 before use.	0x0
31:16	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.7 Interrupt Status register

The INTSTAT register provides register provides a view of those interrupt flags that are currently enabled. This can simplify software handling of interrupts. See Table 363 for detailed descriptions of the interrupt flags.

Table 370. I²C Interrupt Status register (INTSTAT, offset 0x818) bit description

Bit	Symbol	Description	Reset value
0	MSTPENDING	Master Pending.	0x1
3:1	-	Reserved.	-
4	MSTARBLOSS	Master Arbitration Loss flag.	0x0
5	-	Reserved. Read value is undefined, only zero should be written.	-
6	MSTSTSTPERR	Master Start/Stop Error flag.	0x0
7	-	Reserved. Read value is undefined, only zero should be written.	-
8	SLVPENDING	Slave Pending.	0x0
10:9	-	Reserved. Read value is undefined, only zero should be written.	-
11	SLVNOTSTR	Slave Not Stretching status.	0x1
14:12	-	Reserved. Read value is undefined, only zero should be written.	-
15	SLVDESEL	Slave Deselected flag.	0x0
16	MONRDY	Monitor Ready.	0x0
17	MONOV	Monitor Overflow flag.	0x0
18	-	Reserved. Read value is undefined, only zero should be written.	-
19	MONIDLE	Monitor Idle flag.	0x0
23:20	-	Reserved. Read value is undefined, only zero should be written.	-
24	EVENTTIMEOUT	Event time-out Interrupt flag.	0x0
25	SCLTIMEOUT	SCL time-out Interrupt flag.	0x0
31:26	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.8 Master Control register

The MSTCTL register contains bits that control various functions of the I2C Master interface. Only write to this register when the master is pending (MSTPENDING: 1 in the STAT register, Table 363).

Software should always write a complete value to MSTCTL, and not OR new control bits into the register as is possible in other registers such as CFG. This is due to the fact that MSTSTART and MSTSTOP are not self-clearing flags. ORing in new data following a Start or Stop may cause undesirable side effects.

After an initial I2C Start, MSTCTL should generally only be written when the MSTPENDING flag in the STAT register is set, after the last bus operation has completed. An exception is when DMA is being used and a transfer completes. In this case there is no MSTPENDING flag, and the MSTDMA control bit would be cleared by software potentially at the same time as setting either the MSTSTOP or MSTSTART control bit.

Remark: When in the idle or slave NACKed states (see Table 364), set the MSTDMA bit either with or after the MSTCONTINUE bit. MSTDMA can be cleared at any time.

Table 371. Master Control register (MSTCTL, offset 0x820) bit description

Bit	Symbol	Value	Description	Reset value
0	MSTCONTINUE		Master Continue. This bit is write-only.	0x0
		0	No effect.	
		1	Continue. Informs the Master function to continue to the next operation. This must done after writing transmit data, reading received data, or any other housekeeping related to the next bus operation.	
1	MSTSTART		Master Start control. This bit is write-only.	0x0
		0	No effect.	
		1	Start. A Start will be generated on the I2C bus at the next allowed time.	
2	MSTSTOP		Master Stop control. This bit is write-only.	0x0
		0	No effect.	
		1	Stop. A Stop will be generated on the I2C bus at the next allowed time, preceded by a NACK to the slave if the master is receiving data from the slave (Master Receiver mode).	
3	MSTDMA		Master DMA enable. Data operations of the I2C can be performed with DMA. Protocol type operations such as Start, address, Stop, and address match must always be done with software, typically via an interrupt. Address acknowledgement must also be done by software except when the I2C is configured to be HSCAPABLE (and address acknowledgement is handled entirely by hardware) or when Automatic Operation is enabled. When a DMA data transfer is complete, MSTDMA must be cleared prior to beginning the next operation, typically a Start or Stop.This bit is read/write.	0x0
		0	Disable. No DMA requests are generated for master operation.	
		1	Enable. A DMA request is generated for I2C master data operations. When this I2C master is generating Acknowledge bits in Master Receiver mode, the acknowledge is generated automatically.	
31:4	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.9 Master Time register

The MSTTIME register allows programming of certain times that may be controlled by the Master function. These include the clock (SCL) high and low times, repeated Start setup time, and transmitted data setup time.

The I^2C clock pre-divider is described in Table 369.

Table 372. Master Time register (MSTTIME, offset 0x824) bit description

Bit	Symbol	Value	Description	Reset value
2:0	MSTSCLLOW		Master SCL Low time. Specifies the minimum low time that will be asserted by this master on SCL. Other devices on the bus (masters or slaves) could lengthen this time. This corresponds to the parameter t_{LOW} in the I^2C bus specification. I^2C bus specification parameters t_{BUF} and $t_{SU;STA}$ have the same values and are also controlled by MSTSCLLOW.	0x7
		0x0	2 clocks. Minimum SCL low time is 2 clocks of the I^2C clock pre-divider.	
		0x1	3 clocks. Minimum SCL low time is 3 clocks of the I^2C clock pre-divider.	
		0x2	4 clocks. Minimum SCL low time is 4 clocks of the I^2C clock pre-divider.	
		0x3	5 clocks. Minimum SCL low time is 5 clocks of the I^2C clock pre-divider.	
		0x4	6 clocks. Minimum SCL low time is 6 clocks of the I^2C clock pre-divider.	
		0x5	7 clocks. Minimum SCL low time is 7 clocks of the I^2C clock pre-divider.	
		0x6	8 clocks. Minimum SCL low time is 8 clocks of the I^2C clock pre-divider.	
		0x7	9 clocks. Minimum SCL low time is 9 clocks of the I^2C clock pre-divider.	
3	-	-	Reserved.	-
6:4	MSTSCLHIGH		Master SCL High time. Specifies the minimum high time that will be asserted by this master on SCL. Other masters in a multi-master system could shorten this time. This corresponds to the parameter t_{HIGH} in the I^2C bus specification. I^2C bus specification parameters $t_{SU;STO}$ and $t_{HD;STA}$ have the same values and are also controlled by MSTSCLHIGH.	0x7
		0x0	2 clocks. Minimum SCL high time is 2 clock of the I^2C clock pre-divider.	
		0x1	3 clocks. Minimum SCL high time is 3 clocks of the I^2C clock pre-divider .	
		0x2	4 clocks. Minimum SCL high time is 4 clock of the I^2C clock pre-divider.	
		0x3	5 clocks. Minimum SCL high time is 5 clock of the I^2C clock pre-divider.	
		0x4	6 clocks. Minimum SCL high time is 6 clock of the I^2C clock pre-divider.	
		0x5	7 clocks. Minimum SCL high time is 7 clock of the I^2C clock pre-divider.	
		0x6	8 clocks. Minimum SCL high time is 8 clock of the I^2C clock pre-divider.	
		0x7	9 clocks. Minimum SCL high time is 9 clocks of the I^2C clock pre-divider.	
31:7	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.10 Master Data register

The MSTDAT register provides the means to read the most recently received data for the Master function, and to transmit data using the Master function.

Table 373. Master Data register (MSTDAT, offset 0x828) bit description

Bit	Symbol	Description	Reset value
7:0	DATA	Master function data register. Read: read the most recently received data for the Master function. Write: transmit data using the Master function.	0x0
31:8	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.11 Slave Control register

The SLVCTL register contains bits that control various functions of the I^2C Slave interface. Only write to this register when the slave is pending (SLVPENDING: 1 in the STAT register, Table 363).

Refer to Section 23.7.8 for details of the AUTOACK, AUTOMATCHREAD, and related settings.

Remark: When in the slave address state (slave state 0, see Table 365), set the SLVDMA bit either with or after the SLVCONTINUE bit. SLVDMA can be cleared at any time.

Table 374. Slave Control register (SLVCTL, offset 0x840) bit description

Bit	Symbol	Value	Description	Reset Value
0	SLVCONTINUE		Slave Continue.	0x0
		0	No effect.	
		1	Continue. Informs the Slave function to continue to the next operation, by clearing the SLVPENDING flag in the STAT register. This must be done after writing transmit data, reading received data, or any other housekeeping related to the next bus operation. Automatic Operation has different requirements. SLVCONTINUE should not be set unless SLVPENDING: 1.	
1	SLVNACK		Slave NACK.	0x0
		0	No effect.	
		1	NACK. Causes the Slave function to NACK the master when the slave is receiving data from the master (Slave Receiver mode).	
2	-	-	Reserved. Read value is undefined, only zero should be written.	-
3	SLVDMA		Slave DMA enable.	0x0
		0	Disabled. No DMA requests are issued for Slave mode operation.	
		1	Enabled. DMA requests are issued for I^2C slave data transmission and reception.	
7:4	-	-	Reserved. Read value is undefined, only zero should be written.	-

Table 374. Slave Control register (SLVCTL, offset 0x840) bit description

Bit	Symbol	Value	Description	Reset Value
8	AUTOACK		Automatic Acknowledge.When this bit is set, it will cause an I^{2}C header which matches SLVADR0 and the direction set by AUTOMATCHREAD to be ACKed immediately; this is used with DMA to allow processing of the data without intervention. If this bit is cleared and a header matches SLVADR0, the behavior is controlled by AUTONACK in the SLVADR0 register: allowing NACK or interrupt.	0x0
		0	Normal, non-automatic operation. If AUTONACK: 0, an SlvPending interrupt is generated when a matching address is received. If AUTONACK: 1, received addresses are NACKed (ignored).	
		1	A header with matching SLVADR0 and matching direction as set by AUTOMATCHREAD will be ACKed immediately, allowing the master to move on to the data bytes. The ACK will clear this bit. If the address matches SLVADR0, but the direction does not match AUTOMATCHREAD, the behavior will depend on the bit in the SLVADR0 register: if AUTONACK is set, then it will be Nacked; else if AUTONACK is clear, then a SlvPending interrupt is generated.	
9	AUTOMATCHREAD		When AUTOACK is set, this bit controls whether it matches a read or write request on the next header with an address matching SLVADR0. Since DMA needs to be configured to match the transfer direction, the direction needs to be specified. This bit allows a direction to be chosen for the next operation.	0x0
		0	The expected next operation in Automatic Mode is an I^{2}C write.	
		1	The expected next operation in Automatic Mode is an I^{2}C read.	
31:10	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.12 Slave Data register

The SLVDAT register provides the means to read the most recently received data for the Slave function and to transmit data using the Slave function.

Table 375. Slave Data register (SLVDAT, offset 0x844) bit description

Bit	Symbol	Description	Reset value
7:0	DATA	Slave function data register. Read: read the most recently received data for the Slave function. Write: transmit data using the Slave function.	0x0
31:8	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.13 Slave Address 0 register

The SLVADR0 register allows enabling and defining one of the addresses that can be automatically recognized by the I^{2}C slave hardware.

The I^{2}C slave function has a total of 4 address comparators. The value in SLVADR0 can be qualified by the setting of the SLVQUAL0 register. The additional 3 address comparators do not include the address qualifier feature. For handling of the general call address, one of the 4 address registers can be programmed to respond to address 0.

Refer to Section 23.7.8 for details of AUTONACK and related settings.

Table 376. Slave Address 0 register (SLVADR[0], offset 0x848) bit description

Bit	Symbol	Value	Description	Reset value
0	SADISABLE0		Slave Address 0 Disable.	0x1
		0	Enabled. Slave Address 0 is enabled.	
		1	Ignored Slave Address 0 is ignored.	
7:1	SLVADR0	-	Slave Address. Seven bit slave address that is compared to received addresses if enabled. The compare can be affected by the setting of the SLVQUAL0 register.	0x0
14:8	-	-	Reserved. Read value is undefined, only zero should be written.	-
15	AUTONACK		Automatic NACK operation. Used in conjunction with AUTOACK and AUTOMATCHREAD, allows software to ignore I^2C traffic while handling previous I^2C data or other operations.	0x0
		0	Normal operation, matching I^2C addresses are not ignored.	
		1	Automatic-only mode. All incoming addresses are ignored (NACKed), unless AUTOACK is set, it matches SLVADR0, and AUTOMATCHREAD matches the direction.	
31:16	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.14 Slave Address 1, 2, and 3 registers

These slave address registers provide for three additional addresses that can be automatically recognized by the I2C slave hardware.

Table 377. Slave Address registers (SLVADR[1:3], offset [0x84C:0x854]) bit description

Bit	Symbol	Value	Description	Reset value
0	SADISABLE		Slave Address n Disable.	0x1
		0	Enabled. Slave Address n is enabled.	
		1	Ignored Slave Address n is ignored.	
7:1	SLVADR		Slave Address. Seven bit slave address that is compared to received addresses if enabled.	0x0
14:8	-	-	Reserved. Read value is undefined, only zero should be written.	-
15	AUTONACK		Automatic NACK operation. Used in conjunction with AUTOACK and AUTOMATCHREAD, allows software to ignore I^2C traffic while handling previous I^2C data or other operations.	0x0
		0	Normal operation, matching I^2C addresses are not ignored.	
		1	Automatic-only mode. All incoming addresses are ignored (NACKed), unless AUTOACK is set, it matches SLVADR0, and AUTOMATCHREAD matches the direction.	
31:16	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.15 Slave address Qualifier 0 register

The SLVQUAL0 register can alter how Slave Address 0 (specified by the SLVADR0 register) is interpreted.

Table 378. Slave address Qualifier 0 register (SLVQUAL0, offset 0x858) bit description

Bit	Symbol	Value	Description	Reset Value
0	QUALMODE0		Qualify mode for slave address 0.	0x0
		0	Mask. The SLVQUAL0 field is used as a logical mask for matching address 0.	
		1	Extend. The SLVQUAL0 field is used to extend address 0 matching in a range of addresses.	
7:1	SLVQUAL0	-	Slave address Qualifier for address 0. A value of 0 causes the address in SLVADR0 to be used as-is, assuming that it is enabled. If QUALMODE0: 0, any bit in this field which is set to 1 will cause an automatic match of the corresponding bit of the received address when it is compared to the SLVADR0 register. If QUALMODE0: 1, an address range is matched for address 0. This range extends from the value defined by SLVADR0 to the address defined by SLVQUAL0 (address matches when SLVADR0[7:1] ≤ received address ≤ SLVQUAL0[7:1]).	0x0
31:8	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.16 Monitor data register

The read-only MONRXDAT register provides information about events on the I²C bus, primarily to facilitate debugging of the I²C during application development. All data addresses and data passing on the bus and whether these were acknowledged, as well as Start and Stop events, are reported.

The Monitor function must be enabled by the MONEN bit in the CFG register. Monitor mode can be configured to stretch the I²C clock if data is not read from the MONRXDAT register in time to prevent it, via the MONCLKSTR bit in the CFG register. This can help ensure that nothing is missed but can cause the Monitor function to be somewhat intrusive (by potentially adding clock delays, depending on software or DMA response time). In order to improve the chance of collecting all Monitor information if clock stretching is not enabled, Monitor data is buffered such that it is available until the end of the next piece of information from the I²C bus.

Details of clock stretching are different in HS mode, see "Clock stretching".

Table 379. Monitor data register (MONRXDAT, offset 0x880) bit description

Bit	Symbol	Value	Description	Reset value
7:0	MONRXDAT	-	Monitor function Receiver Data. This reflects every data byte that passes on the I²C pins.	0x0
8	MONSTART		Monitor Received Start.	0x0
		0	No start detected. The Monitor function has not detected a Start event on the I²C bus.	
		1	Start detected. The Monitor function has detected a Start event on the I²C bus.	
9	MONRESTART		Monitor Received Repeated Start.	0x0
		0	No repeated start detected. The Monitor function has not detected a Repeated Start event on the I²C bus.	
		1	Repeated start detected. The Monitor function has detected a Repeated Start event on the I²C bus.	

Table 379. Monitor data register (MONRXDAT, offset 0x880) bit description

Bit	Symbol	Value	Description	Reset value
10	MONNACK		Monitor Received NACK.	0x0
		0	Acknowledged. The data currently being provided by the Monitor function was acknowledged by at least one master or slave receiver.	
		1	Not acknowledged. The data currently being provided by the Monitor function was not acknowledged by any receiver.	
31:11	-	-	Reserved. Read value is undefined, only zero should be written.	-

23.6.17 Module identification register

The ID register identifies the type and revision of the I^{2}C module. A generic SW driver can make use of this information register to implement module type or revision specific behavior.

Table 380. Module identification register (ID - offset 0xFFC) bit description

Bit	Symbol	Description	Reset Value
7:0	APERTURE	Aperture: encoded as (aperture size/4K) -1, so 0x00 means a 4K aperture.	0x00
11:8	MINOR_REV	Minor revision of module implementation, starting at 0. Minor revision of module implementation, starting at 0. Software compatibility is expected between minor revisions.	-
15:12	MAJOR_REV	Major revision of module implementation, starting at 0. There may not be software compatibility between major revisions.	-
31:16	ID	Unique module identifier for this IP block.	0xE030

23.7 Functional description

23.7.1 AHB bus access

The bus interface to the I^2C registers contained in the Flexcomm Interface support only word writes. Byte and halfword writes are not supported in conjunction with the I^2C function.

23.7.2 Bus rates and timing considerations

Due to the nature of the I^2C bus, it is generally not possible to guarantee a specific clock rate on the SCL pin. On the I^2C-bus, the clock can be stretched by any slave device, extended by software overhead time, etc.

In a multi-master system, the master that provides the shortest SCL high time will cause that time to appear on SCL as long as that master is participating in I^2C traffic (i.e. when it is the only master on the bus, or during arbitration between masters).

In addition, I^2C implementations generally base subsequent actions on what actually happens on the bus lines. For instance, a bus master allows SCL to go high. It then monitors the line to make sure it actually did go high (this would be required in a multi-master system). This results in a small delay before the next action on the bus, caused by the rise time of the open drain bus line.

Rate calculations give a base frequency that represents the fastest that the I^2C bus could operate if nothing slows it down.

23.7.2.1 Rate calculations

Master timing

SCL high time (in Flexcomm Interface function clocks):
(CLKDIV + 1) * (MSTSCLHIGH + 2)

SCL low time (in Flexcomm Interface function clocks):
(CLKDIV + 1) * (MSTSCLLOW + 2)

Nominal SCL rate:
Flexcomm Interface function clock rate / (SCL high time + SCL low time)

Slave timing

Most aspects of slave operation are controlled by SCL received from the I^2C bus master. However, if the slave function stretches SCL to allow for software response, it must provide sufficient data setup time to the master before releasing the stretched clock. This is accomplished by inserting one clock time of CLKDIV at that point.

If CLKDIV is already configured for master operation, that is sufficient. If only the slave function is used, CLKDIV should be configured such that one clock time is greater than the tSU;DAT value noted in the I^2C bus specification for the I^2C mode that is being used.

23.7.2.2 Bus rate support

The I^2C interface can support up to 400 kbits/s rate.

Clock stretching: The I^2C interface automatically stretches the clock when it does not have sufficient information on how to proceed, i.e. software has not supplied data and/or instructions to generate a start or stop. In principle, at least, I^2C can allow the clock to be stretched by any bus participant at any time that SCL is low.

In practice, the I^2C interface described here may stretch SCL at the following times:

- As a Slave:
  - after an address is received that complies with at least one slave address (before the address is acknowledged)
  - as a slave receiver, after each data byte received (software then acknowledges the data)
  - as a slave transmitter, after each data byte is sent and the matching acknowledge is received from the master
- As a master:

  after each
  - address is sent and the acknowledge bit has been received
  - as a master receiver, after each after each data byte is received (software then acknowledges the data)
  - as a master transmitter, after each data byte is sent and the matching acknowledge bit has been received from the slave

In each case, the relevant pending flag (MSTPENDING or SLVPENDING) is set at the point where clock stretching occurs.

23.7.3 Time-out

A time-out feature on an I^2C interface can be used to detect a “stuck” bus and potentially do something to alleviate the condition. Two different types of time-out are supported. Both types apply whenever the I^2C interface and the time-out function are both enabled. Master, Slave, or Monitor functions do not need to be enabled.

In the first type of time-out, reflected by the EVENTTIMEOUT flag in the STAT register, the time between bus events governs the time-out check. These events include Start, Stop, and all changes on the I^2C clock (SCL). This time-out is asserted when the time between any of these events is longer than the time configured in the TIMEOUT register. This time-out could be useful in monitoring an I^2C bus within a system as part of a method to keep the bus running of problems occur.

The second type of I^2C time-out is reflected by the SCLTIMEOUT flag in the STAT register. This time-out is asserted when the SCL signal remains low longer than the time configured in the TIMEOUT register. In this situation, a slave could reset its own I^2C interface in case it is the offending device. If all listening slaves (including masters that can be addressed as slaves) do this, then the bus will be released unless it is a current master causing the problem.

Both types of time-out are generated only when the I^2C bus is considered busy, i.e. when there has been a Start condition more recently than a Stop condition.

23.7.4 Ten-bit addressing

Ten-bit addressing is accomplished by the I^{2}C master sending a second address byte to extend a particular range of standard 7-bit addresses. In the case of the master writing to the slave, the I^{2}C frame simply continues with data after the 2 address bytes. For the master to read from a slave, it needs to reverse the data direction after the second address byte. This is done by sending a Repeated Start, followed by a repeat of the same standard 7-bit address, with a Read bit. The slave must remember that it had been addressed by the previous write operation and stay selected for the subsequent read with the correct partial I^{2}C address.

For the Master function, the I^{2}C is simply instructed to perform the 2-byte addressing as a normal write operation, followed either by more write data, or by a Repeated Start with a repeat of the first part of the 10-bit slave address and then reading in the normal fashion.

For the Slave function, the first part of the address is automatically matched in the same fashion as 7-bit addressing. The slave address qualifier feature (see Section 23.6.15) can be used to intercept all potential 10-bit addresses (first address byte values F0 through F6), or just one. In the case of Slave Receiver mode, data is received in the normal fashion after software matches the first data byte to the remaining portion of the 10-bit address. The Slave function should record the fact that it has been addressed, in case there is a follow-up read operation.

For Slave Transmitter mode, the slave function responds to the initial address in the same fashion as for Slave Receiver mode, and checks that it has previously been addressed with a full 10-bit address. If the address matched is address 0, and address qualification is enabled, software must check that the first part of the 10-bit address is a complete match to the previous address before acknowledging the address.

23.7.5 Clocking and power considerations

The Master function of the I^{2}C always requires a peripheral clock to be running in order to operate. The Slave function can operate without any internal clocking when the slave is not currently addressed. This means that reduced power modes up to deep-sleep mode can be entered, and the device will wake up when the I^{2}C Slave function recognizes an address. Monitor mode can similarly wake up the device from a reduced power mode when information becomes available.

23.7.6 Interrupt handling

The I^{2}C provides a single interrupt output that handles all interrupts for Master, Slave, and Monitor functions.

23.7.7 DMA

DMA with the I^{2}C is done only for data transfer, DMA cannot handle control of the I^{2}C. Once DMA is transferring data, I^{2}C acknowledges are handled implicitly. No CPU intervention is required while DMA is transferring data.

Generally, data transfers can be handled by DMA for Master mode after an address is sent and acknowledged by a slave, and for Slave mode after software has acknowledged an address. In either mode, software is always involved in the address portion of a message. In master and slave modes, data receive and transmit data can be transferred by the DMA. The DMA supports three DMA requests: data transfer in master mode, slave mode, and Monitor mode.

DMA may be used in connection with Automatic Operation in order to minimize software overhead time for I^{2}C handling.

A received NACK (from a slave in Master mode, or from a master in Slave mode) will cause DMA to stop and an interrupt to be generated. A Repeated Start sensed on the bus will similarly cause DMA to stop and an interrupt to be generated.

The Monitor function may be used with DMA if a channel is available See "DMA with I^{2}C-bus monitor mode" for how DMA channels are used with the Monitor function.

23.7.7.1 DMA as a Master transmitter

A basic sequence for a Master transmitter:

- Software sets up DMA to transmit a message.
- Software causes a slave address with write command to be sent and checks that the address was acknowledged.
- Software turns on DMA mode in the I^{2}C.
- DMA transfers data and eventually completes the transfer.
- Software causes a stop (or repeated start) to be sent.

Software will be invoked to handle any exceptions to the standard transfer, such as the slave sending a NACK before the end of the transfer.

23.7.7.2 DMA as a Master receiver

A basic sequence for a Master receiver:

- Software sets up DMA to receive a message.
- Software causes a slave address with read command to be sent and checks that the address was acknowledged.
- Software starts DMA.
- DMA completes.
- Software causes a stop or repeated start to be sent.

Software will be invoked to handle any exceptions to the standard transfer.

23.7.7.3 DMA as a Slave transmitter

A basic sequence for a Slave transmitter:

- Software acknowledges an I^{2}C address.
- Software sets up DMA to transmit a message.
- Software starts DMA.
- DMA completes.

23.7.7.4 DMA as a Slave receiver

A basic sequence for a Slave receiver:

- Software receives an interrupt for a slave address received, and acknowledges the address.
- Software sets up DMA to receive a message, less the final data byte.

- Software starts DMA.
- DMA completes.
- Software sets SLVNACK prior to receiving the final data byte.
- Software receives the final data byte.

23.7.8 Automatic operation

Automatic operation modes provide a way to reduce software overhead for I2C slave functions with some limitations. They are intended to be used primarily in conjunction with slave DMA. Related control bits are SLVDMA, AUTOACK, and AUTOMATCHREAD in the SLCCTL register, and the AUTONACK bit in the SLVADR0 register. Table 27 shows how these controls may be used. These cases apply when an address matching SLVADR0, qualified by SLVQUAL0, is received.

Table 381: Automatic operation cases

Conditions:			Response:		
AUTONACK bit	**AUTOACK bit**	**Received R/W bit matches AUTOMATCHREAD**	**SLVPENDING interrupt generated?**	**ACK/NACK on I2C bus**	**Description**
0	0	x	Yes	None	Normal, non-automatic operation.
0	1	No	Yes	None	Automatic slave DMA: unexpected read/write case. Same as normal non-automatic operation.
x	1	Yes	No	ACK	Automatic slave DMA: expected read/write case. When the automatic Ack is sent, the SLVDMA bit is set and the AUTOACK bit is cleared.
1	0	x	No	NACK	Bus is ignored until software changes the setup.
1	1	No	No	NACK	Bus is ignored until software changes the setup.

24. Quadrature decoder (QDEC)

24.1 Introduction

Two Quadrature decoder (QDEC0, QDEC1) are available on all QN908x parts. Each QDEC provides decoding of quadrature-encoded signals.

24.2 Features

- Each QDEC supports optional input debounce filters.
- Configurable sample clock and debounce filters sample clock.
- Each QDEC contains two accumulation registers to accumulate valid motion sample values and invalid samples respectively.
- Auto clearing is supported when sample ends.

24.3 Basic configuration

The QDEC is configured as follows:

- If needed, write 1 to SET_QDEC0_RST or SET_QDEC1_RST bit in RST_SW_SET register to reset QDEC0 or QDEC1. Then write 1 to CLR_QDEC0_RST or CLR_QDEC1_RST bit in RST_SW_CLR registers to release reset of QDEC0 or QDEC1.
- Write 1 to CLK_QDEC0_EN or CLK_QDEC1_EN bit in CLK_EN register to enable clock for the related QDEC.
- Enable QDEC, select auto clear mode, enable or disable debounce filter by writing to CTRL register (Section 24.6.1) QDEC.
- Configure sample clock, sample points and debounce filter sample clock by writing to SAMP_CTRL register of the related QDEC.
- Configure INTEN register to set the interrupt mask.
- Start QDEC by writing 1 to START bit in CTRL register.
- Wait until the SAMPLE_END interrupt generates, then configure SOFT_CLR bit to 1 if AUTO_CLR_EN bit was not set before the QDEC started.
- Read ACC_R register to check accumulator results.
- Write 1 to STOP bit in CTRL register to stop QDEC.

24.4 Pin description

Each QDEC allows 2 pin connections. The function pins are assigned to external physical pins via PINMUX, see Section 8.3.2.1. For example, if PA02 and PA03 are used as QDEC function, PIO_FUNC_CFG0 register bit 8~15 should be configured to 0x01, since QDEC is function 1 of these pins.

Table 382. QDEC pin description

Pin	Type	Pin name used in Pin Description chapter	Description
A_PHASE	I	QDECn_A	Quadrature encoder signal phase A on QDECn.
B_PHASE	I	QDECn_B	Quadrature encoder signal phase B on QDECn.

24.5 General description

The architecture of QDEC is shown in Figure 74.

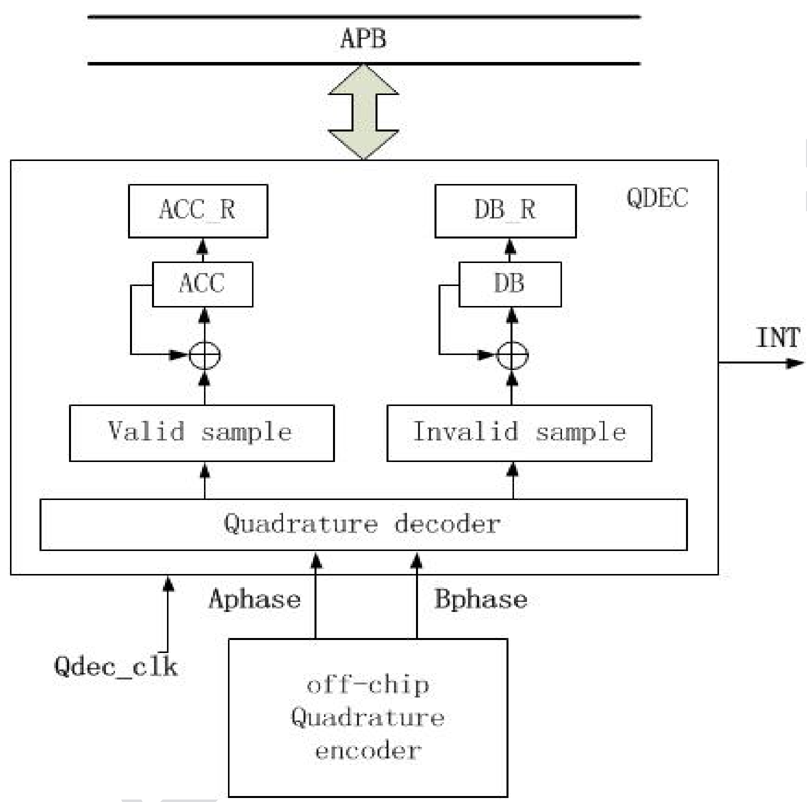


Fig 74. QDEC block diagram

24.5.1 Sampling and Decoding

The QDEC samples phase input pins from an off-chip quadrature encoder, then decodes it.

The two signals phase A and phase B generated by an incremental motion encoder are always 90 degree out of phase. Which of the two waveforms changes first indicates the direction of movement. However, invalid transition may occur because of the inconsistence between signal frequency and the sample rate. It means the two waveforms switch simultaneously in the sampled data. Obviously, this case may occur when the wheel rotates faster than the sample rate of QDEC.

The sample rate is decided by symbol DIVIDE in SAMP_CTL register. If DIVIDE needs to be changed, it is strongly recommended that the QDEC should be stopped first. The QDEC can be restarted after the change is finished. Actually, it is good practice to change any other registers only when the QDEC is stopped.

The QDEC samples the two input pins and decode them by comparing the current sample pair with the previous sample pair. The details of decoding is shown in the table below.

Table 383. Sampled value decoding

Prev sample pair		Curr sapmle pair		Sample register	ACC register	DB register	description
A	B	A	B				
0	0	0	0	0	unchanged	unchanged	No movement
		0	1	-1	decrement	unchanged	Negative movement
		1	0	1	increment	unchanged	Positive movement
		1	1	2	unchanged	increment	Double transition
0	1	0	0	1	increment	unchanged	Positive movement
		0	1	0	unchanged	unchanged	No movement
		1	0	2	unchanged	increment	Double transition
		1	1	-1	decrement	unchanged	Negative movement
1	0	0	0	-1	decrement	unchanged	Negative movement
		0	1	2	unchanged	increment	Double transition
		1	0	0	unchanged	unchanged	No movement
		1	1	1	increment	unchanged	Positive movement
1	1	0	0	2	unchanged	increment	Double transition
		0	1	1	increment	unchanged	Positive movement
		1	0	-1	decrement	unchanged	Negative movement
		1	1	0	unchanged	unchanged	No movement

24.5.2 Debounce Filter

For both QDEC0 and QDEC1, each of the two inputs have optional debounce filters. The debounce filter can be enabled by configuring symbol DB_FILTER_EN in CTRL register. The filter inputs are sampled at a configurable sample rate during the entire sample period, and the filters requires all of the samples within one sample period to be equal so that the input signal can be accepted. The sample rate of debounce filter is determined by symbol DB_SAMP_DIV in SAMP_CTRL register.

24.5.3 Accumulator Registers

There are two accumulator registers in each QDEC, ACC and DB, that accumulate valid samples and the number of invalid samples (double transition) respectively.

The ACC_R and DB_R registers are going to get the snapshot of ACC and DB immediately when sample ends.

The ACC register accumulates all valid sample values (+1/-1). By using ACC register the application does not to get every sample values from SAMPLE register. If the decoder receives a sample value that would cause the ACC to overflow or underfow, the ACC_OF interrupt flag in INT register will be set, and this sample value will be discarded.

The DB register accumulates the number of double transitions. The overflow of DB will set the DB_OF interrupt flag in INT register.

In common application, PTS should be configure to decide how many sample points will be get in one sample operation. After the sample ends, auto clear will occur if AUTO_CLR_EN bit was set, otherwise, writing 1 to SOFT_CLR should be done by software. Both clear operation (auto or manual) will cause the snapshot transfer to ACC_R and DB_R registers, and reset ACC and DB registers.

A START or STOP operation of QDEC will reset ACC, ACC_R, DB, DB_R registers, so do not start or stop QDEC before you read the useful information from accumulator registers.

24.6 Register description

Each QDEC contains same register file. The base addresses of QDEC0 and QDEC1 can be found in the table below.

Table 384. QDEC register base address

Base address	Peripheral	Instance	Description
0x4000 9000	QDEC	QDEC0	Quadrature decoder 0
0x4000 9800	QDEC	QDEC1	Quadrature decoder 1

Table 385. QDEC Register overview for one QDEC

Name	Access	Offset	Description	Reset value	Section
CTRL	R/W	0x00	QDEC configuration and control register.	0x0	24.6.1
SAMP_CTRL	R/W	0x04	QDEC sample control register.	0x0	24.6.2
SAMPLE	R	0x08	QDEC sample value register.	0x0	24.6.3
ACC	R	0x0C	QDEC valid sample value accumulator register.	0x0	24.6.4
ACC_R	R	0x10	ACC snapshot register.	0x0	24.6.5
DB	R	0x14	QDEC invalid sample value accumulator register.	0x0	24.6.6
DB_R	R	0x18	DB snapshot register.	0x0	24.6.7
INT	R/W	0x1C	QDEC interrupt status register.	0x0	24.6.8
INTEN	R/W	0x20	QDEC interrupt mask register.	0x0	24.6.9
STAT	R	0x24	QDEC status register.	0x0	24.6.10

24.6.1 QDEC Configuration and Control register

The CTRL register contains the basic operation of QDEC and enable for QDEC, debounce filters and auto clear mode.

Table 386. QDEC Configuration and Control register (CTRL, offset: 0x00) bit description

Bit	Symbol	Value	Description	Access	Reset Value
31:7	-	-	Reserved	-	-
6	DB_FILTER_EN		Debounce filter enable.	R/W	0x0
		0	Debounce filter disable.		
		1	Debounce filter enable.		
5	SINGLE_SAMPLE_SRST_EN		Single sample soft reset enable.	R/W	0x0
		0	No function.		
		1	Each SINGLE_SAMPLE interrupt will trigger soft reset.		
4	AUTO_CLR_EN		Auto clear enable.	R/W	0x0
		0	No auto clear. Software need to set SOFT_CLR bit to grasp ACC to ACC_R.		
		1	SOFT_CLR can be set automatically when SAMPLE_END interrupt is valid. Software can get ACC value from ACC_R register directly.		
3	SOFT_CLR		Software clear.	W	0x0
		0	No function.		
		1	Writing 1 to this bit cause soft clear operation.		
2	STOP		Stop operation.	W	0x0
		0	No function.		
		1	Stop the current sample and reset QDEC into initial state.		
1	START		Start opration.	W	0x0
		0	No function.		
		1	Software starts QDEC.		

Table 386. QDEC Configuration and Control register (CTRL, offset: 0x00) bit description

Bit	Symbol	Value	Description	Access	Reset Value
0	QDEC_EN	-	QDEC enable.	R/W	0x0
		0	QDEC disable.		
		1	QDEC enable.		

24.6.2 QDEC Sample Control register

The SAMP_CTL register identifies the sample points and sample clock divider.

Table 387. QDEC Sample Control register (SAMP_CTL, offset=0x04) bit description

Bit	Symbol	Value	Description		Access	Reset Value
31:20	-	-	Reserved	-	-	
19:16	DB_SAMP_DIV		Debounce filter sample clock divide. DB_SAMP_DIV can be configured as 0~6,then		R/W	0x0
			DB_SAMP_DIV	divide		
			0	1		
			1	2		
			2	4		
			3	8		
			4	16		
			5	32		
			6	64		
			other	1		
			DB sample clock: Qdec clock / divide			
15:12	-	-	Reserved	-	-	0x0
11:8	PTS		Total sample points.		R/W	0x0
			PTS	Total sample points		
			0	5		
			1	10		
			2	40		
			3	80		
			4	120		
			5	160		
			6	200		
			7	240		
			8	280		
			9	320		
			10	360		
			11	400		
			other	5		

Table 387. QDEC Sample Control register (SAMP_CTL, offset=0x04) bit description

Bit	Symbol	Value	Description		Access	Reset Value
7:5	-	-	Reserved	-	-	0x0
4:0	DIVIDE		Sample clock divide, can be configured as 0_{20},21 modes in total.		R/W	0x0
			sample clock: Qdec clock / (2^DIVIDE)			

24.6.3 QDEC Sample Value register

The SAMPLE register stores each sample values.

Table 388. QDEC Sample Value register (SAMPLE - offset 0x008) bit description

Bit	Symbol	Description	Reset value	Access
1:0	SAMPLE	Sample value each time (2’s complement) Noise transition will not be counted here. 2’b00: no transition 2’b01: +1 transition 2’b10: -1 transition	0x0	R
31:2	-	Reserved	-	-

24.6.4 QDEC Valid Accumulator register

The ACC register accumulates valid motion values.

Table 389. QDEC Valid Accumulator register (ACC - offset 0x0C) bit description

Bit	Symbol	Description	Reset value	Access
10:0	ACC	ACC snapshot when SAMPLE_END interrupt is valid.	0x0	R
31:11	-	Reserved	-	-

24.6.5 ACC Snapshot register

Table 390. ACC snapshot register (ACC_R - offset 0x010) bit description

Bit	Symbol	Description	Reset value	Access
10:0	ACC_R	Valid sample value accumulator with range (-1024~1023) in 2’s complement.2’b11: 2 transition.	0x0	R
31:11	-	Reserved	-	-

24.6.6 QDEC Invalid Accumulator register

The DB register accumulates invalid motion values

Table 391. QDEC Invalid Accumulator register (DB - offset 0x14) bit description

Bit	Symbol	Description	Reset value	Access
3:0	DB	Invalid sample value accumulator for double transition case with a max value 15.	0x0	R
31:11	-	Reserved	-	-

24.6.7 DB Snapshot register

Table 392. DB snapshot register (DB_R - offset 0x018) bit description

Bit	Symbol	Description	Reset value	Access
3:0	DB_R	DB snapshot when SAMPLE_END interrupt is valid.	0x0	R
31:11	-	Reserved	-	-

24.6.8 QDEC Interrupt register

Table 393. QDEC Interrupt register (INT - offset 0x1C) bit description

Bit	Symbol	Description	Reset value	Access
0	SINGLE_SAMPL	When each normal sample is done	0x0	RW
1	SINGLE_END	Sample end event triggered.	0x0	RW
2	ACC_OF	Valid sample accumulator overflow or	0x0	RW
3	DB_OF	Double transition accumulator overflow.	0x0	RW
31:1	-	Reserved	-	-

24.6.9 QDEC Interrupt Enable register

Table 394. QDEC Interrupt Enable register (INTEN- offset 0x20) bit description

Bit	Symbol	Description	Reset value	Access
0	SINGLE_SAMPLE_INTEN	1: Interrupt Enabled; 0: Interrupt Disabled.	0x0	RW
1	SINGLE_END_INTEN	1: Interrupt Enabled; 0: Interrupt Disabled.	0x0	RW
2	ACC_OF_INTEN	1: Interrupt Enabled; 0: Interrupt Disabled.	0x0	RW
3	DB_OF_INTEN	1: Interrupt Enabled; 0: Interrupt Disabled.	0x0	RW
31:11	-	Reserved	-	-

24.6.10 QDEC status register

Table 395. QDEC Status register (STAT - offset 0x24) bit description

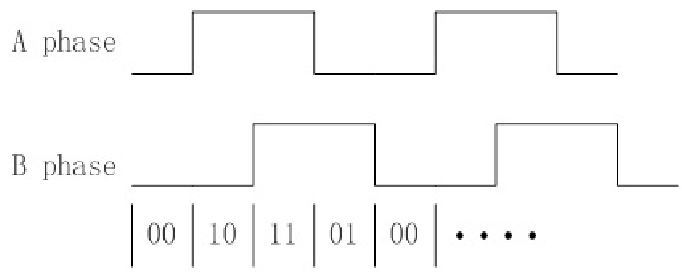
Bit	Symbol	Description	Reset value	Access
0	QDEC_BUSY	QDEC is running.	0x0	R
31:1	-	Reserved	-	-

24.7 Functional description

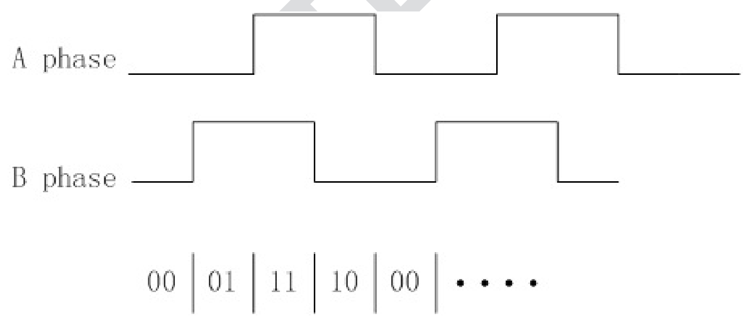
24.7.1 Input Waveform

24.7.1.1 Normal case

1. Movement in positive direction:

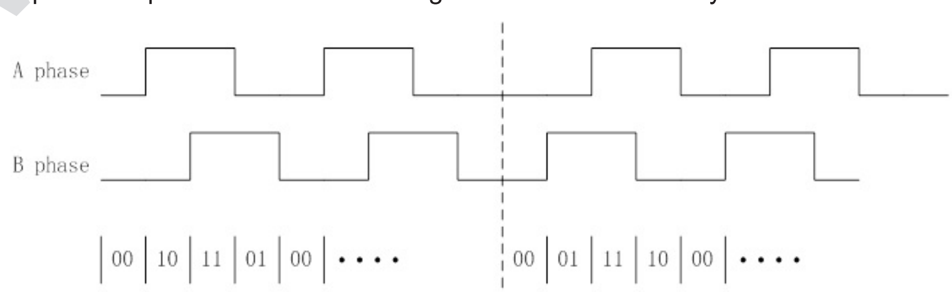


2. Movement in negative direction:

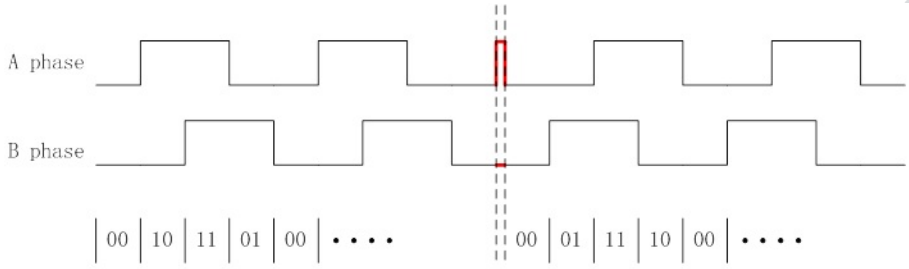


24.7.1.2 Special case

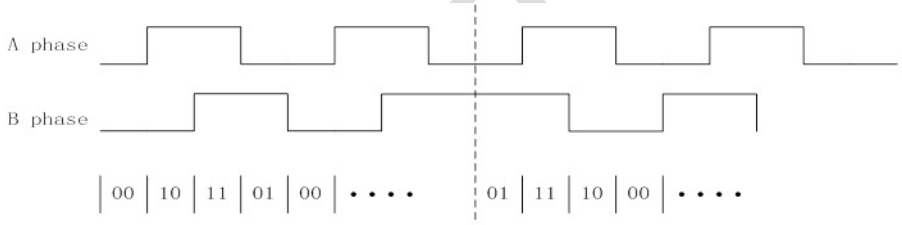
1. Codewheel rotates in positive direction, the light window is close to A phase but NOT passes A phase. Then turns to negative direction suddenly.



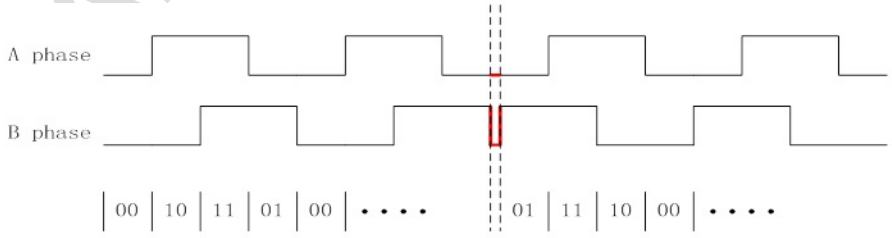
2. Codewheel rotates in positive direction, the light window just passes A phase a little, then turns to negative direction suddenly.



3. Codewheel rotates in positive direction, the light window is close to B phase but NOT passes A phase. Then turns to negative direction suddenly.



4. Codewheel rotates in positive direction, the light window just passes B phase a little, then turns to negative direction suddenly.



24.7.2 Sample Rate Configuration

On discussion about sample rate, suppose that:

Tab: minimum period of A phase or B phase

Tsamp: period of sample clock

Tsamp_db: period of debounce filter sample clock

Qclk: QDEC function clock

DIV: DIVIDE in SAMP_CTL register

DIV_DB: DB_SAMP_DIV in SAMP_CTL register

So, sample rate can be calculated as below:

Sample clock: Qclk/(2^DIV)

Debounce filter sample clock: Qclk/(2^DIV_DB)

Some limitations should be paid attention to with the consideration of the Tab. Generally, sample rate configuration should fulfill:

When debounce filters is disable:

Tab >= 4*Tsamp

When debounce filters is enable:

*Tab >= 4*Tsamp and Tab >= 8*Tsamp_db*

25. SPI Flash Interface (SPIFI)

25.1 Introduction

The SPI flash interface is available on all QN908x parts.

25.2 Features

- Quad SPI Flash Interface (SPIFI) to external flash.
- Transfer rates of up to SPIFI_CLK/2 bytes per second.
- Code in the serial flash memory can be executed as if it is in the internal memory space of the chip. This is accomplished by mapping the external flash memory directly into the same memory space.
- Supports 1-bit, 2-bit, and 4-bit bidirectional serial protocols.
- Half-duplex protocol compatible with various vendors and devices.
- A driver library available from NXP Semiconductors to assist in using the SPIFI.

Remark: The SPIFI software library with the SPIFI API are available on **www.nxp.com**.

25.3 Basic configuration

Initial configuration of the SPIFI peripheral is accomplished as follows:

1. SPIFI clock: to enable the SPIFI clock by writing 1 to CLK_SPIFI_EN (bit 22 of CLK_EN described in Section 2.5.4).
2. Pins: configure PA04~PA09 as SPIFI functions in PIO_FUNC_CFG0 and PIO_FUNC_CFG1 described in Section 2.5.28, for more information, please refer to pin mux table Table 95.

25.4 General description

The SPI Flash Interface (SPIFI) allows low-cost serial flash memories to be connected to the CPU with little performance penalty compared to parallel flash devices with higher pin count.

Many serial flash devices use a half-duplex command-driven SPI protocol for device setup and initialization. Quad devices then use a half-duplex, command-driven 4-bit protocol for normal operation. Different serial flash vendors and devices accept or require different commands and command formats. SPIFI provides sufficient flexibility to be compatible with common flash devices, and includes extensions to help insure compatibility with future devices.

Serial flash devices respond to commands sent by software or automatically sent by the SPIFI when software reads either of the two read-only serial flash regions in the memory map (see Table 396).

Table 396. SPIFI flash memory map

Memory	Address
SPIFI data	0x1000 0000 to 0x17FF FFFF **Remark:** This is the address space allocated to the SPIFI for direct execution. On this device, direct execution is not recommended, the SPIFI has been configured primarily for data access. The area allocated allows a maximum of 128 MB of SPI flash to be mapped into the CPU memory space. In practice, the usable space is limited to the size of the connected device.

25.5 Pin description

Table 397. SPIFI Pin description

Pin function	Type	Description
SPIFI_CLK	O	Serial clock for the flash memory, switched only during active bits on the MOSI/IO0, MISO/IO1, and IO3:2 lines.
SPIFI_CS	O	Chip select for the flash memory, driven low while a command is in progress, and high between commands. In the typical case of one serial slave, this signal can be connected directly to the device. If more than one serial slave is connected, software and off-chip hardware should use general-purpose I/O signals in combination with this signal to generate the chip selects for the various slaves.
SPIFI_MOSI or IO0	I/O	This is an output except in quad/dual input data fields. After a quad/dual input data field, it becomes an output again one serial clock period after CS goes high.
SPIFI_MISO or IO1	I/O	This is an output in quad/dual opcode, address, intermediate, and output data fields, and an input in SPI mode and in quad/dual input data fields. After an input data field in quad/dual mode, it becomes an output again one serial clock period after CS goes high.
SPIFI_SIO[3:2]	I/O	These are outputs in quad opcode, address, intermediate, and output data fields, and inputs in quad input data fields. If the flash memory does not have quad capability, these pins can be assigned to GPIO or other functions.

25.6 Supported devices

Multiple QSPI devices from various vendors can be used with the SPIFI interface and the SPIFI library available www.nxp.com.

25.7 SPIFI hardware

The SPIFI has a base address for the registers and a base address for the memory area in which the serial Flash connected to the SPIFI can be read.

The first operation with the serial Flash is Read JEDEC ID, which is implemented by most serial Flash devices. Depending on the device identity code returned by the serial Flash in this operation, device-specific commands are used for further operation. Programming and other operations on the serial Flash can be performed using a software library available on www.nxp.com. or using the register interface.

25.8 Register description

The SPIFI register interface supports word accesses.

Table 398. Register overview: SPIFI (base address 0x4008 0000)

Name	Access	Offset	Description	Reset value	Section
CTRL	R/W	0x000	SPIFI control register	0x400F FFFF	25.8.1
CMD	R/W	0x004	SPIFI command register	0x0	25.8.2
ADDR	R/W	0x008	SPIFI address register	0x0	25.8.3
IDATA	R/W	0x00C	SPIFI intermediate data register	0x0	25.8.4
CLIMIT	R/W	0x010	SPIFI limit register	0x0800 0000	25.8.5
DATA	R/W	0x014	SPIFI data register	0x0	25.8.6
MCMD	R/W	0x018	SPIFI memory command register	0x0	25.8.7
STAT	R/W	0x01C	SPIFI status register	0x0200 0000	25.8.8

25.8.1 SPIFI control register

The SPIFI control register controls the overall operation of the SPIFI and should be written before any commands are initiated.

Table 399. SPIFI control register (CTRL, offset 0x000) bit description

Bit	Symbol	Value	Description	Reset value
15:0	TIMEOUT	-	This field contains the number of serial clock periods without the processor reading data in memory mode, which will cause the SPIFI hardware to terminate the command by driving the CS pin high and negating the CMD bit in the Status register. (This allows the flash memory to enter a lower-power state.) If the processor reads data from the flash region after a time-out, the command in the Memory Command Register is issued again	0xFFFF
19:16	CSHIGH	-	This field controls the minimum $\overline{CS}$ high time, expressed as a number of serial clock periods minus one	0xF
20	-	-	Reserved	-
21	D_PRFTCH_DIS	-	This bit allows conditioning of memory mode prefetches based on the AHB HPROT (instruction/data) access information. A 1 in this register means that the SPIFI will not attempt a speculative prefetch when it encounters data accesses	0x0

Table 399. SPIFI control register (CTRL, offset 0x000) bit description

Bit	Symbol	Value	Description	Reset value
22	INTEN	-	If this bit is 1 when a command ends, the SPIFI will assert its interrupt request output. See INTRQ in the status register for further details	0x0
23	MODE3		SPI Mode 3 select	0x0
		0	SCK LOW. The SPIFI drives SCK low after the rising edge at which the last bit of each command is captured, and keeps it low while $\overline{CS}$ is HIGH	
		1	SCK HIGH. the SPIFI keeps SCK high after the rising edge for the last bit of each command and while $\overline{CS}$ is HIGH, and drives it low after it drives $\overline{CS}$ LOW. (Known serial flash devices can handle either mode, but some devices may require a particular mode for proper operation) **Remark:** MODE3, RFCLK, and FBCLK should not all be 1, because in this case there is no final falling edge on SCK on which to sample the last data bit of the frame.	
26:24	-	-	Reserved	-
27	PRFTCH_DIS		Cache prefetching enable. The SPIFI includes an internal cache. A 1 in this bit disables prefetching of cache lines.	0x0
		0	Enable. Cache prefetching enabled.	
		1	Disable. Disables prefetching of cache lines.	
28	DUAL		Select dual protocol.	0x0
		0	Quad protocol. This protocol uses IO3:0.	
		1	Dual protocol. This protocol uses IO1:0.	
29	RFCLK		Select active clock edge for input data.	0x0
		0	Rising edge. Read data is sampled on rising edges on the clock, as in classic SPI operation.	
		1	Falling edge. Read data is sampled on falling edges of the clock, allowing a full serial clock of of time in order to maximize the serial clock frequency. **Remark:** MODE3, RFCLK, and FBCLK should not all be 1, because in this case there is no final falling edge on SCK on which to sample the last data bit of the frame.	
30	FBCLK		Feedback clock select.	0x1
		0	Internal clock. The SPIFI samples read data using an internal clock.	
		1	Feedback clock. Read data is sampled using a feedback clock from the SCK pin. This allows slightly more time for each received bit. **Remark:** MODE3, RFCLK, and FBCLK should not all be 1, because in this case there is no final falling edge on SCK on which to sample the last data bit of the frame.	
31	DMAEN	-	A 1 in this bit enables the DMA Request output from the SPIFI. Set this bit only when a DMA channel is used to transfer data in peripheral mode. Do not set this bit when a DMA channel is used for memory-to-memory transfers from the SPIFI memory area. DMAEN should only be used in Command mode.	0x0

25.8.2 SPIFI command register

The Command Register may only be written as a word, but bytes, halfwords, and words may be read from it. It may be written to when the CMD and MCINIT bits in the Status register are 0, and under these circumstances writing initiates the transmission of a new command. For a command that contains an address and/or intermediate data, software should write to the Address and/or Intermediate Data registers, before writing to this

register. If the command contains output data, software should write it to the Data Register after writing to this register. If the command contains input data, software can read it from the Data Register after writing to this register.

Table 400. SPIFI command register (CMD, offset 0x004) bit description

Bit	Symbol	Value	Description	Reset value
13:0	DATALEN	-	Except when the POLL bit in this register is 1, this field controls how many data bytes are in the command. 0 indicates that the command does not contain a data field.	0x0
14	POLL	-	This bit should be written as 1 only with an opcode that a) contains an input data field, and b) causes the serial flash device to return byte status repetitively (e.g., a Read Status command). When this bit is 1, the SPIFI hardware continues to read bytes until the test specified by the DATALEN field is met. The hardware tests the bit in each status byte selected by DATALEN bits 2:0, until a bit is found that is equal to DATALEN bit 3. When the test succeeds, the SPIFI captures the byte that meets this test so that it can be read from the Data Register, and terminates the command by raising $\overline{CS}$. The end-of-command interrupt can be enabled to inform software when this occurs	0x0
15	DOUT		If the DATALEN field is not zero, this bit controls the direction of the data:	0x0
		0	Input from serial flash.	
		1	Output to serial flash.	
18:16	INTLEN	-	This field controls how many intermediate bytes precede the data. (Each such byte may require 8 or 2 SCK cycles, depending on whether the intermediate field is in serial, 2-bit, or 4-bit format.) Intermediate bytes are output by the SPIFI, and include post-address control information, dummy and delay bytes. See the description of the Intermediate Data register for the contents of such bytes.	0x0
20:19	FIELDFORM		This field controls how the fields of the command are sent.	0x0
		0x0	All serial. All fields of the command are serial.	
		0x1	Quad/dual data. Data field is quad/dual, other fields are serial.	
		0x2	Serial opcode. Opcode field is serial. Other fields are quad/dual.	
		0x3	All quad/dual. All fields of the command are in quad/dual format.	
23:21	FRAMEFORM		This field controls the opcode and address fields.	0x0
		0x0	Reserved.	
		0x1	Opcode. Opcode only, no address.	
		0x2	Opcode one byte. Opcode, least significant byte of address.	
		0x3	Opcode two bytes. Opcode, two least significant bytes of address.	
		0x4	Opcode three bytes. Opcode, three least significant bytes of address.	
		0x5	Opcode four bytes. Opcode, 4 bytes of address.	
		0x6	No opcode three bytes. No opcode, 3 least significant bytes of address.	
		0x7	No opcode four bytes. No opcode, 4 bytes of address.	
31:24	OPCODE	-	The opcode of the command (not used for some FRAMEFORM values).	0x0

25.8.3 SPIFI address register

Before writing a command that includes an address field to the Command register, software should write the address to this register. The most significant byte of the address is sent first.

Table 401. SPIFI address register (ADDR, offset 0x008) bit description

Bit	Symbol	Description	Reset value
31:0	ADDRESS	Address.	0x0

25.8.4 SPIFI intermediate data register

Before writing a command to the Command register that requires specific intermediate byte values, software should write the value of the bytes to this register. The least significant byte of this register is sent first. If more than four intermediate bytes are specified in the Command register, zeroes are sent after the 4th byte.

The main use of this register with current serial flash devices is to select the no-opcode mode (continuous read, code execution) using the byte value 0xA5, and cancelling this mode using 0xFF.

Many devices that require dummy (delay) bytes don't care about their contents, in which case this register need not be written.

Table 402. SPIFI intermediate data register (IDATA, offset 0x00C) bit description

Bit	Symbol	Description	Reset value
31:0	IDATA	Value of intermediate bytes.	0x0

25.8.5 SPIFI cache limit register

The SPIFI hardware includes caching of previously-accessed data to improve performance. Software can write an address within the device to this register, to prevent such caching at and above that address. After Reset this register contains the allocated size of the SPIFI memory area, so that all possible accesses are below that value and are thus cacheable.

Table 403. SPIFI cache limit register (CLIMIT, offset 0x010) bit description

Bit	Symbol	Description	Reset value
31:0	CLIMIT	Zero-based upper limit of cacheable memory	0x0800 0000

25.8.6 SPIFI data register

After initiating a command that includes a data output field by writing to the Command Register, software should write output data to this register. Store Byte instructions provide one data byte, Store Halfword instructions provide two bytes, and Store Word instructions provide 4 bytes of output data. Store commands are waited if the FIFO is too full to accept the number of bytes being stored. For Store Halfword and Store Word, the least significant byte is sent first.

After initiating a command that includes a data input field by writing to the Command Register, software should read input data from this register. Load Byte instructions deliver one data byte to software, Load Halfword instructions deliver two bytes, and Load Word instructions deliver 4 bytes of input data. Load commands are waited if a command is in progress and the FIFO does not contain the number of bytes being loaded. For Load Halfword and Load Word commands, the least significant byte is received first.

DATALEN bytes should be read from or written to this register. If such a (read or write) command needs to be terminated before that time, software should write a 1 to the RESET bit in the Status register to accomplish this. If software attempts to read or write more data than was specified in DATALEN, a Data Abort exception will occur.

In polling mode (see the POLL bit in the SPIFI command register), one byte must be read from this register because the poll mechanism writes the matching byte.

This register is not used for commands initiated by reading the flash address range in the memory map. In DMA transfers in peripheral to-or-from-memory mode, the address of this register should be used as the peripheral address.

Table 404. SPIFI Data register (DATA, offset 0x014) bit description

Bit	Symbol	Description	Reset value
31:0	DATA	Input or output data	0x0

25.8.7 SPIFI memory command register

Before accessing the flash area of the memory map, software should set up the device. After optionally writing to the Intermediate Data register, software should write a word to this register to define the command that is used to read data. Thereafter data can be read from the flash memory area, either directly or by means of a DMA channel.

Writing to this register will be ignored when a command is in progress or while data has yet to be written or read from the FIFO for a command issued. Use the MCINIT bit of the Status register to verify that the hardware is in Memory mode. A successful write to this register sets the SPIFI into Memory mode. The content of this register is identical to that of the Command Register, except for the DATALEN (not used), POLL, DOUT, and FRAMEFORM bits.

Table 405. SPIFI memory command register (MCMD, offset 0x018) bit description

Bit	Symbol	Value	Description	Reset value
13:0	-	-	Reserved.	0x0
14	POLL	-	This bit should be written as 0.	0x0
15	DOUT	-	This bit should be written as 0.	0x0
18:16	INTLEN	-	This field controls how many intermediate bytes precede the data. (Each such byte may require 8 or 2 SCK cycles, depending on whether the intermediate field is in serial, 2-bit, or 4-bit format.) Intermediate bytes are output by the SPIFI, and include post-address control information, dummy and delay bytes. See the description of the Intermediate Data register for the contents of such bytes.	0x0
20:19	FIELDFORM		This field controls how the fields of the command are sent.	0x0
		0x0	All serial. All fields of the command are serial.	
		0x1	Quad/dual data. Data field is quad/dual, other fields are serial.	
		0x2	Serial opcode. Opcode field is serial. Other fields are quad/dual.	
		0x3	All quad/dual. All fields of the command are in quad/dual format.	

Table 405. SPIFI memory command register (MCMD, offset 0x018) bit description

Bit	Symbol	Value	Description	Reset value
23:21	FRAMEFORM		This field controls the opcode and address fields.	0x0
		0x0	Reserved.	
		0x1	Opcode. Opcode only, no address.	
		0x2	Opcode one byte. Opcode, least-significant byte of address.	
		0x3	Opcode two bytes. Opcode, 2 least-significant bytes of address.	
		0x4	Opcode three bytes. Opcode, 3 least-significant bytes of address.	
		0x5	Opcode four bytes. Opcode, 4 bytes of address.	
		0x6	No opcode three bytes. No opcode, 3 least-significant bytes of address.	
		0x7	No opcode, 4 bytes of address.	
31:24	OPCODE	-	The opcode of the command (not used for some FRAMEFORM values).	0x0

25.8.8 SPIFI status register

This register indicates the state of the SPIFI.

Table 406. SPIFI status register (STAT, offset 0x01C) bit description

Bit	Symbol	Description	Reset value
0	MCINIT	This bit is set when software successfully writes the Memory Command register, and is cleared by Reset or by writing a 1 to the RESET bit in this register.	0x0
1	CMD	This bit is 1 when the Command register is written. It is cleared by a hardware reset, a write to the RESET bit in this register, or the deassertion of $\overline{CS}$ which indicates that the command has completed communication with the SPI Flash.	0x0
3:2	-	Reserved	-
4	RESET	Write a 1 to this bit to abort a current command or memory mode. This bit is cleared when the hardware is ready for a new command to be written to the Command register.	
5	INTRQ	This bit reflects the SPIFI interrupt request. Write a 1 to this bit to clear it. This bit is set when a CMD was previously 1 and has been cleared due to the deassertion of $\overline{CS}$.	0x0
31:6	-	Reserved	-

25.9 Functional description

25.9.1 Data transfer

Serial SPI uses the signals SPIFI_CLK, $\overline{\text{SPIFI\_CS}}$,SPIFI_MISO, and SPIFI_MISO, while quad mode adds the two IO signals SPIFI_SIO[3:2].

The SPIFI implements basic, dual, and quad SPI in half-duplex mode, in which the SPIFI always sends a command to a serial flash memory at the start of each frame. (A frame is the sequence of bytes transmitted during one period with $\overline{\text{CS}}$ LOW.) In general, commands start with an opcode byte although some serial flashes allow a no-opcode mode in which commands start with the address to be read. In write commands, the SPIFI sends all of the data in the frame, while in read commands, the SPIFI sends the command, and then the serial flash sends data to the SPIFI.

Classic SPI includes four modes (mode 0 to mode 3), of which the SPIFI and most serial flashes implement modes 0 and 3. In mode 0, the SCK line is LOW between frames while in mode 3 it is HIGH. In mode 0, the SPIFI drives the first data bits from the time that it drives $\overline{\text{CS}}$ LOW, and drives the rest of the data on falling edges of SCK. In mode 3, the SPIFI drives SCK LOW one-half clock period after it drives $\overline{\text{CS}}$ LOW, and drives data on the falling edge of SCK. In either mode the serial flash samples the data on the rising edges of SCK.

The same scheme (transmitter changes data on falling edges of SCK, receiver samples data on rising edges) is maintained for the entire frame, including read data sent by the serial flash to the SPIFI.

The SPI protocol avoids all issues of set-up and hold times between the clock and data lines by using half of the SCK period to transmit the data. For high clock speeds, it is necessary to sample read data using a feedback clock. The FBCLK bit enables the feedback clock from the SCK pad sampling method. This provides the best possible timing margin for both read and write data under the opposite-edge scheme.

But maximizing clock frequency is of such importance that further improvement is sometimes needed, by means of using the whole serial clock period to transmit data. This choice is enabled for read data by setting the RFCLK bit. When this bit is 1, the SPIFI samples data on the falling edge of the serial clock that follows the rising edge which is normally used. RFCLK and FBCLK and MODE3 should not all be 1 because in this case there would be no falling edge of the feedback clock to capture the last bit of a frame.

Consult the data sheet of the serial flash device to be used for the formats of the commands that it supports. Figure 75 shows commands consisting of an opcode field only, sent in SPI and quad modes. All fields are multiples of 8 bits long. Bytes are sent with the most significant bit first in SPI mode, and the most significant 4 bits first in quad mode.

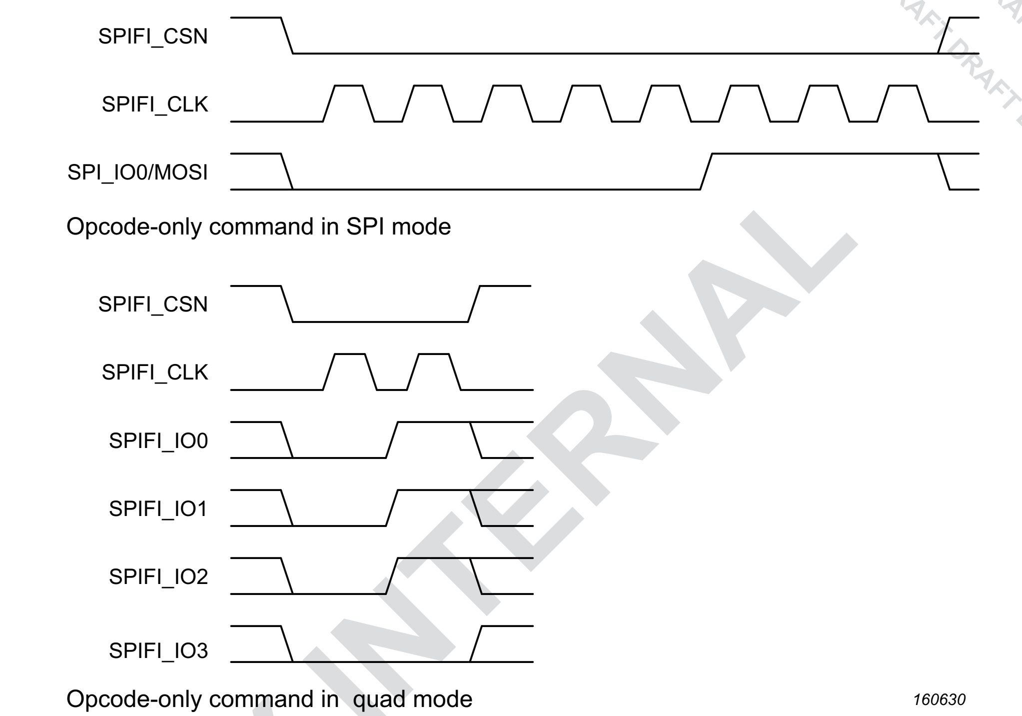


Fig 75. Opcode only commands

Figure 76 shows a command that reads 1 byte from the slave in SPI mode and a command that reads 3 bytes from the slave with the opcode and input data fields both in quad mode.

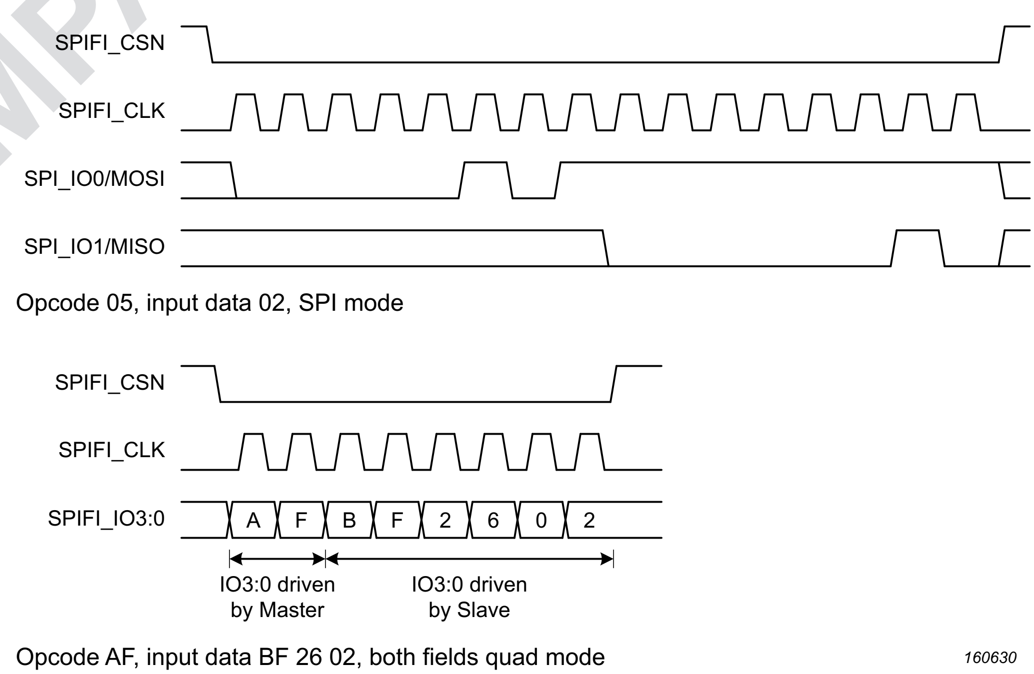


Fig 76. Read commands

In quad mode, the IO3:0 lines are driven by the SPIFI in opcode, address, intermediate and output data fields, and driven by the flash memory in input data fields. In address fields the more significant bytes are sent first.

25.9.2 Software requirements and capabilities

During device set-up, software should initialize the external serial flash device using those commands that place it in its highest-performance mode. When this sequence is complete, software should write the command that will be issued in response to a read from the serial flash region of the memory map, to the Memory Command Register. If software attempts to read the flash region after Reset, power-up, or writing the Command Register without writing the Memory Command Register thereafter, the SPIFI responds with an Abort error.

After writing the Memory Command Register, the contents of the flash would appear to the system as memory mapped. This enables data access or execution from the serial flash over AHB.

SPIFI has two operational modes:

1. Memory Mode - whereby the contents of the FLASH are memory mapped in the chip.
2. Command Mode - whereby the user can manually construct command sequences for the flash.

SPIFI cannot switch over from Memory Mode to Command mode and vice versa without writing 1 to the RESET bit in the SPIFI Status Register and polling until it is cleared by hardware to ensure that the current mode has been aborted.

The SPIFI includes a cache to maximize performance for accesses to the serial flash region of the memory map. The cache is only used in Memory mode and can be disabled.

Because the SPIFI is an AHB device, software or a DMA channel can read bytes, halfwords, or words from the flash region.

Reads from the flash region are delayed by deasserting HREADY when necessary, until the requested bytes are available to be read.

In Memory mode, SPIFI prefetches sequential addresses in order to improve performance.

If no AHB accesses have taken place for a period specified by the time-out (TO) field in the Control register, the SPIFI will deassert $\overline{CS}$. Once a new access occurs that requires a new fetch of data, the SPIFI will reassert $\overline{CS}$ and send a new command to fetch the required data. This is done in order to save power in the SPI flash device.

If software reads or writes more data from the Data Register than was configured in the DATALEN field of the Command register or reads or writes when no command was issued, the SPIFI hardware issues an abort exception.

When the serial flash needs to be programmed or erased, software should not write to the flash region of the address map. Instead, it should write the appropriate sequence of commands to the Command, Address, and Data registers. When an actual erase or program operation is under way in the serial flash device, software should write a Read Status command (with the POLL bit set) to the Command register. Thereafter:

- If INTEN in the Control register is 1, the SPIFI will interrupt the processor when the erase or write operation (and thus the Read Status command) completes.
- If not, software can continually or periodically read the Status register until it indicates that the Read Status command is complete.

When erasing or programming completes, software can do further programming or erasing, or return to normal (memory mode) operation.

25.9.3 Peripheral mode DMA operation

The SPIFI inserts wait states when necessary during read and write operations by the core to maintain synchronization between core accesses and serial data transfer with the serial flash. This mechanism is all that is needed for load and store accesses and for memory-to-memory transfers by a DMA channel.

The peripheral mode is a mode that supports DMA transfers in which the SPIFI acts as a peripheral and drives a request signal to the DMA channel to control data transfer. This mode does not necessarily move data faster than memory-to-memory operation, but it may be advantageous in systems in which software controls dynamic transfer of code and/or data between the serial flash and RAM on an as-needed basis. The advantage is that clock cycles are not lost to wait states, and thus the overall operation of the AHB is more efficient.

The DMA controller should be programmed to present word operations at the fixed address of the Data Register to have a burst size of one transfer. The SPIFI drives the DMA request to the DMA controller.

To use this mode, software should write the Command register to start the command and program a DMA channel as described above to transfer data between the Data register and RAM. The SPIFI asserts the DMA request when:

- DMAEN in the Control register is 1.
- MCINIT is 0.
- There are at least 4 bytes in the FIFO for a read operation, or at least 4 empty byte locations in the FIFO for a write/program operation.

26. Random Number generator (RNG)

26.1 Introduction

RNG functionality is available on all QN908x devices.

26.2 Features

- Collects the ADC lsb data to generate a random data.
- Supports programmed 8-, 16- or 32-bit width.
- Real random number generation.
- Supports RNG interrupt when random data generation is done, the interrupt can be enabled or disabled.

26.3 Basic configuration

Initial configuration of the RNG peripheral is accomplished as follows:

- ADC setting: refer to Section 30.7.1 in ADC.
- Enable RNG peripheral by writing 1 to ENABLE bit in CTRL register as Section 26.5.1
- Set random data bit width by configuring NUM bit in CTRL register as Section 26.5.1
- If needed, enable RNG interrupt by configuring DONE_INTEN bit in INTEN register as Section 26.5.5, and in NVIC.

26.4 General Description

The block diagram of RNG is shown as Figure 77.

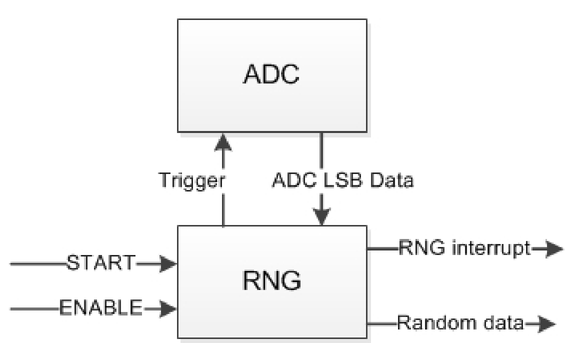


Fig 77. RNG block diagram

26.4.1 RNG Start

Write the START bit as 1 in CTRL register to start RNG. After the generator starts, this bit is cleared automatically. The RNG module generates the RNG trigger signal to start the ADC simultaneously and goes to busy status for collecting the ADC lsb data until the DONE is generated.

Before passing trigger to ADC, correct channel source number should be selected in ADC to get random data. For more information, refer ADC user manual.

Remark: The ENABLE bit of CTRL register (Section 30.6.1) of ADC should be write 1 to enable ADC earlier at lease one ADC clock than RNG start.

26.4.2 RNG Done Interrupt

After random data is generated and saved in DATA register (Section 26.5.3), it also generates a RNG done interrupt signal DONE for the NVIC with interrupt vector number 38. It can be enabled or disabled by writing DONE_INTEN to 1/0.

26.4.3 RNG DATA

After RNG done interrupt occurs, random data with configured width is prepared and it can be read from register DATA.

Maximum ADC data rate: 31.25 kbps/32-bit word: 1000 words.

The best expected result 1 ms generate one 32-bit random number. If we need to generate 1000 32-bit random number, it will take 1 s.

26.5 Register Description

Table 407. RNG registers overview (base address 0x4000 7C00)

Name	Access	Offset	Description	Reset value	Section
CTRL	RW	0x00	RNG control register	0x0	26.5.1
STAT	R	0x04	RNG status register	0x0	26.5.2
DATA	R	0x08	Random data output register	0x0	26.5.3
INT	RW	0x0C	Interrupt register	0x0	26.5.4
INTEN	RW	0x10	Interrupt mask register	0x0	26.5.5

26.5.1 RNG control register

Table 408. RNG control register (CTRL, offset=0x00) bit description

Bit	Symbol	Description	Reset Value	Access
0	ENABLE	RNG enable. 0:RNG disabled. 1:RNG enabled.	0x0	RW
0	START	Write 1 to start random number generation, auto clear	0x0	W1
3:2	-	Reserved	-	-
5:4	NUM	RNG data bit width. 00b: 8bits, DATA[7:0] is valid random data; 01b: 16bits, DATA[15:0] is valid random data; 11b: 32bits, DATA[31:0] is valid random data	0x0	RW
31:6	-	Reserved	-	-

26.5.2 RNG status register

Table 409. RNG status register (STAT, offset=0x04) bit description

Bit	Symbol	Description	Reset Value	Access
0	BUSY	it indicates the module is in processing; 1: RNG is busy in processing; 0: RNG is idle.	0x0	R
31:1	-	Reserved	-	-

26.5.3 RNG Output Data register

Table 410. RNG Output Data register (DATA, offset=0x08) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	DATA	final random data read by SW	0x0	R

26.5.4 Interrupt Status register

Table 411. Interrupt Status register (INT, offset=0x0C) bit description

Bit	Symbol	Description	Reset Value	Access
0	DONE	random data generate done.	0x0	W1C
31:1	-	Reserved	-	-

26.5.5 Interrupt Enable register

Table 412. Interrupt Enable register (INTEN, offset=0x10) bit description

Bit	Symbol	Description	Reset Value	Access
0	DONE_INTEN	random data generate done mask. 1= enable RNG done interrupt; 0: disable RNG done interrupt.	0x0	rw
31:1	-	Reserved	-	-

27. CRC engine

27.1 Introduction

The CRC engine is available on all QN908x parts.

27.2 Features

- Supports three common polynomials CRC-CCITT, CRC-16, and CRC-32.
  - CRC-CCITT: $x^{16} + x^{12} + x^5 + 1$
  - CRC-16: $x^{16} + x^{15} + x^2 + 1$
  - CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- Bit order reverse and 1's complement programmable setting for input data and CRC sum.
- Programmable seed number setting.
- Supports CPU PIO back-to-back transfer.
- Accept any size of data width per write: 8, 16 or 32-bit.
  - 8-bit write: 1-cycle operation
  - 16-bit write: 2-cycle operation (8-bit x 2-cycle)
  - 32-bit write: 4-cycle operation (8-bit x 4-cycle)

27.3 Basic configuration

Writing the CLK_FSP_EN bit in the CLK_EN register (Section 2.5.4) in SYSCON to enable the clock to the CRC engine.

27.4 Pin description

The CRC engine has no configurable pins.

27.5 General description

The Cyclic Redundancy Check (CRC) generator with programmable polynomial settings supports several CRC standards commonly used.

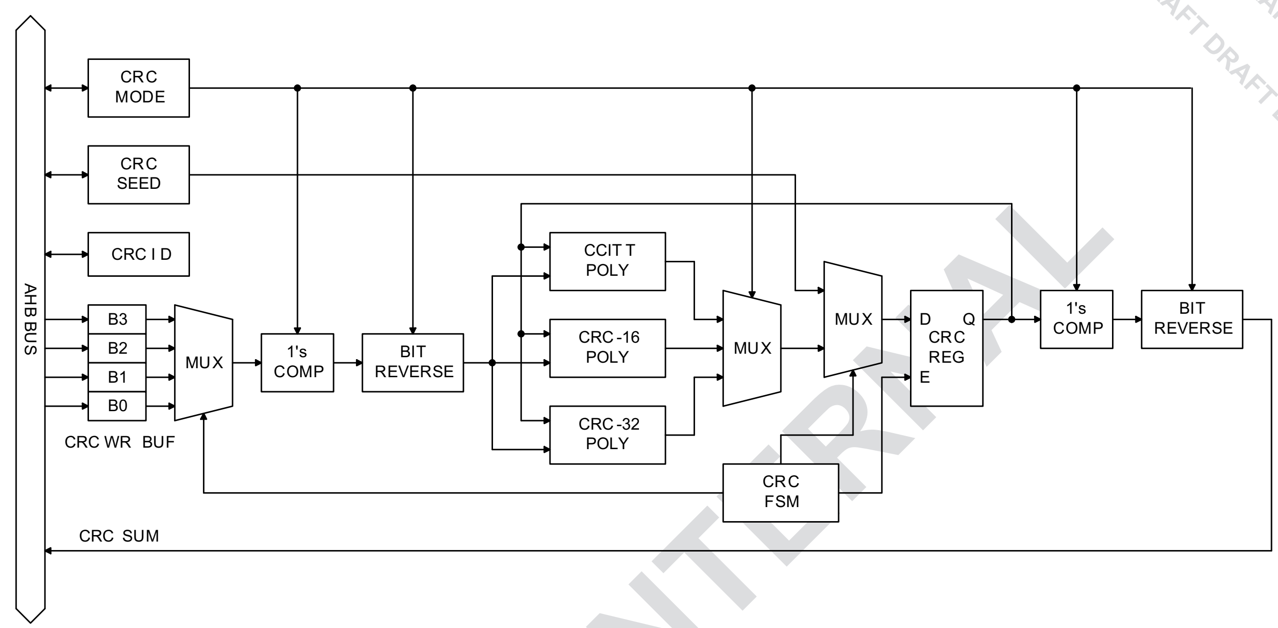


Fig 78. CRC block diagram

27.6 Register description

Table 413. Register overview: CRC engine (base address 0x4009 5000)

Name	Access	Offset	Description	Reset value	Section
MODE	R/W	0x00	CRC mode register	0x0	27.6.1
SEED	R/W	0x04	CRC seed register	0xFFFF	27.6.2
SUM	RO	0x08	CRC checksum register	0xFFFF	27.6.3
WR_DATA	WO	0x08	CRC data register	-	27.6.4

27.6.1 CRC mode register

Table 414. CRC mode register (MODE, offset 0x00) bit description

Bit	Symbol	Description	Reset value
1:0	CRC_POLY	CRC polynomial: 1X: CRC-32 polynomial 01: CRC-16 polynomial 00: CRC-CCITT polynomial	0x0
2	BIT_RVS_WR	Data bit order: 1: Bit order reverse for CRC_WR_DATA (per byte) 0: No bit order reverse for CRC_WR_DATA (per byte)	0x0
3	CMPL_WR	Data complement: 1: 1's complement for CRC_WR_DATA 0: No 1's complement for CRC_WR_DATA	0x0

Table 414. CRC mode register (MODE, offset 0x00) bit description

Bit	Symbol	Description	Reset value
4	BIT_RVS_SUM	CRC sum bit order: 1: Bit order reverse for CRC_SUM 0: No bit order reverse for CRC_SUM	0x0
5	CMPL_SUM	CRC sum complement: 1: 1's complement for CRC_SUM 0: No 1's complement for CRC_SUM	0x0
31:6	-	Reserved. Always 0 when read	-

27.6.2 CRC seed register

Table 415. CRC seed register (SEED, offset 0x04) bit description

Bit	Symbol	Description	Reset value
31:0	CRC_SEED	A write access to this register will load the CRC seed value the SUM register with selected bit order and 1's complement pre-processes. **Remark:** A write access to this register will overrule the CRC calculation in progress.	0xFFFF

27.6.3 CRC checksum register

This register is a Read-only register containing the most recent checksum.

Table 416. CRC checksum register (SUM, offset 0x08) bit description

Bit	Symbol	Description	Reset value
31:0	CRC_SUM	The most recent CRC sum can be read through this register with selected bit order and 1's complement post-processes.	0x0000 FFFF

27.6.4 CRC data register

This register is a Write-only register containing the data block for which the CRC sum will be calculated.

Table 417. CRC data register (WR_DATA, offset 0x08) bit description

Bit	Symbol	Description	Reset value
31:0	CRC_WR_DATA	Data written to this register will be taken to perform CRC calculation with selected bit order and 1's complement pre-process. Any write size 8, 16 or 32-bit are allowed and accept back-to-back transactions.	-

27.7 Functional description

27.7.1 Timing

The CRC engine uses some time to process data, which can depend on how it is accessed.

A write followed by another write:

For a 16-bit write to the CRC data register followed by another write to the same register, there is 1 wait state added.

For a 32-bit write to the CRC data register followed by another write to the same register, there are 3 wait states added.

A write followed by a read:

For an 8-bit write to the CRC data register followed by a read of the CRC checksum register, there is 1 wait state added.

For a 16-bit write to the CRC data register followed by a read of the CRC checksum register, there are 2 wait states added.

For a 32-bit write to the CRC data register followed by a read of the CRC checksum register, there are 4 wait states added.

27.7.2 Setup

The following sections describe the register settings for each supported CRC standard:

CRC-CCITT set-up

Polynomial: $x^{16} + x^{12} + x^{5} + 1$

Seed Value: 0xFFFF

Bit order reverse for data input: NO

1's complement for data input: NO

Bit order reverse for CRC sum: NO

1's complement for CRC sum: NO

CRC_MODE: 0x0000 0000

CRC_SEED: 0x0000 FFFF

CRC-16 set-up

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

Seed Value: 0x0000

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_MODE: 0x0000 0036

CRC_SEED: 0xFFFF FFFF

28. Flash memory controller

28.1 Introduction

The QN908x contains up to 512 KB of internal flash, which is formed by two contiguous standalone areas known as block 0 and block 1. Both the MCU and DMA controller are able to access the two areas independently without any conflict.

Remark: In addition to the ISP commands, the flash related registers can be used to configure flash controller.

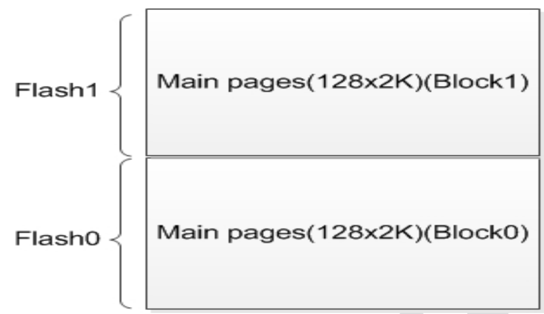
28.2 Features

- The 512 KB of flash in the QN908x is divided into two blocks. Each flash block is made up of 2 KB flash pages.
- The flash can be read in byte, half word, or word sizes.
- The flash supports two write modes: Normal Write mode and HW Smart Write mode.
  - Normal Write: Write a word directly to the flash address.
  - HW Smart Write: After every word write operation, flash controller will read back the value from the address it just wrote to and check whether it is same as the value written. If it is the expected value, the write operation is considered complete. If the value is different, the flash controller will try to write the value to the address again. This write-read-check process will repeat until the read value is correct or the number of retries reaches the value set in the MAX_WRITE bit field in the SMART_CTRL register. If the value is still not correct after repeating the write operation MAX_WRITE number of times, then the write operation is considered failed. A software implementation can be done in the application to accomplish the same write-read-check process.When implementing a software Smart Write, the flash region 0x3000 0000 ~ 0x3007 FFFF should be used for the flash read accesses.
- The flash can be erased page by page or block or block. These erase operations have two modes: Normal Erase and Smart Erase.
  - Normal Erase: Erase the flash without checking if the flash was really erased.
  - Smart Erase: After erasing the flash, the flash controller will read back the values from the erased addresses. If the flash is not erased, the flash controller will try to erase the flash addresses again and will continue to retry until the number of retries reach the value set in the MAX_ERASE bit field in the SMART_CTRL register. If the value is still not correct after repeating the erase operation MAX_ERASE number of times, then the erase operation is considered failed.

28.3 General description

28.3.1 Flash physical organization

The QN908x flash memory is made up of 256 KB blocks, as shown as Figure 79.



Remark: If the product only contains 256KB on-chip flash, there is only Flash1, not Flash0.

Fig 79. Flash physical organization

The main pages of the QN908x consist of 2KB flash page, which can be used to store code or user data. The last 5 pages in flash Block 1 are reserved for system and user information, see Figure 12.

28.3.2 Flash lock and protection

The last page of block 1 contains lock and protection bits to enable a level of security in the QN908x memory. When these lock and protection bits are changed, the new lock and protection scheme is not immediately in effect. The INI_RD_EN bit in the flash initial read enable register must be set to '1' in order for the new lock and protection scheme to be updated in the flash controller. Below table describes the lock and protection bits.

Table 418. Flash lock and protect description

Address offset from the start of flash	Lock and protect information
0x7F800 - 0x7F80F	Flash block 0 page lock configuration. When a flash page is locked, the contents cannot be erased or written to by a normal page erase command. Bit 0 controls the lock for page 0, bit 1 controls the lock for page 1, and so on and so forth. 0: The flash page is locked. 1: The flash page is unlocked.
0x7F810 - 0x7F81F	Flash block 1 page lock configuration. When a flash page is locked, the contents cannot be erased or written to by a normal page erase command. Bit 0 controls the lock for page 0, bit 1 controls the lock for page 1, and so on and so forth. 0: The flash page is locked. 1: The flash page is unlocked.
0x7F820	Bit 0: Mass erase lock bit. When this bit is set to “0”, a normal mass erase command cannot erase the flash. 0: Mass erase lock is active. 1: Mass erase lock is not active. Bit 1: Flash protection bit. When this bit is set, the flash cannot be read, written to, or erased by SWD. 0: Flash protection is not active. 1: Flash protection is active. Bit 2: Memory protection bit. When this bit is set, the SRAM cannot be read, written to, or erased by SWD. 0: Memory protection is not active. 1: Memory protection is active.

In addition to these lock and protection bits, there exists password registers for the SWD and software that must be used to issue a specific erase command. This is done by programming a unique value into the password registers, which will then allow the next erase operation to be completed if the erase command would normally be disallowed by the current protect or lock configuration. After the erase is complete, the password register is then cleared.

Table 419 and Table 420 summarize what erase, write, and read permissions are available when using different lock and protection schemes.

Table 419. SWD behavior for different lock and protect scheme

Flash protect bit	Mass erase lock bit	page erase lock bit	Mass erase	Page erase	Write flash	Read flash
0	0	0	*	×	×	√
0	0	1	*	√	√	√
0	1	0	√	×	×	√
0	1	1	√	√	√	√
1	x	x	*	×	×	×

Table 420. Firmware behavior for different lock and protect scheme

Flash protect bit	Mass erase lock bit	page erase lock bit	Mass erase	Page erase	Write flash	Read flash
x	0	0	**	**	×	√
x	0	1	**	√	√	√
x	1	0	√	**	×	√
x	1	1	√	√	√	√

Remark: [1]*Mass erase through SWD is possible if the DEBUG_PASSWORD register is used.

Remark: [2]**Erase is possible if the ERASE_PASSWORD register is used to bypass the mass erase and page erase lock bit.

28.4 Register description

Table 421. Flash Registers overview (base address 0x4008 1000)

Name	Access	Offset	Description	Reset value	Section
INI_RD_EN	RW	0x0	Flash initial read enable register. Use this register to update the flash lock and protection settings.	0x0	28.4.1
ERASE_CTRL	RW	0x4	Flash erase control register	0x0	28.4.2
ERASE_TIME	RW	0x8	Flash erase time setting register	0x9C400	28.4.3
TIME_CTRL	RW	0xC	Flash operation time setting register	0x4001E	28.4.4
SMART_CTRL	RW	0x10	Smart control register	0x553C	28.4.5
INT_EN	RW	0x14	Interrupt enable register	0x0	28.4.6
INT_STAT	R	0x18	Interrupt status register	0x0	28.4.7
INT_CLR	W	0x1C	Interrupt clear register	0x0	28.4.8
LOCK_STAT_0	R	0x20	Lock status register 0	0xFFFFFFFF	28.4.9
LOCK_STAT_1	R	0x24	Lock status register 1	0xFFFFFFFF	28.4.9
LOCK_STAT_2	R	0x28	Lock status register 2	0xFFFFFFFF	28.4.9
LOCK_STAT_3	R	0x2C	Lock status register 3	0xFFFFFFFF	28.4.9
LOCK_STAT_4	R	0x30	Lock status register 4	0xFFFFFFFF	28.4.9
LOCK_STAT_5	R	0x34	Lock status register 5	0xFFFFFFFF	28.4.9
LOCK_STAT_6	R	0x38	Lock status register 6	0xFFFFFFFF	28.4.9
LOCK_STAT_7	R	0x3C	Lock status register 7	0xFFFFFFFF	28.4.9
LOCK_STAT_8	R	0x40	Lock status register 8	0xF	28.4.10
STATUS1	R	0x48	Status register 1	0x4010001	28.4.11
ERR_INFOL1	R	0x5C	Flash block 0 error information register 1	0x0	28.4.12
ERR_INFOL2	R	0x60	Flash block 0 error information register 2	0x0	28.4.13
ERR_INFOL3	R	0x64	Flash block 0 error information register 3	0x0	28.4.14
ERR_INFOH1	R	0x68	Flash block 1 error information register 1	0x0	28.4.15

Table 421. Flash Registers overview (base address 0x4008 1000) *…continued*

Name	Access	Offset	Description	Reset value	Section
ERR_INFOH2	R	0x6C	Flash block 1 error information register 2	0x0	28.4.16
ERR_INFOH3	R	0x70	Flash block 1 error information register 3	0x0	28.4.17
DEBUG_PASSWORD	RW	0xA8	Debug port access password register	0x0	28.4.18
ERASE_PASSWORD	RW	0xAC	Erase password register	0x0	28.4.19

28.4.1 Flash initial read enable register

Table 422. Flash initial read enable register (INI_RD_EN) bit description

Bit	Symbol	Description	Reset Value	Access
0	INI_RD_EN	Set this bit to ‘1’ to update the flash controller’s lock and protection scheme.	0x0	RW
31:1	-	Reserved	-	-

28.4.2 Flash erase control register

Table 423. Flash erase control register (ERASE_CTRL) bit description

Bit	Symbol	Description	Reset Value	Access
6:0	PAGE_INDXL	The Block 0 page erase index.	0x0	RW
7	-	Reserved	-	-
14:8	PAGE_INDXH	The Block 1 page erase index.	0x0	RW
27:15	-	Reserved	-	-
28	HALF_ERASEL_EN	Write ’1’ to Enable Mass Erase for Block 0. After this bit is set by software, the hardware will clear this bit at the end of the Block 0 flash mass erase operation.	0x0	RW
29	HALF_ERASEH_EN	Write ’1’ to Enable Mass Erase for Block 1. After this bit is set by software, the hardware will clear this bit at the end of the Block 1 flash mass erase operation.	0x0	RW
30	PAGE_ERASEL_EN	Block 0page erase enable. When this bit is set, a page erase operation is initiated on the page index set in the ERASE_CTRL register. After this bit is set by software, the hardware will clear this bit at the end of page erase operation.	0x0	RW
31	PAGE_ERASEH_EN	Block 1 page erase enable. When this bit is set, a page erase operation is initiated on the page index set in the ERASE_CTRL register. After this bit is set by software, the hardware will clear this bit at the end of page erase operation.	0x0	RW

28.4.3 Flash erase time setting register

For page erase operations, the flash controller uses an internal 8 MHz clock source. The flash erase time setting register indicates the number of 8 MHz clock cycles for the flash erase time. The reset value configures the erase time to 80 ms. It is not recommended to change the erase time to values not mentioned in this section.

Table 424. Flash erase time setting register (ERASE_TIME) bit description

Bit	Symbol	Description	Reset Value	Access
19:0	ERASE_TIME_BASE	The flash controller uses an internal 8 MHz clock for erase operation. The value in this register indicates the number of 8 MHz clock cycles to use for the erase time. For page erase, this register should be configured to set an erase time of 80ms while a Smart Erase operation should use 2ms. The reset value sets the erase time to 80ms.	0x9C400	RW
31:20	-	Reserved	-	-

28.4.4 Flash operation time setting register

Table 425. Flash operation time setting register (TIME_CTRL) bit description

Bit	Symbol	Description	Reset Value	Access
11:0	PRGM_CYCLE	Indicates the maximum number of write operations the flash controller can do in one flash page program. This applies to normal writes or software Smart Writes. Valid values are 2 through 64.	0x1E	RW
19:12	TIME_BASE	The flash controller uses timing parameters based on 2 microsecond periods. This bit field should contain the amount of AHB clock cycles equal to 2 us. The reset value is based on an AHB clock of 32 MHz.	0x40	RW
31:20	-	Reserved	-	-

28.4.5 Smart erase control register

Table 426. Smart erase control register (SMART_CTRL)bit description

Bit	Symbol	Description	Reset Value	Access
0	PRGML_EN	Flash Block 0 write enable. Set this bit to enable write operation for Block 0.	0x0	RW
1	PRGMH_EN	Flash Block 1 write enable. Set this bit to enable write operation for Block 1.	0x0	RW
2	SMART_WRITEL_EN	Flash Block 0 Smart Write enable. Set this bit to enable hardwareSmart Write operation for Block 0.	0x1	RW
3	SMART_WRITEH_EN	Flash Block 1 Smart Write enable. Set this bit to enable hardware Smart Write operation for Block 1.	0x1	RW
4	SMART_ERASEL_EN	Flash Block 0 Smart Erase enable. Set this bit to enable Smart Erase operation for Block 0.	0x1	RW
5	SMART_ERASEH_EN	Flash Block 1 Smart Erase enable. Set this bit to enable Smart Erase operation for Block 1.	0x1	RW
7:6	-	Reserved	-	-

Table 426. Smart erase control register (SMART_CTRL)bit description ...*continued*

Bit	Symbol	Description	Reset Value	Access
11:8	MAX_WRITE	The maximum number of write attempts for one Smart Write operation. The actual interpretation by the hardware is such that the value programmed here plus one is used. During an ongoing write command, this bit field cannot be changed.	0x5	RW
17:12	MAX_ERASE	The maximum number of erase attempts for one Smart Erase operation. The actual interpretation by the hardware is such that the value programmed here plus one is used. The recommended value is 100/(ERASE_TIME_BASE). During an ongoing write command, this bit field cannot be changed.	0x5	RW
31:18	-	Reserved	-	-

28.4.6 Interrupt enable register

Table 427. Interrupt enable register (INTEN) bit description

Bit	Symbol	Description	Reset Value	Access
0	AHBL_INT_EN	Write '1' to enable Block 0 AHB error interrupt.	0x0	RW
1	LOCKL_INT_EN	Write '1' to enable Block 0 lock error interrupt.	0x0	RW
2	ERASEL_INT_EN	Write '1' to enable Block 0 erase status interrupt.	0x0	RW
3	WRITEL_INT_EN	Write '1' to enable Block 0 write status interrupt.	0x0	RW
4	WR_BUFL_INT_EN	Write '1' to enable Block 0 write buffer status interrupt.	0x0	RW
7:5	-	Reserved	-	-
8	AHBH_INT_EN	Write '1' to enable Block 1 AHB error interrupt.	0x0	RW
9	LOCKH_INT_EN	Write '1' to enable Block 1 lock error interrupt.	0x0	RW
10	ERASEH_INT_EN	Write '1' to enable Block 1 erase status interrupt.	0x0	RW
11	WRITEH_INT_EN	Write '1' to enable Block 1 write status interrupt.	0x0	RW

Table 427. Interrupt enable register (INTEN) bit description *...continued*

Bit	Symbol	Description	Reset Value	Access
12	WR_BUFH_INT_EN	Write '1' to enable Block 1 write buffer status interrupt.	0x0	RW
30:13	-	Reserved	0x0	-
31	EMBFSH_INT_EN	Write '1' to enable flash interrupts.	0x0	RW

28.4.7 Interrupt status register

Table 428. Interrupt status register (INT_STAT) bit description

Bit	Symbol	Description	Reset Value	Access
0	AHBL_INT	Flash Block 0 AHB error.	0x0	R
1	LOCKL_INT	Flash Block 0 locked page being accessed error.	0x0	R
2	ERASEL_INT	Flash Block 0 erase operation completed.	0x0	R
3	WRITEL_INT	Flash Block 0 write operation completed.	0x0	R
4	WR_BUFL_INT	Flash Block 0 write buffer empty.	0x0	R
5	WRITE_FAIL_L_INT	Smart Write on flash Block 0 failed.	0x0	R
6	ERASE_FAIL_L_INT	Smart erase on flash Block 0 failed.	0x0	R
7	-	Reserved	-	-
8	AHBH_INT	Flash Block 1 AHB error.	0x0	R
9	LOCKH_INT	Flash Block 1 locked page being accessed error.	0x0	R
10	ERASEH_INT	Flash Block 1 erase operation completed.	0x0	R
11	WRITEH_INT	Flash Block 1 write operation completed.	0x0	R
12	WR_BUFH_INT	Flash Block 1 write buffer empty.	0x0	R
13	WRITE_FAIL_H_INT	Smart Write on flash Block 1 failed.	0x0	R
14	ERASE_FAIL_H_INT	Smart erase on flash Block 1 failed.	0x0	R
31:15	-	Reserved	-	-

28.4.8 Interrupt clear register

Table 429. Interrupt clear register (INT_CLR) bit description

Bit	Symbol	Description	Reset Value	Access
0	AHBL_INT_CLR	Clear Block 0 flash AHB error status interrupt	0x0	W
1	LOCKL_INT_CLR	Clear Block 0 flash lock page error status interrupt	0x0	W
2	ERASEL_INT_CLR	Clear Block 0 flash erase status interrupt	0x0	W

Table 429. Interrupt clear register (INT_CLR) bit description *...continued*

Bit	Symbol	Description	Reset Value	Access
3	WRITEL_INT_CLR	Clear Block 0 flash write status interrupt	0x0	W
7:4	-	Reserved	-	-
8	AHBH_INT_CLR	Clear Block 1 flash AHB error status interrupt	0x0	W
9	LOCKH_INT_CLR	Clear Block 1 flash lock page error status interrupt	0x0	W
10	ERASEH_INT_CLR	Clear Block 1 flash erase status interrupt	0x0	W
11	WRITEH_INT_CLR	Clear Block 1 flash write status interrupt	0x0	W
31:12	-	Reserved	-	-

28.4.9 Lock status register x

Each flash page can be individually locked, as mentioned in Section 28.3.2. The lock status registers are 32-bit read only registers that provides the lock status of each flash page. Each bit in these lock status registers represents a page in the flash, allowing four lock registers to cover a single flash block. The first four LOCK_STAT registers (LOCK_STAT0 - LOCK_STAT3) reflect the lock status of flash block 0 while the next four LOCK_STAT registers (LOCK_STAT4 - LOCK_STAT7) reflect the lock status of flash block 1.

Table 430. Lock control register x (LOCK_CTRL_x) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	PAGE_LOCK_x	Flash page lock status. 0: Flash page is locked. 1: Flash page is unlocked.	0xFFFF FFFF	R

28.4.10 Lock status register 8

The lock control register 8 is a read only register that contains the lock status for the mass erase lock, flash protection bit, and memory protection bit. See Section 28.3.2 for more information on what these bits do.

Table 431. Lock control register 8(LOCK_CTRL_8) bit description

Bit	Symbol	Description	Reset Value	Access
0	MASS_ERASE_LOCK_EN	Mass erase lock status. 0: Mass erase is locked. 1: Mass erase is unlocked.	0x1	R
1	FSH_PROTECT_EN	SWD Flash protection status. 0: Flash is unprotected. 1: Flash is protected.	0x1	R
2	MEM_PROTECT_EN	SWD memory protection status. 0: Memory is unprotected. 1: Memory is protected.	0x1	R
31:3	-	Reserved	-	-

28.4.11 Status register 1

Table 432. Status register 1(STATUS1)bit description

Bit	Symbol	Description	Reset Value	Access
8:0	-	Reserved	-	-
9	FSH_ERA_BUSY_L	Flash Block 0 erase operation is in progress.	0x0	R
10	FSH_WR_BUSY_L	Flash Block 0 write operation is in progress.	0x0	R
11	-	Reserved	-	-
12	FSH_ERA_BUSY_H	Flash Block 1 erase operation is in progress.	0x0	R
13	FSH_WR_BUSY_H	Flash Block 1 write operation is in progress.	0x0	R
14	-	Reserved	-	-
15	INI_RD_DONE	flash initial read done.	0x0	R
25:16	-	Reserved	-	-
26	FSH_STA	Flash ready status. The flash is ready for operation when this bit is ‘1’.	0x1	R
31:27	-	Reserved	-	-

28.4.12 Flash block 0 error information register 1

Table 433. Flash block 0 error information register 1(ERR_INFOL_1) bit description

Bit	Symbol	Description	Reset Value	Access
17:0	WR_FAILEDL_ADDR	When a flash Block 0 Smart Write fails, the address is stored in this bit field.	0x0	R
23:18	SMART_FAILL_CTR	The amount of fails during a Smart Write or Smart Erase is stored in this bit field.	0x0	R
31:24	-	Reserved	-	-

28.4.13 Flash block 0 error information register 2

Table 434. Flash block 0 error information register 2(ERR_INFOL_2) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	WR_FAILEDL_DATA	When a flash Block 0 Smart Write fails, the data is stored in this bit field.	0x0	R

28.4.14 Flash block 0 error information register 3

Table 435. Flash block 0 error information register 3(ERR_INFOL_3) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	ERA_FAILEDL_INFO	When a Smart Erase on flash Block 0 fails, the address is stored in this bit field	0x0	R
31:16	-	Reserved	-	-

28.4.15 Flash block 1 error information register 1

Table 436. Flash block 1 error information register 1(ERR_INFOH_1) bit description

Bit	Symbol	Description	Reset Value	Access
17:0	WR_FAILEDH_ADDR	When a flash Block 1 Smart Write fails, the address is stored in this bit field.	0x0	R
23:18	SMART_FAILH_CTR	The amount of fails during a Smart Write or Smart Erase is stored in this bit field.	0x0	R
31:24	-	Reserved	-	-

28.4.16 Flash block 1 error information register 2

Table 437. Flash block 1 error information register 2(ERR_INFOH_2) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	WR_FAILEDH_DATA	When a flash Block 1 Smart Write fails, the data is stored in this bit field.	0x0	R

28.4.17 Flash block 1 error information register 3

Table 438. Flash block 1 error information register 3(ERR_INFOH_3) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	ERA_FAILEDH_INFO	When a Smart Erase on flash Block 1 fails, the address is stored in this bit field.	0x0	R
31:16	-	Reserved	-	-

28.4.18 Debug port access password register

Table 439. SWD erase password register (PASSWORD) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	PASSWORD	When this register is programmed with the value 0xCA1E093F, a SWD initiated Smart Mass Erase operation will bypass the current lock and protection scheme.	0x0	RW

28.4.19 Erase password register

Table 440. Erase password register (ERASE_PASSWORD) bit description

Bit	Symbol	Description	Reset Value	Access
31:0	ERASE_PASSWORD	When this register is programmed with the value 0x5B6C013AU, a FW initiated Smart Mass Erase or page erase operation will bypass the current lock and protection scheme.	0x0	RW

28.5 Functional description

28.5.1 Flash read

The flash controller supports byte, half-word and word read operations. The flash protection bit can be used to disallow a debugger to read the flash through SWD.

28.5.2 Flash write

The QN908x flash supports normal writes and HW Smart Writes. When writing to the flash, only full word writes are supported. Before writing to the flash, the pages to be written should be erased beforehand. The following are examples on how to utilize the different flash write features and the register interaction needed.

Example 1: Normal write for page 0 of Block 0, if the page is not locked.

1. Make sure that page 0 of Block 0 has been successfully erased.
2. Configure TIME_BASE according to AHB clock speed.
3. Configure SMART_WRITEL_EN to 0, and PRGML_EN to 1.
4. Use the MCU or DMA to write data to page 0 of Block0.

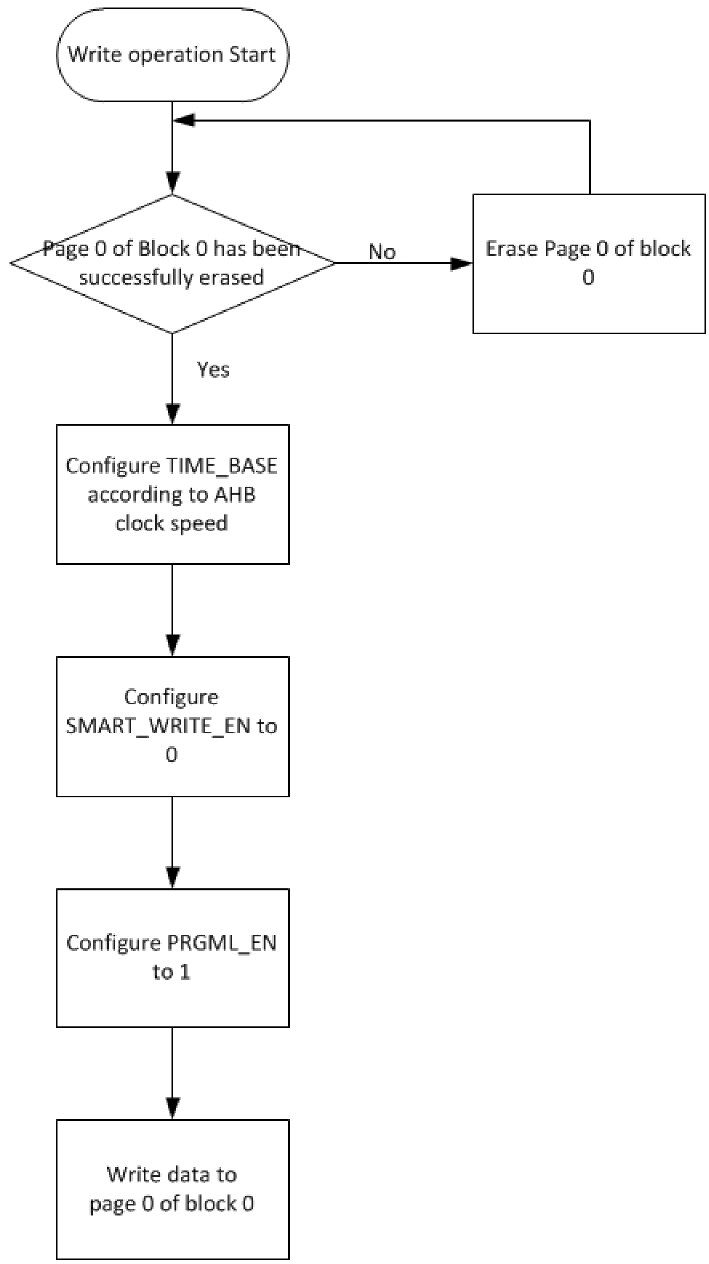


Fig 80. Flash write example 1

Example 2: Write more than one word into page 0 of block 0 in one program operation using DMA, if the page is not locked. Each page are made up of 8 rows of 64 words. DMA can write multiple words into the same row in one flash program operation. The amount of words written in the same row is determined by the PRGM_CYCLE bit field.

1. Make sure that page 0 of Block 0 has been successfully erased.
2. Configure TIME_BASE according to AHB clock speed.
3. Configure SMART_WRITEL_EN to 0, and PRGML_EN to 1.

4. Configure PRGM_CYCLE for the number of data, its range is from 2 to 64.
5. Use DMA to move data from other memory to page 0 of Block 0 by no DMA request method.

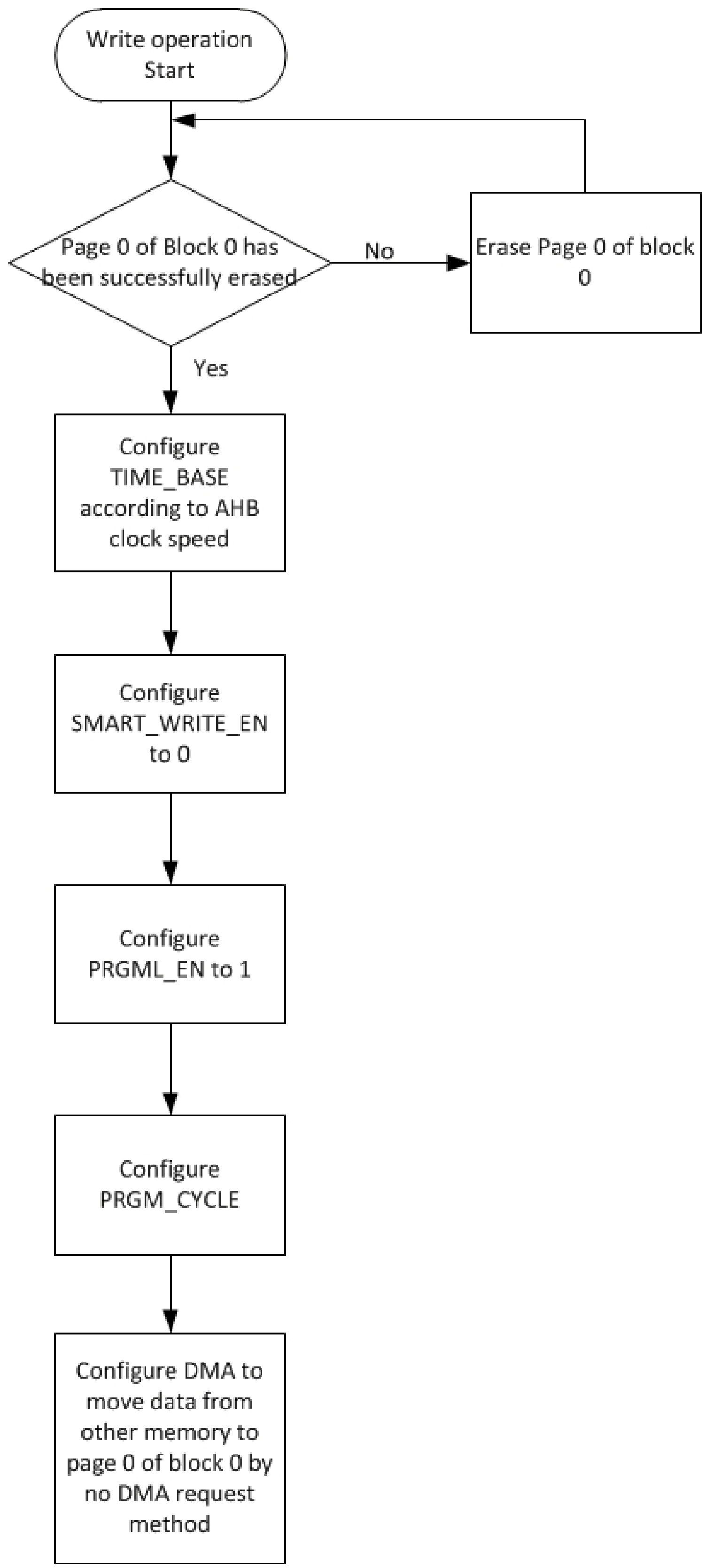


Fig 81. Flash write example 2

Example 3: HW Smart Write for page 0 of Block 0, if the page is not locked.

1. Make sure that page 0 of Block 0 has been successfully erased.
2. Configure TIME_BASE according to AHB clock speed.
3. Configure SMART_WRITEL_EN to 1, and PRGML_EN to 1.
4. Configure MAX_WRITE to 9.
5. Use the MCU or DMA to write one word to page 0 of Block0.
6. Wait for INT_WRITEL_STAT to be set to indicate that the write operation is done.
7. Check WRITE_FAIL_L to see if the Smart Write is successful.

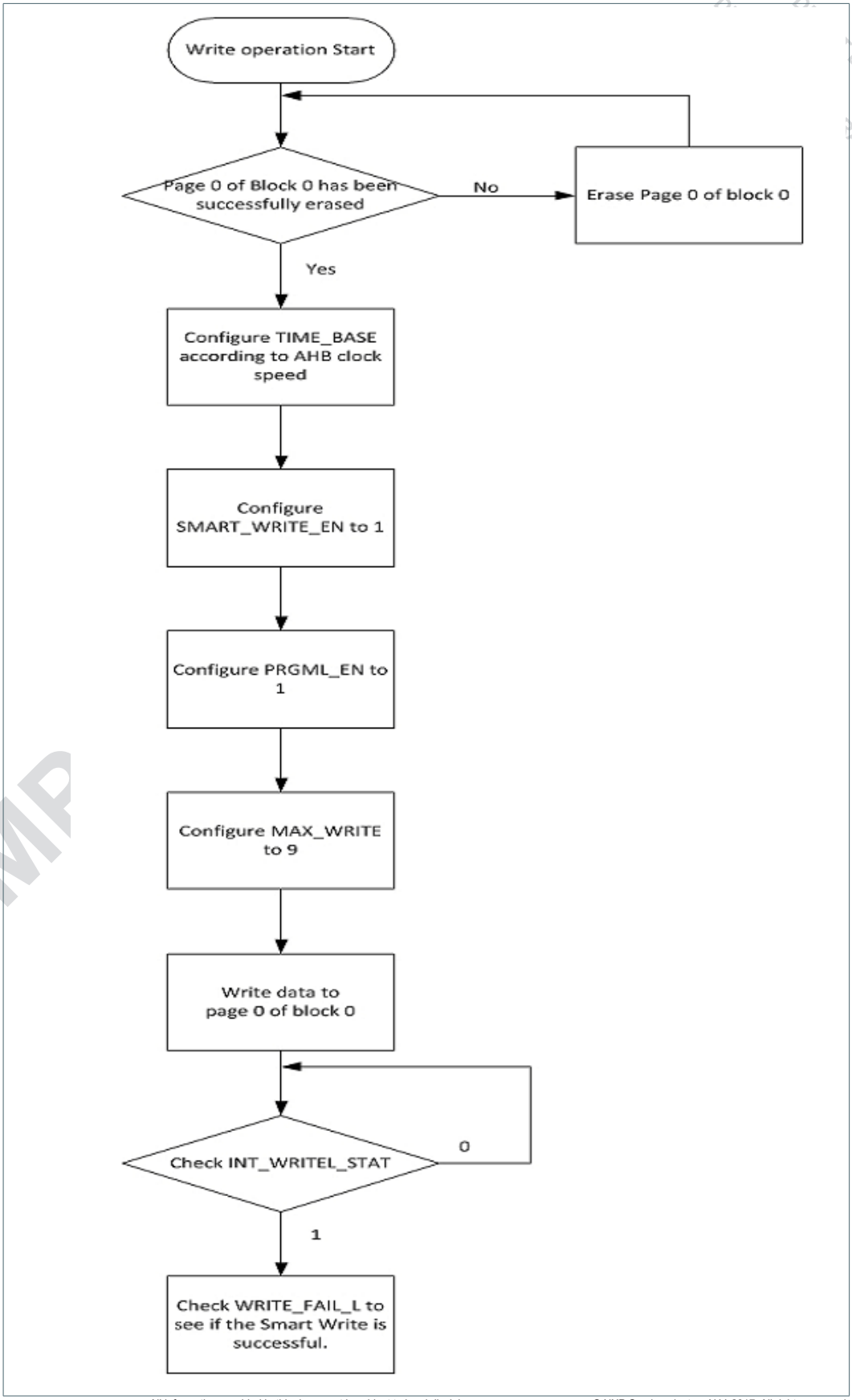


Fig 82. Flash write example 3

Example 4: Software implemented Smart Write for page 0 of Block 0, if the page is not locked.

1. Use example 2 to write data into page 0 of Block 0.
2. Use software to read the written data from address 0x3104xxxx, and check if it is the expected data.
3. If there are wrong data, repeat step 1.
4. Software can define the number of retries.

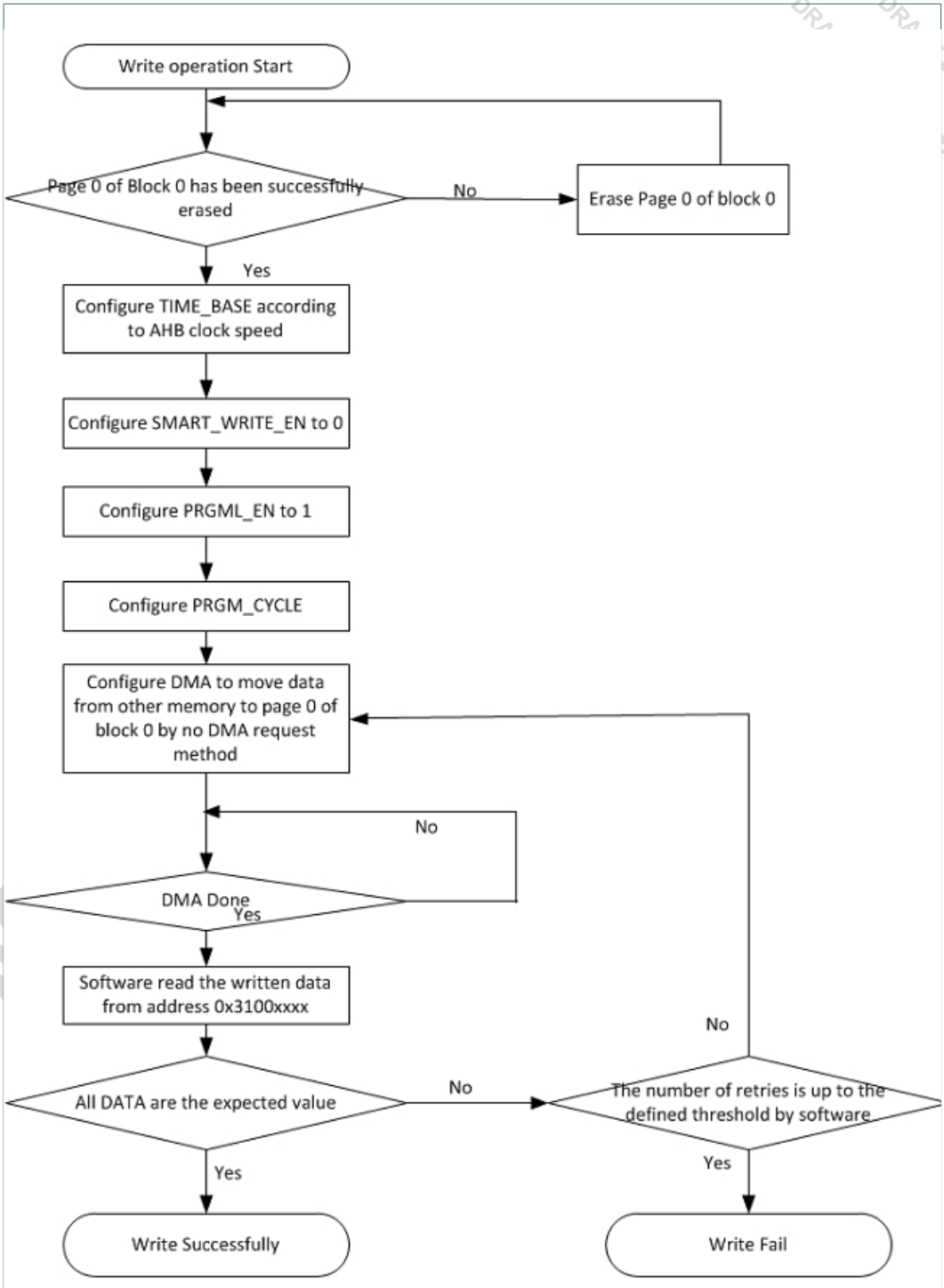


Fig 83. Flash write example 4

28.5.3 Flash erase

QN908x supports mass erase and page erase. Mass erase is used to erase an entire flash block. Page erase is used for erasing a single page in a flash block. When using the flash erase operation, the following are remarks that should be taken into consideration.

- When an erase operation on a flash page is ongoing, another erase operation cannot be issued.
- When enabling Smart Erase, the PAGE_INDX[L,H] register must be 0.
- SWD can only do smart erase.
- When mass erasing both flash blocks, the second mass erase operation should only start after the first is completed.
- If a flash page is locked, all page erase operations will fail without setting the password register to bypass the lock.
- An internal 8 MHz clock is used for erase operations. Make sure this clock is enabled by setting bit 21 of the CLK_EN register of the SYSCON registers.

The following are examples on how to utilize the different flash erase features and the register interaction needed.

Example 1: Normal page erase for page 0 of Block 0, if page 0 isn't locked.

- Make sure there are no ongoing erase operations.
- Configure ERASE_TIME_BASE to 100ms.
- Configure TIME_BASE according to AHB clock speed.
- Configure SMART_ERASEL_EN to 0.
- Configure PAGE_INDXL to 8, and PAGE_ENRASEL_EN to 1.

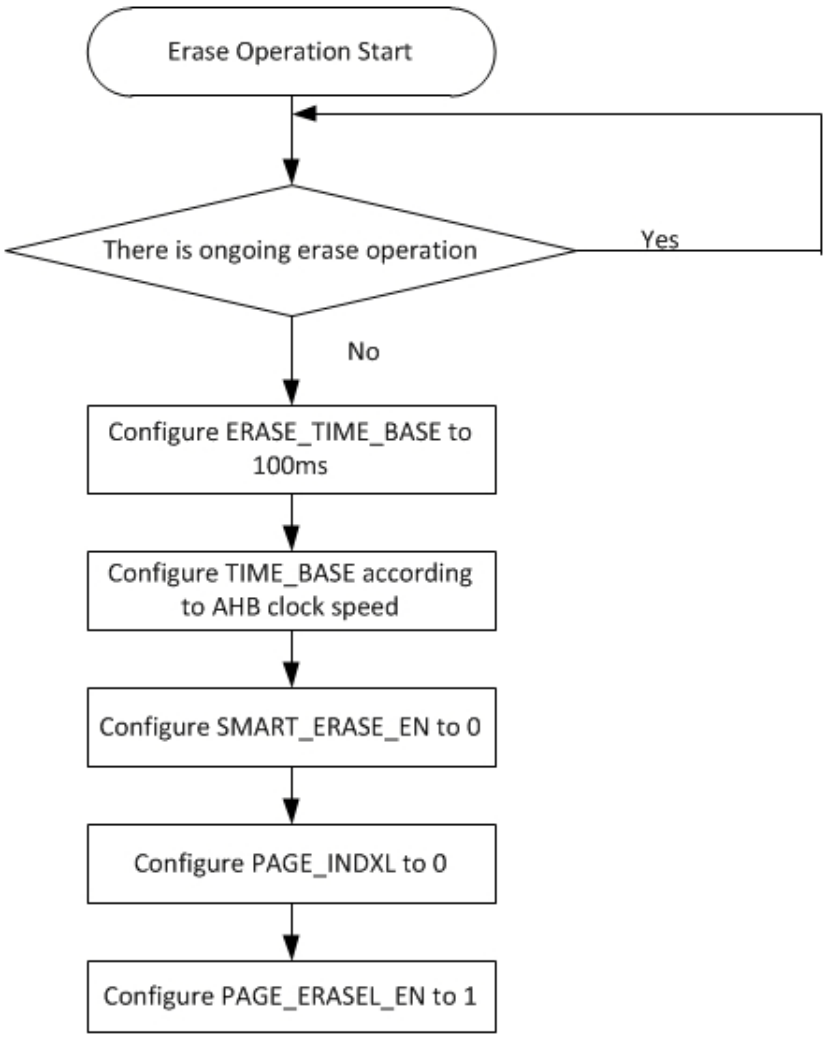


Fig 84. Flash erase example 1

Example 2: Smart mass erase for block 1, if mass erase isn't locked.

- Check all erase operation of block 0 and block 1 is done
- Configure ERASE_TIME_BASE to 2ms.
- Configure TIME_BASE according to AHB clock speed.
- Configure SMART_ERASEH_EN to 1.
- Configure MAX_ERASE to 49.
- Configure PAGE_INDXH to 0, and HALF_ERASEH_EN to 1.

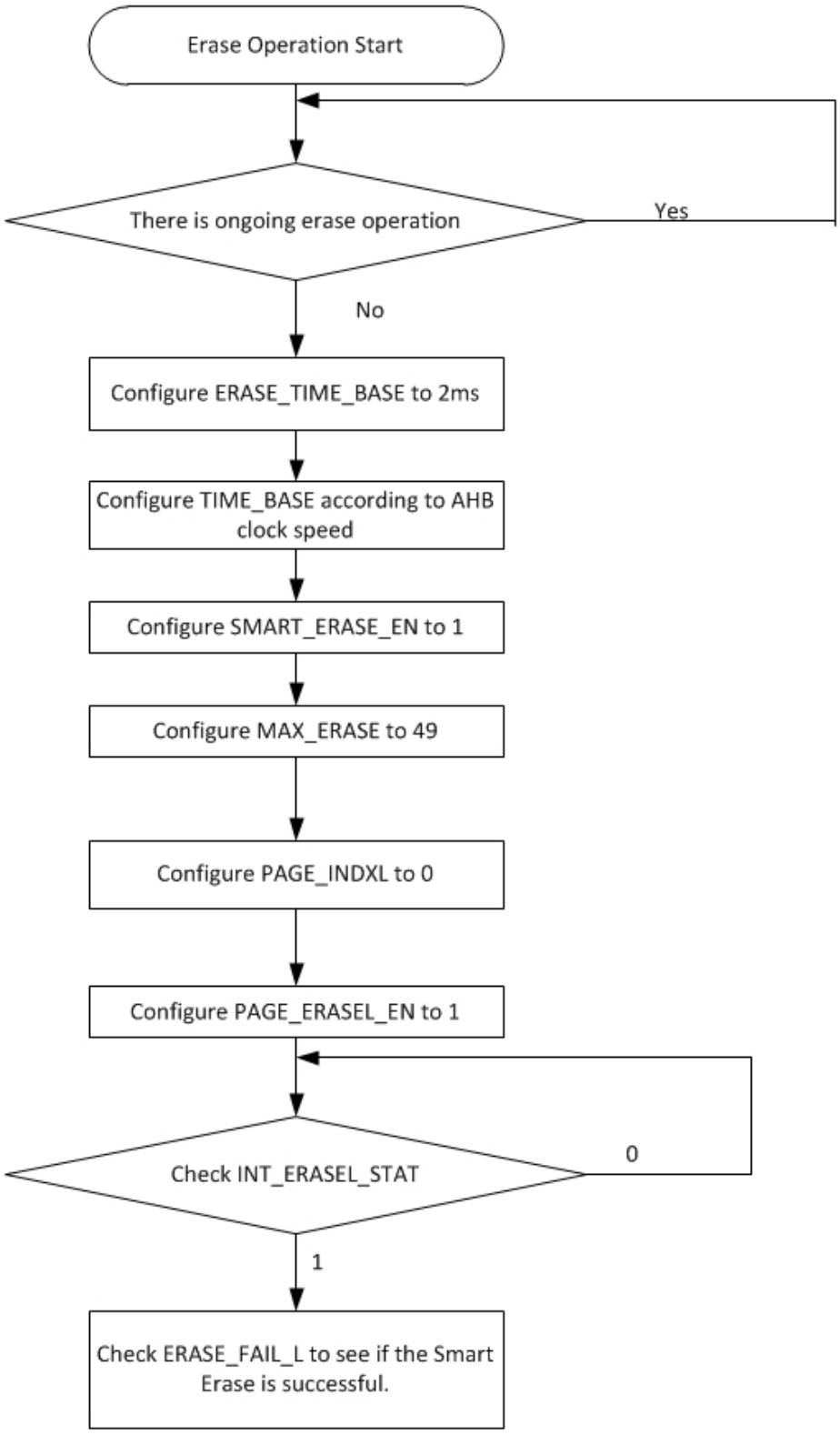


Fig 85. Flash erase example 2

29. DAC

29.1 Introduction

The DAC is available on all QN908x devices.

29.2 Features

- 1 MHz 8-bit Digital-to-analog conversion mode
- Sigma-delta modulation of 20-bit digital input data
- Input data sources:
  - 8-bit or 20-bit digital input from MCU or DMA, with gain control
  - Internal generated 20-bit sine wave with configurable frequency and amplitude
- 8-depth input data FIFO.
- Input FIFO status indication with DMA capability.
- Multiple trigger sources to start conversion:
  - Timer output
  - GPIO
  - Software trigger

29.3 Basic configuration

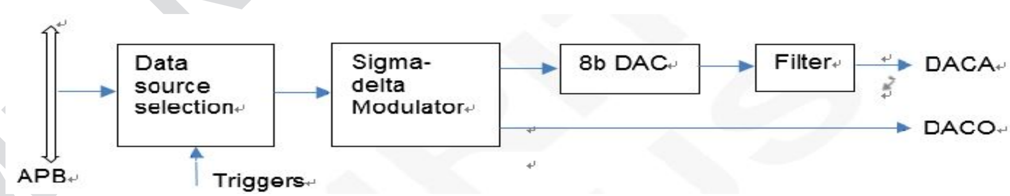


Fig 86. basic configuration

Configure the DAC as follows:

- Clear the DAC peripheral reset by writing 1 to CLR_DAC_RST register (bit 14 of RST_SW_CLR described in Section 2.5.2.
- Set the CLK_DAC_EN bit in the CLK_EN register described in Section 2.5.4 to enable the clock of DAC and start operation.The DAC module supports both analog output (DACA) and 1-bit digital PDM output (DACO).
- Configure pin multiplexing by configure PIO_FUNC_CFG<x> (x: 0 to 3) register. See Section 2.5.27~Section 2.5.30.
- Setup DAC clock speed by CLK_DIV in CTRL register. See Section 29.6.1 for details.
- Configure the DAC data source by SIN_EN bit in CTRL register. See Section 29.6.2 for details.The source data can be internal Sin-wave generator or from APB bus. DAC supports both DMA and CPU write operation into the data FIFO.
- Configure the sigma-delta modulator. See Section 29.6.4 for details.

- If analog output is used, the following needs to be configured. See Section 29.6.5 for details.
  - Clear the DIS_DAC bit in the PMU_CTRL1 register of the SYSCON block to power on DAC and filter.
  - Setup analog DAC middle voltage and gain by register VCM and DAC_AMP.
  - Setup analog output filter bandwidth by register DAC_FIL_150K_EN and DAC_FIL_BW. If necessary, configure the trigger source by TRG_MODE bit in CTRL register. See Section 29.6.6 section for details.
- If triggers are required, configure trigger source and event in CTRL register to reduce the data rate by repeating the data. If necessary, configure the corresponding trigger source block.
- Set the ENABLE bit in CTRL register to enable the DAC function. As the DAC module starts working after setting the ENABLE bit, it should be set after completing the configuration steps. The DAC provides interrupts to NVIC, so interrupt handlers are needed to handle exceptions.

29.4 Pin description

Two types of outputs are supported by this DAC module:

- DACA outputs the analog signal with an integrated filtering.
- DACO outputs the digital 1-bit PDM output with high quantization noise.

These pins can be configured by PIO_FUNC_CFG<x> (x: 0~3)(See Section 2.5.27~Section 2.5.30). Only one kind of output can be enabled at a given time. For example:

- To enable DACA analog output on PA31, PIO_FUNC_CFG3 bit 28 to 30 should be configured to 1.
- To enable DACO digital output on PA17, PIO_FUNC_CFG2 bit 4 to bit 6 should be configured to 2.
- To enable DACO digital output on PA12, PIO_FUNC_CFG1 bit 16 to bit 18 should be configured to 5.

Since there are two pins that can output DACO, they can be both enabled to output the same waveform.

Table 441: Pin description

Pin function	Type	Pin	Description	Configuration
DACA	O	PA31	Analog output	PIO_FUNC_CFG3 <30:28>=1
DACO	O	PA17	Digital PDM output	PIO_FUNC_CFG2 <6:4>: 2
DACO	O	PA12	Digital PDM output	PIO_FUNC_CFG1 <18:16> = 5

29.5 General description

The DAC block diagram is given as follows:

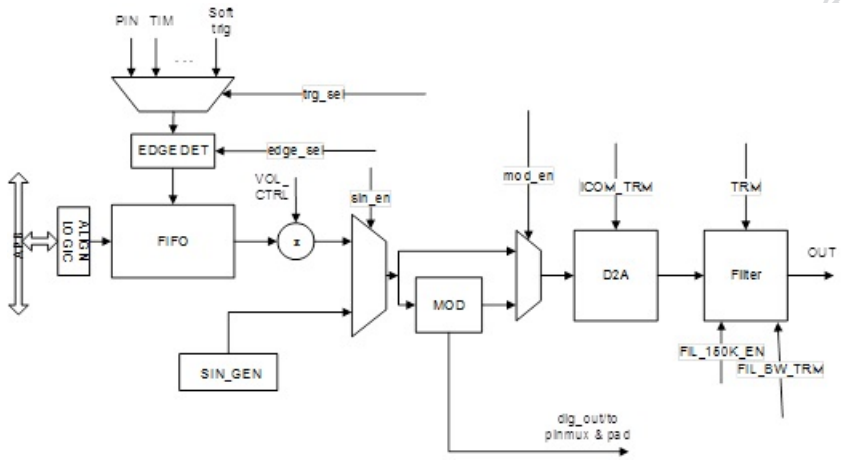


Fig 87. DAC block diagram

Both CPU and DMA can write data into internal FIFO, which is 20 bit and 8 symbol deep. The FIFO is connected to APB bus, and FIFO output can be scaled according to user configuration.

DAC uses a built-in flexible sine wave generator. The frequency, amplitude and DC offset are configurable. It provides an easy way to produce beeps and tones.

For better performance, DAC implements a third order sigma-delta modulator. Both 1 bit and 8 bit output are supported. The 1 bit output can directly drive the DACO output. The 8 bit output can be fed into the D2A converter, which is followed by a filter. The scale and offset of the converter and bandwidth of the filter are configurable.

The modulator can be bypassed to avoid quantization noise, but with limited SNR performance.

A trigger circuity is implemented to further reduce the symbol rate by repeating the data. This feature is suitable for slow changing signals. Various trigger sources such as, GPIO, timer, or PWM, and edge selection are supported.

29.6 Functional description

29.6.1 DAC clock

29.6.1.1 DAC clock DIV

The DAC clock defines the DAC conversion rate. One symbol is converted at each DAC clock.

This DAC module is a peripheral on the APB bus. A divider is used to generate the DAC clock from the APB clock. The frequency of DAC clock can be calculated as per the following expression:

f(DAC_CLK) = f(APB_CLK) / (2 * (DAC_CLK_DIV + 1))

The maximum DAC clock supported is 1 MHz.

29.6.1.2 DAC clock enable

The clock must be enabled for DAC to start functioning. To enable the DAC clock, the CLK_DAC_EN bit in the CLK_EN register of the SYSCON block is set to 1.

29.6.2 DAC DATA

- Internal sine wave generator: This is a convenient feature to produce beeps or tones.
- APB bus through an integrated FIFO: Both DMA and MCU writing to the FIFO are supported.

29.6.2.1 Sine wave generator

To enable this function, SIN_EN bit in CTRL register must be set to 1.

The frequency, amplitude and dc offset of the sine wave are configurable. The output has 20-bit resolution.

Sin_wav: amplitude * sin (2π * frequency * t) + dc_offset.

Sine wave frequency: The generator generates a value at each DAC clock. The output sine wave frequency is calculated as follows:

frequency: f(DAC_CLK)* SIN_FREQ/(2^16)

SIN_FREQ is a 16-bit value, and it can only be configured into 0x0000 - 0x7FFF to satisfy nyquist sample rule.

Remark: The sine wave frequency changes if DAC_CLK_DIV or APB clock is changed.

Sine wave amplitude: The 16 bits of SIN_AMP in CTRL register configures the amplitude of the sine wave. The sine wave amplitude is calculated as follows:

amplitude: SIN_AMP / (2 ^ 16)

Sine wave DC offset: A DC offset is added to the final Sin-wave by SIN_DC. SIN_DC is a 20-bit signed value and it can range from -1 to +1. The DC offset is calculated as follows:

dc_offset: signed(SIN_DC) / (2 ^ 19)

Sine bit: The output of sine wave generator is signed data, so SGN_INV bit in CTRL register should be set to 1.

Date align: Sin wave generator generate 20 bit output, and these 20 bit output either pass through modulator or directly to analog (modulator bypass mode). When modulator is bypass, the BUF_OUT_ALGN bit in CTRL register should be configured to 1, so that the higher 8bit of sin wave generator output will be passed to analog convertor.

At this time, the sin wave generator has no concern with BUF_IN_ALGN.

29.6.3 User data from APB through FIFO

DAC can convert user data from the APB bus by programming SIN_EN bit in CTRL register to be 0. An 8-bit entry FIFO is integrated for smooth data transition. Both MCU and DMA writing to FIFO are supported by writing the DIN register.

Remark: Use DMA to release the MCU. The DMA request signal is effective when the FIFO is not full.

The FIFO generates various interrupts depending on its status: full, not-full, empty, half-empty and overflow. Interrupt handlers are needed for the data flow control.

29.6.3.1 Data alignment

The FIFO is implemented with 20-bit width. Data are truncated from 32-bit APB bus into the FIFO. The BUF_IN_ALGN bit of CTRL register controls the selected bits.

Table 442: FIFO bit selection

BUF_IN_ALGN	Type
1	DIN<31:12>
0	DIN<19:0>

29.6.3.2 Gain control

For the flexibility of the DAC, an 8-bit gain is multiplied to the FIFO output data to scale up or down the data. The gain in GAIN register is in <u, 4, 4> format, and ranges from 0 to 255/16 in 1/16 steps.

DAT_OUT: FIFO_OUT * GAIN / (2 ^ 4)

After scaling, rounding and saturation check is performed to ensure the final data is still within 20 bit. Signal clipping might happen with high gain settings.

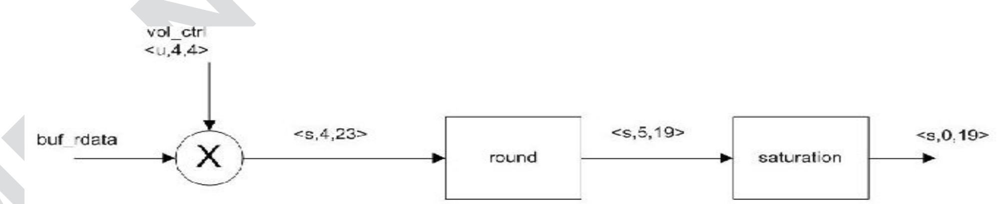


Fig 88. Gain control

29.6.4 Modulator

The sigma-delta modulator converts the 20-bit data into 1-bit (for DACO) or 8-bit (for DACA) stream without sacrificing the in-band SNR performance. This is achieved by oversampling and pushing the quantization noise out of band of interest.

To enable the modulator, MOD_EN is set to 1.

29.6.4.1 Output bit width

The modulator supports both 1-bit and 8-bit output by programming MOD_W bit field in the CTRL register.

Table 443. Modulator output bit width settings

MOD_WD	modulator output bit width
1	8
0	1

The 8-bit output is fed to the analog D2A convert to drive the DACA output.

The 1-bit output can directly drive the DACO pin as digital PDM signal. Due to its reduced resolution, the quantization noise is 42 dB higher than 8-bit output. Typically, an external filter is needed to reduce the noise.

This 1-bit output can also drive the analog D2A converter, but with 42 dB higher noise level.

29.6.4.2 Sample rate conversion

The modulator also convert a slow input symbol rate into a fast data rate for the D2A converter. The slow symbol rate release the MCU load and reduced the data count.

The up sample ratio (Nup) is configured by SMPL_RATE as given in the table below:

Table 444. Up sample ratio (Nup)

SMPL_RATE	Up sample ration(Nup)
0	8
1	16
2	32
3	64
4	128
5	256
6	512
7	1024

The rate the modulator fetches data is defined as the symbol rate:

Symbol rate: f(DAC_CLK) / Nup

For example, if DAC_CLK is set to be 1 MHz, and Nup is 16, the modulator fetches data from the FIFO at every 16 cycles that is 62.5 kHz.

Remark: Nup must be smaller than OSR (see next section).

29.6.4.3 Quantization noise calculation

Quantization noise is the key factor to determine DAC clock speed, bit width and filter bandwidth. To optimize the design, the following parameters are needed from the application:

- Band of interest (B0). For example, B0 is 20 kHz for audio, and 8 kHz for voice.
- In band SNR requirement

The key parameter of the modulator is the oversample ratio defined as follows:

OSR: f(DAC_CLK)/2/B0

The achievable SNR is:

SNR ~ # of bits * 6 dB + 70dB * log OSR/π

For example, if DAC_CLK is 1 MHz, and the signal is in audio band (20 kHz), the oversample ratio is OSR: 1 MHz/2/20 kHz: 25. With an 8-bit modulator, 111 dB SNR audio can be achieved.

Typically, a third-order filter is needed to reduce the out-of-band quantization noise. The filter bandwidth depends on the out-of-band noise requirement and is very application specific.

Remark: This calculation does not provide clear result. Use modeling or simulation for accurate results.

29.6.4.4 Modulator bypass

The modulator can be bypassed by setting MOD_EN to 0.

In this mode, depending on BUF_OUT_ALGN in the CTRL register, 8 bits of the FIFO output are directly fed into analog D2A converter, and the remaining bits are ignored.

Table 445. Bit selection in FIFO

BUF_OUT_ALGN	Bit selection
1	FIFO<19:12>/SIN<19:12>
0	FIFO<7:0>/SIN<7:0>

In the bypass mode, there is no benefit of noise shaping. So the SNR is limited to be 8 bit.

There is no up-sampling that is Nup: 1. So the symbol rate is same as DAC CLK.

29.6.5 D2A converter and filter

If the analog output (DACA) is selected, the analog D2A converter and filter needs to be powered up by clearing the DIS_DAC bit in the PMU_CTRL1 register of the SYSCON block.

Both unsigned value (0 is the minimal value) and 2's complement format (0 is the middle value) are supported. SGN_INV bit field in the CTRL register selects the input data format.

Table 446. Input data format selection

SGN_INV	Input data format
0	Unsigned value
1	2's complement format

Remark: SGN_INV must be 1 if internal sine wave generator is used.

29.6.5.1 D2A converter gain

A programmable gain is added for the flexibility of the converter. Signal clipping can occur at high gain setting.

Table 447. Full scale voltage settings

DAC_AMP	Full scale (V)
0	0.75
1	1.125
2	1.5
3	1.875
4	2.25

Table 447. Full scale voltage settings

DAC_AMP	Full scale (V)
5	2.625
6	3.0
7	3.325

29.6.5.2 D2A converter offset

To support negative data, an offset is added to output. The offset is VCM, where VCM in the ANA_CFG register ranges from 0 to 15.

Table 448. Full scale voltage settings

VCM mode	Voltage
0000b	0.8 V
0001b	0.9 V
0010b	1.0 V
0011b	1.1 V
0100b	1.2 V
0101b	1.3 V
0110b	1.4 V
0111b	1.5 V
1000b	1.6 V
1001b	1.7 V
1010b	1.8 V
1011b	1.9 V
1100b	2.0 V
1101b	2.1 V
1110b	2.2 V
1111b	2.3 V

29.6.5.3 Filter

A second-order filter is integrated to reduce the quantization noise out of the D2A converter. The filter bandwidth can be programmed by FILTER_150k_EN.

Table 449. Filter bandwidth settings

Filter_150k_EN	Filter bandwidth
0	bypassed
1	150 k

The filter also requires adjustment depending on the DACA pin load.

Table 450. DACA pin load settings

Filter_BW	Load of DACA pin
0	1k-1.4k
1	1.4k-3k
2	3k-11k
3	>11k

29.6.6 Data trigger (valid only if data is from APB bus)

The analog D2A conversion rate is DAC CLK. The modulator significantly reduces the symbol rate by a factor of Nup. In the case of very slow changing signal, the data rate can be further reduced by single mode triggers to repeat the data points.

This DAC supports two trigger modes: continuous and single mode.

29.6.6.1 Continuous mode

In this mode, TRG_MODE bit in the CTRL register should be set to 1.

The modulator reads data from FIFO automatically as per the symbol rate, f(DAC_CLK)/Nup. A new data is consumed if FIFO is not empty.

If FIFO is empty, FIFO outputs the previous data, and generates a DAC_ERR_INT. It causes signal distortion, but this feature is particularly useful to generate a DC voltage at DACA. User inputs a value in the FIFO and leaves the DAC running.

To reconstruct a varying signal with high fidelity, MCU or DMA must push data into FIFO fast enough to avoid an empty FIFO.

29.6.6.2 Single mode

In this mode, TRG_MODE bit in the CTRL register should be set to 0.

The modulator reads the data from FIFO at the speed of symbol rate, f(DAC_CLK)/Nup. However, new data is consumed only if a trigger event has happened since the last fetch, otherwise, the previous data is repeated.

In this mode, the final data rate is controlled by the trigger event, which should be slower than the symbol rate.

Ensure there is data in the FIFO when a trigger arrives. If the FIFO is empty when trigger occurs, DAC_ERR_INT becomes effective.

Various trigger source and trigger events are supported as shown in the sections below.

Trigger source: The trigger signal can be selected by TRG_SEL in the CTRL register from the following sources:

- External GPIOA
- External GPIOB
- CTIMER0 output
- CTIMER1 output
- CTIMER2 output
- CTIMER3 output
- SCT output
- Software trigger

Table 451. DAC trigger source selection

TRG_SEL	Trigger source	TRG_SEL	Trigger source
0	PA00	31	PA31
1	PA01	32	PB00
2	PA02	33	PB01
3	PA03	34	PB02
4	PA04	35~37	Software Trigger
5	PA05	38	SCT_OUT0
6	PA06	39	SCT_OUT1
7	PA07	40	SCT_OUT2
8	PA08	41	SCT_OUT3
9	PA09	42	SCT_OUT4
10	PA10	43	SCT_OUT5
11	PA11	44	SCT_OUT6
12	PA12	45	SCT_OUT7
13	PA13	46	SCT_OUT8
14	PA14	47	SCT_OUT9
15	PA15	48	CTIMER0_MAT0
16	PA16	49	CTIMER0_MAT1
17	PA17	50	CTIMER0_MAT2
18	PA18	51	CTIMER0_MAT3
19	PA19	52	CTIMER1_MAT0
20	PA20	53	CTIMER1_MAT1
21	PA21	54	CTIMER1_MAT2
22	PA22	55	CTIMER1_MAT3
23	PA23	56	CTIMER2_MAT0
24	PA24	57	CTIMER2_MAT1
25	PA25	58	CTIMER2_MAT2
26	PA26	59	CTIMER2_MAT3
27	PA27	60	CTIMER3_MAT0
28	PA28	61	CTIMER3_MAT1
29	PA29	62	CTIMER3_MAT2
30	PA30	63	CTIMER3_MAT3

Trigger event: The DAC conversion can be triggered by the edges of trigger signals. The trigger edges can be positive, negative or both, configurable by TRG_EDGE bit field in the CTRL register:

Table 452. DAC trigger edge selection

TRG_EDGE	Mode
00	positive edge
01	negative edge
10	both edge
11	reserved

If external GPIO, timer or PWM output are used as the trigger source, FIFO controller will detect the edges to update one data.

If software trigger is used, please write 1 to SW_TRG bit in CLR_TRG register to update a data. This bit is automatically cleared after conversion.

29.6.7 Interrupt handler

In the case of MCU writing data, and continuous trigger mode, an interrupt handler at non-full or half-empty buffer is needed. In the handler, a new data needs to be generated and written into the FIFO.

29.7 Register description

Table 453. DAC register overview

Name	Access	Offset	Description	Reset value	Section
ANA_CFG	rw	0x00	DAC analog configuration register	0x71022	29.7.1
CTRL	rw	0x04	DAC control register	0x0	29.7.2
SIN_CFG0	rw	0x08	Sin-wave configuration register 0	0x0	29.7.3
SIN_CFG1	rw	0x0C	Sin-wave configuration register 1	0x0	29.7.4
GAIN_CTRL	rw	0x10	Gain control register	0x10	29.7.5
CLR_TRG	WOO	0x14	Clear trigger register	0x0	29.7.6
DIN	w	0x18	DAC data input register	0x0	29.7.7
INT	W1C	0x1C	DAC interrupt register	0x0	29.7.8
INTEN	rw	0x20	DAC interrupt enable register	0x0	29.7.9
INT_STAT	r	0x24	DAC interrupt status register	0x0	29.7.10
STATUS	r	0x28	DAC status register	0x0	29.7.11

29.7.1 DAC analog configuration register

Table 454. DAC analog configuration register (ANG_CFG, offset: 0x00) bit description

Bit	Symbol	Access	Reset value	Description
2:0	FILTER_BM	rw	0x010	Set the filter bias current 000: 50% 001: 75% 010: 100%(100 uA bias current in the output stage) 011: 125% 100: 150% 101: 175% 110: 200% 111: 225%
3	-	-	-	Reserved
6:4	DAC_AMP	rw	0x010	Set the current bias of the DAC 000: 50% 001: 75% 010: 100% 011: 125% 100: 150% 101: 175% 110: 200% 111: 225%
7	-	-	-	Reserved
9:8	FILTER_BW	rw	0x00	Set the Miller compensation capacitance of the OPAMP. This compensation capacitance is determined by the off-chip load resistance 00: 56 fF(1K~1.4K) 01: 97.6 fF(1.4K~3K) 10: 141.5 fF(3K~11K) 11: 183.1 fF(<11K)
11:10	-	-	-	Reserved
12	FILTER_150K_EN	rw	0x1	Set the filter type and bandwidth
15:13	-	-	-	Reserved
19:16	VCM	rw	0x0111	Set the common mode voltage of the filter output. 800, 1200, 100 mV step
31:20	-	-	-	Reserved

29.7.2 DAC control register

Table 455. DAC control register (CTRL, offset: 0x04) bit description

Bit	Symbol	Access	Reset value	Description
0	ENABLE	rw	0b0	DAC module enable
1	SIN_EN	rw	0b0	Sine wave enable
2	MOD_EN	rw	0b0	Modulator enable
3	MOD_WD	rw	0b0	Modulator output width 0: feed back 1 bit in modulator 1: feed back 8 bit in modulator

Table 455. DAC control register (CTRL, offset: 0x04) bit description *…continued*

Bit	Symbol	Access	Reset value	Description
6:4	SMPL_RATE	rw	0x000	Sigma delta modulator down-sample rate 000: 8 001: 16 010: 32 011: 64 100: 128 101: 256 110: 512 111: 1024
7	SIGN_INV	rw	0x0	Sets inverse sign bit 1: inverse sign bit
8	BUF_OUT_ALGN	rw	0x0	FIFO and sine wave generator output data align, when modular is bypassed 0: right align 1: left align
9	BUF_IN_ALGN	rw	0x0	FIFO Input data align 0: right align 1: left align
10	TRG_MODE	rw	0x0	Trigger mode 0: single mode 1: continuous mode
12:11	TRG_EDGE	rw	0x00	Trigger edge select 0: positive edge 1: negative edge 2: both edge
15:13	-	-	-	Reserved
21:16	TRG_SEL	rw	0x000000	Trigger select 0 ~ 31: GPIOA0 ~ GPIOA31 32 ~ 34: GPIOB0 ~ GPIOB2 35 ~ 37: Software Trigger 38 ~ 47: SCT0 Output 0~9 48 ~ 63: CTimer0~3 Output Refer to Table 450 for details.
23:22	-	-	-	Reserved
30:24	CLK_DIV	rw	0x0000000	DAC clock divider
31	CLK_INV	rw	0x0	DAC clock invert, should be set to 0 or 1 after test.

29.7.3 Sin-wave configuration register 0

Table 456. Sin-wave configuration register 0 (SIN_CFG0, offset 0x08) bit description

Bit	Symbol	Access	Reset value	Description
15:0	SIN_FREQ	rw	0x0000000000000000	Sine wave frequency
31:16	SIN_AMP	rw	0x0000000000000000	Sine wave amplitude

29.7.4 Sin-wave configuration register 1

Table 457. Sin-wave configuration register 1(SIN_CFG1, offset 0x0C) bit description

Bit	Symbol	Access	Reset value	Description
19:0	SIN_DC	rw	0x00000000000000000000	DC value of sin wave
31:20	-	-	-	Reserved

29.7.5 Gain control register

Table 458. Gain control register (GAIN_CTRL, offset 0x10) bit description

Bit	Symbol	Access	Reset value	Description
7:0	GAIN	rw	0x00010000	Digital FIFO output multiply with GAIN_CTRL to scale to certain range. Where GAIN_CTRL is a <u, 4, 4> value.
31:8	-	-	-	Reserved

29.7.6 Clear trigger register

Table 459. Clear trigger register (CLR_TRG, offset 0x14) bit description

Bit	Symbol	Access	Reset value	Description
0	BUF_CLR	WOO	0x0	Clear buffer signal write 1 to clear.
1	SW_TRG	WOO	0x0	Software trigger
31:2	-	-	-	Reserved

29.7.7 DAC data input register

Table 460. DAC data input register (DIN, offset 0x18) bit description

Bit	Symbol	Access	Reset value	Description
31:0	DIN	w	0x00000000000000000000000000000000	DAC data input

29.7.8 DAC interrupt register

Table 461. DAC interrupt register (INT, offset 0x1C) bit description

Bit	Symbol	Access	Reset value	Description
0	BUF_NFUL_INT	r	0x0	Buffer not full interrupt
1	BUF_FUL_INT	r	0x0	Buffer full interrupt
2	BUF_EMT_INT	r	0x0	Buffer empty interrupt
3	BUF_HEMT_INT	r	0x0	buffer half empty interrupt
4	BUF_OV_INT	W1C	0x0	Buffer overflow interrupt write 1 to clear
5	BUF_UD_INT	W1C	0x0	Buffer underflow interrupt write 1 to clear
6	BUF_HFUL_INT	r	0x0	Buffer half full interrupt
31:7	-	-	-	Reserved

29.7.9 DAC interrupt enable register

Table 462. DAC interrupt enable register (INTEN, offset 0x20) bit description

Bit	Symbol	Access	Reset value	Description
0	BUF_NFUL_INTEN	0x0	0x0	Buffer not full interrupt enable
1	BUF_FUL_INTEN	0x0	0x0	Buffer full interrupt enable
2	BUF_EMT_INTEN	0x0	0x0	Buffer empty interrupt enable
3	BUF_HEMT_INTEN	0x0	0x0	Buffer half empty interrupt enable
4	BUF_OV_INTEN	0x0	0x0	Buffer over flow interrupt enable
5	BUF_UD_INTEN	0x0	0x0	Buffer under flow interrupt enable
6	BUF_HFUL_INTEN	0x0	0x0	Buffer half full interrupt enable
31:7	-	-	-	Reserved

29.7.10 DAC interrupt status register

Table 463. DAC interrupt status register (INT_STAT, offset 0x24) bit description

Bit	Symbol	Access	Reset value	Description
0	BUF_NFUL_INT_STAT	r	0x0	Buffer not full interrupt status
1	BUF_FUL_INT_STAT	r	0x0	Buffer full interrupt status
2	BUF_EMT_INT_STAT	r	0x0	Buffer empty interrupt status
3	BUF_HEMT_INT_STAT	r	0x0	Buffer half empty interrupt status
4	BUF_OV_INT_STAT	r	0x0	Buffer over flow interrupt status
5	BUF_UD_INT_STAT	r	0x0	Buffer under flow interrupt status
6	BUF_HFUL_INT_STAT	r	0x0	Buffer half full interrupt status
15:7	-	-	-	Reserved
16	DAC_INT_STAT	r	0x0	DAC all interrupt status
31:17	-	-	-	Reserved

29.7.11 DAC status register

Table 464. DAC status register (STATUS, offset 0x28) bit description

Bit	Symbol	Access	Reset value	Description
0	BUSY	r	0x0	Busy
15:1	-	-	-	Reserved
18:16	BUF_WR_PTR	r	0x0	Buffer write pointer
19	-	-	-	Reserved
22:20	BUF_RD_PTR	r	0x0	Buffer read pointer
31:23	-	-	-	Reserved

30. SD ADC controller (ADC)

30.1 Introduction

The ADC controller is available on all QN908x devices.

30.2 Features

- 16-bit sigma-delta ADC, with programmable gain: 0.5, 1, 1.5, and 2.
- Integrated PGA with programmable gain: 1, 2, 4, 8, and 16.
- Configurable ADC over-sampling clock up to 2 MHz.
- Maximum ADC date rate: 31.25 ksps.
- 8 external input channels, which can be configured as 8 single-ended, or 4 differential channels.
- Selectable reference voltage from VCC, internal bandgap, or external reference.
- Configurable 1.5x gain for reference voltage, to extend range of measurement.
- Flexible trigger sources, software trigger, external pin (PA00~31, PB00~02), TIM0~3 timeout, ScTimer timeout, and RNG trigger.
- Supports single conversion and burst conversion.
- Scan mode to scan the selected channels sequentially.
- Programmable digital decimation filter.
- Window compare function with interrupt capability.
- Output data FIFO, with data ready interrupt and over flow interrupt.
- Flexible output data format with channel index, to easy further software processing.
- Supports DMA.

30.3 Pin description

The ADC supports up to eight input channels, which are shared with other functions and GPIO pins. To enable one channel, two configurations are needed:

1. Enable the input channel in the pin mux.
2. Enable the corresponding channel in ADC.

Table 465. ADC pins

Function	Direction	Description	Connected to
ADC0	AI	ADC input channel 0	PA00
ADC1	AI	ADC input channel 1	PA01
ADC2	AI	ADC input channel 2	PA04
ADC3	AI	ADC input channel 3	PA05
ADC4	AI	ADC input channel 4	PA08
ADC5	AI	ADC input channel 5	PA09

Table 465. ADC pins

Function	Direction	Description	Connected to
ADC6	AI	ADC input channel 6	PA10
ADC7	AI	ADC input channel 7	PA11
ADC_VREFI	AI	ADC external reference input	PA07

30.4 Basic configuration

- Power on ADC.
- Configure pin mux in SYSCON, to select the pins used for ADC input.
- Enable the ADC channels in ADC input multiplexer by register CH_SEL.
- Configure ADC, including conversion mode, over-sampling clock, digital decimation filter, reference voltage, output data format, PGA gain setting, etc.
- Enable the ADC interrupt and NVIC.
- Enable the ADC clock.
- Enable ADC for conversion by setting ENABLE to 1, and wait one ADC over-sample clock cycle.
- Select the desired triggers, and ADC would start conversion once triggered.
- Upon the ADC data ready interrupt occurs, MCU or DMA to read ADC results.

30.5 General description

30.5.1 Principle of operation

The Figure 89 shows the block diagram of ADC.

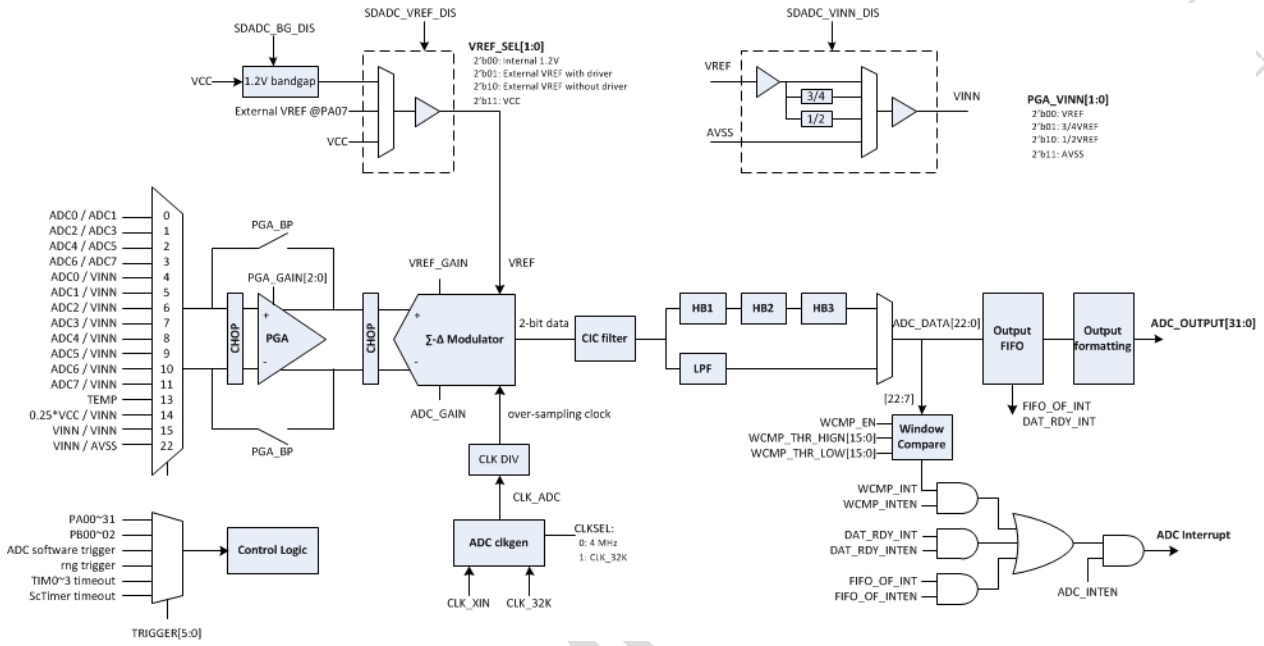


Fig 89. ADC block diagram

This ADC is a differential sigma-delta ADC. The input stage includes an input multiplexer with up to 8 external channels and a Programmable Gain Amplifier (PGA) with maximum gain of 16. The sigma-delta modulator converts the sampled input signal into a digital pulse train whose duty cycle contains the digital information. A programmable low-pass digital filter is then employed to decimate the modulator output data stream to give a valid data conversion result at programmable output rates from 500 Hz to 31.25 kHz. A chopping scheme is also employed to minimize ADC flicker noise and offset.

The ADC reference voltage can be chosen from the internal bandgap 1.2 V, the VCC, or the external reference at PA07. The source of ADC clock can be a 4MHz, or 32 kHz clock, which is then divided according to the signal acquisition requirement before being applied to the ADC.

The output stage includes the digital decimation filter and one 8-entry asynchronous FIFO. The channel index can be optionally added in the ADC result to easy further software processing. Multiple interrupts are generated to indicate the ADC and output FIFO status for MCU or DMA to move output data.

30.5.2 Input stage

30.5.2.1 Input multiplex

The ADC integrates an input multiplexer supporting eight external input channels (ADC0~7) and 3 internal channels for battery monitoring, temperature sensing and offset calibration. The analog inputs ADC0~7 are shared with the GPIO pins as Table 466. The ADC core is a differential ADC. For single-ended usage, the analog input is connected to the positive end, while the negative end input is connected to an internal generated voltage, VINN. The channels 0~3 are for external differential input, while the channels 4~11 are for external single-end input.

Table 466. ADC input multiplex

Channel index	Register bit	Positive Input Vin+	Negative Input Vin-
0	CH_SEL[0]	ADC0/PA00	ADC1/PA01
1	CH_SEL[1]	ADC2/PA04	ADC3/PA05
2	CH_SEL[2]	ADC4/PA08	ADC5/PA09
3	CH_SEL[3]	ADC6/PA10	ADC7/PA11
4	CH_SEL[4]	ADC0/PA00	VINN
5	CH_SEL[5]	ADC1/PA01	VINN
6	CH_SEL[6]	ADC2/PA04	VINN
7	CH_SEL[7]	ADC3/PA05	VINN
8	CH_SEL[8]	ADC4/PA08	VINN
9	CH_SEL[9]	ADC5/PA09	VINN
10	CH_SEL[10]	ADC6/PA10	VINN
11	CH_SEL[11]	ADC7/PA11	VINN
12	CH_SEL[12]	reserved	
13	CH_SEL[13]	Temperature sensor	
14	CH_SEL[14]	0.25 * VCC	VINN
15	CH_SEL[15]	VINN	VINN
16~19	reserved	reserved	
20	CH_SEL[20]	VINN	VSS
21~31	reserved	reserved	

The temperature sensor's output is voltage, changing along with the temperature. Calibration is needed to get the absolute temperature, which will be done before the chip is shipped out. Then temperature can be calculated from the ADC results.

The battery monitor monitors the level of the battery by converting VCC/4 to digital. If in the application system the supply of the chip is regulated to a constant voltage, the VCC/4 is constant as well and can't be used for that purpose. In this case, users have to resort to an input channel to monitor the battery. An external resistor voltage divider circuit may be needed to divide the supply voltage to fit into the dynamic range of the ADC.

To enable one channel to convert, set the corresponding register bit of CH_SEL to 1. In single mode, only the enabled channel with the lowest channel index will be converted. In scan mode, the ADC will scan all the enabled channels from the lowest channel index to the highest channel index.

30.5.2.2 VINN generation

For single-ended analog input usage, the negative input is connected to an internal voltage, VINN, which is generated from the ADC reference voltage (VREF) and configurable from (VREF), 3/4 VREF, 1/2 VREF or GND.

The driving current of VINN input and output drivers can be programmed by register. The higher sampling clock speed, the higher driving current. Refer to Application guideline.

30.5.2.3 Programmable Gain Amplifier (PGA)

The PGA has configurable gain, 1, 2, 4, 8, 16, and it can be bypassed also. The value of the PGA gain can be setup by changing the register PGA_GAIN[2:0].

When the PGA is bypassed, it should be power down to save current consumption. If the input signal has no driving capability, the PGA can be enabled with PGA_GAIN=1.

The driving current of PGA can be programmed by register. The higher sampling clock speed, the higher driving current. Refer to Application guideline.

30.5.2.4 Chopper

The ADC implement a chopping scheme whereby the ADC repeatedly reverses its inputs, which results in reduced flicker noise and excellent dc offset specifications. There are two choppers. One is located right before the PGA. The other is at the input of sigma-delta modulator.

30.5.2.5 PGA output VCM

Adjusting the PGA output voltage of common mode (VCM) is to leave enough dynamic range when PGA amplifier is applied. By default, the PGA output VCM is not adjusted with PGA_VCM_EN=0. The VCM can't be adjusted when PGA is bypassed or PGA is enabled but with gain equal to 1.

The recommended PGA output VCM is VCC/2. When the VCM of the external input signal is not close to VCC/2, it can be adjusted by

±(PGA_VCM[5]+1)*(PGA_VCM[3:0]+1)*40mV

The direction to adjust can be controlled by PGA_VCM_DIR. It's not necessary to adjust the VCM to be exactly VCC/2.

30.5.2.6 ADC sigma-delta modulator input VCM

By default, the ADC input VCM is set to VCC/2. When PGA is enabled with gain larger than 1 and the PGA output PCM is at VCC/2, there is no need to change the ADC input VCM.

When PGA is bypassed or PGA enabled with gain equal to 1, the ADC input VCM can be changed to match the VCM of the external input signal. The configurable VCM levels are 1/16 VCC, 1/8 VCC, 2/8 VCC, 3/8 VCC, 4/8 VCC, 5/8 VCC, 6/8 VCC, 7/8 VCC, by register bits ADC_VCM[2:0].

30.5.3 ADC reference voltage

The ADC reference voltage can be chosen between the internal bandgap 1.2 V, the VCC and the external reference at PA07 using VREF_SEL[2:0] register. A driver for the external reference can be selected.

The reference voltage has a configurable gain of 1.5x. For example, if the internal 1.2 V reference voltage is selected with the 1.5x gain, the reference voltage for ADC conversion is 1.8 V.

The driving current of VREF can be programmed by register. The higher sampling clock speed, the higher driving current. Refer to Application guideline.

30.5.4 ADC sampling clock generation

The ADC source clock can be selected from the 4MHz clock divided from CLK_XIN, or the CLK_32K.

Table 467. ADC source clock selection

CLKSEL	ADC source clock
0	4 MHz
1	32 KHz

Then it is fed into a clock divider, and the divider ratio is 2^(7-CLKDIV[3:0]). The clock divider output clock is the ADC over-sampling clock.

30.5.5 Conversion modes

The ADC supports multiple conversion modes, which are controlled by register bits CONV_MODE and SCAN_MODE.

Table 468. Conversion mode

Conversion mode	CONV_MODE	SCAN_EN
Single mode	0	0
Burst mode	1	0
Single scan mode	0	1
Burst scan mode	1	1

30.5.5.1 Single mode

In this mode, ADC will perform only one conversion, and then stop once the conversion is complete. The channel to convert is configurable by register CH_SEL, and only the channel with the lowest channel index will be converted.

After one conversion done, the ADC will stop operation but is still powered. To save power, the ADC should be powered down by software.

30.5.5.2 Burst mode

In this mode, ADC will perform successive conversion after once triggered, and will not stop until the register bit ENABLE is cleared. The channel to convert is configurable by register CH_SEL, and only the channel with the lowest channel index will be converted.

30.5.5.3 Single scan mode

The single scan conversion mode sweeps all the selected channels by register CH_SEL, from the channel with lowest channel index to highest channel index. After complete one round of scan of all enabled channels, the ADC will stop automatically.

For example of CH_SEL set to 0x0000_1011, the channel 0 is converted first, then the channel 4, and then channel 12, then stop.

Scan interval is the time from the end of one conversion to the start of next channel, which is in cycles of ADC over-sampling clock and configurable 2/4/8/16/32/64/128/256.

30.5.5.4 Burst scan mode

Different from the single scan mode, the burst scan mode will continue to convert after the previous round of scan is complete, and won't stop until the register bit ENABLE is cleared.

For example of CH_SEL set to 0x0000_1011, the ADC will continue to convert in sequence of the channel 0, 4, 12,0, 4,12…

Scan interval is the time from the end of one conversion to the start of next channel, which is in cycles of ADC over-sampling clock and configurable 2/4/8/16/32/64/128/256.

30.5.5.5 Channel configuration

There are two sets of configuration registers CFG0 and CFG1. Each channel can use one of them controlled by CH_CFG register. If CH_CFG[n] is 0, the channel n will use configuration register CFG0, otherwise, CFG1.

Following is the channel configurable registers for the ADC convertor.

Table 469. channel configuration registers

Field	Description
SCAN_INTV[2:0]	Interval from one channel conversion complete to next channel starting conversion, upon switching channels, in 2^(SCAN_INTV+1) cycles of ADC over-sampling clock.
DS_DATA_STABLE[3:0]	Numbers (DS_DATA_STABLE+ 1) of ADC data to discard after switching to a new channel, where the data is not stable at the beginning.
CIC_RATE[1:0]	CIC decimation rate 0=8 1=16 2=32
FILTER	Filter selection 0=half band filter, down sample 8 1=FIR filter, down sample 4
PGA_VCM[5:0]	PGA output common voltage adjustment (PGA_VCM0[5]+1)*(PGA_VCM0[3:0]+1)*40mv
PGA_VCM_DIR	PGA output common voltage adjust direction 0=down 1=up
PGA_VCM_EN	PGA output common voltage control enable.
ADC_VCM[2:0]	ADC input common voltage selection. 000b: 1/16 VCC 001b: 1/8 VCC 010b: 2/8 VCC 011b:3/8 VCC 100b: 4/8 VCC 101b: 5/8 VCC 110b: 6/8 VCC 111b:7/8 VCC

30.5.6 ADC control

30.5.6.1 ADC power control

To reduce power consumption, the power supply of sub-blocks can be controlled individually and should be shut down if not used.

Table 470. ADC power control

Sub-blockbit	Control register	Usage
Modulator	ADC_DIS	Must be enabled to run ADC
PGA	ADC_PGA_DIS	To shut down the PGA when it's bypassed.
1.2v bandgap	ADC_BG_DIS	To shut down the internal 1.2V bandgap when it's not used.
VREF driver	ADC_VREF_DIS	To shut down the VREF driver when it's not used
VINN driver	ADC_VINN_DIS	To shut down the VINN driver when it's not used in the input multiplex.

Users need power up the ADC before a new conversion, and power down the ADC intentionally after the conversion is complete to save power.

30.5.6.2 Enable ADC

The ADC should be enabled first before starting any conversion, by setting the register bit ENABLE of ADC_CTRL register to 1.

Before ADC can be enabled for conversion, it should be powered on about 100us in advance.

30.5.6.3 ADC triggers

Once the ADC is powered up and enabled, wait one ADC over-sample clock cycle time delay, it will wait for ADC trigger to start conversion. The supported trigger sources are listed in below table.

Table 471. ADC trigger sources

TRG_SEL register	Trigger source
0~31	PA00~PA31
32~34	PB00~PB02
35	Software trigger (by writing 1 to SW_START)
36	RNG TRIGGER
37	RESERVED
38~47	SCT_OUT0~9
48~51	CTIMER0_MAT0~3
52~55	CTIMER1_MAT0~3
56~59	CTIMER2_MAT0~3
60~63	CTIMER3_MAT0~3

When using software trigger, set SW_START of register CTRL(Table 474) to 1 to start one conversion. No need to clear it.

While using other trigger sources, ADC conversion will start upon rising edge of the selected trigger source, which should hold high for one ADC sampling clock cycle for proper operation.

30.5.7 Digital decimation filter

The modulated signal is feed into the CIC filter, which has a configurable decimation rate of 8, 16 or 32.

Then the output of CIC filter can be further decimated by three cascaded half-band filters, or a FIR filter. The total decimation rate of half-band filters is 8. The half band filters are more efficient for high latency, high resolution wide band signal. The total gain of half-band filters is NOT 1, but 1/(1 + 2^-6 + 2^-7 + 2^-11), so in software need multiply by (1+2^-6+2^-7+2^-11) on the ADC output to get the desired results. When the half-band filters are selected, the register DS_DATA_STABLE0/1 should be set to 0xD, to discard the unstable output data at the beginning.

The FIR filter provides a decimation rate of 4, which is more efficient for low length, multi-channel narrow band signal. When this filter is selected, please set DS_DATA_STABLE0/1 to 0x2, to discard the unstable output data at the beginning.

30.5.8 Output stage

30.5.8.1 Window compare

The window compare function is to compare the ADC result, against pre-defined thresholds. When the ADC result is larger than WCMP_THR_HIGH, or less than WCMP_THR_LOW, a window compare interrupt WCMP_INT will be generated. The interrupt can be cleared by writing 1. It has the interrupt mask bit to enable/disabled the interrupt.

The threshold registers are only 16-bit signed number, so only the high 16 bits of ADC results are monitored.

30.5.8.2 Output data format

The ADC result is 23-big signed fractional data, with the MSB (bit 22) as the sign, while the output data to MCU is 32-bit. The sign bit of ADC result can be extended in output data, to easy further software processing, controlled by register bit DATA_FORMAT. And the channel index can be added in the output data as well, controlled by register bit CH_IDX_EN, which is useful when there are multiple channels in scan mode.

Table 472. ADC output data format

DATA_FORMAT	CH_IDX_EN	ADC_OUTPUT[31:0]
1	0	ADC_OUTPUT[31:23] is signed extension of ADC_DATA[22], ADC_OUTPUT[22:0]: ADC_DATA[22:0]
1	1	ADC_OUTPUT[31:27]: CH_IDX[4:0], ADC_OUTPUT[27:23] is signed extension of ADC_DATA[22], ADC_OUTPUT[22:0]: ADC_DATA[22:0]
0	0	ADC_OUTPUT[31:9]: ADC_DATA[22:0] ADC_OUTPUT[8:0]: 0x000
0	1	ADC_OUTPUT[31:9]: ADC_DATA[22:0], ADC_OUTPUT[8:5]: 0x0, ADC_OUTPUT[4:0]: CH_IDX[4:0]

30.5.8.3 ADC output FIFO

The ADC result is stored in a 8-entry asynchronous FIFO. Once the ADC result is put into the FIFO, a data ready interrupt DATA_RDY_INT is generated to notify MCU or DMA to move data. The flag is cleared automatically if no data in the FIFO.

If FIFO is not read in time and overflow occurs, a FIFO overflow interrupt FIFO_OF_INT is generated and should be cleared by writing 1.

Both have the interrupt mask bit to enable/disable them to MCU.

30.5.8.4 ADC interrupt

There are three interrupt sources in ADC

- DAT_RDY_INT: ADC data ready interrupt flag
- WCMP_INT: window compare interrupt flag
- FIFO_OF_INT: output FIFO overflow interrupt flag

The three interrupts have individual interrupt mask bit, to enable/disable the interrupt. Then the three are logic OR'ed to generate the ADC interrupt to MCU, which has the interrupt mask bit ADC_INTEN as well.

30.6 Register description

The reset value reflects the data stored in used bits only. It does not include reserved bits content.

Table 473. Register overview: ADC (base address 0x4000 7000)

Name	Access	Offset	Description	Reset value	Section
CTRL	RW	0x00	ADC control register	0x21000	30.6.1
CH_SEL	RW	0x04	ADC channel selection register	0x1	30.6.2
CH_CFG	RW	0x08	ADC channel configuration register	0x0	30.6.3
WCMP_THR	RW	0x0C	Window compare threshold register	0x7fff8000	30.6.4
INTEN	RW	0x10	ADC interrupt mask register	0x0	30.6.5
INT	W1C	0x14	ADC interrupt status register	0x0	30.6.6
DATA	R	0x18	ADC converted data output	0x0	30.6.7
CFG0	RW	0x20	ADC configuration 0 register	0x83802950	30.6.8
CFG1	RW	0x24	ADC configuration 1 register	0x83802950	30.6.9
BG_BF	RW	0x28	ADC bandgap and buffer setting register	0x282	30.6.10
ANA_CTRL	RW	0x2C	ADC core and reference setting register	0x55564	30.6.11

30.6.1 ADC Control register

This register specifies the clock divider value to be used to generate the ADC clock **in synchronous mode** and general operating mode bits including resolution and sampling time.

Table 474. ADC Control register (CTRL, offset 0x0) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ENABLE		ADC enable. Write 1 to Writing 1 before starting conversion and 0 to end conversion.	0	rw
1	CONV_MODE		ADC conversion mode	0	rw
		0	burst conversion		
		1	single conversion		
2	SCAN_EN		1 to enable scan mode	0	rw
3	WCMP_EN		1 to enable window compare	0	rw
6:4	-	-	Reserved	-	-
7	SW_START		Software start ADC conversion write1 to trigger one time ADC conversion, no need clear.	0	WOO
9:8	CLKSEL	-	Sigma-Delta ADC clock select	0	rw
		0	4M		
		1	32K		
		2	8M		

Table 474. ADC Control register (CTRL, offset 0x0) bit description

Bit	Symbol	Value	Description	Reset value	Access
12:10	CLKDIV	-	Sigma-Delta ADC clock frequency selection	4	rw
		0	divide by 128	0	rw
		1	divide by 64		
		2	divide by 32		
		3	divide by 16		
		4	divide by 8		
		5	divide by 4		
		6	divide by 2		
		7	divide by 1		
13	SIG_INV_EN	-	1 to invert Signma-Delta input signal	0	rw
15:14	VREF_SEL	-	Sigma-Delta ADC Reference source selection.	0	rw
		0	Internal vref 1.2V		
		1	External VREF		
		2	Vext without driver		
		3	VCC		
17:16	-	-	Reserved	-	-
18	CH_IDX_EN	-	1 to append channel index in data result to be used in scan mode	0	rw
19	DATA_FOR MAT	-	Data output format. When DATA_FORMAT:=0, When CH_IDX_EN:=0, the ADC_DATA[31:0] is adc data, signed data, 31 bit frac. When CH_IDX_EN:=1, the ADC_DATA[4:0] is channel output, {ADC_DATA[31:5],5°Øh0} is adc data, signed data, 31 bit frac. When DATA_FORMAT:=1, When CH_IDX_EN:=0, the ADC_DATA[22:0] is adc data, signed data, 22 bit frac. When CH_IDX_EN:=1, the ADC_DATA[31:27] is channel output, ADC_DATA[22:0] is adc data, signed data, 22 bit frac.	0	rw
20	VREFO_EN		1 to enable bandgap out-chip capacitor	0	rw
21	SRST_DIS		1 to disable adc reset.	0	rw
22	-	-	Reserved	-	-
28:23	TRIGGER		ADC start trigger source selection: 0 to 31: PA00 to PA31; 32 to 34: PB00 to PB02; 35: software trigger and write 1 to SW_START in CTRL register; 36: RNG trigger; 37: RESERVED 38~47: SCT_OUT0~9 48~51: CTIMER0_MAT0~3 52~55: CTIMER1_MAT0~3 56~59: CTIMER2_MAT0~3 60~63: CTIMER3_MAT0~3	0	rw
31:29	-	-	Reserved	-	-

30.6.2 ADC channel select register

Table 475. ADC channel select register (CH_SEL, offset 0x04) bit description

Bit	Symbol	Value	Description	Reset value	Access
31:0	CH_SEL		In scan conversion mode, the channels with 1 set will be scanned, from LSB to MSB. In none scan conversion mode, only the first channel from LSB with 1 set will be converted. If all bits are set to 0, no ADC conversion will be started.	1	rw

30.6.3 ADC channel configuration register

Table 476. ADC channel configuration register (CH_CFG, offset 0x08) bit description

Bit	Symbol	Value	Description	Reset value	Access
31:0	CH_CFG		when CH_CONFIG[N] is 0, the N channel will select the configure 0 option (see register SD_CONFIG0). when CH_CONFIG[N] is 1, the N channel will select the configure 1 option (see register SD_CONFIG1).	0	rw

30.6.4 Window compare threshold register

Table 477. Window compare threshold register (WCMP_THR, offset 0x0C) bit description

Bit	Symbol	Value	Description	Reset value	Access
15:0	WCMP_THR_LOW	-	<s 0 15> Windows compare low threshold.	0x8000	rw
31:16	WCMP_THR_HIGH	-	<s 0 15> Windows compare high threshold If ADC decimation result is out of the window one compare interrupt will be triggered.	0x7FFF	rw

30.6.5 ADC interrupt mask register

Table 478. ADC interrupt mask register (INTEN, offset 0x10) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DAT_RDY_INTEN	-	1 to enable Data ready interrupt	0	rw
1	WCMP_INTEN	-	1 to enable Window compare interrupt.	0	rw
2	FIFO_OF_INTEN	-	1 to enalble FIFO overflow interrupt.	0	rw
30:3	-	-	Reserved	-	-
31	ADC_INTEN	-	1 to enable ADC interrupt	0	rw

30.6.6 ADC interrupt status register

Table 479. ADC interrupt status register (INT, offset 0x14) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DAT_RDY_INT		Data ready interrupt will be cleared after fifo data is read can not be cleared by write 1.	0	r
1	WCMP_INT		Window compare interrupt.	0	W1C
2	FIFO_OF_INT		FIFO overflow interrupt.	0	W1C
30:3	-	-	Reserved	-	-
31	ADC_INT		ADC interrupt.	0	r

30.6.7 ADC converted data register

Table 480. ADC converted data register (DATA, offset 0x18) bit description

Bit	Symbol	Value	Description	Reset value	Access
31:0	DATA		ADC data read from FIFO.	0	r

30.6.8 ADC configuration 0 register

Table 481. ADC configuration 0 register (CFG0, offset 0x20) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	PGA_GAIN0	-	SD ADC input PGA gain=2^value the range is 1-16.	0	rw
3	PGA_BP0	-	1 to bypass SD ADC input PGA	0	rw
5:4	PGA_VINN0	-	SD ADC PGA VIN input offset selection	1	rw
		0	VREF		rw
		1	3/4 VREF		
		2	1/2 VREF		
		3	AVSS		
7:6	ADC_GAIN0	-	SD ADC gain selection.	1	rw
		0	0.5x		
		1	1x		
		2	1.5x		
		3	2x		
8	VREF_GAIN0	-	SD ADC Reference Gain seletion	1	rw
		0	1x		
		1	1.5x		
11:9	ADC_VCM0	-	SD ADC input common voltage selection.	4	rw
		0	1/16 VCC		
		1	1/8 VCC		
		2	2/8 VCC		
		3	3/8 VCC		
		4	4/8 VCC		
		5	5/8 VCC		
		6	6/8 VCC		
		7	7/8 VCC		
12	PGA_VCM_EN0	-	SD ADC PGA output common voltage control enable signal.	0	rw
13	PGA_VCM_DIR0	-	SD ADC PGA output common voltage control direction signal.	1	rw
		0	down		
		1	up		
19:14	PGA_VCM0	-	SD ADC PGA output common voltage, adjustment: (PGA_VCM0[5]+1)*(PGA_VCM0[3:0]+1)*40mv)	0	rw

Table 481. ADC configuration 0 register (CFG0, offset 0x20) bit description

Bit	Symbol	Value	Description	Reset value	Access
22:20	DOWN_SAMPLE_ RATE	-	Filter down sample rate	0	rw
		1	down sample 32		
		3	down sample 64		
		5	down sample 128		
		4	down sample 256		
28:23	DS_DATA_STABL E0		Down sample date stable number. you can keep the bit 1:0 to 2’b11. DS_DATA_STABLE0[5:2]+1	7	rw
31:29	SCAN_INTV0	-	Interval when switching ADC source; 2/4/8/16/32/64/128/256 clock cycle.	4	rw
		0	2 clock cycle		
		1	4 clock cycle		
		2	8 clock cycle		
		3	16 clock cycle		
		4	32 clock cycle		
		5	64 clock cycle		
		6	128 clock cycle		
		7	256 clock cycle		

30.6.9 ADC configuration 1register

Table 482. ADC configuration 1register (CFG1, offset 0x24) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	PGA_GAIN1	-	SD ADC input PGA (programmable gain amplifier) gain=2^value the range is 1-16.	0	rw
3	PGA_BP1	-	1 to bypass SD ADC input PGA	0	rw
5:4	PGA_VINN1	-	SD ADC PGA VIN input offset selection	1	rw
		0	VREF		
		1	3/4 VREF		
		2	1/2 VREF		
		3	AVSS		
7:6	ADC_GAIN1	-	SD ADC gain selection.	1	rw
		0	0.5x		
		1	1x		
		2	1.5x		
		3	2x		
8	VREF_GAIN1	-	SD ADC Reference Gain seletion	1	rw
		0	1x		
		1	1.5x		

Table 482. ADC configuration 1register (CFG1, offset 0x24) bit description

Bit	Symbol	Value	Description	Reset value	Access
11:9	ADC_VCM1	-	SD ADC input common voltage selection.	4	rw
		0	1/16 VCC		
		1	1/8 VCC		
		2	2/8 VCC		
		3	3/8 VCC		
		4	4/8 VCC		
		5	5/8 VCC		
		6	6/8 VCC		
		7	7/8 VCC		
12	PGA_VCM_EN1	-	SD ADC PGA output common voltage control enable signal.	0	rw
13	PGA_VCM_DIR1	-	SD ADC PGA output common voltage control direction signal.	1	rw
		0	down		
		1	up		
19:14	PGA_VCM1	-	SD ADC PGA output common voltage, adjustment: (PGA_VCM1[5]+1)*(PGA_VCM1[3:0]+1)*40mv)	0	rw
22:20	CIC_RATE1	-	Filter down sample rate	0	rw
		1	down sample 32		
		3	down sample 64		
		5	down sample 128		
		4	down sample 256		
28:23	DS_DATA_STABLE1	-	Down sample date stable number. you can keep the bit 1:0 to 2'b11. DS_DATA_STABLE0[5:2]+1	7	rw
31:29	SCAN_INTV1	-	Interval when switching ADC source; 2/4/8/16/32/64/128/256 clock cycle.	4	rw
		0	2 clock cycle		
		1	4 clock cycle		
		2	8 clock cycle		
		3	16 clock cycle		
		4	32 clock cycle		
		5	64 clock cycle		
		6	128 clock cycle		
		7	256 clock cycle		

30.6.10 ADC bandgap and buffer setting register

Table 483. ADC bandgap and buffer setting register (BG_BF, offset 0x28) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	PGA_BM	-	SD ADC buffer bias current selection.	2	rw
		0	0.5		
		1	0.75		
		2	1		
		3	1.5		
		4	2		
		5	3		
3	-	-	Reserved	-	-
7:4	BG_SEL	-	Bandgap voltage selection to compensate PVT variations8 steps with 5mV each upward. $VBG=1205+5^{BGSEL(mV)}$	8	rw
11:8	-	-	Reserved	-	-
12	TEMP_EN	-	1 to enable temperature sensor	0	rw
13	PGA_CHOP_EN	-	1 to enable chopper in PGA	0	rw
14	PGA_BM_DIV2	-	1 to half PGA bias current. 0 to keep the original PGA bias current value.	0	rw
31:15	-	-	Reserved	-	-

30.6.11 ADC core and reference setting register

Table 484. ADC core and reference setting register (ANA_CTRL, offset 0x2C) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	ADC_BM	-	ADC bias current selection.	4	rw
		0	50%		
		1	75%		
		2	100%		
		3	150%		
		4	200%		
		5	300%		
3	-	-	Reserved	-	-
4	ADC_ORDER	-	1 to enable SD ADC 2 order mode selection	0	rw
		0	3 order		
		1	2 order		
5	DITHER_EN	-	1 to enable SD ADC PN Sequence in chopper mode	1	rw
6	CHOP_EN	-	1 to enable SD ADC chopper	1	rw
7	INV_CLK	-	1 to invert SD ADC Output Clock	0	rw

Table 484. ADC core and reference setting register (ANA_CTRL, offset 0x2C) bit description

Bit	Symbol	Value	Description	Reset value	Access
10:8	VREF_BM	-	SD ADC Reference Driver bias current selection.	5	rw
		0	50%		
		1	75%		
		2	100%		
		3	150%		
		4	200%		
		5	300%		
11:11	VREF_BM_X3	-	SD ADC Reference Driver bias current triple.	0	rw
14:12	VINN_IN_BM	-	PGA VINN Input Driver bias current selection.	5	rw
		0	50%		
		1	75%		
		2	100%		
		3	150%		
		4	200%		
		5	300%		
15	-	-	Reserved	-	-
18:16	VINN_OUT_BM	-	PGA VINN Output Driver bias current selection.	5	rw
		0	50%		
		1	75%		
		2	100%		
		3	150%		
		4	200%		
		5	300%		
19	VINN_OUT_BM_X3	-	PGA VINN Output Driver bias current triple.	0	rw
20	ADC_BM_DIV2	-	1 to half SD ADC bias current half. 0 to keep that.	0	rw
31:21	-	-	Reserved	-	-

30.7 Application examples

Two application examples are provided for temperature sensor and random number generation.

30.7.1 Random number generation

The ADC is utilized to generate random number with below configurations.

- Burst mode with channel 15 only
- VINN=3/4VREF, PGA_GAIN=1
- VREF=1.2V
- Over-sampling clock 500k
- Decimation rate 256 (CIC + half band filter)
- DS_DATA_STABLE0/1 should be set to 0xD

30.7.2 Temperature sensor

The temperature sensor is connected to channel 13, and temperature can be monitored with below configuration

- Enable temperature sensor by setting TEMP_EN to 1
- Channel 13 only
- PGA_GAIN=1, PGA chopper enabled
- VREF=1.2V, ADC_GAIN=1, ADC chopper enabled
- Over-sampling clock 500k
- Decimation rate 256 (CIC + half band filter)
- DS_DATA_STABLE0/1 should be set to 0xD

31. Capacitive sense

31.1 Introduction

The Capacitive Sense is available on all QN908x parts.

31.2 Features

- Self-capacitance sensing.
- Active mode operation capable of up to 8 input channels with integrated hardware scan.
- Configurable oscillation frequency to calibrate drive current.
- Configurable channel scan window.
- Configurable idle period between channel scans.
- Supports a FIFO with 8 entries.
- Interrupts available for when a channel scan for each enabled channel is completed, FIFO is not empty, half full, or full.
- Dedicated LP mode that operates on the 32 kHz clock. When using the LP mode, a single channel can still be scanned while in Power-down mode. An interrupt can be generated to wake up the part when the channel scan returns values that meets the programmable threshold and debounce value.

31.3 Basic configuration

Configure the Capacitive Sense as follows:

- Power: CAP_SEN_DIS bit field in PMU_CTRL1 is 0x1 and the CS is switched off as default. Clear this bit to switch on CS.
- Clock: In the CLK_EN register (Section 2.5.4), set the CLK_CS_EN bit to enable the clock to the Capacitive Sense. (Remark: The Capacitive Sense is disabled on set (CLK_CS_DIS: 1).The CLK_APB is the clock source used for the Capacitive Sense. Use the CLK_DIV bit field in the CTRL0 register to divide the CLK_APB clock to the desired frequency.)
- Reset: The Capacitive Sense may be specifically reset if the SRST bit in the CTRL0 register is set. This bit must be cleared before continuing.
- Pins: Select the desired input channels in the pin mux control registers in SYSCON.
- Interrupts: Interrupts are enabled in the NVIC and INT and INTEN registers.
- Configure the Capacitive Sense for active mode operation:
  - The desired input channels will also have to be enabled in the CS_CTRL bit field of the CTRL1 register.
  - Use the OSC_FREQ bit field in the CTRL0 register to achieve the desired relaxation oscillator frequency given the expected parasitic capacitance.
  - Configure the desired channel scan window by programming the CLK_DIV and PERIOD bit field in the CTRL0 register.
  - Set the number of idle periods with the IDLE_PERIOD register.
- Enable the Capacitive Sense by setting the ENABLE bit in the CTRL0 register.

31.4 Pin description

From all available GPIO pins, four pins can be selected in the SYSCON to serve as a Capacitive Sense input channel. In addition, the CTRL1 register will need to be modified to enable a specific input channel.

Table 485. Capacitive Sense pin description

Channel	Pin	Description
CS0	PA14	Capacitive Sense input channel 0.
CS1	PA15	Capacitive Sense input channel 1.
CS2	PA16	Capacitive Sense input channel 2.
CS3	PA17	Capacitive Sense input channel 3.
CS4	PA18	Capacitive Sense input channel 4.
CS5	PA19	Capacitive Sense input channel 5.
CS6	PA24	Capacitive Sense input channel 6.
CS7	PA25	Capacitive Sense input channel 7.

Remark: While wakeup by CS, the corresponding pin should be High-Z (PIO_PULL_CFG<x> register corresponding bits should be set to 0)

31.5 General description

31.5.1 Capacitive sensing methodology

The Capacitive Sense determines proximity and touch detection based on self-capacitance. As a finger makes contact with a touch pad, the change in capacitance is measured between the input channel and ground. This is accomplished by utilizing a relaxation oscillator that increments a counter every cycle. When there is no contact on the touch pad, the relaxation oscillator will oscillate at a user-configurable frequency based on the parasitic capacitance of the system. When a finger makes contact, the total capacitance increases, causing the frequency of the oscillator to drop. This will cause the counter output to be noticeably lower than when there is no contact with the touch pad. Touch events are determined by observing the change in the counter output.

31.5.2 The operation of the Capacitive Sense

The block diagram of the Capacitive Sense is shown in Figure

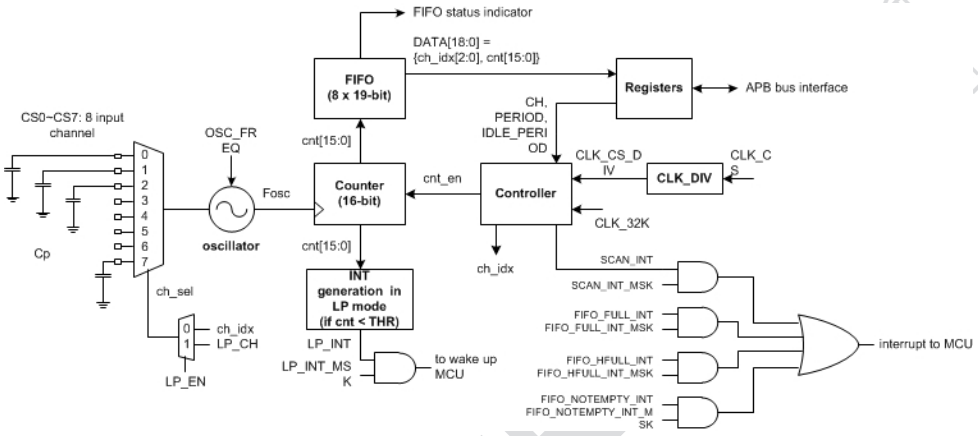


Fig 90. Capacitive Sense Block Diagram

31.5.2.1 Relaxation oscillator

Counting the frequency of the relaxation oscillator is the method the Capacitive Sense utilizes to detect a touch event. This frequency will change depending on the capacitance on the input channel and the amount of current being driven on the pin. In order to set the frequency of the relaxation oscillator to a desired value, the parasitic capacitance of the system will need to be estimated. The drive current and frequency are a function of the programmable OSC_FREQ bit field in the CTRL0 register.

The actual total amount of current being driven to the input channel is approximately 0.4 + 0.2*(OSC_FREQ + 1) μA, and the current driven to the touch pads is 0.2*(OSC_FREQ+1)uA. The frequency is approximately 2*(OSC_FREQ + 1)/RC Hz, where the resistance is about 5 M Ω. For example, if OSC_FREQ: 0 and the input channel capacitance is 10 pF, the current being driven to the pin is 0.2μA with a relaxation oscillator frequency of approximately 40 kHz. Though the calculation of the frequency is a rough estimate and will drift in frequency, a touch event should be distinguishable by detecting a significant change in the counter output.

31.5.2.2 Channel Scanning

The Capacitive Sense supports up to 8 input channels. The Capacitive Sense will scan the enabled channels starting from the lowest index to the highest.

During a scan, a counter will be incremented every clock cycle of the relaxation oscillator. The hardware will scan based on the Capacitive Sense clock source, CLK_CS. The programmable PERIOD bit field in the CTRL1 register contains a value that indicates how many CLK_CS clock cycles to scan for. Between scans, it is possible to set a specific amount of idle time in order to save power. The IDLE_PERIOD register indicates how many CLK_CS clock cycles to wait before starting another scan.

Once a channel scan has completed, the 16-bit counter output along with the 3-bit channel index will be saved into the Capacitive Sense FIFO. Reading the DATA register will return the oldest data in the FIFO.

In active mode, there are four interrupt sources that can occur after scanning: after scanning all of the enabled channels, when the FIFO is not empty, when the FIFO is half full, and when the FIFO is full.

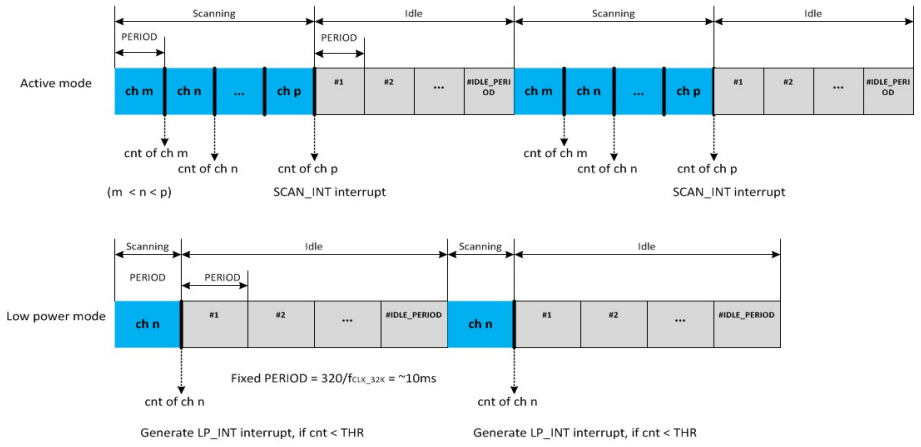


Fig 91. Operation modes

31.5.2.3 Capacitive Sense Low power mode (LP mode)

For applications that require the usage of one of the QN908x's Power-down modes, the Capacitive Sense supports a low power mode (LP mode) that can be used in exchange for some features and flexibility.

In LP mode, only one channel can be used, which acts as a wake-up mechanism. The input channel will wake up the QN908x when the Capacitive Sense controller detects that the counter value fulfills the user programmable criteria. This criteria consists of a threshold value and debounce value. When the counter output value is less than the threshold value for as many CLK_32K cycles as the debounce value, the Capacitive Sense will wake up the QN908x. Table 461 compares the differences between the Capacitive Sense's active and LP mode.

Table 486. Active mode vs. LP mode feature Comparison

Features	Active mode	LP mode
Available when QN908x is in power down mode	No	Yes
Can be used as wake-up source in power down mode	No	Yes
Capacitive Sense input clock source	CLK_APB	CLK_32K
Period to scan one channel	User programmable	Fixed to 320 clock cycles of CLK_32K
Total number of channel supported	Up to 8 channels	1 channel
Oscillation frequency	User programmable	User programmable
Idle period	User programmable	User programmable
Hardware identified touch event	Not supported	Yes, with programmable threshold, and debounce value.

31.6 Register Description

The register addresses of the Capacitive Sense are shown in Table 462. There are separate registers to enable and configure the active mode and LP mode functions.

Table 487. Capacitive Sense registers overview (base address 0x4000 7800)

Name	Access	Offset	Description	Reset value	Section
CTRL0	RW	0x00	Capacitive Sense control register 0. Contains the active mode enable, soft reset enable, drive current control, and clock divide value	0x0	31.6.1
CTRL1	RW	0x04	Capacitive Sense control register 1. Contains the scan period control and channel enable.	0x0	31.6.2
INTSTAT	W1C	0x08	Interrupt status register	0x0	31.6.3
INTEN	RW	0x0C	Interrupt mask register	0x0	31.6.4
FIFODATA	R	0x10	FIFO data register	0x0	31.6.5
LP_CTRL	RW	0x14	Capacitive Sense LP control register. Contains the debounce value, LP mode enable, LP mode channel enable, and threshold value.	0x0	31.6.6
LP_INTSTAT	R	0x18	LP mode interrupt status register	0x0	31.6.7
LP_INTEN	RW	0x1C	LP mode interrupt enable register	0x0	31.6.8
IDLE_PERIOD	RW	0x20	Active and LP mode idle configuration register.	0x0	31.6.9

31.6.1 Capacitive Sense control register 0

Table 488. Capacitive Sense control register 0 (CTRL0, offset=0x00) bit description

Bit	Symbol	Value	Description	Reset Value	Access
0	ENABLE		Capacitive Sense enable.	0x0	rw
		0	Disabled. Active mode of the Capacitive Sense is disabled.		
		1	Enabled. Active mode of the Capacitive Sense is enabled.		
1	SRST		Soft reset enable. Set 1 to reset, and 0 to de-assert.	0x0	rw
		0	Disabled. Soft reset is not asserted.		
		1	Enabled. Software reset is asserted. FIFOs will be cleared.		
7:2	OSC_FREQ		Relaxation oscillator frequency configuration. The value programmed in this bit field will adjust the current being driven to input channels based on the following equation: 0.4 + 0.2 (OSC_FREQ + 1) μA	0x0	rw

Table 488. Capacitive Sense control register 0 (CTRL0, offset=0x00) bit description *…continued*

Bit	Symbol	Value	Description	Reset Value	Access
15:8	-		Reserved	-	-
24:16	CLK_DIV		In active mode, the CLK_APB is divided by this value plus one to produce the clock for the Capacitive Sense, CLK_CS. The amount of cycles programmed into the scan window and idle periods are based on CLK_CS.	0x0	rw
31:25	-		Reserved	-	-

31.6.2 Capacitive Sense control register 1

Table 489. Capacitive Sense control register 1(CTRL1, offset=0x04) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	PERIOD	This bit field indicates how many CLK_CS clock cycles to scan a channel for.	0x0	rw
23:16	CH	Selects which Capacitive Sense channel is enabled for input scanning. A 1 in any bit of this field will cause the corresponding channel to be included for scanning where bit 16 corresponds to channel 0, bit 17 to channel 1 and so forth.	0x0	rw
31:24	-	Reserved	-	-

31.6.3 Interrupt Status register

Table 490. Interrupt Status register (INTSTAT, offset=0x08) bit description

Bit	Symbol	Description	Reset Value	Access
0	FIFO_NOTEMPTY_INT	FIFO not empty status indicator. Will clear automatically when the FIFO becomes empty.	0x0	r
1	FIFO_HFULL_INT	FIFO half full status indicator. Will clear automatically once the FIFO is less than half full.	0x0	r
2	FIFO_FULL_INT	FIFO full status indicator. Will clear automatically once the FIFO is no longer full.	0x0	r
3	SCAN_INT	Scan done status flag for all enabled channels. Write 1 to clear interrupt.	0x0	W1C
32:4	-	Reserved	-	-

31.6.4 Interrupt Mask register

Table 491. Interrupt Mask register (INTEN, offset=0x0C) bit description

Bit	Symbol	Description	Reset Value	Access
0	FIFO_NOTEMPTY_INTEN	Interrupt mask of FIFO_NOTEMPTY_INT. Set this bit to enable the interrupt.	0x0	rw
1	FIFO_HFULL_INTEN	Interrupt mask of FIFO_HFULL_INT. Set this bit to enable the interrupt.	0x0	rw
2	FIFO_FULL_INTEN	Interrupt mask of FIFO_FULL_INT. Set this bit to enable the interrupt.	0x0	rw
3	SCAN_INTEN	Interrupt mask of SCAN_INT. Set this bit to enable the interrupt.	0x0	rw
32:4	-	Reserved	-	-

31.6.5 Output Data register

Table 492. Output Data register (FIFODATA, offset=0x10) bit description

Bit	Symbol	Description	Reset Value	Access
18:0	DATA	Data from the top of the Capacitive Sense FIFO. DATA[18:16] contains the channel index address. DATA[15:0] contains the counter output for the corresponding channel index.	0x0	r
31:19	-	Reserved	-	-

31.6.6 Low Power mode control register

Table 493. Low Power mode control register (LP_CTRL, offset=0x14) bit description

Bit	Symbol	Value	Description	Reset Value	Access
3:0	DEBONCE_NUM		This bit field indicates how many consecutive CLK_32K cycles need to meet the threshold value before the wake up interrupt is triggered. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.	0x0	rw
4	LP_EN		Capacitive Sense LP mode enable.	0x0	rw
		0	Disabled. LP mode of the Capacitive Sense is disabled.		
		1	Enabled. LP mode of the Capacitive Sense is enabled.		
7:5	LP_CH		Selects which channel index to monitor while in LP mode.	0x0	rw
15:8	-		Reserved	-	-
31:16	THR		The value programmed in this bit field represents the upper bound of the counter output in terms of what is considered a valid touch event.	0x0	rw

31.6.7 Low Power Interrupt register

Table 494. Low Power Interrupt register (LP_INTSTAT, offset=0x18) bit description

Bit	Symbol	Description	Reset Value	Access
0	LP_INT	Status flag indicating that the counter output is less than the programmed threshold value.	0x0	r
31:1	-	Reserved	-	-

31.6.8 low power interrupt enable register

Table 495. low power interrupt mask register (LP_INTEN, offset=0x1C) bit description

Bit	Symbol	Description	Reset Value	Access
0	LP_INTEN	Enable interrupt for the LP mode interrupt. Set this bit to enable the interrupt.	0x0	rw
31:1	-	Reserved	-	-

31.6.9 Idle period number register

Table 496. Idle period number register (IDLE_PERIOD, offset=0x20) bit description

Bit	Symbol	Description	Reset Value	Access
15:0	IDLE_PERIOD	This bit field indicates how many clock cycles to wait between channel scans. A value of zero represents no idle time between consecutive scans.	0x0	rw
31:16	-	Reserved	-	-

32. Analog comparator (ACMP)

32.1 Introduction

Two analog comparators are present on all QN908x devices.

32.2 Features

The main features of Analog Comparator are as follows:

- Ultra low power operation
- Can operate in power down mode to wake up MCU from power down
- Input pins multiplexed with I/O pins
- Selectable internal reference voltage
- Configurable edge for interrupt
- Output routed to pins, INT Controller, Wakeup Source

32.3 Basic configuration

- Enable power of ACMP by configuring ACMP0/1_EN bit in ANA_EN (Section 2.5.40) register.
- Enable hysteresis of ACMP0/1 by writing ACMP0/1_HYST_EN to 1.
- If needed, enable ACMP 0/1 interrupt in NVIC and configure ACMP0/1_INTEN and ACMP0/1_EDGE_SEL register bit in ANA_EN register.

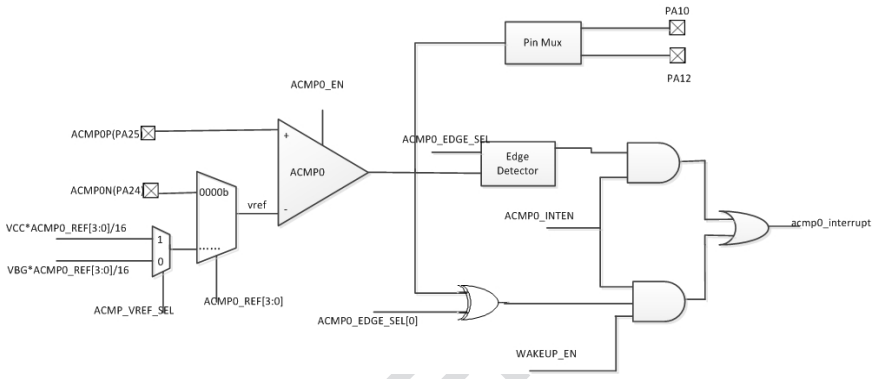
32.4 Pin information

Table 497: ACMP Pin Description

Type	Name used in pin mux	PAD name	PINMUX register and configure value
I	ACMP0N/CS6(A)	PA24	PIO_FUNC_CFG3[2:0]: 0x1 (FUNC1)
I	ACMP0P/CS7(A)	PA25	PIO_FUNC_CFG3[6:4]: 0x1 (FUNC1)
I	ACMP1N/CS6(A)	PA29	PIO_FUNC_CFG3[22:20]: 0x1 (FUNC1)
I	ACMP1P/CS7(A)	PA30	PIO_FUNC_CFG3[26:24]: 0x1(FUNC1)
O	ACMP0_OUT(O)	PA10 or PA12	PIO_FUNC_CFG1[10:8]: 0x5(FUNC5) or PIO_FUNC_CFG1[18:16]: 0x3(FUNC3)
O	ACMP1_OUT(O)	PA11 or PA13	PIO_FUNC_CFG1[14:12]: 0x5(FUNC5) or PIO_FUNC_CFG1[18:20]: 0x3(FUNC3)

32.5 General description

A block diagram of the comparator module is illustrated in figure below.



Remark: VBG is the on-chip bandgap voltage, VBG=1.2V +/- 3%. ACMP_VREF_SEL applies to both ACMP0 and ACMP1.

Fig 92. ACMP block diagram

32.6 Register description

For information of ACMP related registers, please refer to chapter Syscon ANA_EN (Section 2.5.40) register.

32.7 Functional description

32.7.1 Comparator Inputs

The input to the comparator positive end is fixed to be from external pin input by configuring the in Pin mux. The input to the comparator negative end is configurable as below table.

Table 498. Comparator input configuration

ACMP0/1 negative end input	Configuration
From external pin input	configure ACMP0/1_REF as 0000b
From internal reference voltage	Acmp_vref selection by configuring ACMP_VREF_SEL; Configure ACMP0/1_REF as other values except 0000b; The value of vref can be got from below equation: vref0/1=Acmp_vref*ACMP0/1_REF/16;

32.7.2 Comparator Outputs

While the value of comparator positive end is bigger than vref, the comparator output is 1, conversely, it outputs 0. The output of comparator are routed to specific pins, interrupt source and wakeup source.

As described in Section 32.4, output of ACMP0 can be routed to PA10 and PA12, output of ACMP1 can be routed to PA11 and PA13.

To use ACMP0/1 output as interrupt source or wakeup source, please follow below flow:

- Enable ACMP0/1 interrupt by configuring ACMP0/1_INTEN(Section 2.5.40) and NVIC;
- Select the edge for interrupt by configuring ACMP0_EDGE_SEL(Section 2.5.40)
- For wakeup, configure ACMP0_EDGE_SEL[0] (Section 2.5.40) and WAKEUP_EN bit in PMU_CTRL0 register (Section 2.5.38)

33. Appendix - ARM Cortex

33.1 ARM Cortex-M4 Details

ARM Limited publishes the document “Cortex™-M4 Devices Generic User Guide”, which is available on their website at:

- For the online manual, go to “infocenter.arm.com”, then search for “Cortex-M4 user guide”. This will bring up links to chapters of the user guide.
- There are links at the bottom of user guide chapters to download a PDF file of the user guide.

This section of this manual describes the Cortex-M4 implementation options and other distinctions that apply for the QN908x devices.

33.1.1 Cortex-M4 implementation options

The Cortex™-M4 CPU provides a number of implementation options. These are given below for the QN908x.

- The MPU is included for the Cortex-M4. The MPU provides fine grain memory control, enabling applications to implement security privilege levels, separating code, data and stack on a task-by-task basis.
- The FPU is included for the Cortex-M4. The FPU supports single-precision floating-point computation functionality in compliance with the ANSI/IEEE Standard 754-2008. The FPU provides add, subtract, multiply, divide, multiply and accumulate, and square root operations. It also performs a variety of conversions between fixed-point, floating-point, and integer data formats.
- 40 interrupt slots are implemented for the Cortex-M4. Not all interrupts are available on all part numbers.
- 3 interrupt priority bits are implemented on the Cortex-M4, providing 8 priority levels.
- Sleep mode power-saving: NXP microcontrollers extend the number of reduced power modes beyond what is directly supported by the Cortex-M4. Details of reduced power modes and wake-up possibilities can be found in Section 5.
- Reset of the Cortex-M4 resets the CPU register bank.
- Memory features: The memory map for QN908x devices is shown in Section 3.1.6.
- Bit banding is included on the Cortex-M4. APB peripherals are located in bit-band space.

34. Abbreviations

Table 499. Abbreviations

Acronym	Description
ADC	Analog-to-Digital Converter
AHB	Advanced High-performance Bus
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
API	Application Programming Interface
BOD	Brown-Out Detector
BOR	Brown-Out Reset
Boot	At power-up or chip reset, any method of importing code from an external source to execute from on-chip SRAM, or code executed in place from the external memory.
BSDL	Boundary-Scan Description Language
CRC	Cyclic Redundancy Check
CS	Capacitive Sense
DCC	Debug Communication Channel
DMA	Direct Memory Access
ETM	Embeded Trace Macrocell
FIFO	First-In-First-Out
FMC	Flash Memory Controller
FRO	Internal Free-Running Oscillator, tuned to the factory specified frequency
FSP	Fusion Signal Processing
GPIO	General Purpose Input/Output
I2C or IIC	Inter-Integrated Circuit bus
IAP	In-Application Programming
I2S	Inter-IC Sound or Integrated Interchip Sound. A serial audio data communication method.
ISP	In-System Programming. These are methods of programming any on-chip flash on a blank or previously programmed device.
ISR	Interrupt Service Routine
JTAG	Joint Test Action Group
LIN	Local Interconnect Network
NVIC	Nested Vectored Interrupt Controller
PDM	Pulse Density Modulation. This is the data format used by the digital microphone inputs.
PMU	Power Management Unit
PLL	Phase-Locked Loop
POR	Power-On Reset
PWM	Pulse Width Modulator
RAM	Random Access Memory
RCO	RC Oscillator
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory

Table 499. Abbreviations *…continued*

Acronym	Description
SWD	Serial-Wire Debug
TAP	Test Access Port
USART	Universal Synchronous/Asynchronous Receiver/Transmitter
WOO	Write Only One
W1C	Write 1 to Clear

35. References

[1] **Cortex-M4 TRM —** ARM Cortex-M4 Processor Technical Reference Manual

[2] **Cortex-M0+ TRM —** ARM Cortex-M0+ Processor Technical Reference Manual

[3] **AN11538 —** AN11538 application note and code bundle (SCT cookbook)

[4] **UM10204 —** I^2C-bus specification and user manual

36. Legal information

36.1 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.

36.2 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. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

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 and its suppliers accept 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.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

36.3 Licenses

Purchase of NXP <xxx> components
<License statement text> (replace by text inset t001lic<1nn>)

36.4 Patents

Notice is herewith given that the subject device uses one or more of the following patents and that each of these patents may have corresponding patents in other jurisdictions.

<Patent ID> — owned by <Company name>

36.5 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

<Name> — is a trademark of NXP Semiconductors N.V.

37. Tables

Table 1. Reset source types for QN908x. 10
Table 2. System register overview (base address 0x4000 0000) 11
Table 3. RST_SW_SET - Software Reset Set Register (offset = 0x0) bit description 12
Table 4. RST_SW_CLR - Software Reset Clear register (offset = 0x4) bit description 13
Table 5. CLK_DIS - Clock Disable register (offset = 0x8) bit description 14
Table 6. CLK_EN - Clock Enable register (offset = 0xC) bit description 15
Table 7. CLK_CTRL - System Clock Source and Divider register (offset = 0x10) bit description. 18
Table 8. SYS_MODE_CTRL, - System Mode and Address Remap register (offset=0x14) bit description . .19
Table 9. SYS_STAT - System Status register (offset = 0x80) bit description 19
Table 10. SYS_TICK - Systick Timer Control register (offset = 0x100) bit description 20
Table 11. SRAM_CTRL - Exchange memory base address register (offset = 0x104) bit description. 20
Table 12. CHIP_ID - Chip ID register (offset = 0x108) bit description 20
Table 13. ANA_CTRL0 - Crystal and PA register (offset = 0x110) bit description 22
Table 14. XTAL_CTRL - Crystal Control register (offset = 0x180) bit description 22
Table 15. BUCK - Buck control register (offset = 0x184) bit description 23
Table 16. FC_FRG - Flexcomm Clock Divider register (offset = 0x200) bit description 24
Table 17. PIO_PULL_CFG0 - Pad Pull Control 0 register (offset = 0x800) bit description 24
Table 18. PIO_PULL_CFG1 - Pad Pull Control 1 register (offset = 0x804) bit description 25
Table 19. PIO_PULL_CFG2 - Pad Pull Control 2 register (offset = 0x808) bit description 25
Table 20. IO_CAP - IO Status Capture register (offset = 0x80C) bit description. 25
Table 21. PIO_DRV_CFG0 - Pad Drive Strength 0 register (offset = 0x810) bit description 26
Table 22. PIO_DRV_CFG1 - Pad Drive Strength 1 register (offset = 0x814) bit description 26
Table 23. PIO_DRV_CFG2 - Pad Drive Extra register (offset = 0x818) bit description 27
Table 24. PIO_CFG_MISC - Pin Misc Control register (offset = 0x81C) bit description 27
Table 25. PIO_WAKEUP_LVL0 - Pin Wake-up Polarity register 0(offset = 0x820) bit description 28
Table 26. PIO_WAKEUP_LVL1 - Pin Wake-up Polarity register 1(offset = 0x824) bit description 29
Table 27. PIO_IE_CFG0 - Pad Input Enable register 0(offset = 0x828) bit description 29
Table 28. PIO_IEN_CFG1 - Pad Input Enable register 1(offset = 0x82C) bit description 30
Table 29. PIO_FUNC_CFG0 - Pin MUX Control register 0(offset = 0x830) bit description. 30
Table 30. PIO_FUNC_CFG1 - Pin MUX Control register 1(offset = 0x834) bit description 31
Table 31. PIO_FUNC_CFG2 - Pin MUX Control register 2(offset = 0x838) bit description. 31
Table 32. PIO_FUNC_CFG3 - Pin MUX Control register 3(offset = 0x83C) bit description 32
Table 33. PIO_WAKEUP_EN0 - Pin Function Selection in Sleep Mode register 0(offset = 0x840) bit description 33
Table 34. PIO_WAKEUP_EN1 - Pin Function Selection in Sleep Mode register 1(offset = 0x844) bit description 34
Table 35. PIO_CAP_OE0 - Pin Captured Output Enable Status register 0 (offset = 0x848) bit description . 35
Table 36. PIO_CAP_OE1 - Pin Captured Output Enable Status register 1(offset = 0x84C) bit description . 36
Table 37. PIO_CAP_OUT0 - Pin Captured Output Status register 0(offset = 0x850) bit description 36
Table 38. PIO_CAP_OUT1 - Pin Captured Output Status register 1(offset = 0x854) bit description 37
Table 39. RST_CAUSE_SRC - Reset Source Status register (offset = 0x858) bit description. 37
Table 40. PMU_CTRL0 - Power Management Unit Control register 0 (offset = 0x85C) bit description 37
Table 41. PMU_CTRL1 - Power Management Unit Control register 1(offset = 0x860) bit description. 38
Table 42. ANA_EN - Analog Setting register (offset = 0x864) bit description. 39
Table 43. XTAL32K_CTRL - 32k Crystal Control (offset = 0x868) bit description 40
Table 44. USB_CFG - USB Configuration register (offset = 0x86C) bit description 41
Table 45. PMU_CTRL2 - Power Management Unit Control register 2 (offset = 0x880) bit description 41
Table 46. ANA_CTRL1 - IVREF and DVREG setting register (offset = 0x884) bit description. 42
Table 47. MISC - MISC register (offset = 0x890) bit

continued >>

description 43
Table 48. QN9080 SRAM blocks 44
Table 49. Collection of interrupt sources to the NVIC . . .48
Table 50. Register overview: NVIC (base address 0xE000 E000) 50
Table 51. ISER0- Interrupt Set-Enable Register 0 (offset = 0x100) bit description .52
Table 52. ISER1 - Interrupt Set-Enable Register 1 (offset = 0x104) bit description .53
Table 53. ICER0 - Interrupt Clear-Enable Register 0 (offset = 0x180) bit description 53
Table 54. ICER1 - Interrupt Clear-Enable Register 1 (offset = 0x184) bit description 54
Table 55. ISPR0 - Interrupt Set-Pending Register 0 (offset = 0x200) bit description .54
Table 56. ISPR1 - Interrupt Set-Pending Register 1 (offset = 0x204) bit description .54
Table 57. ICPR0 - Interrupt Clear-Pending Register 0 (offset = 0x280) bit description 54
Table 58. ICPR1 - Interrupt Clear-Pending Register 1 (offset = 0x284) bit description 55
Table 59. IABR0 - Interrupt Active Bit Register 0 (offset = 0x300) bit description .55
Table 60. IABR1 - Interrupt Active Bit Register 1 (offset = 0x304) bit description .55
Table 61. IPR0 - Interrupt Priority Register 0 (offset = 0x400) bit description 55
Table 62. IPR1 - Interrupt Priority Register 1 (offset = 0x404) bit description 56
Table 63. IPR2 - Interrupt Priority Register 2 (offset = 0x408) bit description 56
Table 64. IPR3 - Interrupt Priority Register 3 (offset = 0x40C) bit description .56
Table 65. IPR4 - Interrupt Priority Register 4 (offset = 0x410) bit description 57
Table 66. IPR5 - Interrupt Priority Register 5 (offset = 0x414) bit description 57
Table 67. IPR6 - Interrupt Priority Register 6 (offset=0x418) bit description 57
Table 68. IPR7 - Interrupt Priority Register 7 (offset = 0x41C) bit description .58
Table 69. IPR8 - Interrupt Priority Register 8 (offset = 0x420) bit description 58
Table 70. IPR9 - Interrupt Priority Register 9 (offset = 0x424) bit description 58
Table 71. IPR10 - Interrupt Priority Register 10 (offset = 0x428) bit description .59
Table 72. IPR11 - Interrupt Priority Register 11 (offset = 0x42C) bit description .59
Table 73. IPR12 - Interrupt Priority Register 12 (offset = 0x42C) bit description .59
Table 74. STIR - Software Trigger Interrupt Register (offset = 0xF00) bit description 59
Table 75. Switched off power of RCO32K 62
Table 76. Switched off power of 32.768 kHz crystal 62
Table 77. Peripherals configuration in reduced power modes 62
Table 78. Wake-up sources for reduced power modes . . 63
Table 79. Clock pin description .64
Table 80. Register overview: Main system configuration (base address 0x4000 0000) 67
Table 81: Boot loader. 70
Table 82: Boot information content 72
Table 83: BLE confirm and value 75
Table 84: ISP commands. 76
Table 85. Available pins and configuration registers 80
Table 86. Pin pull control status definition 81
Table 87. Drive current (replace the number with defined levels?). 82
Table 88. Pin MUX table (please refer to PIO_FUNC_CFG0~3 for pin function configuration) 83
Table 89. INPUT MUX pin description 86
Table 90. Register overview: Input multiplexing (base address 0x4000 6000) 88
Table 91. Pin interrupt select registers (PINTSEL[0:3], offsets [0x200:0x20C]) bit description 89
Table 92. DMA trigger input MUX registers (DMA_ITRIG_INMUX[0:19], offsets [0x400:0x80C]) bit description 89
Table 93. DMA output trigger feedback MUX registers (DMA_OTRIG_INMUX[0:3], offset [0xA00:0xA0C]) bit description. 90
Table 94. GPIO register overview (base address 0x4008 C000 for GPIO A, 0x4008 D000 for GPIO B) . 92
Table 95. GPIO DATA register (DATA, offset: 0x00) bit description 92
Table 96. DATAOUT - DATAOUT register (offset: 0x04) bit description 92
Table 97. OUTENSET - GPIO port Output Enable Set register (offset = 0x10) bit description 92
Table 98. OUTENCLR - GPIO Output Enable Clear register (offset = 0x14) bit description. 92
Table 99. INTENSET - GPIO port Interrupt Enable Set register (offset = 0x20) bit description 93
Table 100. INTENCLR - GPIO port Interrupt Enable Clear register (offset = 0x24) bit description 93
Table 101. INTTYPESET - GPIO Interrupt Type Set register (offset = 0x28) bit description 93
Table 102. INTTYPECLR - GPIO Interrupt Type Clear register (offset = 0x2C) bit description 93
Table 103. INTPOLSET - GPIO Interrupt Polarity Set

continued >>

register (offset = 0x30) bit description. 93
Table 104. INTPOLCLR - GPIO Interrupt Polarity Clear register (offset = 0x34) bit description 93
Table 105. INTSTATUS - GPIO Interrupt Status register (offset = 0x38) bit description 93
Table 106. Interrupt feature 94
Table 107. Register overview: Pin interrupts/pattern match engine (base address 0x4000 6000) 100
Table 108. Pin interrupt mode register (ISEL, offset 0x00) bit description 100
Table 109. Pin interrupt level or rising edge interrupt enable register (IENR, offset 0x04) bit description . . 101
Table 110. Pin interrupt level or rising edge interrupt set register (SIENR, offset 0x08) bit description .101
Table 111. Pin interrupt level or rising edge interrupt clear register (CIENR, offset 0x0C) bit description .101
Table 112. Pin interrupt active level or falling edge interrupt enable register (IENF, offset 0x10) bit description 102
Table 113. Pin interrupt active level or falling edge interrupt set register (SIENF, offset 0x14) bit description . . 102
Table 114. Pin interrupt active level or falling edge interrupt clear register (CIENF, offset 0x18) bit description 102
Table 115. Pin interrupt rising edge register (RISE, offset 0x1C) bit description 103
Table 116. Pin interrupt falling edge register (FALL, offset 0x20) bit description .103
Table 117. Pin interrupt status register (IST, offset 0x24) bit description 103
Table 118. Pattern match interrupt control register (PMCTRL, offset 0x28) bit description 104
Table 119. Pattern match bit-slice source register (PMSRC, offset 0x2C) bit description 104
Table 120. Pattern match bit slice configuration register (PMCFG, offset 0x30) bit description 106
Table 121. Pin interrupt registers for edge- and level-sensitive pins 109
Table 122. DMA requests and trigger MUXes 114
Table 123. DMA with the I2C monitor function. 114
Table 124. DMA trigger sources 115
Table 125: Channel descriptor. .116
Table 126: Reload descriptors. .117
Table 127: Channel descriptor for a single transfer. 117
Table 128: Example descriptors for ping-pong operation: peripheral to buffer 117
Table 129. Register overview: DMA controller (base address 0x4008 2000) 119
Table 130. Control register (CTRL, offset 0x000) bit description 123
Table 131. Interrupt Status register (INTSTAT, offset 0x004) bit description 123
Table 132. SRAM base address register (SRAMBASE, offset 0x008) bit description 123
Table 133. Channel descriptor map 123
Table 134. Enable read and Set register 0 (ENABLESET0, offset 0x020) bit description 124
Table 135. Enable clear register 0 (ENABLECLR0, offset 0x028) bit description 124
Table 136. Active status register 0 (ACTIVE0, offset 0x030) bit description 125
Table 137. Busy status register 0 (BUSY0, offset 0x038) bit description 125
Table 138. Error Interrupt register 0 (ERRINT0, offset 0x040) bit description 125
Table 139. Interrupt enable read and Set register 0 (INTENSET0, offset 0x048) bit description . . 126
Table 140. Interrupt enable clear register 0 (INTENCLR0, offset 0x050) bit description 126
Table 141. Interrupt A register 0 (INTA0, offset 0x058) bit description 126
Table 142. Interrupt B register 0 (INTB0, offset 0x060) bit description 127
Table 143. Set valid 0 register (SETVALID0, offset 0x068) bit description 127
Table 144. Set trigger 0 register (SETTRIG0, offset 0x070) bit description 127
Table 145. Abort 0 register (ABORT0, offset 0x078) bit description 128
Table 146. Channel configuration registers bit description . . 128
Table 147. Trigger setting summary 129
Table 148. Channel control and Status registers bit description 130
Table 149. Channel transfer configuration registers bit description 131
Table 150. SCT0 pin description (inputs) 135
Table 151. SCT0 pin description (outputs) 135
Table 152. Register overview: SCTimer/PWM (base address 0x4008 5000) 138
Table 153. SCT configuration register (CONFIG, offset 0x000) bit description 143
Table 154. SCT control register (CTRL, offset 0x004) bit description 145
Table 155. SCT limit event select register (LIMIT, offset 0x008) bit description 146
Table 156. SCT halt event select register (HALT, offset 0x00C) bit description 147
Table 157. SCT stop event select register (STOP, offset 0x010) bit description 148
Table 158. SCT start event select register (START, offset

continued >>

0x014) bit description 148
Table 159. SCT counter register (COUNT, offset 0x040) bit description 149
Table 160. SCT state register (STATE, offset 0x044) bit description 150
Table 161. SCT input register (INPUT, offset 0x048) bit description 150
Table 162. SCT match/capture mode register (REGMODE, offset 0x04C) bit description 151
Table 163. SCT output register (OUTPUT, offset 0x050) bit description 151
Table 164. SCT bidirectional output control register (OUTPUTDIRCTRL, offset 0x054) bit description 152
Table 165. SCT conflict resolution register (RES, offset 0x058) bit description 153
Table 166. SCT DMA 0 request register (DMAREQUEST0, offset 0x05C) bit description 154
Table 167. SCT DMA 1 request register (DMAREQUEST1, offset 0x060) bit description 154
Table 168. SCT event interrupt enable register (EVEN, offset 0x0F0) bit description 154
Table 169. SCT event flag register (EVFLAG, offset 0x0F4) bit description 154
Table 170. SCT conflict interrupt enable register (CONEN, offset 0x0F8) bit description 155
Table 171. SCT conflict flag register (CONFLAG, offset 0x0FC) bit description 155
Table 172. SCT match registers 0 to 9 (MATCH[0:9], offset 0x100 (MATCH0) to 0x124 (MATCH9)) bit description (REGMODEn bit: 0) 156
Table 173. SCT capture registers 0 to 9 (CAP[0:9], offset 0x100 (CAP0) to 0x124 (CAP9)) bit description (REGMODEn bit: 1) 156
Table 174. SCT match reload registers 0 to 9 (MATCHREL[0:9], offset 0x200 (MATCHREL0) to 0x224 (MATCHREL9)) bit description (REGMODEn bit: 0) 156
Table 175. SCT capture control registers 0 to 9 (CAPCTRL[0:9], offset 0x200 (CAPCTRL0) to 0x224 (CAPCTRL9)) bit description (REGMODEn bit: 1) 157
Table 176. SCT event state mask registers 0 to 9 (EV[0:9]_STATE, offsets 0x300 (EV0_STATE) to 0x348 (EV9_STATE)) bit description 157
Table 177. SCT event control register 0 to 9 (EV[0:9]_CTRL, offset 0x304 (EV0_CTRL) to 0x34C (EV9_CTRL)) bit description 158
Table 178. SCT output set register (OUT[0:9]_SET, offset 0x500 (OUT0_SET) to 0x538 (OUT7_SET) bit description 159
Table 179. SCT output clear register (OUT[0:9]_CLR, offset 0x504 (OUT0_CLR) to 0x53C (OUT7_CLR)) bit description 160
Table 180. Event conditions .163
Table 181. SCT configuration example 168
Table 182. Timer/Counter pin description 173
Table 183. CTIMER[0:3] pin description (inputs) 173
Table 184. CTIMER[0:3] pin description (outputs) 173
Table 185. Register overview: CTIMER0/1/2/3 (register base addresses 0x4000 2000 (CTIMER0), 0x4000 3000 (CTIMER1), 0x4000 4000 (CTIMER2), 0x4005 0000 (CTIMER3)) 174
Table 186. Interrupt Register (IR, offset 0x000) bit description 175
Table 187. Timer Control Register (TCR, offset 0x004) bit description 175
Table 188. Timer counter registers (TC, offset 0x08) bit description 175
Table 189. Timer prescale registers (PR, offset 0x00C) bit description 176
Table 190. Timer prescale counter registers (PC, offset 0x010) bit description 176
Table 191. Match control register (MCR, offset 0x014) bit description 176
Table 192. Timer match registers (MR[0:3], offset [0x018:0x024]) bit description 177
Table 193. Capture Control Register (CCR, offset 0x028) bit description 177
Table 194. Timer capture registers (CR[0:2], offsets [0x02C:0x038]) bit description 178
Table 195. Timer external match registers (EMR, offset 0x03C) bit description 178
Table 196. Count Control Register (CTCR, offset 0x070) bit description 180
Table 197: PWM Control Register (PWMC, offset 0x074)) bit description 181
Table 198. Register overview:Watchdog timer (base address 0x4000 1000) 188
Table 199. WDT LOAD register (LOAD, offset: 0x00) bit description 188
Table 200. WDT VALUE register (VALUE, offset: 0x04) bit description 188
Table 201. WDT CONTROL register (CTRL, offset: 0x08) bit description 188
Table 202. WDT Interrupt Clear register (INT_CLR, offset: 0x0C) bit description 189
Table 203. WDT Raw Interrupt Status register (INT_RAW, offset=0x10) bit description 189
Table 204. WDT Masked Interrupt Status register (MIS,offset=0x14) bit description 189
Table 205. WDT LOCK register (LOCK, offset=0x20) bit

continued >>

description 189
Table 206. Register Overview: RTC (base address 0x4000 B000) 194
Table 207. RTC control register (CTRL, offset 0x00) bit description 195
Table 208. RTC status register (STATUS, offset 0x04) bit description 195
Table 209. RTC second register (SEC, offset 0x08) bit description 195
Table 210. RTC second register (CAL, offset 0x10) bit description 196
Table 211. RTC Count Value register (CNT_VAL, offset 0x14) bit description 196
Table 212. RTC free running control register (CNT2_CTRL, offset 0x20) bit description 196
Table 213. RTC Count Value register (THR_INT, offset 0x24) bit description 196
Table 214. RTC FR Counter Reset Threshold register (THR_RST, offset 0x28) bit description 196
Table 215. RTC Free Running count value(CNT2, offset 0x2C) bit description 196
Table 216. Register overview: SysTick timer (base address 0xE000 E000). 199
Table 217. SysTick Timer Control and status register (SYST_CSR, offset 0x010) bit description . . .199
Table 218. System Timer Reload value register (SYST_RVR, offset 0x014) bit description . . .199
Table 219. System Timer Current value register (SYST_CVR, offset 0x018) bit description . . .200
Table 220. System Timer Calibration value register (SYST_CALIB, offset 0x01C) bit description .200
Table 221. Address allocation 206
Table 222. Data address arrangement 207
Table 223. Output result 208
Table 224. Address decoder content 208
Table 225. Error handling 209
Table 226. registers for data of different channels. 214
Table 227. Cordic equation in native mode 217
Table 228. output in native mode. 218
Table 229. Matrix operation mode 218
Table 230. MOU data format 219
Table 231. Defined minimum value in matrix inversion . .219
Table 232. TE mode 220
Table 233. TE IO mode 220
Table 234. Points of transform. 220
Table 235. 221
Table 236. SE index of max/min 223
Table 237. SE data type 223
Table 238. COR data format 224
Table 239. FSP Register overview (base address 0x4008 8000). 224
Table 240. FSP system control register (SYS_CTRL, offset: 0x000) bit description 228
Table 241. FSP status register (STATUS, offset: 0x004) bit description 228
Table 242. FSP interrupt register (INT, offset: 0x008) bit description 229
Table 243. FSP interrupt enable register (INTEN, offset: 0x00C) bit description 230
Table 244. FSP Transform Engine control register (TE_CTRL, offset: 0x020) bit description 232
Table 245. TE source data memory base register (TE_SRC_BASE, offset: 0x024) bit description . . 232
Table 246. TE destination data memory base register (TE_DST_BASE, offset: 0x028) bit description . . 233
Table 247. FSP MOU control register (MOU_CTRL, offset: 0x040) bit description 233
Table 248. matrix A/B source data memory base register (MA_SRC_BASE/MB_SRC_BASE, offset: 0x048) bit description 234
Table 249. matrix output data memory base register (MO_DST_BASE, offset: 0x04C) bit description . 234
Table 250. MOU scale coefficient A/B register (MOU_SCALEA/B, offset: 0x054) bit description . 234
Table 251. FSP SE control register (SE_CTRL, offset: 0x060) bit description 234
Table 252. SE source data memory base register (SE_SRC_BASE, offset: 0x064) bit description . . 235
Table 253. Max or Min dta index register (SE_IDX_BASE, offset: 0x068) bit description 235
Table 254. SE array summary result register (SE_SUM, offset: 0x06C) bit description 235
Table 255. SE array power result register (SE_PWR, offset: 0x07C) bit description 235
Table 256. FSP COR control register (COR_CTRL, offset: 0x080) bit description 236
Table 257. correlation X/Y sequence base register (CX/Y_SRC_BASE, offset: 0x084/0x088) bit description 236
Table 258. correlation output sequence base register (CX/Y_SRC_BASE, offset: 0x08C) bit description 236
Table 259. correlation offset register (COR_OFFSET, offset: 0x090) bit description 236
Table 260. FIR channel 0 configuration register (FIR_CFG_CH0, offset: 0x0A0) bit description . . 237

continued >>

Table 261. FIR channel [1,8]configuration register (FIR_CFG_CH[1,8], offset: 0x0A4/0x0A8/0x0AC/0x0B0/0x0B4/0x0B8/0x0BC /0x0C0) bit description 237
Table 262. FIR channel x fix-point data register (FIR_DATx_FX, offset: 0x0D0) bit description 237
Table 263. FIR channel x float data register (FIR_DATx_FL, offset: 0x100) bit description. 237
Table 264. Sin & cos IXOX mode data address register (SIN_COS_IXOX, offset: 0x140) bit description . . 238
Table 265. Sin & cos IXOL mode data address register (SIN_COS_IXOL, offset: 0x144) bit description . . 238
Table 266. Sin & cos ILOX mode data address register (SIN_COS_ILOX, offset: 0x148) bit description . . 238
Table 267. Sin & cos ILOL mode data address register (SIN_COS_ILOL, offset: 0x14C) bit description . . 239
Table 268. LN & SQRT IXOX mode data address register (LN_SQRT_IXOX, offset: 0x150) bit description . 239
Table 269. LN & SQRT IXOL mode data address register (LN_SQRT_IXOL, offset: 0x154) bit description. . 239
Table 270. LN & SQRT ILOX mode data address register (LN_SQRT_ILOX, offset: 0x158) bit description. . 240
Table 271. LN & SQRT ILOL mode data address register (LN_SQRT_ILOL, offset: 0x15C) bit description. . 240
Table 272. Cordic T0/1 UP IXOX mode data address register (CORDIC_T0/1UP_IXOX, offset = 0x160/0x180) bit description 240
Table 273. Cordic T0/1 UP IXOL mode data address register (CORDIC_T0/1UP_IXOL, offset = 0x164/0x184) bit description 241
Table 274. Cordic T0/1 UP ILOX mode data address register (CORDIC_T0/1UP_ILOX, offset = 0x168/0x188) bit description 241
Table 275. Cordic T0/1 UP ILOL mode data address register (CORDIC_T0/1UP_ILOL, offset = 0x17C/0x18C) bit description. 242
Table 276. Cordic T0/1 UN IXOX mode data address register (CORDIC_T0/1UN_IXOX, offset = 0x170/0x190) bit description 242
Table 277. Cordic T0/1 UN IXOL mode data address register (CORDIC_T0/1UN_IXOL, offset = 0x174/0x194) bit description 243
Table 278. Cordic T0/1 UN ILOX mode data address register (CORDIC_T0/1UN_ILOX, offset = 0x178/0x198) bit description 243
Table 279. Cordic T0/1 UN ILOL mode data address register (CORDIC_T0/1UN_ILOL, offset = 0x17C/0x19C) bit description 244
Table 280. USB pull-up resister connection control 245
Table 281. USB pull-up resistor strength control. 246
Table 282. Fixed endpoint configuration 248
Table 283. USB device pin description 250
Table 284. Register overview: USB (base address 0x4008 4000) 251
Table 285. USB Device Command/Status register (DEVCMDSTAT, offset 0x00) bit description . 251
Table 286. USB info register (INFO, offset 0x04) bit description 253
Table 287. USB EP Command/Status List start address (EPLISTSTART, offset 0x08) bit description. . 254
Table 288. USB Data buffer start address (DATABUFSTART, offset 0x0C) bit description . . 254
Table 289. Link Power Management register (LPM, offset 0x10) bit description. 255
Table 290. USB Endpoint skip (EPSKIP, offset 0x14) bit description 255
Table 291. USB Endpoint Buffer in use (EPINUSE, offset 0x18) bit description. 255
Table 292. USB Endpoint Buffer Configuration (EPBUFCFG, offset 0x1C) bit description. 256
Table 293. USB interrupt status register (INTSTAT, offset 0x20) bit description. 256
Table 294. USB interrupt enable register (INTEN, offset 0x24) bit description. 258
Table 295. USB set interrupt status register (INTSETSTAT, offset 0x28) bit description 259
Table 296. USB Endpoint toggle (EPTOGGLE, offset 0x34) bit description. 259
Table 297. Endpoint commands 260
Table 298: Flexcomm Interface base addresses and functions. 266
Table 299: Flexcomm Interface Pin Description 268
Table 300: Register map for the first channel pair within one Flexcomm Interface .268
Table 301. IO mode register (IOMODE - offset 0xF00) bit description 269
Table 302. Peripheral Select and Flexcomm Interface ID register (PSELID - offset 0xFF8) bit description. . 270
Table 303. Peripheral identification register (PID - offset 0xFFC) bit description 271
Table 304. USART pin description 274
Table 305: USART pin configuration example 274

continued >>

Table 306: USART register overview 276
Table 307. USART Configuration register (CFG, offset 0x000) bit description 277
Table 308. USART Control register (CTL, offset 0x004) bit description 279
Table 309. USART status register (STAT, offset 0x008) bit description 280
Table 310. USART interrupt enable read and set register (INTENSET, offset 0x00C) bit description . . . 281
Table 311. USART interrupt enable clear register (INTENCLR, offset 0x010) bit description . . . 282
Table 312. USART baud rate generator register (BRG, offset 0x020) bit description 283
Table 313. USART interrupt status register (INTSTAT, offset 0x024) bit description 283
Table 314. Oversample selection register (OSR, offset 0x028) bit description 284
Table 315. Address register (ADDR, offset 0x02C) bit description 284
Table 316. FIFO configuration register (FIFOCFG - offset 0xE00) bit description 285
Table 317. FIFO status register (FIFOSTAT - offset 0xE04) bit description 285
Table 318. FIFO trigger level settings register (FIFOTRIG - offset 0xE08) bit description 286
Table 319. FIFO interrupt enable set and read register (FIFOINTENSET - offset 0xE10) bit description . 287
Table 320. FIFO interrupt enable clear and read (FIFOINTENCLR - offset 0xE14) bit description . 288
Table 321. FIFO interrupt status register (FIFOINTSTAT - offset 0xE18) bit description 288
Table 322. FIFO write data register (FIFOWR - offset 0xE20) bit description 288
Table 323. FIFO read data register (FIFORD - offset 0xE30) bit description 289
Table 324. FIFO data read with no FIFO pop (FIFORDNOPOP - offset 0xE40) bit description. . 289
Table 325. Module identification register (ID - offset 0xFFC) bit description 289
Table 326: SPI Pin Description 295
Table 327: SPI pin configuration example 296
Table 328. SPI register overview 297
Table 329. SPI Configuration register (CFG, offset 0x400) bit description 298
Table 330. SPI Delay register (DLY, offset 0x404) bit description 299
Table 331. SPI Status register (STAT, offset 0x408) bit description 300
Table 332. SPI Interrupt Enable read and Set register (INTENSET, offset 0x40C) bit description . . . 301
Table 333. SPI Interrupt Enable clear register (INTENCLR, offset 0x410) bit description 301
Table 334. SPI Divider register (DIV, offset 0x424) bit description 302
Table 335. SPI Interrupt Status register (INTSTAT, offset 0x428) bit description 302
Table 336. FIFO Configuration register (FIFOCFG - offset 0xE00) bit description. 303
Table 337. FIFO status register (FIFOSTAT - offset 0xE04) bit description 303
Table 338. FIFO trigger settings register (FIFOTRIG - offset 0xE08) bit description. 304
Table 339. FIFO interrupt enable set and read register (FIFOINTENSET - offset 0xE10) bit description. . 305
Table 340. FIFO interrupt enable clear and read (FIFOINTENCLR - offset 0xE14) bit description . 306
Table 341. FIFO interrupt status register (FIFOINTSTAT - offset 0xE18) bit description. 306
Table 342. FIFO write data register (FIFOWR - offset 0xE20) bit description 307
Table 343. FIFO read data register (FIFORD - offset 0xE30) bit description 308
Table 344. FIFO data read with no FIFO pop (FIFORDNOPOP - offset 0xE40) bit description . 309
Table 345. Module identification register (ID - offset 0xFFC) bit description 309
Table 346: SPI mode summary 310
Table 347. I^{2}C-bus pin description 318
Table 348: USART pin configuration example 318
Table 349. Code Example 320
Table 350. Code Example 321
Table 351. Code Example 322
Table 352. Code Example 322
Table 353. I^{2}C register overview 324
Table 354. I^{2}C Configuration register (CFG, offset 0x800) bit description 325
Table 355. I^{2}C Status register (STAT, offset 0x804) bit description 326
Table 356. Master function state codes (MSTSTATE) . . 329
Table 357. Slave function state codes (SLVSTATE) 330
Table 358. Interrupt Enable Set and read register (INTENSET, offset 0x808) bit description. . . . 330
Table 359. Interrupt Enable Clear register (INTENCLR, offset 0x80C) bit description 331
Table 360. Time-out value register (TIMEOUT, offset 0x810) bit description 332

continued >>

Table 361. I2C Clock Divider register (CLKDIV, offset 0x814) bit description . . . 333
Table 362. I2C Interrupt Status register (INTSTAT, offset 0x818) bit description . . . 333
Table 363. Master Control register (MSTCTL, offset 0x820) bit description . . . 334
Table 364. Master Time register (MSTTIME, offset 0x824) bit description . . . 335
Table 365. Master Data register (MSTDAT, offset 0x828) bit description . . . 336
Table 366. Slave Control register (SLVCTL, offset 0x840) bit description . . . 336
Table 367. Slave Data register (SLVDAT, offset 0x844) bit description . . . 337
Table 368. Slave Address 0 register (SLVADR[0], offset 0x848) bit description . . . 338
Table 369. Slave Address registers (SLVADR[1:3], offset [0x84C:0x854]) bit description . . . 338
Table 370. Slave address Qualifier 0 register (SLVQUAL0, offset 0x858) bit description . . . 339
Table 371. Monitor data register (MONRXDAT, offset 0x880) bit description . . . 339
Table 372. Module identification register (ID - offset 0xFFC) bit description . . . 340
Table 374. QDEC pin description . . . 346
Table 375. Sampled value decoding . . . 348
Table 376. QDEC register base address . . . 349
Table 377. QDEC Register overview for one QDEC. . . 349
Table 378. QDEC Configuration and Control register (CTRL, offset: 0x00) bit description . . . 350
Table 379. QDEC Sample Control register (SAMP_CTL, offset=0x04) bit description . . . 351
Table 380. QDEC Sample Value register (SAMPLE - offset 0x008) bit description . . . 352
Table 381. QDEC Valid Accumulator register (ACC - offset 0x0C) bit description . . . 352
Table 382. ACC snapshot register (ACC_R - offset 0x010) bit description . . . 352
Table 383. QDEC Invalid Accumulator register (DB - offset 0x14) bit description . . . 353
Table 384. DB snapshot register (DB_R - offset 0x018) bit description . . . 353
Table 385. QDEC Interrupt register (INT - offset 0x1C) bit description . . . 353
Table 386. QDEC Interrupt Enable register (INTEN- offset 0x20) bit description . . . 353
Table 387. QDEC Status register (STAT - offset 0x24) bit description . . . 353
Table 388. SPIFI flash memory map . . . 358
Table 389. SPIFI Pin description . . . 358
Table 390. Register overview: SPIFI (base address 0x4008 0000) . . . 359
Table 391. SPIFI control register (CTRL, offset 0x000) bit description . . . 359
Table 392. SPIFI command register (CMD, offset 0x004) bit description . . . 361
Table 393. SPIFI address register (ADDR, offset 0x008) bit description . . . 362
Table 394. SPIFI intermediate data register (IDATA, offset 0x00C) bit description . . . 362
Table 395. SPIFI cache limit register (CLIMIT, offset 0x010) bit description . . . 362
Table 396. SPIFI Data register (DATA, offset 0x014) bit description . . . 363
Table 397. SPIFI memory command register (MCMD, offset 0x018) bit description . . . 363
Table 398. SPIFI status register (STAT, offset 0x01C) bit description . . . 364
Table 399. RNG registers overview (base address 0x4000 7C00) . . . 370
Table 400. RNG control register (CTRL, offset=0x00) bit description . . . 370
Table 401. RNG status register (STAT, offset=0x04) bit description . . . 371
Table 402. RNG Output Data register (DATA, offset=0x08) bit description . . . 371
Table 403. Interrupt Status register (INT, offset=0x0C) bit description . . . 371
Table 404. Interrupt Enable register (INTEN, offset=0x10) bit description . . . 371
Table 405. Register overview: CRC engine (base address 0x4009 5000) . . . 373
Table 406. CRC mode register (MODE, offset 0x00) bit description . . . 373
Table 407. CRC seed register (SEED, offset 0x04) bit description . . . 374
Table 408. CRC checksum register (SUM, offset 0x08) bit description . . . 374
Table 409. CRC data register (WR_DATA, offset 0x08) bit description . . . 374
Table 410. Flash lock and protect description. . . 378
Table 411. SWD behavior for different lock and protect scheme. . . 378
Table 412. Firmware behavior for different lock and protect scheme. . . 379
Table 413. Flash Registers overview (base address 0x4008 1000) . . . 379
Table 414. Flash initial read enable register (INI_RD_EN) bit description . . . 380
Table 415. Flash erase control register (ERASE_CTRL) bit description . . . 380
Table 416. Flash erase time setting register

continued >>

(ERASE_TIME) bit description 381
Table 417. Flash operation time setting register (TIME_CTRL) bit description 381
Table 418. Smart erase control register (SMART_CTRL)bit description 381
Table 419. Interrupt enable register (INTEN) bit description 382
Table 420. Interrupt status register (INT_STAT) bit description 383
Table 421. Interrupt clear register (INT_CLR) bit description 383
Table 422. Lock control register x (LOCK_CTRL_x) bit description 384
Table 423. Lock control register 8(LOCK_CTRL_8) bit description 384
Table 424. Status register 1(STATUS1)bit description . .385
Table 425. Flash block 0 error information register 1(ERR_INFOL_1) bit description 385
Table 426. Flash block 0 error information register 2(ERR_INFOL_2) bit description 385
Table 427. Flash block 0 error information register 3(ERR_INFOL_3) bit description 386
Table 428. Flash block 1 error information register 1(ERR_INFOH_1) bit description 386
Table 429. Flash block 1 error information register 2(ERR_INFOH_2) bit description 386
Table 430. Flash block 1 error information register 3(ERR_INFOH_3) bit description 386
Table 431. SWD erase password register (PASSWORD) bit description 386
Table 432. Erase password register (ERASE_PASSWORD) bit description 387
Table 433: Pin description 399
Table 434: FIFO bit selection. 402
Table 435. Modulator output bit width settings 402
Table 436. Up sample ratio (Nup) 403
Table 437. Bit selection in FIFO. 404
Table 438. Input data format selection. 404
Table 439. Full scale voltage settings 404
Table 440. Full scale voltage settings 405
Table 441. Filter bandwidth settings 405
Table 442. DACA pin load settings 405
Table 443. DAC trigger source selection 407
Table 444. DAC trigger edge selection 407
Table 445. DAC register overview 408
Table 446. DAC analog configuration register (ANG_CFG, offset: 0x00) bit description 409
Table 447. DAC control register (CTRL, offset: 0x04) bit description 409
Table 448. Sin-wave configuration register 0 (SIN_CFG0, offset 0x08) bit description 410
Table 449. Sin-wave configuration register 1(SIN_CFG1, offset 0x0C) bit description 411
Table 450. Gain control register (GAIN_CTRL, offset 0x10) bit description 411
Table 451. Clear trigger register (CLR_TRG, offset 0x14) bit description 411
Table 452. DAC data input register (DIN, offset 0x18) bit description 411
Table 453. DAC interrupt register (INT, offset 0x1C) bit description 411
Table 454. DAC interrupt enable register (INTEN, offset 0x20) bit description 412
Table 455. DAC interrupt status register (INT_STAT, offset 0x24) bit description 412
Table 456. DAC status register (STATUS, offset 0x28) bit description 412
Table 457. ADC pins 413
Table 458. ADC input multiplex .416
Table 459. ADC source clock selection. 418
Table 460. Conversion mode 418
Table 461. channel configuration registers 419
Table 462. ADC power control .420
Table 463. ADC trigger sources 420
Table 464. ADC output data format. 421
Table 465. Register overview: ADC (base address 0x4000 7000) 423
Table 466. ADC Control register (CTRL, offset 0x0) bit description 423
Table 467. ADC channel select register (CH_SEL, offset 0x04) bit description. 425
Table 468. ADC channel configuration register (CH_CFG, offset 0x08) bit description 425
Table 469. Window compare threshold register (WCMP_THR, offset 0x0C) bit description. . .425
Table 470. ADC interrupt mask register (INTEN, offset 0x10) bit description. 425
Table 471. ADC interrupt status register (INT, offset 0x14) bit description 425
Table 472. ADC converted data register (DATA, offset 0x18) bit description. 426
Table 473. ADC configuration 0 register (CFG0, offset 0x20) bit description. 426
Table 474. ADC configuration 1register (CFG1, offset 0x24) bit description. 427
Table 475. ADC bandgap and buffer setting register (BG_BF, offset 0x28) bit description 429
Table 476. ADC core and reference setting register (ANA_CTRL, offset 0x2C) bit description. . . .429
Table 477. Capacitive Sense pin description 433
Table 478. Active mode vs. LP mode feature Comparison . 435

continued >>

Table 479. Capacitive Sense registers overview (base address 0x4000 7800) 436
Table 480. Capacitive Sense control register 0 (CTRL0, offset=0x00) bit description 436
Table 481. Capacitive Sense control register 1(CTRL1, offset=0x04) bit description. 437
Table 482. Interrupt Status register (INTSTAT, offset=0x08) bit description. 437
Table 483. Interrupt Mask register (INTEN, offset=0x0C) bit description 437
Table 484. Output Data register (FIFODATA, offset=0x10) bit description. 438
Table 485. Low Power mode control register (LP_CTRL, offset=0x14) bit description. 438
Table 486. Low Power Interrupt register (LP_INTSTAT, offset=0x18) bit description. 438
Table 487. low power interrupt mask register (LP_INTEN, offset=0x1C) bit description 438
Table 488. Idle period number register (IDLE_PERIOD, offset=0x20) bit description. 439
Table 489: ACMP Pin Description 439
Table 490. Comparator input configuration 440
Table 491. Abbreviations 442

continued >>

38. Figures

Fig 1. HVQFN48 Pin configuration 6
Fig 2. WLCSP pin 6
Fig 3. System architecture 8
Fig 4. Main memory map 45
Fig 5. APB memory map 46
Fig 6. LDO and DC-to-DC 58
Fig 7. BOR/BOR block diagram 59
Fig 8. Power mode state machine 61
Fig 9. Clock generation 64
Fig 10. 32K Clock Output 66
Fig 11. High Frequency Clock Output 67
Fig 12. Flash arrangement 69
Fig 13. Startup flowchart 71
Fig 14. Fast boot flowchart 72
Fig 15. ISP flowchart 72
Fig 16. ISP PDU format 73
Fig 17. ISP PDU interaction 75
Fig 18. Pin configuration 79
Fig 19. Pin Mux Diagram (need to be updated) 81
Fig 20. Pin interrupt multiplexing 85
Fig 21. DMA trigger multiplexing 85
Fig 22. Pin interrupt connections 94
Fig 23. Pattern match engine connections 96
Fig 24. Pattern match bit slice with detect logic 97
Fig 25. Pattern match engine examples: sticky edge detect 108
Fig 26. Pattern match engine examples: Windowed non-sticky edge detect evaluates as true 109
Fig 27. Pattern match engine examples: Windowed non-sticky edge detect evaluates as false 109
Fig 28. DMA block diagram 111
Fig 29. SCT clocking (the AHBCLKCTRL1[2] should be replaced with CLK_PWM_EN) 132
Fig 30. SCT connections 132
Fig 31. SCTimer/PWM block diagram 135
Fig 32. SCTimer/PWM counter and select logic 135
Fig 33. SCT event configuration and selection registers 138
Fig 34. Match logic 158
Fig 35. Capture logic 159
Fig 36. Event selection 159
Fig 37. Output slice i 160
Fig 38. SCT interrupt generation 160
Fig 39. SCT configuration example 166
Fig 40. 32-bit counter/timer block diagram 170
Fig 41. A timer cycle in which PR=2, MRx=6, and both interrupt and reset on match are enabled 180
Fig 42. A timer cycle in which PR=2, MRx=6, and both interrupt and stop on match are enabled 180
Fig 43. Sample PWM waveforms with a PWM cycle length of 100 (selected by MR3) and MAT3:0 enabled as PWM outputs by the PWCON register 181
Fig 44. Watchdog timer block diagram 184
Fig 45. RTC clocking 189
Fig 46. RTC Block Diagram 190
Fig 47. RTC Counter Unit 191
Fig 48. System tick timer block diagram 195
Fig 49. 208
Fig 50. FP conversion module 210
Fig 51. USB block diagram 245
Fig 52. USB software interface 246
Fig 53. Endpoint command/status list (see also Table 297)(Lines should be added in the figure) 258
Fig 54. Flowchart of control endpoint 0 - OUT direction 260
Fig 55. Flowchart of control endpoint 0 - IN direction . . 261
Fig 56. Flexcomm Interface block diagram 265
Fig 57. USART block diagram 273
Fig 58. Hardware flow control using RTS and CTS 289
Fig 59. SPI block diagram 294
Fig 60. Basic SPI operating modes 308
Fig 61. Pre_delay and Post_delay 309
Fig 62. Frame_delay 310
Fig 63. Transfer_delay 311
Fig 64. Examples of data stalls 315
Fig 65. I2C block diagram 322
Fig 66. QDEC block diagram 345
Fig 67. Opcode only commands 364
Fig 68. Read commands 364
Fig 69. RNG block diagram 367
Fig 70. CRC block diagram 371
Fig 71. Flash physical organization 375
Fig 72. Flash write example 1 386
Fig 73. Flash write example 2 389
Fig 74. Flash write example 3 391
Fig 75. Flash write example 4 393
Fig 76. Flash erase example 1 394
Fig 77. Flash erase example 2 395
Fig 78. basic configuration 396
Fig 79. DAC block diagram 398
Fig 80. Gain control 400
Fig 81. ADC block diagram 413
Fig 82. Capacitive Sense Block Diagram 432
Fig 83. Operation modes 433
Fig 84. ACMP block diagram . . . . . . 438Fig 69. RNG block diagram 367
Fig 70. CRC block diagram 371
Fig 71. Flash physical organization 375

continued >>

Fig 72. Flash write example 1 386
Fig 73. Flash write example 2 389
Fig 74. Flash write example 3 391
Fig 75. Flash write example 4 393
Fig 76. Flash erase example 1. .394
Fig 77. Flash erase example 2. .395
Fig 78. basic configuration. 396
Fig 79. DAC block diagram 398
Fig 80. Gain control 400
Fig 81. ADC block diagram 413
Fig 82. Capacitive Sense Block Diagram. 432
Fig 83. Operation modes 433
Fig 84. ACMP block diagram 438

39. Contents

2.4.1 Reset sources 9
2.5 System Register Description 11
2.5.1 Software Reset Set Register 12
2.5.2 Software Reset Clear register 13
2.5.3 Clock Disable Register 14
2.5.4 Clock Enable Register 16
2.5.5 System Clock Source and Divider register 18
2.5.6 System Mode and Address Remap register 18
2.5.7 System status register 19
2.5.8 Systick timer control register 19
2.5.9 Exchange memory base address register 20
2.5.10 Chip ID register 20
2.5.11 Crystal and PA register 21
2.5.12 Crystal Control register 22
2.5.13 Buck control register 22
2.5.14 Flexcomm Clock Divider register 23
2.5.15 Pad Pull Control 0 register 23
2.5.16 Pad Pull Control 1 register 24
2.5.17 Pad Pull Control 2 register 24
2.5.18 Sleep Capture register 25
2.5.19 Pad Drive Strength 0 register 25
2.5.20 Pad Drive Strength 1 register 26
2.5.21 Pad Drive Extra register 26
2.5.22 Pad Misc Control register 26
2.5.23 Pin Wake-up Polarity register 0 27
2.5.24 Pin Wake-up Polarity register 1 28
2.5.25 Pad Input Enable register 0 28
2.5.26 Pad Input Enable register 1 29
2.5.27 Pin Mux Control register 0 29
2.5.28 Pin Mux Control register 1 30
2.5.29 Pin Mux Control register 2 30
2.5.30 Pin Mux Control register 3 31
2.5.31 Pin Function Selection in Sleep Mode register 0 32
2.5.32 Pin Function Selection in Sleep Mode register 1 33
2.5.33 Pin captured output enable status register 0 34
2.5.34 Pin output enable status register 1 35
2.5.35 Pin captured status register 0 35
2.5.36 Pin captured output register 1 36
2.5.37 Reset Source Status register 36
2.5.38 Power Management Unit Control register 0 36
2.5.39 Power Management Unit Control register 1 37
2.5.40 Analog Setting Register 38
2.5.41 32K Crystal Control register 39
2.5.42 USB Configuration register 39
2.5.43 Power management unit control register 2 40
2.5.44 IVREF and DVREG setting register 40
2.5.45 MISC register 41
3 Memory map 42
3.1 General description 42
3.1.1 ROM 42
3.1.2 SRAM 42
3.1.3 Flash 42
3.1.4 SPI flash interface (SPIFI) 43
3.1.5 Bit-band addressing 43
3.1.6 Memory mapping 43
3.1.7 Memory Remap 45
3.1.8 Memory Protection Unit (MPU) 45
4 Nested Vectored Interrupt Controller (NVIC) 46
4.1 Introduction 46
4.2 Features 46
4.3 General description 46
4.3.1 Interrupt sources 46
4.4 Register description 48
4.4.1 Interrupt Set-Enable Register 0 register 50
4.4.2 Interrupt Set-Enable Register 1 register 50
4.4.3 Interrupt Clear-Enable Register 0 50
4.4.4 Interrupt Clear-Enable Register 1 register 50
4.4.5 Interrupt Set-Pending Register 0 register 51
4.4.6 Interrupt Set-Pending Register 1 register 51
4.4.7 Interrupt Clear-Pending Register 0 register 51
4.4.8 Interrupt Clear-Pending Register 1 register 51
4.4.9 Interrupt Active Bit Register 0 52
4.4.10 Interrupt Active Bit Register 1 52
4.4.11 Interrupt Priority Register 0 52
4.4.12 Interrupt Priority Register 1 52
4.4.13 Interrupt Priority Register 2 53
4.4.14 Interrupt Priority Register 3 53
4.4.15 Interrupt Priority Register 4 53
4.4.16 Interrupt Priority Register 5 54
4.4.17 Interrupt Priority Register 6 54
4.4.18 Interrupt Priority Register 7 54
4.4.19 Interrupt Priority Register 8 55
4.4.20 Interrupt Priority Register 9 55
4.4.21 Interrupt Priority Register 10 55
4.4.22 Interrupt Priority Register 11 56
4.4.23 Interrupt Priority Register 12 56
4.4.24 Software Trigger Interrupt Register 56
5 Power management unit 57
5.1 Introduction 57
5.2 General description 57
5.2.1 Power Supply 57
5.2.2 Power Supervisor (BOD/BOR) 57
5.2.3 Power Mode 58

continued >>

5.2.4 Power mode state machine . . . 59
5.2.5 Wake-up process . . . 60
6 Clock management unit . . . 61
6.1 Introduction . . . 61
6.2 Basic configuration . . . 61
6.2.1 System clock configuration . . . 61
6.2.2 Clock Gating . . . 61
6.3 Pin description . . . 61
6.4 General description . . . 61
6.4.1 Clock Source . . . 61
6.4.2 Clock generation . . . 62
6.5 Register description . . . 64
6.6 Functional description . . . 64
6.6.1 Start-up behavior . . . 64
6.6.2 Configure BLE clock . . . 64
6.6.3 Configure AHB and APB clock . . . 64
6.6.4 Clock Output . . . 65
7 Boot process . . . 67
7.1 Features . . . 67
7.2 Pin description . . . 67
7.3 General description . . . 67
7.3.1 Boot mode . . . 67
7.3.2 Flash layout . . . 68
7.4 Boot process . . . 69
7.4.1 Startup flow chart . . . 69
7.4.2 Fast boot flowchart . . . 71
7.4.3 ISP flowchart . . . 71
7.5 ISP . . . 72
7.5.1 ISP transportation layer . . . 72
7.5.2 ISP protocol data unit (PDU) format . . . 72
7.5.3 ISP commands . . . 73
7.5.4 ISP PDU interaction . . . 74
7.5.5 ISP PDU description . . . 75
8 I/O configuration and pin mux . . . 77
8.1 Introduction . . . 77
8.2 Features . . . 77
8.3 General description . . . 77
8.3.1 Pin configuration . . . 77
8.3.2 Pin Mux . . . 79
8.3.3 IO status under Power down . . . 82
8.4 Register description . . . 82
9 Input Multiplexing (Input Mux) . . . 83
9.1 Introduction . . . 83
9.2 Features . . . 83
9.3 Basic configuration . . . 83
9.4 Pin description . . . 83
9.5 General description . . . 83
9.5.1 Pin interrupt input multiplexing . . . 83
9.5.2 DMA trigger input multiplexing . . . 84
9.6 Register description . . . 85
9.6.1 Pin interrupt select registers . . . 85
9.6.2 DMA trigger input mux registers 0 to 19 . . . 86
9.6.3 DMA output trigger feedback mux registers 0 to 3 87
10 General purpose I/O . . . 88
10.1 Introduction . . . 88
10.2 Basic configuration . . . 88
10.3 Features . . . 88
10.4 General description . . . 88
10.5 Register description . . . 88
10.5.1 DATA register . . . 89
10.5.2 DATAOUT register . . . 89
10.5.3 GPIO Port Output enable set register . . . 89
10.5.4 GPIO Port Output enable clear register . . . 89
10.5.5 GPIO Port Interrupt Enable set registers . . . 89
10.5.6 GPIO port Interrupt Enable clear registers . . . 90
10.5.7 GPIO Interrupt type set registers . . . 90
10.5.8 GPIO Interrupt type clear registers . . . 90
10.5.9 GPIO Interrupt Polarity set registers . . . 90
10.5.10 GPIO Interrupt Polarity clear registers . . . 90
10.5.11 GPIO Interrupt status registers . . . 90
10.6 Functional description . . . 91
10.6.1 General Purpose I/O . . . 91
10.6.2 External interrupt/wakeup lines . . . 91
11 Pin interrupt and pattern match (PINT) . . . 92
11.1 Introduction . . . 92
11.2 Features . . . 92
11.3 Basic configuration . . . 92
11.3.1 Configure pins as pin interrupts or as inputs to the pattern match engine . . . 92
11.4 Pin description . . . 93
11.5 General description . . . 93
11.5.1 Pin interrupts . . . 93
11.5.2 Pattern match engine . . . 93
11.6 Register description . . . 97
11.6.1 Pin interrupt mode register . . . 97
11.6.2 Pin interrupt level or rising edge interrupt enable register . . . 97
11.6.3 Pin interrupt level or rising edge interrupt set register . . . 98
11.6.4 Pin interrupt level or rising edge interrupt clear register . . . 98
11.6.5 Pin interrupt active level or falling edge interrupt enable register . . . 98
11.6.6 Pin interrupt active level or falling edge interrupt set register . . . 99
11.6.7 Pin interrupt active level or falling edge interrupt clear register . . . 99
11.6.8 Pin interrupt rising edge register . . . 99
11.6.9 Pin interrupt falling edge register . . . 100

continued >>

11.6.10 Pin interrupt status register 100
11.6.11 Pattern Match Interrupt Control Register . . . 101
11.6.12 Pattern Match Interrupt Bit-Slice Source register 101
11.6.13 Pattern Match Interrupt Bit Slice Configuration register 102
11.7 Functional description 106
11.7.1 Pin interrupts 106
11.7.2 Pattern Match engine example 106
11.7.3 Pattern match engine edge detect examples 107
12 DMA controller 109
12.1 Introduction 109
12.2 Features 109
12.3 Basic configuration 109
12.4 Pin description 110
12.5 General description 110
12.5.1 DMA requests and triggers 110
12.5.2 DMA Modes 113
12.5.3 Single buffer 114
12.5.4 Ping-Pong 114
12.5.5 Linked transfers (linked list) 115
12.5.6 Address alignment for data transfers 115
12.5.7 Channel chaining 115
12.5.8 DMA in reduced power modes 116
12.6 Register description 116
12.6.1 Control register 119
12.6.2 Interrupt Status register 120
12.6.3 SRAM Base address register 120
12.6.4 Enable read and Set registers 121
12.6.5 Enable Clear register 121
12.6.6 Active status register 122
12.6.7 Busy status register 122
12.6.8 Error Interrupt register 122
12.6.9 Interrupt Enable read and Set register 123
12.6.10 Interrupt Enable Clear register 123
12.6.11 Interrupt A register 123
12.6.12 Interrupt B register 123
12.6.13 Set Valid register 124
12.6.14 Set Trigger register 124
12.6.15 Abort register 124
12.6.16 Channel configuration registers 125
12.6.17 Channel control and status registers 127
12.6.18 Channel transfer configuration registers . . . 127
13 SCTimer/PWM (SCT) 130
13.1 How to read this chapter 130
13.2 Features 130
13.3 Basic configuration 131
13.4 Pin description 132
13.5 General description 132
13.6 Register description 134
13.6.1 Register functional grouping 136
13.6.2 SCT configuration register 139
13.6.3 SCT control register 141
13.6.4 SCT limit event select register 143
13.6.5 SCT halt event select register 144
13.6.6 SCT stop event select register 144
13.6.7 SCT start event select register 145
13.6.8 SCT counter register 145
13.6.9 SCT state register 146
13.6.10 SCT input register 147
13.6.11 SCT match/capture mode register 147
13.6.12 SCT output register 148
13.6.13 SCT bi-directional output control register . . . 148
13.6.14 SCT conflict resolution register 149
13.6.15 SCT DMA request 0 and 1 registers 150
13.6.16 SCT event interrupt enable register 151
13.6.17 SCT event flag register 151
13.6.18 SCT conflict interrupt enable register 152
13.6.19 SCT conflict flag register 152
13.6.20 SCT match registers 0 to 9 (REGMODEn bit: 0) . 152
13.6.21 SCT capture registers 0 to 9 (REGMODEn bit: 1) 153
13.6.22 SCT match reload registers 0 to 9 (REGMODEn bit: 0) 153
13.6.23 SCT capture control registers 0 to 9 (REGMODEn bit: 1) 153
13.6.24 SCT event enable registers 0 to 9 154
13.6.25 SCT event control registers 0 to 9 154
13.6.26 SCT output set registers 0 to 7 156
13.6.27 SCT output clear registers 0 to 7 156
13.7 Functional description 157
13.7.1 Match logic 157
13.7.2 Capture logic 158
13.7.3 Event selection 158
13.7.4 Output generation 158
13.7.5 State logic 159
13.7.6 Interrupt generation 159
13.7.7 Clearing the prescaler 160
13.7.8 Match vs. I/O events 160
13.7.9 SCT operation 160
13.7.10 Configure the SCT 161
13.7.11 Run the SCT 163
13.7.12 Configure the SCT without using states 164
13.7.13 SCT PWM Example 164
14 Standard counter/timers (CTIMER 0 to 3) . . 167
14.1 Introduction 167
14.2 Basic configuration 167
14.3 Features 167
14.4 General description 168

continued >>

14.4.1 Capture inputs 168
14.4.2 Match outputs 168
14.4.3 Applications. 168
14.4.4 Architecture 168
14.5 Pin description 170
14.5.1 Multiple CAP and OUT pins 170
14.6 Register description 171
14.6.1 Interrupt Register 172
14.6.2 Timer Control Register 172
14.6.3 Timer Counter registers 172
14.6.4 Prescale register 173
14.6.5 Prescale Counter register 173
14.6.6 Match Control Register 173
14.6.7 Match Registers 174
14.6.8 Capture Control Register 174
14.6.9 Capture Registers. 175
14.6.10 External Match Register 175
14.6.11 Count Control Register 177
14.6.12 PWM Control Register 178
14.7 Functional description. 179
14.7.1 Rules for single edge controlled PWM outputs. . . 179
14.7.2 DMA operation 180
15 Watchdog Timer (WDT) 182
15.1 Introduction. 182
15.2 Features 182
15.3 Basic configuration 182
15.4 Pin description 182
15.5 General description. 182
15.5.1 Block diagram 182
15.5.2 WDT reset. 183
15.5.3 Using the WDT lock features 183
15.5.4 Programming the timeout interval. 184
15.6 Register description 185
15.6.1 WDT LOAD register 185
15.6.2 WDT VALUE register 185
15.6.3 WDT CTRL register 185
15.6.4 WDT Interrupt Clear register. 186
15.6.5 WDT RAW Interrupt Status register 186
15.6.6 WDT Masked Interrupt Status register 186
15.6.7 WDT LOCK register 186
16 Real-Time Clock (RTC) 187
16.1 Introduction. 187
16.2 Features 187
16.3 Basic configuration 187
16.4 Pin Description 188
16.5 General description. 189
16.5.1 RTC Count Unit 189
16.5.2 Free-Running counter unit 191
16.6 Register description 191
16.6.1 RTC CTRL register. 192
16.6.2 RTC Status register 192
16.6.3 RTC Second register 192
16.6.4 RTC Calibration register. 193
16.6.5 RTC Count Value register 193
16.6.6 RTC Free running control register 193
16.6.7 RTC Interrupt Threshold register 193
16.6.8 RTC Reset Threshold register 193
16.6.9 RTC Freerunning Count Value register 193
17 CPU system tick timer (SYSTICK) 194
17.1 How to read this chapter 194
17.2 Basic configuration. 194
17.3 Features. 194
17.4 General description 194
17.5 Register description 196
17.5.1 System Timer Control and status register . . 196
17.5.2 System Timer Reload value register 196
17.5.3 System Timer Current value register 197
17.5.4 System Timer Calibration value register . . . 197
17.6 Functional description 197
17.7 Example timer calculations 197
18 Fusion Sensor Processing (FSP). 199
18.1 Introduction 199
18.2 Features. 199
18.3 Basic configuration. 200
18.4 Pin description 202
18.5 General description 203
18.5.1 Cordic. 203
18.5.2 MOU 207
18.5.3 TE. 208
18.5.4 Statistics Engine (SE) 209
18.5.5 Correlation Module (COR) 210
18.5.6 Multi-channel Finite Impulse Response (FIR) Filter 211
18.6 Function Description 212
18.6.1 CORDIC. 212
18.6.2 Matrix Operations Unit (MOU) 215
18.6.3 Transform Engine (TE). 217
18.6.4 Statistics Engine (SE) 219
18.6.5 COR 221
18.6.6 Finite Impulse Response (FIR) Filter 221
18.7 Register description 222
18.7.1 FSP system control register. 225
18.7.2 FSP status register. 226
18.7.3 FSP interrupt register. 226
18.7.4 FSP interrupt enable register 227
18.7.5 FSP Transform Engine control register 229
18.7.6 TE source data memory base register 230
18.7.7 TE destination data memory base register. . 230
18.7.8 FSP MOU control register 230

continued >>

18.7.9 Matrix A/B source data memory base register . . . 231
18.7.10 Matrix output data memory base register . . . 231
18.7.11 MOU scale coefficient A/B register 231
18.7.12 FSP SE control register 232
18.7.13 SE source data memory base register 232
18.7.14 SE max or min data index register 233
18.7.15 SE array summary result register 233
18.7.16 SE array power result register 233
18.7.17 FSP COR control register 233
18.7.18 Correlation X/Y sequence base register 234
18.7.19 Correlation output sequence base register . . 234
18.7.20 Correlation offset register 234
18.7.21 FIR channel 0 configuration register 234
18.7.22 FIR channel [1,8]configuration register 235
18.7.23 FIR channel x fix-point data register 235
18.7.24 FIR channel x float data register 235
18.7.25 Sin & cos IXOX mode data address register 235
18.7.26 Sin & cos IXOL mode data address register. 236
18.7.27 Sin & cos ILOX mode data address register. 236
18.7.28 Sin & cos ILOL mode data address register . 236
18.7.29 LN &SQRT IXOX mode data address register . . . 237
18.7.30 LN &SQRT IXOL mode data address register . . . 237
18.7.31 LN &SQRT ILOX mode data address register . . . 237
18.7.32 LN &SQRT ILOL mode data address register 238
18.7.33 Cordic T0/1UP IXOX mode data address register 238
18.7.34 Cordic T0/1UP IXOL mode data address register 239
18.7.35 Cordic T0/1UP ILOX mode data address register 239
18.7.36 Cordic T0/1UP ILOL mode data address register. 240
18.7.37 Cordic T0/1UN IXOX mode data address register 240
18.7.38 Cordic T0/1UN IXOL mode data address register 241
18.7.39 Cordic T0/1UN ILOX mode data address register 241
18.7.40 Cordic T0/1UN ILOL mode data address register 242
19 USB 2.0 device controller 243
19.1 Introduction 243
19.2 Basic configuration 243
19.3 Features 243
19.4 General description. 243
19.4.1 USB software interface 245
19.4.2 Fixed endpoint configuration 245
19.4.3 SoftConnect 246
19.4.4 Interrupts 246
19.4.5 Suspend and resume 246
19.4.6 Clocking 247
19.5 Pin description 247
19.6 Register description 248
19.6.1 USB Device Command/Status register 248
19.6.2 USB Info register 250
19.6.3 USB EP Command/Status List start address 251
19.6.4 USB Data buffer start address 251
19.6.5 USB Link Power Management register 252
19.6.6 USB Endpoint skip 252
19.6.7 USB Endpoint Buffer in use 252
19.6.8 USB Endpoint Buffer Configuration 253
19.6.9 USB interrupt status register 253
19.6.10 USB interrupt enable register 255
19.6.11 USB set interrupt status register 256
19.6.12 USB Endpoint toggle 256
19.7 Functional description 256
19.7.1 Endpoint command/status list 256
19.7.2 Control endpoint 0 259
19.7.3 Generic endpoint: single-buffering 260
19.7.4 Generic endpoint: double-buffering 261
19.7.5 Special cases 261
19.7.6 USB wake-up 262
20 Flexcomm interface serial communication. 263
20.1 How to read this chapter 263
20.2 Introduction 263
20.3 Features 263
20.4 Basic configuration 263
20.5 Architecture 264
20.5.1 Choosing a peripheral function 264
20.5.2 FIFO usage 264
20.5.3 DMA 264
20.5.4 AHB bus access 264
20.6 Pin description 265
20.7 Register description 265
20.7.1 IO Mode register 266
20.7.2 Peripheral Select and Flexcomm Interface ID register 267
20.7.3 Peripheral identification register. 268
21 USART 269
21.1 Introduction 269
21.2 Features 269
21.3 Basic configuration 269
21.3.1 Configure the Flexcomm Interface clock and USART baud rate 270
21.3.2 Configure the USART for wake-up. 270
21.4 Pin description 271

continued >>

21.5 General description. 271
21.6 Register description 273
21.6.1 USART Configuration register 274
21.6.2 USART Control register 276
21.6.3 USART Status register 277
21.6.4 USART Interrupt Enable read and set register . . 278
21.6.5 USART Interrupt Enable Clear register 279
21.6.6 USART Baud Rate Generator register 280
21.6.7 USART Interrupt Status register 280
21.6.8 Oversample selection register 281
21.6.9 Address register 281
21.6.10 FIFO Configuration register 282
21.6.11 FIFO status register 282
21.6.12 FIFO trigger level settings register 283
21.6.13 FIFO interrupt enable set and read. 284
21.6.14 FIFO interrupt enable clear and read 285
21.6.15 FIFO interrupt status register 285
21.6.16 FIFO write data register 285
21.6.17 FIFO read data register 286
21.6.18 FIFO data read with no FIFO pop. 286
21.6.19 Module identification register 286
21.7 Functional description. 287
21.7.1 AHB bus access 287
21.7.2 Clocking and baud rates 287
21.7.3 DMA 288
21.7.4 Synchronous mode. 288
21.7.5 Flow control 288
21.7.6 Auto-baud function 289
21.7.7 RS-485 support. 289
21.7.8 Oversampling 289
21.7.9 Break generation and detection 290
21.7.10 LIN bus 290
21.7.11 7816 Protocol Support 290
22 Serial Peripheral Interfaces (SPI) 291
22.1 How to read this chapter. 291
22.2 Features 291
22.3 Basic configuration 291
22.3.1 Configure the SPI for wake-up 291
22.4 Pin description 292
22.5 General description. 293
22.6 Register description 294
22.6.1 SPI Configuration register. 295
22.6.2 SPI Delay register. 296
22.6.3 SPI Status register 297
22.6.4 SPI Interrupt Enable read and Set register . 298
22.6.5 SPI Interrupt Enable Clear register. 298
22.6.6 SPI Divider register. 298
22.6.7 SPI Interrupt Status register 299
22.6.8 FIFO Configuration register 300
22.6.9 FIFO status register 300
22.6.10 FIFO trigger settings register 301
22.6.11 FIFO interrupt enable set and read 302
22.6.12 FIFO interrupt enable clear and read 303
22.6.13 FIFO interrupt status register 303
22.6.14 FIFO write data register 303
22.6.15 FIFO read data register 305
22.6.16 FIFO data read with no FIFO pop 305
22.6.17 Module identification register 306
22.7 Functional description 306
22.7.1 AHB bus access 306
22.7.2 Operating modes: clock and phase selection 306
22.7.3 Frame delays 308
22.7.4 Clocking and data rates 311
22.7.5 Slave select 311
22.7.6 DMA operation 312
22.7.7 Data lengths greater than 16 bits 313
22.7.8 Data stalls 313
23 I²C-bus interfaces 315
23.1 Introduction 315
23.2 Features 315
23.3 Pin description 315
23.4 Basic configuration. 315
23.4.1 I²C transmit/receive in master mode 316
23.4.2 I²C receive/transmit in slave mode. 318
23.4.3 Configure the I²C for wake-up 319
23.5 General description 320
23.6 Register description 320
23.6.1 I2C Configuration register 321
23.6.2 I2C Status register 322
23.6.3 Interrupt Enable Set and read register. 326
23.6.4 Interrupt Enable Clear register. 327
23.6.5 Time-out value register. 328
23.6.6 Clock Divider register. 328
23.6.7 Interrupt Status register 329
23.6.8 Master Control register. 330
23.6.9 Master Time register 331
23.6.10 Master Data register. 332
23.6.11 Slave Control register. 332
23.6.12 Slave Data register. 333
23.6.13 Slave Address 0 register 333
23.6.14 Slave Address 1, 2, and 3 registers 334
23.6.15 Slave address Qualifier 0 register 334
23.6.16 Monitor data register 335
23.6.17 Module identification register 336
23.7 Functional description 337
23.7.1 AHB bus access 337
23.7.2 Bus rates and timing considerations 337
23.7.3 Time-out 338
23.7.4 Ten-bit addressing 339

continued >>

23.7.5 Clocking and power considerations 339
23.7.6 Interrupt handling 339
23.7.7 DMA 339
23.7.8 Automatic operation 341
24 Quadrature decoder. 342
24.1 Introduction 342
24.2 Features 342
24.3 Basic configuration 342
24.4 Pin description 342
24.5 General description. 343
24.5.1 Sampling and Decoding 343
24.5.2 Debounce Filter 344
24.5.3 Accumulator Registers 344
24.6 Register description 345
24.6.1 QDEC Configuration and Control register . . 346
24.6.2 QDEC Sample Control register. 347
24.6.3 QDEC Sample Value register 348
24.6.4 QDEC Valid Accumulator register. 348
24.6.5 ACC Snapshot register 348
24.6.6 QDEC Invalid Accumulator register 348
24.6.7 DB Snapshot register 349
24.6.8 QDEC Interrupt register 349
24.6.9 QDEC Interrupt Enable register 349
24.6.10 QDEC status register 349
24.7 Functional description. 350
24.7.1 Input Waveform 350
24.7.2 Sample Rate Configuration. 351
25 SPI Flash Interface (SPIFI) 353
25.1 Introduction 353
25.2 Features 353
25.3 Basic configuration 353
25.4 General description. 354
25.5 Pin description 354
25.6 Supported devices 355
25.7 SPIFI hardware. 355
25.8 Register description 355
25.8.1 SPIFI control register 355
25.8.2 SPIFI command register. 356
25.8.3 SPIFI address register 357
25.8.4 SPIFI intermediate data register. 358
25.8.5 SPIFI cache limit register 358
25.8.6 SPIFI data register 358
25.8.7 SPIFI memory command register. 359
25.8.8 SPIFI status register 360
25.9 Functional description. 361
25.9.1 Data transfer. 361
25.9.2 Software requirements and capabilities 363
25.9.3 Peripheral mode DMA operation 364
26 Random Number generator (RNG). 365
26.1 Introduction 365
26.2 Features 365
26.3 Basic configuration. 365
26.4 General Description 365
26.4.1 RNG Start. 365
26.4.2 RNG Done Interrupt 366
26.4.3 RNG DATA 366
26.5 Register Description. 366
26.5.1 RNG control register 366
26.5.2 RNG status register 367
26.5.3 RNG Output Data register 367
26.5.4 Interrupt Status register 367
26.5.5 Interrupt Enable register. 367
27 CRC engine 368
27.1 Introduction 368
27.2 Features 368
27.3 Basic configuration. 368
27.4 Pin description 368
27.5 General description 368
27.6 Register description 369
27.6.1 CRC mode register 369
27.6.2 CRC seed register 370
27.6.3 CRC checksum register 370
27.6.4 CRC data register 370
27.7 Functional description 370
27.7.1 Timing. 370
27.7.2 Setup 371
28 Flash memory controller 372
28.1 How to read this chapter 372
28.2 Features 372
28.3 General description 372
28.3.1 Flash physical organization 372
28.3.2 Flash lock and protection 373
28.4 Register description 375
28.4.1 Flash initial read enable register 376
28.4.2 Flash erase control register 376
28.4.3 Flash erase time setting register 376
28.4.4 Flash operation time setting register 377
28.4.5 Smart erase control register. 377
28.4.6 Interrupt enable register. 378
28.4.7 Interrupt status register 379
28.4.8 Interrupt clear register 380
28.4.9 Lock status register x 380
28.4.10 Lock status register 8 380
28.4.11 Status register 1 381
28.4.12 Flash block 0 error information register 1. . . 382
28.4.13 Flash block 0 error information register 2. . . 382
28.4.14 Flash block 0 error information register 3. . . 382
28.4.15 Flash block 1 error information register 1. . . 382
28.4.16 Flash block 1 error information register 2. . . 382
28.4.17 Flash block 1 error information register 3. . . 382

continued >>

28.4.18 Debug port access password register 383
28.4.19 Erase password register 383
28.5 Functional description 383
28.5.1 Flash read 383
28.5.2 Flash write 383
28.5.3 Flash erase 384
29 DAC 386
29.1 Introduction 386
29.2 Features 386
29.3 Basic configuration 386
29.4 Pin description 387
29.5 General description 387
29.6 Functional description 388
29.6.1 DAC clock 388
29.6.2 DAC DATA 389
29.6.3 User data from APB through FIFO 389
29.6.4 Modulator 390
29.6.5 D2A converter and filter 392
29.6.6 Data trigger (valid only if data is from APB bus) . . 394
29.6.7 Interrupt handler 396
29.7 Register description 396
29.7.1 DAC analog configuration register 397
29.7.2 DAC control register 397
29.7.3 Sin-wave configuration register 0 398
29.7.4 Sin-wave configuration register 1 399
29.7.5 Gain control register 399
29.7.6 Clear trigger register 399
29.7.7 DAC data input register 399
29.7.8 DAC interrupt register 399
29.7.9 DAC interrupt enable register 400
29.7.10 DAC interrupt status register 400
29.7.11 DAC status register 400
30 SD ADC controller (ADC) 401
30.1 How to read this chapter 401
30.2 Features 401
30.3 Pin description 401
30.4 Basic configuration 402
30.5 General description 402
30.5.1 Principle of operation 402
30.5.2 Input stage 403
30.5.3 ADC reference voltage 405
30.5.4 ADC sampling clock generation 405
30.5.5 Conversion modes 406
30.5.6 ADC control 407
30.5.7 Digital decimation filter 408
30.5.8 Output stage 409
30.6 Register description 411
30.6.1 ADC Control register 411
30.6.2 ADC channel select register 413
30.6.3 ADC channel configuration register 413
30.6.4 Window compare threshold register 413
30.6.5 ADC interrupt mask register 413
30.6.6 ADC interrupt status register 413
30.6.7 ADC converted data register 414
30.6.8 ADC configuration 0 register 414
30.6.9 ADC configuration 1register 415
30.6.10 ADC bandgap and buffer setting register . . 417
30.6.11 ADC core and reference setting register . . . 417
30.7 Application examples 418
30.7.1 Random number generation 418
30.7.2 Temperature sensor 419
31 Capacitive sense 420
31.1 How to read this chapter 420
31.2 Features 420
31.3 Basic configuration 420
31.4 Pin description 421
31.5 General description 421
31.5.1 Capacitive sensing methodology 421
31.5.2 The operation of the Capacitive Sense 421
31.6 Register Description 424
31.6.1 Capacitive Sense control register 0 424
31.6.2 Capacitive Sense control register 1 425
31.6.3 Interrupt Status register 425
31.6.4 Interrupt Mask register 425
31.6.5 Output Data register 426
31.6.6 Low Power mode control register 426
31.6.7 Low Power Interrupt register 426
31.6.8 low power interrupt enable register 426
31.6.9 Idle period number register 427
32 Analog comparator (ACMP) 427
32.1 Introduction 427
32.2 Features 427
32.3 Basic configuration 427
32.4 Pin information 427
32.5 General description 428
32.6 Register description 428
32.7 Functional description 428
32.7.1 Comparator Inputs 428
32.7.2 Comparator Outputs 428
33 Appendix - ARM Cortex 429
33.1 ARM Cortex-M4 Details 429
33.1.1 Cortex-M4 implementation optionsQN908x user manual 429
34 Abbreviations 430
35 References 431
36 Legal information 432
36.1 Definitions 432
36.2 Disclaimers 432

continued >>

36.3 Licenses 433
36.4 Patents 433
36.5 Trademarks. 433
37 Tables 434
38 Figures 442
39 Contents 443

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 17 March 2017

Document identifier: UM11023