Summary

Atmel's SAM4L series is a member of a family of Flash microcontrollers based on the high performance 32-bit ARM Cortex-M4 RISC processor running at frequencies up to 48MHz.

The SAM4L series embeds state-of-the-art picoPower technology for ultra-low power consumption. Combined power control techniques are used to bring active current consumption down to 90µA/MHz. The device allows a wide range of options between functionality and power consumption, giving the user the ability to reach the lowest possible power consumption with the feature set required for the application. The WAIT and RETENTION modes provide full logic and RAM retention, associated with fast wake-up capability (<1.5µs) and a very low consumption of, respectively, 3 µA and 1.5 µA. In addition, WAIT mode supports SleepWalking features. In BACKUP mode, CPU, peripherals and RAM are powered off and, while consuming less than 0.9µA with external interrupt wake-up supported.

The SAM4L series offers a wide range of peripherals such as segment LCD controller, embedded hardware capacitive touch (QTouch), USB device & embedded host, 128-bit AES and audio interfaces in addition to high speed serial peripherals such as USART, SPI and I^2C. Additionally the Peripheral Event System and SleepWalking allows the peripherals to communicate directly with each other and make intelligent decisions and decide to wake-up the system on a qualified events on a peripheral level; such as I^2C address match or and ADC threshold.

Features

- **Core**
  - **ARM® Cortex™-M4 running at up to 48MHz**
  - **Memory Protection Unit (MPU)**
  - **Thumb®-2 instruction set**
- **picoPower® Technology for Ultra-low Power Consumption**
  - **Active mode downto 90µA/MHz with configurable voltage scaling**
  - **High performance and efficiency: 28 coremark/mA**
  - **Wait mode downto 3µA with fast wake-up time (<1.5µs) supporting SleepWalking**
  - **Full RAM and Logic Retention mode downto 1.5µA with fast wake-up time (<1.5µs)**
  - **Ultra low power Backup mode with/without RTC downto 1,5/0.9µA**
- **Memories**
  - **From 128 to 512Kbytes embedded Flash, 64-bit wide access,**
    - **0 wait-state capability up to 24MHz**
  - **up to 64Kbytes embedded SRAM**
- **System Functions**
  - **Embedded voltage linear and switching regulator for single supply operation**
  - **Two Power-on-Reset and Two Brown-out Detectors (BOD)**
  - **Quartz or ceramic resonator oscillators: 0.6 to 30MHz main power with Failure Detection and low power 32.768 kHz for RTC or device clock**
  - **High precision 4/8/12MHz factory trimmed internal RC oscillator**
  - **Slow Clock Internal RC oscillator as permanent low-power mode device clock**
  - **High speed 80MHz internal RC oscillator**
  - **Low power 32kHz internal RC oscillator**

Atmel

ATSAM ARM-based Flash MCU

SAM4L Series

42023H–11/2016

  - PLL up to 240MHz for device clock and for USB
  - Digital Frequency Locked Loop (DFLL) with wide input range
  - Up to 16 peripheral DMA (PDCA) channels
- Peripherals
  - USB 2.0 Device and Embedded Host: 12 Mbps, up to 8 bidirectional Endpoints and Multi-packet Ping-pong Mode. On-Chip Transceiver
  - Liquid Crystal Display (LCD) Module with Capacity up to 40 Segments and up to 4 Common Terminals
  - One USART with ISO7816, IrDA®, RS-485, SPI, Manchester and LIN Mode
  - Three USART with SPI Mode
  - One PicoUART for extended UART wake-up capabilities in all sleep modes
  - Windowed Watchdog Timer (WDT)
  - Asynchronous Timer (AST) with Real-time Clock Capability, Counter or Calendar Mode Supported
  - Frequency Meter (FREQM) for Accurate Measuring of Clock Frequency
  - Six 16-bit Timer/Counter (TC) Channels with capture, waveform, compare and PWM mode
  - One Master/Slave Serial Peripheral Interface (SPI) with Chip Select Signals
  - Four Master and Two Slave Two-wire Interfaces (TWI), up to 3.4Mbit/s I^2C-compatible
  - One Advanced Encryption System (AES) with 128-bit key length
  - One 16-channel ADC 300Ksps (ADC) with up to 12 Bits Resolution
  - One DAC 500Ksps (DACC) with up to 10 Bits Resolution
  - Four Analog Comparators (ACIFC) with Optional Window Detection
  - Capacitive Touch Module (CATB) supporting up to 32 buttons
  - Audio Bitstream DAC (ABDACB) Suitable for Stereo Audio
  - Inter-IC Sound (IISC) Controller, Compliant with Inter-IC Sound (I^2S) Specification
  - Peripheral Event System for Direct Peripheral to Peripheral Communication
  - 32-bit Cyclic Redundancy Check Calculation Unit (CRCCU)
  - Random generator (TRNG)
  - Parallel Capture Module (PARC)
  - Glue Logic Controller (GLOC)
- I/O
  - Up to 75 I/O lines with external interrupt capability (edge or level sensitivity), debouncing, glitch filtering and slew-rate control
  - Up to Six High-drive I/O Pins
- Single 1.68-3.6V Power Supply
- Packages
  - 100-lead LQFP, 14 x 14 mm, pitch 0.5 mm/100-ball VFBGA, 7x7 mm, pitch 0.65 mm
  - 64-lead LQFP, 10 x 10 mm, pitch 0.5 mm/64-pad QFN 9x9 mm, pitch 0.5 mm
  - 64-ball WLCSP, 4,314x4,434 mm, pitch 0.5 mm for SAM4LC4/2 and SAM4LS4/2 series
  - 64-ball WLCSP, 5,270x5,194 mm, pitch 0.5 mm for SAM4LC8 and SAM4LS8 series
  - 48-lead LQFP, 7 x 7 mm, pitch 0.5 mm/48-pad QFN 7x7 mm, pitch 0.5 mm

1. Description

Atmel's SAM4L series is a member of a family of Flash microcontrollers based on the high performance 32-bit ARM Cortex-M4 RISC processor running at frequencies up to 48MHz.

The processor implements a Memory Protection Unit (MPU) and a fast and flexible interrupt controller for supporting modern and real-time operating systems.

The ATSAM4L8/L4/L2 embeds state-of-the-art picoPower technology for ultra-low power consumption. Combined power control techniques are used to bring active current consumption down to 90µA/MHz. The device allows a wide range of options between functionality and power consumption, giving the user the ability to reach the lowest possible power consumption with the feature set required for the application. On-chip regulator improves power efficiency when used in swichting mode with an external inductor or can be used in linear mode if application is noise sensitive.

The ATSAM4L8/L4/L2 supports 4 power saving strategies. The SLEEP mode put the CPU in idle mode and offers different sub-modes which automatically switch off/on bus clocks, PLL, oscillators. The WAIT and RETENTION modes provide full logic and RAM retention, associated with fast wake-up capability (<1.5µs) and a very low consumption of, respectively, 3 µA and 1.5 µA. In addition, WAIT mode supports SleepWalking features. In BACKUP mode, CPU, peripherals and RAM are powered off and, while consuming less than 0.5µA, the device is able to wake-up from external interrupts.

The ATSAM4L8/L4/L2 incorporates on-chip Flash tightly coupled to a low power cache (LPCACHE) for active consumption optimization and SRAM memories for fast access.

The LCD controller is intended for monochrome passive liquid crystal display (LCD) with up to 4 Common terminals and up to 40 Segments terminals. Dedicated Low Power Waveform, Contrast Control, Extended Interrupt Mode, Selectable Frame Frequency and Blink functionality are supported to offload the CPU, reduce interrupts and reduce power consumption. The controller includes integrated LCD buffers and integrated power supply voltage.

The low-power and high performance capacitive touch module (CATB) is introduced to meet the demand for a low power capacitive touch solution that could be used to handle buttons, sliders and wheels. The CATB provides excellent signal performance, as well as autonomous touch and proximity detection for up to 32 sensors. This solution includes an advanced sequencer in addition to an hardware filtering unit.

The Advanced Encryption Standard module (AESA) is compliant with the *FIPS (Federal Information Processing Standard) Publication 197, Advanced Encryption Standard (AES)*, which specifies a symmetric block cipher that is used to encrypt and decrypt electronic data. *Encryption* is the transformation of a usable message, called the *plaintext*, into an unreadable form, called the *ciphertext*. On the other hand, *decryption* is the transformation that recovers the plaintext from the ciphertext. AESA supports 128 bits cryptographic key sizes.

The Peripheral Direct Memory Access (DMA) controller enables data transfers between peripherals and memories without processor involvement. The Peripheral DMA controller drastically reduces processing overhead when transferring continuous and large data streams.

The Peripheral Event System (PES) allows peripherals to receive, react to, and send peripheral events without CPU intervention. Asynchronous interrupts allow advanced peripheral operation in low power modes.

The Power Manager (PM) improves design flexibility and security. The Power Manager supports SleepWalking functionality, by which a module can be selectively activated based on peripheral

events, even in sleep modes where the module clock is stopped. Power monitoring is supported by on-chip Power-on Reset (POR18, POR33), Brown-out Detectors (BOD18, BOD33). The device features several oscillators, such as Phase Locked Loop (PLL), Digital Frequency Locked Loop (DFLL), Oscillator 0 (OSC0), Internal RC 4,8,12MHz oscillator (RCFAST), system RC oscillator (RCSYS), Internal RC 80MHz, Internal 32kHz RC and 32kHz Crystal Oscillator. Either of these oscillators can be used as source for the system clock. The DFLL is a programmable internal oscillator from 40 to 150MHz. It can be tuned to a high accuracy if an accurate reference clock is running, e.g. the 32kHz crystal oscillator.

The Watchdog Timer (WDT) will reset the device unless it is periodically serviced by the software. This allows the device to recover from a condition that has caused the system to be unstable.

The Asynchronous Timer (AST) combined with the 32kHz crystal oscillator supports powerful real-time clock capabilities, with a maximum timeout of up to 136 years. The AST can operate in counter or calendar mode.

The Frequency Meter (FREQM) allows accurate measuring of a clock frequency by comparing it to a known reference clock.

The Full-speed USB 2.0 device and embedded host interface (USBC) supports several USB classes at the same time utilizing the rich end-point configuration.

The device includes six identical 16-bit Timer/Counter (TC) channels. Each channel can be independently programmed to perform frequency measurement, event counting, interval measurement, pulse generation, delay timing, and pulse width modulation.

The ATSAM4L8/L4/L2 also features many communication interfaces, like USART, SPI, or TWI, for communication intensive applications. The USART supports different communication modes, like SPI Mode and LIN Mode.

A general purpose 16-channel ADC is provided, as well as four analog comparators (ACIFC). The ADC can operate in 12-bit mode at full speed. The analog comparators can be paired to detect when the sensing voltage is within or outside the defined reference window.

Atmel offers the QTouch Library for embedding capacitive touch buttons, sliders, and wheels functionality. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys as well as Adjacent Key Suppression® (AKS®) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop, and debug your own touch applications.

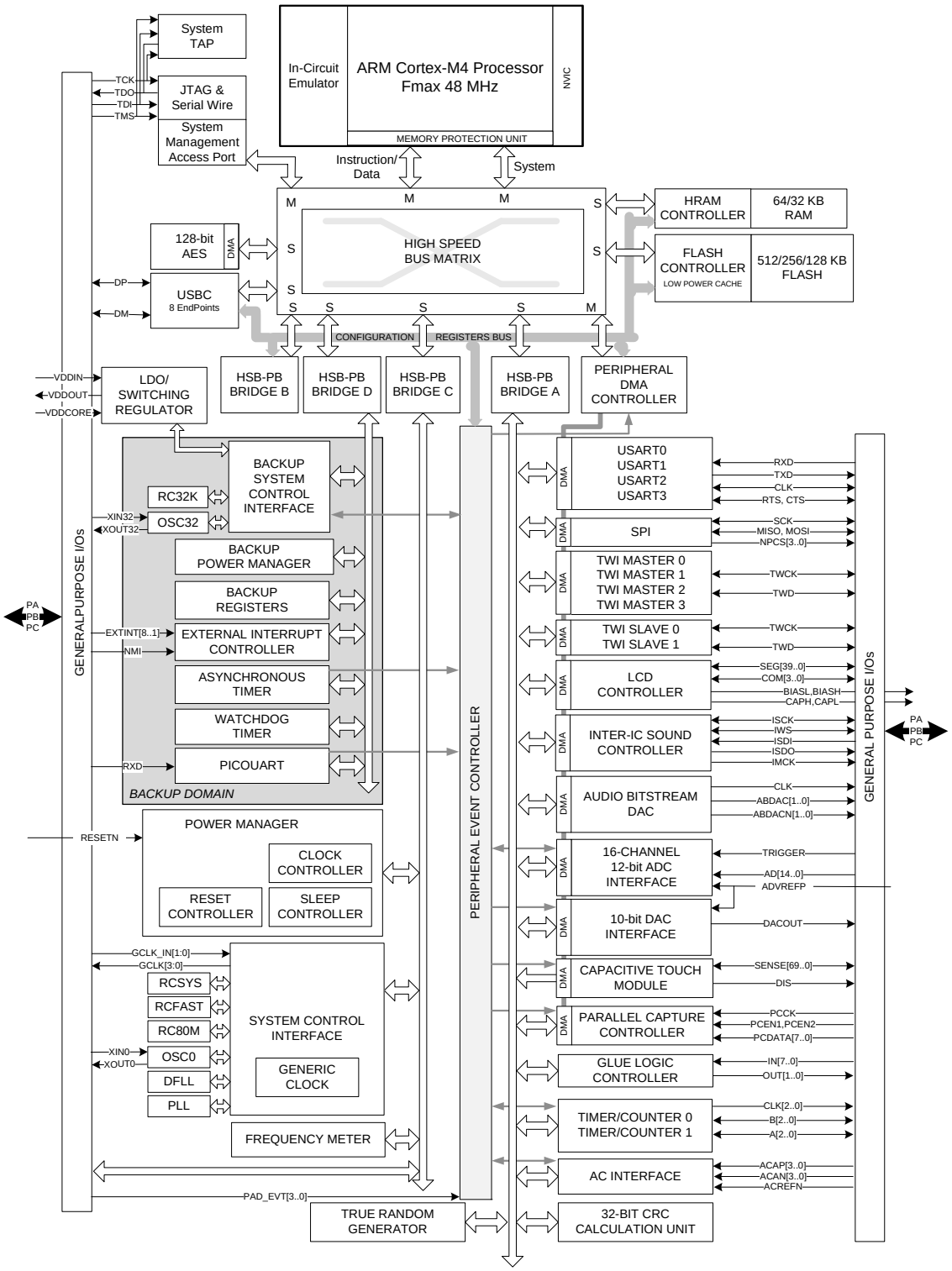
The Audio Bitstream DAC (ABDACB) converts a 16-bit sample value to a digital bitstream with an average value proportional to the sample value. Two channels are supported, making the ABDAC particularly suitable for stereo audio.

The Inter-IC Sound Controller (IISC) provides a 5-bit wide, bidirectional, synchronous, digital audio link with external audio devices. The controller is compliant with the Inter-IC Sound (I2S) bus specification.

2. Overview

2.1 Block Diagram

Figure 2-1. Block Diagram



2.2 Configuration Summary

Table 2-1. Sub Series Summary

Feature	ATSAM4LC	ATSAM4LS
SEGMENT LCD	Yes	No
AESA	Yes	No
USB	Device + Host	Device Only

Table 2-2. ATSAM4LC Configuration Summary

Feature	ATSAM4LC8/4/2C	ATSAM4LC8/4/2B	ATSAM4LC8/4/2A
Number of Pins	100	64	48
Max Frequency	48MHz		
Flash	512/256/128KB		
SRAM	64/32/32KB		
SEGMENT LCD	4x40	4x23	4x13
GPIO	75	43	27
High-drive pins	6	3	1
External Interrupts	8 + 1 NMI		
TWI	2 Masters + 2 Masters/Slaves		1 Master + 1 Master/Slave
USART	4		3 in LC sub series 4 in LS sub series
PICOUART	1		0
Peripheral DMA Channels	16		
AESA	1		
Peripheral Event System	1		
SPI	1		
Asynchronous Timers	1		
Timer/Counter Channels	6	3	
Parallel Capture Inputs	8		
Frequency Meter	1		
Watchdog Timer	1		
Power Manager	1		
Glue Logic LUT	2		1

Table 2-2. ATSAM4LC Configuration Summary

Feature	ATSAM4LC8/4/2C	ATSAM4LC8/4/2B	ATSAM4LC8/4/2A
Oscillators	Digital Frequency Locked Loop 20-150MHz (DFLL) Phase Locked Loop 48-240MHz (PLL) Crystal Oscillator 0.6-30MHz (OSC0) Crystal Oscillator 32kHz (OSC32K) RC Oscillator 80MHz (RC80M) RC Oscillator 4,8,12MHz (RCFAST) RC Oscillator 115kHz (RCSYS) RC Oscillator 32kHz (RC32K)		
ADC	15-channel	7-channel	3-channel
DAC	1-channel		
Analog Comparators	4	2	1
CATB Sensors	32	32	26
USB	1		
Audio Bitstream DAC	1		
IIS Controller	1		
Packages	TQFP/VFBGA	TQFP/QFN/ WLCSP	TQFP/QFN

.

Table 2-3. ATSAM4LS Configuration Summary

Feature	ATSAM4LS8/4/2C	ATSAM4LS8/4/2B	ATSAM4LS8/4/2A
Number of Pins	100	64	48
Max Frequency	48MHz		
Flash	512/256/128KB		
SRAM	64/32/32KB		
SEGMENT LCD	NA		
GPIO	80	48	32
High-drive pins	6	3	1
External Interrupts	8 + 1 NMI		
TWI	2 Masters + 2 Masters/Slaves		1 Master + 1 Master/Slave
USART	4		3 in LC sub series 4 in LS sub series
PICOUART	1		0
Peripheral DMA Channels	16		
AESA	NA		
Peripheral Event System	1		
SPI	1		
Asynchronous Timers	1		

Table 2-3. ATSAM4LS Configuration Summary

Feature	ATSAM4LS8/4/2C	ATSAM4LS8/4/2B	ATSAM4LS8/4/2A
Timer/Counter Channels	6	3	
Parallel Capture Inputs	8		
Frequency Meter	1		
Watchdog Timer	1		
Power Manager	1		
Glue Logic LUT	2		1
Oscillators	Digital Frequency Locked Loop 20-150 MHz (DFLL) Phase Locked Loop 48-240 MHz (PLL) Crystal Oscillator 0.6-30 MHz (OSC0) Crystal Oscillator 32 kHz (OSC32K) RC Oscillator 80 MHz (RC80M) RC Oscillator 4,8,12 MHz (RCFAST) RC Oscillator 115 kHz (RCSYS) RC Oscillator 32 kHz (RC32K)		
ADC	15-channel	7-channel	3-channel
DAC	1-channel		
Analog Comparators	4	2	1
CATB Sensors	32	32	26
USB	1		
Audio Bitstream DAC	1		
IIS Controller	1		
Packages	TQFP/VFBGA	TQFP/QFN/ WLCSP	TQFP/QFN

3. Package and Pinout

3.1 Package

The device pins are multiplexed with peripheral functions as described in Section 3.2 ”Peripheral Multiplexing on I/O lines” on page 19.

3.1.1 ATSAM4LCx Pinout

Figure 3-1. ATSAM4LC TQFP100 Pinout

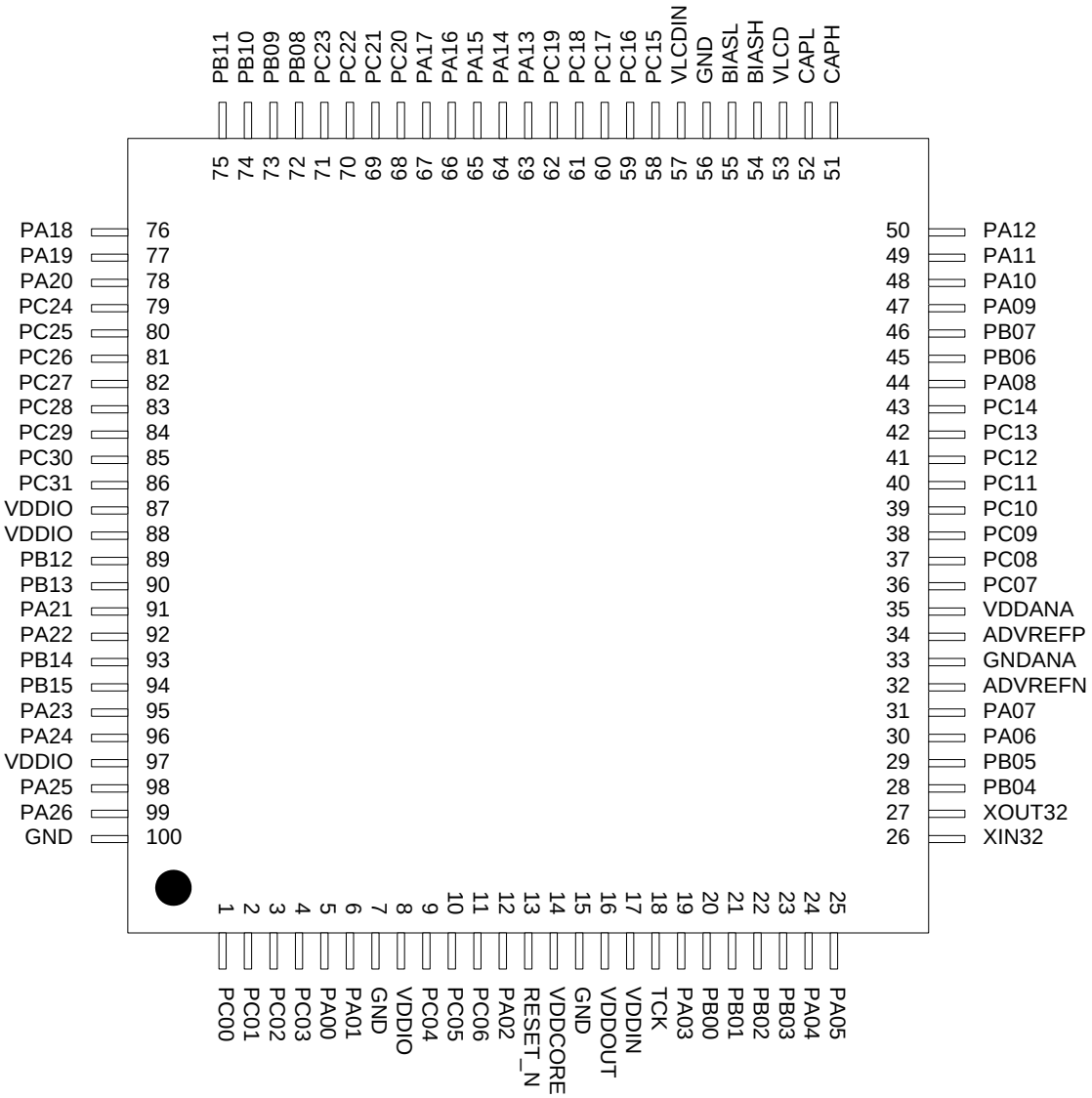


Figure 3-2. ATSAM4LC VFBGA100 Pinout

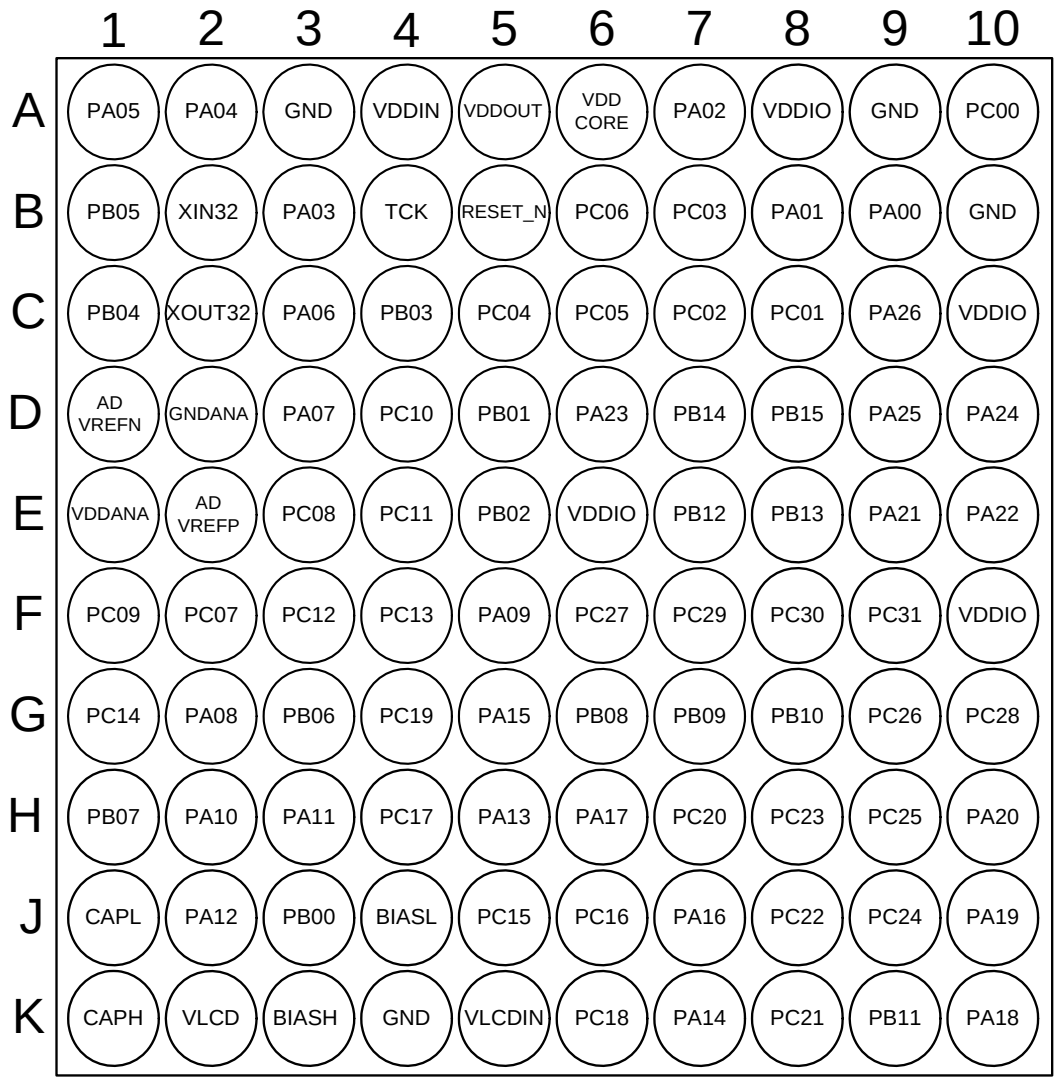


Figure 3-3. ATSAM4LC WLCSP64 Pinout

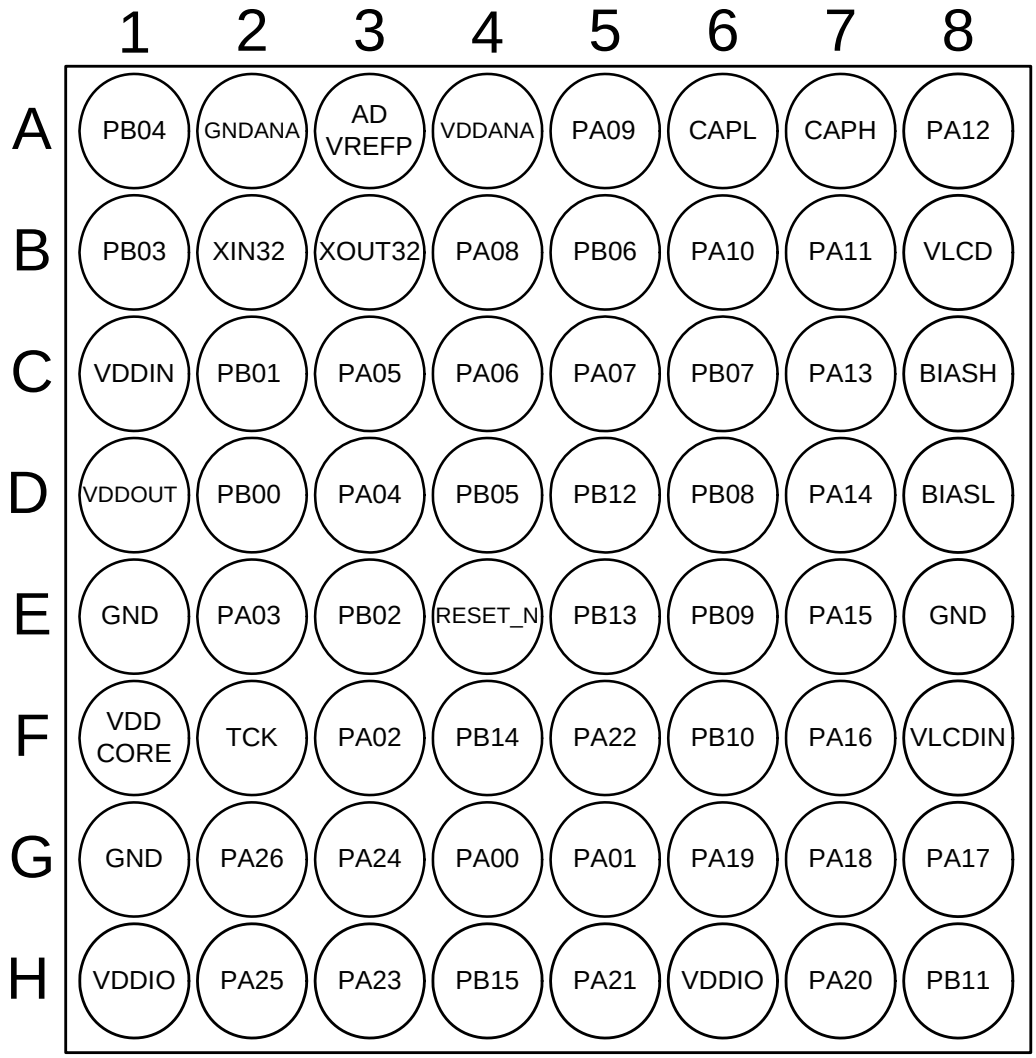


Figure 3-4. ATSAM4LC TQFP64/QFN64 Pinout

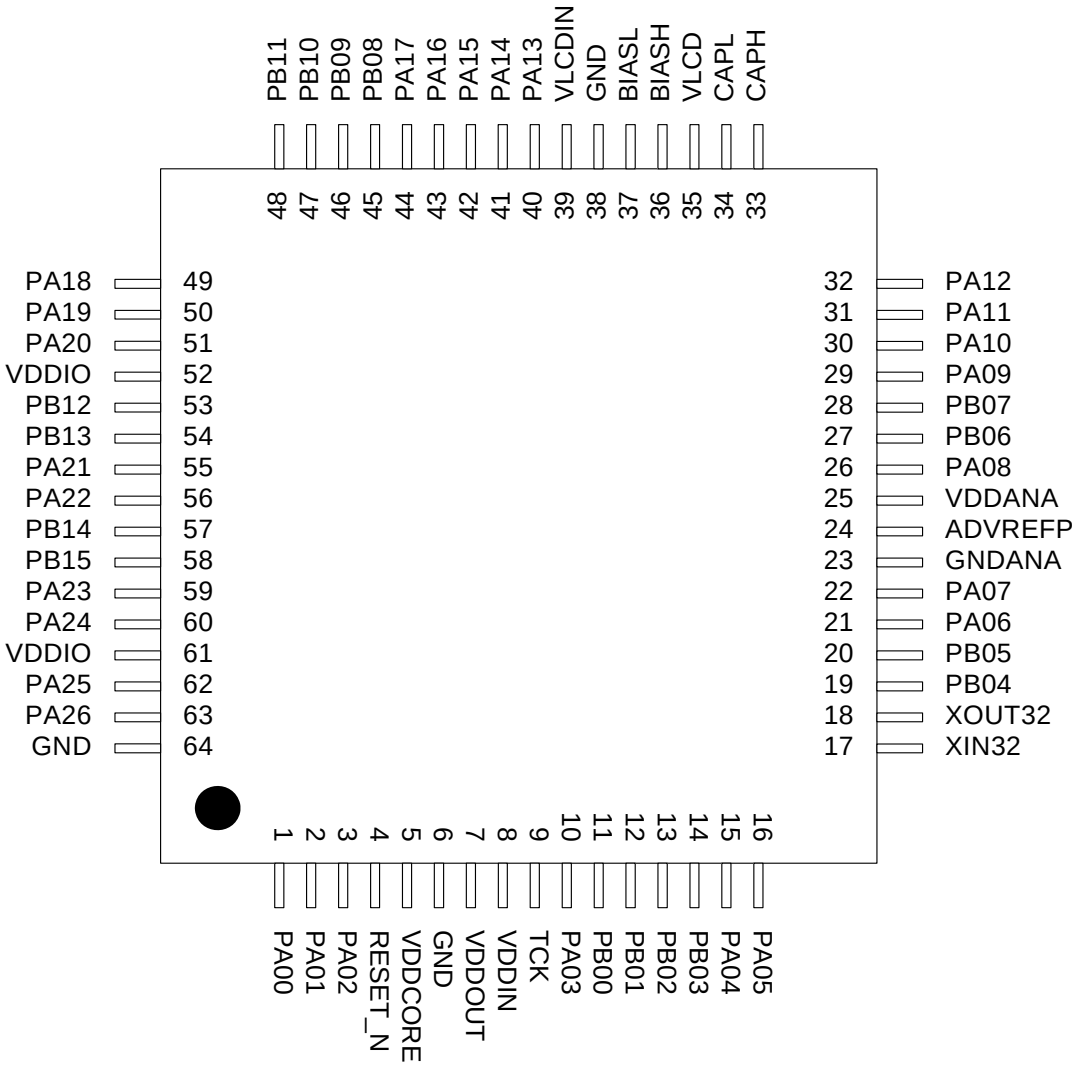
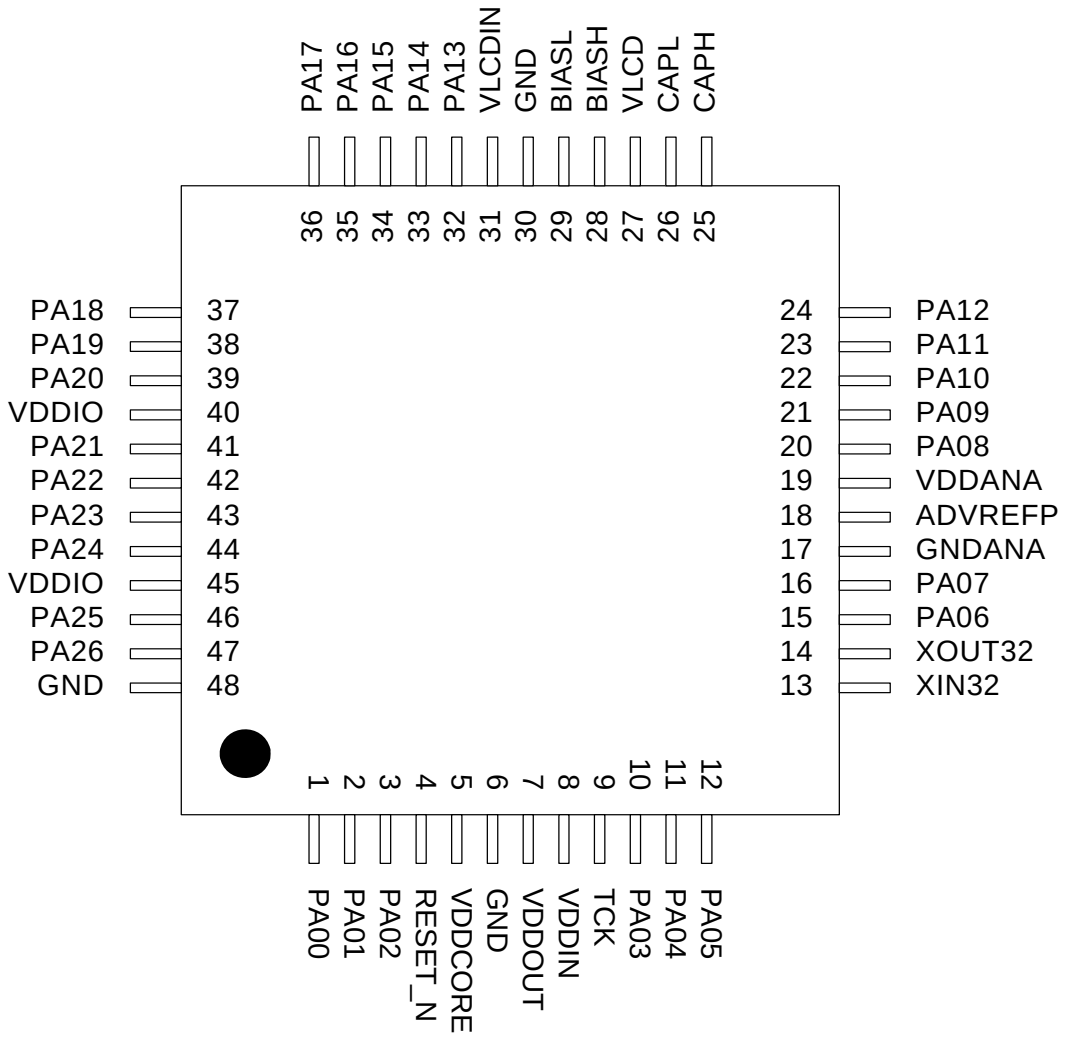


Figure 3-5. ATSAM4LC TQFP48/QFN48 Pinout



3.1.2 ATSAM4LSx Pinout

Figure 3-6. ATSAM4LS TQFP100 Pinout

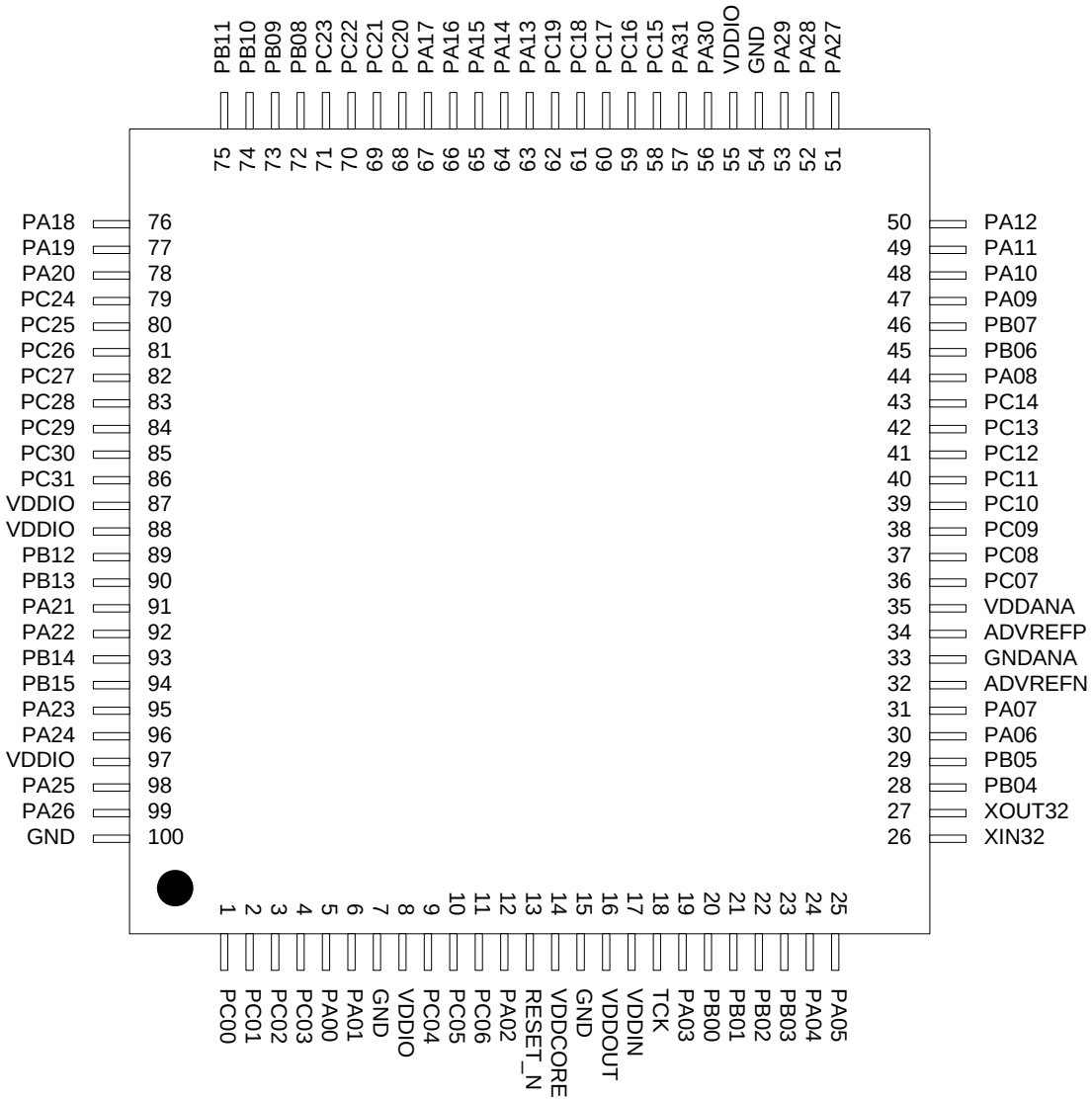


Figure 3-7. ATSAM4LS VFBGA100 Pinout

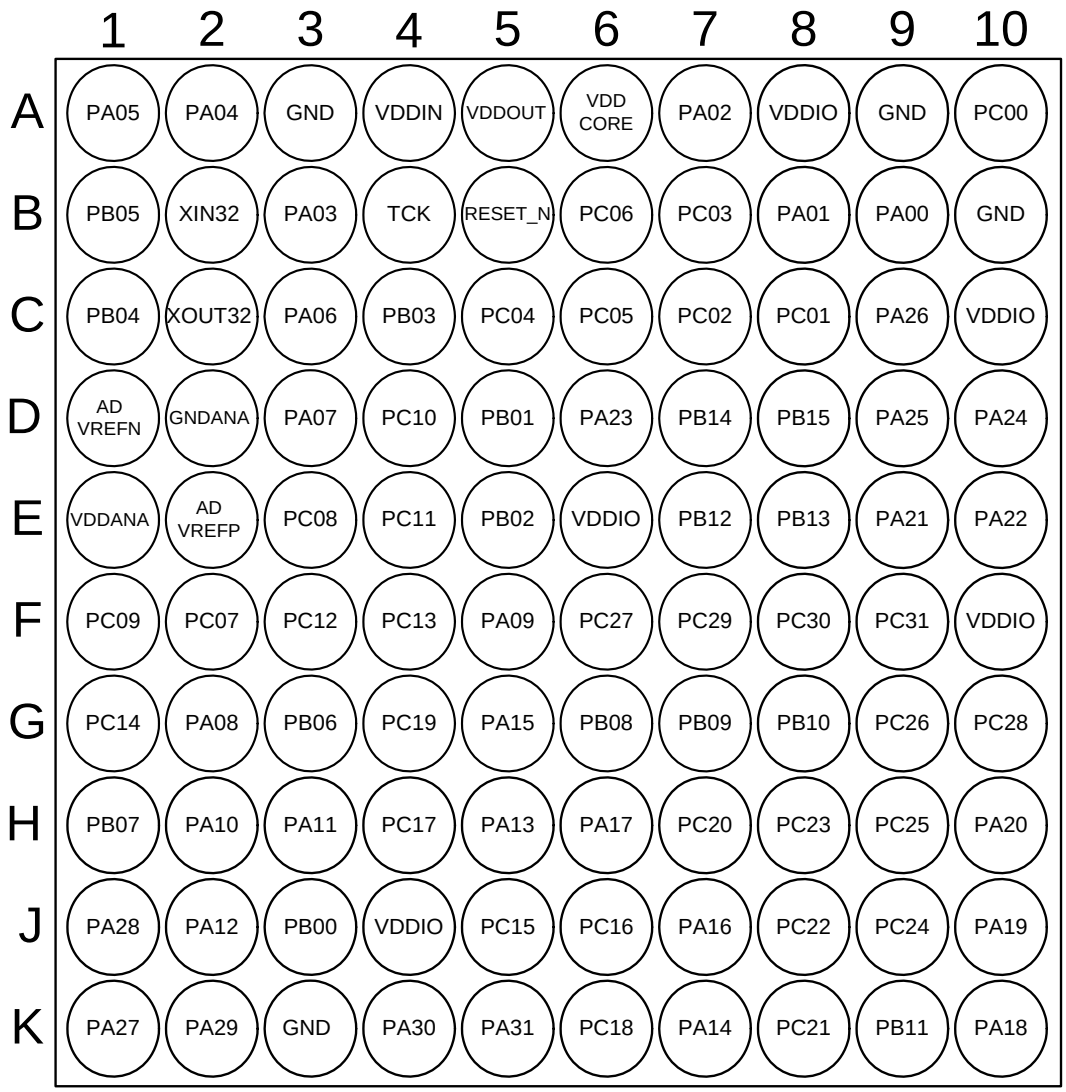


Figure 3-8. ATSAM4LS WLCSP64 Pinout

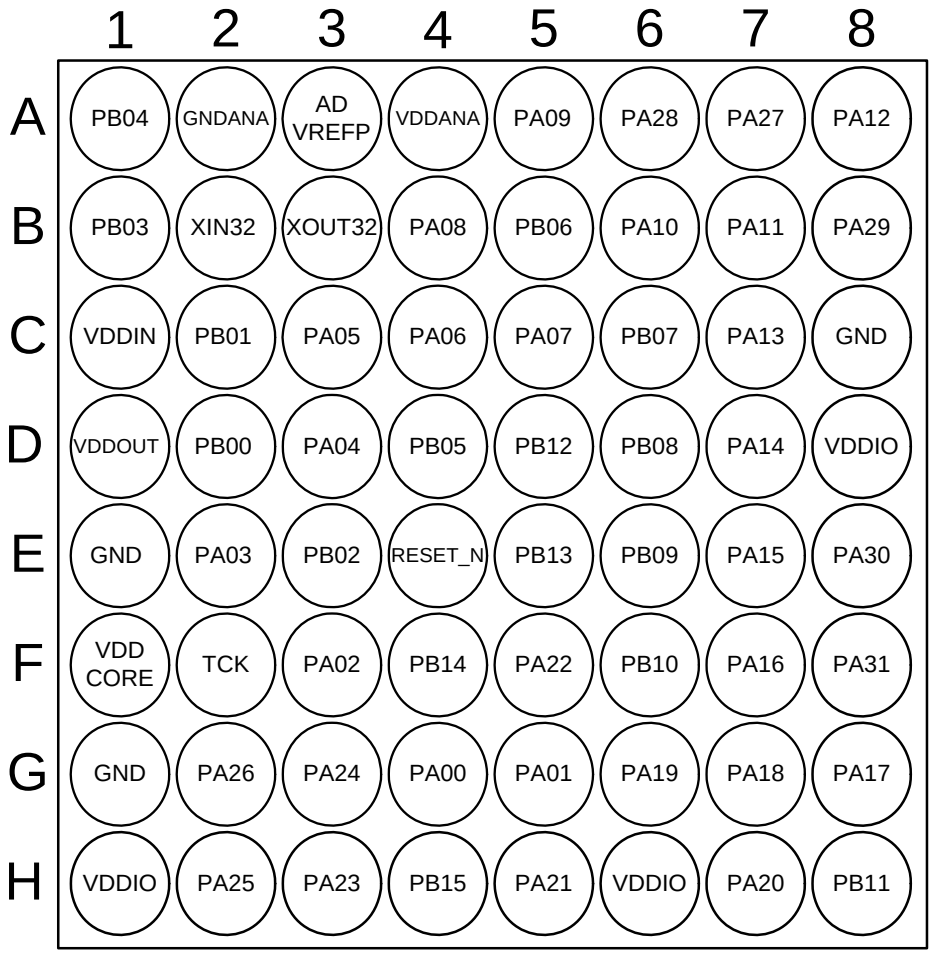


Figure 3-9. ATSAM4LS TQFP64/QFN64 Pinout

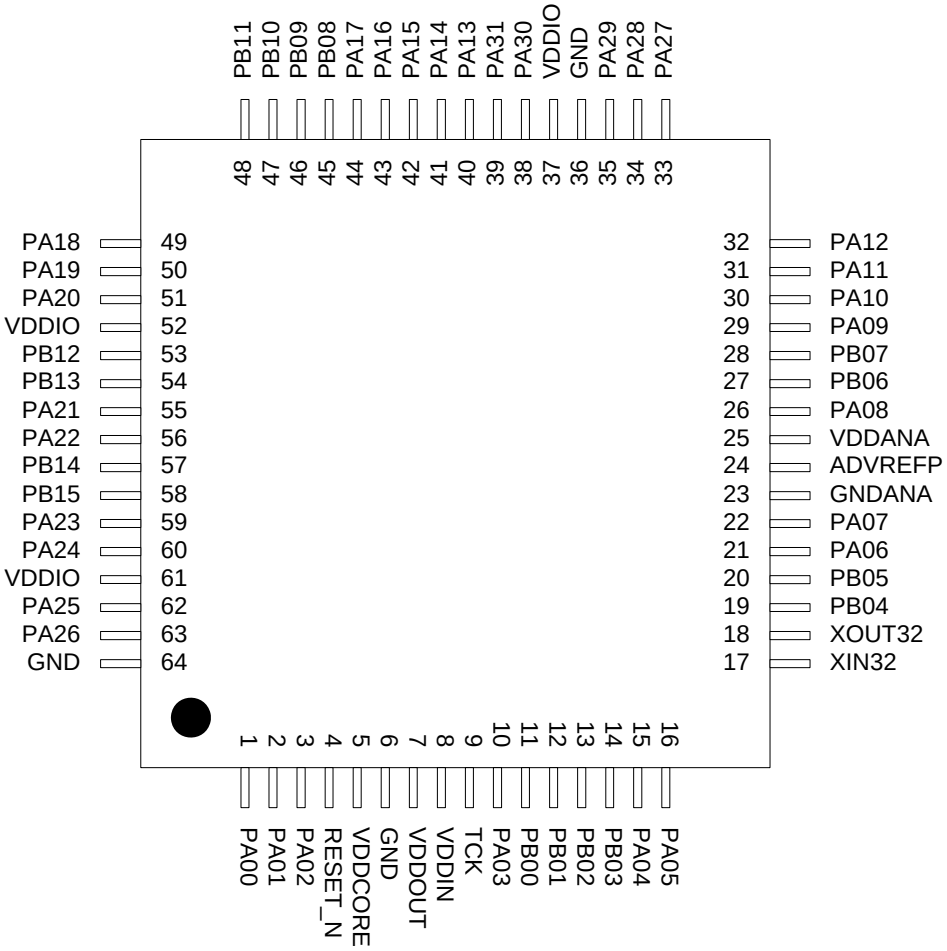
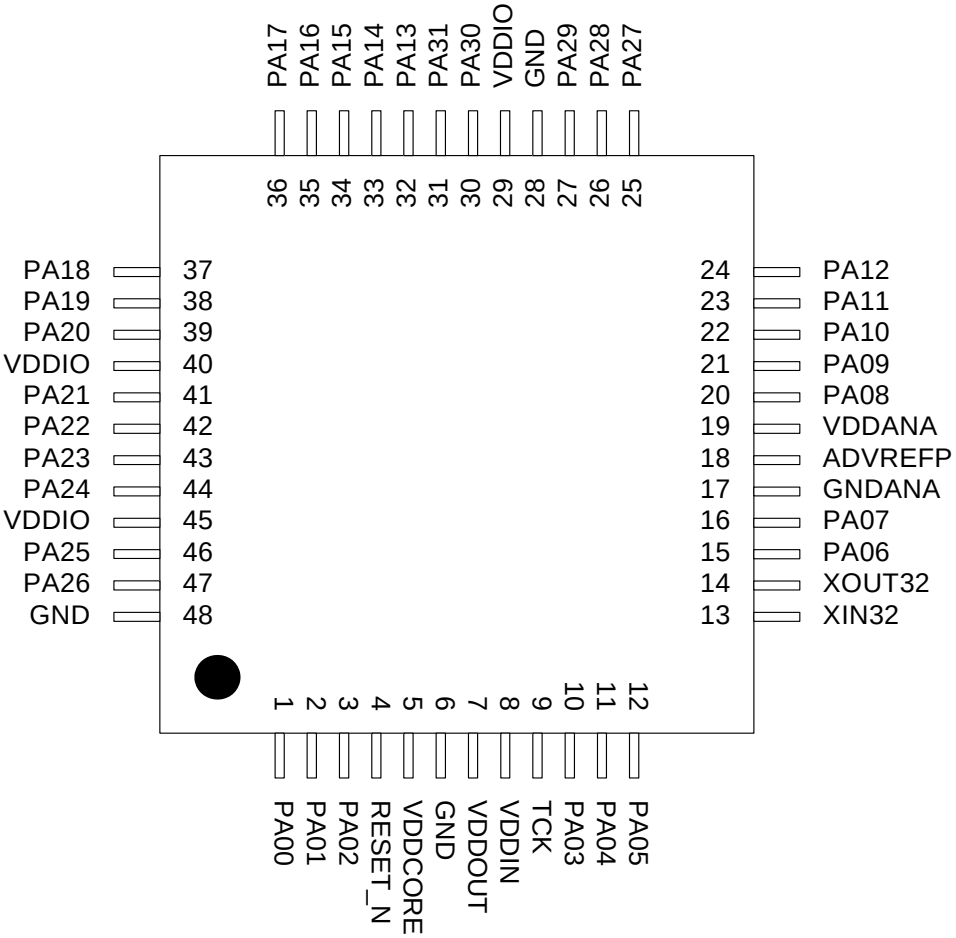


Figure 3-10. ATSAM4LS TQFP48/QFN48 Pinout



See Section 3.3 ”Signals Description” on page 31 for a description of the various peripheral signals.

Refer to ”Electrical Characteristics” on page 1121 for a description of the electrical properties of the pin types used.

3.2 Peripheral Multiplexing on I/O lines

3.2.1 Multiplexed Signals

Each GPIO line can be assigned to one of the peripheral functions. The following tables (Section 3-1 "100-pin GPIO Controller Function Multiplexing" on page 19 to Section 3-4 "48-pin GPIO Controller Function Multiplexing" on page 28) describes the peripheral signals multiplexed to the GPIO lines.

Peripheral functions that are not relevant in some parts of the family are grey-shaded.

For description of differents Supply voltage source, refer to the Section 5. "Power and Startup Considerations" on page 43.

Table 3-1. 100-pin GPIO Controller Function Multiplexing (Sheet 1 of 4)

ATSAM4LC		ATSAM4LS		Pin	GPIO	Supply	GPIO Functions						
QFN	VFBGA	QFN	VFBGA				A	B	C	D	E	F	G
5	B9	5	B9	PA00	0	VDDIO							
6	B8	6	B8	PA01	1	VDDIO							
12	A7	12	A7	PA02	2	VDDIN	SCIF GCLK0	SPI NPCS0					CATB DIS
19	B3	19	B3	PA03	3	VDDIN		SPI MISO					
24	A2	24	A2	PA04	4	VDDANA	ADCIFE AD0	USART0 CLK	EIC EXTINT2	GLOC IN1			CATB SENSE0
25	A1	25	A1	PA05	5	VDDANA	ADCIFE AD1	USART0 RXD	EIC EXTINT3	GLOC IN2	ADCIFE TRIGGER		CATB SENSE1
30	C3	30	C3	PA06	6	VDDANA	DACC VOUT	USART0 RTS	EIC EXTINT1	GLOC IN0	ACIFC ACAN0		CATB SENSE2
31	D3	31	D3	PA07	7	VDDANA	ADCIFE AD2	USART0 TXD	EIC EXTINT4	GLOC IN3	ACIFC ACAP0		CATB SENSE3
44	G2	44	G2	PA08	8	LCDA	USART0 RTS	TC0 A0	PEVC PAD EVT0	GLOC OUT0		LCDCA SEG23	CATB SENSE4
47	F5	47	F5	PA09	9	LCDA	USART0 CTS	TC0 B0	PEVC PAD EVT1	PARC PCDATA0		LCDCA COM3	CATB SENSE5
48	H2	48	H2	PA10	10	LCDA	USART0 CLK	TC0 A1	PEVC PAD EVT2	PARC PCDATA1		LCDCA COM2	CATB SENSE6
49	H3	49	H3	PA11	11	LCDA	USART0 RXD	TC0 B1	PEVC PAD EVT3	PARC PCDATA2		LCDCA COM1	CATB SENSE7
50	J2	50	J2	PA12	12	LCDA	USART0 TXD	TC0 A2		PARC PCDATA3		LCDCA COM0	CATB DIS
63	H5	63	H5	PA13	13	LCDA	USART1 RTS	TC0 B2	SPI NPCS1	PARC PCDATA4		LCDCA SEG5	CATB SENSE8
64	K7	64	K7	PA14	14	LCDA	USART1 CLK	TC0 CLK0	SPI NPCS2	PARC PCDATA5		LCDCA SEG6	CATB SENSE9
65	G5	65	G5	PA15	15	LCDA	USART1 RXD	TC0 CLK1	SPI NPCS3	PARC PCDATA6		LCDCA SEG7	CATB SENSE10

Table 3-1. 100-pin GPIO Controller Function Multiplexing (Sheet 2 of 4)

ATSAM4LC		ATSAM4LS		Pin	GPIO	Supply	GPIO Functions						
QFN	VFBGA	QFN	VFBGA				A	B	C	D	E	F	G
66	J7	66	J7	PA16	16	LCDA	USART1 TXD	TC0 CLK2	EIC EXTINT1	PARC PCDATA7		LCDCA SEG8	CATB SENSE11
67	H6	67	H6	PA17	17	LCDA	USART2 RTS	ABDACB DAC0	EIC EXTINT2	PARC PCCK		LCDCA SEG9	CATB SENSE12
76	K10	76	K10	PA18	18	LCDA	USART2 CLK	ABDACB DACN0	EIC EXTINT3	PARC PCEN1		LCDCA SEG18	CATB SENSE13
77	J10	77	J10	PA19	19	LCDA	USART2 RXD	ABDACB DAC1	EIC EXTINT4	PARC PCEN2	SCIF GCLK0	LCDCA SEG19	CATB SENSE14
78	H10	78	H10	PA20	20	LCDA	USART2 TXD	ABDACB DACN1	EIC EXTINT5	GLOC IN0	SCIF GCLK1	LCDCA SEG20	CATB SENSE15
91	E9	91	E9	PA21	21	LCDC	SPI MISO	USART1 CTS	EIC EXTINT6	GLOC IN1	TWIM2 TWD	LCDCA SEG34	CATB SENSE16
92	E10	92	E10	PA22	22	LCDC	SPI MOSI	USART2 CTS	EIC EXTINT7	GLOC IN2	TWIM2 TWCK	LCDCA SEG35	CATB SENSE17
95	D6	95	D6	PA23	23	LCDC	SPI SCK	TWIMS0 TWD	EIC EXTINT8	GLOC IN3	SCIF GCLK IN0	LCDCA SEG38	CATB DIS
96	D10	96	D10	PA24	24	LCDC	SPI NPCS0	TWIMS0 TWCK		GLOC OUT0	SCIF GCLK IN1	LCDCA SEG39	CATB SENSE18
98	D9	98	D9	PA25	25	VDDIO	USBC DM	USART2 RXD					CATB SENSE19
99	C9	99	C9	PA26	26	VDDIO	USBC DP	USART2 TXD					CATB SENSE20
		51	K1	PA27	27	LCDA	SPI MISO	IISC ISCK	ABDACB DAC0	GLOC IN4	USART3 RTS		CATB SENSE0
		52	J1	PA28	28	LCDA	SPI MOSI	IISC ISDI	ABDACB DACN0	GLOC IN5	USART3 CTS		CATB SENSE1
		53	K2	PA29	29	LCDA	SPI SCK	IISC IWS	ABDACB DAC1	GLOC IN6	USART3 CLK		CATB SENSE2
		56	K4	PA30	30	LCDA	SPI NPCS0	IISC ISDO	ABDACB DACN1	GLOC IN7	USART3 RXD		CATB SENSE3
		57	K5	PA31	31	LCDA	SPI NPCS1	IISC IMCK	ABDACB CLK	GLOC OUT1	USART3 TXD		CATB DIS
20	J3	20	J3	PB00	32	VDDIN	TWIMS1 TWD	USART0 RXD					CATB SENSE21
21	D5	21	D5	PB01	33	VDDIN	TWIMS1 TWCK	USART0 TXD	EIC EXTINT0				CATB SENSE22
22	E5	22	E5	PB02	34	VDDANA	ADCIFE AD3	USART1 RTS	ABDACB DAC0	IISC ISCK	ACIFC ACBN0		CATB SENSE23
23	C4	23	C4	PB03	35	VDDANA	ADCIFE AD4	USART1 CLK	ABDACB DACN0	IISC ISDI	ACIFC ACBP0		CATB DIS
28	C1	28	C1	PB04	36	VDDANA	ADCIFE AD5	USART1 RXD	ABDACB DAC1	IISC ISDO	DACC EXT TRIG0		CATB SENSE24
29	B1	29	B1	PB05	37	VDDANA	ADCIFE AD6	USART1 TXD	ABDACB DACN1	IISC IMCK			CATB SENSE25
45	G3	45	G3	PB06	38	LCDA	USART3 RTS		GLOC IN4	IISC IWS		LCDCA SEG22	CATB SENSE26
46	H1	46	H1	PB07	39	LCDA	USART3 CTS		GLOC IN5	TC0 A0		LCDCA SEG21	CATB SENSE27

Table 3-1. 100-pin GPIO Controller Function Multiplexing (Sheet 3 of 4)

ATSAM4LC		ATSAM4LS		Pin	GPIO	Supply	GPIO Functions						
QFN	VFBGA	QFN	VFBGA				A	B	C	D	E	F	G
72	G6	72	G6	PB08	40	LCDA	USART3 CLK		GLOC IN6	TC0 B0		LCDCA SEG14	CATB SENSE28
73	G7	73	G7	PB09	41	LCDA	USART3 RXD	PEVC PAD EVT2	GLOC IN7	TC0 A1		LCDCA SEG15	CATB SENSE29
74	G8	74	G8	PB10	42	LCDA	USART3 TXD	PEVC PAD EVT3	GLOC OUT1	TC0 B1	SCIF GCLK0	LCDCA SEG16	CATB SENSE30
75	K9	75	K9	PB11	43	LCDA	USART0 CTS	SPI NPCS2		TC0 A2	SCIF GCLK1	LCDCA SEG17	CATB SENSE31
89	E7	89	E7	PB12	44	LCDC	USART0 RTS	SPI NPCS3	PEVC PAD EVT0	TC0 B2	SCIF GCLK2	LCDCA SEG32	CATB DIS
90	E8	90	E8	PB13	45	LCDC	USART0 CLK	SPI NPCS1	PEVC PAD EVT1	TC0 CLK0	SCIF GCLK3	LCDCA SEG33	CATB SENSE0
93	D7	93	D7	PB14	46	LCDC	USART0 RXD	SPI MISO	TWIM3 TWD	TC0 CLK1	SCIF GCLK IN0	LCDCA SEG36	CATB SENSE1
94	D8	94	D8	PB15	47	LCDC	USART0 TXD	SPI MOSI	TWIM3 TWCK	TC0 CLK2	SCIF GCLK IN1	LCDCA SEG37	CATB SENSE2
1	A10	1	A10	PC00	64	VDDIO	SPI NPCS2	USART0 CLK		TC1 A0			CATB SENSE3
2	C8	2	C8	PC01	65	VDDIO	SPI NPCS3	USART0 RTS		TC1 B0			CATB SENSE4
3	C7	3	C7	PC02	66	VDDIO	SPI NPCS1	USART0 CTS	USART0 RXD	TC1 A1			CATB SENSE5
4	B7	4	B7	PC03	67	VDDIO	SPI NPCS0	EIC EXTINT5	USART0 TXD	TC1 B1			CATB SENSE6
9	C5	9	C5	PC04	68	VDDIO	SPI MISO	EIC EXTINT6		TC1 A2			CATB SENSE7
10	C6	10	C6	PC05	69	VDDIO	SPI MOSI	EIC EXTINT7		TC1 B2			CATB DIS
11	B6	11	B6	PC06	70	VDDIO	SPI SCK	EIC EXTINT8		TC1 CLK0			CATB SENSE8
36	F2	36	F2	PC07	71	VDDANA	ADCIFE AD7	USART2 RTS	PEVC PAD EVT0	TC1 CLK1			CATB SENSE9
37	E3	37	E3	PC08	72	VDDANA	ADCIFE AD8	USART2 CLK	PEVC PAD EVT1	TC1 CLK2	USART2 CTS		CATB SENSE10
38	F1	38	F1	PC09	73	VDDANA	ADCIFE AD9	USART3 RXD	ABDACB DAC0	IISC ISCK	ACIFC ACAN1		CATB SENSE11
39	D4	39	D4	PC10	74	VDDANA	ADCIFE AD10	USART3 TXD	ABDACB DACN0	IISC ISDI	ACIFC ACAP1		CATB SENSE12
40	E4	40	E4	PC11	75	VDDANA	ADCIFE AD11	USART2 RXD	PEVC PAD EVT2				CATB SENSE13
41	F3	41	F3	PC12	76	VDDANA	ADCIFE AD12	USART2 TXD	ABDACB CLK	IISC IWS			CATB SENSE14
42	F4	42	F4	PC13	77	VDDANA	ADCIFE AD13	USART3 RTS	ABDACB DAC1	IISC ISDO	ACIFC ACBN1		CATB SENSE15
43	G1	43	G1	PC14	78	VDDANA	ADCIFE AD14	USART3 CLK	ABDACB DACN1	IISC IMCK	ACIFC ACBP1		CATB DIS
58	J5	58	J5	PC15	79	LCDA	TC1 A0			GLOC IN4		LCDCA SEG0	CATB SENSE16

Table 3-1. 100-pin GPIO Controller Function Multiplexing (Sheet 4 of 4)

ATSAM4LC		ATSAM4LS		Pin	GPIO	Supply	GPIO Functions						
QFN	VFBGA	QFN	VFBGA				A	B	C	D	E	F	G
59	J6	59	J6	PC16	80	LCDA	TC1 B0			GLOC IN5		LCDCA SEG1	CATB SENSE17
60	H4	60	H4	PC17	81	LCDA	TC1 A1			GLOC IN6		LCDCA SEG2	CATB SENSE18
61	K6	61	K6	PC18	82	LCDA	TC1 B1			GLOC IN7		LCDCA SEG3	CATB SENSE19
62	G4	62	G4	PC19	83	LCDA	TC1 A2			GLOC OUT1		LCDCA SEG4	CATB SENSE20
68	H7	68	H7	PC20	84	LCDA	TC1 B2					LCDCA SEG10	CATB SENSE21
69	K8	69	K8	PC21	85	LCDA	TC1 CLK0			PARC PCCK		LCDCA SEG11	CATB SENSE22
70	J8	70	J8	PC22	86	LCDA	TC1 CLK1			PARC PCEN1		LCDCA SEG12	CATB SENSE23
71	H8	71	H8	PC23	87	LCDA	TC1 CLK2			PARC PCEN2		LCDCA SEG13	CATB DIS
79	J9	79	J9	PC24	88	LCDB	USART1 RTS	EIC EXTINT1	PEVC PAD EVT0	PARC PCDATA0		LCDCA SEG24	CATB SENSE24
80	H9	80	H9	PC25	89	LCDB	USART1 CLK	EIC EXTINT2	PEVC PAD EVT1	PARC PCDATA1		LCDCA SEG25	CATB SENSE25
81	G9	81	G9	PC26	90	LCDB	USART1 RXD	EIC EXTINT3	PEVC PAD EVT2	PARC PCDATA2	SCIF GCLK0	LCDCA SEG26	CATB SENSE26
82	F6	82	F6	PC27	91	LCDB	USART1 TXD	EIC EXTINT4	PEVC PAD EVT3	PARC PCDATA3	SCIF GCLK1	LCDCA SEG27	CATB SENSE27
83	G10	83	G10	PC28	92	LCDB	USART3 RXD	SPI MISO	GLOC IN4	PARC PCDATA4	SCIF GCLK2	LCDCA SEG28	CATB SENSE28
84	F7	84	F7	PC29	93	LCDB	USART3 TXD	SPI MOSI	GLOC IN5	PARC PCDATA5	SCIF GCLK3	LCDCA SEG29	CATB SENSE29
85	F8	85	F8	PC30	94	LCDB	USART3 RTS	SPI SCK	GLOC IN6	PARC PCDATA6	SCIF GCLK IN0	LCDCA SEG30	CATB SENSE30
86	F9	86	F9	PC31	95	LCDB	USART3 CLK	SPI NPCS0	GLOC OUT1	PARC PCDATA7	SCIF GCLK IN1	LCDCA SEG31	CATB SENSE31

Table 3-2. 64-pin GPIO Controller Function Multiplexing (Sheet 1 of 3)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
QFP QFN	QFP QFN				A	B	C	D	E	F	G
1	1	PA00	0	VDDIO							
2	2	PA01	1	VDDIO							
3	3	PA02	2	VDDIN	SCIF GCLK0	SPI NPCS0					CATB DIS
10	10	PA03	3	VDDIN		SPI MISO					

Table 3-2. 64-pin GPIO Controller Function Multiplexing (Sheet 2 of 3)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
QFP QFN	QFP QFN				A	B	C	D	E	F	G
15	15	PA04	4	VDDANA	ADCIFE AD0	USART0 CLK	EIC EXTINT2	GLOC IN1			CATB SENSE0
16	16	PA05	5	VDDANA	ADCIFE AD1	USART0 RXD	EIC EXTINT3	GLOC IN2	ADCIFE TRIGGER		CATB SENSE1
21	21	PA06	6	VDDANA	DACC VOUT	USART0 RTS	EIC EXTINT1	GLOC IN0	ACIFC ACAN0		CATB SENSE2
22	22	PA07	7	VDDANA	ADCIFE AD2	USART0 TXD	EIC EXTINT4	GLOC IN3	ACIFC ACAP0		CATB SENSE3
26	26	PA08	8	LCDA	USART0 RTS	TC0 A0	PEVC PAD EVT0	GLOC OUT0		LCDCA SEG23	CATB SENSE4
29	29	PA09	9	LCDA	USART0 CTS	TC0 B0	PEVC PAD EVT1	PARC PCDATA0		LCDCA COM3	CATB SENSE5
30	30	PA10	10	LCDA	USART0 CLK	TC0 A1	PEVC PAD EVT2	PARC PCDATA1		LCDCA COM2	CATB SENSE6
31	31	PA11	11	LCDA	USART0 RXD	TC0 B1	PEVC PAD EVT3	PARC PCDATA2		LCDCA COM1	CATB SENSE7
32	32	PA12	12	LCDA	USART0 TXD	TC0 A2		PARC PCDATA3		LCDCA COM0	CATB DIS
40	40	PA13	13	LCDA	USART1 RTS	TC0 B2	SPI NPCS1	PARC PCDATA4		LCDCA SEG5	CATB SENSE8
41	41	PA14	14	LCDA	USART1 CLK	TC0 CLK0	SPI NPCS2	PARC PCDATA5		LCDCA SEG6	CATB SENSE9
42	42	PA15	15	LCDA	USART1 RXD	TC0 CLK1	SPI NPCS3	PARC PCDATA6		LCDCA SEG7	CATB SENSE10
43	43	PA16	16	LCDA	USART1 TXD	TC0 CLK2	EIC EXTINT1	PARC PCDATA7		LCDCA SEG8	CATB SENSE11
44	44	PA17	17	LCDA	USART2 RTS	ABDACB DAC0	EIC EXTINT2	PARC PCCK		LCDCA SEG9	CATB SENSE12
49	49	PA18	18	LCDA	USART2 CLK	ABDACB DACN0	EIC EXTINT3	PARC PCEN1		LCDCA SEG18	CATB SENSE13
50	50	PA19	19	LCDA	USART2 RXD	ABDACB DAC1	EIC EXTINT4	PARC PCEN2	SCIF GCLK0	LCDCA SEG19	CATB SENSE14
51	51	PA20	20	LCDA	USART2 TXD	ABDACB DACN1	EIC EXTINT5	GLOC IN0	SCIF GCLK1	LCDCA SEG20	CATB SENSE15
55	55	PA21	21	LCDC	SPI MISO	USART1 CTS	EIC EXTINT6	GLOC IN1	TWIM2 TWD	LCDCA SEG34	CATB SENSE16
56	56	PA22	22	LCDC	SPI MOSI	USART2 CTS	EIC EXTINT7	GLOC IN2	TWIM2 TWCK	LCDCA SEG35	CATB SENSE17
59	59	PA23	23	LCDC	SPI SCK	TWIMS0 TWD	EIC EXTINT8	GLOC IN3	SCIF GCLK IN0	LCDCA SEG38	CATB DIS
60	60	PA24	24	LCDC	SPI NPCS0	TWIMS0 TWCK		GLOC OUT0	SCIF GCLK IN1	LCDCA SEG39	CATB SENSE18
62	62	PA25	25	VDDIO	USBC DM	USART2 RXD					CATB SENSE19
63	63	PA26	26	VDDIO	USBC DP	USART2 TXD					CATB SENSE20

Table 3-2. 64-pin GPIO Controller Function Multiplexing (Sheet 3 of 3)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
QFP QFN	QFP QFN				A	B	C	D	E	F	G
	33	PA27	27	LCDA	SPI MISO	IISC ISCK	ABDACB DAC0	GLOC IN4	USART3 RTS		CATB SENSE0
	34	PA28	28	LCDA	SPI MOSI	IISC ISDI	ABDACB DACN0	GLOC IN5	USART3 CTS		CATB SENSE1
	35	PA29	29	LCDA	SPI SCK	IISC IWS	ABDACB DAC1	GLOC IN6	USART3 CLK		CATB SENSE2
	38	PA30	30	LCDA	SPI NPCS0	IISC ISDO	ABDACB DACN1	GLOC IN7	USART3 RXD		CATB SENSE3
	39	PA31	31	LCDA	SPI NPCS1	IISC IMCK	ABDACB CLK	GLOC OUT1	USART3 TXD		CATB DIS
11	11	PB00	32	VDDIN	TWIMS1 TWD	USART0 RXD					CATB SENSE21
12	12	PB01	33	VDDIN	TWIMS1 TWCK	USART0 TXD	EIC EXTINT0				CATB SENSE22
13	13	PB02	34	VDDANA	ADCIFE AD3	USART1 RTS	ABDACB DAC0	IISC ISCK	ACIFC ACBN0		CATB SENSE23
14	14	PB03	35	VDDANA	ADCIFE AD4	USART1 CLK	ABDACB DACN0	IISC ISDI	ACIFC ACBP0		CATB DIS
19	19	PB04	36	VDDANA	ADCIFE AD5	USART1 RXD	ABDACB DAC1	IISC ISDO	DACC EXT TRIG0		CATB SENSE24
20	20	PB05	37	VDDANA	ADCIFE AD6	USART1 TXD	ABDACB DACN1	IISC IMCK			CATB SENSE25
27	27	PB06	38	LCDA	USART3 RTS		GLOC IN4	IISC IWS		LCDCA SEG22	CATB SENSE26
28	28	PB07	39	LCDA	USART3 CTS		GLOC IN5	TC0 A0		LCDCA SEG21	CATB SENSE27
45	45	PB08	40	LCDA	USART3 CLK		GLOC IN6	TC0 B0		LCDCA SEG14	CATB SENSE28
46	46	PB09	41	LCDA	USART3 RXD	PEVC PAD EVT2	GLOC IN7	TC0 A1		LCDCA SEG15	CATB SENSE29
47	47	PB10	42	LCDA	USART3 TXD	PEVC PAD EVT3	GLOC OUT1	TC0 B1	SCIF GCLK0	LCDCA SEG16	CATB SENSE30
48	48	PB11	43	LCDA	USART0 CTS	SPI NPCS2		TC0 A2	SCIF GCLK1	LCDCA SEG17	CATB SENSE31
53	53	PB12	44	LCDC	USART0 RTS	SPI NPCS3	PEVC PAD EVT0	TC0 B2	SCIF GCLK2	LCDCA SEG32	CATB DIS
54	54	PB13	45	LCDC	USART0 CLK	SPI NPCS1	PEVC PAD EVT1	TC0 CLK0	SCIF GCLK3	LCDCA SEG33	CATB SENSE0
57	57	PB14	46	LCDC	USART0 RXD	SPI MISO	TWIM3 TWD	TC0 CLK1	SCIF GCLK IN0	LCDCA SEG36	CATB SENSE1
58	58	PB15	47	LCDC	USART0 TXD	SPI MOSI	TWIM3 TWCK	TC0 CLK2	SCIF GCLK IN1	LCDCA SEG37	CATB SENSE2

Table 3-3. 64-pin GPIO Controller Function Multiplexing for WLCSP package (Sheet 1 of 3)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
WLCSP	WLCSP				A	B	C	D	E	F	G
G4	G4	PA00	0	VDDIO							
G5	G5	PA01	1	VDDIO							
F3	F3	PA02	2	VDDIN	SCIF GCLK0	SPI NPCS0					CATB DIS
E2	E2	PA03	3	VDDIN		SPI MISO					
D3	D3	PA04	4	VDDANA	ADCIFE AD0	USART0 CLK	EIC EXTINT2	GLOC IN1			CATB SENSE0
C3	C3	PA05	5	VDDANA	ADCIFE AD1	USART0 RXD	EIC EXTINT3	GLOC IN2	ADCIFE TRIGGER		CATB SENSE1
C4	C4	PA06	6	VDDANA	DACC VOUT	USART0 RTS	EIC EXTINT1	GLOC IN0	ACIFC ACAN0		CATB SENSE2
C5	C5	PA07	7	VDDANA	ADCIFE AD2	USART0 TXD	EIC EXTINT4	GLOC IN3	ACIFC ACAP0		CATB SENSE3
B4	B4	PA08	8	LCDA	USART0 RTS	TC0 A0	PEVC PAD EVT0	GLOC OUT0		LCDCA SEG23	CATB SENSE4
A5	A5	PA09	9	LCDA	USART0 CTS	TC0 B0	PEVC PAD EVT1	PARC PCDATA0		LCDCA COM3	CATB SENSE5
B6	B6	PA10	10	LCDA	USART0 CLK	TC0 A1	PEVC PAD EVT2	PARC PCDATA1		LCDCA COM2	CATB SENSE6
B7	B7	PA11	11	LCDA	USART0 RXD	TC0 B1	PEVC PAD EVT3	PARC PCDATA2		LCDCA COM1	CATB SENSE7
A8	A8	PA12	12	LCDA	USART0 TXD	TC0 A2		PARC PCDATA3		LCDCA COM0	CATB DIS
C7	C7	PA13	13	LCDA	USART1 RTS	TC0 B2	SPI NPCS1	PARC PCDATA4		LCDCA SEG5	CATB SENSE8
D7	D7	PA14	14	LCDA	USART1 CLK	TC0 CLK0	SPI NPCS2	PARC PCDATA5		LCDCA SEG6	CATB SENSE9
E7	E7	PA15	15	LCDA	USART1 RXD	TC0 CLK1	SPI NPCS3	PARC PCDATA6		LCDCA SEG7	CATB SENSE10
F7	F7	PA16	16	LCDA	USART1 TXD	TC0 CLK2	EIC EXTINT1	PARC PCDATA7		LCDCA SEG8	CATB SENSE11
G8	G8	PA17	17	LCDA	USART2 RTS	ABDACB DAC0	EIC EXTINT2	PARC PCCK		LCDCA SEG9	CATB SENSE12
G7	G7	PA18	18	LCDA	USART2 CLK	ABDACB DACN0	EIC EXTINT3	PARC PCEN1		LCDCA SEG18	CATB SENSE13
G6	G6	PA19	19	LCDA	USART2 RXD	ABDACB DAC1	EIC EXTINT4	PARC PCEN2	SCIF GCLK0	LCDCA SEG19	CATB SENSE14
H7	H7	PA20	20	LCDA	USART2 TXD	ABDACB DACN1	EIC EXTINT5	GLOC IN0	SCIF GCLK1	LCDCA SEG20	CATB SENSE15
H5	H5	PA21	21	LCDC	SPI MISO	USART1 CTS	EIC EXTINT6	GLOC IN1	TWIM2 TWD	LCDCA SEG34	CATB SENSE16
F5	F5	PA22	22	LCDC	SPI MOSI	USART2 CTS	EIC EXTINT7	GLOC IN2	TWIM2 TWCK	LCDCA SEG35	CATB SENSE17

Table 3-3. 64-pin GPIO Controller Function Multiplexing for WLCSP package (Sheet 2 of 3)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
WLCSP	WLCSP				A	B	C	D	E	F	G
H3	H3	PA23	23	LCDC	SPI SCK	TWIMS0 TWD	EIC EXTINT8	GLOC IN3	SCIF GCLK IN0	LCDCA SEG38	CATB DIS
G3	G3	PA24	24	LCDC	SPI NPCS0	TWIMS0 TWCK		GLOC OUT0	SCIF GCLK IN1	LCDCA SEG39	CATB SENSE18
H2	H2	PA25	25	VDDIO	USBC DM	USART2 RXD					CATB SENSE19
G2	G2	PA26	26	VDDIO	USBC DP	USART2 TXD					CATB SENSE20
	A7	PA27	27	LCDA	SPI MISO	IISC ISCK	ABDACB DAC0	GLOC IN4	USART3 RTS		CATB SENSE0
	A6	PA28	28	LCDA	SPI MOSI	IISC ISDI	ABDACB DACN0	GLOC IN5	USART3 CTS		CATB SENSE1
	B8	PA29	29	LCDA	SPI SCK	IISC IWS	ABDACB DAC1	GLOC IN6	USART3 CLK		CATB SENSE2
	E8	PA30	30	LCDA	SPI NPCS0	IISC ISDO	ABDACB DACN1	GLOC IN7	USART3 RXD		CATB SENSE3
	F8	PA31	31	LCDA	SPI NPCS1	IISC IMCK	ABDACB CLK	GLOC OUT1	USART3 TXD		CATB DIS
D2	D2	PB00	32	VDDIN	TWIMS1 TWD	USART0 RXD					CATB SENSE21
C2	C2	PB01	33	VDDIN	TWIMS1 TWCK	USART0 TXD	EIC EXTINT0				CATB SENSE22
E3	E3	PB02	34	VDDANA	ADCIFE AD3	USART1 RTS	ABDACB DAC0	IISC ISCK	ACIFC ACBN0		CATB SENSE23
B1	B1	PB03	35	VDDANA	ADCIFE AD4	USART1 CLK	ABDACB DACN0	IISC ISDI	ACIFC ACBP0		CATB DIS
A1	A1	PB04	36	VDDANA	ADCIFE AD5	USART1 RXD	ABDACB DAC1	IISC ISDO	DACC EXT TRIG0		CATB SENSE24
D4	D4	PB05	37	VDDANA	ADCIFE AD6	USART1 TXD	ABDACB DACN1	IISC IMCK			CATB SENSE25
B5	B5	PB06	38	LCDA	USART3 RTS		GLOC IN4	IISC IWS		LCDCA SEG22	CATB SENSE26
C6	C6	PB07	39	LCDA	USART3 CTS		GLOC IN5	TC0 A0		LCDCA SEG21	CATB SENSE27
D6	D6	PB08	40	LCDA	USART3 CLK		GLOC IN6	TC0 B0		LCDCA SEG14	CATB SENSE28
E6	E6	PB09	41	LCDA	USART3 RXD	PEVC PAD EVT2	GLOC IN7	TC0 A1		LCDCA SEG15	CATB SENSE29
F6	F6	PB10	42	LCDA	USART3 TXD	PEVC PAD EVT3	GLOC OUT1	TC0 B1	SCIF GCLK0	LCDCA SEG16	CATB SENSE30
H8	H8	PB11	43	LCDA	USART0 CTS	SPI NPCS2		TC0 A2	SCIF GCLK1	LCDCA SEG17	CATB SENSE31
D5	D5	PB12	44	LCDC	USART0 RTS	SPI NPCS3	PEVC PAD EVT0	TC0 B2	SCIF GCLK2	LCDCA SEG32	CATB DIS

Table 3-3. 64-pin GPIO Controller Function Multiplexing for WLCSP package (Sheet 3 of 3)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
WLCSP	WLCSP				A	B	C	D	E	F	G
E5	E5	PB13	45	LCDC	USART0 CLK	SPI NPCS1	PEVC PAD EVT1	TC0 CLK0	SCIF GCLK3	LCDCA SEG33	CATB SENSE0
F4	F4	PB14	46	LCDC	USART0 RXD	SPI MISO	TWIM3 TWD	TC0 CLK1	SCIF GCLK IN0	LCDCA SEG36	CATB SENSE1
H4	H4	PB15	47	LCDC	USART0 TXD	SPI MOSI	TWIM3 TWCK	TC0 CLK2	SCIF GCLK IN1	LCDCA SEG37	CATB SENSE2

Table 3-4. 48-pin GPIO Controller Function Multiplexing (Sheet 1 of 2)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
					A	B	C	D	E	F	G
1	1	PA00	0	VDDIO							
2	2	PA01	1	VDDIO							
3	3	PA02	2	VDDIN	SCIF GCLK0	SPI NPCS0					CATB DIS
10	10	PA03	3	VDDIN		SPI MISO					
11	11	PA04	4	VDDANA	ADCIFE AD0	USART0 CLK	EIC EXTINT2	GLOC IN1			CATB SENSE0
12	12	PA05	5	VDDANA	ADCIFE AD1	USART0 RXD	EIC EXTINT3	GLOC IN2	ADCIFE TRIGGER		CATB SENSE1
15	15	PA06	6	VDDANA	DACC VOUT	USART0 RTS	EIC EXTINT1	GLOC IN0	ACIFC ACAN0		CATB SENSE2
16	16	PA07	7	VDDANA	ADCIFE AD2	USART0 TXD	EIC EXTINT4	GLOC IN3	ACIFC ACAP0		CATB SENSE3
20	20	PA08	8	LCDA	USART0 RTS	TC0 A0	PEVC PAD EVT0	GLOC OUT0		LCDCA SEG23	CATB SENSE4
21	21	PA09	9	LCDA	USART0 CTS	TC0 B0	PEVC PAD EVT1	PARC PCDATA0		LCDCA COM3	CATB SENSE5
22	22	PA10	10	LCDA	USART0 CLK	TC0 A1	PEVC PAD EVT2	PARC PCDATA1		LCDCA COM2	CATB SENSE6
23	23	PA11	11	LCDA	USART0 RXD	TC0 B1	PEVC PAD EVT3	PARC PCDATA2		LCDCA COM1	CATB SENSE7
24	24	PA12	12	LCDA	USART0 TXD	TC0 A2		PARC PCDATA3		LCDCA COM0	CATB DIS
32	32	PA13	13	LCDA	USART1 RTS	TC0 B2	SPI NPCS1	PARC PCDATA4		LCDCA SEG5	CATB SENSE8
33	33	PA14	14	LCDA	USART1 CLK	TC0 CLK0	SPI NPCS2	PARC PCDATA5		LCDCA SEG6	CATB SENSE9
34	34	PA15	15	LCDA	USART1 RXD	TC0 CLK1	SPI NPCS3	PARC PCDATA6		LCDCA SEG7	CATB SENSE10
35	35	PA16	16	LCDA	USART1 TXD	TC0 CLK2	EIC EXTINT1	PARC PCDATA7		LCDCA SEG8	CATB SENSE11
36	36	PA17	17	LCDA	USART2 RTS	ABDACB DAC0	EIC EXTINT2	PARC PCCK		LCDCA SEG9	CATB SENSE12
37	37	PA18	18	LCDA	USART2 CLK	ABDACB DACN0	EIC EXTINT3	PARC PCEN1		LCDCA SEG18	CATB SENSE13
38	38	PA19	19	LCDA	USART2 RXD	ABDACB DAC1	EIC EXTINT4	PARC PCEN2	SCIF GCLK0	LCDCA SEG19	CATB SENSE14
39	39	PA20	20	LCDA	USART2 TXD	ABDACB DACN1	EIC EXTINT5	GLOC IN0	SCIF GCLK1	LCDCA SEG20	CATB SENSE15
41	41	PA21	21	LCDC	SPI MISO	USART1 CTS	EIC EXTINT6	GLOC IN1	TWIM2 TWD	LCDCA SEG34	CATB SENSE16
42	42	PA22	22	LCDC	SPI MOSI	USART2 CTS	EIC EXTINT7	GLOC IN2	TWIM2 TWCK	LCDCA SEG35	CATB SENSE17
43	43	PA23	23	LCDC	SPI SCK	TWIMS0 TWD	EIC EXTINT8	GLOC IN3	SCIF GCLK IN0	LCDCA SEG38	CATB DIS

Table 3-4. 48-pin GPIO Controller Function Multiplexing (Sheet 2 of 2)

ATSAM4LC	ATSAM4LS	Pin	GPIO	Supply	GPIO Functions						
					A	B	C	D	E	F	G
44	44	PA24	24	LCDC	SPI NPCS0	TWIMS0 TWCK		GLOC OUT0	SCIF GCLK IN1	LCDCA SEG39	CATB SENSE18
46	46	PA25	25	VDDIO	USBC DM	USART2 RXD					CATB SENSE19
47	47	PA26	26	VDDIO	USBC DP	USART2 TXD					CATB SENSE20
	25	PA27	27	LCDA	SPI MISO	IISC ISCK	ABDACB DAC0	GLOC IN4	USART3 RTS		CATB SENSE0
	26	PA28	28	LCDA	SPI MOSI	IISC ISDI	ABDACB DACN0	GLOC IN5	USART3 CTS		CATB SENSE1
	27	PA29	29	LCDA	SPI SCK	IISC IWS	ABDACB DAC1	GLOC IN6	USART3 CLK		CATB SENSE2
	30	PA30	30	LCDA	SPI NPCS0	IISC ISDO	ABDACB DACN1	GLOC IN7	USART3 RXD		CATB SENSE3
	31	PA31	31	LCDA	SPI NPCS1	IISC IMCK	ABDACB CLK	GLOC OUT1	USART3 TXD		CATB DIS

3.2.2 Peripheral Functions

Each GPIO line can be assigned to one of several peripheral functions. The following table describes how the various peripheral functions are selected. The last listed function has priority in case multiple functions are enabled on the same pin.

Table 3-5. Peripheral Functions

Function	Description
GPIO Controller Function multiplexing	GPIO and GPIO peripheral selection A to H
JTAG port connections	JTAG debug port
Oscillators	OSC0

3.2.3 JTAG Port Connections

If the JTAG is enabled, the JTAG will take control over a number of pins, irrespectively of the I/O Controller configuration.

Table 3-6. JTAG Pinout

48-pin Packages	64-pin QFP/QFN	64-pin WLSCP	100-pin QFN	100-ball VFBGA	Pin Name	JTAG Pin
10	10	E2	19	B3	PA03	TMS
43	59	H3	95	D6	PA23	TDO
44	60	G3	96	D10	PA24	TDI
9	9	F2	18	B4	TCK	TCK

3.2.4 ITM Trace Connections

If the ITM trace is enabled, the ITM will take control over the pin PA23, irrespectively of the I/O Controller configuration. The Serial Wire Trace signal is available on pin PA23

3.2.5 Oscillator Pinout

The oscillators are not mapped to the normal GPIO functions and their muxings are controlled by registers in the System Control Interface (SCIF) or Backup System Control Interface (BSCIF). Refer to the Section 13. ”System Control Interface (SCIF)” on page 203 and Section 12. ”Backup System Control Interface (BSCIF)” on page 161 for more information about this.

Table 3-7. Oscillator Pinout

48-pin Packages	64-pin QFN/QFP	64-pin WLCSP	100-pin Packages	100-ball VFBGA	Pin Name	Oscillator Pin
1	1	G4	5	B9	PA00	XIN0
13	17	B2	26	B2	XIN32	XIN32
2	2	G5	6	B8	PA01	XOUT0
14	18	B3	27	C2	XOUT32	XOUT32

3.3 Signals Description

The following table gives details on signal names classified by peripheral.

Table 3-8. Signal Descriptions List (Sheet 1 of 4)

Signal Name	Function	Type	Active Level	Comments
Audio Bitstream DAC - ABDACB				
CLK	D/A clock output	Output		
DAC1 - DAC0	D/A bitstream outputs	Output		
DACN1 - DACN0	D/A inverted bitstream outputs	Output		
Analog Comparator Interface - ACIFC				
ACAN1 - ACAN0	Analog Comparator A negative references	Analog		
ACAP1 - ACAP0	Analog Comparator A positive references	Analog		
ACBN1 - ACBN0	Analog Comparator B negative references	Analog		
ACBP1 - ACBP0	Analog Comparator B positive references	Analog		
ADC controller interface - ADCIFE				
AD14 - AD0	Analog inputs	Analog		
ADVREFP	Positive voltage reference	Analog		
TRIGGER	External trigger	Input		
Backup System Control Interface - BSCIF				
XIN32	32 kHz Crystal Oscillator Input	Analog/ Digital		
XOUT32	32 kHz Crystal Oscillator Output	Analog		
Capacitive Touch Module B - CATB				
DIS	Capacitive discharge line	Output		
SENSE31 - SENSE0	Capacitive sense lines	I/O		
DAC Controller - DACC				
DAC external trigger	DAC external trigger	Input		
DAC voltage output	DAC voltage output	Analog		
Enhanced Debug Port For ARM Products - EDP				
TCK/SWCLK	JTAG / SW Debug Clock	Input		
TDI	JTAG Debug Data In	Input		
TDO/TRACESWO	JTAG Debug Data Out / SW Trace Out	Output		
TMS/SWDIO	JTAG Debug Mode Select / SW Data	I/O		
External Interrupt Controller - EIC				
EXTINT8 - EXTINT0	External interrupts	Input		
Glue Logic Controller - GLOC				
IN7 - IN0	Lookup Tables Inputs	Input		
OUT1 - OUT0	Lookup Tables Outputs	Output		

Table 3-8. Signal Descriptions List (Sheet 2 of 4)

Signal Name	Function	Type	Active Level	Comments
Inter-IC Sound (I2S) Controller - IISC				
IMCK	I2S Master Clock	Output		
ISCK	I2S Serial Clock	I/O		
ISDI	I2S Serial Data In	Input		
ISDO	I2S Serial Data Out	Output		
IWS	I2S Word Select	I/O		
LCD Controller - LCDCA				
BIASL	Bias voltage (1/3 VLCD)	Analog		
BIASH	Bias voltage (2/3 VLCD)	Analog		
CAPH	High voltage end of flying capacitor	Analog		
CAPL	Low voltage end of flying capacitor	Analog		
COM3 - COM0	Common terminals	Analog		
SEG39 - SEG0	Segment terminals	Analog		
VLCD	Bias voltage	Analog		
Parallel Capture - PARC				
PCCK	Clock	Input		
PCDATA7 - PCDATA0	Data lines	Input		
PCEN1	Data enable 1	Input		
PCEN2	Data enable 2	Input		
Peripheral Event Controller - PEVC				
PAD_EVT3 - PAD_EVT0	Event Inputs	Input		
Power Manager - PM				
RESET_N	Reset	Input	Low	
System Control Interface - SCIF				
GCLK3 - GCLK0	Generic Clock Outputs	Output		
GCLK_IN1 - GCLK_IN0	Generic Clock Inputs	Input		
XIN0	Crystal 0 Input	Analog/ Digital		
XOUT0	Crystal 0 Output	Analog		
Serial Peripheral Interface - SPI				
MISO	Master In Slave Out	I/O		
MOSI	Master Out Slave In	I/O		
NPCS3 - NPCS0	SPI Peripheral Chip Selects	I/O	Low	
SCK	Clock	I/O		
Timer/Counter - TC0, TC1				

Table 3-8. Signal Descriptions List (Sheet 3 of 4)

Signal Name	Function	Type	Active Level	Comments
A0	Channel 0 Line A	I/O		
A1	Channel 1 Line A	I/O		
A2	Channel 2 Line A	I/O		
B0	Channel 0 Line B	I/O		
B1	Channel 1 Line B	I/O		
B2	Channel 2 Line B	I/O		
CLK0	Channel 0 External Clock Input	Input		
CLK1	Channel 1 External Clock Input	Input		
CLK2	Channel 2 External Clock Input	Input		
Two-wire Interface - TWIMS0, TWIMS1, TWIM2, TWIM3				
TWCK	Two-wire Serial Clock	I/O		
TWD	Two-wire Serial Data	I/O		
Universal Synchronous Asynchronous Receiver Transmitter - USART0, USART1, USART2, USART3				
CLK	Clock	I/O		
CTS	Clear To Send	Input	Low	
RTS	Request To Send	Output	Low	
RXD	Receive Data	Input		
TXD	Transmit Data	Output		
USB 2.0 Interface - USBC				
DM	USB Full Speed Interface Data -	I/O		
DP	USB Full Speed Interface Data +	I/O		
Power				
GND	Ground	Ground		
GNDANA	Analog Ground	Ground		
VDDANA	Analog Power Supply	Power Input		1.68V to 3.6V
VDDCORE	Core Power Supply	Power Input		1.68V to 1.98V
VDDIN	Voltage Regulator Input	Power Input		1.68V to 3.6V
VDDIO	I/O Pads Power Supply	Power Input		1.68V to 3.6V. VDDIO must always be equal to or lower than VDDIN.
VDDOUT	Voltage Regulator Output	Power Output		1.08V to 1.98V
General Purpose I/O				

Table 3-8. Signal Descriptions List (Sheet 4 of 4)

Signal Name	Function	Type	Active Level	Comments
PA31 - PA00	Parallel I/O Controller I/O Port A	I/O		
PB15 - PB00	Parallel I/O Controller I/O Port B	I/O		
PC31 - PC00	Parallel I/O Controller I/O Port C	I/O		

Note: 1. See "Power and Startup Considerations" section.

3.4 I/O Line Considerations

3.4.1 SW/JTAG Pins

The JTAG pins switch to the JTAG functions if a rising edge is detected on TCK low after the RESET_N pin has been released. The TMS, and TDI pins have pull-up resistors when used as JTAG pins. The TCK pin always has pull-up enabled during reset. The JTAG pins can be used as GPIO pins and multiplexed with peripherals when the JTAG is disabled. Refer to Section 3.2.3 "JTAG Port Connections" on page 29 for the JTAG port connections.

For more details, refer to Section 8.7 "Enhanced Debug Port (EDP)" on page 67.

3.4.2 RESET_N Pin

The RESET_N pin is a schmitt input and integrates a permanent pull-up resistor to VDDIN. As the product integrates a power-on reset detector, the RESET_N pin can be left unconnected in case no reset from the system needs to be applied to the product.

3.4.3 TWI Pins

When these pins are used for TWI, the pins are open-drain outputs with slew-rate limitation and-inputs with inputs with spike-filtering. When used as GPIO-pins or used for other peripherals, the pins have the same characteristics as GPIO pins.

3.4.4 GPIO Pins

All the I/O lines integrate a pull-up/pull-down resistor and slew rate controller. Programming these features is performed independently for each I/O line through the GPIO Controllers. After reset, I/O lines default as inputs with pull-up and pull-down resistors disabled and slew rate enabled.

3.4.5 High-drive Pins

The six pins PA02, PB00, PB01, PC04, PC05 and PC06 have high-drive output capabilities. Refer to Section 42.6.2 "High-drive I/O Pin : PA02, PC04, PC05, PC06" on page 1137 for electrical characteristics.

3.4.6 USB Pins

When these pins are used for USB, the pins are behaving according to the USB specification. When used as GPIO pins or used for other peripherals, the pins have the same behavior as other normal I/O pins, but the characteristics are different. Refer to Section 42.6.3 "USB I/O Pin : PA25, PA26" on page 1138 for electrical characteristics.

These pins are compliant to USB standard only when VDDIO power supply is 3.3V nominal.

3.4.7 ADC Input Pins

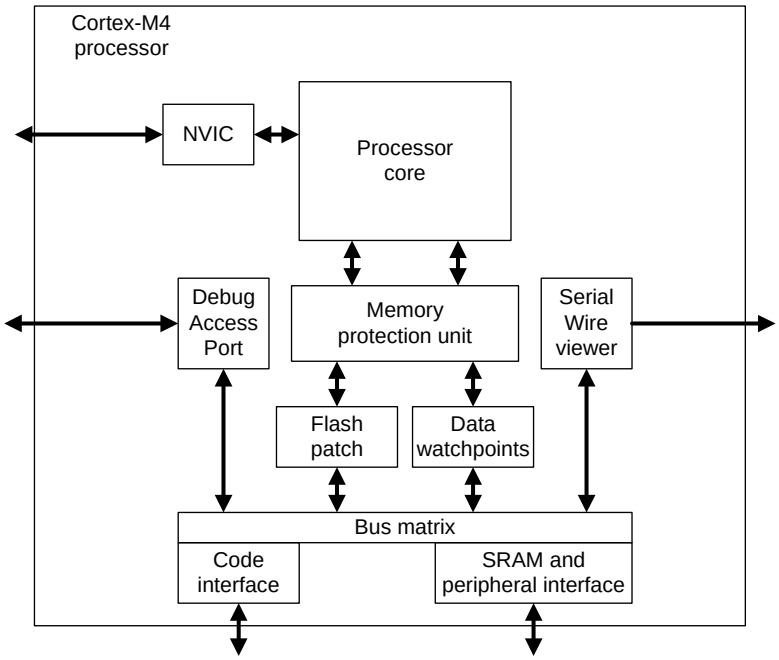
These pins are regular I/O pins powered from the VDDANA.

4. Cortex-M4 processor and core peripherals

4.1 Cortex-M4

The Cortex-M4 processor is a high performance 32-bit processor designed for the microcontroller market. It offers significant benefits to developers, including:

- **outstanding processing performance combined with fast interrupt handling**
- **enhanced system debug with extensive breakpoint and trace capabilities**
- **efficient processor core, system and memories**
- **ultra-low power consumption with integrated sleep modes**
- **platform security robustness, with integrated memory protection unit (MPU).**



The Cortex-M4 processor is built on a high-performance processor core, with a 3-stage pipeline Harvard architecture, making it ideal for demanding embedded applications. The processor delivers exceptional power efficiency through an efficient instruction set and extensively optimized design, providing high-end processing hardware including a range of single-cycle and SIMD multiplication and multiply-with-accumulate capabilities, saturating arithmetic and dedicated hardware division.

To facilitate the design of cost-sensitive devices, the Cortex-M4 processor implements tightly-coupled system components that reduce processor area while significantly improving interrupt handling and system debug capabilities. The Cortex-M4 processor implements a version of the Thumb® instruction set based on Thumb-2 technology, ensuring high code density and reduced program memory requirements. The Cortex-M4 instruction set provides the exceptional performance expected of a modern 32-bit architecture, with the high code density of 8-bit and 16-bit microcontrollers.

The Cortex-M4 processor closely integrates a configurable Nested Vectored Interrupt Controller (NVIC), to deliver industry-leading interrupt performance. The NVIC includes a *non-maskable interrupt* (NMI), and provides up to 80 interrupt priority levels. The tight integration of the proces-

sor core and NVIC provides fast execution of interrupt service routines (ISRs), dramatically reducing the interrupt latency. This is achieved through the hardware stacking of registers, and the ability to suspend load-multiple and store-multiple operations. Interrupt handlers do not require wrapping in assembler code, removing any code overhead from the ISRs. A tail-chain optimization also significantly reduces the overhead when switching from one ISR to another.

To optimize low-power designs, the NVIC integrates with the sleep modes, that include a deep sleep function enabling the entire device to be rapidly powered down while still retaining program state.

4.2 System level interface

The Cortex-M4 processor provides multiple interfaces using AMBA® technology to provide high speed, low latency memory accesses. It supports unaligned data accesses and implements atomic bit manipulation that enables faster peripheral controls, system spinlocks and thread-safe Boolean data handling.

The Cortex-M4 processor has an *memory protection unit* (MPU) that provides fine grain memory control, enabling applications to utilize multiple privilege levels, separating and protecting code, data and stack on a task-by-task basis. Such requirements are becoming critical in many embedded applications such as automotive.

4.3 Integrated configurable debug

The Cortex-M4 processor implements a complete hardware debug solution. This provides high system visibility of the processor and memory through either a traditional JTAG port or a 2-pin *Serial Wire Debug* (SWD) port that is ideal for microcontrollers and other small package devices.

For system trace the processor integrates an *Instrumentation Trace Macrocell* (ITM) alongside data watchpoints and a profiling unit. To enable simple and cost-effective profiling of the system events these generate, a *Serial Wire Viewer* (SWV) can export a stream of software-generated messages, data trace, and profiling information through a single pin.

The *Flash Patch and Breakpoint Unit* (FPB) provides 8 hardware breakpoint comparators that debuggers can use. The comparators in the FPB also provide remap functions of up to 8 words in the program code in the CODE memory region. This enables applications stored on a non-erasable, ROM-based microcontroller to be patched if a small programmable memory, for example flash, is available in the device. During initialization, the application in ROM detects, from the programmable memory, whether a patch is required. If a patch is required, the application programs the FPB to remap a number of addresses. When those addresses are accessed, the accesses are redirected to a remap table specified in the FPB configuration, which means the program in the non-modifiable ROM can be patched.

A specific Peripheral Debug (PDBG) register is implemented in the Private Peripheral Bus address map. This register allows the user to configure the behavior of some modules in debug mode.

4.4 Cortex-M4 processor features and benefits summary

- **tight integration of system peripherals reduces area and development costs**
- **Thumb instruction set combines high code density with 32-bit performance**
- **code-patch ability for ROM system updates**
- **power control optimization of system components**
- **integrated sleep modes for low power consumption**
- **fast code execution permits slower processor clock or increases sleep mode time**
- **hardware division and fast digital-signal-processing orientated multiply accumulate**
- **saturating arithmetic for signal processing**
- **deterministic, high-performance interrupt handling for time-critical applications**
- ***memory protection unit* (MPU) for safety-critical applications**
- **extensive debug and trace capabilities:**
  - **Serial Wire Debug and Serial Wire Trace reduce the number of pins required for debugging, tracing, and code profiling.**

4.5 Cortex-M4 core peripherals

These are:

Nested Vectored Interrupt Controller

The NVIC is an embedded interrupt controller that supports low latency interrupt processing.

System control block

The *System control block* (SCB) is the programmers model interface to the processor. It provides system implementation information and system control, including configuration, control, and reporting of system exceptions.

System timer

The system timer, SysTick, is a 24-bit count-down timer. Use this as a Real Time Operating System (RTOS) tick timer or as a simple counter.

Memory protection unit

The *Memory protection unit* (MPU) improves system reliability by defining the memory attributes for different memory regions. It provides up to eight different regions, and an optional predefined background region.

The complete Cortex-M4 User Guide can be found on the ARM web site:

http://infocenter.arm.com/help/topic/com.arm.doc.dui0553a/DUI0553A_cortex_m4_dgug.pdf

4.6 Cortex-M4 implementations options

This table provides the specific configuration options implemented in the SAM4L series

Option	Implementation
Inclusion of MPU	yes
Inclusion of FPU	No
Number of interrupts	80
Number of priority bits	4
Inclusion of the WIC	No
Embedded Trace Macrocell	No
Sleep mode instruction	Only WFI supported
Endianness	Little Endian
Bit-banding	No
SysTick timer	Yes
Register reset values	No

Table 4-1. Cortex-M4 implementation options

4.7 Cortex-M4 Interrupts map

The table below shows how the interrupt request signals are connected to the NVIC.

Table 4-2. Interrupt Request Signal Map (Sheet 1 of 3)

Line	Module	Signal
0	Flash Controller	HFLASHC
1	Peripheral DMA Controller	PDCA 0
2	Peripheral DMA Controller	PDCA 1
3	Peripheral DMA Controller	PDCA 2
4	Peripheral DMA Controller	PDCA 3
5	Peripheral DMA Controller	PDCA 4
6	Peripheral DMA Controller	PDCA 5
7	Peripheral DMA Controller	PDCA 6
8	Peripheral DMA Controller	PDCA 7
9	Peripheral DMA Controller	PDCA 8
10	Peripheral DMA Controller	PDCA 9
11	Peripheral DMA Controller	PDCA 10

Table 4-2. Interrupt Request Signal Map (Sheet 2 of 3)

Line	Module	Signal
12	Peripheral DMA Controller	PDCA 11
13	Peripheral DMA Controller	PDCA 12
14	Peripheral DMA Controller	PDCA 13
15	Peripheral DMA Controller	PDCA 14
16	Peripheral DMA Controller	PDCA 15
17	CRC Calculation Unit	CRCCU
18	USB 2.0 Interface	USBC
19	Peripheral Event Controller	PEVC TR
20	Peripheral Event Controller	PEVC OV
21	Advanced Encryption Standard	AESA
22	Power Manager	PM
23	System Control Interface	SCIF
24	Frequency Meter	FREQM
25	General-Purpose Input/Output Controller	GPIO 0
26	General-Purpose Input/Output Controller	GPIO 1
27	General-Purpose Input/Output Controller	GPIO 2
28	General-Purpose Input/Output Controller	GPIO 3
29	General-Purpose Input/Output Controller	GPIO 4
30	General-Purpose Input/Output Controller	GPIO 5
31	General-Purpose Input/Output Controller	GPIO 6
32	General-Purpose Input/Output Controller	GPIO 7
33	General-Purpose Input/Output Controller	GPIO 8
34	General-Purpose Input/Output Controller	GPIO 9
35	General-Purpose Input/Output Controller	GPIO 10
36	General-Purpose Input/Output Controller	GPIO 11
37	Backup Power Manager	BPM
38	Backup System Control Interface	BSCIF
39	Asynchronous Timer	AST ALARM
40	Asynchronous Timer	AST PER
41	Asynchronous Timer	AST OVF
42	Asynchronous Timer	AST READY
43	Asynchronous Timer	AST CLKREADY
44	Watchdog Timer	WDT
45	External Interrupt Controller	EIC 1
46	External Interrupt Controller	EIC 2
47	External Interrupt Controller	EIC 3

Table 4-2. Interrupt Request Signal Map (Sheet 3 of 3)

Line	Module	Signal
48	External Interrupt Controller	EIC 4
49	External Interrupt Controller	EIC 5
50	External Interrupt Controller	EIC 6
51	External Interrupt Controller	EIC 7
52	External Interrupt Controller	EIC 8
53	Inter-IC Sound (I2S) Controller	IISC
54	Serial Peripheral Interface	SPI
55	Timer/Counter	TC00
56	Timer/Counter	TC01
57	Timer/Counter	TC02
58	Timer/Counter	TC10
59	Timer/Counter	TC11
60	Timer/Counter	TC12
61	Two-wire Master Interface	TWIM0
62	Two-wire Slave Interface	TWIS0
63	Two-wire Master Interface	TWIM1
64	Two-wire Slave Interface	TWIS1
65	Universal Synchronous Asynchronous Receiver Transmitter	USART0
66	Universal Synchronous Asynchronous Receiver Transmitter	USART1
67	Universal Synchronous Asynchronous Receiver Transmitter	USART2
68	Universal Synchronous Asynchronous Receiver Transmitter	USART3
69	ADC controller interface	ADCIFE
70	DAC Controller	DACC
71	Analog Comparator Interface	ACIFC
72	Audio Bitstream DAC	ABDACB
73	True Random Number Generator	TRNG
74	Parallel Capture	PARC
75	Capacitive Touch Module B	CATB
77	Two-wire Master Interface	TWIM2
78	Two-wire Master Interface	TWIM3
79	LCD Controller A	LCDCA

4.8 Peripheral Debug

The PDBG register controls the behavior of asynchronous peripherals when the device is in debug mode.When the corresponding bit is set, that peripheral will be in a frozenstate in debug mode.

4.8.1 Peripheral Debug

Name: PDBG

Access Type: Read/Write

Address: 0xE0042000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	PEVC	AST	WDT

- **WDT: Watchdog PDBG bit**
  WDT = 0: The WDT counter is not frozen during debug operation.
  WDT = 1: The WDT counter is frozen during debug operation when Core is halted
- **AST: Asynchronous Timer PDBG bit**
  AST = 0: The AST prescaler and counter is not frozen during debug operation.
  AST = 1: The AST prescaler and counter is frozen during debug operation when Core is halted.
- **PEVC: PEVC PDBG bit**
  PEVC= 0: PEVC is not frozen during debug operation.
  PEVC= 1: PEVC is frozen during debug operation when Core is halted.

5. Power and Startup Considerations

5.1 Power Domain Overview

Figure 5-1. ATSAM4LS Power Domain Diagram

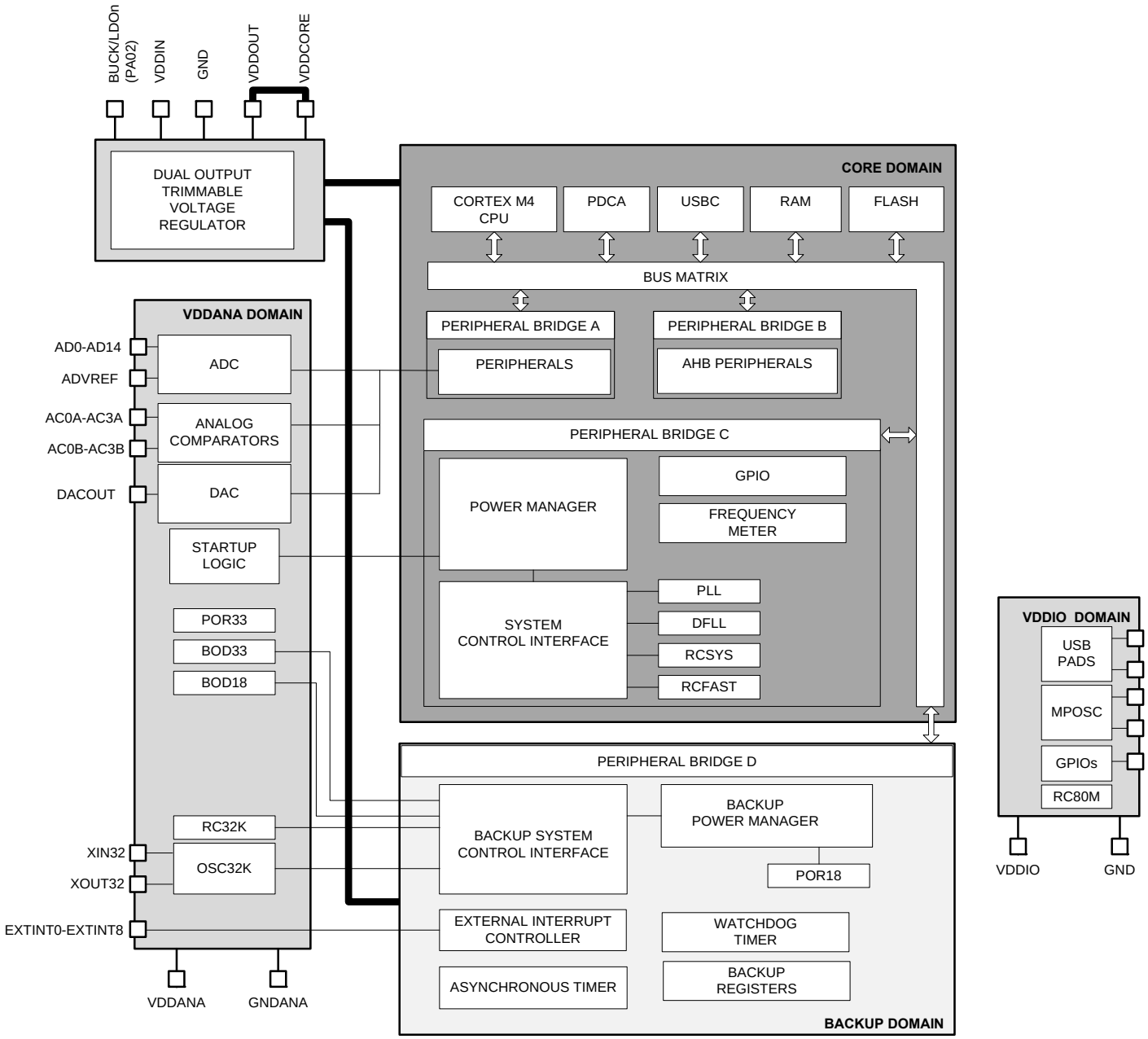
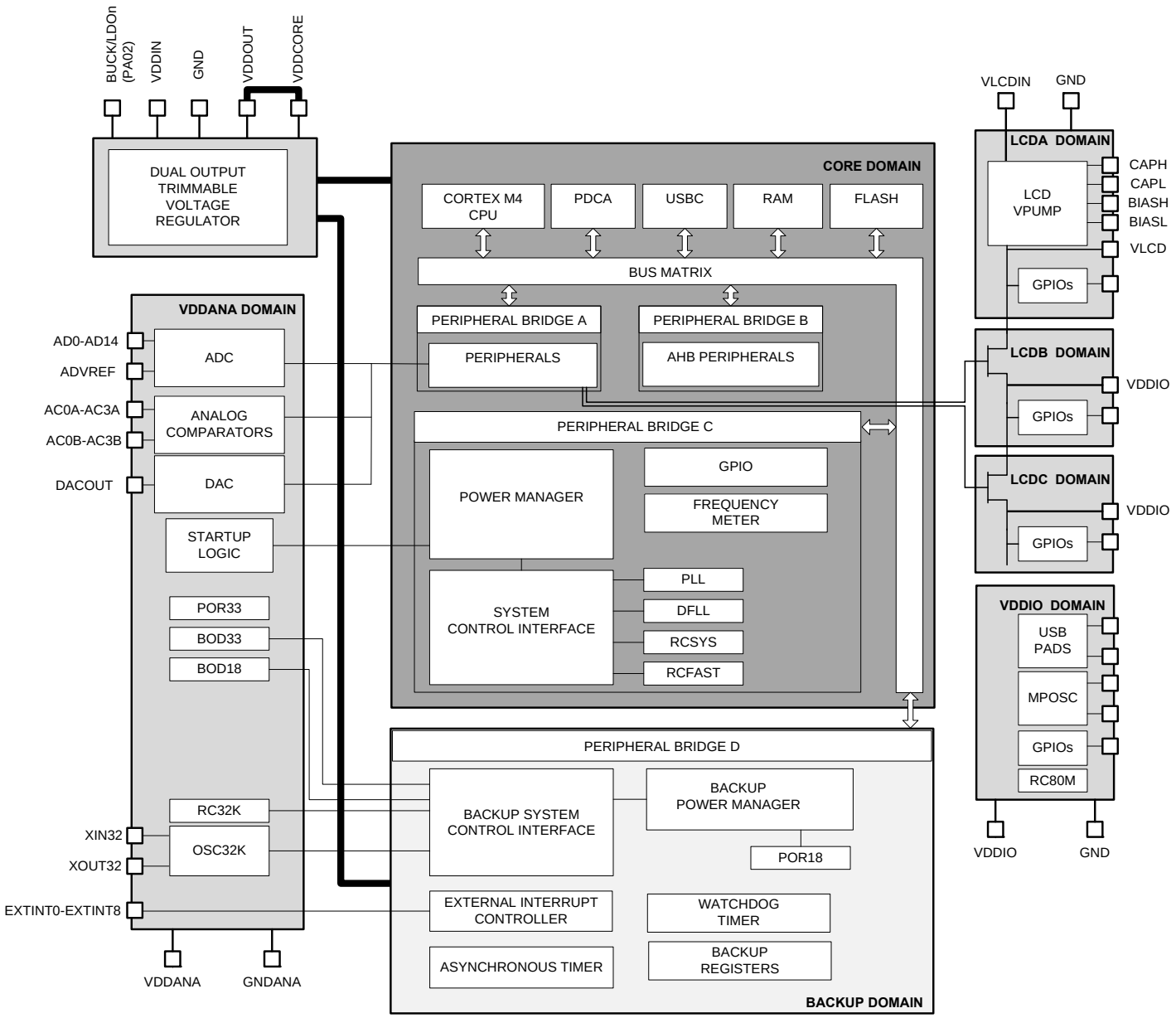


Figure 5-2. ATSAM4LC Power Domain Diagram



5.2 Power Supplies

The ATSAM4L8/L4/L2 has several types of power supply pins:

- VDDIO: Powers I/O lines, the general purpose oscillator (OSC), the 80MHz integrated RC oscillator (RC80M) . Voltage is 1.68V to 3.6V.
- VLCDIN: (ATSAM4LC only) Powers the LCD voltage pump. Voltage is 1.68V to 3.6V.
- VDDIN: Powers the internal voltage regulator. Voltage is 1.68V to 3.6V.
- VDDANA: Powers the ADC, the DAC, the Analog Comparators, the 32kHz oscillator (OSC32K), the 32kHz integrated RC oscillator (RC32K)and the Brown-out detectors (BOD18 and BOD33). Voltage is 1.68V to 3.6V nominal.
- VDDCORE: Powers the core, memories, peripherals, the PLL, the DFLL, the 4MHz integrated RC oscillator (RCFAST) and the 115kHz integrated RC oscillator (RCSYS).
  - VDDOUT is the output voltage of the regulator and must be connected with or without an inductor to VDDCORE.

The ground pins GND are common to VDDCORE, VDDIO, and VDDIN. The ground pin for VDDANA is GNDANA.

For decoupling recommendations for the different power supplies, refer to the schematic document.

5.2.1 Voltage Regulator

An embedded voltage regulator supplies all the digital logic in the Core and the Backup power domains.

The regulator has two functionnal mode depending of BUCK/LDOn (PA02) pin value. When this pin is low, the regulator is in linear mode and VDDOUT must be connected to VDDCORE externally. When this pin is high, it behaves as a switching regulator and an inductor must be placed between VDDOUT and VDDCORE. The value of this pin is sampled during the power-up phase when the Power On Reset 33 reaches V_{POT+} (Section 42.9 ”Analog Characteristics” on page 1151)

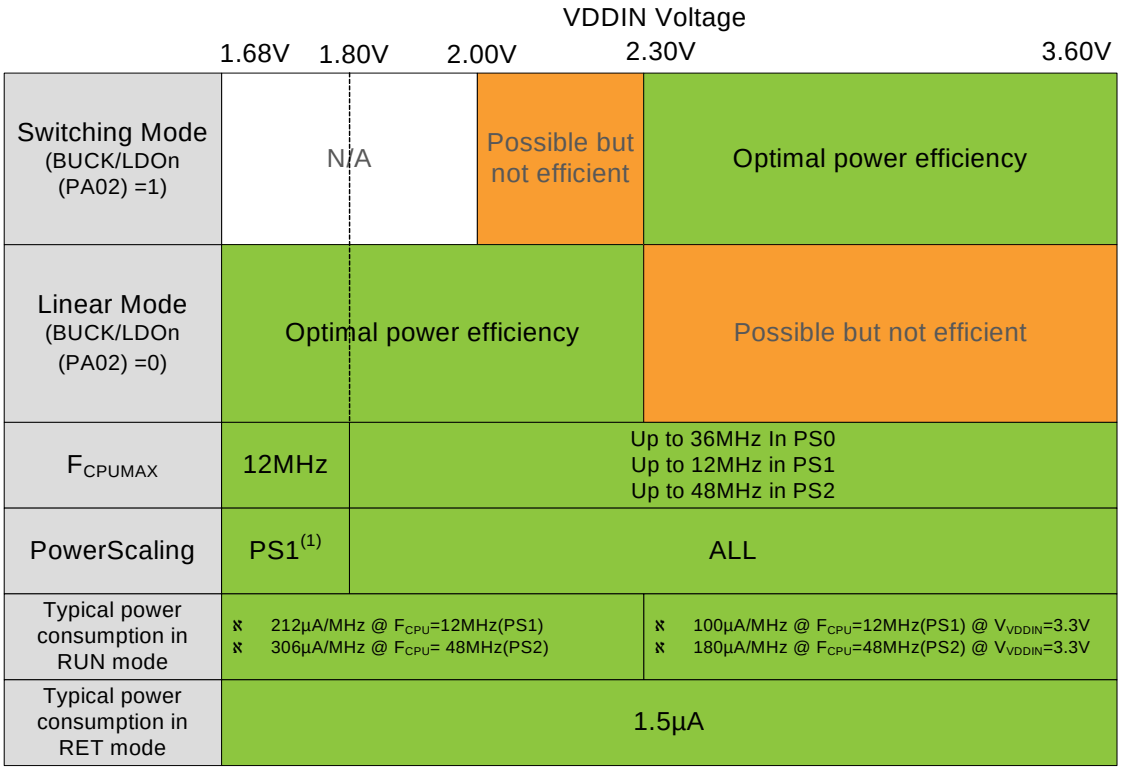
Its output voltages in the Core domain (V_{CORE}) and in the Backup domain (V_{BKUP}) are always equal except in Backup mode where the Core domain is not powered (V_{CORE}=0). The Backup domain is always powered. The voltage regulator features three different modes:

- Normal mode: the regulator is configured as linear or switching regulator. It can support all different Run and Sleep modes.
- Low Power (LP) mode: the regulator consumes little static current. It can be used in Wait modes.
- Ultra Low Power (ULP) mode: the regulator consumes very little static current . It is dedicated to Retention and Backup modes. In Backup mode, the regulator only supplies the backup domain.

5.2.2 Typical Powering Schematics

The ATSAM4L8/L4/L2 supports the Single supply mode from 1.68V to 3.6V. Depending on the input voltage range and on the final application frequency, it is recommended to use the following table in order to choose the most efficient power strategy

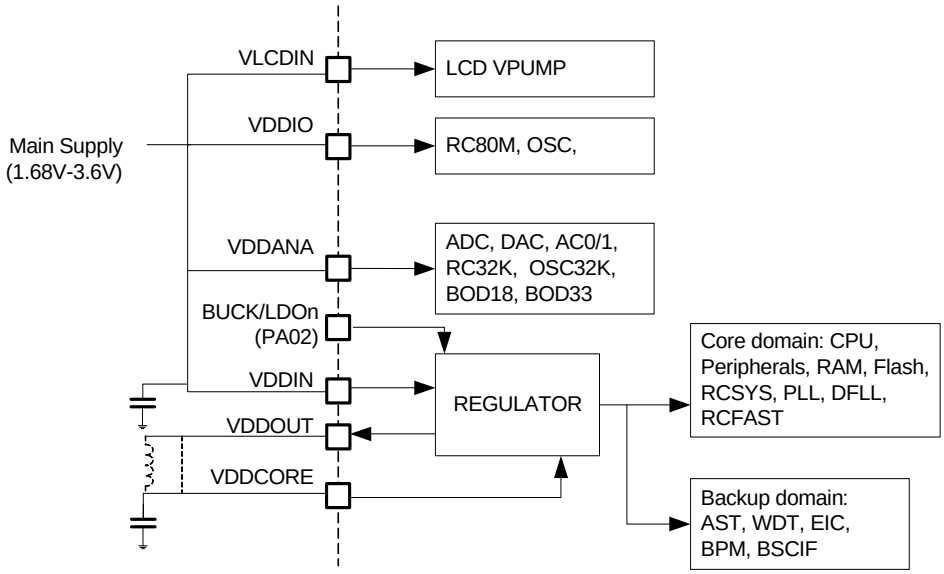
Figure 5-3. Efficient power strategy:



Note 1. The SAM4L boots in PS0 on RCSYS(115kHz), then the application must switch to PS1 before running on higher frequency (<12MHz)

The internal regulator is connected to the VDDIN pin and its output VDDOUT feeds VDDCORE in linear mode or through an inductor in switching mode. Figure 5-4 shows the power schematics to be used. All I/O lines will be powered by the same power (V_{VDDIN}=V_{VDDIO}=V_{VDDANA}).

Figure 5-4. Single Supply Mode



5.2.3 LCD Power Modes

5.2.3.1 Principle

LCD lines is powered using the device internal voltage sources provided by the LCDPWR block. When enabled, the LCDPWR blocks will generate the VLCD, BIASL, BIASH voltages.

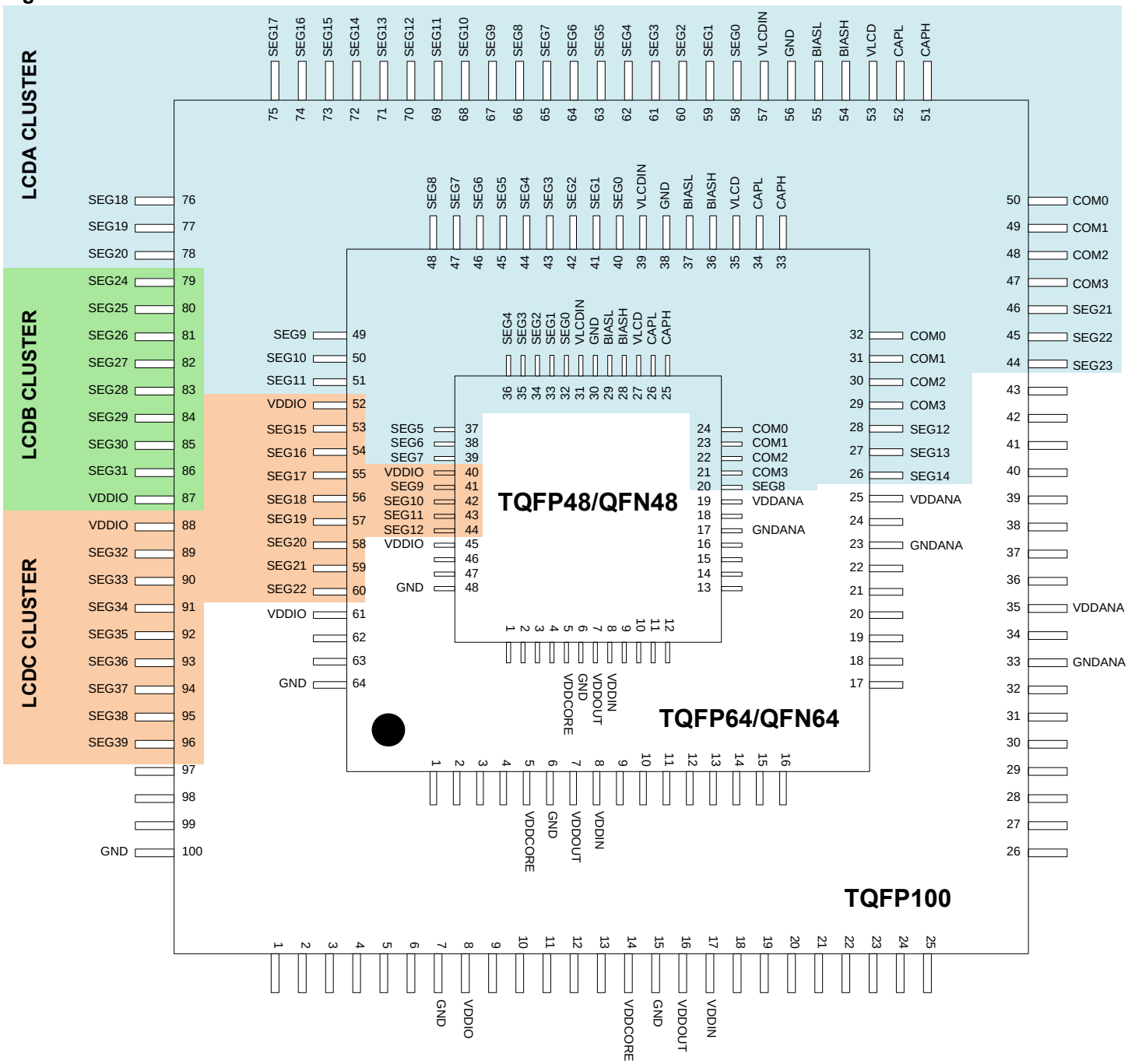
LCD pads are splitted into three clusters that can be powered independently namely clusters A, B and C. A cluster can either be in GPIO mode or in LCD mode.

When a cluster is in GPIO mode, its VDDIO pin must be powered externally. None of its GPIO pin can be used as a LCD line

When a cluster is in LCD mode, each clusters VDDIO pin can be either forced externally (1.8-3.6V) or unconnected (nc). GPIOs in a cluster are not available when it is in LCD mode. A cluster is set in LCD mode by the LCDCA controller when it is enabled depending on the number of segments configured. The LCDPWR block is powered by the VLCDIN pin inside cluster A

When LCD feature is not used, VLCDIN must be always powered (1.8-3.6V). VLCD, CAPH, CAPL, BIASH, BIASL can be left unconnected in this case

Figure 5-5. LCD clusters in the device



5.2.3.2 *Internal LCD Voltage*

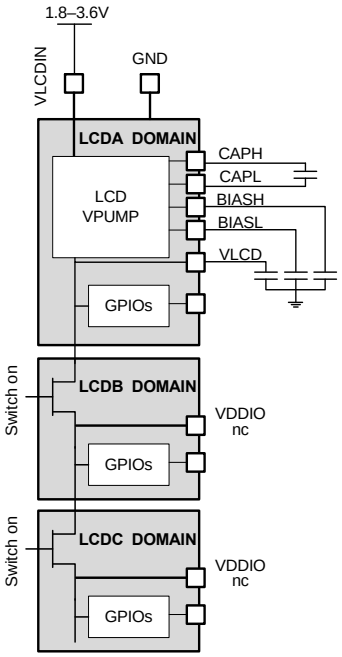
In this mode the LCD voltages are internally generated. Depending of the number of segments required by the application, LCDB and LDCC clusters VDDIO pin must be unconnected (nc) or

connected to an external voltage source (1.8-3.6V). LCDB cluster is not available in 64 and 48 pin packages

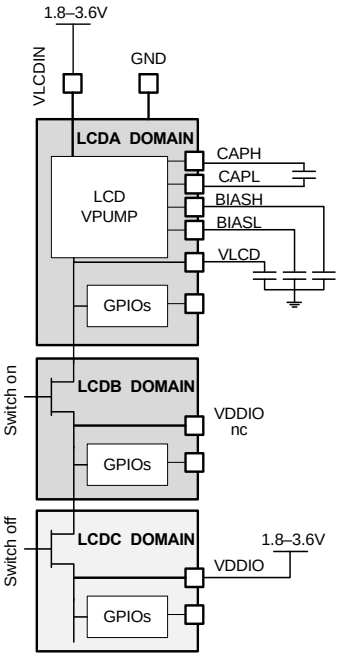
Table 5-1. LCD powering when using the internal voltage pump

Package	Segments in use	VDDIO LCDB	VDDIO LCDC
100-pin packages	[1,24]	1.8-3.6V	1.8-3.6V
	[1, 32]	nc	1.8-3.6V
	[1, 40]	nc	nc
64-pin packages	[1,15]	-	1.8-3.6V
	[1, 23]	-	nc
48-pin packages	[1,9]	-	1.8-3.6V
	[1,13]	-	nc

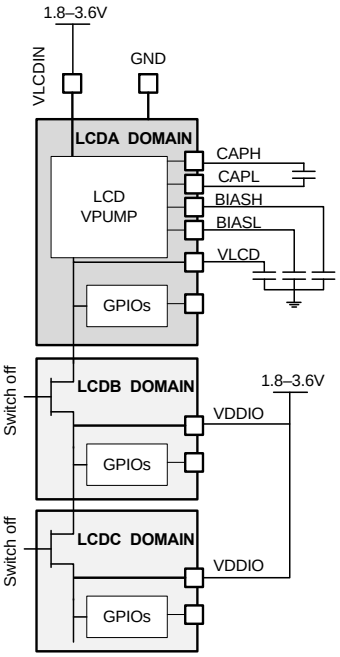
Up to 4x40 segments
No GPIO in LCD clusters



Up to 4x32 segments
Up to 8 GPIOs in LCDC clusters



Up to 4x24 segments
Up to 16 GPIOs in LCDB & LCDC clusters



5.2.4 Power-up Sequence

5.2.4.1 Maximum Rise Rate

To avoid risk of latch-up, the rise rate of the power supplies must not exceed the values described in Table 42-3 on page 1122.

5.2.4.2 Minimum Rise Rate

The integrated Power-on Reset (POR33) circuitry monitoring the VDDIN powering supply requires a minimum rise rate for the VDDIN power supply.

See Table 42-3 on page 1122 for the minimum rise rate value.

If the application can not ensure that the minimum rise rate condition for the VDDIN power supply is met, the following configuration can be used:

- A logic “0” value is applied during power-up on pin RESET_N until VDDIN rises above 1.6 V.

5.3 Startup Considerations

This section summarizes the boot sequence of the ATSAM4L8/L4/L2. The behavior after power-up is controlled by the Power Manager. For specific details, refer to Section 10. ”Power Manager (PM)” on page 109.

5.3.1 Starting of Clocks

After power-up, the device will be held in a reset state by the power-up circuitry for a short time to allow the power to stabilize throughout the device. After reset, the device will use the System RC Oscillator (RCSYS) as clock source. Refer to Section 42. ”Electrical Characteristics” on page 1121 for the frequency for this oscillator.

On system start-up, the DFLL and the PLLs are disabled. Only the necessary clocks are active allowing software execution. Refer to Section 10-5 ”Maskable Module Clocks in ATSAM4L.” on page 120 to know the list of peripheral clock running.. No clocks have a divided frequency; all parts of the system receive a clock with the same frequency as the System RC Oscillator.

5.3.2 Fetching of Initial Instructions

After reset has been released, the Cortex M4 CPU starts fetching PC and SP values from the reset address, which is 0x00000000. Refer to the ARM Architecture Reference Manual for more information on CPU startup. This address points to the first address in the internal Flash.

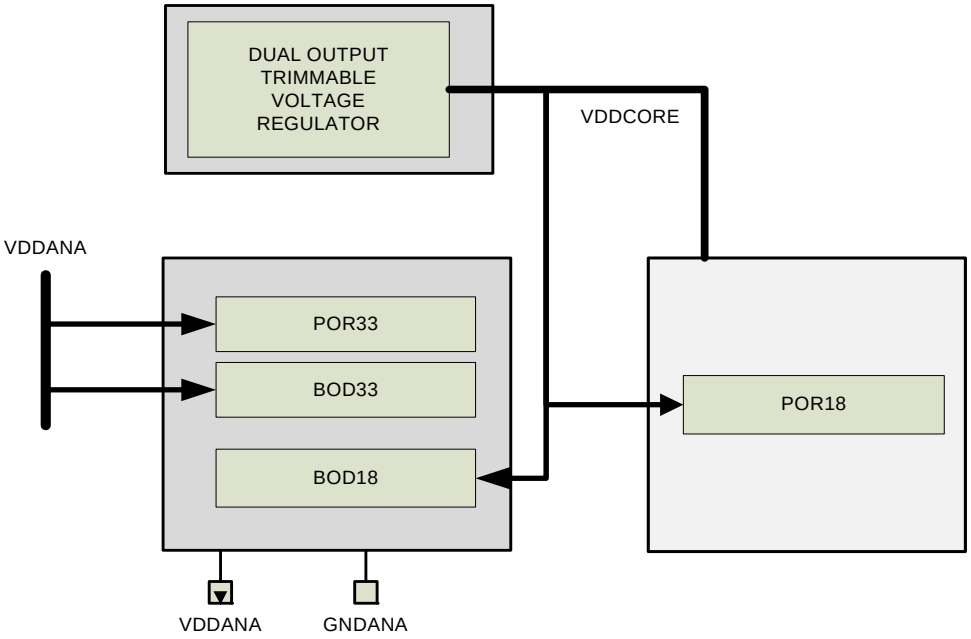
The code read from the internal flash is free to configure the clock system and clock sources.

5.4 Power-on-Reset, Brownout and Supply Monitor

The SAM4L embeds four features to monitor, warm, and/or reset the device:

- POR33: Power-on-Reset on VDDANA
- BOD33: Brownout detector on VDDANA
- POR18: Power-on-Reset on VDDCORE
- BOD18: Brownout detector on VDDCORE

Figure 5-6. Supply Monitor Schematic



5.4.1 Power-on-Reset on VDDANA

POR33 monitors VDDANA. It is always activated and monitors voltage at startup but also during all the Power Save Mode. If VDDANA goes below the threshold voltage, the entire chip is reset.

5.4.2 Brownout Detector on VDDANA

BOD33 monitors VDDANA. Refer to Section 12. ”Backup System Control Interface (BSCIF)” on page 161to get more details.

5.4.3 Power-on-Reset on VDDCORE

POR18 monitors the internal VDDCORE. Refer to Section 12. ”Backup System Control Interface (BSCIF)” on page 161 to get more details.

5.4.4 Brownout Detector on VDDCORE

Once the device is startup, the BOD18 monitors the internal VDDCORE. Refer to Section 12. ”Backup System Control Interface (BSCIF)” on page 161 to get more details.

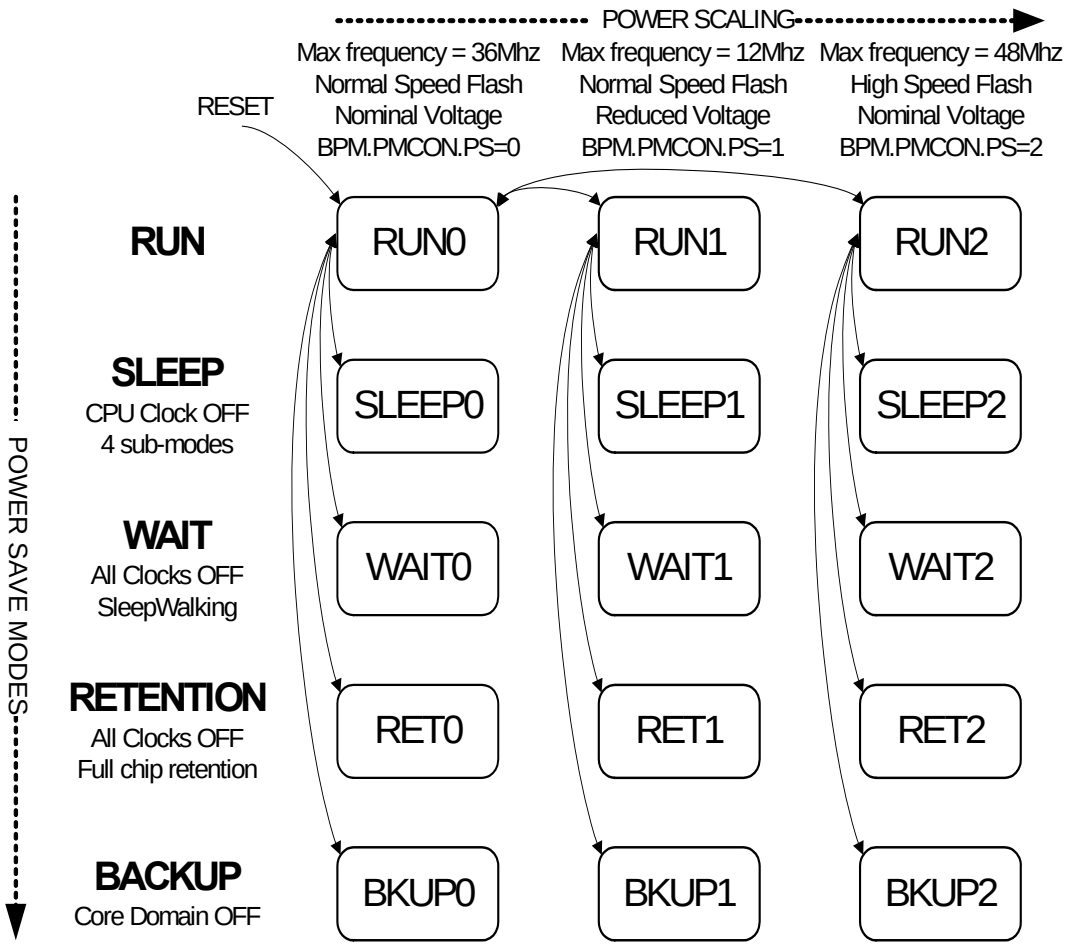
6. Low Power Techniques

The ATSAM4L8/L4/L2 supports multiple power configurations to allow the user to optimize its power consumption in different use cases. The Backup Power Manager (BPM) implements different solutions to reduce the power consumption:

- The Power Save modes intended to reduce the logic activity and to adapt the power configuration. See "Power Save Modes" on page 52.
- The Power Scaling intended to scale the power configuration (voltage scaling of the regulator). See "Power Scaling" on page 57.

These two techniques can be combined together.

Figure 6-1. Power Scaling and Power Save Mode Overview



6.1 Power Save Modes

Refer to Section 5. "Power and Startup Considerations" on page 43 to get definition of the core and the backup domains.

At power-up or after a reset, the ATSAM4L8/L4/L2 is in the RUN0 mode. Only the necessary clocks are enabled allowing software execution. The Power Manager (PM) can be used to adjust the clock frequencies and to enable and disable the peripheral clocks.

When the CPU is entering a Power Save Mode, the CPU stops executing code. The user can choose between four Power Save Modes to optimize power consumption:

- SLEEP mode: the Cortex-M4 core is stopped, optionally some clocks are stopped, peripherals are kept running if enabled by the user.
- WAIT mode: all clock sources are stopped, the core and all the peripherals are stopped except the modules running with the 32kHz clock if enabled. This is the lowest power configuration where SleepWalking is supported.
- RETENTION mode: similar to the WAIT mode in terms of clock activity. This is the lowest power configuration where the logic is retained.
- BACKUP mode: the Core domain is powered off, the Backup domain is kept powered.

A wake up source exits the system to the RUN mode from which the Power Save Mode was entered.

A reset source always exits the system from the Power Save Mode to the RUN0 mode.

The configuration of the I/O lines are maintained in all Power Save Modes. Refer to Section 11. ”Backup Power Manager (BPM)” on page 140.

6.1.1 SLEEP mode

The SLEEP mode allows power optimization with the fastest wake up time.

The CPU is stopped. To further reduce power consumption, the user can switch off modules-clocks and synchronous clock sources through the BPM.PMCON.SLEEP field (See Table 6-1). The required modules will be halted regardless of the bit settings of the mask registers in the Power Manager (PM.AHBMASK, PM.APBxMASK).

Table 6-1. SLEEP mode Configuration

BPM.PSAVE.SLEEP	CPU clock	AHB clocks	APB clocks GCLK	Clock sources: OSC, RCFAST, RC80M, PLL, DFLL	RCSYS	OSC32K RC32K(2)	Wake up Sources
0	Stop	Run	Run	Run	Run	Run	Any interrupt
1	Stop	Stop	Run	Run	Run	Run	Any interrupt(1)
2	Stop	Stop	Stop	Run	Run	Run	Any interrupt(1)
3	Stop	Stop	Stop	Stop	Run	Run	Any interrupt(1)

Notes: 1. from modules with clock running.
2. OSC32K and RC32K will only remain operational if pre-enabled.

6.1.1.1 Entering SLEEP mode

The SLEEP mode is entered by executing the WFI instruction.

Additionally, if the SLEEPONEXIT bit in the Cortex-M4 System Control Register (SCR) is set, the SLEEP mode will also be entered when the Cortex-M4 exits the lowest priority ISR. This

mechanism can be useful for applications that only require the processor to run when an interrupt occurs.

Before entering the SLEEP mode, the user must configure:

- the SLEEP mode configuration field (BPM.PMCON.SLEEP), Refer to Table 6-1.
- the SCR.SLEEPDEEP bit to 0. (See the Power Management section in the ARM Cortex-M4 Processor chapter).
- the BPM.PMCON.RET bit to 0.
- the BPM.PMCON.BKUP bit to 0.

6.1.1.2 Exiting SLEEP mode

The NVIC wakes the system up when it detects any non-masked interrupt with sufficient priority to cause exception entry. The system goes back to the RUN mode from which the SLEEP mode was entered. The CPU and affected modules are restarted. Note that even if an interrupt is enabled in SLEEP mode, it will not trigger if the source module is not clocked.

6.1.2 WAIT Mode and RETENTION Mode

The WAIT and RETENTION modes allow achieving very low power consumption while maintaining the Core domain powered-on. Internal SRAM and registers contents of the Core domain are preserved.

In these modes, all clocks are stopped except the 32kHz clocks (OSC32K, RC32K) which are kept running if enabled.

In RETENTION mode, the SleepWalking feature is not supported and must not be used.

6.1.2.1 Entering WAIT or RETENTION Mode

The WAIT or RETENTION modes are entered by executing the WFI instruction with the following settings:

- set the SCR.SLEEPDEEP bit to 1. (See the Power Management section in the ARM Cortex-M4 Processor chapter).
- set the BPM.PSAVE.BKUP bit to 0.
- set the BPM.PMCON.RET bit to RETENTION or WAIT mode.

SLEEPONEXIT feature is also available. See ”Entering SLEEP mode” on page 53.

6.1.2.2 Exiting WAIT or RETENTION Mode

In WAIT or RETENTION modes, synchronous clocks are stopped preventing interrupt sources from triggering. To wakeup the system, asynchronous wake up sources (AST, EIC, USBC ...) should be enabled in the peripheral (refer to the documentation of the peripheral). The PM.AWEN (Asynchronous Wake Up Enable) register should also be enabled for all peripheral except for EIC and AST.

When the enabled asynchronous wake up event occurs and the system is waken-up, it will generate either:

- an interrupt on the PM WAKE interrupt line if enabled (Refer to Section 10. ”Power Manager (PM)” on page 109). In that case, the PM.WCAUSE register indicates the wakeup source.
- or an interrupt directly from the peripheral if enabled (Refer to the section of the peripheral).

When waking up, the system goes back to the RUN mode mode from which the WAIT or RETENTION mode was entered.

6.1.3 BACKUP Mode

The BACKUP mode allows achieving the lowest power consumption possible in a system which is performing periodic wake-ups to perform tasks but not requiring fast startup time.

The Core domain is powered-off. The internal SRAM and register contents of the Core domain are lost. The Backup domain is kept powered-on. The 32kHz clock (RC32K or OSC32K) is kept running if enabled to feed modules that require clocking.

In BACKUP mode, the configuration of the I/O lines is preserved. Refer to Section 11. ”Backup Power Manager (BPM)” on page 140 to have more details.

6.1.3.1 Entering BACKUP Mode

The Backup mode is entered by using the WFI instruction with the following settings:

- set the SCR.SLEEPDEEP bit to 1. (See the Power Management section in the ARM Cortex-M4 Processor chapter).
- set the BPM.PSAVE.BKUP bit to 1.

6.1.3.2 Exiting BACKUP Mode

Exit from BACKUP mode happens if a reset occurs or if an enabled wake up event occurs.

The reset sources are:

- BOD33 reset
- BOD18 reset
- WDT reset
- External reset in RESET_N pin

The wake up sources are:

- EIC lines (level transition only)
- BOD33 interrupt
- BOD18 interrupt
- AST alarm, periodic, overflow
- WDT interrupt

The RC32K or OSC32K should be used as clock source for modules if required. The PMCON.CK32S is used to select one of these two 32kHz clock sources.

Exiting the BACKUP mode is triggered by:

- a reset source: an internal reset sequence is performed according to the reset source. Once VDDCORE is stable and has the correct value according to RUN0 mode, the internal reset is released and program execution starts. The corresponding reset source is flagged in the Reset Cause register (RCAUSE) of the PM.
- a wake up source: the Backup domain is not reset. An internal reset is generated to the Core domain, and the system switches back to the previous RUN mode. Once VDDCORE is stable and has the correct value, the internal reset in the Core domain is released and program execution starts. The BKUP bit is set in the Reset Cause register (RCAUSE) of the PM. It allows the user to discriminate between the reset cause and a wake up cause from the BACKUP mode. The wake up cause can be found in the Backup Wake up Cause register (BPM.BKUPWCAUSE).

6.1.4 Wakeup Time

6.1.4.1 Wakeup Time From SLEEP Mode

The latency depends on the clock sources wake up time. If the clock sources are not stopped, there is no latency to wake the clocks up.

6.1.4.2 Wakeup Time From WAIT or RETENTION Mode

The wake up latency consists of:

- the switching time from the low power configuration to the RUN mode power configuration. By default, the switching time is completed when all the voltage regulation system is ready. To speed-up the startup time, the user can set the Fast Wakeup bit in BPM.PMCON register.
- the wake up time of the RC oscillator used to start the system up. By default, the RCSYS oscillator is used to startup the system. The user can use another clock source (RCFAST for example) to speed up the startup time by configuring the PM.FASTWKUP register. Refer to Section 10. ”Power Manager (PM)” on page 109.
- the Flash memory wake up time.

To have the shortest wakeup time, the user should:

- set the BPM.PMCON.FASTWKUP bit.

- configure the PM.FASTSLEEP.FASTRCOSC field to use the RCFAST main clock.

- enter the WAIT or RETENTION mode

Upon a wakeup, this is required to keep the main clock connected to RCFAST until the voltage regulation system is fully ready (when BPM.ISR.PSOK bit is one). During this wakeup period, the FLASHCALW module is automatically configured to operate in “1 wait state mode”.

6.1.4.3 Wake time from BACKUP mode

It is equal to the Core domain logic reset latency (similar to the reset latency caused by an external reset in RESET_N pin) added to the time required for the voltage regulation system to be stabilized.

6.1.5 Power Save Mode Summary Table

The following table shows a summary of the main Power Save modes:

Table 6-2. Power Save mode Configuration Summary

Mode	Mode Entry	Wake up sources	Core domain	Backup domain
SLEEP	WFI SCR.SLEEPDEEP bit = 0 BPM.PMCON.BKUP bit = 0	Any interrupt	CPU clock OFF Other clocks OFF depending on the BPM.PMCON.SLEEP field see "SLEEP mode" on page 53	Clocks OFF depending on the BPM.PMCON.SLEEP field see "SLEEP mode" on page 53
WAIT	WFI SCR.SLEEPDEEP bit = 1 BPM.PMCON.RET bit = 0 BPM.PMCON.BKUP bit = 0	PM WAKE interrupt	All clocks are OFF Core domain is retained	All clocks are OFF except RC32K or OSC32K if running
RETENTION	WFI SCR.SLEEPDEEP bit = 1 BPM.PMCON.RET bit = 1 BPM.PMCON.BKUP bit = 0	PM WAKE interrupt	All clocks are OFF Core domain is retained	All clocks are OFF except RC32K or OSC32K if running
BACKUP	WFI + SCR.SLEEPDEEP bit = 1 + BPM.PMCON.BKUP bit = 1	EIC interrupt BOD33, BOD18 interrupt and reset AST alarm, periodic, overflow WDT interrupt and reset external reset on RESET_N pin	OFF (not powered)	All clocks are OFF except RC32K or OSC32K if running

6.2 Power Scaling

The Power Scaling technique consists of adjusting the internal regulator output voltage (voltage scaling) to reduce the power consumption. According to the requirements in terms of performance, operating modes, and current consumption, the user can select the Power Scaling configuration that fits the best with its application.

The Power Scaling configuration field (PMCON.PS) is provided in the Backup Power Manager (BPM) module.

In RUN mode, the user can adjust on the fly the Power Scaling configuration

The Figure 6.1 summarizes the different combination of the Power Scaling configuration which can be applied according to the Power Save Mode.

Power scaling from a current power configuration to a new power configuration is done by halting the CPU execution

Power scaling occurs after a WFI instruction. The system is halted until the new power configuration is stabilized. After handling the PM interrupt, the system resumes from WFI.

To scale the power, the following sequence is required:

- Check the BPM.SR.PSOK bit to make sure the current power configuration is stabilized.

- Set the clock frequency to be supported in both power configurations.
- Set the high speed read mode of the FLASH to be supported in both power scaling configurations
  - Only relevant when entering or exiting BPM.PMCON.PS=2
- Configure the BPM.PMCON.PS field to the new power configuration.
- Set the BPM.PMCON.PSCREQ bit to one.
- Disable all the interrupts except the PM WCAUSE interrupt and enable only the PSOK asynchronous event in the AWEN register of PM.
- Execute the WFI instruction.
- WAIT for PM interrupt.

The new power configuration is reached when the system is waken up by the PM interrupt thanks to the PSOK event.

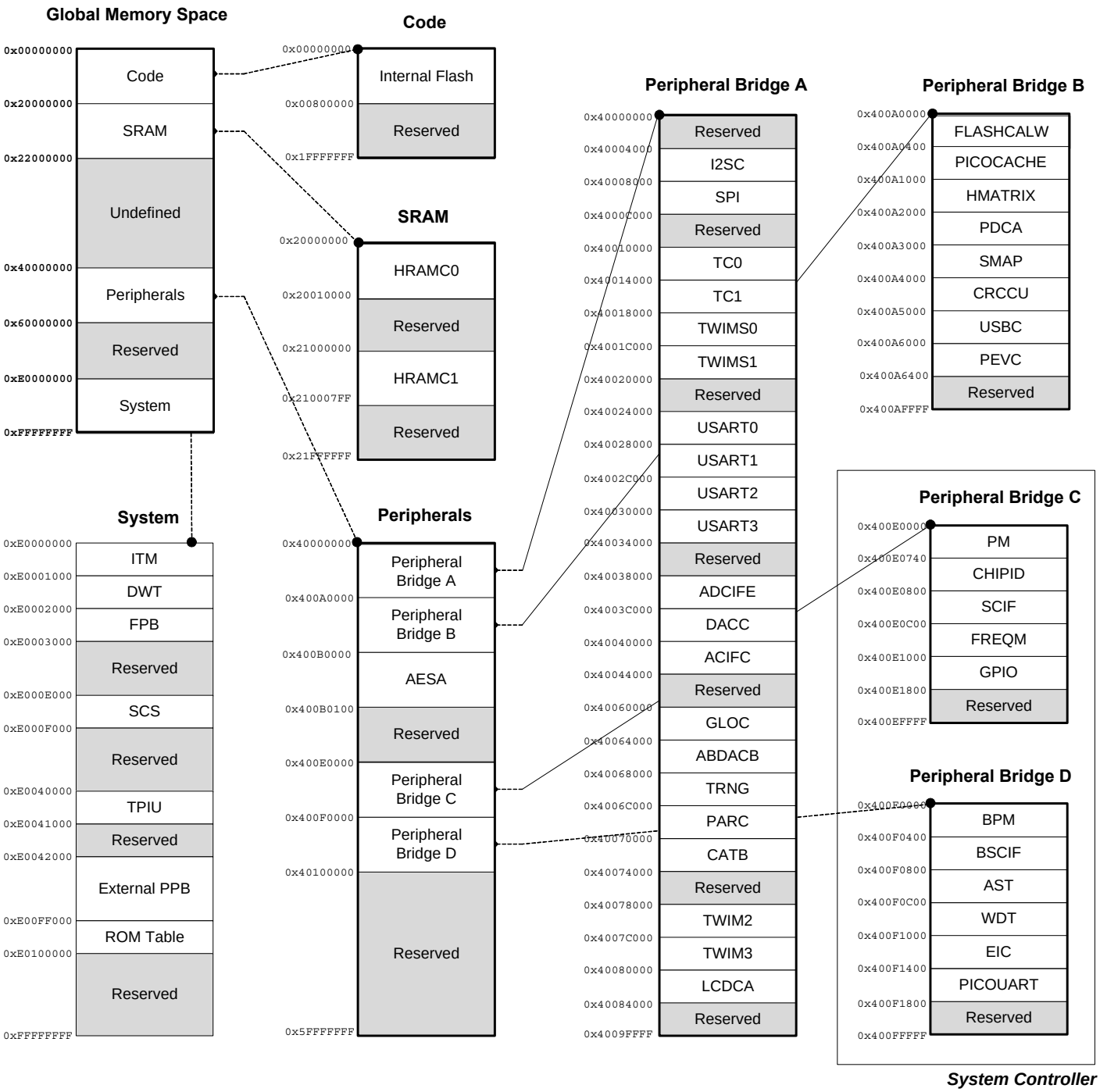
By default, all features are available in all Power Scaling modes. However some specific features are not available in PS1 (BPM.PMCON.PS=1) mode :

- USB
- DFLL
- PLL
- Programming/Erasing in Flash

7. Memories

7.1 Product Mapping

Figure 7-1. ATSAM4L8/L4/L2 Product Mapping



7.2 Embedded Memories

- **Internal high-speed flash**
  - **512Kbytes (ATSAM4Lx8)**
  - **256Kbytes (ATSAM4Lx4)**
  - **128Kbytes (ATSAM4Lx2)**
    - **Pipelined flash architecture, allowing burst reads from sequential flash locations, hiding penalty of 1 wait state access**
    - **Pipelined flash architecture typically reduces the cycle penalty of 1 wait state operation compared to 0 wait state operation**
    - **100 000 write cycles, 15-year data retention capability**
    - **Sector lock capabilities, bootloader protection, security bit**
    - **32 fuses, erased during chip erase**
    - **User page for data to be preserved during chip erase**
- **Internal high-speed SRAM, single-cycle access at full speed**
  - **64Kbytes (ATSAM4Lx8)**
  - **32Kbytes (ATSAM4Lx4, ATSAM4Lx2)**

7.3 Physical Memory Map

The system bus is implemented as a bus matrix. All system bus addresses are fixed, and they are never remapped in any way, not even during boot. The 32-bit physical address space is mapped as follows:

Table 7-1. ATSAM4L8/L4/L2 Physical Memory Map

Memory	Start Address	Size	Size
		ATSAM4Lx4	ATSAM4Lx2
Embedded Flash	0x00000000	256Kbytes	128Kbytes
Embedded SRAM	0x20000000	32Kbytes	32Kbytes
Cache SRAM	0x21000000	4Kbytes	4Kbytes
Peripheral Bridge A	0x40000000	64Kbytes	64Kbytes
Peripheral Bridge B	0x400A0000	64Kbytes	64Kbytes
AESA	0x400B0000	256 bytes	256 bytes
Peripheral Bridge C	0x400E0000	64Kbytes	64Kbytes
Peripheral Bridge D	0x400F0000	64Kbytes	64Kbytes

Memory	Start Address	Size
		ATSAM4Lx8
Embedded Flash	0x00000000	512Kbytes
Embedded SRAM	0x20000000	64Kbytes
Cache SRAM	0x21000000	4Kbytes
Peripheral Bridge A	0x40000000	64Kbytes
Peripheral Bridge B	0x400A0000	64Kbytes

Memory	Start Address	Size
		ATSAM4Lx8
AESA	0x400B0000	256 bytes
Peripheral Bridge C	0x400E0000	64 Kbytes
Peripheral Bridge D	0x400F0000	64 Kbytes

Table 7-2. Flash Memory Parameters

Device	Flash Size (*FLASH_PW*)	Number of Pages (*FLASH_P*)	Page Size (*FLASH_W*)
ATSAM4Lx8	512 Kbytes	1024	512 bytes
ATSAM4Lx4	256 Kbytes	512	512 bytes
ATSAM4Lx2	128 Kbytes	256	512 bytes

8. Debug and Test

8.1 Features

- **IEEE1149.1 compliant JTAG Debug Port**
- **Serial Wire Debug Port**
- **Boundary-Scan chain on all digital pins for board-level testing**
- **Direct memory access and programming capabilities through debug ports**
- **Flash Patch and Breakpoint (FPB) unit for implementing breakpoints and code patches**
- **Data Watchpoint and Trace (DWT) unit for implementing watchpoints, data tracing, and system profiling**
- **Instrumentation Trace Macrocell (ITM) for support of printf style debugging**
- **Chip Erase command and status**
- **Unlimited Flash User page read access**
- **Cortex-M4 core reset source**
- **CRC32 of any memory accessible through the bus matrix**
- **Debugger Hot Plugging**

8.2 Overview

Debug and test features are made available to external tools by:

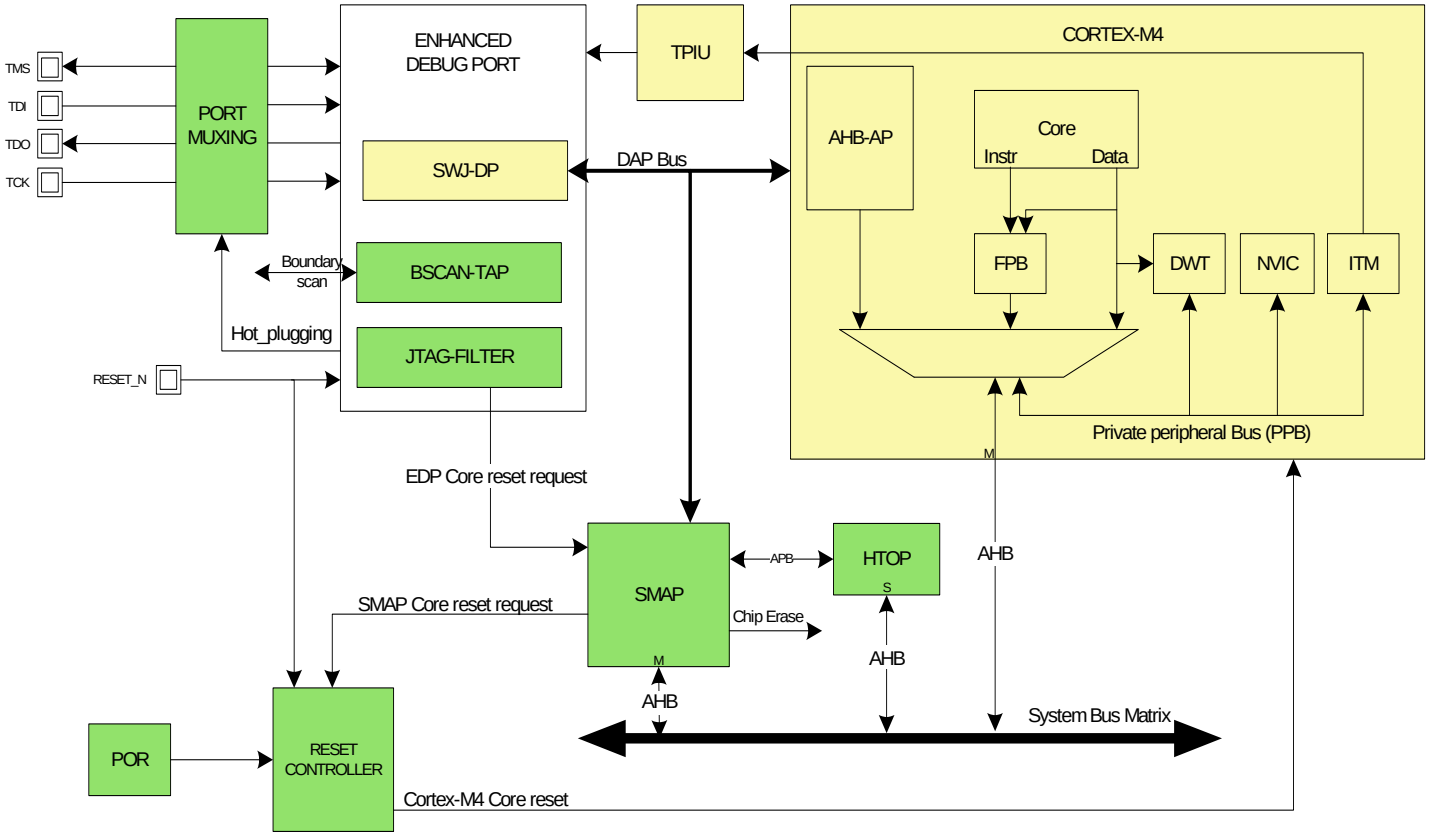
- The Enhanced Debug Port (EDP) embedding:
  - a Serial Wire Debug Port (SW-DP) part of the ARM coresight architecture
  - an IEEE 1149.1 JTAG Debug Debug Port (JTAG-DP) part of the ARM coresight architecture
  - a supplementary IEEE 1149.1 JTAG TAP machine that implements the boundary scan feature
- The System Manager Acces Port (SMAP) providing unlimited flash User page read access, CRC32 of any memory accessible through the bus matrix and Cortex-M4 core reset services
- The AHB Access Port (AHB-AP) providing Direct memory access, programming capabilities and standard debugging functions
- The Instrumentation Trace macrocell part of the ARM coresight architecture

For more information on ARM debug components, refer to:

- ARMv7-M Architecture Reference Manual
- ARM Debug Interface v5.1 Architecture Specification document
- ARM CoreSight Architecture Specification
- ARM ETM Architecture Specification v3.5
- ARM Cortex-M4 Technical Reference Manual

8.3 Block Diagram

Figure 8-1. Debug and Test Block Diagram



8.4 I/O Lines Description

Refer to Section 8.7.4 “I/O Lines Description” on page 68.

8.5 Product Dependencies

8.5.1 I/O Lines

Refer to Section 8.7.5.1 “I/O Lines” on page 69.

8.5.2 Power Management

Refer to Section 8.7.5.2 “Power Management” on page 69.

8.5.3 Clocks

Refer to Section 8.7.5.3 “Clocks” on page 69.

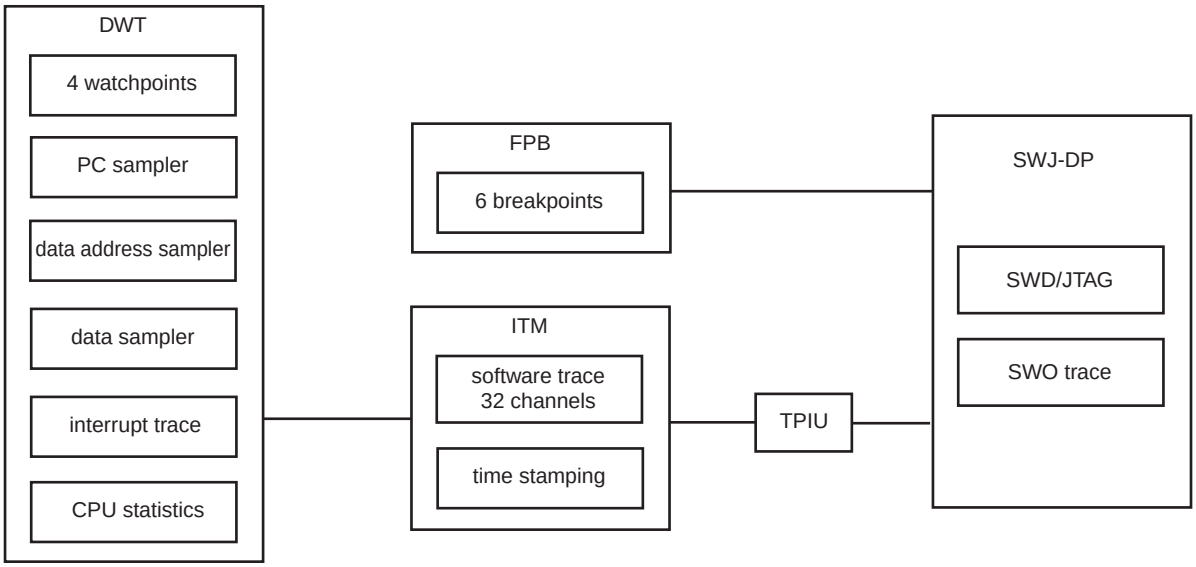
8.6 Core Debug

Figure 8-2 shows the Debug Architecture used in the SAM4L. The Cortex-M4 embeds four functional units for debug:

- FPB (Flash Patch Breakpoint)
- DWT (Data Watchpoint and Trace)
- ITM (Instrumentation Trace Macrocell)
- TPIU (Trace Port Interface Unit)

The debug architecture information that follows is mainly dedicated to developers of SWJ-DP Emulators/Probes and debugging tool vendors for Cortex-M4 based microcontrollers. For further details on SWJ-DP see the Cortex-M4 technical reference manual.

Figure 8-2. Debug Architecture



8.6.1 FPB (Flash Patch Breakpoint)

The FPB:

- Implements hardware breakpoints
- Patches (on the fly) code and data being fetched by the Cortex-M4 core from code space with data in the system space. Definition of code and system spaces can be found in the System Address Map section of the ARMv7-M Architecture Reference Manual.

The FPB unit contains:

- Two literal comparators for matching against literal loads from Code space, and remapping to a corresponding area in System space.
- Six instruction comparators for matching against instruction fetches from Code space and remapping to a corresponding area in System space.
- Alternatively, comparators can also be configured to generate a Breakpoint instruction to the processor core on a match.

8.6.2 DWT (Data Watchpoint and Trace)

The DWT contains four comparators which can be configured to generate the following:

- PC sampling packets at set intervals
- PC or Data watchpoint packets
- Watchpoint event to halt core

The DWT contains counters for the items that follow:

- Clock cycle (CYCCNT)
- Folded instructions
- Load Store Unit (LSU) operations
- Sleep Cycles
- CPI (all instruction cycles except for the first cycle)
- Interrupt overhead

8.6.3 ITM (Instrumentation Trace Macrocell)

The ITM is an application driven trace source that supports printf style debugging to trace Operating System (OS) and application events, and emits diagnostic system information. The ITM emits trace information as packets which can be generated by three different sources with several priority levels:

- **Software trace**: This can be done thanks to the printf style debugging. For more information, refer to Section "How to Configure the ITM:".
- **Hardware trace**: The ITM emits packets generated by the DWT.
- **Time stamping**: Timestamps are emitted relative to packets. The ITM contains a 21-bit counter to generate the timestamp.

How to Configure the ITM:

The following example describes how to output trace data in asynchronous trace mode.

- Configure the TPIU for asynchronous trace mode (refer to Section "5.4.3. How to Configure the TPIU")
- Enable the write accesses into the ITM registers by writing "0xC5ACCE55" into the Lock Access Register (Address: 0xE0000FB0)
- Write 0x00010015 into the Trace Control Register:
  - Enable ITM
  - Enable Synchronization packets
  - Enable SWO behavior

  - Fix the ATB ID to 1
- Write 0x1 into the Trace Enable Register:
  - Enable the Stimulus port 0
- Write 0x1 into the Trace Privilege Register:
  - Stimulus port 0 only accessed in privileged mode (Clearing a bit in this register will result in the corresponding stimulus port being accessible in user mode.)
- Write into the Stimulus port 0 register: TPIU (Trace Port Interface Unit)

  The TPIU acts as a bridge between the on-chip trace data and the Instruction Trace Macrocell (ITM).

  The TPIU formats and transmits trace data off-chip at frequencies asynchronous to the core.

Asynchronous Mode:

The TPIU is configured in asynchronous mode, trace data are output using the single TRACESWO pin. The TRACESWO signal is multiplexed with the TDO signal of the JTAG Debug Port. As a consequence, asynchronous trace mode is only available when the Serial Wire Debug mode is selected since TDO signal is used in JTAG debug mode.

Two encoding formats are available for the single pin output:

- Manchester encoded stream. This is the reset value.
- NRZ_based UART byte structure

5.4.3. How to Configure the TPIU

This example only concerns the asynchronous trace mode.

- Set the TRCENA bit to 1 into the Debug Exception and Monitor Register (0xE000EDFC) to enable the use of trace and debug blocks.
- Write 0x2 into the Selected Pin Protocol Register
  - Select the Serial Wire Output – NRZ
- Write 0x100 into the Formatter and Flush Control Register
- Set the suitable clock prescaler value into the Async Clock Prescaler Register to scale the baud rate of the asynchronous output (this can be done automatically by the debugging tool).

8.7 Enhanced Debug Port (EDP)

Rev.: 1.0.0.0

8.7.1 Features

- **IEEE1149.1 compliant JTAG debug port**
- **Serial Wire Debug Port**
- **Boundary-Scan chain on all digital pins for board-level testing**
- **Debugger Hot-Plugging**
- **SMAP core reset request source**

8.7.2 Overview

The enhanced debug port embeds a standard ARM debug port plus some specific hardware intended for testability and activation of the debug port features. All the information related to the ARM Debug Interface implementation can be found in the ARM Debug Interface v5.1 Architecture Specification document.

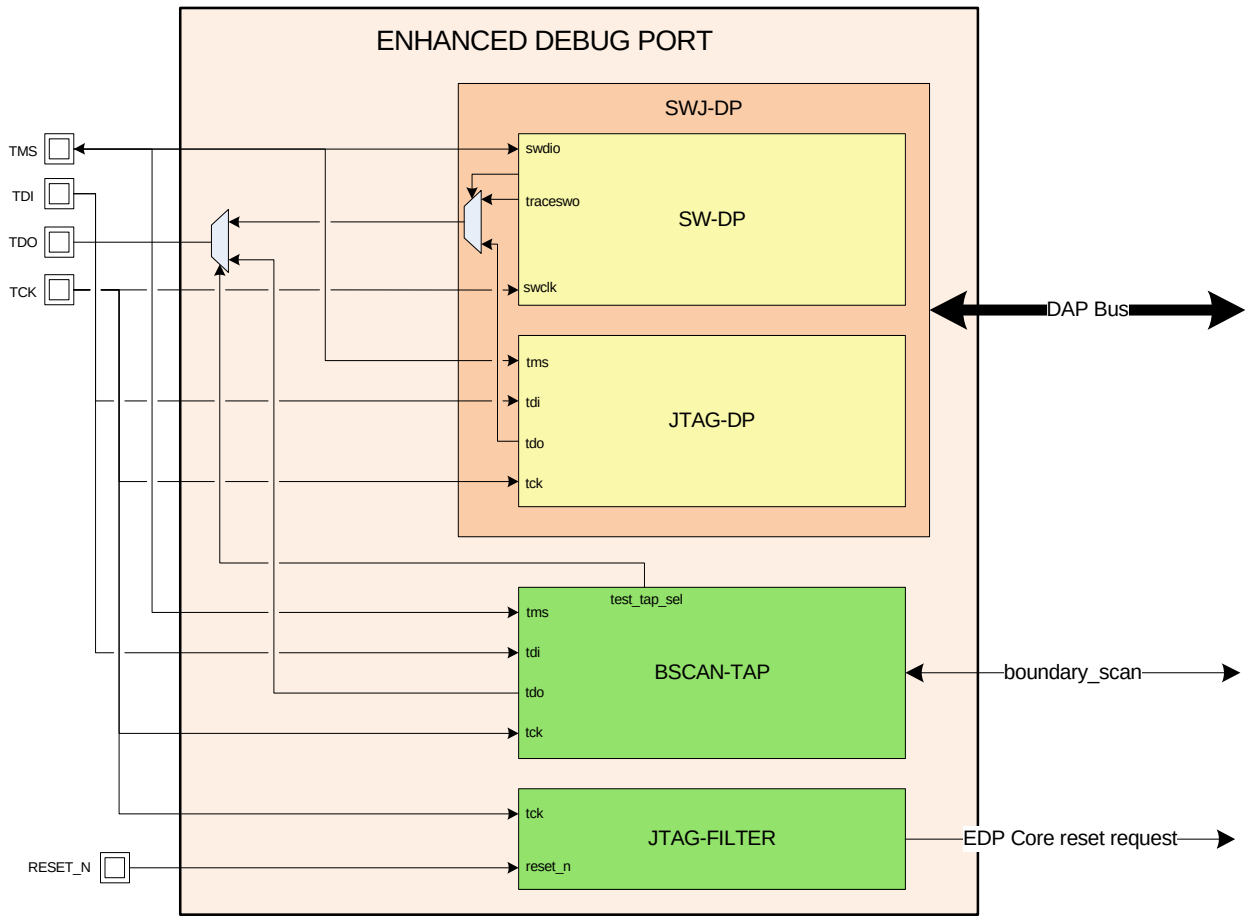
It features:

- A single Debug Port (SWJ-DP), that provides the external physical connection to the interface and supports two DP implementations:
  - the JTAG Debug Port (JTAG-DP)
  - the Serial Wire Debug Port (SW-DP)
- A supplementary JTAG TAP (BSCAN-TAP) connected in parallel with the JTAG-DP that implements the boundary scan instructions detailed in
- A JTAG-FILTER module that monitors TCK and RESET_N pins to handle specific features like the detection of a debugger hot-plugging and the request of reset of the Cortex-M4 at startup.

The JTAG-FILTER module detects the presence of a debugger. When present, JTAG pins are automatically assigned to the Enhanced Debug Port(EDP). If the SWJ-DP is switched to the SW mode, then TDI and TDO alternate functions are released. The JTAG-FILTER also implements a CPU halt mechanism. When triggered, the Cortex-M4 is maintained under reset after the external reset is released to prevent any system corruption during later programmation operations.

8.7.3 Block Diagram

Figure 8-3. Enhanced Debug Port Block Diagram



8.7.4 I/O Lines Description

Table 8-1. I/O Lines Description

Name	JTAG Debug Port		SWD Debug Port	
	Type	Description	Type	Description
TCK/SWCLK	I	Debug Clock	I	Serial Wire Clock
TDI	I	Debug Data in	-	NA
TDO/TRACESWO	O	Debug Data Out	O	Trace asynchronous Data Out
TMS/SWDIO	I	Debug Mode Select	I/O	Serial Wire Input/Output
RESET_N	I	Reset	I	Reset

8.7.5 Product Dependencies

8.7.5.1 I/O Lines

The TCK pin is dedicated to the EDP. The other debug port pins default after reset to their GPIO functionality and are automatically reassigned to the JTAG functionalities on detection of a debugger. In serial wire mode, TDI and TDO can be used as GPIO functions. Note that in serial wire mode TDO can be used as a single pin trace output.

8.7.5.2 Power Management

When a debugger is present, the connection is kept alive allowing debug operations. As a side effect, the power is never turned off. The hot plugging functionality is always available except when the system is in BACKUP Power Save Mode.

8.7.5.3 Clocks

The SWJ-DP uses the external TCK pin as its clock source. This clock must be provided by the external JTAG master device.

Some of the JTAG Instructions are used to access an Access Port (SMAP or AHB-AP). These instructions require the CPU clock to be running.

If the CPU clock is not present because the CPU is in a Power Save Mode where this clock is not provided, the Power Manager(PM) will automatically restore the CPU clock on detection of a debug access.

The RCSYS clock is used as CPU clock when the external reset is applied to ensure correct Access Port operations.

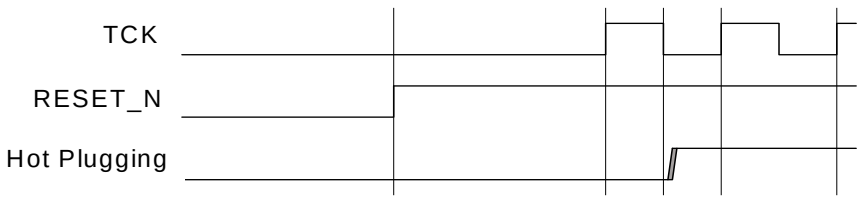
8.7.6 Module Initialization

This module is enabled as soon as a TCK falling edge is detected when RESET_N is not asserted (refer to Section 8.7.7 below). Moreover, the module is synchronously reseted as long as the TAP machine is in the TEST_LOGIC_RESET (TLR) state. It is advised asserting TMS at least 5 TCK clock periods after the debugger has been detected to ensure the module is in the TLR state prior to any operation. This module also has the ability to maintain the Cortex-M4 under reset (refer to the Section 8.7.8 “SMAP Core Reset Request Source” on page 70).

8.7.7 Debugger Hot Plugging

The TCK pin is dedicated to the EDP. After reset has been released, the EDP detects that a debugger has been attached when a TCK falling edge arises.

Figure 8-4. Debugger Hot Plugging Detection Timings Diagram

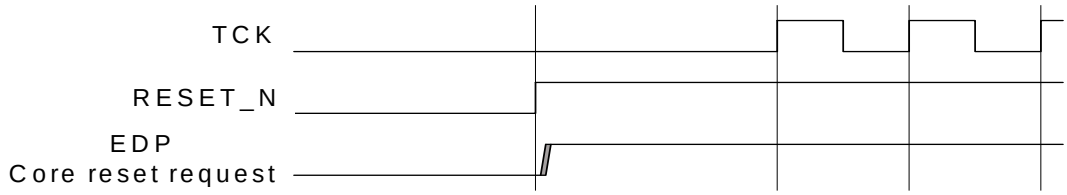


The Debug Port pins assignation is then forced to the EDP function even if they were already assigned to another module. This allows to connect a debugger at any time without reseting the device. The connection is non-intrusive meaning that the chip will continue its execution without being disturbed. The CPU can of course be halted later on by issuing Cortex-M4 OCD features.

8.7.8 SMAP Core Reset Request Source

The EDP has the ability to send a request to the SMAP for a Cortex-M4 Core reset. The procedure to do so is to hold TCK low until RESET_N is released. This mechanism aims at halting the CPU to prevent it from changing the system configuration while the SMAP is operating.

Figure 8-5. SMAP Core Reset Request Timings Diagram



The SMAP can de-assert the core reset request for this operation, refer to Section 8.8.8 “Cortex-M4 Core Reset Source” on page 78.

8.7.9 SWJ-DP

The Cortex-M4 embeds a SWJ-DP Debug port which is the standard CoreSight™ debug port. It combines Serial Wire Debug Port (SW-DP), from 2 to 3 pins and JTAG debug Port(JTAG-DP), 5 pins.

By default, the JTAG Debug Port is active. If the host debugger wants to switch to the Serial Wire Debug Port, it must provide a dedicated JTAG sequence on TMS/SWDIO and TCK/SWCLK which disables JTAG-DP and enables SW-DP.

When the EDP has been switched to Serial Wire mode, TDO/TRACESWO can be used for trace (for more information refer to the section below). The asynchronous TRACE output (TRACESWO) is multiplexed with TDO. So the asynchronous trace can only be used with SW-DP, not JTAG-DP.

The SWJ-DP provides access to the AHB-AP and SMAP access ports which have the following APSEL value:

Figure 8-6. Access Ports APSEL

Acces Port (AP)	APSEL
AHB-AP	0
SMAP	1

Refer to the ARM Debug Interface v5.1 Architecture Specification for more details on SWJ-DP.

8.7.10 SW-DP and JTAG-DP Selection Mechanism

After reset, the SWJ-DP is in JTAG mode but it can be switched to the Serial Wire mode. Debug port selection mechanism is done by sending specific **SWDIOTMS** sequence. The JTAG-DP is selected by default after reset.

- Switch from JTAG-DP to SW-DP. The sequence is:
  - Send more than 50 **SWCLKTCK** cycles with **SWDIOTMS** = 1
  - Send the 16-bit sequence on **SWDIOTMS** = 0111100111100111 (0x79E7 MSB first)
  - Send more than 50 **SWCLKTCK** cycles with **SWDIOTMS** = 1
- Switch from SWD to JTAG. The sequence is:
  - Send more than 50 **SWCLKTCK** cycles with **SWDIOTMS** = 1
  - Send the 16-bit sequence on **SWDIOTMS** = 0011110011100111 (0x3CE7 MSB first)

Send more than 50 **SWCLKTCK** cycles with **SWDIOTMS** = 1

Note that the BSCAN-TAP is not available when the debug port is switched to Serial Mode. Boundary scan instructions are not available.

8.7.11 JTAG-DP and BSCAN-TAP Selection Mechanism

After the DP has been enabled, the BSCAN-TAP and the JTAG-DP run simultaneously has long as the SWJ-DP remains in JTAG mode. Each TAP captures simultaneously the JTAG instructions that are shifted. If an instruction is recognized by the BSCAN-TAP, then the BSCAN-TAP TDO is selected instead of the SWJ-DP TDO. TDO selection changes dynamically depending on the current instruction held in the BSCAN-TAP instruction register.

8.7.12 JTAG Instructions Summary

The implemented JTAG instructions are shown in the table below.

Table 8-2. Implemented JTAG instructions list

IR instruction value	Instruction	Description	availability when protected	Component
b0000	EXTEST	Select boundary-scan chain as data register for testing circuitry external to the device.	yes	BSCAN-TAP
b0001	SAMPLE_PRELOAD	Take a snapshot of external pin values without affecting system operation.	yes	
b0100	INTEST	Select boundary-scan chain for internal testing of the device.	yes	
b0101	CLAMP	Bypass device through Bypass register, while driving outputs from boundary-scan register.	yes	
b1000	ABORT	ARM JTAG-DP Instruction	yes	SWJ-DP (in JTAG mode)
b1010	DPACC	ARM JTAG-DP Instruction	yes	
b1011	APACC	ARM JTAG-DP Instruction	yes	
b1100	-	Reserved	yes	
b1101	-	Reserved	yes	
b1110	IDCODE	ARM JTAG-DP Instruction	yes	
b1111	BYPASS	Bypass this device through the bypass register.	yes	

8.7.13 Security Restrictions

The SAM4L provide a security restrictions mechanism to lock access to the device. The device in the protected state when the Flash Security Bit is set. Refer to section Flash Controller for more details.

When the device is in the protected state the AHB-AP is locked. Full access to the AHB-AP is re-enabled when the protected state is released by issuing a Chip Erase command. Note that the protected state will read as programmed only after the system has been reseted.

8.7.13.1 Notation

Table 8-4 on page 73 shows bit patterns to be shifted in a format like "**p01**". Each character corresponds to one bit, and eight bits are grouped together for readability. The least significant bit is always shifted first, and the most significant bit shifted last. The symbols used are shown in Table 8-3.

Table 8-3. Symbol Description

Symbol	Description
0	Constant low value - always reads as zero.
1	Constant high value - always reads as one.
p	The chip protected state.
x	A don't care bit. Any value can be shifted in, and output data should be ignored.
e	An error bit. Read as one if an error occurred, or zero if not.
b	A busy bit. Read as one if the SMAP was busy, or zero if it was not.
s	Startup done bit. Read as one if the system has started-up correctly.

In many cases, it is not required to shift all bits through the data register. Bit patterns are shown using the full width of the shift register, but the suggested or required bits are emphasized using **bold** text. I.e. given the pattern "**01010101** xxxxxxxx xxxxxxxx xxxxxxxx", the shift register is 32 bits, but the test or debug unit may choose to shift only 8 bits "**01010101**".

The following describes how to interpret the fields in the instruction description tables:

Table 8-4. Instruction Description

Instruction	Description
IR input value	Shows the bit pattern to shift into IR in the Shift-IR state in order to select this instruction. The pattern is show both in binary and in hexadecimal form for convenience. Example: **1000** (0x8)
IR output value	Shows the bit pattern shifted out of IR in the Shift-IR state when this instruction is active. Example: p00s

Table 8-4. Instruction Description (Continued)

Instruction	Description
DR Size	Shows the number of bits in the data register chain when this instruction is active. Example: 32 bits
DR input value	Shows which bit pattern to shift into the data register in the Shift-DR state when this instruction is active.
DR output value	Shows the bit pattern shifted out of the data register in the Shift-DR state when this instruction is active.

8.7.14 JTAG Instructions

Refer to the ARM Debug Interface v5.1 Architecture Specification for more details on ABORT, DPACC, APACC and IDCODE instructions.

8.7.14.1 EXTEST

This instruction selects the boundary-scan chain as Data Register for testing circuitry external to the chip package. The contents of the latched outputs of the boundary-scan chain is driven out as soon as the JTAG IR-register is loaded with the EXTEST instruction.

Starting in Run-Test/Idle, the EXTEST instruction is accessed the following way:

1. Select the IR Scan path.
2. In Capture-IR: The IR output value is latched into the shift register.
3. In Shift-IR: The instruction register is shifted by the TCK input.
4. In Update-IR: The data from the boundary-scan chain is applied to the output pins.
5. Return to Run-Test/Idle.
6. Select the DR Scan path.
7. In Capture-DR: The data on the external pins is sampled into the boundary-scan chain.
8. In Shift-DR: The boundary-scan chain is shifted by the TCK input.
9. In Update-DR: The data from the scan chain is applied to the output pins.
10. Return to Run-Test/Idle.

Table 8-5. EXTEST Details

Instructions	Details
IR input value	**0000** (0x0)
IR output value	p00s
DR Size	Depending on boundary-scan chain, see BSDL-file.
DR input value	Depending on boundary-scan chain, see BSDL-file.
DR output value	Depending on boundary-scan chain, see BSDL-file.

8.7.14.2 SAMPLE_PRELOAD

This instruction takes a snap-shot of the input/output pins without affecting the system operation, and pre-loading the scan chain without updating the DR-latch. The boundary-scan chain is selected as Data Register.

Starting in Run-Test/Idle, the Device Identification register is accessed in the following way:

1. Select the IR Scan path.
2. In Capture-IR: The IR output value is latched into the shift register.
3. In Shift-IR: The instruction register is shifted by the TCK input.
4. Return to Run-Test/Idle.
5. Select the DR Scan path.
6. In Capture-DR: The Data on the external pins are sampled into the boundary-scan chain.
7. In Shift-DR: The boundary-scan chain is shifted by the TCK input.
8. Return to Run-Test/Idle.

Table 8-6. SAMPLE_PRELOAD Details

Instructions	Details
IR input value	**0001** (0x1)
IR output value	p00s
DR Size	Depending on boundary-scan chain, see BSDL-file.
DR input value	Depending on boundary-scan chain, see BSDL-file.
DR output value	Depending on boundary-scan chain, see BSDL-file.

8.7.14.3 INTEST

This instruction selects the boundary-scan chain as Data Register for testing internal logic in the device. The logic inputs are determined by the boundary-scan chain, and the logic outputs are captured by the boundary-scan chain. The device output pins are driven from the boundary-scan chain.

Starting in Run-Test/Idle, the INTEST instruction is accessed the following way:

1. Select the IR Scan path.
2. In Capture-IR: The IR output value is latched into the shift register.
3. In Shift-IR: The instruction register is shifted by the TCK input.
4. In Update-IR: The data from the boundary-scan chain is applied to the internal logic inputs.
5. Return to Run-Test/Idle.
6. Select the DR Scan path.
7. In Capture-DR: The data on the internal logic is sampled into the boundary-scan chain.
8. In Shift-DR: The boundary-scan chain is shifted by the TCK input.
9. In Update-DR: The data from the boundary-scan chain is applied to internal logic inputs.
10. Return to Run-Test/Idle.

Table 8-7. INTEST Details

Instructions	Details
IR input value	**0100** (0x4)
IR output value	p001
DR Size	Depending on boundary-scan chain, see BSDL-file.
DR input value	Depending on boundary-scan chain, see BSDL-file.
DR output value	Depending on boundary-scan chain, see BSDL-file.

8.7.14.4 CLAMP

This instruction selects the Bypass register as Data Register. The device output pins are driven from the boundary-scan chain.

Starting in Run-Test/Idle, the CLAMP instruction is accessed the following way:

1. Select the IR Scan path.
2. In Capture-IR: The IR output value is latched into the shift register.
3. In Shift-IR: The instruction register is shifted by the TCK input.
4. In Update-IR: The data from the boundary-scan chain is applied to the output pins.
5. Return to Run-Test/Idle.
6. Select the DR Scan path.
7. In Capture-DR: A logic ‘0’ is loaded into the Bypass Register.
8. In Shift-DR: Data is scanned from TDI to TDO through the Bypass register.
9. Return to Run-Test/Idle.

Table 8-8. CLAMP Details

Instructions	Details
IR input value	**0101** (0x5)
IR output value	p00s
DR Size	1
DR input value	x
DR output value	x

8.8 System Manager Access Port (SMAP)

Rev.: 1.0.0.0

8.8.1 Features

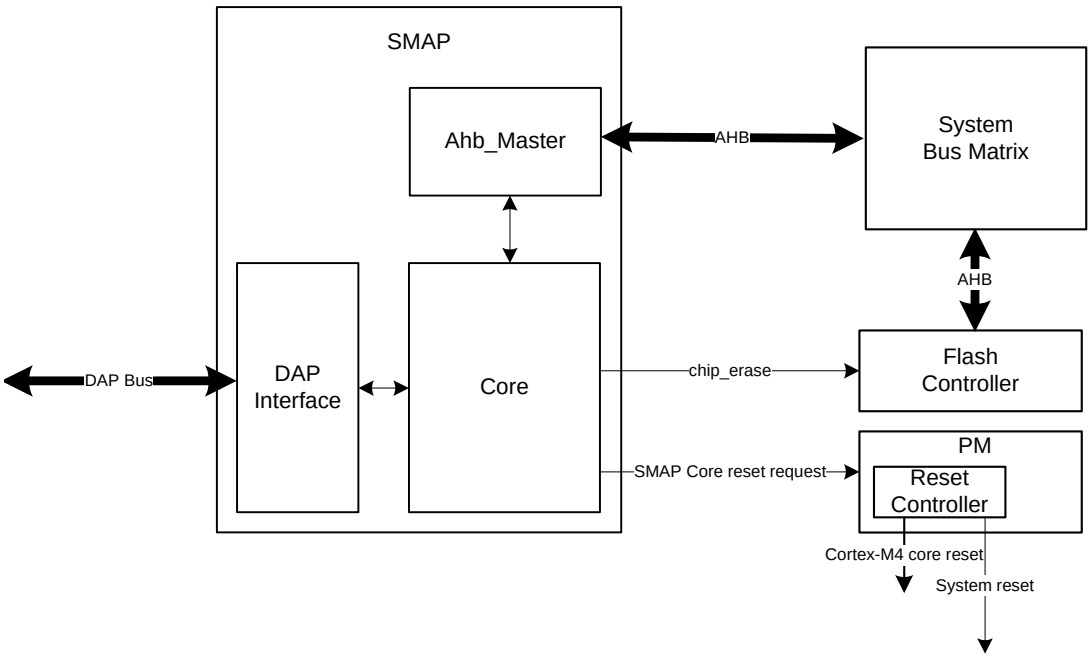
- **Chip Erase command and status**
- **Cortex-M4 core reset source**
- **32-bit Cyclic Redundancy check of any memory accessible through the bus matrix**
- **Unlimited Flash User page read access**
- **Chip identification register**

8.8.2 Overview

The SMAP provides memory-related services and also Cortex-M4 core reset control to a debugger through the Debug Port. This makes possible to halt the CPU and program the device after reset.

8.8.3 Block Diagram

Figure 8-7. SMAP Block Diagram



8.8.4 Initializing the Module

The SMAP can be accessed only if the CPU clock is running and the SWJ-DP has been activated by issuing a CDBGPWRUP request. For more details, refer to the ARM Debug Interface v5.1 Architecture Specification.

Then it must be enabled by writing a one to the EN bit of the CR register (CR.EN) before writing or reading other registers. If the SMAP is not enabled it will discard any read or write operation.

8.8.5 Stopping the Module

To stop the module, the user must write a one to the DIS bit of the CR register (CR.DIS). All the user interface and internal registers will be cleared and the internal clock will be stopped.

8.8.6 Security Considerations

In protected state this module may access sensible information located in the device memories. To avoid any risk of sensible data extraction from the module registers, all operations are non interruptible except by a disable command triggered by writing a one to CR.DIS. Issuing this command clears all the interface and internal registers.

Some registers have some special protection:

- It is not possible to read or write the LENGTH register when the part is protected.
- In addition, when the part is protected and an operation is ongoing, it is not possible to read the ADDR and DATA registers. Once an operation has started, the user has to wait until it has terminated by polling the DONE field in the Status Register (SR.DONE).

8.8.7 Chip Erase

The Chip erase operation consists in:

1. clearing all the volatile memories in the system
2. clearing the whole flash array
3. clearing the protected state

No proprietary or sensitive information is left in volatile memories once the protected state is disabled.

This feature is operated by writing a one to the CE bit of the Control Register (CR.CE). When the operation completes, SR.DONE is asserted.

8.8.8 Cortex-M4 Core Reset Source

The SMAP processes the EDP Core hold reset requests (Refer to Section 8.7.8 “SMAP Core Reset Request Source” on page 70). When requested, it instructs the Power Manager to hold the Cortex-M4 core under reset.

The SMAP can de-assert the core reset request if a one is written to the Hold Core Reset bit in the Status Clear Register (SCR.HCR). This has the effect of releasing the CPU from its reset state. To assert again this signal, a new reset sequence with TCK tied low must be issued.

Note that clearing HCR with this module is only possible when it is enabled, for more information refer to Section 8.8.4 “Initializing the Module” on page 77. Also note that asserting RESET_N automatically clears HCR.

8.8.9 Unlimited Flash User Page Read Access

The SMAP can access the User page even if the protected state is set. Prior to operate such an access, the user should check that the module is not busy by checking that SR.STATE is equal to zerp. Once the offset of the word to access inside the page is written in ADDR.ADDR, the read operation can be initiated by writing a one in CR.FSPR. The SR.STATE field will indicate the FSPR state. Addresses written to ADDR.ADDR must be world aligned. Failing to do so will result in unpredictable behavior. The result can be read in the DATA register as soon as SR.DONE rises. The ADDR field is used as an offset in the page, bits outside a page boundary will be silently discarded. The ADDR register is automatically incremented at the end of the read operation making possible to dump consecutive words without writing the next offset into ADDR.ADDR.

8.8.10 32-bit Cyclic Redundancy Check (CRC)

The SMAP unit provides support for calculating a Cyclic Redundancy Check (CRC) value for a memory area. The algorithm used is the industry standard CRC32 algorithm using the generator polynomial 0xEDB88320.

8.8.10.1 Starting CRC Calculation

To calculate CRC for a memory range, the start address must be written into the ADDR register, and the size of the memory range into the LENGTH register. Both the start address and the length must be word aligned.

The initial value used for the CRC calculation must be written to the DATA register. This value will usually be 0xFFFFFFFF, but can be e.g. the result of a previous CRC calculation if generating a common CRC of separate memory blocks.

Once completed, the calculated CRC value can be read out of the DATA register. The read value must be inverted to match standard CRC32 implementations, or kept non-inverted if used as starting point for subsequent CRC calculations.

If the device is in protected state, it is only possible to calculate the CRC of the whole flash array. In most cases this area will be the entire onboard nonvolatile memory. The ADDR, LENGTH, and DATA registers will be forced to predefined values once the CRC operation is started, and user-written values are ignored. This allows the user to verify the contents of a protected device.

The actual test is started by writing a one in CR.CRC. A running CRC operation can be cancelled by disabling the module (write a one in CR.DIS). This has the effect of resetting the module. The module has to be restarted by issuing an enable command (write a one in CR.EN).

8.8.10.2 Interpreting the Results

The user should monitor the SR register (Refer to Section 8.8.11.2 “Status Register” on page 82). When the operation is completed SR.DONE is set. Then the SR.BERR and SR.FAIL must be read to ensure that no bus error nor functional error occured.

8.8.11 SMAP User Interface

Table 8-9. SMAP Register Memory Map

Offset	Register	Register Name	Access (unprotected)	Access (protected)	Reset
0x0000	Control Register	CR	Write-Only	Write-Only (partial)(2)	0x00000000
0x0004	Status Register	SR	Read-Only	Read-Only	0x00000000
0x0008	Status Clear Register	SCR	Write-Only	Write-Only (partial)(3)	0x00000000
0x000C	Address Register	ADDR	Read/Write	Read/Write (partial)(4)	0x00000000
0x0010	Length Register	LENGTH	Read/Write	denied	0x00000000
0x0014	Data Register	DATA	Read/Write	Read/Write (partial)(4)	0x00000000
0x0028	VERSION Register	VERSION	Read-Only	Read-Only	-(1)
0x00F0	Chip ID Register	CIDR	Read-Only	Read-Only	-(1)
0x00F4	Chip ID Extension Register	EXID	Read-Only	Read-Only	-(1)
0x00FC	AP Identification register	IDR	Read-Only	Read-Only	0x003E0000

Note:
1. The reset value for this register is device specific. Refer to the Module Configuration section at the end of this chapter.
2. CR.MBIST is ignored
3. SCR.HCR is ignored
4. Access is not allowed when an operation is ongoing

8.8.11.1 Control Register

Name: CR

Access Type: Write-Only

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	CE	FSPR	CRC	DIS	EN

Writing a zero to a bit in this register has no effect.

- **CE: Chip Erase**
  Writing a one to this bit triggers the FLASH Erase All (EA) operation which clears all volatile memories, the whole flash array, the general purpose fuses and the protected state. The Status register DONE field indicates the completion of the operation.
  Reading this bit always returns 0
- **FSPR: Flash User Page Read**
  Writing a one to this bit triggers a read operation in the User page. The word pointed by the ADDR register in the page is read and written to the DATA register. ADDR is post incremented allowing a burst of reads without modifying ADDR. SR.DONE must be read high prior to reading the DATA register.
  Reading this bit always returns 0
- **CRC: Cyclic Redundancy Code**
  Writing a one triggers a CRC calculation over a memory area defined by the ADDR and LENGTH registers. Reading this bit always returns 0
  Note: This feature is restricted while in protected state
- **DIS: Disable**
  Writing a one to this bit disables the module. Disabling the module resets the whole module immediately.
- **EN: Enable**
  Writing a one to this bit enables the module.

8.8.11.2 Status Register

Name: SR

Access Type: Read-Only

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	STATE		

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	DBGP	PROT	EN

7	6	5	4	3	2	1	0
-	-	-	LCK	FAIL	BERR	HCR	DONE

- **STATE: State**

Value	State	Description
0	IDLE	Idle state
1	CE	Chip erase operation is ongoing
2	CRC32	CRC32 operation is ongoing
3	FSPR	Flash User Page Read
4-7	-	reserved

- **DBGP: Debugger present**
  1: A debugger is present (TCK falling edge detected)
  0: No debugger is present
- **PROT: Protected**
  1: The protected state is set. The only way to overcome this is to issue a Chip Erase command.
  0: The protected state is not set
- **EN: Enabled**
  1: The block is in ready for operation
  0: the block is disabled. Write operations are not possible until the block is enabled by writing a one in CR.EN.
- **LCK: Lock**
  1: An operation could not be performed because chip protected state is on.
  0: No security issues have been detected sincle last clear of this bit
- **FAIL: Failure**
  1: The requested operation failed
  0: No failure has been detected sincle last clear of this bit
- **BERR: Bus Error**
  1: A bus error occured due to the unability to access part of the requested memory area.

0: No bus error has been detected sincle last clear of this bit

- **HCR: Hold Core reset**
  1: The Cortex-M4 core is held under reset
  0: The Cortex-M4 core is not held under reset
- **DONE: Operation done**
  1: At least one operation has terminated since last clear of this field
  0: No operation has terminated since last clear of this field

8.8.11.3 Status Clear Register

Name: SCR

Access Type: Write-Only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	LCK	FAIL	BERR	HCR	DONE

Writing a zero to a bit in this register has no effect.
Writing a one to a bit clears the corresponding SR bit

Note: *Writing a one to bit HCR while the chip is in protected state has no effect*

8.8.11.4 Address Register

Name: ADDR

Access Type: Read/Write

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR						-	-

- **ADDR: Address Value**
  Addess values are always world aligned

8.8.11.5 Length Register

Name: LENGTH

Access Type: Read/Write

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
LENGTH							

23	22	21	20	19	18	17	16
LENGTH							

15	14	13	12	11	10	9	8
LENGTH							

7	6	5	4	3	2	1	0
LENGTH						-	-

- **LENGTH: Length Value, Bits 1-0 are always zero**

8.8.11.6 Data Register

Name: DATA

Access Type: Read/Write

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DATA							

23	22	21	20	19	18	17	16
DATA							

15	14	13	12	11	10	9	8
DATA							

7	6	5	4	3	2	1	0
DATA							

- **DATA: Generic data register**

8.8.11.7 Module Version

Name: VERSION

Access Type: Read-Only

Offset: 0x28

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION			

7	6	5	4	3	2	1	0
VERSION							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

8.8.11.8 *Chip Identification Register*

Name: CIDR

Access Type: Read-Only

Offset: 0xF0

Reset Value: -

31	30	29	28	27	26	25	24
EXT	NVPTYP			ARCH			

23	22	21	20	19	18	17	16
ARCH				SRAMSIZ			

15	14	13	12	11	10	9	8
NVPSIZ2				NVPSIZ			

7	6	5	4	3	2	1	0
EPROC				VERSION			

Note: Refer to section CHIPID for more information on this register.

8.8.11.9 Chip Identification Extension Register

Name: EXID

Access Type: Read-Only

Offset: 0xF4

Reset Value: -

31	30	29	28	27	26	25	24
EXID							

23	22	21	20	19	18	17	16
EXID							

15	14	13	12	11	10	9	8
EXID							

7	6	5	4	3	2	1	0
EXID							

Note: Refer to section CHIPID for more information on this register.

8.8.11.10 Identification Register

Name: IDR

Access Type: Read-Only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
REVISION				CC			

23	22	21	20	19	18	17	16
IC							CLSS

15	14	13	12	11	10	9	8
Reserved							

7	6	5	4	3	2	1	0
APID				APIDV			

- **REVISION: Revision**
- **CC: JEP-106 Continuation Code**
  Atmel continuation code is 0x0
- **IC: JEP-106 Identity Code**
  Atmel identification code is 0x1F
- **CLSS: Class**
  0: This AP is not a Memory Access Port
  1: This AP is a Memory Access Port
- **APID: AP Identification**
- **APIDV: AP Identification Variant**

For more information about this register, refer to the ARM Debug Interface v5.1 Architecture Specification document.

8.9 AHB-AP Access Port

The AHB-AP is a Memory Access Port (MEM-AP) as defined in the ARM Debug Interface v5 Architecture Specification. The AHB-AP provides access to all memory and registers in the system, including processor registers through the System Control Space (SCS). System access is independent of the processor status. Either SW-DP or SWJ-DP is used to access the AHB-AP. The AHB-AP is a master into the Bus Matrix. Transactions are made using the AHB-AP programmers model (refer to the ARM Cortex-M4 Technical Reference Manual), which generates AHB-Lite transactions into the Bus Matrix. The AHB-AP does not perform back-to-back transactions on the bus, so all transactions are non-sequential. The AHB-AP can perform unaligned and bit-band transactions. The Bus Matrix handles these. The AHB-AP transactions are not subject to MPU lookups. AHB-AP transactions bypass the FPB, and so the FPB cannot remap AHB-AP transactions. AHB-AP transactions are little-endian.

Note that while an external reset is applied, AHB-AP accesses are not possible. In addition, access is denied when the protected state is set. In order to discard the protected state, a chip erase operation is necessary.

8.10 Available Features in Protected State

Table 8-10. Features availablility when in protected state

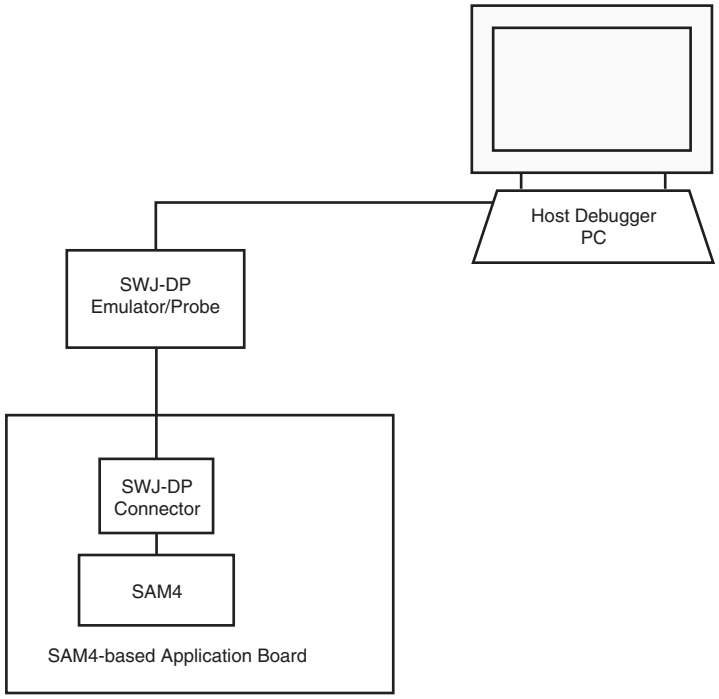
Feature	Provider	Availability when protected
Hot plugging	EDP	yes
System bus R/W Access	AHB-AP	no
Flash User Page read access	SMAP	yes
Core Hold Reset clear from the SMAP interface	SMAP	no
CRC32 of any memory accessible through the bus matrix	SMAP	restricted (limited to the entire flash array)
Chip Erase	SMAP	yes
IDCODE	EDP	yes
FPB, DWT & ITM	Cortex-M4	no

8.11 Functional Description

8.11.1 Debug Environment

Figure 8-8 shows a complete debug environment example. The SWJ-DP interface is used for standard debugging functions, such as downloading code and single-stepping through the program and viewing core and peripheral registers.

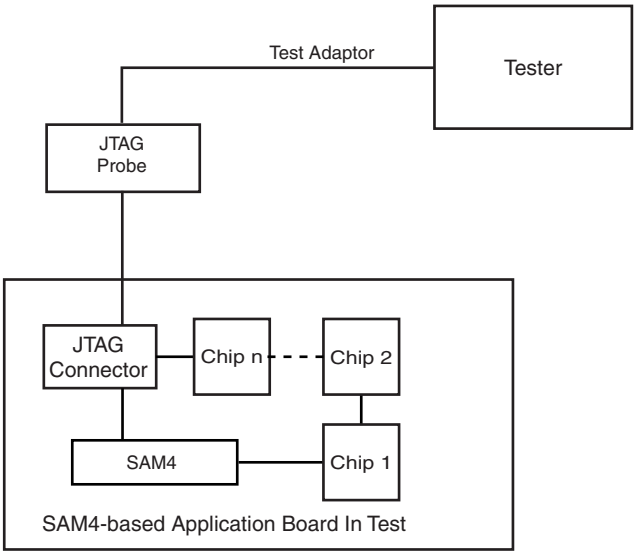
Figure 8-8. Application Debug Environment Example



8.11.2 Test Environment

Figure 8-9 shows a test environment example (JTAG Boundary scan). Test vectors are sent and interpreted by the tester. In this example, the “board in test” is designed using a number of JTAG-compliant devices. These devices can be connected to form a single scan chain.

Figure 8-9. Application Test Environment Example



8.11.3 How to Initialize Test and Debug Features

To enable the JTAG pins a falling edge event must be detected on the TCK pin at any time after the RESET_N pin is released.

In some specific cases, the user would prevent system from running code after reset is released. This is done by holding low the TCK pin when the RESET_N is released. This makes the SMAP assert the core_hold_reset signal that hold the Cortex-M4 core under reset.

To make the CPU run again, the user should clear the CHR bit in the Status Register (SR.CHR) to de-assert the core_hold_reset signal. Independent of the initial state of the TAP Controller, the Test-Logic-Reset state can always be entered by holding TMS high for 5 TCK clock periods. This sequence should always be applied at the start of a JTAG session and after enabling the JTAG pins to bring the TAP Controller into a defined state before applying JTAG commands. Applying a zero on TMS for 1 TCK period brings the TAP Controller to the Run-Test/Idle state, which is the starting point for JTAG operations.

8.11.4 How to Disable Test and Debug Features

To disable the JTAG pins the TCK pin must be held high while RESET_N pin is released.

8.11.5 Typical JTAG Sequence

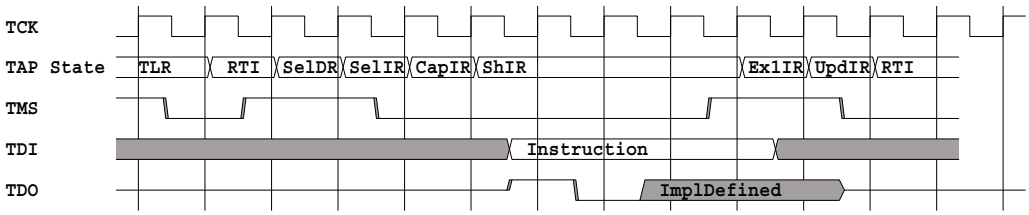
Assuming Run-Test/Idle is the present state, a typical scenario for using the JTAG interface is:

8.11.5.1 Scanning in JTAG Instruction

At the TMS input, apply the sequence 1, 1, 0, 0 at the rising edges of TCK to enter the Shift Instruction Register - Shift-IR state. While in this state, shift the 4 bits of the JTAG instructions into the JTAG instruction register from the TDI input at the rising edge of TCK. The TMS input must be held low during input of the 4 LSBs in order to remain in the Shift-IR state. The JTAG Instruction selects a particular Data Register as path between TDI and TDO and controls the circuitry surrounding the selected Data Register.

Apply the TMS sequence 1, 1, 0 to re-enter the Run-Test/Idle state. The instruction is latched onto the parallel output from the shift register path in the Update-IR state. The Exit-IR, Pause-IR, and Exit2-IR states are only used for navigating the state machine.

Figure 8-10. Scanning in JTAG Instruction



8.11.5.2 Scanning In/Out Data

At the TMS input, apply the sequence 1, 0, 0 at the rising edges of TCK to enter the Shift Data Register - Shift-DR state. While in this state, upload the selected Data Register (selected by the present JTAG instruction in the JTAG Instruction Register) from the TDI input at the rising edge of TCK. In order to remain in the Shift-DR state, the TMS input must be held low. While the Data Register is shifted in from the TDI pin, the parallel inputs to the Data Register captured in the Capture-DR state is shifted out on the TDO pin.

Apply the TMS sequence 1, 1, 0 to re-enter the Run-Test/Idle state. If the selected Data Register has a latched parallel-output, the latching takes place in the Update-DR state. The Exit-DR, Pause-DR, and Exit2-DR states are only used for navigating the state machine.

As shown in the state diagram, the Run-Test/Idle state need not be entered between selecting JTAG instruction and using Data Registers.

8.11.6 Boundary-Scan

The Boundary-Scan chain has the capability of driving and observing the logic levels on the digital I/O pins, as well as the boundary between digital and analog logic for analog circuitry having off-chip connections. At system level, all ICs having JTAG capabilities are connected serially by the TDI/TDO signals to form a long shift register. An external controller sets up the devices to drive values at their output pins, and observe the input values received from other devices. The controller compares the received data with the expected result. In this way, Boundary-Scan provides a mechanism for testing interconnections and integrity of components on Printed Circuits Boards by using the 4 TAP signals only.

The four IEEE 1149.1 defined mandatory JTAG instructions IDCODE, BYPASS, SAMPLE/PRELOAD, and EXTEST can be used for testing the Printed Circuit Board. Initial scanning of the data register path will show the ID-code of the device, since IDCODE is the default JTAG instruction. It may be desirable to have the device in reset during test mode. If not reset, inputs to the device may be determined by the scan operations, and the internal software may be in an undetermined state when exiting the test mode. Entering reset, the outputs of any Port Pin will instantly enter the high impedance state, making the HIGHZ instruction redundant. If needed, the BYPASS instruction can be issued to make the shortest possible scan chain through the device. The device can be set in the reset state by pulling the external RESET_N pin low.

The EXTEST instruction is used for sampling external pins and loading output pins with data. The data from the output latch will be driven out on the pins as soon as the EXTEST instruction is loaded into the JTAG IR-register. Therefore, the SAMPLE/PRELOAD should also be used for setting initial values to the scan ring, to avoid damaging the board when issuing the EXTEST

instruction for the first time. SAMPLE/PRELOAD can also be used for taking a snapshot of the external pins during normal operation of the part.

When using the JTAG interface for Boundary-Scan, the JTAG TCK clock is independent of the internal chip clock, which is not required to run.

NOTE: For pins connected to 5V lines care should be taken to not drive the pins to a logic one using boundary scan, as this will create a current flowing from the 3,3V driver to the 5V pullup on the line. Optionally a series resistor can be added between the line and the pin to reduce the current.

8.11.7 Flash Programming Typical Procedure

Flash programming is performed by operating Flash controller commands. The Flash controller is connected to the system bus matrix and is then controllable from the AHP-AP. The AHB-AP cannot write the FLASH page buffer while the core_hold_reset is asserted. The AHB-AP cannot be accessed when the device is in protected state. It is important to ensure that the CPU is halted prior to operating any flash programming operation to prevent it from corrupting the system configuration. The recommended sequence is shown below:

1. At power up, RESET_N is driven low by a debugger. The on-chip regulator holds the system in a POR state until the input supply is above the POR threshold. The system continues to be held in this static state until the internally regulated supplies have reached a safe operating.
2. PM starts, clocks are switched to the slow clock (Core Clock, System Clock, Flash Clock, and any Bus Clocks that do not have clock gate control). Internal resets are maintained due to the external reset.
   – The Debug Port (DP) and Access Ports (AP) receives a clock and leave the reset state,
3. The debugger maintains a low level on TCK and release RESET_N.
   – The SMAP asserts the core_hold_reset signal
4. The Cortex-M4 core remains in reset state, meanwhile the rest of the system is released.
5. The debugger then configures the NVIC to catch the Cortex-M4 core reset vector fetch. For more information on how to program the NVIC, refer to the ARMv7-M Architecture Reference Manual.
6. The debugger writes a one in the SMAP SCR.HCR to release the Cortex-M4 core reset to make the system bus matrix accessible from the AHB-AP.
7. The Cortex-M4 core initializes the SP, then read the exception vector and stalls
8. Programming is available through the AHB-AP
9. After operation is completed, the chip can be restarted either by asserting RESET_N or switching power off/on or clearing SCR.HCR. Make sure that the TCK pin is high when releasing RESET_N not to halt the core.

8.11.8 Chip Erase Typical Procedure

The chip erase operation is triggered by writing a one in the CE bit in the Control Register (CR.CE). This clears first all volatile memories in the system and second the whole flash array. Note that the User page is not erased in this process. To ensure that the chip erase operation is completed, check the DONE bit in the Status Register (SR.DONE). Also note that the chip erase operation depends on clocks and power management features that can be altered by the CPU. It is important to ensure that it is stopped. The recommended sequence is shown below:

1. At power up, RESET_N is driven low by a debugger. The on-chip regulator holds the system in a POR state until the input supply is above the POR threshold. The system continues to be held in this static state until the internally regulated supplies have reached a safe operating.
2. PM starts, clocks are switched to the slow clock (Core Clock, System Clock, Flash Clock, and any Bus Clocks that do not have clock gate control). Internal resets are maintained due to the external reset.
   – The debug port and access ports receives a clock and leave the reset state
3. The debugger maintains a low level on TCK and release RESET_N.
   – The SMAP asserts the core_hold_reset signal
4. The Cortex-M4 core remains in reset state, meanwhile the rest of the system is released.
5. The Chip erase operation can be performed by issuing the SMAP Chip Erase command. In this case:
   – volatile memories are cleared first
   – followed by the clearing of the flash array
   – followed by the clearing of the protected state
6. After operation is completed, the device must be restarted by either controling RESET_N or switching power off/on. Make sure that the TCK pin is high when releasing RESET_N not to halt the core.

8.11.9 Setting the Protected State

This is done by issuing a specific flash controller command, for more information, refer to Section 14. “Flash Controller (FLASHCALW)” on page 263 and to Section 8.11.7 “Flash Programming Typical Procedure” on page 97. The protected state is defined by a highly secure Flash builtin mechanism. Note that for this programmation to propagate, it is required to reset the device.

9. Chip Identifier (CHIPID)

9.1 Description

Chip Identifier registers permit recognition of the device and its revision. These registers provide the sizes and types of the on-chip memories, as well as the set of embedded peripherals.

Two chip identifier registers are embedded: CIDR (Chip ID Register) and EXID (Extension ID). Both registers contain a hard-wired value that is read-only. The first register contains the following fields:

- EXT - shows the use of the extension identifier register
- NVPTYP and NVPSIZ - identifies the type of embedded non-volatile memory and its size
- ARCH - identifies the set of embedded peripherals
- SRAMSIZ - indicates the size of the embedded SRAM
- EPROC - indicates the embedded ARM processor
- VERSION - gives the revision of the silicon

The second register is device-dependent and reads 0 if the bit EXT is 0.

9.2 Embedded Characteristics

- Chip ID Registers
  - Identification of the Device Revision, Sizes of the Embedded Memories, Set of Peripherals, Embedded Processor

Table 9-1. ATSAM4L Chip IDs Register

Chip Name	Family	Flash/RAM	Package	CHIPID_CIDR	CHIPID_EXID
ATSAM4LC8C (Rev A)	ATSAM4LC	512K/64K	100-pin	0xAB0B0AE0	0x1400000F
ATSAM4LC4C (Rev A)	ATSAM4LC	256K/32K	100-pin	0xAB0A09E0	0x0400000F
ATSAM4LC2C(Rev A)	ATSAM4LC	128K/32K	100-pin	0xAB0A07E0	0x0400000F
ATSAM4LC8B (Rev A)	ATSAM4LC	512K/64K	64-pins	0xAB0B0AE0	0x1300000F
ATSAM4LC4B (Rev A)	ATSAM4LC	256K/32K	64-pins	0xAB0A09E0	0x0300000F
ATSAM4LC2B (Rev A)	ATSAM4LC	128K/32K	64-pins	0xAB0A07E0	0x0300000F
ATSAM4LC8A (Rev A)	ATSAM4LC	512K/64K	48-pins	0xAB0B0AE0	0x1200000F
ATSAM4LC4A (Rev A)	ATSAM4LC	256K/32K	48-pins	0xAB0A09E0	0x0200000F
ATSAM4LC2A (Rev A)	ATSAM4LC	128K/32K	48-pins	0xAB0A07E0	0x0200000F
ATSAM4LS8C (Rev A)	ATSAM4LS	512K/64K	100-pins	0xAB0B0AE0	0x14000002
ATSAM4LS4C (Rev A)	ATSAM4LS	256K/32K	100-pins	0xAB0A09E0	0x04000002
ATSAM4LS2C (Rev A)	ATSAM4LS	128K/32K	100-pins	0xAB0A07E0	0x04000002
ATSAM4LS8B (Rev A)	ATSAM4LS	512K/64K	64-pins	0xAB0B0AE0	0x13000002
ATSAM4LS4B (Rev A)	ATSAM4LS	256K/32K	64-pins	0xAB0A09E0	0x03000002
ATSAM4LS2B (Rev A)	ATSAM4LS	128K/32K	64-pins	0xAB0A07E0	0x03000002

Table 9-1. ATSAM4L Chip IDs Register

Chip Name	Family	Flash/RAM	Package	CHIPID_CIDR	CHIPID_EXID
ATSAM4LS8A (Rev A)	ATSAM4LS	512K/64K	48-pins	0xAB0B0AE0	0x12000002
ATSAM4LS4A (Rev A)	ATSAM4LS	256K/32K	48-pins	0xAB0A09E0	0x02000002
ATSAM4LS2A (Rev A)	ATSAM4LS	128K/32K	48-pins	0xAB0A07E0	0x02000002

9.3 User Interface

Table 9-2. CHIPID Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0	Chip ID Register	CIDR	Read-only	–
0x4	Chip ID Extension Register	EXID	Read-only	–

9.3.1 Chip ID Register

Name: CIDR

Access: Read-only

Offset: 0x0

Reset Value: -

31	30	29	28	27	26	25	24
EXT	NVPTYP			ARCH[7:4]			
23	22	21	20	19	18	17	16
ARCH[3:0]				SRAMSIZ			
15	14	13	12	11	10	9	8
NVPSIZ2				NVPSIZ			
7	6	5	4	3	2	1	0
EPROC			VERSION				

- **EXT: Extension Flag**
  0 = Chip ID has a single register definition without extension
  1 = An extended Chip ID exists.

- **NVPTYP: Nonvolatile Program Memory Type**

Value	Name	Description
0	ROM	ROM
1	ROMLESS	ROMless or on-chip Flash
4	SRAM	SRAM emulating ROM
2	FLASH	Embedded Flash Memory
3	ROM_FLASH	ROM and Embedded Flash Memory NVPSIZ is ROM size NVPSIZ2 is Flash size

- **ARCH: Architecture Identifier**

Value	Name	Description
0x19	AT91SAM9xx	AT91SAM9xx Series
0x29	AT91SAM9XExx	AT91SAM9XExx Series
0x34	AT91x34	AT91x34 Series
0x37	CAP7	CAP7 Series
0x39	CAP9	CAP9 Series
0x3B	CAP11	CAP11 Series
0x40	AT91x40	AT91x40 Series
0x42	AT91x42	AT91x42 Series
0x55	AT91x55	AT91x55 Series
0x60	AT91SAM7Axx	AT91SAM7Axx Series
0x61	AT91SAM7AQxx	AT91SAM7AQxx Series
0x63	AT91x63	AT91x63 Series
0x70	AT91SAM7Sxx	AT91SAM7Sxx Series
0x71	AT91SAM7XCxx	AT91SAM7XCxx Series
0x72	AT91SAM7SExx	AT91SAM7SExx Series
0x73	AT91SAM7Lxx	AT91SAM7Lxx Series
0x75	AT91SAM7Xxx	AT91SAM7Xxx Series
0x76	AT91SAM7SLxx	AT91SAM7SLxx Series
0x80	SAM3UxC	SAM3UxC Series (100-pin version)
0x81	SAM3UxE	SAM3UxE Series (144-pin version)
0x83	SAM3AxC	SAM3AxC Series (100-pin version)
0x83	SAM4AxC	SAM4AxC Series (100-pin version)
0x84	SAM3XxC	SAM3XxC Series (100-pin version)
0x84	SAM4XxC	SAM4XxC Series (100-pin version)
0x85	SAM3XxE	SAM3XxE Series (144-pin version)
0x85	SAM4XxE	SAM4XxE Series (144-pin version)
0x86	SAM3XxG	SAM3XxG Series (208/217-pin version)
0x86	SAM4XxG	SAM4XxG Series (208/217-pin version)
0x88	SAM3SxA	SAM3SxASeries (48-pin version)
0x88	SAM4SxA	SAM4SxA Series (48-pin version)
0x89	SAM3SxB	SAM3SxB Series (64-pin version)
0x89	SAM4SxB	SAM4SxB Series (64-pin version)
0x8A	SAM3SxC	SAM3SxC Series (100-pin version)
0x8A	SAM4SxC	SAM4SxC Series (100-pin version)
0x92	AT91x92	AT91x92 Series

Value	Name	Description
0x93	SAM3NxA	SAM3NxA Series (48-pin version)
0x94	SAM3NxB	SAM3NxB Series (64-pin version)
0x95	SAM3NxC	SAM3NxC Series (100-pin version)
0x99	SAM3SDxB	SAM3SDxB Series (64-pin version)
0x9A	SAM3SDxC	SAM3SDxC Series (100-pin version)
0xA5	SAM5A	SAM5A
0xB0	SAM4L	SAM4Lxx Series
0xF0	AT75Cxx	AT75Cxx Series

- **SRAMSIZ: Internal SRAM Size**

Value	Name	Description
0	48K	48K bytes
1	1K	1K bytes
2	2K	2K bytes
3	6K	6K bytes
4	24K	24K bytes
5	4K	4K bytes
6	80K	80K bytes
7	160K	160K bytes
8	8K	8K bytes
9	16K	16K bytes
10	32K	32K bytes
11	64K	64K bytes
12	128K	128K bytes
13	256K	256K bytes
14	96K	96K bytes
15	512K	512K bytes

- **NVPSIZ2: Second Nonvolatile Program Memory Size**

Value	Name	Description
0	NONE	None
1	8K	8K bytes
2	16K	16K bytes
3	32K	32K bytes
4		Reserved
5	64K	64K bytes
6		Reserved
7	128K	128K bytes
8		Reserved
9	256K	256K bytes
10	512K	512K bytes
11		Reserved
12	1024K	1024K bytes
13		Reserved
14	2048K	2048K bytes
15		Reserved

- **EPROC: Embedded Processor**

Value	Name	Description
1	ARM946ES	ARM946ES
2	ARM7TDMI	ARM7TDMI
3	CM3	Cortex-M3
4	ARM920T	ARM920T
5	ARM926EJS	ARM926EJS
6	CA5	Cortex-A5
7	CM4	Cortex-M4

- **NVPSIZ: Nonvolatile Program Memory Size**

Value	Name	Description
0	NONE	None
1	8K	8K bytes
2	16K	16K bytes
3	32K	32K bytes

Value	Name	Description
4		Reserved
5	64K	64K bytes
6		Reserved
7	128K	128K bytes
8		Reserved
9	256K	256K bytes
10	512K	512K bytes
11		Reserved
12	1024K	1024K bytes
13		Reserved
14	2048K	2048K bytes
15		Reserved

- **VERSION: Version of the Device**
  Current version of the device.

9.3.2 Extension Register

Name: EXID

Access: Read-only

Offset: 0x4

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	PACKAGE		

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	LCD	USBFULL	USB	AES

- **PACKAGE: Package Type**

Value	Description
0	24-pin package
1	32-pin package
2	48-pin package
3	64-pin package
4	100-pin package
5	144-pin package
6	reserved
7	

- **LCD: LCD Option**

Value	Description
0	LCD is not implemented
1	LCD is implemented

- **USB: USB Option**

Value	Description
0	USB is not implemented
1	USB is implemented

- **USBFULL: USB Configuration**

Value	Description
0	USB is Device-only
1	USB is Device and Host

- **AES: AES Option**

Value	Description
0	AES is not implemented
1	AES is implemented

10. Power Manager (PM)

Rev: 4.4.1.1

10.1 Features

- **Generates clocks and resets for digital logic**
- **On-the-fly frequency change of CPU, HSB and PBx clocks**
- **Module-level clock gating through maskable peripheral clocks**
- **Controls resets of the device**

10.2 Overview

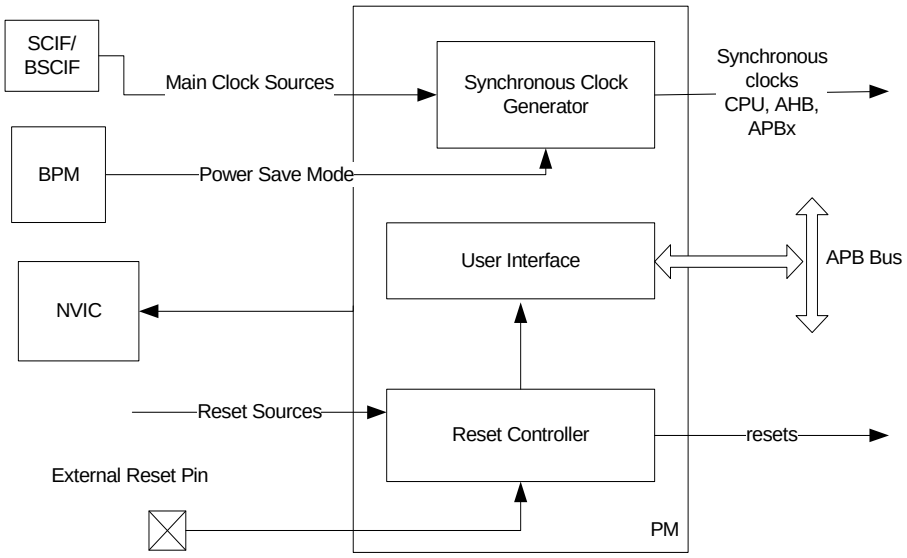
The Power Manager (PM) provides synchronous clocks used to clock the main digital logic in the device, namely the CPU, and the modules and peripherals connected to the High Speed Bus (AHB) and the Peripheral Buses (APBx).

The synchronous clocks are divided into a number of clock domains, one for the CPU and AHB and one for each APBx. The clocks can run at different speeds, so the user can save power by running peripherals at a relatively low clock, while maintaining a high CPU performance. Additionally, the clocks can be independently changed on-the-fly, without halting any peripherals. This enables the user to adjust the speed of the CPU and memories to the dynamic load of the application, without disturbing or re-configuring active peripherals. Each module also has a separate clock, enabling the user to switch off the clock for inactive modules, to save further power. Additionally, clocks and oscillators can be automatically switched off during Power Save Mode periods by using the Power Save Mode feature of the Backup Power Manager module (BPM).

The Power Manager also contains a Reset Controller, which collects all possible reset sources, generates hard and soft resets, and allows the reset source to be identified by software.

10.3 Block Diagram

Figure 10-1. PM Block Diagram



10.4 I/O Lines Description

Table 10-1. I/O Lines Description

Name	Description	Type	Active Level
RESET_N	Reset	Input	Low

10.5 Product Dependencies

10.5.1 Interrupt

The PM interrupt line is connected to one of the internal sources of the NVIC. Using the PM interrupt requires the NVIC to be programmed first.

10.5.2 Clock Implementation

In ATSAM4L, the AHB shares the source clock with the CPU.

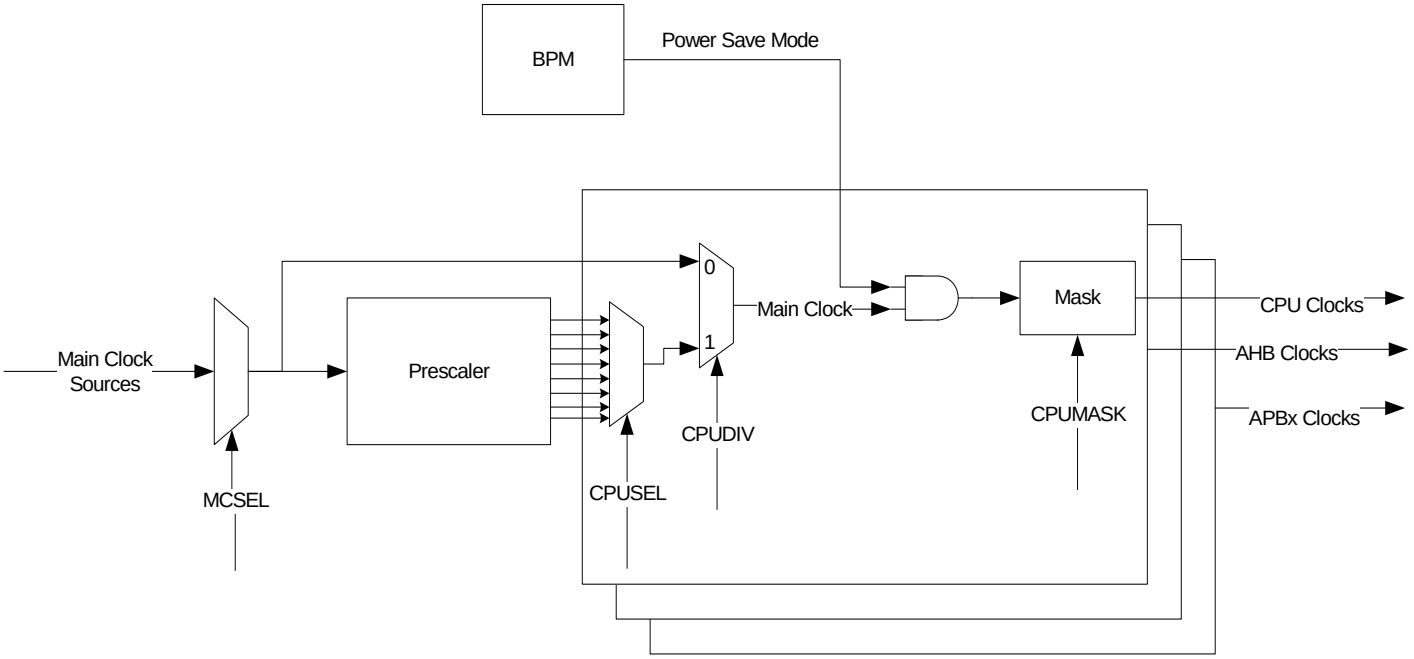
The clock for the PM bus interface (CLK_PM) is generated by the Power Manager. This clock is enabled at reset, and can be disabled in the Power Manager. If disabled, it can only be re-enabled by a reset.

10.6 Functional Description

10.6.1 Synchronous Clocks

The System RC Oscillator (RCSYS) or a set of other clock sources provide the source for the main clock, which is the common root for the synchronous clocks for the CPU, AHB and APBx modules. For details about the other main clock sources, refer to the register description of the Main Clock Control Register (MCCTRL). The main clock is divided by an 8-bit prescaler, and each of these synchronous clocks can run from any tapping of this prescaler, or the undivided main clock, as long as $f_{CPU} \geq f_{APBx,}$. The synchronous clock source can be changed on-the fly, responding to varying load in the application. The clocks for each module in each synchronous clock domain can be individually masked, to avoid power consumption in inactive modules. Depending on the Power Save mode, some clock domains can be cut.

Figure 10-2. Synchronous Clock Generation



10.6.1.1 Selecting the Main Clock Source

The common main clock can be connected to RCSYS or a set of other clock sources. For details about the other main clock sources, refer to Section 10.7.1 ”Main Clock Control” on page 117. By default, the main clock will be connected to RCSYS. The user can connect the main clock to an other source by writing the MCSEL field in the MCCTRL register. This must only be done after that unit has been enabled and is ready, otherwise a deadlock will occur. Care should also be taken that the new frequency of the synchronous clocks does not exceed the maximum frequency for each clock domain.

10.6.1.2 Selecting Synchronous Clock Division Ratio

The main clock feeds an 8-bit prescaler, which can be used to generate the synchronous clocks. By default, the synchronous clocks run on the undivided main clock. The user can select a prescaler division for the CPU clock by writing CPUDIV in CPUSEL register to one and CPUSEL in CPUSEL register to the value, resulting in a CPU clock frequency:

$$f_{CPU} = f_{main} / 2^{(CPUSEL+1)}$$

Similarly, the clock for the APBx can be divided by writing their respective registers. To ensure correct operation, frequencies must be selected so that $f_{CPU} \geq f_{APBx}$. Also, frequencies must never exceed the specified maximum frequency for each clock domain.

CPUSEL and PBxSEL can be written without halting or disabling peripheral modules. Writing CPUSEL and PBxSEL allows a new clock setting to be written to all synchronous clocks at the same time. It is possible to keep one or more clocks unchanged by writing a one to the registers. This way, it is possible to, e.g., scale CPU and speed according to the required performance, while keeping the APBx frequency constant.

For modules connected to the AHB bus, the APB clock frequency must be set to the same frequency as the CPU clock.

10.6.1.3 Clock Ready Flag

There is a slight delay from CPUSEL and PBxSEL being written to the new clock setting taking effect. During this interval, the Clock Ready bit in the Status Register (SR.CKRDY) will read as zero. When the clock settings change is completed, the bit will read as one. The Clock Select registers (CPUSEL, PBxSEL) must not be written to while SR.CKRDY is zero, or the system may become unstable or hang.

The Clock Ready bit in the Interrupt Status Register (ISR.CKRDY) is set on a SR.CKRDY zero-to-one transition. If the Clock Ready bit in the Interrupt Mask Register (IMR.CKRDY) is set, an interrupt request is generated. IMR.CKRDY is set by writing a one to the corresponding bit in the Interrupt Enable Register (IER.CKRDY).

10.6.2 Peripheral Clock Masking

By default, only the necessary clocks are enabled (see the reset value of the MASK registers). It is possible to disable or enable the clock for a module in the CPU, or APBx clock domain by writing the corresponding bit in the Clock Mask register (CPU/HSB/PBx) to zero or one. When a module is not clocked, it will cease operation, and its registers cannot be read or written. The module can be re-enabled later by writing the corresponding mask bit to one.

A module may be connected to several clock domains, in which case it will have several mask bits.

The Maskable Module Clocks table contains a list of implemented maskable clocks.

10.6.2.1 Cautionary Note

Note that clocks should only be switched off if it is certain that the module will not be used. Switching off the clock for the flash controller will cause a problem if the CPU needs to read from the flash. Switching off the clock to the Power Manager (PM), which contains the mask registers, or the corresponding APBx bridge, will make it impossible to write the mask registers again. In this case, they can only be re-enabled by a system reset.

10.6.2.2 SleepWalking™

In all power save modes except in BACKUP mode, where the APBx clocks are stopped, the device can wake partially up if a APBx module asynchronously discovers that it needs its clock. Only the requested clocks and clock sources needed will be started, and all other clocks will be masked to zero. E.g. if the main clock source is OSC0, only OSC0 will be started even if other clock sources were enabled in RUN mode. Also generic clocks can be started in a similar way. The state where only requested clocks are running is referred to as SleepWalking.

The time spent to start the requested clock is mostly limited by the startup time of the given clock source. This allows APBx modules to handle incoming requests, while still keeping the power consumption at a minimum.

When the device is SleepWalking any asynchronous wake up sources can wake up the device at any time without stopping the requested APBx clock.

All requests to start clocks can be masked by writing to the Peripheral Power Control Register (PPCR), all requests are enabled at reset.

During SleepWalking the NVIC clock will be running except in BACKUP mode. If an interrupt is pending when entering SleepWalking, this will wake up the whole device.

10.6.3 Speeding-up Sleep Modes Wake Up Times

The normal way for the Power Manager to enter WAIT mode involves automatically switching the main clock to RCSYS before stopping all oscillators. During wake-up, the main clock is automatically switched back from RCSYS to the oscillator selected before the WFI instruction was executed. The delay needed to switch to/from the RCSYS oscillator is around three RCSYS clock cycles, plus oscillator startup times when waking up.

It is possible to speed-up the enter/wake-up times for the WAIT modes by disabling the automatic switch to RCSYS. This is only useful if a high frequency oscillator with a very fast startup time is used. The user has to select the main clock source by writing a one to the corresponding bit in the FASTSLEEP register. The device will not wake up correctly if the right oscillator in the FASTSLEEP register is not selected.

To have a fast wakeup time in WAIT and RETENTION power save mode, refer to Section 6.1.4 ”Wakeup Time” on page 56

10.6.4 Divided APB Clocks

The clock generator in the Power Manager provides divided APBx clocks for use by peripherals that require a prescaled APBx clock. This is described in the documentation for the relevant modules.

The divided clocks are directly maskable, and are stopped in Power Save modes where the APBx clocks are stopped.

10.6.5 Reset Controller

The Reset Controller collects the various reset sources and generates resets for the device.

The device contains a Power-On Detector, which keeps the system reset until power is stable. This eliminates the need for external reset circuitry to guarantee stable operation when powering up the device.

It is also possible to reset the device by pulling the RESET_N pin low. This pin has an internal pull-up, and does not need to be driven externally during normal operation. Table 10-2 on page 113 lists these and other reset sources supported by the Reset Controller.

Table 10-2. Reset Description

Reset source	Description
POR18 Reset	Supply voltage below the 1.8V power-on reset detector threshold voltage
External Reset	RESET_N pin asserted
BOD18 Reset	Supply voltage on VDDCORE below the 1.8V brownout reset detector threshold voltage
BOD33 Reset	Supply voltage on I/O below the 3.3V brownout reset detector threshold voltage
POR33 Reset	Supply voltage on I/O below the 3.3 V power-on reset detector threshold voltage
Watchdog Timer	See Watchdog Timer documentation
Backup Reset	Reset from backup mode when a wake up source occurs
OCD Reset	See On-Chip Debug and Test chapter

Depending on the reset source, when a Reset occurs, some parts of the device are not always reset. Only the Power On Reset (POR) will force a whole device reset. Refer to the table in the Module Configuration section at the end of this chapter for further details. The latest reset cause can be read in the RCAUSE register, and can be read during the applications boot sequence in order to determine proper action.

The table below lists parts of the device that are reset, depending on the reset type. The cause of the last reset can be read from the RCAUSE register. This register contains one bit for each reset source, and can be read during the boot sequence of an application.

10.6.6 Clock Failure Detector

This mechanism allows switching the main clock to the safe RCSYS clock, when the main clock source is considered off. This may happen when a external crystal is selected as the clock source of the main clock but the crystal is not mounted on the board. The mechanism is to detect, during a RCSYS period, at least one rising edge of the main clock. If no rising edge is seen the clock is considered failed.

Example:

* RCSYS = 115khz

=> Failure detected if the main clock is < 115 kHz

As soon as the detector is enabled, the clock failure detector will monitor the divided main clock. Note that the detector does not monitor if the RCSYS is the source of the main clock, or if the main clock is temporarily not available (startup-time after a wake-up, switching timing etc.), or in SLEEP mode where the main clock is driven by the RCSYS (SLEEP mode level 3). When a clock failure is detected, the main clock automatically switches to the RCSYS clock and the CFD interrupt is generated if enabled.

The MCCTRL register that selects the source clock of the main clock is changed by hardware to indicate that the main clock comes from RCSYS.

10.6.7 Interrupts

The PM has a number of interrupts:

- AE - Access Error,
  - A lock protected register is written to without first being unlocked.
- CKRDY - Clock Ready:
  - New Clock Select settings in the CPUSEL/PBxSEL registers have taken effect. (A zero-to-one transition on SR.CKRDY is detected).
- CFD - Clock Failure Detected:
  - The system detects that the main clock is not running.
- WAKE - Asynchronous wake Detected:
  - An asynchronous wakeup from a peripheral is detetcted.

The Interrupt Status Register contains one bit for each interrupt source. A bit in this register is set on a zero-to-one transition of the corresponding bit in the Status Register (SR), and cleared by writing a one to the corresponding bit in the Interrupt Clear Register (ICR). The interrupt sources will generate an interrupt request if the corresponding bit in the Interrupt Mask Register is set. The interrupt sources are ORed together to form one interrupt request. The Power Man-

ager will generate an interrupt request if at least one of the bits in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in the Interrupt Status Register (ISR) is cleared by writing a one to the corresponding bit in the Interrupt Clear Register (ICR). Because all the interrupt sources are ORed together, the interrupt request from the Power Manager will remain active until all the bits in ISR are cleared.

10.7 User Interface

Table 10-3. PM Register Memory Map

Offset	Register	Register Name	Access	Reset
0x000	Main Clock Control	MCCTRL	Read/Write	0x00000000
0x004	CPU Clock Select	CPUSEL	Read/Write	0x00000000
0x00C	PBA Clock Select	PBASEL	Read/Write	0x00000000
0x010	PBB Clock Select	PBBSEL	Read/Write	0x00000000
0x014	PBC Clock Select	PBCSEL	Read/Write	0x00000000
0x018	PBD Clock Select	PBDSEL	Read/Write	0x00000000
0x020	CPU Mask	CPUMASK	Read/Write	0x00000001
0x024	HSB Mask	HSBMASK	Read/Write	0x000001E2
0x028	PBA Mask	PBAMASK	Read/Write	0x00000000
0x02C	PBB Mask	PBBMASK	Read/Write	0x00000001
0x030	PBC Mask	PBCMASK	Read/Write	0x0000001F
0x034	PBD Mask	PBDMASK	Read/Write	0x0000003F
0x040	PBA Divided Mask	PBADIVMASK	Read/Write	0x00000000
0x054	Clock Failure Detector Control	CFDCTRL	Read/Write	0x00000000
0x058	Unlock Register	UNLOCK	Write-only	0x00000000
0x0C0	Interrupt Enable Register	IER	Write-only	0x00000000
0x0C4	Interrupt Disable Register	IDR	Write-only	0x00000000
0x0C8	Interrupt Mask Register	IMR	Read-only	0x00000000
0x0CC	Interrupt Status Register	ISR	Read-only	0x00000000
0x0D0	Interrupt Clear Register	ICR	Write-only	0x00000000
0x0D4	Status Register	SR	Read-only	0x00000000
0x160	Peripheral Power Control Register	PPCR	Read/Write	0x000001FE
0x180	Reset Cause Register	RCAUSE	Read-only	-(2)
0x184	Wake Cause Register	WCAUSE	Read-only	-(3)
0x188	Asynchronous Wake Enable	AWEN	Read/Write	0x00000000
0x18C	Protection Control Register	PROTCTRL	Read/Write	0x00000000
0x194	Fast Sleep Register	FASTSLEEP	Read/Write	0x00000000
0x3F8	Configuration Register	CONFIG	Read-only	0x0000000F
0x3FC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value is device specific. Refer to the Module Configuration section at the end of this chapter.
2. Latest Reset Source.
3. Latest Wake Source.

10.7.1 Main Clock Control

Name: MCCTRL

Access Type: Read/Write

Offset: 0x000

Reset Value: 0x00000000

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

- **MCSEL: Main Clock Select**

Table 10-4. Main clocks in ATSAM4L.

MCSEL[2:0]	Main clock source
0	System RC oscillator (RCSYS)
1	Oscillator0 (OSC0)
2	PLL
3	DFLL
4	80MHz RC oscillator (RC80M)(1)
5	4/8/12 MHz RC oscillator (RCFAST)
6	1 MHz RC oscillator (RC1M)
7	Reserved

Note: 1. If the 80MHz RC oscillator is selected as main clock source, it must be divided by at least 2 before being used as clock source for the CPU. This division is selected by writing to the CPUSEL and CPUDIV bits in the CPUSEL register, before switching to RC80M as main clock source.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.2 CPU Clock Select

Name: CPUSEL

Access Type: Read/Write

Offset: 0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
CPUDIV	-	-	-	-	CPUSEL		

- **CPUDIV, CPUSEL: CPU Division and Clock Select**
  CPUDIV = 0: CPU clock equals main clock.
  CPUDIV = 1: CPU clock equals main clock divided by $2^{(CPUSEL+1)}$.

Note that if CPUDIV is written to 0, CPUSEL should also be written to 0 to ensure correct operation.

Also note that writing this register clears SR.CKRDY. The register must not be re-written until CKRDY goes high.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.3 PBx Clock Select

Name: PBxSEL

Access Type: Read/Write

Offset: 0x00C-0x018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
PBDIV	-	-	-	-	PBSEL		

- **PBDIV, PBSEL: PBx Division and Clock Select**
  PBDIV = 0: APBx clock equals main clock.
  PBDIV = 1: APBx clock equals main clock divided by $2^{(PBSEL+1)}$.

Note that if PBDIV is written to 0, PBSEL should also be written to 0 to ensure correct operation.

Also note that writing this register clears SR.CKRDY. The register must not be re-written until SR.CKRDY goes high.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.4 Clock Mask

Name: CPUMASK/HSBMASK/PBAMASK/PBBMASK/PBCMASK/PBDMASK

Access Type: Read/Write

Offset: 0x020-0x034

Reset Value: 0x00000001/0x000001E2/0x00000000/0x00000001/0x0000001F/0x0000003F-

31	30	29	28	27	26	25	24
MASK[31:24]							

23	22	21	20	19	18	17	16
MASK[23:16]							

15	14	13	12	11	10	9	8
MASK[15:8]							

7	6	5	4	3	2	1	0
MASK[7:0]							

- **MASK: Clock Mask**
  If bit n is cleared, the clock for module n is stopped. If bit n is set, the clock for module n is enabled according to the current power mode. The number of implemented bits in each mask register, as well as which module clock is controlled by each bit, is shown in Table 10-5. After reset, some modules are enabled by default (shown as gray cell).

Table 10-5. Maskable Module Clocks in ATSAM4L.

Bit	CPUMASK	HSBMASK	PBAMASK	PBBMASK	PBCMASK	PBDMASK
0	OCD	PDCA	IISC	FLASHCALW	PM	BPM
1	-	FLASHCALW	SPI	HRAMC1	CHIPID	BSCIF
2	-	FLASHCALW (picoCache RAM)	TC0	HMATRIX	SCIF	AST
3	-	USBC	TC1	PDCA	FREQM	WDT
4	-	CRCCU	TWIM0	CRCCU	GPIO	EIC
5	-	APBA bridge	TWIS0	USBC		PICOUART
6	-	APBB bridge	TWIM1	PEVC		-
7	-	APBC bridge	TWIS1	-		-
8	-	APBD bridge	USART0	-		-
9	-	AESA	USART1	-		-
10	-	-	USART2	-		-
11	-	-	USART3	-		-
12	-	-	ADCIFE	-	-	-

Table 10-5. Maskable Module Clocks in ATSAM4L.

Bit	CPUMASK	HSBMASK	PBAMASK	PBBMASK	PBCMASK	PBDMASK
13	-	-	DACC	-	-	-
14	-	-	ACIFC	-	-	-
15	-	-	GLOC	-	-	-
16	-	-	ABDACB	-	-	-
17	-	-	TRNG	-	-	-
18	-	-	PARC	-	-	-
19	-	-	CATB	-	-	-
20	-	-	-	-	-	-
21	-	-	TWIM2	-	-	-
22	-	-	TWIM3	-	-	-
23	-	-	LCDCA	-	-	-
24	-	-	-	-	-	-
25	-	-	-	-	-	-
31:26	-	-	-	-	-	-

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.5 Divided Clock Mask

Name: PBADIVMASK

Access Type: Read/Write

Offset: 0x040

Reset Value: 0x0000007F

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-		-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	MASK[6:0]						

- **MASK: Clock Mask**
  If bit n is written to zero, the clock divided by $2^{(n+1)}$ is stopped. If bit n is written to one, the clock divided by $2^{(n+1)}$ is enabled according to the current power mode. Table 10-6 shows what clocks are affected by the different MASK bits.

Table 10-6. Divided Clock Mask

Bit	USART0	USART1	USART2	USART3	TC0	TC1
0	-	-	-	-	TIMER_CLOCK2	TIMER_CLOCK2
1	-	-	-	-	-	-
2	CLK_USART/ DIV	CLK_USART/ DIV	CLK_USART/ DIV	CLK_USART/ DIV	TIMER_CLOCK3	TIMER_CLOCK3
3	-	-	-	-	-	-
4	-	-	-	-	TIMER_CLOCK4	TIMER_CLOCK4
5	-	-	-	-	-	-
6	-	-	-	-	TIMER_CLOCK5	TIMER_CLOCK5

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.6 Clock Failure Detector Control Register

Name: CFDCTRL

Access Type: Read/Write

Offset: 0x054

Reset Value: 0x00000000

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

- **SFV: Store Final Value**
  0: The register is read/write.
  1: The register is read-only, to protect against further accidental writes.
- **CFDEN: Clock Failure Detection Enable**
  0: Clock Failure Detector is disabled.
  1: Clock Failure Detector is enabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.7 PM Unlock Register

Name: UNLOCK

Access Type: Write-Only

Offset: 0x058

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	ADDR[9:8]	

7	6	5	4	3	2	1	0
ADDR[7:0]							

To unlock a write protected register, first write to the UNLOCK register with the address of the register to unlock in the ADDR field and 0xAA in the KEY field. Then, in the next APB access write to the register specified in the ADDR field.

- **KEY: Unlock Key**
  Write this bit field to 0xAA to enable unlock.
- **ADDR: Unlock Address**
  Write the address of the register to unlock to this field.

10.7.8 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x0C0

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	WAKE
7	6	5	4	3	2	1	0
-	-	CKRDY	-	-	-	-	CFD

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

10.7.9 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x0C4

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	WAKE
7	6	5	4	3	2	1	0
-	-	CKRDY	-	-	-	-	CFD

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

10.7.10 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x0C8

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	WAKE
7	6	5	4	3	2	1	0
-	-	CKRDY	-	-	-	-	CFD

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
This bit is cleared when the corresponding bit in IDR is written to one.
This bit is set when the corresponding bit in IER is written to one.

10.7.11 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x0CC

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	WAKE
7	6	5	4	3	2	1	0
-	-	CKRDY	-	-	-	-	CFD

0: The corresponding interrupt is cleared.
1: The corresponding interrupt is pending.
This bit is cleared when the corresponding bit in ICR is written to one.
This bit is set when the corresponding interrupt occurs.

10.7.12 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x0D0

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	WAKE
7	6	5	4	3	2	1	0
-	-	CKRDY	-	-	-	-	CFD

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR.

10.7.13 Status Register

Name: SR

Access Type: Read-only

Offset: 0x0D4

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	WAKE
7	6	5	4	3	2	1	0
-	-	CKRDY	-	-	-	-	CFD

- **AE: Access Error**
  - 0: No access error has occurred.
  - 1: A write to lock protected register without unlocking it has occurred.
- **WAKE: Wake up**
  - 0: No wakeup has occurred.
  - 1: A wakeup event has occurred. Refer the WCAUSE register to know the wakeup source.
- **CKRDY: Clock Ready**
  - 0: The CKSEL register has been written, and the new clock setting is not yet effective.
  - 1: The synchronous clocks have frequencies as indicated in the CKSEL register.
- **CFD: Clock Failure Detected**
  - 0: Main clock is running correctly.
  - 1: Failure on main clock detected. Main clock reverting to RCSYS.

10.7.14 Peripheral Power Control Register

Name: PPCR

Access Type: Read/Write

Offset: 0x160

Reset Value: 0x000001FE

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

Table 10-7. Peripheral Power Control

Bit	Name
0	RSTPUN
1	CATBRCMASK
2	ACIFCRCMASK
3	ASTRCMASK
4	TWIS0RCMASK
5	TWIS1RCMASK
6	PEVCRCMASK
7	ADCIFERCMASK
8	VREGRCMASK
9	FWBGREF
10	FWBOD18
31:11	-

- **FWBOD18: Flash Wait BOD18**
  0: At powerup, flash waits for the BOD18 to be ready and calibrated to go back to normal mode.
  1: At power up, flash does not wait for the BOD18 to be ready.
- **FWBGREF: Flash Wait BGREF**
  0: When waking up, flash controller waits for the main bandgap to be ready to go back to normal mode.
  1: When waking up, flash controller does not wait for the main bandgap to be ready.

- **VREGRCMASK: VREG Request Clock Mask**
  0: VREG Request Clock is disabled
  1: VREG Request Clock is enabled
- **ADCIFERCMASK: ADCIFE Request Clock Mask**
  0: ADCIFE Request Clock is disabled
  1: ADCIFE Request Clock is enabled
- **PEVCRCMASK: PEVC Request Clock Mask**
  0: PEVC Request Clock is disabled
  1: PEVC Request Clock is enabled
- **TWIS1RCMASK: TWIS1 Request Clock Mask**
  0: TWIS1 Request Clock is disabled
  1: TWIS1 Request Clock is enabled
- **TWIS0RCMASK: TWIS0 Request Clock Mask**
  0: TWIS0 Request Clock is disabled
  1: TWIS0 Request Clock is enabled
- **ASTRCMASK: AST Request Clock Mask**
  0: AST Request Clock is disabled
  1: AST Request Clock is enabled
- **ACIFCRCMASK: ACIFC Request Clock Mask**
  0: ACIFC Request Clock is disabled
  1: ACIFC Request Clock is enabled
- **CATBRCMASK: CAT Request Clock Mask**
  0: CAT Request Clock is disabled
  1: CAT Request Clock is enabled
- **RSTPUN: Reset Pullup**
  0: Pullup on the external reset pin is disabled.
  1: Pullup on the external reset pin is enabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 10.7.7 “PM Unlock Register” on page 124 for details.

10.7.15 Reset Cause

Name: RCAUSE

Access Type: Read-only

Offset: 0x180

Reset Value: Latest Reset Source

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	BOD33	-	-	POR33	-	OCDRST

7	6	5	4	3	2	1	0
-	BKUP	-	-	WDT	EXT	BOD	POR

- **BOD33: Brown-out 3.3V Reset**
  This bit is set when the last reset was due to the supply voltage being lower than the BOD 3.3V (BOD33) threshold level.
- **POR33: Power-on Reset**
  This bit is set when the last reset was due to the I/O voltage being lower than the POR33 threshold.
- **OCDRST: OCD Reset**
  This bit is set when the last reset was due to the SYSRESETREQ bit in the AIRCR register of the CPU having been written to one.
- **BKUP: Backup reset**
  This bit is set when the last reset was due to a wake up from the Backup mode.
- **WDT: Watchdog Reset**
  This bit is set when the last reset was due to a watchdog time-out.
- **EXT: External Reset Pin**
  This bit is set when the last reset was due to the RESET_N pin being pulled low.
- **BOD: Brown-out Reset**
  This bit is set when the last reset was due to the core supply voltage being lower than the brown-out threshold level.
- **POR: Power-on Reset**
  This bit is set when the last reset was due to the core supply voltage being lower than the power-on threshold level,

10.7.16 Wake Cause Register

Name: WCAUSE

Access Type: Read-only

Offset: 0x184

Reset Value: Latest Wake Source

31	30	29	28	27	26	25	24
WCAUSE[31:24]							

23	22	21	20	19	18	17	16
WCAUSE[23:16]							

15	14	13	12	11	10	9	8
WCAUSE[15:8]							

7	6	5	4	3	2	1	0
WCAUSE[7:0]							

A bit in this register is set on wake up caused by the peripheral referred to in Table 10-8 on page 134.

Table 10-8. Wake Cause

Bit	Wake Cause
0	TWI Slave 0
1	TWI Slave 1
2	USBC
3	PSOK
4	BOD18 IRQ
5	BOD33 IRQ
6	PICOUART
7	LCDCA
15:8	-
16	EIC
17	AST
31:18	-

10.7.17 Asynchronous Wake Up Enable Register

Name: AWEN

Access Type: Read/Write

Offset: 0x188

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AWEN[31:24]							

23	22	21	20	19	18	17	16
AWEN[23:16]							

15	14	13	12	11	10	9	8
AWEN[15:8]							

7	6	5	4	3	2	1	0
AWEN[7:0]							

Each bit in this register corresponds to an asynchronous wake up, according to Table 10-9 on page 135.

0: The corresponding wake up is disabled.

1: The corresponding wake up is enabled

Table 10-9. Asynchronous Wake Up

Bit	Asynchronous Wake Up
0	TWI Slave 0
1	TWI Slave 1
2	USBC
3	PSOK
4	BOD18 IRQ
5	BOD33 IRQ
6	PICOUART
7	LCDCA
31:6	-

10.7.18 Fast Sleep Register

Name: FASTSLEEP

Access Type: Read/Write

Offset: 0x194

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	DFLL
23	22	21	20	19	18	17	16
-	-	-	FASTRCOSC				
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	PLL
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OSC

Each bit in this register corresponds to a clock source set as the main clock just before entering power save mode and just after wake-up to make the wakeup time faster.

0: The corresponding clock source is not set as the main clock after wake-up.

1: The corresponding clock source is set as the main clock after wake-up.

- **OSC: Oscillator**
- **PLL: PLL**
- **FASTRCOSC:**

  00001: RC80

  00010: RCFAST

  00100: RC1M

  01000: Reserved

  10000: Reserved
- **DFLL: DFLL**

10.7.19 Configuration Register

Name: CONFIG

Access Type: Read-Only

Offset: 0x3F8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
HSBPEVC	-	-	-	PBD	PBC	PBB	PBA

This register shows the configuration of the PM.

- **HSBPEVC: HSB PEVC Clock Implemented**
  0: HSBPEVC not implemented.
  1: HSBPEVC implemented.
- **PBD: APBD Implemented**
  0: APBD not implemented.
  1: APBD implemented.
- **PBC: APBC Implemented**
  0: APBC not implemented.
  1: APBC implemented.
- **PBB: APBB Implemented**
  0: APBB not implemented.
  1: APBB implemented.
- **PBA: APBA Implemented**
  0: APBA not implemented.
  1: APBA implemented.

10.7.20 Version Register

Name: VERSION

Access Type: Read-Only

Offset: 0x3FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

10.8 Module Configuration

The specific configuration for each PM instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to the “Synchronous Clocks”, “Peripheral Clock Masking” sections for details.

Table 10-10. Power Manager Clocks

Clock Name	Description
CLK_PM	Clock for the PM bus interface

Table 10-11. Register Reset Values

Register	Reset Value
VERSION	0x00000441

Table 10-12. Effect of the Different Reset Events

	POR reset	BOD33 reset	BOD18 reset	External Reset	WDT Reset	OCD Reset	Backup Reset
AST 32 kHz XOSC oscillator 32 kHz RC oscillator	Y	N	N	N	N	N	N
RC Oscillators Calibration register	Y	N	N	N	N	N	Y
Watchdog registers	Y	Y	Y	Y	N	Y	N
BOD33 control register	Y	N	Y	Y	Y	Y	N
BOD18 control register	Y	Y	N	Y	Y	Y	N
Backup domain except AST, WDT and 32 kHz oscillators	Y	Y	Y	Y	Y	Y	N
Core domain excluding OCD	Y	Y	Y	Y	Y	Y	Y
OCD system and OCD registers	Y	Y	Y	Y	N	N	Y

11. Backup Power Manager (BPM)

Rev: 1.2.0.5

11.1 Features

- **Supports the Power Scaling Technique**
- **Controls the Power Save Modes**
- **Manages I/O lines pin muxing for Backup mode**
- **Manages I/O lines retention in Backup mode**
- **Stores the wake up source from Backup mode**

11.2 Overview

The Backup Power Manager (BPM) located in the Backup domain is an extension of the Power Manager (PM) module.

To optimize power consumption, the BPM supports the Power Scaling Technique. The BPM allows changing the Power Scaling configuration in both RUN and Power Save Mode.

The BPM allows the user to choose between different Power Save Modes depending on application requirements. It controls the way the microcontroller enters or exits a Power Save Mode.

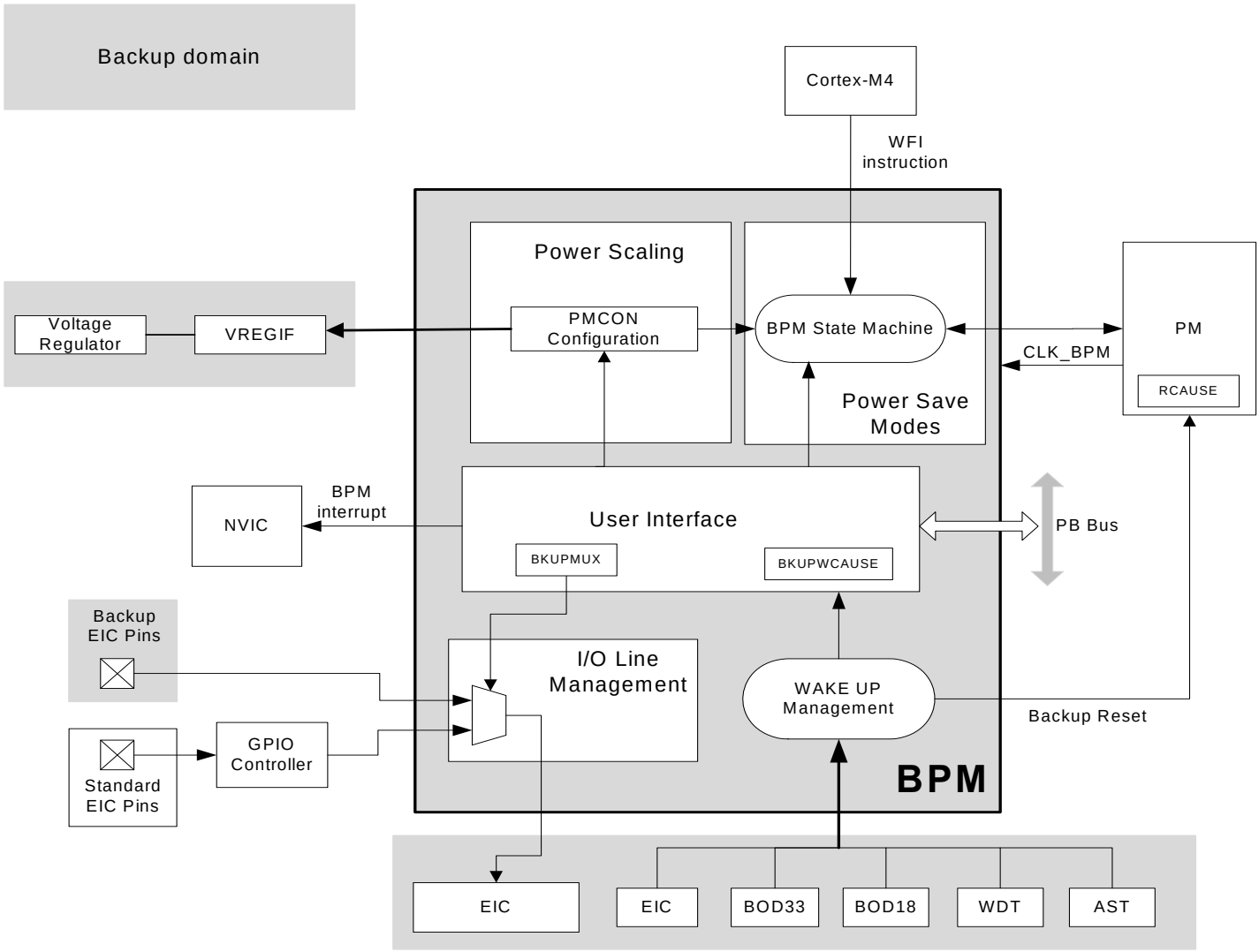
In BACKUP mode, some I/O lines are dedicated to system wakeup. The BPM takes control of these I/O line pins muxing needed to wake up the system.

The configuration of the I/O lines is kept in all Power Save Modes. Exiting the BACKUP mode will reset the Core domain and by default will put the I/O lines in their reset state. The BPM allows retaining the I/O line state when exiting the BACKUP mode, preventing I/O lines from toggling during wake up.

The BPM flags the wake up source when the BACKUP mode is exited.

11.3 Block Diagram

Figure 11-1. BPM Block Diagram



11.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

11.4.1 Clocks

The clock for the BPM bus interface (CLK_BPM) is generated by the Power Manager. This clock is enabled at reset, and can be disabled in the Power Manager.

11.4.2 Interrupts

The BPM interrupt line is connected to the NVIC. Using the BPM interrupt requires the NVIC to be programmed first.

11.4.3 Debug Operation

If a BACKUP mode is requested by the system while in debug mode, the core domain is kept to those in RUN mode, and the debug modules are kept running to allow the debugger to access internal registers.

When exiting the BACKUP mode upon a wakeup event, all the core domain is reset but the debug session. It allows the user to keep using its current debug session.

Note: Hot plugging in backup mode is not supported.

11.5 Functional Description

11.5.1 Power Scaling

The power scaling feature is programmed thanks to the PMCON register where the user can select the Power Scaling Configuration (PS) and the Power Scaling Change Request (PSCREQ).

When exiting the BACKUP mode, the Power Scaling Configuration field (PMCON.PS) is unchanged. It ensures that the system is running in the same power scaling configuration.

When a reset occurs (See ”Reset Description” on page 113.), the PMCON.PS field is reset to its default value, and then, the system goes back to the reset power configuration mode (RUN0 mode).

Refer to the Low Power Technique chapter (See ”Power Scaling” on page 57.) to understand the sequence to apply.

11.5.2 Power Save Modes

The Power Save Mode feature is supported thanks to the PMCON register where the user can select the SLEEP mode configuration (SLEEP), the BACKUP mode (BKUP) and the RETENTION mode (RET). The SLEEPDEEP bit located in the System Control Register in the Cortex M4 should also be used to manage the different Power Save Modes.

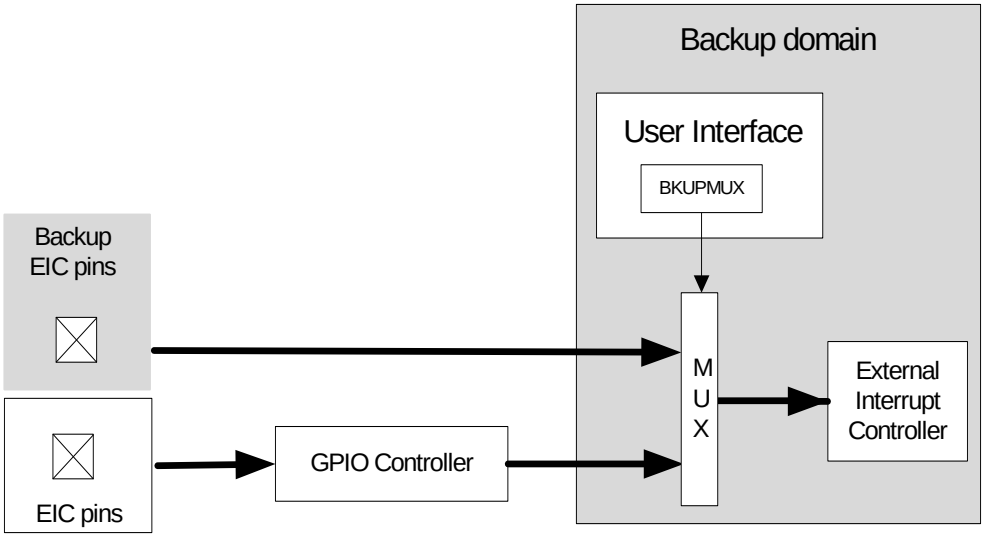
The selected power save mode is entered after a WFI instruction.

Refer to the Low Power Technique chapter (See ”Power Scaling” on page 57.) to have more details.

11.5.3 I/O Lines Pin Muxing in Backup Mode

In all modes including the BACKUP mode, a subset of pins (external interrupts) can be directly routed to a list of peripheral function bypassing the GPIO controller.

Figure 11-2. I/O Lines Pin Muxing Diagram



To enable the backup pin muxing, the user should set a one to the corresponding bit of the Backup Pin Muxing register (BKUPPMUX) in the BPM. The backup alternate function overrides the GPIO function.

To allow the system to be waken up by an external interrupt pin in backup mode, the backup pin muxing should be enabled.

11.5.4 I/O Lines Retention in Backup Mode

When the device exits the BACKUP mode by a wake up source, the I/O line configuration can be released or stretched, based on the RET bit in the IORET register of the BPM:

- if the BPM.IORET.RET bit is set to 0: the I/O lines are released and then driven by the reset value of the GPIO controller module.
- if the BPM.IORET.RET bit is set to 1: the Backup configuration of I/O lines are kept until the user sets the RET bit to 0. It allows the I/O lines to be retained until the user has programmed the GPIO controller to its correct value, preventing I/O lines from toggling during wake up.

11.5.5 Wakeup From BACKUP Mode

Any enabled asynchronous sources, located in the backup domain, can wake the system up.

The user should first enable the wakeup source by programming the Backup Wake up Enable Register (BKUPWEN).

The wakeup source is stored inside the Backup Wake up Cause Register. The user can read it once the BACKUP mode is exited and the RUN mode is back.

See ”BACKUP Mode” on page 55. in the Low Power Technique chapter to have more details.

Note: If EIC is used to wakeup the system, the Backup Pin Muxing bit must be set (See ”I/O Lines Pin Muxing in Backup Mode” on page 142.)

11.5.6 Precautions When Entering Power Save Mode.

Modules communicating with external circuits should be disabled before entering a Power Save Mode that will stop the module operation. This prevents erratic behavior when entering or exiting Power Save Mode. Refer to the relevant module documentation for recommended actions.

Communication between the synchronous clock domains is disturbed when entering and exiting Power Save Modes. This means that bus transactions are not allowed between clock domains affected by the Power Save Mode. The system may hang if the bus clocks are stopped in the middle of a bus transaction.

When entering a Power Save Mode where AHB clocks are stopped, all AHB masters must be stopped before entering the Power Save Mode. Also, if there is a chance that any APB write operations are incomplete, the CPU should perform a read operation from any register on the APB bus before executing the WFI instruction. This will stall the CPU while waiting for any pending APB operations to complete.

Note: All these recommendations also apply when changing the power scaling as the system is halted.

11.5.7 Interrupts

The BPM has a number of interrupts:

- AE: Access Error, set if a lock protected register is written without first being unlocked.
- PSOK: Power Scaling configuration OK, set if the requested Power configuration is ready (regulator mode ready, regulator output voltage as expected).

11.6 User Interface

Table 11-1. BPM Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0000	Interrupt Enable Register	IER	Write-only	0x00000000
0x0004	Interrupt Disable Register	IDR	Write-only	0x00000000
0x0008	Interrupt Mask Register	IMR	Read-only	0x00000000
0x000C	Interrupt Status Register	ISR	Read-only	0x00000000
0x0010	Interrupt Clear Register	ICR	Write-only	0x00000000
0x0014	Status Register	SR	Read-only	0x00000000
0x0018	Unlock Register	UNLOCK	Read/Write	0x00000000
0x001C	Power Mode Control Register	PMCON	Read/Write	0x00000000
0x0028	Backup Wake up Cause Register	BKUPWCAUSE	Read-only	0x00000000
0x002C	Backup Wake up Enable Register	BKUPWEN	Read/Write	0x00000000
0x0030	Backup Pin Muxing Register	BKUPPMUX	Read/Write	0x00000000
0x0034	Input Output Retention Register	IORET	Read/Write	0x00000000
0x00FC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value for this register is device specific. Refer to the Module Configuration section at the end of this chapter.

11.6.1 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x0000

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

11.6.2 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x0004

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

11.6.3 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x0008

Reset Value: 0x00000000

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

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
This bit is cleared when the corresponding bit in IDR is written to one.
This bit is set when the corresponding bit in IER is written to one.

11.6.4 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x000C

Reset Value: 0x00000000

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

0: The corresponding interrupt is cleared.
1: The corresponding interrupt is pending.
This bit is cleared when the corresponding bit in ICR is written to one.
This bit is set when the corresponding interrupt occurs.

11.6.5 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x0010

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR.

11.6.6 Status Register

Name: SR

Access Type: Read-only

Offset: 0x0014

Reset Value: 0x00000000

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

- **AE: Access Error**
  0: No access error has occured.
  1: A write to lock protected register without unlocking it has occured.
- **PSOK: Power Scaling OK**
  0: The Power Scaling configuration is not yet ready.
  1: The current Power Scaling configuration is ready. The regulator mode is ready, the regulator output voltage has reached the expected value, the flash is ready.

11.6.7 Unlock Register

Name: UNLOCK

Access Type: Write-Only

Offset: 0x018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	ADDR[9:8]	

7	6	5	4	3	2	1	0
ADDR[7:0]							

To unlock a write protected register, first write to the UNLOCK register with the address of the register to unlock in the ADDR field and 0xAA in the KEY field. Then, in the next PB access write to the register specified in the ADDR field.

- **KEY: Unlock Key**
  Write this bit field to 0xAA to enable unlock.
- **ADDR: Unlock Address**
  Write the address of the register to unlock to this field.

11.6.8 Power Mode Control Register

Name: PMCON

Access Type: Read/Write

Offset: 0x001C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	FASTWKUP

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	CK32S

15	14	13	12	11	10	9	8
-		SLEEP		-	-	RET	BKUP

7	6	5	4	3	2	1	0
-	-	-	-		PSCREQ	PS	

- **FASTWKUP: Fast Wakeup**
  0: Normal wakeup time for analog modules.
  1: Fast wakeup time for analog modules. To have an optimal fast wakeup, this is also required to configure the PM.FASTSLEEP register to wakeup the system on a Fast RC oscillator clock source. If the flash high speed mode is enabled, the fast wakeup has no effect.
  During the wakeup period, the FLASHCALW module is forced to operate in wait state 1 until the BPM.SR.PSOK bit is one.

- **CK32S: 32kHz-1kHz Clock Source Selection**
  0: the 32kHz and 1kHz clock sources are from the OSC32K.
  1: the 32kHz and 1kHz clock sources are from the RC32K.

- **SLEEP: SLEEP mode Configuration**

Table 11-2. SLEEP mode Configuration

SLEEP	Description
0	The CPU clock is stopped.
1	The CPU and AHB clocks are stopped.
2	The CPU, AHB, PB and GCLK clocks are stopped. Clock sources (OSC, Fast RC Oscillators, PLL, DFLL) are still running.
3	The CPU, AHB, PB, GCLK clocks and clock sources (OSC, RCFast, PLL, DFLL) are stopped. RCSYS is still running. RC32K or OSC32K are still running if enabled.

- **RET: RETENTION Mode**
  0: the Power Save Mode will not be the RETENTION mode.

1: the Power Save Mode will be the RETENTION mode if the SCR.SLEEPDEEP bit is set to 1 and the Backup Mode is set to zero.

- **BKUP: BACKUP Mode**
  0: the Power Save Mode will not be the BACKUP mode.
  1: the Power Save Mode will be the BACKUP mode if the SCR.SLEEPDEEP bit is set to 1.
- **PSCREQ: Power Scaling Change Request**
  0: A new power scaling is not requested.
  1: A new power scaling is requested.
  This bit is cleared by hardware after the completion of the power scaling change.
- **PS: Power Scaling Configuration Value**
  Refer to Section 6.1.5 ”Power Save Mode Summary Table” on page 57 and to Figure 6-1 on page 52
  This field is reset to its default value when exiting the backup mode.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 11.6.7 ”Unlock Register” on page 152 for details.

11.6.9 Backup Wake up Cause Register

Name: BKUPWCAUSE

Access Type: Read-only

Offset: 0x0028

Reset Value: Latest Wake up Source from Backup mode

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

A bit in this register is set when the system is waking up from the Backup mode and the wake up source is caused by the peripheral referred to in Table 11-3 on page 155.

Table 11-3. Backup Wake up Cause

Bit	Wake Cause
0	EIC
1	AST
2	WDT interrupt
3	BOD33 interrupt
4	BOD18 interrupt
5	PICOUART interrupt
31:6	-

11.6.10 Backup Wake up Enable Register

Name: BKUPWEN

Access Type: Read/Write

Offset: 0x002C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
BKUPWEN[31:24]							

23	22	21	20	19	18	17	16
BKUPWEN[23:16]							

15	14	13	12	11	10	9	8
BKUPWEN[15:8]							

7	6	5	4	3	2	1	0
BKUPWEN[7:0]							

Each bit in this register corresponds to a wake up cause from the Backup mode, according to Table 11-3 on page 155.

0: The corresponding wake up source is disabled.

1: The corresponding wake up source is enabled.

Table 11-4. Asynchronous Wake Up

Bit	Asynchronous Wake Up
0	EICEN
1	ASTEN
2	WDTEN
3	BOD33EN
4	BOD18EN
5	PICOUARTEN
31:6	-

11.6.11 Backup Pin Muxing Register

Name: BKUPPMUX

Access Type: Read/Write

Offset: 0x0030

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	BKUPPMUX[8]

7	6	5	4	3	2	1	0
BKUPPMUX[7:0]							

- **BKUPPMUX: Backup Pin Muxing**

  Map a peripheral function required in Backup mode to a dedicated pad.

  Each bit in this register corresponds to an alternate function associated to a pin, according to Table 11-5 on page 157.

  0: The corresponding backup pin is not selected.

  1: The corresponding backup pin is selected.

Table 11-5. Backup Pin Muxing

Bit	Backup Pin Name	Alternate function
0	**PB01**	EIC[0]
1	**PA06**	EIC[1]
2	**PA04**	EIC[2]
3	**PA05**	EIC[3]
4	**PA07**	EIC[4]
5	**PC03**	EIC[5]
6	**PC04**	EIC[6]
7	**PC05**	EIC[7]
8	**PC06**	EIC[8]
31-9	-	-

11.6.12 Input Output Retention Register

Name: IORET

Access Type: Read/Write

Offset: 0x0034

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	RET

- **RET: Retention on I/O lines after waking up from the BACKUP mode**
  0: I/O lines are not held after waking up from the BACKUP mode.
  1: I/O lines are held after waking up from the BACKUP mode, until RET is written to 0.

11.6.13 Version Register

Name: VERSION

Access Type: Read-Only

Offset: 0x00FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

11.7 Module Configuration

The specific configuration for each BPM instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to the “Synchronous Clocks”, “Peripheral Clock Masking” and “Power Save Modes” sections for details.

Table 11-6. Power Manager Clocks

Clock Name	Description
CLK_BPM	Clock for the BPM bus interface

Table 11-7. Register Reset Values

Register	Reset Value
VERSION	0x00000120

12. Backup System Control Interface (BSCIF)

Rev: 1.0.0.0

12.1 Features

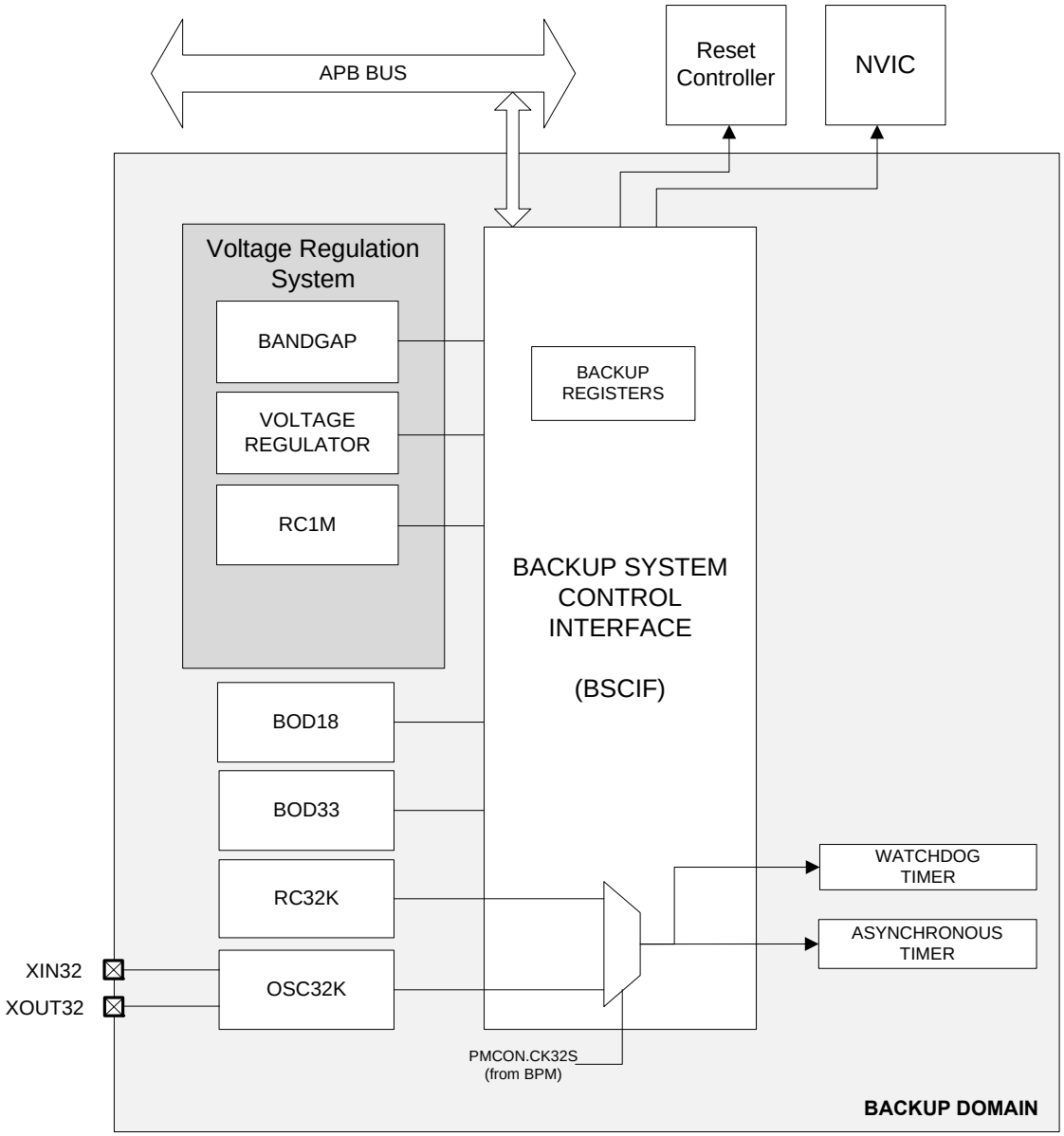
- **Supports 32KHz ultra-low-power oscillator (OSC32K)**
- **Supports 32kHz RC oscillator (RC32K)**
- **Supports 1MHz RC oscillator (RC1M)**
- **Controls Brown-out detectors (BOD18 and BOD33)**
- **Controls the Voltage Regulation System**
- **Four 32-bit general-purpose backup registers**

12.2 Overview

The Backup System Control Interface (BSCIF) controls the oscillators, BODs, Voltage Regulation System and Backup Registers. It is located in Backup domain.

12.3 Block Diagram

Figure 12-1. BSCIF Block Diagram



12.4 I/O Lines Description

Table 12-1. I/O Lines Description

Pin Name	Pin Description	Type
XIN32	Crystal 32 Input	Analog/Digital
XOUT32	Crystal 32 Output	Analog

12.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

12.5.1 Power Management

The 32 kHz oscillators (RC32K and OSC32K) are not turned off in Power Save modes. BODs are turned off in some Power Save modes and turned automatically on when the device wakes up. The Voltage Regulation System is controlled by the BPM, refer to Section 11. “Backup Power Manager (BPM)” on page 140 for details.

12.5.2 Clocks

The BSCIF controls the 32kHz oscillators as well as the 1MHz RC in the device. The oscillators can be used as source for the CPU and peripherals, selection of source is done in the Power Manager. The oscillator can also be used as source for generic clocks. Device modules using a 32 kHz clock must choose between the 32 kHz crystal oscillator and the RC 32 kHz oscillator for the clock source. Setting the CK32S bit to one in the PMCON register (PMCON.CK32S) will select the RC32K oscillator as the 32 kHz clock source, and leaving it to zero will select the 32 kHz crystal oscillator output.

12.5.3 Interrupts

The BSCIF interrupt request line is connected to the NVIC. Using the BSCIF interrupt requires the NVIC to be programmed first.

12.5.4 Debug Operation

The BSCIF does not interact with debug operations.

12.6 Functional Description

12.6.1 32KHz Oscillator (OSC32K) Operation

Rev: 2.0.0.0

The 32KHz oscillator operates as the standard main oscillator. The 32KHz oscillator can be used as source clock for the Asynchronous Timer (AST), the Watchdog Timer (WDT) and the Brown Out Detectors (BODs) when they are used in sampling mode. The 32KHz oscillator can also be used as source for the generic clocks.

The oscillator is disabled by default after reset.

The oscillator is enabled by writing a one to the OSC32 Enable bit in the 32KHz Oscillator Control Register (OSCCTRL32.OSC32EN). The oscillator is disabled by writing a zero to the OSC32EN bit, while keeping the other bits unchanged. Writing to OSC32EN while also writing to other bits may result in unpredictable behavior. Operation mode (external clock or crystal) is selected by writing to the Oscillator Mode bit in OSCCTRL32 (OSCCTRL32.MODE). The oscillator is an ultra low power design and remains enabled in all Power Save Modes.

The start-up time of the 32KHz oscillator is selected by writing to the Oscillator Start-up Time field in the OSCCTRL32 register (OSCCTRL32.STARTUP). The BSCIF masks the oscillator output during the start-up time, to ensure that no unstable clock cycles propagate to the digital logic.

The OSC32 Ready bit in the Power and Clock Status Register (PCLKSR.OSC32RDY) is set when the oscillator is stable and ready to be used as clock source. An interrupt can be gener-

ated on a zero-to-one transition on PCLKSR.OSC32RDY if the OSC32RDY bit in the Interrupt Mask Register (IMR.OSC32RDY) is set. This bit is set by writing a one to the corresponding bit in the Interrupt Enable Register (IER.OSC32RDY).

As a crystal oscillator usually requires a very long start-up time (up to 1 second), the 32KHz oscillator will keep running across resets, except a Power-on Reset (POR).

The current driven into the crystal can be adjusted according to the component specification requirements via the SELCURR bitfield in the OSC32CTRL register (OSC32CTRL.SELCURR).

The 32KHz oscillator also has a 1KHz output (CLK_1K). This is enabled by writing a one to the Enable 1KHz output bit in the OSCCTRL32 register (OSCCTRL32.EN1K). If the 32KHz output clock is not needed when the CLK_1K is enabled, this can be disabled by writing a zero to the Enable 32KHz output bit in the OSCCTRL32 register (OSCCTRL32.EN32K). OSCCTRL32.EN32K is set after a POR. The CLK_1K output is only usable when XIN32 is connected to a crystal, and not when an external digital clock is applied on XIN32.

12.6.2 32 KHz RC Oscillator (RC32K)

Rev: 1.0.0.1

The RC32K oscillator provides a tunable, low speed, low-power clock source. The RC32K is enabled by default as a system clock, except in Static mode. The RC32K is enabled as a 32 KHz Generic clock source by writing a one to the enable bit in the RC32K Control Register (RC32KCR.EN). It is disabled by writing a zero to RC32KCR.EN. The RC32K is also available for direct use as a 32 KHz clock, enabled by writing a one to the Enable 32KHz bit (RC32KCR.EN32K). A 1 KHz clock output can be enabled by writing a one to the Enable 1kHz bit (RC32KCR.EN1K).

The RC32K oscillator supports temperature compensation, which can be enabled for a more stable frequency. The temperature compensation is enabled by writing a one to the Temperature Compensation Enable bit (RC32KCR.TCEN), and disabled by writing a zero.

The frequency of the RC32K oscillator is controlled by two values in the RC32K Tuning Register RC32KTUNE.COARSE and RC32KTUNE.FINE. COARSE is loaded by the fuses, and compensates for process variation, while FINE is used to keep the frequency accurate over the temperature range. The COARSE value can be overridden by the user by writing to the COARSE field. The FINE value can be tuned in two different modes, selected by the RC32KCR.MODE bit where a zero is open loop mode and a one is closed loop mode.

In open loop mode the user can tune the frequency by writing to the RC32KTUNE register. When writing to RC32KTUNE, the user must wait for the PCLKSR.RC32KRDY bit to go high before the value has been committed to the oscillator. See Section 42.7 “Oscillator Characteristics” on page 1143 for how FINE and COARSE affects the frequency.

In closed loop mode the RC32K automatically tunes FINE to synchronize the RC32K oscillator with an external 32768 Hz reference. The reference source is selected by the Reference Select bit (RC32KCR.REF). In closed loop mode, the RC32K does a binary search on the FINE field to converge towards the reference. When the RC32K has converged on the reference frequency, the RC32K lock bit (PCLKSR.RC32KLOCK) is set, and the RC32K continues to measure and track the reference frequency. If the user switches from closed loop mode to open loop mode and back to closed loop again, RC32K will restart the synchronization procedure, disregarding previous locks.

If the reference clock stops when the RC32K is in closed loop mode, the RC32K Reference Error bit (PCLKSR.RC32KREFE) is set, and the RC32K stays in closed loop mode, keeping the current FINE value. When the reference clock starts again, the RC32K automatically resumes the tracking of the reference clock.

In closed loop mode, if the correct frequency is out of range for the FINE value, it will saturate due to an incorrect COARSE setting. The RC32K Saturation bit (PCLKSR.RC32KSAT) is set and the user can read the current FINE value to determine further action. Note that RC32KTUNE can only be written to when closed loop mode is disabled. When disabling closed loop mode, wait for the RC32K Ready bit (PCLKSR.RC32KRDY) to be set before writing to RC32KTUNE, and then wait for it to be set again before going back to closed loop mode. The FINE field will always read as zeroes when in closed loop mode.

If the range of FINE is not sufficient to obtain the correct frequency, RC32KTUNE can be modified by the user with a new COARSE value to change the FINE offset. The user should select a FINE value of 0x20 before enabling closed loop mode.

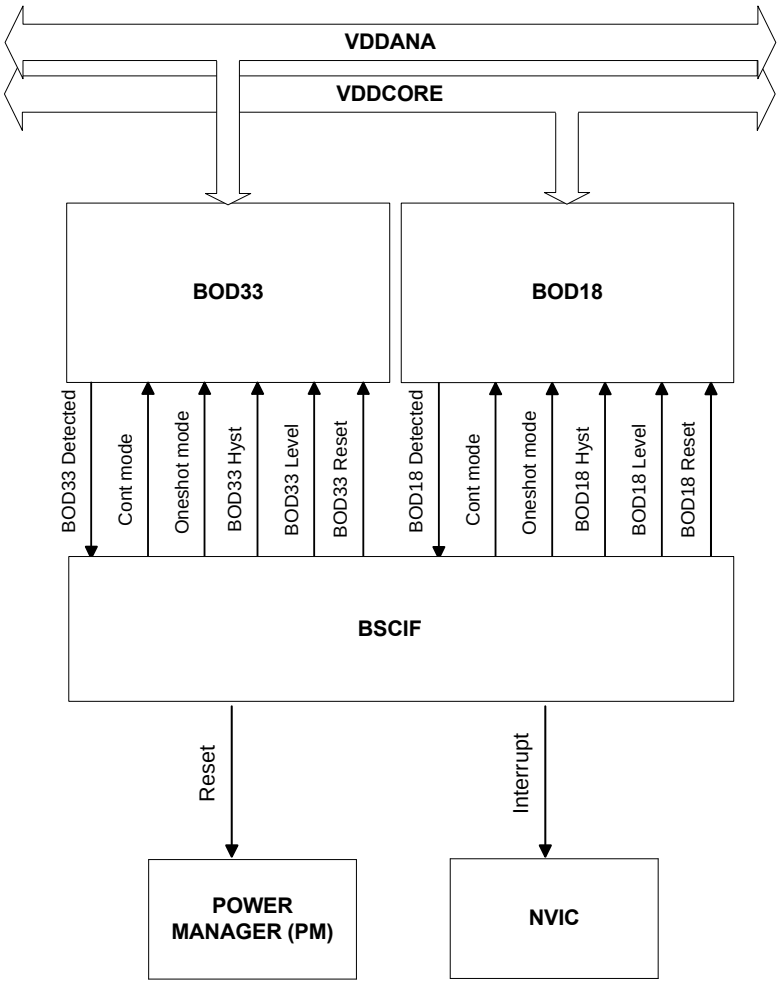
When the RC32K is enabled with either EN, EN32K, or EN1K in the RC32KCR register, the oscillator is kept running during all Power Save Modes.

12.6.3 Brow-Out Detector (BOD) Operation

Rev: 1.1.0.0

The Brown-out Detectors BOD18 and BOD 33 monitors the VDDCORE and VDDIO supplies respectively, and compares the voltage with the brown-out trigger level, as set in the BOD18/33LEVEL register. The BOD is disabled by default, but can be enabled either by software or by flash fuses. The Brown-out Detector can generate either an interrupt or a reset when the supply voltage is below the Brown-out trigger level, as shown in Figure 12-2. The BOD18/33 detection status can be read from the Power and Clock Status Register (PCLKSR.BOD18/33DET).

Figure 12-2. BOD18/33 Block Diagram



The BOD18/33 is enabled by writing a one to the Enable bit in the BOD18/33 Control Register (BOD18/33CTRL.EN). When enabled, the BOD18/33 output will be masked during half a System RC oscillator (RCSYS) clock cycle and an additional two CPU clock cycles, in order to avoid false results. See Section 42.9 “Analog Characteristics” on page 1151 for parametric details.

12.6.3.1 Monitored Voltages

- VDDIO (BOD33)
- VDDCORE (BOD18)

12.6.3.2 Supported Modes

The BOD18/33 can operate in two different modes:

- Continuous mode
- Sampling mode

The Operation Mode bit in the Control Register (BOD18/33CTRL.MODE) is used to select the mode of operation.

12.6.3.3 Continuous Mode

When BOD18/33CTRL.MODE is zero, the BOD18/33 operates in continuous mode. In continuous mode the BOD18/33 is enabled and continously monitoring the supply voltage. The continuos mode is the default mode.

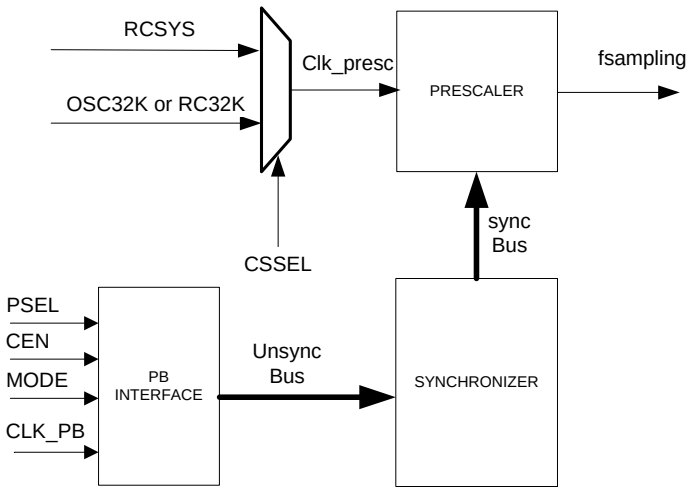
12.6.3.4 Sampling Mode

The sampling mode, also called the one-shot mode, is a low power mode where the BOD18/33 is being repeatedly enabled by incoming positive edges from a configurable choice of sampling clock sources including the System RC oscillator, the 32 KHz crystal oscillator (OSC32K) and the RC 32KHz oscillator (RC32K). In sampling mode, the BOD18/33 will monitor the VDDIO/VDDCORE voltages for a very short amount of time before it turns itself off again. Sampling mode is enabled by writing a one to BOD18/33CTRL.MODE. The frequency of the BOD18/33 one-shot pulses can be varied by the Prescaler Select field in the BOD18/33SAMPLING register (BOD18/33SAMPLING.PSEL).

$$f_{sampling} = f_{clk\_presc}/2^{(PSEL+1)}$$

Since the sampling mode clock is different from the PB clock domain, synchronization among the clocks is necessary. Figure 12-3 shows a block diagram of the sampling mode. The BOD18/33 synchronization ready bit in PCLKSR (PCLKSR.BOD18/33SYNRDY) shows the ready status of the synchronizer. Writing attempts to the BOD18/33SAMPLING register are ignored while PCLKSR.BOD18/33SYNRDY is zero.

Figure 12-3. Sampling Mode Block Diagram



12.6.3.5 Clock Sources

The clock mux is used to select the clock source for the sampling mode. The clock source is Enabled/Disabled by the Clock Enable bit in the BOD18/33SAMPLING register (BOD18/33SAMPLING.CEN). The Clock Source Select bit (BOD18/33SAMPLING.CSSEL) is used to select the clock source for the sampling mode.

There are two possible clock sources for the sampling prescaler:

- RCSYS: This oscillator is always enabled when selected as clock source. Refer to Section 10. “Power Manager (PM)” on page 109 for details about RCSYS and the Power Save Modes. Refer to the Section 42.7 “Oscillator Characteristics” on page 1143 chapter for the oscillator’s frequency characteristics.
- OSC32K or RC32K: This oscillator has to be enabled in the BSCIF Interface before using it as clock source for the BOD18/33. The BOD18/33 will not be able to detect if this clock is stopped.

It is recommended to disable the BOD18/33 before disabling the clock, to avoid freezing the BOD18/33 in an undefined state.

12.6.3.6 *Changing Clock Source*

The BOD18/33SAMPLING.CEN bit should always be disabled before changing the sampling clock source. To change the clock for the BOD18/33 the following steps need to be taken:

1. Write a zero to the CEN bit, leaving the other bits unchanged, to stop the clock.
2. Wait until CEN reads as zero, giving the clock time to stop.
3. Select a new clock source by writing to the BOD18/33SAMPLING.CSSEL bit.
4. Write a one to the CEN bit, leaving the other bits unchanged, to enable the clock.
5. Wait until CEN reads as one, giving the clock time to start.

12.6.3.7 *Changing the Prescaler*

To change the Prescaler value for the BOD18/33 during sampling mode the following steps need to be taken:

1.Wait until the PCLKSR.BOD18/33SYNRDY bit is one.

2.Write the selected value to the PSEL field.

12.6.3.8 *BOD18/33 Interrupts/Reset*

The BOD18/33 can generate either an interrupt or a reset based on the ACTION field in the BOD18/33CTRL register. The Interrupt/Reset is triggered when there is a voltage (VDDIO/VDDCORE) failure. Before enabling the BOD18/33 reset mode the ACTION field must first be initialized to zero in order to avoid any false BOD18/33 detection when the BOD18/33 is not ready.

12.6.3.9 *Hysteresis*

The hysteresis functionality may be used in both continuous and sampling mode. Writing a one to the HYST bit in BOD18/33CTRL can add hysteresis to the BOD18/33 trigger level.

12.6.3.10 *BOD18 Threshold Ranges*

The BOD18 has two threshold voltage ranges. There is a Standard Threshold Range, for operation in Power Scaling 0 and 2 Mode (PS0 andPS2), and a Low Threshold Range for operation at lower supply voltage values, such as Power Scaling 1 Mode or deep Power Save Mode. The user must take care of using the right threshold settings according to the different Modes. Refer to Section 42.9 “Analog Characteristics” on page 1151 for regulator minimal values in the different modes

12.6.3.11 *Power Save Modes*

In continuous mode the BOD18/33 will be off during WAIT, RETENTION and BACKUP Power Save Modes. In sampling mode the BOD18/33 is enabled in all Power Save Modes.

Table 12-2. BOD18/33 in different modes

Mode	Continuous Mode Possible actions during different modes		Sampling Mode Possible actions during different Modes	
RUN	Reset	Interrupt	Reset	Interrupt
SLEEP	Reset	Interrupt	Reset	Interrupt
WAIT	OFF	OFF	Reset	Delayed Interrupt
RETENTION	OFF	OFF	Reset	Delayed Interrupt
BACKUP	OFF	OFF	Reset	Delayed Interrupt

12.6.3.12 *Flash Fuses*

It is not recommended to override the default factory settings, but can be done (after reset) by writing to the BOD18/33 registers. Refer to the Fuse settings chapter for more details about BOD18/33 fuses and how to program these fuses.

If the Flash Calibration Done (FCD) bit in the BOD18/33CTRL register is 0 at reset, the flash calibration will be redone and the BOD18/33CTRL.FCD bit will be set before program execution starts in the CPU. If BOD18/33CTRL.FCD is one, the BOD18/33 configuration will not be altered after reset.

12.6.4 Voltage Regulator (VREG)

The embedded voltage regulator can be used to supply all the digital logic in the Core and the Backup power domains. The VREG features three different modes: normal mode, low power mode (LP) and Ultra Low Power mode (ULP).

To support Power Scaling and Power Save mode features, the VREG is controlled by the Backup Power Manager (BPM).

After a reset, the VREG is enabled. If an external voltage is applied on the VDDCORE pin, the VREG can be disabled by writing a one to the Disable bit of the VREG Configuration register (VREGCR.DIS).

When the regulator has reached its expected value, the VREGOK bit in the PCLKSR register is set. An interrupt is generated on a zero-to-one transition of VREGOK.

12.6.4.1 *Register Protection*

To prevent unexpected writes due to software bugs, write access to the VREGCR register is protected by a locking mechanism, for details refer to Section 12.7.7 “Unlock Register” on page 180.

To prevent further modifications by software, the contents of the calibration (VREG.CALIB) and disable (VREGCR.DIS) fields can be set as read-only by writing a one to the Store Final Value bit (VREGCR.SFV). Once this bit is set, the contents of VREGCR can not be modified until a POR18 reset is applied.

12.6.4.2 1MHz RC Oscillator

The 1MHz RC Oscillator which is used for switching regulator operation can also be internally routed to be used as a logic clock. This oscillator is controlled by the RC1MCR register. Its startup calibration value is read from flash fuses.

12.6.5 Bandgap

Rev: 1.1.0.7

The Bandgap provides a stable voltage reference used by the internally by the device. This reference is automatically turned on at startup. To save power consumption, the Bandgap features a Low Power Bandgap voltage reference which may be automatically enabled in Wait or Ret mode.

All modules using the Bandgap voltage reference, get their signal through a bandgap buffer. These buffers are controlled automatically.

12.6.6 Backup Registers (BR)

Rev: 1.0.0.1

Four 32-bit backup registers are available to store values when the device is in BACKUP Power Save Mode. These registers will keep their content even when the VDDCORE supply and the internal regulator supply voltage supplies are removed. The backup registers can be accessed by reading from and writing to the BR0, BR1, BR2, and BR3 registers.

12.6.7 Interrupts

The SCIF has the following interrupt sources:

- AE - Access Error:
  - A protected SCIF register was accessed without first being correctly unlocked.
- LPBGRDY- Access Error:
  - A 0 to 1 transition on the PCLKSR.LPBGRDY bit is detected..
- VREGOK: Voltage Regulator OK:
  - Set on 0 to 1 transition on the PCLKSR.VREGOK bit is detected.
- SSWRDY: Buck voltage regulator has stopped switching:
  - Set on 0 to 1 transition on the PCLKSR.SSWRDY bit is detected.
- BOD18SYNRDY - BOD18 synchronization ready:
  - Set on 0 to 1 transition on the PCLKSR.BOD18SYNRDY bit is detected.
- BOD33SYNRDY - BOD33 synchronization ready:
  - Set on 0 to 1 transition on the PCLKSR.BOD33SYNRDY bit is detected.
- BOD18DET - Brown out 1.8 Detected:
  - Set on 0 to 1 transition on the PCLKSR.BOD18DET bit is detected.

- BOD33DET - Brown out 3.3 Detected:
  - Set on 0 to 1 transition on the PCLKSR.BOD33DET bit is detected.
- RC32KSAT- RC32K FINE value saturated:
  - 0: The RC32K autocalibration has not saturated the FINE value.
  - 1: The RC32K autocalibration has brought FINE into saturation.
- RC32KREFE - RC32K Reference error:
  - 0: The RC32K has detected a running reference frequency.
  - 1: The RC32K reference frequency has stopped.
- RC32KLOCK - RC32K Locked to reference:
  - 0: The RC32K has not obtained a lock to the reference frequency.
  - 1: The RC32K has obtained alock to the reference frequency.
- RC32KRDY - RC32K Ready:
  - 0: Reads from FINE and COARSE will yield invalid values. Writes to FINE and COARSE is not yet complete
  - 1: FINE and COARSE can be read, writes to FINE and COARSE have been commited.
- OSC32RDY - 32kHz Oscillator Ready:
  - A 0 to 1 transition on the PCLKSR.OSC32RDY bit is detected.

The interrupt sources will generate an interrupt request if the corresponding bit in the Interrupt Mask Register is set. The interrupt sources are ORed together to form one interrupt request. The BSCIF will generate an interrupt request if at least one of the bits in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in the Interrupt Status Register (ISR) is cleared by writing a one to the corresponding bit in the Interrupt Clear Register (ICR). Because all the interrupt sources are ORed together, the interrupt request from the BSCIF will remain active until all the bits in ISR are cleared.

12.7 User Interface

Table 12-3. BSCIF Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0000	Interrupt Enable Register	IER	Write-only	0x00000000
0x0004	Interrupt Disable Register	IDR	Write-only	0x00000000
0x0008	Interrupt Mask Register	IMR	Read-only	0x00000000
0x000C	Interrupt Status Register	ISR	Read-only	0x00000000
0x0010	Interrupt Clear Register	ICR	Write-only	0x00000000
0x0014	Power and Clocks Status Register	PCLKSR	Read-only	0x00000000
0x0018	Unlock Register	UNLOCK	Write-only	0x00000000
0x001C	Chip Specific Configuration Register	CSCR	Read/Write	0x00000000
0x0020	Oscillator 32 Control Register	OSCCTRL32	Read/Write	0x00000004
0x0024	32kHz RC Oscillator Control Register	RC32KCR	Read/Write	0x00000000
0x0028	32kHz RC Oscillator Tuning Register	RC32KTUNE	Read/Write	0x00000000
0x002C	BOD33 Control Register	BOD33CTRL	Read/Write	-(2)
0x0030	BOD33 Level Register	BOD33LEVEL	Read/Write	-(2)
0x0034	BOD33 Sampling Control Register	BOD33SAMPLING	Read/Write	0x00000000
0x0038	BOD18 Control Register	BOD18CTRL	Read/Write	-(2)
0x003C	BOD18 Level Register	BOD18LEVEL	Read/Write	-(2)
0x0040	BOD18 Sampling Control Register	BOD18SAMPLING	Read/Write	0x00000000
0x0044	Voltage Regulator Configuration Register	VREGCR	Read/Write	-(1)
0x0058	1MHz RC Clock Configuration Register	RC1MCR	Read/Write	0x00000F00
0x0060	Bandgap Control Register	BGCTRL	Read/Write	0x00000000
0x0064	Bandgap Status Register	BGSR	Read-only	-(2)
0x0078 - 0x0084	Backup register n	BR	Read/Write	0x00000000
0x03E4	Backup Register Interface Version Register	BRIFBVERSION	Read-only	-(1)
0x03E8	BGREFIF Version Register	BGREFIFBVERSION	Read-only	-(1)
0x03EC	Voltage Regulator Version Register	VREGIFGVERSION	Read-only	-(1)
0x03F0	BOD Version Register	BODIFCVERSION	Read-only	-(1)
0x03F4	32kHz RC Oscillator Version Register	RC32KIFBVERSION	Read-only	-(1)
0x03F8	32 kHz Oscillator Version Register	OSC32IFAVERSION	Read-only	-(1)
0x03FC	BSCIF Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value is device specific. Refer to the Module Configuration section at the end of this chapter.
2. The reset value of this register depends on factory calibration.

12.7.1 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x0000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	LPBGRDY	-	VREGOK	SSWRDY	BOD18SYNRDY
7	6	5	4	3	2	1	0
BOD33SYNRDY	BOD18DET	BOD33DET	RC32SAT	RC32KREFE	RC32KLOCK	RC32KRDY	OSC32RDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

12.7.2 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x0004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	LPBGRDY	-	VREGOK	SSWRDY	BOD18SYNRDY
7	6	5	4	3	2	1	0
BOD33SYNRDY	BOD18DET	BOD33DET	RC32SAT	RC32KREFE	RC32KLOCK	RC32KRDY	OSC32RDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

12.7.3 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x0008

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	LPBGRDY	-	VREGOK	SSWRDY	BOD18SYNRDY
7	6	5	4	3	2	1	0
BOD33SYNRDY	BOD18DET	BOD33DET	RC32SAT	RC32KREFE	RC32KLOCK	RC32KRDY	OSC32RDY

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

12.7.4 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x000C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	LPBGRDY	-	VREGOK	SSWRDY	BOD18SYNRDY

7	6	5	4	3	2	1	0
BOD33SYNRDY	BOD18DET	BOD33DET	RC32SAT	RC32KREFE	RC32KLOCK	RC32KRDY	OSC32RDY

0: The corresponding interrupt is cleared.
1: The corresponding interrupt is pending.
A bit in this register is cleared when the corresponding bit in ICR is written to one.
A bit in this register is set when the corresponding interrupt occurs.

12.7.5 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x0010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	LPBGRDY	-	VREGOK	SSWRDY	BOD18SYNRDY

7	6	5	4	3	2	1	0
BOD33SYNRDY	BOD18DET	BOD33DET	RC32SAT	RC32KREFE	RC32KLOCK	RC32KRDY	OSC32RDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR.

12.7.6 Power and Clocks Status Register

Name: PCLKSR

Access Type: Read-only

Offset: 0x0014

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	LPBGRDY	RC1MRDY	VREGOK	SSWRDY	BOD18SYNRDY

7	6	5	4	3	2	1	0
BOD33SYNRDY	BOD18DET	BOD33DET	RC32SAT	RC32KREFE	RC32KLOCK	RC32KRDY	OSC32RDY

- **LPBGRDY:**
  0: Low Power Bandgap not enabled or not ready.
  1: Low Power Bandgap is stable and ready to be used as voltage reference source.
- **RC1MRDY:**
  0: RC1M oscillator not enabled or not ready.
  1: RC1M oscillator is stable and ready to be used as clock source.
- **SSWRDY: Buck voltage regulator has stopped switching:**
  0: Buck voltage regulator has not requested to stop switching or the stop switching request is not yet acknowledged.
  1: Buck voltage regulator has stopped switching after a stop switching request.
  This bit should be read only when the switching regulator feature is used.
- **BOD18SYNRDY:**
  1: No BOD18 synchronization is ongoing.
  0: A BOD18 synchronization is ongoing.
- **BOD33SYNRDY:**
  1: No BOD33 synchronization is ongoing.
  0: A BOD33 synchronization is ongoing.
- **BOD18DET:**
  0: No BOD18 Event.
  1: BOD18 has detected that power supply is going below BOD18 reference value.
- **BOD33DET:**
  0: No BOD33 Event.
  1: BOD33 has detected that power supply is going below BOD33 reference value.
- **RC32KSAT:**
  0: The RC32K autocalibration has not saturated the FINE value.

1: The RC32K autocalibration has brought FINE into saturation.

- **RC32KREFE:**
  0: The RC32K has detected a running reference frequency.
  1: The RC32K reference frequency has stopped.
- **RC32KLOCK:**
  0: The RC32K has not obtained a lock to the reference frequency.
  1: The RC32K has obtained alock to the reference frequency.
- **RC32KRDY:**
  0: Reads from FINE and COARSE will yield invalid values. Writes to FINE and COARSE is not yet complete
  1: FINE and COARSE can be read, writes to FINE and COARSE have been comitted.
- **OSC32RDY:**
  0: OSC32K not enabled or not ready.
  1: OSC32K is stable and ready to be used as clock source.

12.7.7 Unlock Register

Name: UNLOCK

Access Type: Write-only

Offset: 0x0018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	ADDR[9:8]	

7	6	5	4	3	2	1	0
ADDR[7:0]							

To unlock a write protected register, first write to the UNLOCK register with the address of the register to unlock in the ADDR field and 0xAA in the KEY field. Then, in the next PB access write to the register specified in the ADDR field.

- **KEY: Unlock Key**
  Write this bit field to 0xAA to enable unlock.
- **ADDR: Unlock Address**
  Write the address offset of the register to unlock to this field.

12.7.8 32KHz Oscillator Control Register

Name: OSCCTRL32

Access Type: Read/Write

Reset Value: 0x00000004

31	30	29	28	27	26	25	24
RESERVED	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	STARTUP[2:0]		

15	14	13	12	11	10	9	8
SELCURR[3:0]				-	MODE[2:0]		

7	6	5	4	3	2	1	0
-	-	-	-	EN1K	EN32K	-	OSC32EN

Note: This register is only reset by Power-On Reset

- **RESERVED**
  This bit must always be written to zero.
- **STARTUP: Oscillator Start-up Time**
  Select start-up time for 32 KHz oscillator

Table 12-4. Start-up Time for 32 KHz Oscillator

STARTUP	Number of RCSYS Clock Cycle	Approximative Equivalent Time (RCOSC = 115 kHz)
0	0	0
1	128	1.1 ms
2	8192	72.3 ms
3	16384	143 ms
4	65536	570 ms
5	131072	1.1 s
6	262144	2.3 s
7	524288	4.6 s

- **SELCURR: Current Selection**

Select current driven into the crystal.

Table 12-5. Crystal Current selection

SELCURR	Current Value (nA)
0	50
1	75
2	100
3	125
4	150
5	175
6	200
7	225
8	250
9	275
10	300 (recommended value)
11	325
12	350
13	375
14	400
15	425

- **MODE: Oscillator Mode**

Table 12-6. Operation Mode for 32 KHz Oscillator

MODE	Description
0	External clock connected to XIN32
1	Crystal mode. Crystal is connected to XIN32/XOUT32.
2	Reserved
3	Crystal mode with amplitude controlled mode. Crystal is connected to XIN32/XOUT32.
4	Crystal and high current mode. Crystal is connected to XIN32/XOUT32.
5	Crystal with high current mode and amplitude controlled mode. Crystal is connected to XIN32/XOUT32.
6	Reserved
7	Reserved

- **EN1K: 1 KHz output Enable**
  0: The 1 KHz output is disabled.
  1: The 1 KHz output is enabled.
- **EN32K: 32 KHz output Enable**
  0: The 32 KHz output is disabled.
  1: The 32 KHz output is enabled.
- **OSC32EN: 32 KHz Oscillator Enable**
  0: The 32 KHz Oscillator is disabled

1: The 32 KHz Oscillator is enabled

12.7.9 32 KHz RC Oscillator Control Register

Name: RC32KCR

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
FCD	-	REF	MODE	EN1K	EN32K	TCEN	EN

- **FCD: Flash calibration done**
  Signifies that COARSE in the RC32KCALIB register has been loaded with value from fuses.
  0: Fuses not loaded. CALIB will be loaded on reset.
  1: Fuses loaded. CALIB will keep its current value on reset.
- **REF: Reference select**
  Selects the reference clock for the Closed Loop mode.
  0: The OSC32K is selected as the reference.
  1: Generic clock is selected for reference. See module configuration for the generic clock configuration.
- **MODE: Mode Selection**
  0: RC32K operates in Open Loop mode.
  1: RC32K operates in Closed Loop mode.
- **EN1K: Enable 1 kHz output**
  0: 1 kHz output is disabled.
  1: 1 kHz output is enabled.
- **EN32K: Enable 32 KHz output**
  0: 32 kHz output is disabled.
  1: 32 kHz output is enabled.
- **TCEN: Temperature Compensation Enable**
  0: The oscillator is not temperature compensated.
  1: The oscillator is temperature compensated.
- **EN: Enable as Generic clock source**
  0: The oscillator is not enabled as a Generic clock source.
  1: The oscillator is enabled as a Generic clock source.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 12.7.7 “Unlock Register” on page 180 for details.

12.7.10 RC32K Tuning Register

Name: RC32KTUNE

Access Type: Read/Write

Reset Value: 0x002C0020

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	COARSE[4:0]						

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	FINE[5:0]					

- **COARSE: Coarse Value**
  The offset value for FINE.
- **FINE: Fine value**
  The fine calibration value. If in closed loop mode, this field is not accessible where reads return undefined values and writes are discarded.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 12.7.7 “Unlock Register” on page 180 for details.

12.7.11 BOD Control Register

Name: BOD18/33CTRL

Access Type: Read/Write

Reset Value: -

31	30	29	28	27	26	25	24
SFV	FCD	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	MODE
15	14	13	12	11	10	9	8
-	-	-	-	-	-	ACTION	
7	6	5	4	3	2	1	0
-	-	-	-	-	-	HYST	EN

- **SFV: BOD Control Register Store Final Value**
  0: BOD Control Register is not locked.
  1: BOD Control Register is locked.
  Once locked, the Control Register can not be re-written, only a reset unlocks the SFV bit.
- **FCD: BOD Fuse Calibration Done**
  This bit is set to one after any reset.
  0:Flash calibration to be redone after a BOD reset.
  1:Flash calibration not to be redone after a BOD reset.
- **MODE: Operation modes**
  0: Continuous mode.
  1: Sampling mode.
- **ACTION: Action**

Table 12-7. Action

Action[1:0]	Description
00	No Action.
01	The BOD generates a reset.
10	The BOD generates an interrupt.
11	No Action.

- **HYST: BOD Hysteresis**
  0: No hysteresis.
  1: Hysteresis enabled.

- **EN: Enable**
  0: The BOD is disabled.
  1: The BOD is enabled.

12.7.12 BOD Sampling Control Register

Name: BOD18/33SAMPLING

Access Type: Read/Write

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	PSEL[3:0]			

7	6	5	4	3	2	1	0
-	-	-	-	-	-	CSSEL	CEN

- **PSEL: Prescaler Select**
  Select the prescaler divide-by output for the BOD Sampling mode.
- **CSSEL: Clock Source Select**
  0: Selects the RCSYS as BOD Sampling clock source.
  1: Selects the 32kHz clock as BOD Sampling clock source. Setting the CK32S bit to one in the PMCON register (PMCON.CK32S) will select the RC32K oscillator as the 32 KHz clock source, and leaving it to zero will select the 32 KHz crystal oscillator output.
- **CEN: Clock Enable**
  Writing a zero to this bit will stop the BOD sampling clock.
  Writing a one to this bit will start the BOD sampling clock.
  0: The BOD Sampling clock is either disabled and stopped, or enabled but not yet stable.
  1: The BOD Sampling clock is either enabled and stable, or disabled but not yet stopped.

12.7.13 BOD Level Register

Name: BOD18/33LEVEL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
RANGE	-	-	-	-	-	-	

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	VAL[5:0]					

- **RANGE: BOD Threshold Range (available for BOD18 only)**
  0: The Standard Threshold Range is selected. See Section 42.9 “Analog Characteristics” on page 1151 for actual voltage levels.
  1: The Low Threshold Range is selected. See Section 42.9 “Analog Characteristics” on page 1151 for actual voltage levels.
  Note that any change to the RANGE field of the BOD18LEVEL register should be done with the BOD18 deactivated, this is to avoid spurious resets or interrupts.
- **VAL: BOD Value**
  This field sets the triggering voltage threshold for the BOD. See Section 42.9 “Analog Characteristics” on page 1151 for actual voltage levels.
  Note that any change to the VAL field of the BOD18/33LEVEL register should be done with the BOD deactivated, this is to avoid spurious resets or interrupts.

12.7.14 Voltage Regulator Configuration Register

Name: VREGCR

Access Type: Read/Write

Reset Value: -

31	30	29	28	27	26	25	24
SFV	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	SSWEVT	SSW	SSG
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	DIS

- **SFV: Store Final Value**
  0: The DIS field is read/write.
  1: The DIS field is read-only, to protect against further accidental writes.
- **SSWEVT: Stop Switching On Event Enable**
  0: The stop switching mechanism is not controlled by the Peripheral Event Controller.
  1: The stop switching mechanism is controlled by the Peripheral Event Controller.
  This bit should be set only when the switching regulator feature is used.
- **SSW: Stop Switching**
  0: The switching regulator is not forced to stop switching.
  1: The switching regulator is forced to stop switching if the SSWEVT bit is set to zero.
  This bit should be set only when the switching regulator feature is used.
- **SSG: Spread Spectrum Generator Enable**
  0: The spread spectrum of the internal switching oscillator is disabled.
  1: The spread spectrum of the internal switching oscillator is enabled.
  To operate, the spread spectrum functionality requires the CLKOEN in the RC1MCR register to be enabled.
  This bit should be set only when the switching regulator feature is used.
- **DIS: Voltage Regulator disable**
  0: The voltage regulator is enabled.
  1: The voltage regulator is disabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 12.7.7 “Unlock Register” on page 180 for details.

12.7.15 1MHz RC Clock Configuration Register

Name: RC1MCR

Access Type: Read/Write

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	CLKCAL[4:0]				

7	6	5	4	3	2	1	0
FCD	-	-	-	-	-	-	CLKOEN

- **CLKCAL: 1MHz RC Osc Calibration**
  Calibration field of the internal RC oscillator.
- **CLKOEN: 1MHz RC Osc Clock Output Enable**
  0: The 1MHz RC oscillator is not output.
  1: The 1MHz RC oscillator is output, available as a logic clock.
- **FCD: Flash Calibration Done**
  This bit is set to 1 when the CLKCAL field has been loaded by the flash fuses after a reset (read only).
  0: The flash calibration will be redone after any reset.
  1: The flash calibration will only be redone after a power-on reset (POR18).

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 12.7.7 “Unlock Register” on page 180 for details.

12.7.16 Bandgap Control Register

Name: BGCTRL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	TSEN

7	6	5	4	3	2	1	0
-	-	-	-	-	-	ADCISEL	

- **ADCISEL: ADC Input Selection**
  0: no connection.
  1: Reserved.
  2: ADC is connected to the voltage reference.
  3: Reserved.

12.7.17 Bandgap Status Register

Name: BGSR

Access Type: Read Only

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VREF		LPBGRDY	BGRDY

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
BGBUFRDY[7:0]							

- **VREF: Voltage Reference Used by the System**
  0: Both the Bandgap voltage reference and the Low Power Bandgap voltage reference are currently used by the system.
  1: The Bandgap voltage reference is currently used by the system.
  2: The Low Power Bandgap voltage reference is currently used by the system.
  3: Neither the Bandgap voltage reference nor the Low Power Bandgap voltage reference are used by the system.
- **LPBGRDY: Low Power Bandgap Voltage Reference Ready**
  0: The Low Power Bandgap voltage reference is not ready, therefore all modules using it as a source will not be able to operate correctly.
  1: The Low Power Bandgap voltage reference is ready.
- **BGRDY: Bandgap Voltage Reference Ready**
  0: The Bandgap voltage reference is not ready, therefore all modules using it as a source will not be able to operate correctly.
  1: The Bandgap voltage reference is ready.
- **BGBUFRDY: Bandgap Buffer Ready**
  0: The corresponding Bandgap Buffer is not ready, therefore all modules using it as a source will not be able to operate correctly.
  1: The corresponding Bandgap Buffer is ready. Refer to the “Bandgap Buffer Mapping” table in the SCIF Module Configuration section for details.

12.7.18 Backup Register n

Name: BRn

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DATA[31:24]							

23	22	21	20	19	18	17	16
DATA[23:16]							

15	14	13	12	11	10	9	8
DATA[15:8]							

7	6	5	4	3	2	1	0
DATA[7:0]							

This is a set of general-purpose read/write registers. Data stored in these registers is retained when the device is in BACKUP Power Save Mode.

Note that this registers are protected by a lock. To write to these registers the UNLOCK register has to be written first. Refer to Section 12.7.7 “Unlock Register” on page 180 for details.

12.7.19 Backup Register Interface Version Register

Name: BRIFBVERSION

Access Type: Read-only

Offset: 0x03E4

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.7.20 Bandgap Reference Interface Version Register

Name: BGREFIFBVERSION

Access Type: Read-only

Offset: 0x03E8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.7.21 Voltage Regulator Version Register

Name: VREGIFGVERSION

Access Type: Read-only

Offset: 0x03EC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.7.22 Brown-Out Detector Version Register

Name: BODIFCVERSION

Access Type: Read-only

Offset: 0x03F0

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.7.23 32kHz RC Oscillator Version Register

Name: RC32KIFBVERSION

Access Type: Read-only

Offset: 0x03F4

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.7.24 32kHz Oscillator Version Register

Name: OSC32IFAVERSION

Access Type: Read-only

Offset: 0x03F8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.7.25 BSCIF Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x03FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:0]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

12.8 Module Configuration

The specific configuration for each BSCIF instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 12-8. MODULE Clock Name

Module Name	Clock Name	Description
BSCIF	CLK_BSCIF	Clock for the BSCIF bus interface

Table 12-9. Bandgap Buffer Mapping

Bit	BG Buffer	Description
0	Flash	Flash
1	PLL	PLL
2	VREG	Main Regulator
4	ADC	ADC
5	LCD	LCD

Table 12-10. Register Reset Values

Register	Reset Value
BRIFBVERSION	0x00000100
BGREFIFBVERSION	0x00000110
VREGIFGVERSION	0x00000110
BODIFCVERSION	0x00000110
RC32KIFBVERSION	0x00000100
OSC32IFAVERSION	0x00000200
BSCIFVERSION	0x00000100

13. System Control Interface (SCIF)

Rev: 1.3.0.0

13.1 Features

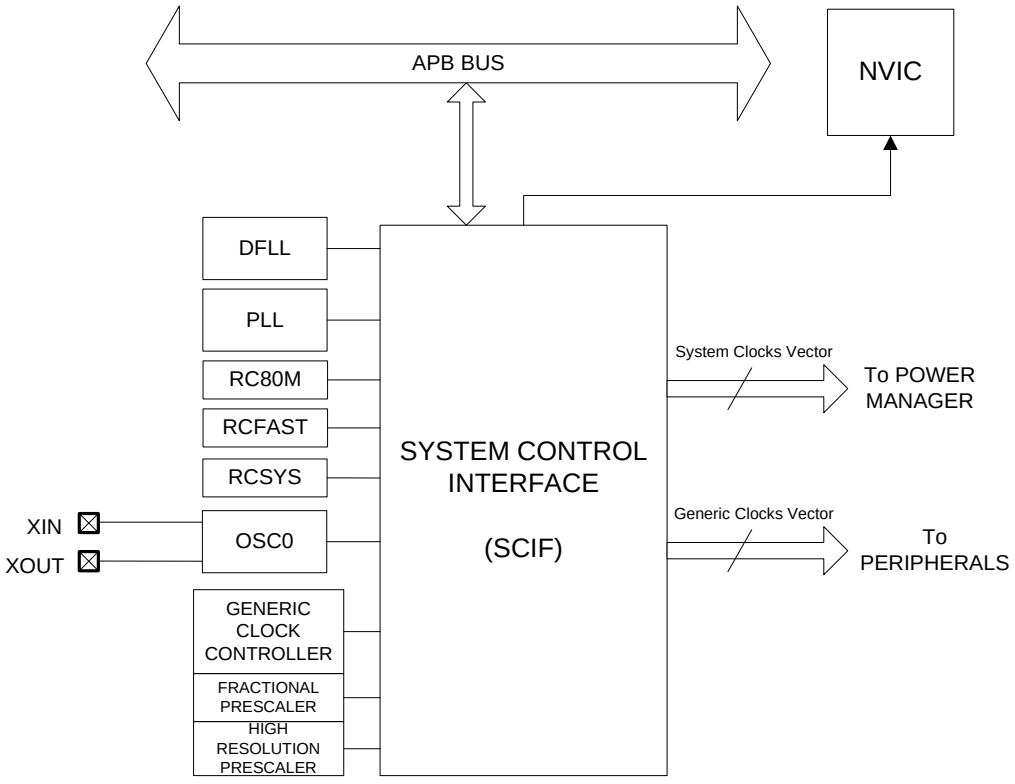
- **Supports crystal oscillator 0.6-30 MHz (OSC0)**
- **Supports Digital Frequency Locked Loop 20-150 MHz (DFLL)**
- **Supports Phase Locked Loop 48-240 MHz (PLL)**
- **Integrated low-power RC oscillator (RCSYS)**
- **Provides Generic clocks (GCLK) with wide frequency range**
- **Controls 80 MHz integrated RC Oscillator (RC80M)**
- **Controls 4/8/12 MHz integrated RC Oscillator (RCFAST)**

13.2 Overview

The System Control Interface (SCIF) controls the oscillators, Generic Clocks, PLL and DFLL. It is located in Core domain.

13.3 Block Diagram

Figure 13-1. SCIF Block Diagram



13.4 I/O Lines Description

Table 13-1. I/O Lines Description

Pin Name	Pin Description	Type
XIN	Crystal Input	Analog/Digital
XOUT	Crystal Output	Analog
GCLK3-GCLK0	Generic Clock Output	Output
GCLK_IN1-GCLK_IN0	Generic Clock Input	Input

13.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

13.5.1 I/O Lines

The SCIF provides a number of generic clock outputs, which can be connected to output pins, multiplexed with GPIO lines. The user must first program the GPIO controller to assign these pins to their peripheral function. If the I/O pins of the SCIF are not used by the application, they can be used for other purposes by the GPIO controller. Oscillator pins are also multiplexed with GPIO. When oscillators are used, the related pins are controlled directly by the SCIF, overriding GPIO settings.

13.5.2 Power Management

Oscillators are turned off in some Power Save modes and turned automatically on when the device wakes up.

13.5.3 Clocks

The SCIF controls some oscillators in the device. The oscillators can be used as source for the CPU and peripherals, selection of source is done in the Power Manager. The oscillator can also be used as source for generic clocks.

13.5.4 Interrupts

The SCIF interrupt request line is connected to the NVIC. Using the SCIF interrupt requires the NVIC to be programmed first.

13.5.5 Debug Operation

The SCIF does not interact with debug operations.

13.6 Functional Description

13.6.1 Oscillator (OSC) Operation

Rev: 1.1.4.0

The main oscillator (OSCn) is designed to be used with an external 0.6 to 30 MHz crystal and two biasing capacitors, as shown in the Electrical Characteristics chapter, or with an external clock connected to the XIN. The oscillator can be used as source for the main clock in the device, as described in the Power Manager chapter. The oscillator can be used as source for the generic clocks, as described in the Generic Clocks section.

The oscillator is disabled by default after reset. When the oscillator is disabled, the XIN and XOUT pins can be used as general purpose I/Os. When the oscillator is enabled, the XIN and XOUT pins are controlled directly by the SCIF, overriding GPIO settings. When the oscillator is configured to use an external clock, the clock must be applied to the XIN pin while the XOUT pin can be used as general purpose I/O.

The oscillator is enabled by writing a one to the Oscillator Enable bit in the Oscillator Control register (OSCCTRLn.OSCEN). Operation mode (external clock or crystal) is selected by writing to the Oscillator Mode bit in OSCCTRLn (OSCCTRLn.MODE). The oscillator is automatically disabled in certain sleep modes to reduce power consumption, as described in the Power Manager chapter.

After a hard reset, or when waking up from a sleep mode where the oscillators were disabled, the oscillator will need a certain amount of time to stabilize on the correct frequency. This start-up time can be set in the OSCCTRLn register.

The SCIF masks the oscillator outputs during the start-up time, to ensure that no unstable clocks propagate to the digital logic.

The OSCn Ready bit in the Power and Clock Status Register (PCLKSR.OSCnRDY) is set when the oscillator is stable and ready to be used as clock source. An interrupt can be generated on a zero-to-one transition on OSCnRDY if the OSCnRDY bit in the Interrupt Mask Register (IMR.OSCnRDY) is set. This bit is set by writing a one to the corresponding bit in the Interrupt Enable Register (IER.OSCnRDY).

13.6.2 PLL Operation

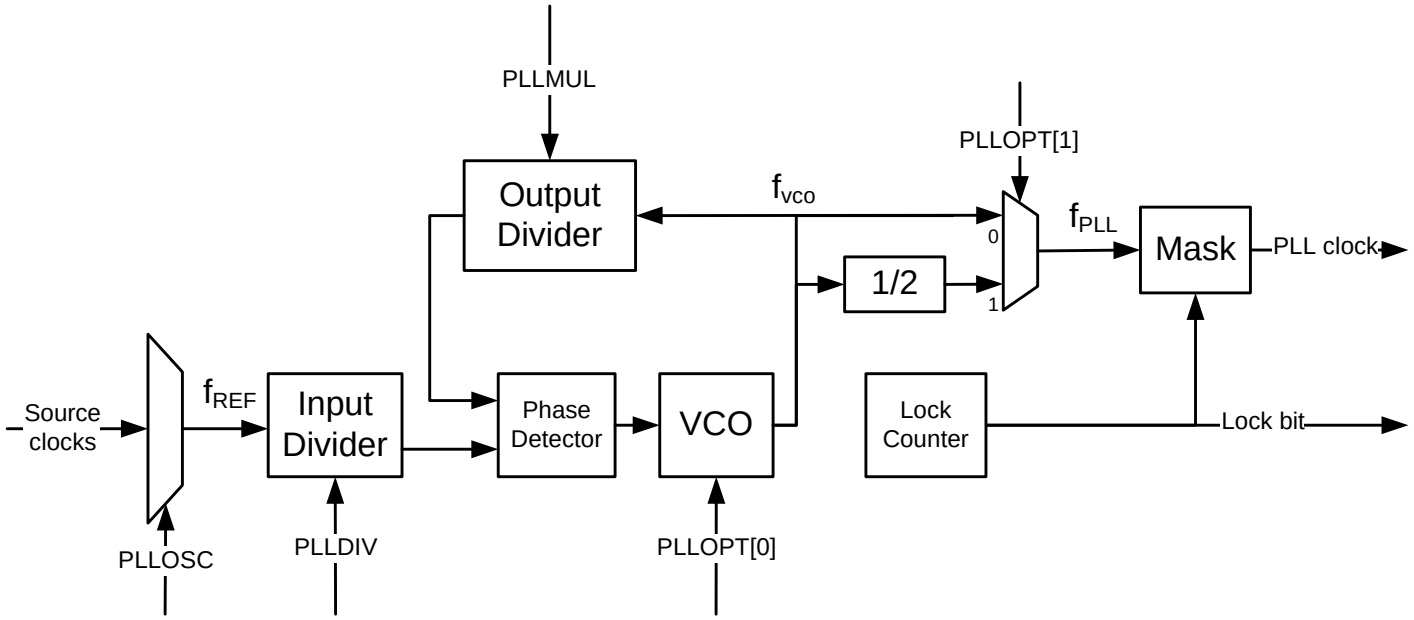
Rev: 1.1.2.0

The device contains one Phase Locked Loop (PLL), which is controlled by the Phase Locked Loop Interface (PLLIF). The PLL is disabled by default, but can be enabled to provide high frequency source clocks for synchronous or generic clocks. The PLL can use different clock sources as reference clock, refer to the “PLL Clock Sources” table in the SCIF Module Configuration section for details. The PLL output is divided by a multiplication factor, and the PLL compares the phase of the resulting clock to the reference clock. The PLL will adjust its output frequency until the two compared clocks phases are equal, thus locking the output frequency to a multiple of the reference clock frequency.

When the PLL is switched on, or when changing the clock source or multiplication factor for the PLL, the PLL is unlocked and the output frequency is undefined. The PLL clock for the digital logic is automatically masked when the PLL is unlocked, to prevent the connected digital logic from receiving a too high frequency and thus become unstable.

The PLL can be configured by writing the PLL Control Register (PLLn). To prevent unexpected writes due to software bugs, write access to the PLLn register is protected by a locking mechanism, for details refer to Section 13.7.7 “Unlock Register” on page 229.

Figure 13-2. PLL with Control Logic and Filters



13.6.2.1 Enabling the PLL

Before the PLL is enabled it must be set up correctly. The PLL Oscillator Select field (PLLOSC) selects a source for the reference clock. The PLL Multiply Factor (PLLMUL) and PLL Division Factor (PLLDIV) fields must be written with the multiplication and division factors, respectively. The PLLMUL must always be greater than 1, creating the PLL frequency:

$f_{vco} = (PLLMUL+1)/PLLDIV \bullet f_{REF}$, if PLLDIV >0

$f_{vco} = 2\bullet(PLLMUL+1) \bullet f_{REF}$, if PLLDIV = 0

The PLL Options (PLLOPT) field should be configured to proper values according to the PLL operating frequency. The PLLOPT field can also be configured to divide the output frequency of the PLL by 2 and Wide-Bandwidth mode, which allows faster startup time and out-of-lock time.

It is not possible to change any of the PLL configuration bits when the PLL is enabled, Any write to PLLn while the PLL is enabled will be discarded.

After setting up the PLL, the PLL is enabled by writing a one to the PLL Enable (PLLEN) bit in the PLLn register.

13.6.2.2 Disabling the PLL

The PLL is disabled by writing a zero to the PLL Enable (PLLEN) bit in the PLLn register. After disabling the PLL, the PLL configuration fields becomes writable.

13.6.2.3 PLL Lock

The lock signal for each PLL is available as a PLLLOCKn flag in the PCLKSR register. If the lock for some reason is lost, the PLLLOCKLOSTn flag in PCLKSR register will be set. An interrupt can be generated on a 0 to 1 transition of these bits.

13.6.3 Digital Frequency Locked Loop (DFLL) Operation

Rev.: 1.1.0.0

The number of DFLLs is device specific. A specific DFLL is referred to as DFLLx, where x can be any number from 0 to n, where n refers to the last DFLL instance. Refer to the module configuration section for details. The DFLLx is controlled by the corresponding DFLLx registers. DFLLx is disabled by default, but can be enabled to provide a high-frequency source clock for synchronous and generic clocks.

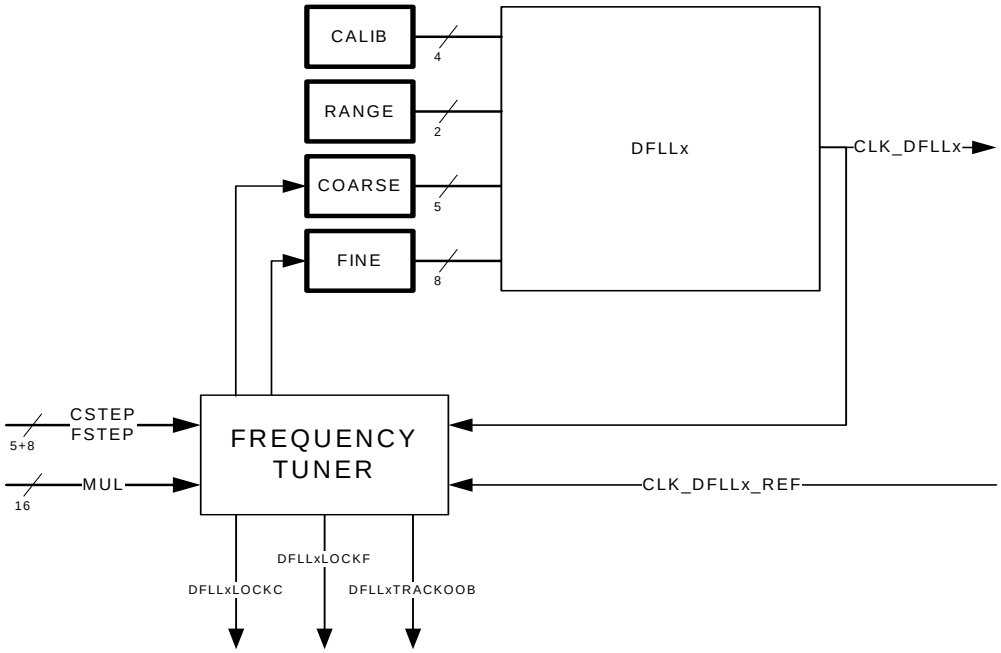
Features:

- Internal oscillator with no external components
- 20-150 MHz output frequency
- Can operate standalone as a high-frequency programmable oscillator in open loop mode
- Can operate as an accurate frequency multiplier against a known frequency in closed loop mode
- Optional spread-spectrum clock generation
- Very high-frequency multiplication supported - can generate all frequencies from a 32 KHz reference clock

DFLLx can operate in both open loop mode and closed loop mode. In closed loop mode a low-frequency clock with high accuracy can be used as reference clock to get high accuracy on the output clock (CLK_DFLLx).

To prevent unexpected writes due to software bugs, write access to the configuration registers is protected by a locking mechanism. For details refer to Section 13.7.7 “Unlock Register” on page 229.

Figure 13-3. Block Diagram



13.6.3.1 Enabling the DFLL

DFLLx is enabled by writing a one to the Enable bit in the DFLLx Configuration Register (DFLLxCONF.EN). No other bits or fields in DFLLxCONF must be changed simultaneously, or before DFLLx is enabled.

13.6.3.2 Internal Synchronization

Due to multiple clock domains, values in the DFLLx configuration registers need to be synchronized to other clock domains. The status of this synchronization can be read from the Power and Clocks Status Register (PCLKSR). Before writing to any of the DFLLx configuration registers, the user must check that the DFLLx Ready bit in PCLKSR is one (PCLKSR.DFLLxRDY). When this bit is set, the DFLLx can be configured, and CLK_DFLLx is ready to be used. Any write to any of the DFLLx configuration register while DFLLxRDY is zero will be ignored. An interrupt can be generated on a zero-to-one transition of DFLLxRDY.

Before reading the value in any of the DFLLx configuration registers a one must be written to the Synchronization bit in the DFLLx Synchronization Register (DFLLxSYNC.SYNC). The DFLLx configuration registers are ready to be read when PCLKSR.DFLLxRDY is set.

13.6.3.3 Disabling the DFLL

DFLLx is disabled by writing a zero to DFLLxCONF.EN. No other bits or fields in DFLLxCONF must be changed simultaneously.

13.6.3.4 Open Loop Operation

After enabling DFLLx, open loop mode is selected. When operating in open loop mode the output frequency of the DFLLx will be determined by the values written to the Calibration Value field and the Range Value field in DFLLxCONF (DFLLxCONF.CALIB and DFLLxCONF.RANGE), and the Coarse Value field and the Fine Value field in the DFLLx Value Register (DFLLxVAL.COARS and DFLLxVAL.FINE). CALIB is used for process calibration, and should not be changed by the user. It is loaded with a factory defined value stored in the Flash fuses. The Fuse Calibration Done bit in DFLLxCONF (DFLLXCONF.FCD) is set when the fuse values are loaded. Writing a zero to this bit will reload the CALIB value from the Flash fuses after any reset. Refer to the Fuse Settings chapter for more details about how to program the fuses. RANGE selects the frequency range of the DFLLx, see Table 13-2.

Table 13-2. DFLL Frequency Range

RANGE	Frequency range [MHz]
0	96-150
1	50-110
2	25-55
3	20-30

It is possible to change the value of DFLLxCONF.CALIB, DFLLxCONF.RANGE, DFLLxVAL.COARSE, and DFLLxVAL.FINE, and thereby the output frequency of the DFLLx output clock, CLK_DFLLx, while the DFLL is enabled and in use.

CLK_DFLLx is ready to be used when PCLKSR.DFLLxRDY is set after enabling the DFLLx.

The frequency range in open loop mode is 20-150MHz, but maximum frequency can be higher, and the minimum frequency can be lower. The best way to start the DFLL at a specific frequency

in open loop mode is to first configure it for closed loop mode, see Section 13.6.3.5. When a lock is achieved, read back the COARSE and FINE values and switch to open loop mode using these values. An alternative approach is to use the Frequency Meter (FREQM) to monitor the DFLL frequency and adjust the COARSE and FINE values based on measurement results form the FREQM. Refer to the FREQM chapter for more information on how to use it. Note that the output frequency of the DFLL will drift when in open loop mode due to temperature and voltage changes. Refer to Section 42. “Electrical Characteristics” on page 1121 for details.

13.6.3.5 Closed Loop Operation

DFLLx must be correctly configured before closed loop operation can be enabled. After enabling DFLLx, the DFLLx must be configured in the following way:

1. Enable and select a reference clock (CLK_DFLLx_REF). CLK_DFLLx_REF is a generic clock, refer to Generic Clocks section for details.
2. Write an appropriate value to DFLLxCONF.RANGE to choose desired range of the output frequency.
3. Select the multiplication factor in the Multiply Factor field in the DFLLx Multiplier Register (DFLLxMUL.MUL). Care must be taken when choosing MUL so the output frequency does not exceed the maximum frequency of the device.
4. Select the maximum step size allowed in finding the COARSE and FINE values by writing the appropriate values to the Coarse Maximum Step field and Fine Maximum Step field in the DFLLx Maximum Step Register (DFLLxSTEP.CSTEP and DFLLxSTEP.FSTEP). A small step size will ensure low overshoot on the output frequency, but can typically result in longer lock times. A high value might give a big overshoot, but can typically give faster locking. DFLLxSTEP.CSTEP and DFLLxSTEP.FSTEP must not be higher than 50% of the maximum value of DFLLxVAL.COARSE and DFLLxVAL.FINE respectively.
5. Optional: Select start values for COARSE and FINE by writing the appropriate values to DFLLxVAL.COARSE and DFLLxVAL.FINE respectively. Selecting values for COARSE and FINE that are close to the final values and small step sizes for CSTEP and FSTEP will reduce the time required to achieving lock on COARSE and FINE. If this step is skipped, COARSE will start at its current value and FINE will start at half its maximum value.
6. Start the closed loop mode by writing a one to Mode Selection bit in DFLLxCONF (DFLLxCONF.MODE).

The frequency of CLK_DFLLx (f_{CLK_DFLLx}) is given by:

$$f_{CLK\_DFLLx} = \text{DFLLxMUL.MUL} \cdot f_{CLK\_DFLLx\_REF}$$

where $f_{CLK\_DFLLx\_REF}$ is the frequency of the reference clock (CLK_DFLLx_REF). DFLLxVAL.COARSE and DFLLxVAL.FINE are read-only in closed loop mode, and are controlled by the frequency tuner shown in Figure 13-3 to meet user specified frequency.

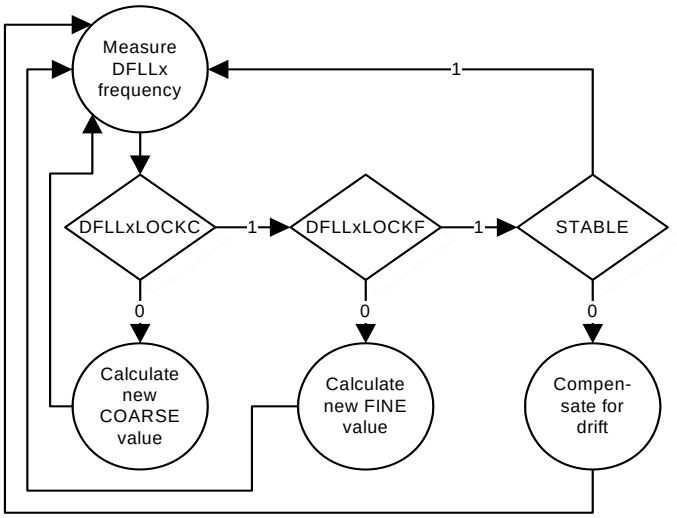
In closed loop mode, the value in DFLLxVAL.COARSE is used by the frequency tuner as a starting point for COARSE and half the maximum value of FINE will be used as a starting point for FINE. Writing DFLLxVAL.COARSE to a value close to the final value before entering closed loop mode will reduce the time needed to get a lock on COARSE. This can be done both before or after writing the multiplication value to DFLLxMUL.MUL. If this is done after writing DFLLxMUL.MUL, the value written to DFLLxVAL.FINE will also be used as a starting point for FINE instead of the default behavior which is to use half the maximum value of FINE.

Writing to DFLLxMUL.MUL while in closed loop mode will reset the locks. FINE will be set to half its maximum value and the full locking sequence will be restarted.

Frequency locking

The locking of the frequency in closed loop mode is divided into two stages. In the COARSE stage the control logic quickly finds the correct value for DFLLxVAL.COARSE and thereby sets the output frequency to a value close to the correct frequency. The DFLLx Locked on Coarse Value bit in PCLKSR (PCLKSR.DFLLxLOCKC) will be set when this is done. An interrupt can be generated on a zero-to-one transition of PCLKSR.DFLLxLOCKC. In the FINE stage the control logic tunes the value in DFLLxVAL.FINE so the output frequency will be very close to the desired frequency. DFLLx Locked on Fine Value bit in PCLKSR (PCLKSR.DFLLxLOCKF) will be set when this is done. An interrupt can be generated on a zero-to-one transition of PCLKSR.DFLLx-LOCKF. Figure 13-4 shows the state diagram for the closed loop mode, and how the lock bits are set.

Figure 13-4. DFLL Closed Loop State Diagram



CLK_DFLLx is ready to be used when PCLKSR.DFLLxRDY is set after enabling DFLLx. However, the accuracy of the output frequency depends on which locks are set.

For lock times, refer to Section 42. “Electrical Characteristics” on page 1121 chapter.

Frequency error measurement

The ratio between CLK_DFLLx_REF and CLK_DFLLx is measured automatically. The difference between this ratio and DFLLxMUL is stored in the Multiplication Ratio Difference field in the DFLLx Ratio Register (DFLLxRATIO.RATIODIFF). The relative error on CLK_DFLLx compared to the target frequency can be calculated as follows:

$$\text{ERROR} = \frac{\text{RATIODIFF} \cdot f_{\text{CLK\_DFLLx\_REF}}}{f_{\text{CLK\_DFLLx}}}$$

Drift compensation

If the Stable DFLL Frequency bit in DFLLxCONF (DFLLxCONF.STABLE) is zero, the frequency tuner will automatically compensate for drift in the f_{CLK_DFLLx} without losing either of the locks. This will result in that DFLLxVAL.FINE can change after every measurement of CLK_DFLLx.

If a one is written to DFLLxCONF.STABLE, DFLLxVAL.FINE will never change after Fine Lock is set. The frequency will be measured and the error value can be read from the DFLLxRATIO.RATIODIFF.

It is possible to change the value of DFLLxCONF.STABLE while in lock, without loosing the locks. This enables the user to let the DFLLx compensate for drift if DFLLxRATIO.RATIODIFF is too big, and having a stable frequency when this is required.

Reference clock stop detection

If CLK_DFLLx_REF stops or is running at a very slow frequency, the DFLLx Reference Clock Stopped bit in PCLKSR will be set (PCLKSR.DFLLxRCS). Note that the detection of a stopped reference clock will take a long time. The DFLLx operate as if it was in open loop mode if it detects that the reference clock has stopped. Closed loop mode operation will automatically resume if the CLK_DFLLx_REF is restarted. An interrupt can be generated on a zero-to-one transition on PCLKSR.DFLLxRCS.

13.6.3.6 Dealing With Delay in the DFLL

The time from selecting a new CLK_DFLLx frequency until this frequency is output by the DFLLx, can be up to several micro seconds. If the difference between the desired output frequency (CLK_DFLLx) and the frequency of CLK_DFLLx_REF is small this can lead to an instability in the DFLLx locking mechanism, which can prevent the DFLLx from achieving locks. To avoid this, a chill cycle where the CLK_DFLLx frequency is not measured can be enabled. The chill cycle is enabled by default, but can be disabled by writing a one to the Chill Cycle Disable bit in DFLLxCONF (DFLLxCONF.CCDIS). Enabling chill cycles might double the lock time.

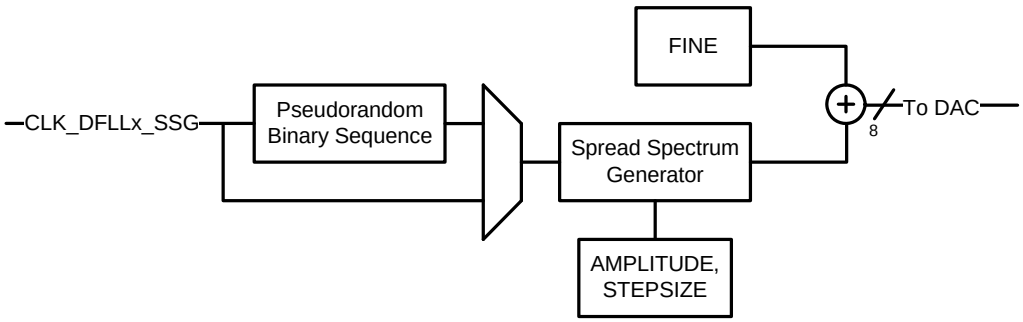
Another solution to the same problem is to use less strict lock requirements. This is called Quick Lock (QL), which is also enabled by default, but can be disabled by writing a one to the Quick Lock Disable bit in DFLLxCONF (DFLLxCONF.QLDIS). The QL might lead to bigger spread in the outputted frequency than chill cycles, but the average output frequency is the same.

If the target frequency is below 40 MHz, one of these should always be enabled.

13.6.3.7 Spread Spectrum Generator (SSG)

When CLK_DFLLx is used as the main clock source for the device, the EMI radiated from the device will be synchronous to f_{CLK_DFLLx}. To provide better Electromagnetic Compatibility (EMC) DFLLx can provide a clock with the energy spread in the frequency domain (spread spectrum). This is done by adding or subtracting values from the FINE value. SSG is enabled by writing a one to the Enable bit in the DFLLx Spread Spectrum Generator Control Register (DFLLx-SSG.EN). The Spread Spectrum Generator Block Diagram is shown in Figure 13-5.

Figure 13-5. Spread Spectrum Generator Block Diagram.



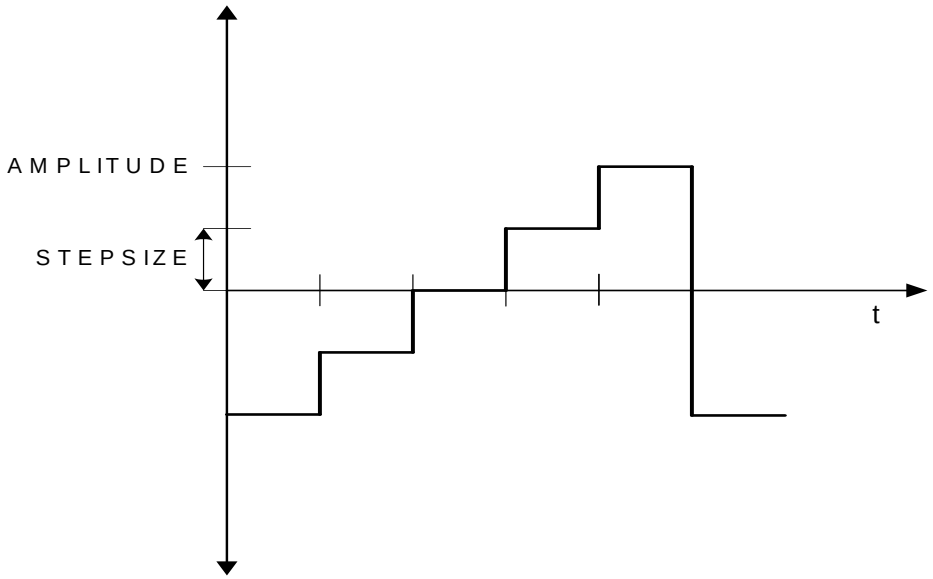
The generic clock CLK_DFLLx_SSG must be configured and enabled before SSG is enabled, refer to Generic Clocks section for details. This clock sets the rate at which the SSG changes the frequency of the DFLL clock to generate a spread spectrum. The frequency of this clock should be higher than $f_{CLK\_DFLLx\_REF}$ to ensure that the DFLLx can lock.

Optionally, the clock ticks can be qualified by a Pseudo Random Binary Sequence (PRBS) if the PRBS bit in DFLLxSSG is one (DFLLxSSG.PRBS). This reduces the modulation effect of CLK_DFLLx_SSG frequency onto f_{CLK_DFLLx}.

The step size of the SSG is selected by writing to the SSG Step Size field in DFLLxSSG (DFLLxSSG.STEPSIZE). If the step size is *n*, the output value from the SSG will be incremented/decremented by *n* on every tick of the source clock. DFLLxSSG.STEPSIZE equal to zero or one will result in a step size equal to one.

The amplitude of the frequency variation can be selected by writing an appropriate value to the SSG Amplitude field in DFLLxSSG (DFLLxSSG.AMPLITUDE). If DFLLxSSG.AMPLITUDE is larger than one, the sequence added to the FINE value will depend on both DFLLxSSG.AMPLITUDE and DFLLxSSG.STEPSIZE, as shown in Figure 13-6. If DFLLxSSG.AMPLITUDE is zero the SSG will toggle on the LSB of the FINE value. If DFLLxSSG.AMPLITUDE is one the SSG will add the sequence {1,-1, 0} to FINE.

Figure 13-6. Spread Spectrum Sequence added to FINE Value



The Spread Spectrum Generator is available in both open and closed loop mode.

When spread spectrum is enabled in closed loop mode, and the DFLLxSSG.AMPLITUDE value is high, an overflow/underflow in FINE is more likely to occur.

13.6.3.8 Wake From Power Save Modes

DFLLx may optionally reset its lock bits when waking from a Power Save Mode which disables the DFLLx. This is configured by the Lose Lock After Wake bit in DFLLxCONF (DFLLxCONF.LLAW). If DFLLxCONF.LLAW is zero the DFLL will be re-enabled and start running with the same configuration as before going to sleep even if the reference clock is not available. The locks will not be lost. When the reference clock has restarted, the FINE tracking will quickly compensate for any frequency drift during sleep if DFLLxCONF.STABLE is zero. If

DFLLxCONF.LLAW is one when going to a Power Save Mode where the DFLLx is turned off, the DFLLx will lose all its locks when waking up, and needs to regain these through the full lock sequence.

13.6.3.9 Accuracy

There are mainly three factors that decide the accuracy of the f_{CLK_DFLLx}. These can be tuned to obtain maximum accuracy when fine lock is achieved.

- FINE resolution: The frequency step between two FINE values. This is relatively smaller for high output frequencies.
- Resolution of the measurement: If the resolution of the measured f_{CLK_DFLLx} is low, i.e. the ratio between the CLK_DFLLx frequency and the CLK_DFLLx_REF frequency is small, then the DFLLx might lock at a frequency that is lower than the targeted frequency. It is recommended to use a reference clock frequency of 32 KHz or lower to avoid this issue for low target frequencies.
- The accuracy of the reference clock.

13.6.4 System RC Oscillator (RCSYS)

Rev: 1.1.4.0

The system RC oscillator has a startup time of three cycles, and is always available except in some sleep modes. Refer to Section 10. “Power Manager (PM)” on page 109 for details. The system RC oscillator operates at a nominal frequency of 115 kHz, and is calibrated using the Calibration Value field (CALIB) in the RC Oscillator Calibration Register (RCCR). After a Power-on Reset (POR), the RCCR.CALIB field is loaded with a factory defined value stored in the Flash fuses. Refer to the Fuse setting chapter for more details about RCCR fuses and how to program the fuses.

If the Flash Calibration Done (FCD) bit in the RCCR is zero at any reset, the flash calibration will be redone and the RCCR.FCD bit will be set before program execution starts in the CPU. If the RCCR.FCD is one, the flash calibration will only be redone after a Power-on Reset.

To prevent unexpected writes to RCCR due to software bugs, write access to this register is protected by a locking mechanism. For details refer to Section 13.7.7 “Unlock Register” on page 229.

Although it is not recommended to override default factory settings, it is still possible to override the default values by writing to RCCR.CALIB.

13.6.5 4/8/12MHz RC Oscillator (RCFAST) Operation

Rev: 2.0.02

The RCFAST can be used as the main clock in the device, as described in Section 10. “Power Manager (PM)” on page 109. The RCFAST can also be used as source for the generic clocks, as described in the “Generic Clocks” section of the SCIF.

The RCFAST is enabled by writing a one to the Oscillator Enable bit in the RCFAST Configuration Register (RCFASTCFG.EN), and disabled by writing a zero to this bit. When enabling the RCFAST, RCFASTCFG.EN must be read back until it reads one. The user must ensure that the RCFAST is fully disabled before enabling, and that the RCFAST is fully enabled before disabling by reading back RCFASTCFG.EN.

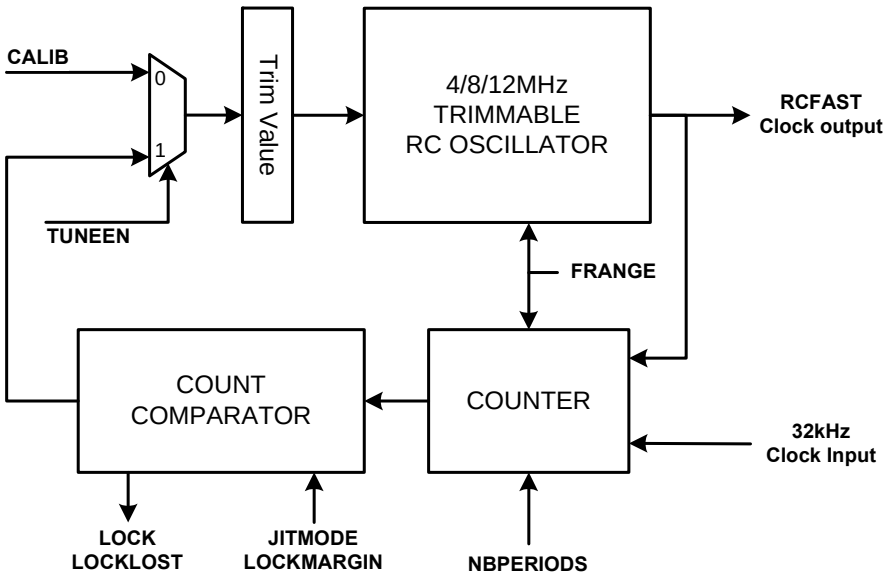
After a Power-On Reset (POR), the Calibration field (RCFASTCFG.CALIB) is loaded with a factory-defined value stored in the Flash fuses. The Flash Calibration Done bit (RCFASTCFG.FCD) is set when RCFASTCFG.CALIB has been loaded from flash.

For testing purposes, it is possible to override the default fuse values by writing to the RCFASTCFG.CALIB field. To prevent unexpected frequency change, the RCFAST must be disabled before modifying the CALIB field.

To prevent unexpected writes due to software bugs, write access to the RCFASTCFG register is protected by a locking mechanism. For details refer to Section 13.7.7 “Unlock Register” on page 229. The RCFAST is automatically switched off in certain sleep modes to reduce power consumption, as described in Section 10. “Power Manager (PM)” on page 109.

The RCFAST is an internal oscillator that can, when operating in open loop mode, output a 4-, 8- or 12-MHz frequency, factory-calibrated under typical voltage and temperature. Alternatively, the frequency can be further tuned to a multiplication ratio relative to the 32 kHz reference clock.

Figure 13-7. RCFAST Block Diagram



13.6.5.1 *Product Dependencies*

The tuner uses a 32 kHz clock as reference clock and should only be enabled when the 32 kHz clock is running. Refer to the 32 kHz Clock section for details.

13.6.5.2 *General Use*

The RCFAST consists of two parts, one oscillator and one tuner. The oscillator is enabled by writing a one to the Oscillator Enable bit (EN) in the 4/8/12MHz RC Oscillator Configuration Register (RCFASTCFG). The tuner is enabled by writing a one to the Tuner Enable bit (TUNEEN) in RCFASTCFG. The oscillator has to be enabled for the tuner to work.

13.6.5.3 *Open Loop Mode*

If the tuner is disabled (RCFASTCFG.TUNEEN is zero), the RCFAST is in open loop mode. In open loop mode, the user can control the oscillation frequency by writing to the Frequency Range (FRANGE) and Calibration (CALIB) fields of the 4/8/12MHz RC Oscillator Configuration

Register (RCFASTCFG). The FRANGE and CALIB fields should only be updated when the RCFAST is disabled. Since this is in open loop mode, the frequency will be voltage, temperature, and process dependent. Refer to Section 42. “Electrical Characteristics” on page 1121 for details.

13.6.5.4 Closed Loop Mode

If the tuner is enabled (RCFASTCFG.TUNEEN is one), the RCFAST is in closed loop mode. In closed loop mode, the frequency is controlled by a tuner which measures the ratio between the RCFAST output frequency and the 32 kHz reference clock frequency. The initial output frequency is first chosen, when the RCFAST is disabled, by writing to the Frequency Range (FRANGE) and Calibration (CALIB) fields of the 4/8/12 MHz RC Oscillator Configuration Register (RCFASTCFG). The RCFAST clock periods are counted during one or several 32kHz clock periods. And that count allows the tuner to adjust the RCFAST trim value to get closer to the target number of 32 kHz clock periods. The value of the RCFASTCFG.NBPERIODS field is used to improve the tuner precision by counting RCFAST clock periods across (1 << NBPERIODS) periods of the 32 kHz clock. The NBPERIODS, LOCKMARGIN and JITMODE fields should only be updated when the tuner is disabled.

The 32 kHz reference clock is controlled by the BSCIF, refer to the 32 kHz Clock section for details.

When a tuning procedure is completed, the Lock bit (RCFASTLOCK) in the Status Register (PCLKSR) will be set if the current count is within RCFASTCFG.LOCKMARGIN of the target count. Otherwise, the Lock Lost bit (RCFASTLOCKLOST) in PCLKSR will be set.

13.6.5.5 Factory Calibration

If the Flash Calibration Done (FCD) bit in RCFASTCFG is zero at any reset, then the calibration will be redone, using the factory calibration value stored in the Flash fuses, and the RCFASTCFG.FCD bit will be set before program execution starts in the CPU. If the RCFASTCFG.FCD bit is one, then the RCFAST configuration will not be changed during any reset except POR reset.

13.6.5.6 Register Protection

To prevent unexpected writes to RCFAST registers due to software bugs, write access to the registers are protected by a locking mechanism. For details refer to Section 13.7.7 “Unlock Register” on page 229.

13.6.6 80MHz RC Oscillator (RC80M) Operation

Rev: 1.0.0.1

The RC80M can be used as the main clock in the device, as described in Section 10. “Power Manager (PM)” on page 109. The RC80M can also be used as source for the generic clocks, as described in the “Generic Clocks” section of the SCIF.

The RC80M is enabled by writing a one to the Enable bit in the RC80M Control register (RC80MCR.EN), and disabled by writing a zero to this bit. When enabling the RC80M, RC80MCR.EN must be read back until it reads one. The user must ensure that the RC80M is fully disabled before enabling, and that the RC80M is fully enabled before disabling by reading back RC80MCR.EN.

After a Power-On Reset (POR), the Calibration Value field (RC80MCR.CALIB) is loaded with a factory defined value stored in the Flash fuses.

To prevent unexpected writes due to software bugs, write access to the RC80MCR register is protected by a locking mechanism. For details refer to Section 13.7.7 “Unlock Register” on page 229. The RC80M is automatically switched off in certain sleep modes to reduce power consumption, as described in Section 10. “Power Manager (PM)” on page 109.

13.6.7 Generic Clock Prescalers

Rev: 1.0.2.0

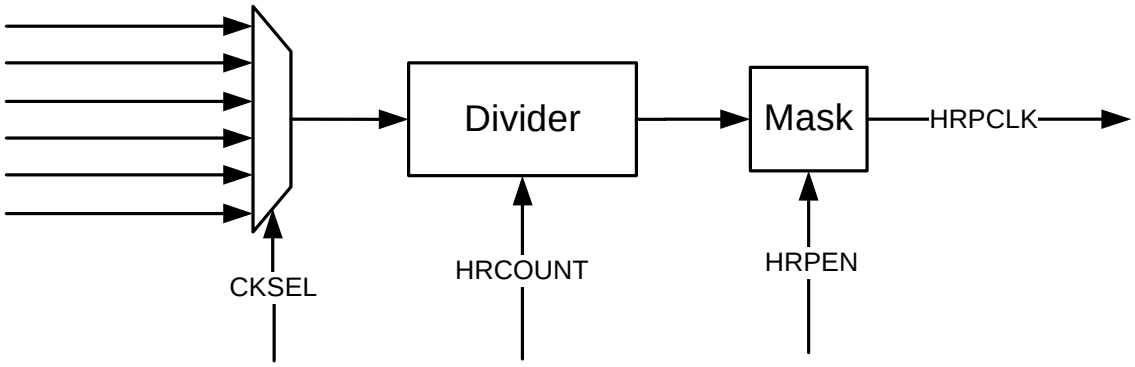
The generic clocks can be sourced by two special prescalers to increase the generic clock frequency precision.

These prescalers are named the High Resolution Prescaler (HRP) and the Fractional Prescaler (FP).

13.6.7.1 High Resolution Prescaler

The HRP is a 24-bit counter that can generate a very accurate clock waveform. The clock obtained has 50% duty cycle.

Figure 13-8. High Resolution Prescaler Generation



The HRP is enabled by writing a one to the High Resolution Prescaler Enable (HRPEN) bit in the High Resolution Prescaler Control Register (HRPCR).

The user can select a clock source for the HRP by writing to the Clock Selection (CKSEL) field of the HRPCR register.

The user must configure the High Resolution Prescaler Clock (HRPCLK) frequency by writing to the High Resolution Count (HRCOUNT) field of the High Resolution Counter (HRPCR) register. This results in the output frequency:

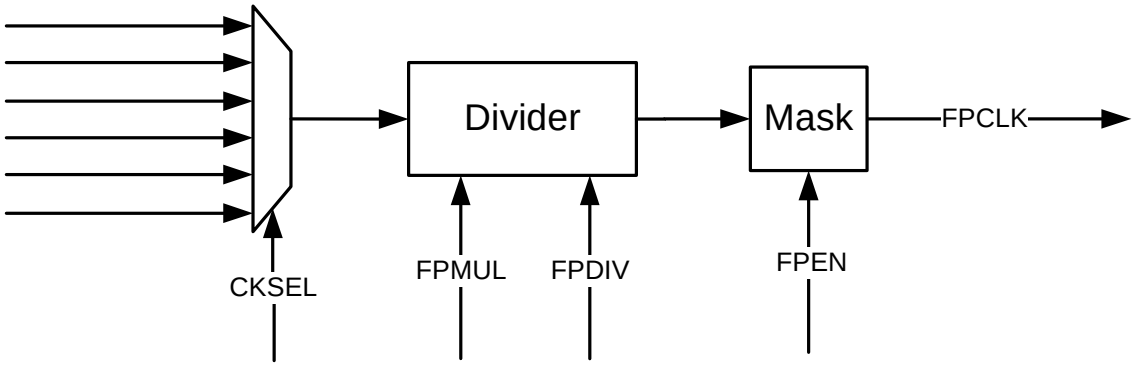
$$f_{HRPCLK} = f_{SRC} / (2*(HRCOUNT+1))$$

The CKSEL field can not be changed dynamically but the HRCOUNT field can be changed on-the-fly.

13.6.7.2 Fractional Prescaler

The FP generates a clock whose average frequency is more precise than the HRP. However, this clock frequency is subject to jitter around the target clock frequency. This jitter influence can be decreased by dividing this clock with the GCLK divider. Moreover the duty cycle of this clock is not precisely 50%.

Figure 13-9. Fractional Prescaler Generation



The FP is enabled by writing a one to the FPEN bit in the Fractional Prescaler Control Register (FPCR).

The user can select a clock source for the FP by writing to the CKSEL field of the FPCR register.

The user must configure the FP frequency by writing to the FPMUL and FPDIV fields of the FPMUL and FPDIV registers. FPMUL and FPDIV must not be equal to zero and FPDIV must be greater or equal to FPMUL. This results in the output frequency:

$$f_{FPCLK} = f_{SRC} * FPMUL/ (2*FPDIV)$$

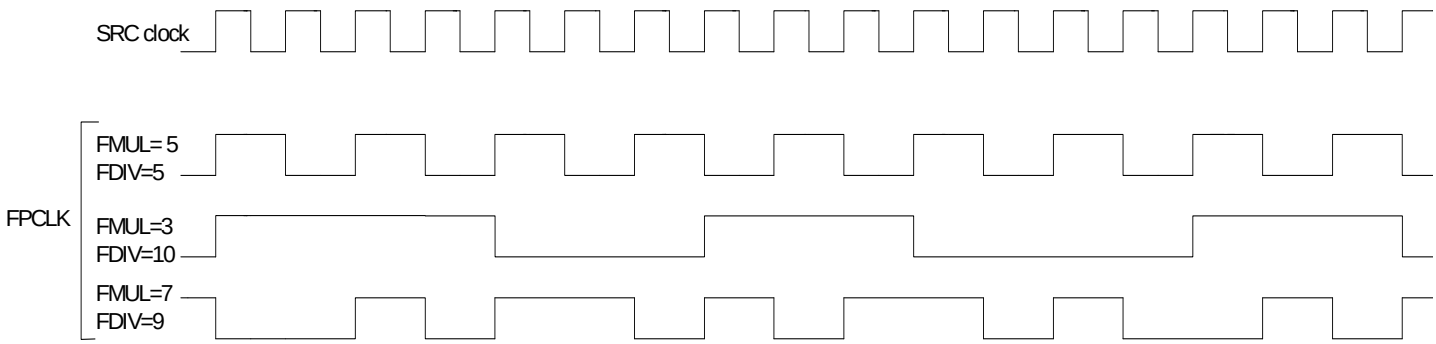
The CKSEL field can not be changed dynamically but the FPMUL and FPDIV fields can be changed on-the-fly.

- Jitter description

As described in Figure 13-10, the CLKFP half period lengths are integer multiples of the source clock period but are not always equals. However the difference between the low level half period length and the high level half period length is at the most one source clock period.

This induces when FPDIV is not an integer multiple of FPMUL a jitter on the FPCLK. The more the FPCLK frequency is low, the more the jitter incidence is reduced.

Figure 13-10. Fractional Prescaler Jitter Examples



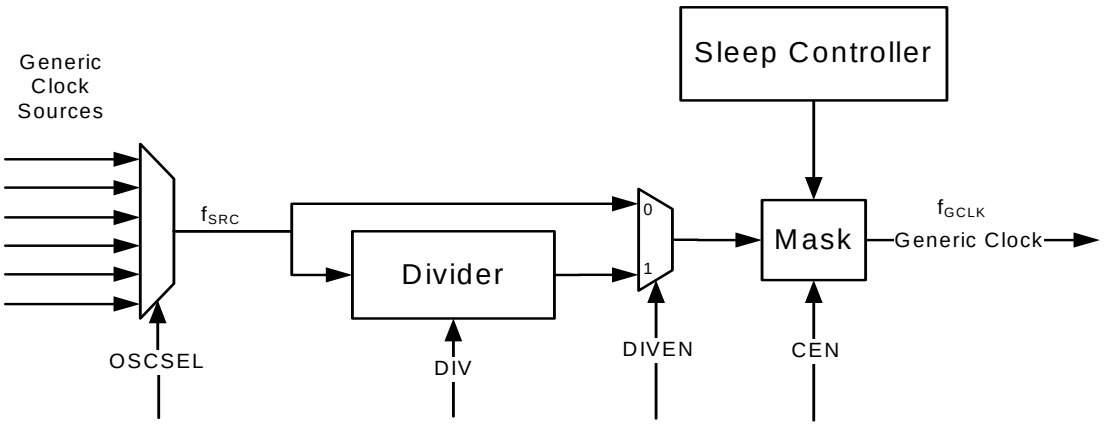
13.6.8 Generic Clocks

Rev: 1.1.2.0

Timers, communication modules, and other modules connected to external circuitry may require specific clock frequencies to operate correctly. The SCIF defines a number of generic clocks that can provide a wide range of accurate clock frequencies.

Each generic clock runs from either clock source listed in the “Generic Clock Sources” table in the SCIF Module Configuration section. The selected source can optionally be divided by any even integer up to 512. Each clock can be independently enabled and disabled, and is also automatically disabled along with peripheral clocks by the Sleep Controller in the Power Manager.

Figure 13-11. Generic Clock Generation



13.6.8.1 *Enabling a Generic Clock*

A generic clock is enabled by writing a one to the Clock Enable bit (CEN) in the Generic Clock Control Register (GCCTRL). Each generic clock can individually select a clock source by writing to the Oscillator Select field (OSCSEL). The source clock can optionally be divided by writing a one to the Divide Enable bit (DIVEN) and the Division Factor field (DIV), resulting in the output frequency:

$$f_{GCLK} = \frac{f_{SRC}}{2(DIV + 1)}$$

where f_{SRC} is the frequency of the selected source clock, and f_{GCLK} is the output frequency of the generic clock.

13.6.8.2 *Disabling a Generic Clock*

A generic clock is disabled by writing a zero to CEN or entering a sleep mode that disables the PB clocks. In either case, the generic clock will be switched off on the first falling edge after the disabling event, to ensure that no glitches occur. After CEN has been written to zero, the bit will still read as one until the next falling edge occurs, and the clock is switched off. When writing a zero to CEN the other bits in GCCTRL should not be changed until CEN reads as zero, to avoid glitches on the generic clock. The generic clocks will be automatically re-enabled when waking from sleep.

13.6.8.3 *Changing Clock Frequency*

When changing the generic clock frequency by changing OSCSEL or DIV, the clock should be disabled before being re-enabled with the new clock source or division setting. This prevents glitches during the transition.

13.6.8.4 Generic Clock Allocation

The generic clocks are allocated to different functions as shown in the “Generic Clock Allocation” table in the SCIF Module Configuration section.

13.6.9 Interrupts

The SCIF has the following interrupt sources:

- AE - Access Error:
  - A protected SCIF register was accessed without first being correctly unlocked.
- RCFASTLOCKLOST - RCFASTLock Lost
  - A to 1 transition on the PCLKSR.RCFASTLOCKLOST bit is detected.
- RCFASTLOCK - RCFAST Lock
  - A 0 to 1 transition on the PCLKSR.RCFASTLOCK bit is detected.
- PLLLOCKLOST - PLL Lock Lost
  - A to 1 transition on the PCLKSR.PLLLOCKLOST bit is detected.
- PLLLOCK - PLL Lock
  - A 0 to 1 transition on the PCLKSR.PLLLOCK bit is detected.
- DFLL0RCS - DFLL Reference Clock Stopped:
  - A 0 to 1 transition on the PCLKSR.DFLLRCS bit is detected.
- DFLL0RDY - DFLL Ready:
  - A 0 to 1 transition on the PCLKSR.DFLLRDY bit is detected.
- DFLL0LOCKF - DFLL Locked on Fine value:
  - A 0 to 1 transition on the PCLKSR.DFLLLOCKF bit is detected.
- DFLL0LOCKC - DFLL Locked on Coarse value:
  - A 0 to 1 transition on the PCLKSR.DFLLLOCKC bit is detected.
- OSCRDY - OSCReady:
  - A 0 to 1 transition on the PCLKSR.OSCRDY bit is detected.

The interrupt sources will generate an interrupt request if the corresponding bit in the Interrupt Mask Register is set. The interrupt sources are ORed together to form one interrupt request. The SCIF will generate an interrupt request if at least one of the bits in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in the Interrupt Status Register (ISR) is cleared by writing a one to the corresponding bit in the Interrupt Clear Register (ICR). Because all the interrupt sources are ORed together, the interrupt request from the SCIF will remain active until all the bits in ISR are cleared.

13.7 User Interface

Table 13-3. SCIF Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0000	Interrupt Enable Register	IER	Write Only	0x00000000
0x0004	Interrupt Disable Register	IDR	Write Only	0x00000000
0x0008	Interrupt Mask Register	IMR	Read Only	0x00000000
0x000C	Interrupt Status Register	ISR	Read Only	0x00000000
0x0010	Interrupt Clear Register	ICR	Write Only	0x00000000
0x0014	Power and Clocks Status Register	PCLKSR	Read Only	0x00000000
0x0018	Unlock Register	UNLOCK	Write Only	0x00000000
0x001C	Chip Specific Configuration Register	CSCR	Read/Write	0x00000000
0x0020	Oscillator Control Register	OSCCTRL0	Read/Write	0x00000000
0x0024	PLL0 Control Register	PLL0	Read/Write	0x00000000
0x0028	DFLL0 Config Register	DFLL0CONF	Read/Write	0x00000000
0x002C	DFLL Value Register	DFLL0VAL	Read/Write	0x00000000
0x0030	DFLL0 Multiplier Register	DFLL0MUL	Read/Write	0x00000000
0x0034	DFLL0 Step Register	DFLL0STEP	Read/Write	0x00000000
0x0038	DFLL0 Spread Spectrum Generator Control Register	DFLL0SSG	Read/Write	0x00000000
0x003C	DFLL0 Ratio Register	DFLL0RATIO	Read Only	0x00000000
0x0040	DFLL0 Synchronization Register	DFLL0SYNC	Write Only	0x00000000
0x0044	System RC Oscillator Calibration Register	RCCR	Read/Write	-(2)
0x0048	4/8/12MHz RC Oscillator Configuration Register	RCFASTCFG	Read/Write	0x00000000
0x004C	4/8/12MHz RC Oscillator Status Register	RCFASTSR	Read Only	0x00000000
0x0050	80MHz RC Oscillator Register	RC80MCR	Read/Write	0x00000000
0x0064	High Resolution Prescaler Control Register	HRPCR	Read/Write	
0x0068	Fractional Prescaler Control Register	FPCR	Read/Write	
0x006C	Fractional Prescaler Multiplier Register	FPMUL	Read/Write	
0x0070	Fractional Prescaler DIVIDER Register	FPDIV	Read/Write	
0x0074	Generic Clock Control0	GCCTRL0	Read/Write	0x00000000
0x0078	Generic Clock Control1	GCCTRL1	Read/Write	0x00000000
0x007C	Generic Clock Control2	GCCTRL2	Read/Write	0x00000000
0x0080	Generic Clock Control3	GCCTRL3	Read/Write	0x00000000
0x0084	Generic Clock Control4	GCCTRL4	Read/Write	0x00000000
0x0088	Generic Clock Control5	GCCTRL5	Read/Write	0x00000000
0x008C	Generic Clock Control6	GCCTRL6	Read/Write	0x00000000

Table 13-3. SCIF Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0090	Generic Clock Control7	GCCTRL7	Read/Write	0x00000000
0x0094	Generic Clock Control8	GCCTRL8	Read/Write	0x00000000
0x0098	Generic Clock Control9	GCCTRL9	Read/Write	0x00000000
0x009C	Generic Clock Control10	GCCTRL10	Read/Write	0x00000000
0x00A0	Generic Clock Control11	GCCTRL11	Read/Write	0x00000000
0x03D8	4/8/12MHz RC Oscillator Version Register	RCFASTVERSION	Read-only	-(1)
0x03DC	Generic Clock Prescaler Version Register	GCLKPRESCVERSION	Read-only	-(1)
0x03E0	PLL Version Register	PLLIFAVERSION	Read-only	-(1)
0x03E4	Oscillator0 Version Register	OSCIFAVERSION	Read-only	-(1)
0x03E8	DFLL Version Register	DFLLIFBVERSION	Read-only	-(1)
0x03EC	System RC Oscillator Version Register	RCOSCIFAVERSION	Read-only	-(1)
0x03F4	80MHz RC Oscillator Version Register	RC80MVERSION	Read-only	-(1)
0x03F8	Generic Clock Version Register	GCLKVERSION	Read-only	-(1)
0x03FC	SCIF Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value is device specific. Refer to the Module Configuration section at the end of this chapter.
2. The reset value of this register depends on factory calibration.

13.7.1 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x0000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-		-	-		-	-	-

15	14	13	12	11	10	9	8
-	RCFASTLOCKLOST	RCFASTLOCK	-	-	-	-	-

7	6	5	4	3	2	1	0
PLL0LOCKLOST	PLL0LOCK	-	DFLL0RCS	DFLL0RDY	DFLL0LOCKF	DFLL0LOCKC	OSC0RDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

13.7.2 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x0004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	RCFASTLOCKLOST	RCFASTLOCK	-	-	-	-	-

7	6	5	4	3	2	1	0
PLL0LOCKLOST	PLL0LOCK	-	DFLL0RCS	DFLL0RDY	DFLL0LOCKF	DFLL0LOCKC	OSC0RDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

13.7.3 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x0008

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	RCFASTLOCKLOST	RCFASTLOCK	-	-	-	-	-

7	6	5	4	3	2	1	0
PLL0LOCKLOST	PLL0LOCK	-	DFLL0RCS	DFLL0RDY	DFLL0LOCKF	DFLL0LOCKC	OSC0RDY

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

13.7.4 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x000C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	RCFASTLOCKLOST	RCFASTLOCK	-	-	-	-	-

7	6	5	4	3	2	1	0
PLL0LOCKLOST	PLL0LOCK	-	DFLL0RCS	DFLL0RDY	DFLL0LOCKF	DFLL0LOCKC	OSC0RDY

0: The corresponding interrupt is cleared.
1: The corresponding interrupt is pending.
A bit in this register is cleared when the corresponding bit in ICR is written to one.
A bit in this register is set when the corresponding interrupt occurs.

13.7.5 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x0010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
AE	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	RCFASTLOCKLOST	RCFASTLOCK	-	-	-	-	-

7	6	5	4	3	2	1	0
PLL0LOCKLOST	PLL0LOCK	-	DFLL0RCS	DFLL0RDY	DFLL0LOCKF	DFLL0LOCKC	OSC0RDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR.

13.7.6 Power and Clocks Status Register

Name: PCLKSR

Access Type: Read-only

Offset: 0x0014

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-		-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	RCFASTLOCKLOST	RCFASTLOCK	-	-	-	-	-

7	6	5	4	3	2	1	0
PLL0LOCKLOST	PLL0LOCK	-	DFLL0RCS	DFLL0RDY	DFLL0LOCKF	DFLL0LOCKC	OSC0RDY

- **RCFASTLOCKLOST: RCFAST lock lost value**
  0: RCFAST has not lost it's lock or has never been enabled.
  1: RCFAST has lost it's lock.
- **RCFASTLOCK: RCFAST Locked on Accurate value**
  0: RCFAST is not locked on accurate value.
  1: RCFAST is locked on accurate value, the tuning procedure is completed and the current count is within the RCFASTCFG.LOCKMARGIN of the target count.
- **PLL0LOCKLOST: PLL0 lock lost value**
  0: PLL0 has not lost it's lock or has never been enabled.
  1: PLL0 has lost it's lock, either by disabling the PLL0 or due to faulty operation.
- **PLL0LOCK: PLL0 Locked on Accurate value**
  0: PLL0 is unlocked on accurate value.
  1: PLL0 is locked on accurate value, and is ready to be selected as clock source with an accurate output clock.
- **DFLL0RCS: DFLL0 Reference Clock Stopped**
  0: The DFLL reference clock is running, or has never been enabled.
  1: The DFLL reference clock has stopped or is too slow.
- **DFLL0RDY: DFLL0 Synchronization Ready**
  0: Read or write to DFLL registers is invalid.
  1: Read or write to DFLL registers is valid.
- **DFLL0LOCKF: DFLL0 Locked on Fine Value**
  0: DFLL is unlocked on Fine value.
  1: DFLL is locked on Fine value, and is ready to be selected as clock source with a high accuracy on the output clock.
- **DFLL0LOCKC: DFLL0 Locked on Coarse Value**
  0: DFLL is unlocked on Coarse value.

1: DFLL is locked on Coarse value, and is ready to be selected as clock source with medium accuracy on the output clock.

- **OSC0RDY: OSC0 Ready**
  0: Oscillator not enabled or not ready.
  1: Oscillator is stable and ready to be used as clock source.

13.7.7 Unlock Register

Name: UNLOCK

Access Type: Write-only

Offset: 0x0018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	ADDR[9:8]	

7	6	5	4	3	2	1	0
ADDR[7:0]							

To unlock a write protected register, first write to the UNLOCK register with the address of the register to unlock in the ADDR field and 0xAA in the KEY field. Then, in the next PB access write to the register specified in the ADDR field.

- **KEY: Unlock Key**
  Write this bit field to 0xAA to enable unlock.
- **ADDR: Unlock Address**
  Write the address offset of the register to unlock to this field.

13.7.8 Oscillator Control Register

Name: OSCCTRLn

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	OSCEN

15	14	13	12	11	10	9	8
-	-	-	-	STARTUP[3:0]			

7	6	5	4	3	2	1	0
-	-	-		AGC	GAIN[1:0]		MODE

- **OSCEN: Oscillator Enable**
  0: The oscillator is disabled.
  1: The oscillator is enabled.
- **STARTUP: Oscillator Start-up Time**
  Select start-up time for the oscillator. Refer to the “Oscillator Startup Time” table in the SCIF Module Configuration section for details.
- **AGC: Automatic Gain Control**
  For test purposes.
- **GAIN: Gain**
  Selects the gain for the oscillator. Refer to the “Oscillator Gain Settings” table in the SCIF Module Configuration section for details.
- **MODE: Oscillator Mode**
  0: External clock connected on XIN. XOUT can be used as general-purpose I/O (no crystal).
  1: Crystal is connected to XIN/XOUT.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.9 PLL Control Register

Name: PLLn

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	PLLCOUNT					

23	22	21	20	19	18	17	16
-	-	-	-	PLLMUL			

15	14	13	12	11	10	9	8
-	-	-	-	PLLDIV			

7	6	5	4	3	2	1	0
-	-	PLLOPT			PLLOSC		PLLEN

- **PLLCOUNT: PLL Count**
  Specifies the number of RCSYS clock cycles before ISR.PLLLOCKn will be set after PLLn has been written, or after PLLn has been automatically re-enabled after exiting a sleep mode.
- **PLLMUL: PLL Multiply Factor**
- **PLLDIV: PLL Division Factor**
  These fields determine the ratio of the PLL output frequency to the source oscillator frequency:
  f_{vco} = (PLLMUL+1)/PLLDIV • f_{REF} if PLLDIV >0
  f_{vco} = 2•(PLLMUL+1) • f_{REF} if PLLDIV = 0
  Note that the PLLMUL field should always be greater than 1 or the behavior of the PLL will be undefined.
- **PLLOPT: PLL Option**
  PLLOPT[0]: Selects the VCO frequency range (f_{vco}).
  0: 80MHz<f_{vco}<180MHz
  1: 160MHz<f_{vco}<240MHz
  PLLOPT[1]: Divides the output frequency by 2.
  0: f_{PLL}= f_{vco}
  1: f_{PLL} = f_{vco}/2
  PLLOPT[2]:Wide-Bandwidth mode.
  0: Wide Bandwidth Mode enabled.
  1: Wide Bandwidth Mode disabled.
- **PLLOSC: PLL Oscillator Select**
  Reference clock source select for the reference clock, refer to the “PLL Clock Sources” table in the SCIF Module Configuration section for details.

- **PLLEN: PLL Enable**

  0: PLL is disabled.

  1: PLL is enabled.

  Note that it is not possible to change any of the PLL configuration bits when the PLL is enabled, Any write to PLLn while the PLL is enabled will be discarded.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.10 DFLLx Configuration Register

Name: DFLLxCONF

Access Type: Read/Write

Reset Value: 0x0X100000[1]

31	30	29	28	27	26	25	24
-	-	-	-	CALIB			

23	22	21	20	19	18	17	16
FCD	-	-	-	-	-	RANGE	

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	QLDIS	CCDIS	-	LLAW	STABLE	MODE	EN

Note: 1. The reset value of CALB depends on factory calibration.

- **CALIB: Calibration Value**
  Sets the Calibration Value for the DFLLx.
- **FCD: Fuse Calibration Done**
  Set to 1 when CALIB field has been updated by the Flash fuses after a reset.
  0: The flash calibration will be redone after any reset
  1: The flash calibration will only be redone after a power-on reset.
- **RANGE: Range Value**
  Set the value of the range calibration register. Refer to the table DFLL Frequency Range in the Open loop operation section.
- **QLDIS: Quick Lock Disable**
  0: Quick Lock is enabled.
  1: Quick Lock is disabled.
- **CCDIS: Chill Cycle Disable**
  0: Chill Cycle is enabled.
  1: Chill Cycle is disabled.
- **LLAW: Lose Lock After Wake**
  0: Locks will not be lost after waking up from sleep modes.
  1: Locks will be lost after waking up from sleep modes where the DFLL clock has been stopped.
- **STABLE: Stable DFLL Frequency**
  0: FINE calibration tracks changes in output frequency.
  1: FINE calibration register value will be fixed after fine lock.
- **MODE: Mode Selection**
  0: DFLL is in Open Loop operation.
  1: DFLL is in Closed Loop operation.
- **EN: Enable**
  0: DFLL is disabled.
  1: DFLL is enabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.11 DFLLx Value Register

Name: DFLLxVAL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	COARSE				

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
FINE							

- **COARSE: Coarse Value**
  Set the value of the coarse calibration register. In closed loop mode, this field is read-only.
- **FINE: Fine value**
  Set the value of the fine calibration register. In closed loop mode, this field is read-only.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.12 DFLLx Multiplier Register

Name: DFLLxMUL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
MUL[15:8]							

7	6	5	4	3	2	1	0
MUL[7:0]							

- **MUL: DFLL Multiply Factor**
  This field determines the ratio of the CLK_DFLLx output frequency to the CLK_DFLLx_REF input frequency.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.13 DFLLx Maximum Step Register

Name: DFLLxSTEP

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	CSTEP				

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
FSTEP							

- **CSTEP: Coarse Maximum Step**
  This indicates the maximum step size during coarse adjustment in closed loop mode. When adjusting to a new frequency, the expected overshoot of that frequency depends on this step size.
- **FSTEP: Fine Maximum Step**
  This indicates the maximum step size during fine adjustment in closed loop mode. When adjusting to a new frequency, the expected overshoot of that frequency depends on this step size.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.14 DFLLx Spread Spectrum Generator Control Register

Name: DFLLxSSG

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	STEPSIZE				

15	14	13	12	11	10	9	8
-	-	-	AMPLITUDE				

7	6	5	4	3	2	1	0
-	-	-	-	-	-	PRBS	EN

- **STEPSIZE: SSG Step Size**
  Selects the step size of the spread spectrum. If zero or one, the value added to the FINE bits will be incremented/decremented with one on every positive edge of the input clock. If $n > 1$, the value added to the FINE bits will be incremented/decremented with n on every positive edge of the input clock.
- **AMPLITUDE: SSG Amplitude**
  Selects the amplitude of the spread spectrum. If zero, only the LSB of the FINE bits will be affected. If one, the sequence {1, 0, -1, 0} will be added to FINE bits, etc.
- **PRBS: Pseudo Random Bit Sequence**
  0: Each spread spectrum frequency is applied at constant intervals
  1: Each spread spectrum frequency is applied at pseudo-random intervals
- **EN: Enable**
  0: SSG is disabled.
  1: SSG is enabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.15 DFLLx Ratio Register

Name: DFLLxRATIO

Access Type: Read-only

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
RATIODIFF[15:8]							

7	6	5	4	3	2	1	0
RATIODIFF[7:0]							

- **RATIODIFF: Multiplication Ratio Difference**
  In closed loop mode, this field indicates the error in the ratio between the CLK_DFLLx frequency and the target frequency. This value is not updated in open loop mode, and should be considered invalid in that case.

13.7.16 DFLLx Synchronization Register

Name: DFLLxSYNC

Access Type: Write-only

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	SYNC

- **SYNC: Synchronization**
  To be able to read the current value of DFLLxVAL or DFLLxRATIO, this bit must be written to one. The updated value are available in DFLLxVAL or DFLLxRATIO when PCLKSR.DFLL0RDY is set.

13.7.17 System RC Oscillator Calibration Register

Name: RCCR

Access Type: Read/Write

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	FCD

15	14	13	12	11	10	9	8
-	-	-	-	-	-	CALIB[9:8]	

7	6	5	4	3	2	1	0
CALIB[7:0]							

- **FCD: Flash Calibration Done**
  0: The flash calibration will be redone after any reset.
  1: The flash calibration will only be redone after a Power-on Reset.
  This bit is cleared after a POR.
  This bit is set when the CALIB field has been updated by the flash fuses after a reset.
- **CALIB: Calibration Value**
  Calibration Value for the System RC oscillator (RCSYS).

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.18 4/8/12MHz RC Oscillator Configuration Register

Name: RCFASTCFG

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	CALIB						

15	14	13	12	11	10	9	8
LOCKMARGIN				-	-	FRANGE	

7	6	5	4	3	2	1	0
FCD	NBPERIODS			-	JITMODE	TUNEEN	EN

- **CALIB: Oscillator Calibration Value**
  Writing a value to this field sets the oscillator trim value. A small value will produce a fast RCFAST clock and a big value will produce a slow clock. If the tuner is enabled, writing to this field will cause the tuner to start tuning from the written value.
- **LOCKMARGIN: Accepted Count Error for Lock**
  Writing a value to this field defines the maximum absolute value of the current count error with respect to the theoretical count that will be trigger the RCFASTLOCK condition:
  If abs(Count - Count_theor) <= LOCKMARGIN, then RCFASTLOCK is set, otherwise RCFASTLOCKLOST is set, with Count_theor = 122 x (1 + FRANGE) << NBPERIODS.
- **FRANGE: Frequency Range**
  00: 4MHz range selected.
  01: 8MHz range selected.
  10: 12MHz range selected.
  11: Reserved.
- **FCD: RCFAST Fuse Calibration Done**
  Always read as 1 (read only).
  Set to 1 when the CALIB field has been updated by the Flash Fuses after a reset (read only).
  0: Calibration from Flash Fuses has not been loaded. Calibration from Flash Fuses will be redone after any reset.
  1: Calibration from Flash Fuses has been loaded. Calibration from Flash Fuses will only be redone after a Power-on Reset.
- **NBPERIODS: Number of 32kHz Periods**
  Writing a value to this field defines the number of periods of the 32kHz clock used to compare the RCFAST clock output. A higher value increases the tuner precision and smoothes the calibration adjustments.
- **JITMODE: Jitter Mode**
  0: The RCFAST trim value is only updated when the RCFASTLOCKLOST is set; it is kept stable as long as the RCFASTLOCK condition is set.
  1: The RCFAST trim value is continuously updated.
- **TUNEEN: Tuner Enable**
  0: Tuner is not enabled, and the RCFAST operates in open loop mode.
  1: Tuner is enabled, and the RCFAST operates in closed loop mode.
- **EN: Oscillator Enable**
  0: The RCFAST clock is disabled, and the output clock is stopped.

1: The RCFAST clock is enabled, and the output clock is running.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Refer to Section 13.7.7 “Unlock Register” on page 229 for details.

13.7.19 4/8/12MHz RC Oscillator Status Register

Name: RCFASTSR

Access Type: Read-only

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
UPDATED	-	-	-	-	-	LOCKLOST	LOST

23	22	21	20	19	18	17	16
-	-	SIGN	CNTERR				

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	CURTRIM						

- **UPDATED: Current Trim Value Updated**
  This field toggles each time the tuning procedure completes.
- **LOCKLOST: Lock Lost**
  This field indicates that the current count is outside RCFASTCFG.LOCKMARGIN from the theoretical count. It is updated after each tuning procedure.
- **LOCK: Lock**
  This field indicates that the current count is within RCFASTCFG.LOCKMARGIN of the theoretical count. It is updated after each tuning procedure.
- **SIGN: Sign of Current Count Error**
  This field returns the sign of the current difference between the current count and the theoretical count. It is updated after each tuning procedure.
- **CNTERR: Current Count Error**
  This field returns the current absolute difference between the current count and the theoretical count. It is updated after each tuning procedure.
- **CURTRIM: Current Trim Value**
  This field returns the current trim value used by the RCFAST oscillator. It is updated after each tuning procedure.

13.7.20 80MHz RC Oscillator Control Register

Name: RC80MCR

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-		

23	22	21	20	19	18	17	16
-	-	-	-	-	-	CALIB	

15	14	13	12	11	10	9	8
-	-	-	-	-	-		

7	6	5	4	3	2	1	0
FCD			-	-	-	-	EN

- **CALIB: Calibration Value**
  Calibration Value for the RC oscillator (read-only).
- **FCD: Flash Calibration Done**
- **EN: Enable**
  0: The oscillator is disabled.
  1: The oscillator is enabled.

13.7.21 High Resolution Prescaler Control Register

Name: HRPCR

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
HRCOUNT[23:16]							

23	22	21	20	19	18	17	16
HRCOUNT[15:8]							

15	14	13	12	11	10	9	8
HRCOUNT[7:0]							

7	6	5	4	3	2	1	0
-	-	-	-	CKSEL			HRPEN

- **HRCOUNT: High Resolution Counter**
  Specify the input clock period to count to generate the output clock edge.
  HRCOUNT can be written to dynamically in order to tune the HRPCLK frequency on-the-go.
- **CKSEL: Clock input selection**
  This field selects the Clock input for the prescaler. See the “HRP clock sources” table in the SCIF Module Configuration section for details. It must not be changed if the HRPEN is one.
- **HRPEN: High Resolution Prescaler Enable**
  0: The High Resolution Prescaler is disabled.
  1: The High Resolution Prescaler is enabled.

13.7.22 Fractional Prescaler Control Register

Name: FPCR

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	CKSEL			FPEN

- **CKSEL: Clock input selection**
  This field selects the Clock input for the prescaler. See the “FP clock sources” table in the SCIF Module Configuration section for details. It must not be changed if the FPEN is one.
- **FPEN: High Resolution Prescaler Enable**
  0: The Fractional Prescaler is disabled.
  1: The Fractional Prescaler is enabled.

13.7.23 Fractional Prescaler Mul Register

Name: FPMUL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
FPMUL[15:8]							

7	6	5	4	3	2	1	0
FPMUL[7:0]							

- **FPMUL: Fractional Prescaler Multiplication Factor**

  This field selects the multiplication factor for the prescaler.

  Notice that FPMUL is always smaller than FPDIV. FPMUL can be written to dynamically in order to tune the FPCLK frequency on-the-go.

13.7.24 Fractional Prescaler Div Register

Name: FPDIV

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
FPDIV[15:8]							

7	6	5	4	3	2	1	0
FPDIV[7:0]							

- **FPDIV: Fractional Prescaler Division Factor**

  This field selects the division factor for the prescaler.

  Notice that FPMUL must be smaller than FPDIV. FPDIV can be written to dynamically in order to tune the FPCLK frequency on-the-go.

13.7.25 Generic Clock Control

Name: GCCTRL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DIV[15:8]							
23	22	21	20	19	18	17	16
DIV[7:0]							
15	14	13	12	11	10	9	8
-	-	-	OSCSEL[4:0]				
7	6	5	4	3	2	1	0
-	-	-	-	-	-	DIVEN	CEN

There is one GCCTRL register per generic clock in the design.

- **DIV: Division Factor**
  The number of DIV bits for each generic clock is as shown in the “Generic Clock number of DIV bits” table in the SCIF Module Configuration section.
- **OSCSEL: Oscillator Select**
  Selects the source clock for the generic clock. Refer to the “Generic Clock Sources” table in the SCIF Module Configuration section.
- **DIVEN: Divide Enable**
  0: The generic clock equals the undivided source clock.
  1: The generic clock equals the source clock divided by 2*(DIV+1).
- **CEN: Clock Enable**
  0: The generic clock is disabled.
  1: The generic clock is enabled.

13.7.26 4/8/12MHz RC Oscillator Version Register

Name: RCFASTVERSION

Access Type: Read-only

Offset: 0x03D8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.27 Generic Clock Prescalers Version Register

Name: GCLKPRESCVERSION

Access Type: Read-only

Offset: 0x03DC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.28 PLL Version Register

Name: PLLIFAVERSION

Access Type: Read-only

Offset: 0x03E0

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.29 Oscillator 0 Version Register

Name: OSCIFAVERSION

Access Type: Read-only

Offset: 0x03E4

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.30 Digital Frequency Locked Loop Version Register

Name: DFLLIFBVERSION

Access Type: Read-only

Offset: 0x03E8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.31 RC Oscillator Version Register

Name: RCOSCIFAVERSION

Access Type: Read-only

Offset: 0x03EC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.32 80MHz RC Oscillator Version Register

Name: RC80MVERSION

Access Type: Read-only

Offset: 0x03F4

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.33 Generic Clock Version Register

Name: GCLKIFVERSION

Access Type: Read-only

Offset: 0x03F8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.7.34 SCIF Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x03FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:0]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

13.8 Module Configuration

The specific configuration for each SCIF instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 13-4. SCIF Clock Name

Module Name	Clock Name	Description
SCIF	CLK_SCIF	Clock for the SCIF bus interface

Table 13-5. Oscillator Startup Times

STARTUP	Number of System RC oscillator clock cycle	Approximative Equivalent time (RCSYS = 115 kHz)
0	0	0
1	64	557 us
2	128	1.1 ms
3	2048	18 ms
4	4096	36 ms
5	8192	71 ms
6	16384	143 ms
7	32768	285 ms
8	4	35 us
9	8	70 us
10	16	139 us
11	32	278 us
12	256	2.2 ms
13	512	4.5 ms
14	1024	8.9 ms
15	32768	285 ms

Table 13-6. Oscillator Gain Settings

GAIN[1:0]	Function
0	Oscillator is used with gain G0 (XIN from 0.6MHz to 2.0MHz)
1	Oscillator is used with gain G1 (XIN from 2.0MHz to 4.0MHz)
2	Oscillator is used with gain G2 (XIN from 4.0MHz to 8.0MHz)
3	Oscillator is used with gain G3 (XIN from 8.0MHz to 16.0MHz)
4	Oscillator is used with gain G4 (XIN from 16.0MHz to 30.0MHz)

In ATSAM4L8/L4/L2, there are 12 generic clocks. These are allocated to different functions as shown in Table 13-7.

Table 13-7. Generic Clock Allocation

Clock number	Function
0	DFLLIF main reference and GCLK0 pin (CLK_DFLLIF_REF)
1	DFLLIF dithering and SSG reference and GCLK1 pin (CLK_DFLLIF_DITHER)
2	AST and GCLK2 pin
3	CATB and GCLK3 pin
4	AESA
5	GLOC, TC0 and RC32KIFB_REF
6	ABDACB and IISC
7	USBC
8	TC1 and PEVC[0]
9	PLL0 and PEVC[1]
10	ADCIFE
11	Master generic clock. Can be used as source for other generic clocks.

Table 13-8. Generic Clock Sources

OSCSEL	Clock/Oscillator	Description
0	RCSYS	System RC oscillator clock
1	OSC32K	Output clock from OSC32K
2	DFLL0	Output clock from DFLL0
3	OSC0	Output clock from Oscillator0
4	RC80M	Output from 80MHz RCOSC
5	RCFAST	Output from 4,8,12MHz RCFAST
6	RC1M	Output from 1MHz RC1M
7	CLK_CPU	The clock the CPU runs on
8	CLK_HSB	High Speed Bus clock
9	CLK_PBA	Peripheral Bus A clock
10	CLK_PBB	Peripheral Bus B clock
11	CLK_PBC	Peripheral Bus C clock
12	CLK_PBD	Peripheral Bus D clock
13	RC32K	Output from 32kHz RCOSC
14	reserved	

Table 13-8. Generic Clock Sources

OSCSEL	Clock/Oscillator	Description
15	CLK_1K	1kHz output clock from OSC32K
16	PLL0	Output clock from PLL0
17	HRP	High Resolution Prescaler Output
18	FP	Fractionnal Prescaler Output
19-20	GCLK_IN[0-1]	GCLK_IN[0-1] pins, digital clock input
21	GCLK11	Generic Clock 11. Can not be use as input to itself.
22-31	Reserved	

.

Table 13-9. PLL Clock Sources

PLLOSC	Clock/Oscillator	Description
0	OSC0	Output clock from Oscillator0
1	GCLK9	Generic clock 9
2-3	Reserved	

Table 13-10. Generic Clock number of DIV bits

Generic Clock	Number of DIV bits
0	8
1	8
2	8
3	8
4	8
5	8
6	8
7	8
8	8
9	8
10	8
11	16

Table 13-11. HRP and FP Clock Sources

CKSEL	Clock/Oscillator	Description
0	OSC0	Output clock from Oscillator0
1	PLL0	Output clock from PLL0
2	DFLL0	Output clock from DFLL0
3	reserved	
4	RC80M	Output from 80MHz RCOSC

Table 13-12. Register Reset Values

Register	Reset Value
RCFASTVERSION	0x00000200
GCLKPRESCVERSION	0x00000102
PLLIFAVERSION	0x00000112
OSCIFAVERSION	0x00000114
DFLLIFBVERSION	0x00000110
RCOSCIFAVERSION	0x00000114
RC80MVERSION	0x00000100
GCLKIFVERSION	0x00000112
VERSION	0x00000130

14. Flash Controller (FLASHCALW)

Rev: 1.1.0.1

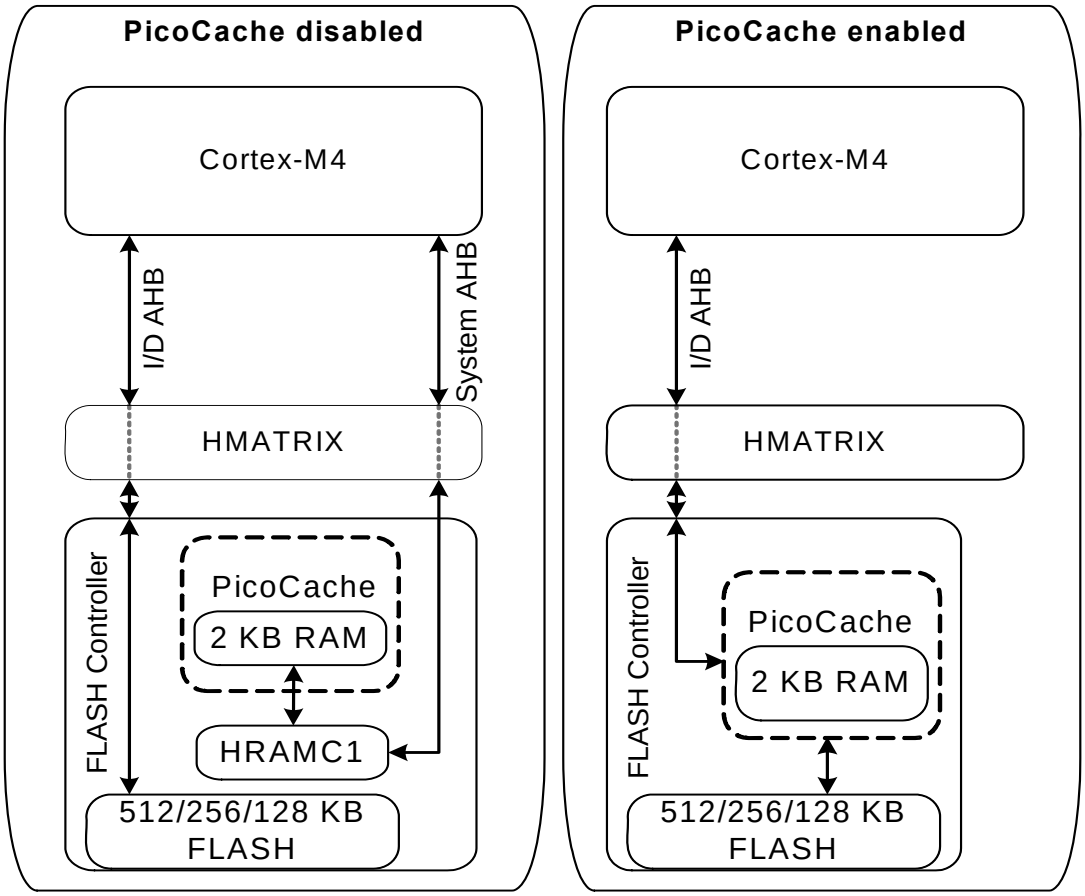
14.1 Features

- **Controls on-chip flash memory**
- **Supports 0 and 1 wait state bus access**
- **Flash Read-Standby mode to achieve very low power flash operation**
- **Unified direct mapped PicoCache to minimize flash active power and improve performance**
- **32-bit HSB interface for reads from flash and writes to page buffer**
- **32-bit PB interface for issuing commands to and configuration of the controller**
- **32-bit HSB interface for reads and writes to the PicoCache RAM when it is disabled**
- **Flash memory is divided into 16 regions that can be individually protected or unprotected**
- **Supports reads and writes of general-purpose Non Volatile Memory (NVM) bits**
- **Supports reads and writes of additional NVM pages**
- **Supports strong device protection**

14.2 Overview

The Flash Controller (FLASHCALW) interfaces the on-chip flash memory with the 32-bit internal HSB bus. The controller manages the reading, writing, erasing, locking, and unlocking sequences. To minimize power consumption, the module is tigthly coupled to a direct mapped cache resulting in a significant decrease of the flash active power consumption and also in a performance increase when running with one wait state.

14.3 Block Diagram



14.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

14.4.1 Power Management

If the CPU enters a Power Save Mode that disables clocks used by the FLASHCALW, the FLASHCALW will stop functioning and resume operation after the system wakes up from Power Save Mode.

Write and Erase operation are not allowed when the device in Power Scaling Configuration 1 (BPM.PMCON.PS=1)

14.4.2 Clocks

The FLASHCALW has three bus clocks connected: Two High Speed Bus clock (CLK_FLASHCALW_AHB and CLK_HRAMC1_AHB) and one Peripheral Bus clock (CLK_FLASHCALW_APB). These clocks are generated by the Power Manager. Both clocks are enabled at reset, and can be disabled by writing to the Power Manager. The user has to ensure that CLK_FLASHCALW_AHB is not turned off before reading the flash or writing the pagebuffer

and that CLK_FLASHCALW_APB is not turned off before accessing the FLASHCALW configuration and control registers. Failing to do so may deadlock the bus.

14.4.3 Interrupts

The FLASHCALW interrupt request lines are connected to the NVIC. Using the FLASHCALW interrupts requires that it is programmed first.

14.4.4 Debug Operation

When an external debugger forces the CPU into debug mode, the FLASHCALW continues normal operation. If the FLASHCALW is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

14.5 Functional Description

14.5.1 Bus Interfaces

The FLASHCALW has three bus interfaces, two High Speed Bus (HSB) interfaces for:

- reads from the flash memory and writes to the page buffer
- read/write in the PicoCache internal RAM when it is disabled

and one Peripheral Bus (PB) interface for issuing commands and reading status from the controller.

14.5.2 Flash Memory Organization

The flash memory is divided into a set of pages. A page is the basic unit addressed when programming the flash. A page consists of several words. The pages are grouped into 16 regions of equal size. _AHBEach of these regions can be locked by a dedicated fuse bit, protecting it from accidental modification.

- *p* pages (*FLASH_P*)
- *w* bytes in each page and in the page buffer (*FLASH_W*)
- *pw* bytes in total (*FLASH_PW*)
- *f* general-purpose fuse bits (*FLASH_F*), used as region lock bits and for other device-specific purposes
- security fuses
- 1 User page

14.5.3 User Page

The User page is an additional page, outside the regular flash array, that can be used to store various data, such as calibration data and serial numbers. This page is not erased by regular chip erase. The User page can only be written and erased by a special set of commands. Read accesses to the User page are performed just as any other read accesses to the flash. The address map of the User page is given in Figure 14-2 on page 269.

14.5.4 Read Operations

The on-chip flash memory is typically used for storing instructions to be executed by the CPU. The CPU will address instructions using the HSB bus, and the FLASHCALW will access the flash memory and return the addressed 32-bit word.

In systems where the HSB clock period is slower than the access time of the flash memory, the FLASHCALW can operate in 0 wait state mode, and output one 32-bit word on the bus per clock cycle. If the clock frequency allows, the user should use 0 wait state mode, because this gives the highest performance as no stall cycles are encountered.

The FLASHCALW can also operate in systems where the HSB bus clock period is faster than the access speed of the flash memory. Wait state support and a read granularity of 64 bits ensure efficiency in such systems.

Performance for systems with high clock frequency is increased since the internal read word width of the flash memory is 64 bits. When a 32-bit word is to be addressed, the word itself and also the other word in the same 64-bit location is read.

The user can select the wait states required by writing to the FWS field in the Flash Control Register (FCR). It is the responsibility of the user to select a number of wait states compatible with the clock frequency and timing characteristics of the flash memory.

In 0ws mode, no wait states are encountered on any flash read operations. In 1 ws mode, one stall cycle is encountered every 64-bit aligned transfer. If the PicoCache is enabled and there is a hit, then it will service the transfer with no penalty.

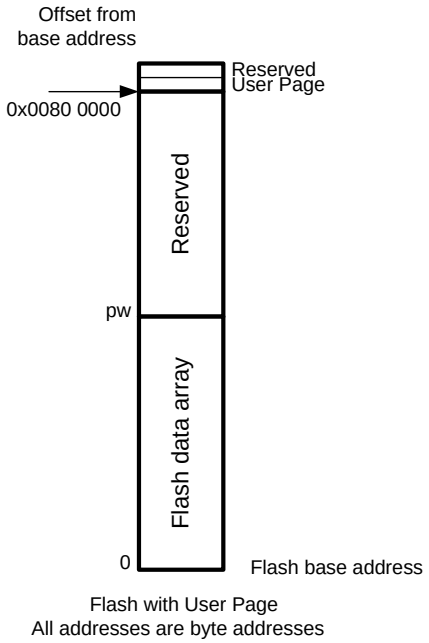
The Flash Controller address space is displayed in Figure 14-1. The memory space between address *pw* and the User page is reserved, and reading addresses in this space returns an

undefined result. The User page is permanently mapped to an offset of 0x00800000 from the start address of the flash memory.

Table 14-1. User Page Addresses

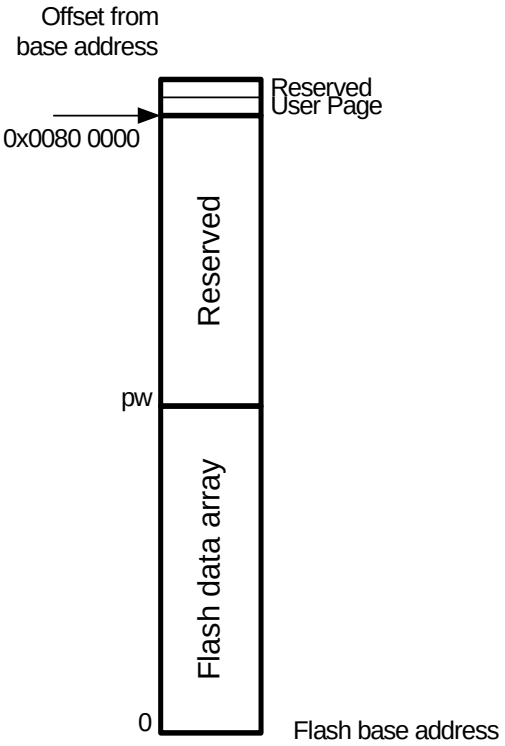
Memory type	Start address, byte sized	Size
Main array	0	*pw* bytes
User	0x00800000	w bytes

Figure 14-1. Memory Map for the FLASH Memories



Flash with User Page
All addresses are byte addresses

Figure 14-2. Memory Map for the Flash Memories

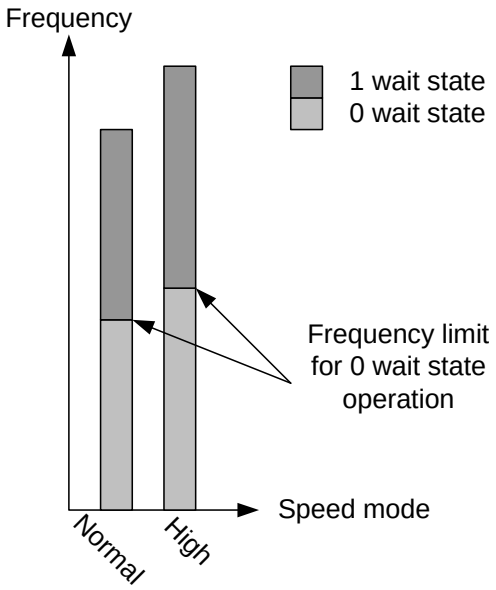


14.5.5 High Speed Read Mode

The flash provides a High Speed Read Mode, offering slightly higher flash read speed at the cost of higher power consumption. Two dedicated commands, High Speed Read Mode Enable (HSEN) and High Speed Read Mode Disable (HSDIS) control the speed mode. The High Speed Mode (HSMODE) bit in the Flash Status Register (FSR) shows which mode the flash is in. After reset, the High Speed Mode is disabled, and must be manually enabled if the user wants to.

Refer to Section 42. ”Electrical Characteristics” on page 1121 at the end of this datasheet for details on the maximum clock frequencies in Normal and High Speed Read Mode.

Figure 14-3. High Speed Mode



14.5.6 Quick Page Read

A dedicated command, Quick Page Read (QPR), is provided to read all words in an addressed page. All bits in all words in this page are AND'ed together, returning a 1-bit result. This result is placed in the Quick Page Read Result (QPRR) bit in Flash Status Register (FSR). The QPR command is useful to check that a page is in an erased state. The QPR instruction is much faster than performing the erased-page check using a regular software subroutine.

14.5.7 Quick User Page Read

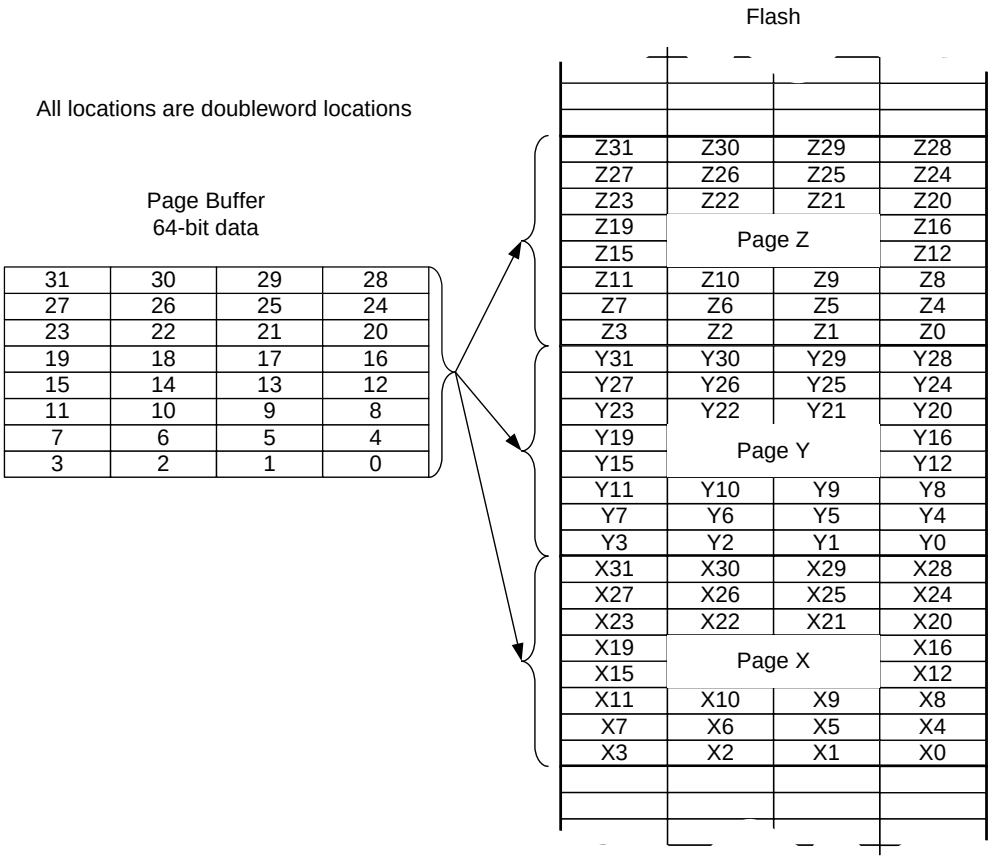
A dedicated command, Quick User Page Read (QPRUP), is provided to read all words in the user page. All bits in all words in this page are AND'ed together, returning a 1-bit result. This result is placed in the Quick Page Read Result (QPRR) bit in Flash Status Register (FSR). The QPRUP command is useful to check that a page is in an erased state. The QPRUP instruction is much faster than performing the erased-page check using a regular software subroutine.

14.5.8 Page Buffer Operations

The flash memory has a write and erase granularity of one page; data is written and erased in chunks of one page. When programming a page, the user must first write the new data into the Page Buffer. The contents of the entire Page Buffer is copied into the desired page in flash memory when the user issues the Write Page command, Refer to Section 14.6.1 on page 276.

In order to program data into flash page Y, write the desired data to locations Y0 to Y31 in the regular flash memory map. Writing to an address A in the flash memory map will not update the flash memory, but will instead update location A%32 in the page buffer. The PAGEN field in the Flash Command (FCMD) register will at the same time be updated with the value A/32.

Figure 14-4. Mapping from Page Buffer to Flash



Internally, the flash memory stores data in 64-bit doublewords. Therefore, the native data size of the Page Buffer is also a 64-bit doubleword. All locations shown in Figure 14-4 are therefore doubleword locations. Since the HSB bus only has a 32-bit data width, two 32-bit HSB transfers must be performed to write a 64-bit doubleword into the Page Buffer. The FLASHCALW has logic to combine two 32-bit HSB transfers into a 64-bit data before writing this 64-bit data into the Page Buffer. This logic requires the word with the low address to be written to the HSB bus before the word with the high address. To exemplify, to write a 64-bit value to doubleword X0 residing in page X, first write a 32-bit word to the byte address pointing to address X0, thereafter write a word to the byte address pointing to address (X0+4).

The page buffer is word-addressable and should only be written with aligned word transfers, never with byte or halfword transfers. The page buffer cannot be read.

The page buffer is also used for writes to the User page.

Page buffer write operations are performed with 4 wait states. Any accesses attempted to the FLASHCALW on the HSB bus during these cycles will be automatically stalled.

Writing to the page buffer can only change page buffer bits from one to zero, i.e. writing 0xAAAAAAAA to a page buffer location that has the value 0x00000000 will not change the page buffer value. The only way to change a bit from zero to one is to erase the entire page buffer with the Clear Page Buffer command.

The page buffer is not automatically reset after a page write. The user should do this manually by issuing the Clear Page Buffer flash command. This can be done after a page write, or before the page buffer is loaded with data to be stored to the flash page.

14.5.9 PicoCache Description

14.5.9.1 Overview

The PicoCache is an unified direct mapped cache controller. It integrates a controller, a tag directory, a data memory, a metadata memory and a configuration interface. When it is not activated, the cache memory is accessible as a supplementary RAM connected on the bus matrix. Note that for security reasons, this memory is cleared by a chip erase operation.

14.5.9.2 Cache Operation

On reset, the cache controller data entries are all invalidated and the cache is disabled. The cache is transparent to processor operations making flash access timings predicatable. The cache controller is activated through the use of its configuration registers. Use the following sequence to enable the cache controller

1. Verify that the cache controller is disabled, reading the value of the CSTS (cache status) field of the SR register.
2. Enable the cache controller writing one to CEN (cache enable) field of the CTRL register.

14.5.9.3 Cache Invalidate By Line Operation

When an invalidate by line command is issued the cache controller reset the valid bit information of the decoded cache line. As the line is no longer valid the replacement counter points to that line.

Use the following sequence to invalidate one line of cache:

1. Disable the cache controller writing 0 to the CEN field of the CTRL register.
2. Check CSTS field of the SR to verify that the cache is successfully disabled.
3. Perform an invalidate by line writing the field index in the MAINT1 register.
4. Enable the cache controller writing 1 to the CEN field of the CTRL register.

14.5.9.4 Cache Invalidate All Operation

Use the following sequence to invalidate all cache entries:

1. Write 1 to the INVALL field of the MAINT0 register.

14.5.9.5 Cache Performance Monitoring

This module includes a programmable monitor 32-bit counter. The monitor can be configured to count the number of clock cycles, the number of data hit or the number of instruction hit.

Use the following sequence to activate the counter

1. Configure the monitor counter writing the MODE field of the CFG register.
2. Enable the counter writing one to the MENABLE field of the MEN register.
3. If required reset the counter, writing one to the SWRST field of the MCTRL register.
4. Check the value of the monitor counter, reading EVENT_CNT field of the SR

14.5.10 Accessing the PicoCache Memory Block As a Regular Memory On the System Bus

This is only possible when the PicoCache is disabled. In case of an access in this memory while the PicoCache is active, the memory slave will return an error response which will result in a

Cortex-M4 Memory Fault exception. When operating a chip erase, the PicoCache will be cleared whether it was enabled or not.

14.6 Flash Commands

The FLASHCALW offers a command set to manage programming of the flash memory, locking and unlocking of regions, and full flash erasing. See Section 14.10.2 for a complete list of commands.

To run a command, the CMD field in the Flash Command Register (FCMD) has to be written with the command number. As soon as the FCMD register is written, the FRDY bit in the Flash Status Register (FSR) is automatically cleared. Once the current command is complete, the FSR.FRDY bit is automatically set. If an interrupt has been enabled by writing a one to FCR.FRDY, the interrupt request line of the Flash Controller is activated. All flash commands except for Quick Page Read (QPR) and Quick User Page Read (QPRUP) will generate an interrupt request upon completion if FCR.FRDY is one.

Any HSB bus transfers attempting to read flash memory when the FLASHCALW is busy executing a flash command will be stalled, and allowed to continue when the flash command is complete.

After a command has been written to FCMD, the programming algorithm should wait until the command has been executed before attempting to read instructions or data from the flash or writing to the page buffer, as the flash will be busy. The waiting can be performed either by polling the Flash Status Register (FSR) or by waiting for the flash ready interrupt. The command written to FCMD is initiated on the first clock cycle where the HSB bus interface in FLASHCALW is IDLE. The user must make sure that the access pattern to the FLASHCALW HSB interface contains an IDLE cycle so that the command is allowed to start. Make sure that no bus masters such as DMA controllers are performing endless burst transfers from the flash. Also, make sure that the CPU does not perform endless burst transfers from flash. This is done by letting the CPU enter Power Save mode after writing to FCMD, or by polling FSR for command completion. This polling will result in an access pattern with IDLE HSB cycles.

All the commands are protected by the same keyword, which has to be written in the eight highest bits of the FCMD register. Writing FCMD with data that does not contain the correct key and/or with an invalid command has no effect on the flash memory; however, the PROGE bit is set in the Flash Status Register (FSR). This bit is automatically cleared by a read access to the FSR register.

Writing a command to FCMD while another command is being executed has no effect on the flash memory; however, the PROGE bit is set in the Flash Status Register (FSR). This bit is automatically cleared by a read access to the FSR register.

If the current command writes or erases a page in a locked region the command has no effect on the flash memory; however, the LOCKE bit is set in the FSR register. This bit is automatically cleared by a read access to the FSR register.

14.6.1 Write/Erase Page Operation

Flash technology requires that an erase must be done before programming. The entire flash can be erased by an Erase All command. Alternatively, pages can be individually erased by the Erase Page command.

The User page can be written and erased using the mechanisms described in this chapter.

After programming, the page can be locked to prevent miscellaneous write or erase sequences. Locking is performed on a per-region basis, so locking a region locks all pages inside the region.

Data to be written is stored in an internal buffer called the page buffer. The page buffer contains *w* words. The page buffer wraps around within the internal memory area address space and appears to be repeated by the number of pages in it. Writing of 8-bit and 16-bit data to the page buffer is not allowed and may lead to unpredictable data corruption.

Data must be written to the page buffer before the programming command is written to the Flash Command Register (FCMD). The sequence is as follows:

- Erase the page to be programmed.
- Reset the page buffer with the Clear Page Buffer command.
- Fill the page buffer with the desired contents as described in Section 14.5.8 on page 271.
- Programming starts as soon as the programming key and the programming command are written to the Flash Command Register. The PAGEN field in the Flash Command Register (FCMD) must contain the address of the page to write. PAGEN is automatically updated when writing to the page buffer, but can also be written to directly. The FRDY bit in the Flash Status Register (FSR) is automatically cleared when the page write operation starts.
- When programming is completed, the FRDY bit in the Flash Status Register (FSR) is set. If an interrupt was enabled by writing FCR.FRDY to one, an interrupt request is generated.

Two errors can be detected in the FSR register after a programming sequence:

- Programming Error: A bad keyword and/or an invalid command have been written in the FCMD register.
- Lock Error: Can have two different causes:
  - The page to be programmed belongs to a locked region. A command must be executed to unlock the corresponding region before programming can start.
  - A bus master without secure status attempted to program a page requiring secure privileges.

14.6.2 Erase All Operation

The entire memory is erased if the Erase All command (EA) is written to the Flash Command Register (FCMD). Erase All erases all bits in the flash array. The User page is not erased. All flash memory locations, the general-purpose fuse bits, and the security fuses are erased (reset to 0xFF) after an Erase All.

The EA command also ensures that all volatile memories, such as register file and RAMs, are erased before the security fuses are erased.

Erase All operation is allowed only if no regions are locked. Thus, if at least one region is locked, the bit LOCKE in FSR is set and the command is cancelled. If the LOCKE bit in FCR is one, an interrupt request is set generated.

When the command is complete, the FRDY bit in the Flash Status Register (FSR) is set. If an interrupt has been enabled by writing FCR.FRDY to one, an interrupt request is generated. Two errors can be detected in the FSR register after issuing the command:

- Programming Error: A bad keyword and/or an invalid command have been written in the FCMD register.
- Lock Error: At least one lock region is protected. The erase command has been aborted and no page has been erased. A “Unlock region containing given page” (UP) command must be executed to unlock any locked regions.

14.6.3 Region Lock Bits

The flash memory has *p* pages, and these pages are grouped into 16 lock regions, each region containing *p*/16 pages. Each region has a dedicated lock bit preventing writing and erasing pages in the region. After production, the device may have some regions locked. These locked regions are reserved for a boot or default application. Locked regions can be unlocked to be erased and then programmed with another application or other data.

To lock or unlock a region, the commands Lock Region Containing Page (LP) and Unlock Region Containing Page (UP) are provided. Writing one of these commands, together with the number of the page whose region should be locked/unlocked, performs the desired operation.

One error can be detected in the FSR register after issuing the command:

- Programming Error: A bad keyword and/or an invalid command have been written in the FCMD register.

The lock bits are implemented using the lowest 16 general-purpose fuse bits. This means that lock bits can also be set/cleared using the commands for writing/erasing general-purpose fuse bits, see Section 14.7. The general-purpose bit being in an erased (1) state means that the region is unlocked.

14.7 General-purpose Fuse Bits

The flash memory has a number of general-purpose fuse bits that the application programmer can use freely. The fuse bits can be written and erased using dedicated commands, and read through a dedicated Peripheral Bus address. Some of the general-purpose fuse bits are reserved for special purposes, and should not be used for other functions:

Table 14-2. General-purpose Fuses with Special Functions

General-Purpose fuse number	Name	Usage
15:0	LOCK	Region lock bits.

To erase or write a general-purpose fuse bit, the commands Write General-Purpose Fuse Bit (WGPB) and Erase General-Purpose Fuse Bit (EGPB) are provided. Writing one of these commands, together with the number of the fuse to write/erase, performs the desired operation.

An entire General-Purpose Fuse byte can be written at a time by using the Program GP Fuse Byte (PGPFB) instruction. A PGPFB to GP fuse byte 2 is not allowed if the flash is locked by the security fuses. The PFB command is issued with a parameter in the PAGEN field:

- PAGEN[2:0] - byte to write

- PAGEN[10:3] - Fuse value to write

All general-purpose fuses can be erased by the Erase All General-Purpose fuses (EAGP) command. An EAGP command is not allowed if the flash is locked by the security fuses.

Two errors can be detected in the FSR register after issuing these commands:

- Programming Error: A bad keyword and/or an invalid command have been written in the FCMD register.
- Lock Error:

The lock bits are implemented using the lowest 16 general-purpose fuse bits. This means that the 16 lowest general-purpose fuse bits can also be written/erased using the commands for locking/unlocking regions, see Section 14.6.3.

14.8 Security Fuses

The security fuses allow the entire device to be locked from external JTAG or other debug access for code security. The security fuses can be written by a dedicated command, Set Security Fuses (SSB). Once set, the only way to clear the security fuses is through the JTAG Chip Erase command.

Once the security fuses are set, the following Flash Controller commands will be unavailable and return a lock error if attempted:

- Program General-Purpose Fuse Byte (PGPFB) of fuse byte 2
- Erase All General-Purpose Fuses (EAGPF)

One error can be detected in the FSR register after issuing the command:

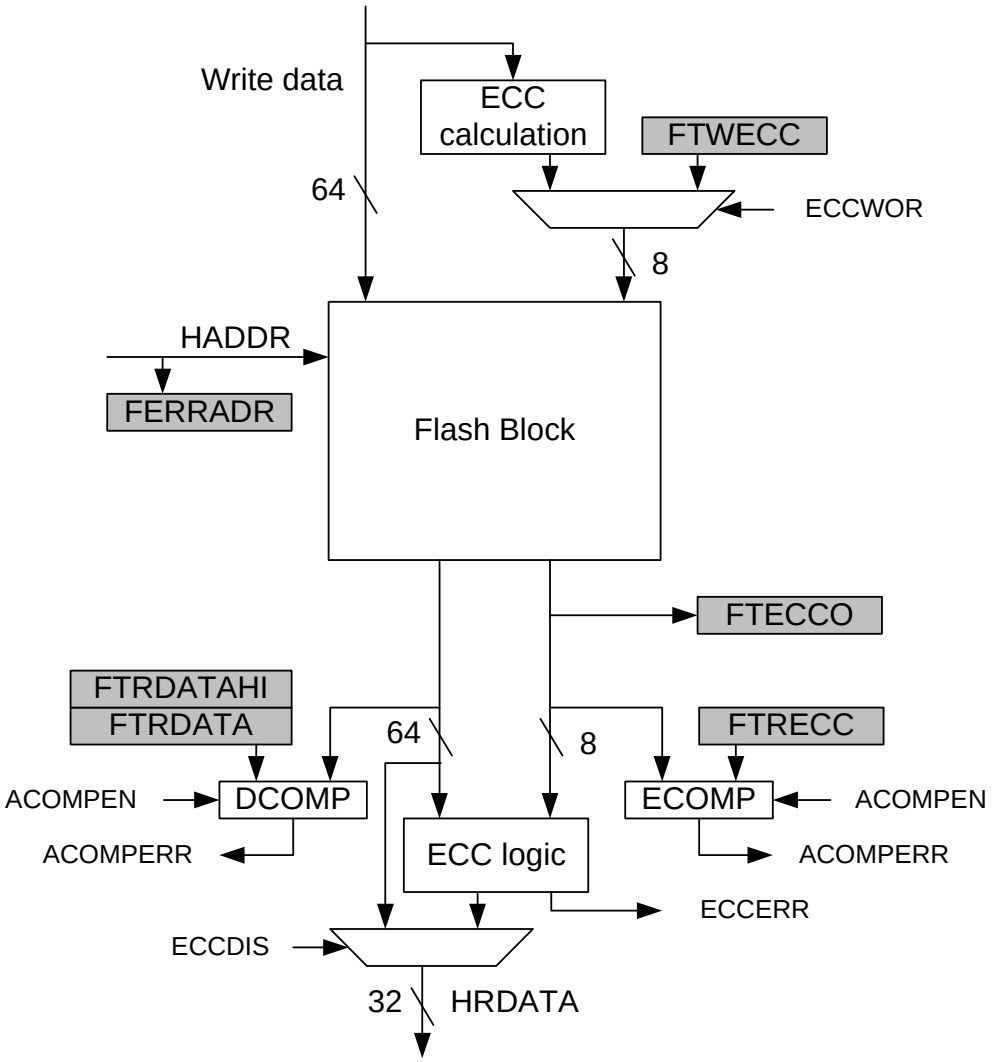
- Programming Error: A bad keyword and/or an invalid command have been written in the FCMD register.

14.9 Error Correcting Code

Error Correcting Code (ECC) logic is implemented to detect and correct errors that may arise in the flash array. The ECC logic is able to detect two bit errors and correct one bit error per 32-bit word in the array. An interrupt is requested upon detection of an ECC error if the ECC Error Interrupt Enable (ECCE) bit in the Flash Control Register (FCR) is set is set. The ECCERR field in Flash Status Register (FSR) indicates the ECC status of the words read from the flash.

Upon detection of an ECC error, the FERRADR register is updated with the failing address. If the ECCE bit is set, FERRADR is loaded only on the first occurrence of an ECC error. Otherwise, it is loaded on all occurrences. ECC checking is performed on a 64-bit basis, so an ECC failure may be present in any of the two words that are output from the flash, not necessarily the word that is addressed on the bus.

Figure 14-5. ECC system



14.10 User Interface

Table 14-3. FLASHCALW Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Flash Control Register	FCR	Read/Write	0x00000000
0x04	Flash Command Register	FCMD	Read/Write	0x00000000
0x08	Flash Status Register	FSR	Read-only	-(1)
0x0C	Flash Parameter Register	FPR	Read-only	-(3)
0x10	Flash Version Register	FVR	Read-only	-(3)
0x14	Flash General Purpose Fuse Register Hi	FGPFRHI	Read-only	-(2)
0x18	Flash General Purpose Fuse Register Lo	FGPFRLO	Read-only	-(2)
0x408	PicoCache Control Register	CTRL	Write-only	0x00000000
0x40C	PicoCache Status Register	SR	Read/Write	0x00000000
0x420	PicoCache Maintenance Register 0	MAINT0	Write-only	-(3)
0x424	PicoCache Maintenance Register 1	MAINT1	Write-only	-(3)
0x428	PicoCache Monitor Configuration Register	MCFG	Read/Write	0x00000000
0x42C	PicoCache Monitor Enable Register	MEN	Read/Write	0x00000000
0x430	PicoCache Monitor Control Register	MCTRL	Write-only	-(3)
0x434	PicoCache Monitor Status Register	MSR	Read-only	0x00000000
0x4FC	Version Register	PVR	Read-only	-(3)

Note:
1. The value of the Lock bits depend on their programmed state. All other bits in FSR are 0.
2. All bits in FGPRHI/LO are dependent on the programmed state of the fuses they map to. Any bits in these registers not mapped to a fuse read as 0.
3. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

14.10.1 Flash Control Register

Name: FCR

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	reserved

7	6	5	4	3	2	1	0
WS1OPT	FWS	-	-ECCE	PROGE	LOCKE	-	FRDY

- **reserved:**
  Must be 0.
-

- **WS1OPT: Wait State 1 Optimization**
  0: No action
  1: Optimize read accesses when configured with one wait state
- **FWS: Flash Wait State**
  0: The flash is read with 0 wait states.
  1: The flash is read with 1 wait state.

- **ECCE: ECC Error Interrupt Enable**
  0: ECC Error does not generate an interrupt.
  1: ECC Error generates an interrupt.
- **PROGE: Programming Error Interrupt Enable**
  0: Programming Error does not generate an interrupt request.
  1: Programming Error generates an interrupt request.
- **LOCKE: Lock Error Interrupt Enable**
  0: Lock Error does not generate an interrupt request.
  1: Lock Error generates an interrupt request.
- **FRDY: Flash Ready Interrupt Enable**
  0: Flash Ready does not generate an interrupt request.
  1: Flash Ready generates an interrupt request.

14.10.2 Flash Command Register

Name: FCMD

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

The FCMD can not be written if the flash is in the process of performing a flash command. Doing so will cause the FCR write to be ignored, and the PROGE bit in FSR to be set.

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
PAGEN [15:8]							

15	14	13	12	11	10	9	8
PAGEN [7:0]							

7	6	5	4	3	2	1	0
-	-	CMD					

- **KEY: Write protection key**
  This field should be written with the value 0xA5 to enable the command defined by the bits of the register. If the field is written with a different value, the write is not performed and no action is started.
  This field always reads as 0.
- **PAGEN: Page number**
  The PAGEN field is used to address a page or fuse bit for certain operations. In order to simplify programming, the PAGEN field is automatically updated every time the page buffer is written to. For every page buffer write, the PAGEN field is updated with the page number of the address being written to. Hardware automatically masks writes to the PAGEN field so that only bits representing valid page numbers can be written, all other bits in PAGEN are always 0. As an example, in a flash with 1024 pages (page 0 - page 1023), bits 15:10 will always be 0.

Table 14-4. Semantic of PAGEN field in different commands

Command	PAGEN description
No operation	Not used
Write Page	The number of the page to write
Clear Page Buffer	Not used
Lock region containing given Page	Page number whose region should be locked
Unlock region containing given Page	Page number whose region should be unlocked
Erase All	Not used
Write General-Purpose Fuse Bit	GPFUSE #
Erase General-Purpose Fuse Bit	GPFUSE #
Set Security Fuses	Not used

Table 14-4. Semantic of PAGEN field in different commands

Command	PAGEN description
Program GP Fuse Byte	WriteData[7:0], ByteAddress[2:0]
Erase All GP Fuses	Not used
Quick Page Read	Page number
Write User Page	Not used
Erase User Page	Not used
Quick Page Read User Page	Not used
High Speed Mode Enable	Not used
High Speed Mode Disable	Not used

- **CMD: Command**
  This field defines the flash command. Issuing any unused command will cause the Programming Error bit in FSR to be set, and the corresponding interrupt to be requested if the PROGE bit in FCR is one.

Table 14-5. Set of commands

Command	Value	Mnemonic
No operation	0	NOP
Write Page	1	WP
Erase Page	2	EP
Clear Page Buffer	3	CPB
Lock region containing given Page	4	LP
Unlock region containing given Page	5	UP
Erase All	6	EA
Write General-Purpose Fuse Bit	7	WGPB
Erase General-Purpose Fuse Bit	8	EGPB
Set Security Fuses	9	SSB
Program GP Fuse Byte	10	PGPFB
Erase All GPFuses	11	EAGPF
Quick Page Read	12	QPR
Write User Page	13	WUP
Erase User Page	14	EUP
Quick Page Read User Page	15	QPRUP
High Speed Mode Enable	16	HSEN
High Speed Mode Disable	17	HSDIS
RESERVED	20-31	

14.10.3 Flash Status Register

Name: FSR

Access Type: Read-only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
LOCK15	LOCK14	LOCK13	LOCK12	LOCK11	LOCK10	LOCK9	LOCK8
23	22	21	20	19	18	17	16
LOCK7	LOCK6	LOCK5	LOCK4	LOCK3	LOCK2	LOCK1	LOCK0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-ECCERR	-ECCERR
7	6	5	4	3	2	1	0
-	HSMODE	QPRR	SECURITY	PROGE	LOCKE	-	FRDY

- **LOCKx: Lock Region x Lock Status**
  0: The corresponding lock region is not locked.
  1: The corresponding lock region is locked.
- **ECCERR: ECC Error Status**
  Automatically cleared when FSR is read.
  Indicates the status of the ECC system. ECCERR can only be only updated when ECCERR == 0, ie for the first ECC error encountered.

- **HSMODE: High-Speed Mode**
  0: High-speed mode disabled.
  1: High-speed mode enabled.
- **QPRR: Quick Page Read Result**
  0: The result is zero, i.e. the page is not erased.
  1: The result is one, i.e. the page is erased.
- **SECURITY: Security Fuses Status**
  0: The security fuses value indicates a non protected state..
  1: The security fuses value indicates a protected state.
- **PROGE: Programming Error Status**
  Automatically cleared when FSR is read.
  0: No invalid commands and no bad keywords were written in the Flash Command Register FCMD.
  1: An invalid command and/or a bad keyword was/were written in the Flash Command Register FCMD.
- **LOCKE: Lock Error Status**
  Automatically cleared when FSR is read.
  0: No programming of at least one locked lock region has happened since the last read of FSR.
  1: Programming of at least one locked lock region has happened since the last read of FSR.
- **FRDY: Flash Ready Status**
  0: The Flash Controller is busy and the application must wait before running a new command.
  1: The Flash Controller is ready to run a new command.

14.10.4 Flash Parameter Register

Name: FPR

Access Type: Read-only

Offset: 0x0C

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	PSZ		

7	6	5	4	3	2	1	0
-	-	-	-	FSZ			

- **PSZ: Page Size**
  The size of each flash page.

Table 14-6. Flash Page Size

PSZ	Page Size
0	32 Byte
1	64 Byte
2	128 Byte
3	256 Byte
4	512 Byte
5	1024 Byte
6	2048 Byte
7	4096 Byte

- **FSZ: Flash Size**
  The size of the flash. Not all device families will provide all flash sizes indicated in the table.

Table 14-7. Flash Size

FSZ	Flash Size	FSZ	Flash Size
0	4 Kbyte	8	192 Kbyte
1	8 Kbyte	9	256 Kbyte
2	16 Kbyte	10	384 Kbyte
3	32 Kbyte	11	512 Kbyte
4	48 Kbyte	12	768 Kbyte
5	64 Kbyte	13	1024 Kbyte
6	96 Kbyte	14	2048 Kbyte
7	128 Kbyte	15	Reserved

14.10.5 Flash Version Register

Name: FVR

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

14.10.6 Flash General Purpose Fuse Register High

Name: FGPFRHI

Access Type: Read-only

Offset: 0x14

Reset Value: -

31	30	29	28	27	26	25	24
GPF63	GPF62	GPF61	GPF60	GPF59	GPF58	GPF57	GPF56
23	22	21	20	19	18	17	16
GPF55	GPF54	GPF53	GPF52	GPF51	GPF50	GPF49	GPF48
15	14	13	12	11	10	9	8
GPF47	GPF46	GPF45	GPF44	GPF43	GPF42	GPF41	GPF40
7	6	5	4	3	2	1	0
GPF39	GPF38	GPF37	GPF36	GPF35	GPF34	GPF33	GPF32

This register is only used in systems with more than 32 GP fuses.

- **GPFxx: General Purpose Fuse xx**
  0: The fuse has a written/programmed state.
  1: The fuse has an erased state.

14.10.7 Flash General Purpose Fuse Register Low

Name: FGPFRLO

Access Type: Read-only

Offset: 0x18

Reset Value: -

31	30	29	28	27	26	25	24
GPF31	GPF30	GPF29	GPF28	GPF27	GPF26	GPF25	GPF24
23	22	21	20	19	18	17	16
GPF23	GPF22	GPF21	GPF20	GPF19	GPF18	GPF17	GPF16
15	14	13	12	11	10	9	8
GPF15	GPF14	GPF13	GPF12	GPF11	GPF10	GPF09	GPF08
7	6	5	4	3	2	1	0
GPF07	GPF06	GPF05	GPF04	GPF03	GPF02	GPF01	GPF00

- **GPFxx: General Purpose Fuse xx**
  0: The fuse has a written/programmed state.
  1: The fuse has an erased state.

14.10.8 PicoCache Control Register

Name: CTRL

Access Type: Write-only

Offset: 0x408

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	CEN

- **CEN: Cache Enable**
  0: Cache is disabled
  1: Cache is enabled

14.10.9 PicoCache Status Register

Name: SR

Access Type: Read-only

Offset: 0x40C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	CSTS

- **CSTS: Cache Controller Status**
  0: When read as 0, this field indicates that the cache controller is disabled.
  1: When read as 1, this field indicates that the cache controller is enabled.

14.10.10 PicoCache Maintenance Register 0

Name: MAINT0

Access Type: Write-only

Offset: 0x420

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	INVALL

- **INVALL: Cache Controller Invalidate All**
  0: no effect.
  1: When set to one, this field invalidate all cache entries.

14.10.11 PicoCache Maintenance Register 1

Name: MAINT1

Access Type: Write-only

Offset: 0x424

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
reserved		-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
INDEX				-	-	-	-

- **reserved:**
  must be null.
- **INDEX: Invalidate Index**

This field indicates the cache line that is to be invalidated

14.10.12 PicoCache Monitor Configuration Register

Name: MCFG

Access Type: Read/Write

Offset: 0x428

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	MODE	

- **MODE: Cache Controller Monitor Counter Mode**

Value	Name	Description
0	CYCLE_COUNT	cycle counter
1	IHIT_COUNT	instruction hit counter
2	DHIT_COUNT	data hit counter

14.10.13 PicoCache Monitor Enable Register

Name: MEN

Access Type: Write-only

Offset: 0x42C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	MENABLE

- **MENABLE: Monitor Enable**
  0: The monitor counter is disabled
  1: The monitor counter is enabled

14.10.14 PicoCache Monitor Control Register

Name: MCTRL

Access Type: Write-only

Offset: 0x430

Reset Value: 0x00000000

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

- **SWRST: Monitor Software Reset**
  0: No effect
  1: Writing a one in this field resets the event counter register

14.10.15 PicoCache Monitor Status Register

Name: MSR

Access Type: Read-only

Offset: 0x434

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
EVENT_CNT							

23	22	21	20	19	18	17	16
EVENT_CNT							

15	14	13	12	11	10	9	8
EVENT_CNT							

7	6	5	4	3	2	1	0
EVENT_CNT							

- **EVENT_CNT: Monitor Event Counter**

14.10.16 PicoCache Version Register

Name: PVR

Access Type: Read-only

Offset: 0x4FC

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	MFN		

15	14	13	12	11	10	9	8
-	-	-	-	VERSION			

7	6	5	4	3	2	1	0
VERSION							

- **VERSION**
- **MFN**

14.11 Fuse Settings

The flash contains 32 general purpose fuses. These 32 fuses can be found in the Flash General Purpose Fuse Register Low (FGPFRLO). The Flash General Purpose Fuse Register High (FGPFRHI) is not used. In addition to the general purpose fuses, parts of the flash user page can have a defined meaning outside of the flash controller and will also be described in this section.

Note that when writing to the user page the values do not get loaded by the other modules on the device until a device reset occurs.

The general purpose fuses are erased by a JTAG chip erase.

14.11.1 Flash General Purpose Fuse Register Low (FGPFRLO)

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
LOCK[15:8]							

7	6	5	4	3	2	1	0
LOCK[7:0]							

Default Fuse Value: The devices are shipped with the FGPFRLO register value:0xFFFFFFFF:
Reserved fuses set to 1.

LOCK fuses set to 1111111111111111. No region locked.
After the JTAG chip erase command, the FGPFR register value is 0xFFFFFFFF.

14.11.2 First Word of the User Page (Address 0x808000004)

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	BOD18HYST	BOD18ACTION		BOD18EN	BOD18LEVEL[5]

15	14	13	12	11	10	9	8
BOD18LEVEL[4:0]					BOD33HYST	BOD33ACTION	

7	6	5	4	3	2	1	0
BOD33EN	BOD33LEVEL						WDTAUTO

- **BOD18HYST: 1v8 Brown Out Detector Hysteresis**
  0: The Brown out detector hysteresis is disabled
  1: The Brown out detector hysteresis is enabled
- **BOD18ACTION: 1v8 Brown Out Detector Action**

Action[1:0]	Description
00	No Action.
01	The BOD generates a reset.
10	The BOD generates an interrupt.
11	No Action.

- **BOD18EN: 1v8 Brown Out Detector Enable**
  0: The Brown out detector is disabled.
  1: The Brown out detector is enabled.
- **BOD18LEVEL: 1v8 Brown Out Detector Level**
  This controls the voltage trigger level for the 1v8 Brown out detector. Refer to Section 42. “Electrical Characteristics” on page 1121
- **BOD33HYST: 3v3 Brown Out Detector Hysteresis**
  0: The Brown out detector hysteresis is disabled
  1: The Brown out detector hysteresis is enabled
- **BOD33ACTION: 3v3 Brown Out Detector Action**

Action[1:0]	Description
00	No Action.
01	The BOD generates a reset.
10	The BOD generates an interrupt.
11	No Action.

- **BOD33EN: 3v3 Brown Out Detector Enable**
  0: The Brown out detector is disabled.
  1: The Brown out detector is enabled.

- **BOD33LEVEL: 3v3 Brown Out Detector Level**
  This controls the voltage trigger level for the 3v3 Brown out detector. Refer to Section 42. “Electrical Characteristics” on page 1121.
- **WDTAUTO: WatchDog Timer Auto Enable at Startup**
  0: The WDT is automatically enabled at startup.
  1: The WDT is not automatically enabled at startup.
  Refer to the WDT chapter for detail about timeout settings when the WDT is automatically enabled.

  Default user page first word value:The devices are shipped with the user page erased (all bits 1):
  WDTAUTO set to 1, WDT disabled.

14.11.3 Second Word of the User Page (Address 0x80800000)

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

Default user page second word value: The devices are shipped with the User page erased (all bits 1).

14.12 Serial Number

Each device has a unique 120 bits serial number readable from address 0x0080020C to 0x0080021A.

14.13 Module Configuration

The specific configuration for each FLASHCALW instance is listed in the following tables.

Table 14-8. Module Configuration

Feature	ATSAM4LC4CA, ATSAM4LC4BA, ATSAM4LC4AA, ATSAM4LS4CA, ATSAM4LS4BA, ATSAM4LS4AA	ATSAM4LC2CA, ATSAM4LC2BA, ATSAM4LC2AA, ATSAM4LS2CA, ATSAM4LS2BA, ATSAM4LS2AA
Flash size	256Kbytes	128Kbytes
Number of pages	512	256

Feature	ATSAM4LC8CA, ATSAM4LC8BA, ATSAM4LC8AA, ATSAM4LS8CA, ATSAM4LS8BA, ATSAM4LS8AA
Flash size	512Kbytes
Number of pages	1024

The page size is 512 bytes for all configurations.

The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details

Table 14-9. Module Clock Name

Module Name	Clock Name	Description
FLASHCALW	CLK_FLASHCALW_AHB	Clock for the FLASHCALW AHB interface
	CLK_HRAMC1_AHB	Clock for the HRAMC1 interface
	CLK_FLASHCALW_APB	Clock for the FLASHCALW PB interface

Table 14-10. Register Reset Values

Register	ATSAM4LC4CA, ATSAM4LC4BA, ATSAM4LC4AA, ATSAM4LS4CA, ATSAM4LS4BA, ATSAM4LS4AA	ATSAM4LC2CA, ATSAM4LC2BA, ATSAM4LC2AA, ATSAM4LS2CA, ATSAM4LS2BA, ATSAM4LS2AA
FVR	0x00000110	0x00000110
FPR	0x00000409	0x00000407

Register	ATSAM4LC8CA, ATSAM4LC8BA, ATSAM4LC8AA, ATSAM4LS8CA, ATSAM4LS8BA, ATSAM4LS8AA
FVR	0x00000110
FPR	0x0000040B

15. HSB Bus Matrix (HMATRIXB)

Rev: 1.3.0.3

15.1 Features

- **User Interface on peripheral bus**
- **Configurable number of masters (up to 16)**
- **Configurable number of slaves (up to 16)**
- **One decoder for each master**
- **Programmable arbitration for each slave**
  - **Round-Robin**
  - **Fixed priority**
- **Programmable default master for each slave**
  - **No default master**
  - **Last accessed default master**
  - **Fixed default master**
- **One cycle latency for the first access of a burst**
- **Zero cycle latency for default master**
- **One special function register for each slave (not dedicated)**

15.2 Overview

The Bus Matrix implements a multi-layer bus structure, that enables parallel access paths between multiple High Speed Bus (HSB) masters and slaves in a system, thus increasing the overall bandwidth. The Bus Matrix interconnects up to 16 HSB Masters to up to 16 HSB Slaves. The normal latency to connect a master to a slave is one cycle except for the default master of the accessed slave which is connected directly (zero cycle latency). The Bus Matrix provides 16 Special Function Registers (SFR) that allow the Bus Matrix to support application specific features.

15.3 Product Dependencies

In order to configure this module by accessing the user registers, other parts of the system must be configured correctly, as described below.

15.3.1 Clocks

The clock for the HMATRIX bus interface (CLK_HMATRIX) is generated by the Power Manager. This clock is disabled at reset, and can be enaabled in the Power Manager.

15.4 Functional Description

15.4.1 Special Bus Granting Mechanism

The Bus Matrix provides some speculative bus granting techniques in order to anticipate access requests from some masters. This mechanism reduces latency at first access of a burst or single transfer. This bus granting mechanism sets a different default master for every slave.

At the end of the current access, if no other request is pending, the slave remains connected to its associated default master. A slave can be associated with three kinds of default masters: no default master, last access master, and fixed default master.

To change from one kind of default master to another, the Bus Matrix user interface provides the Slave Configuration Registers, one for each slave, that set a default master for each slave. The Slave Configuration Register contains two fields: DEFMSTR_TYPE and FIXED_DEFMSTR. The 2-bit DEFMSTR_TYPE field selects the default master type (no default, last access master, fixed default master), whereas the 4-bit FIXED_DEFMSTR field selects a fixed default master provided that DEFMSTR_TYPE is set to fixed default master. Refer to the Bus Matrix user interface description.

15.4.1.1 No Default Master

At the end of the current access, if no other request is pending, the slave is disconnected from all masters. No Default Master suits low-power mode.

15.4.1.2 Last Access Master

At the end of the current access, if no other request is pending, the slave remains connected to the last master that performed an access request.

15.4.1.3 Fixed Default Master

At the end of the current access, if no other request is pending, the slave connects to its fixed default master. Unlike last access master, the fixed master does not change unless the user modifies it by a software action (field FIXED_DEFMSTR of the related SCFG).

15.4.2 Arbitration

The Bus Matrix provides an arbitration mechanism that reduces latency when conflict cases occur, i.e. when two or more masters try to access the same slave at the same time. One arbiter per HSB slave is provided, thus arbitrating each slave differently.

The Bus Matrix provides the user with the possibility of choosing between 2 arbitration types for each slave:

1. Round-Robin Arbitration (default)
2. Fixed Priority Arbitration

This is selected by the ARBT field in the Slave Configuration Registers (SCFG).

Each algorithm may be complemented by selecting a default master configuration for each slave.

When a re-arbitration must be done, specific conditions apply. This is described in “Arbitration Rules” .

15.4.2.1 Arbitration Rules

Each arbiter has the ability to arbitrate between two or more different master requests. In order to avoid burst breaking and also to provide the maximum throughput for slave interfaces, arbitration may only take place during the following cycles:

1. Idle Cycles: When a slave is not connected to any master or is connected to a master which is not currently accessing it.
2. Single Cycles: When a slave is currently doing a single access.
3. End of Burst Cycles: When the current cycle is the last cycle of a burst transfer. For defined length burst, predicted end of burst matches the size of the transfer but is managed differently for undefined length burst. This is described below.
4. Slot Cycle Limit: When the slot cycle counter has reached the limit value indicating that the current master access is too long and must be broken. This is described below.

- Undefined Length Burst Arbitration

In order to avoid long slave handling during undefined length bursts (INCR), the Bus Matrix provides specific logic in order to re-arbitrate before the end of the INCR transfer. A predicted end of burst is used as a defined length burst transfer and can be selected among the following five possibilities:

1. Infinite: No predicted end of burst is generated and therefore INCR burst transfer will never be broken.
2. One beat bursts: Predicted end of burst is generated at each single transfer inside the INCP transfer.
3. Four beat bursts: Predicted end of burst is generated at the end of each four beat boundary inside INCR transfer.
4. Eight beat bursts: Predicted end of burst is generated at the end of each eight beat boundary inside INCR transfer.
5. Sixteen beat bursts: Predicted end of burst is generated at the end of each sixteen beat boundary inside INCR transfer.

This selection can be done through the ULBT field in the Master Configuration Registers (MCFG).

- Slot Cycle Limit Arbitration

The Bus Matrix contains specific logic to break long accesses, such as very long bursts on a very slow slave (e.g., an external low speed memory). At the beginning of the burst access, a counter is loaded with the value previously written in the SLOT_CYCLE field of the related Slave Configuration Register (SCFG) and decreased at each clock cycle. When the counter reaches zero, the arbiter has the ability to re-arbitrate at the end of the current byte, halfword, or word transfer.

15.4.2.2 Round-Robin Arbitration

This algorithm allows the Bus Matrix arbiters to dispatch the requests from different masters to the same slave in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is first serviced, then the others are serviced in a round-robin manner.

There are three round-robin algorithms implemented:

1. Round-Robin arbitration without default master
2. Round-Robin arbitration with last default master
3. Round-Robin arbitration with fixed default master

- Round-Robin Arbitration without Default Master

This is the main algorithm used by Bus Matrix arbiters. It allows the Bus Matrix to dispatch requests from different masters to the same slave in a pure round-robin manner. At the end of the current access, if no other request is pending, the slave is disconnected from all masters. This configuration incurs one latency cycle for the first access of a burst. Arbitration without default master can be used for masters that perform significant bursts.

- Round-Robin Arbitration with Last Default Master

This is a biased round-robin algorithm used by Bus Matrix arbiters. It allows the Bus Matrix to remove the one latency cycle for the last master that accessed the slave. At the end of the cur-

rent transfer, if no other master request is pending, the slave remains connected to the last master that performed the access. Other non privileged masters still get one latency cycle if they want to access the same slave. This technique can be used for masters that mainly perform single accesses.

- Round-Robin Arbitration with Fixed Default Master

This is another biased round-robin algorithm. It allows the Bus Matrix arbiters to remove the one latency cycle for the fixed default master per slave. At the end of the current access, the slave remains connected to its fixed default master. Every request attempted by this fixed default master will not cause any latency whereas other non privileged masters will still get one latency cycle. This technique can be used for masters that mainly perform single accesses.

15.4.2.3 Fixed Priority Arbitration

This algorithm allows the Bus Matrix arbiters to dispatch the requests from different masters to the same slave by using the fixed priority defined by the user. If two or more master requests are active at the same time, the master with the highest priority number is serviced first. If two or more master requests with the same priority are active at the same time, the master with the highest number is serviced first.

For each slave, the priority of each master may be defined through the Priority Registers for Slaves (PRAS and PRBS).

15.4.3 Slave and Master Assignation

The index number assigned to Bus Matrix slaves and masters are described in the Module Configuration section at the end of this chapter.

15.5 User Interface

Table 15-1. HMATRIX Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0000	Master Configuration Register 0	MCFG0	Read/Write	0x00000002
0x0004	Master Configuration Register 1	MCFG1	Read/Write	0x00000002
0x0008	Master Configuration Register 2	MCFG2	Read/Write	0x00000002
0x000C	Master Configuration Register 3	MCFG3	Read/Write	0x00000002
0x0010	Master Configuration Register 4	MCFG4	Read/Write	0x00000002
0x0014	Master Configuration Register 5	MCFG5	Read/Write	0x00000002
0x0018	Master Configuration Register 6	MCFG6	Read/Write	0x00000002
0x001C	Master Configuration Register 7	MCFG7	Read/Write	0x00000002
0x0020	Master Configuration Register 8	MCFG8	Read/Write	0x00000002
0x0024	Master Configuration Register 9	MCFG9	Read/Write	0x00000002
0x0028	Master Configuration Register 10	MCFG10	Read/Write	0x00000002
0x002C	Master Configuration Register 11	MCFG11	Read/Write	0x00000002
0x0030	Master Configuration Register 12	MCFG12	Read/Write	0x00000002
0x0034	Master Configuration Register 13	MCFG13	Read/Write	0x00000002
0x0038	Master Configuration Register 14	MCFG14	Read/Write	0x00000002
0x003C	Master Configuration Register 15	MCFG15	Read/Write	0x00000002
0x0040	Slave Configuration Register 0	SCFG0	Read/Write	0x00000010
0x0044	Slave Configuration Register 1	SCFG1	Read/Write	0x00000010
0x0048	Slave Configuration Register 2	SCFG2	Read/Write	0x00000010
0x004C	Slave Configuration Register 3	SCFG3	Read/Write	0x00000010
0x0050	Slave Configuration Register 4	SCFG4	Read/Write	0x00000010
0x0054	Slave Configuration Register 5	SCFG5	Read/Write	0x00000010
0x0058	Slave Configuration Register 6	SCFG6	Read/Write	0x00000010
0x005C	Slave Configuration Register 7	SCFG7	Read/Write	0x00000010
0x0060	Slave Configuration Register 8	SCFG8	Read/Write	0x00000010
0x0064	Slave Configuration Register 9	SCFG9	Read/Write	0x00000010
0x0068	Slave Configuration Register 10	SCFG10	Read/Write	0x00000010
0x006C	Slave Configuration Register 11	SCFG11	Read/Write	0x00000010
0x0070	Slave Configuration Register 12	SCFG12	Read/Write	0x00000010
0x0074	Slave Configuration Register 13	SCFG13	Read/Write	0x00000010
0x0078	Slave Configuration Register 14	SCFG14	Read/Write	0x00000010
0x007C	Slave Configuration Register 15	SCFG15	Read/Write	0x00000010
0x0080	Priority Register A for Slave 0	PRAS0	Read/Write	0x00000000
0x0084	Priority Register B for Slave 0	PRBS0	Read/Write	0x00000000
0x0088	Priority Register A for Slave 1	PRAS1	Read/Write	0x00000000

Table 15-1. HMATRIX Register Memory Map (Continued)

Offset	Register	Register Name	Access	Reset
0x008C	Priority Register B for Slave 1	PRBS1	Read/Write	0x00000000
0x0090	Priority Register A for Slave 2	PRAS2	Read/Write	0x00000000
0x0094	Priority Register B for Slave 2	PRBS2	Read/Write	0x00000000
0x0098	Priority Register A for Slave 3	PRAS3	Read/Write	0x00000000
0x009C	Priority Register B for Slave 3	PRBS3	Read/Write	0x00000000
0x00A0	Priority Register A for Slave 4	PRAS4	Read/Write	0x00000000
0x00A4	Priority Register B for Slave 4	PRBS4	Read/Write	0x00000000
0x00A8	Priority Register A for Slave 5	PRAS5	Read/Write	0x00000000
0x00AC	Priority Register B for Slave 5	PRBS5	Read/Write	0x00000000
0x00B0	Priority Register A for Slave 6	PRAS6	Read/Write	0x00000000
0x00B4	Priority Register B for Slave 6	PRBS6	Read/Write	0x00000000
0x00B8	Priority Register A for Slave 7	PRAS7	Read/Write	0x00000000
0x00BC	Priority Register B for Slave 7	PRBS7	Read/Write	0x00000000
0x00C0	Priority Register A for Slave 8	PRAS8	Read/Write	0x00000000
0x00C4	Priority Register B for Slave 8	PRBS8	Read/Write	0x00000000
0x00C8	Priority Register A for Slave 9	PRAS9	Read/Write	0x00000000
0x00CC	Priority Register B for Slave 9	PRBS9	Read/Write	0x00000000
0x00D0	Priority Register A for Slave 10	PRAS10	Read/Write	0x00000000
0x00D4	Priority Register B for Slave 10	PRBS10	Read/Write	0x00000000
0x00D8	Priority Register A for Slave 11	PRAS11	Read/Write	0x00000000
0x00DC	Priority Register B for Slave 11	PRBS11	Read/Write	0x00000000
0x00E0	Priority Register A for Slave 12	PRAS12	Read/Write	0x00000000
0x00E4	Priority Register B for Slave 12	PRBS12	Read/Write	0x00000000
0x00E8	Priority Register A for Slave 13	PRAS13	Read/Write	0x00000000
0x00EC	Priority Register B for Slave 13	PRBS13	Read/Write	0x00000000
0x00F0	Priority Register A for Slave 14	PRAS14	Read/Write	0x00000000
0x00F4	Priority Register B for Slave 14	PRBS14	Read/Write	0x00000000
0x00F8	Priority Register A for Slave 15	PRAS15	Read/Write	0x00000000
0x00FC	Priority Register B for Slave 15	PRBS15	Read/Write	0x00000000
0x0110	Special Function Register 0	SFR0	Read/Write	–
0x0114	Special Function Register 1	SFR1	Read/Write	–
0x0118	Special Function Register 2	SFR2	Read/Write	–
0x011C	Special Function Register 3	SFR3	Read/Write	–
0x0120	Special Function Register 4	SFR4	Read/Write	–
0x0124	Special Function Register 5	SFR5	Read/Write	–
0x0128	Special Function Register 6	SFR6	Read/Write	–

Table 15-1. HMATRIX Register Memory Map (Continued)

Offset	Register	Register Name	Access	Reset
0x012C	Special Function Register 7	SFR7	Read/Write	–
0x0130	Special Function Register 8	SFR8	Read/Write	–
0x0134	Special Function Register 9	SFR9	Read/Write	–
0x0138	Special Function Register 10	SFR10	Read/Write	–
0x013C	Special Function Register 11	SFR11	Read/Write	–
0x0140	Special Function Register 12	SFR12	Read/Write	–
0x0144	Special Function Register 13	SFR13	Read/Write	–
0x0148	Special Function Register 14	SFR14	Read/Write	–
0x014C	Special Function Register 15	SFR15	Read/Write	–

15.5.1 Master Configuration Registers

Name: MCFG0...MCFG15

Access Type: Read/Write

Offset: 0x00 - 0x3C

Reset Value: 0x00000002

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

- **ULBT: Undefined Length Burst Type**

Table 15-2. Undefined Length Burst Type

ULBT	Undefined Length Burst Type	Description
000	Inifinite Length Burst	No predicted end of burst is generated and therefore INCR bursts coming from this master cannot be broken.
001	Single-Access	The undefined length burst is treated as a succession of single accesses, allowing re-arbitration at each beat of the INCR burst.
010	4 Beat Burst	The undefined length burst is split into a four-beat burst, allowing re-arbitration at each four-beat burst end.
011	8 Beat Burst	The undefined length burst is split into an eight-beat burst, allowing re-arbitration at each eight-beat burst end.
100	16 Beat Burst	The undefined length burst is split into a sixteen-beat burst, allowing re-arbitration at each sixteen-beat burst end.

15.5.2 Slave Configuration Registers

Name: SCFG0...SCFG15

Access Type: Read/Write

Offset: 0x40 - 0x7C

Reset Value: 0x00000010

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	ARBT

23	22	21	20	19	18	17	16
–	–	FIXED_DEFMSTR				DEFMSTR_TYPE	

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
SLOT_CYCLE							

- **ARBT: Arbitration Type**
  0: Round-Robin Arbitration
  1: Fixed Priority Arbitration
- **FIXED_DEFMSTR: Fixed Default Master**
  This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.
- **DEFMSTR_TYPE: Default Master Type**
  0: No Default Master
  At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters.
  This results in a one cycle latency for the first access of a burst transfer or for a single access.
  1: Last Default Master
  At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it.
  This results in not having one cycle latency when the last master tries to access the slave again.
  2: Fixed Default Master
  At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field.
  This results in not having one cycle latency when the fixed master tries to access the slave again.
- **SLOT_CYCLE: Maximum Number of Allowed Cycles for a Burst**
  When the SLOT_CYCLE limit is reached for a burst, it may be broken by another master trying to access this slave.
  This limit has been placed to avoid locking a very slow slave when very long bursts are used.
  This limit must not be very small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. 16 cycles is a reasonable value for SLOT_CYCLE.

15.5.3 Bus Matrix Priority Registers A For Slaves

Register Name: PRAS0...PRAS15

Access Type: Read/Write

Offset: -

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	M7PR		-	-	M6PR	

23	22	21	20	19	18	17	16
-	-	M5PR		-	-	M4PR	

15	14	13	12	11	10	9	8
-	-	M3PR		-	-	M2PR	

7	6	5	4	3	2	1	0
-	-	M1PR		-	-	M0PR	

- **MxPR: Master x Priority**

  Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.

15.5.4 Priority Registers B For Slaves

Name: PRBS0...PRBS15

Access Type: Read/Write

Offset: -

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	M15PR		-	-	M14PR	

23	22	21	20	19	18	17	16
-	-	M13PR		-	-	M12PR	

15	14	13	12	11	10	9	8
-	-	M11PR		-	-	M10PR	

7	6	5	4	3	2	1	0
-	-	M9PR		-	-	M8PR	

- **MxPR: Master x Priority**

  Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.

15.5.5 Special Function Registers

Name: SFR0...SFR15

Access Type: Read/Write

Offset: 0x110 - 0x14C

Reset Value: -

31	30	29	28	27	26	25	24
SFR							

23	22	21	20	19	18	17	16
SFR							

15	14	13	12	11	10	9	8
SFR							

7	6	5	4	3	2	1	0
SFR							

- **SFR: Special Function Register Fields**

  Those registers are not a HMATRIX specific register. The field of those will be defined where they are used.

15.6 Module Configuration

The specific configuration for each HMATRIX instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 15-3. HMATRIX Clocks

Clock Name	Description
CLK_HMATRIX	Clock for the HMATRIX bus interface

15.6.1 Bus Matrix Connections

The bus matrix has the several masters and slaves. Each master has its own bus and its own decoder, thus allowing a different memory mapping per master. The master number in the table below can be used to index the HMATRIX control registers. For example, HMATRIX MCFG0 register is associated with the CPU IDCODE master interface.

Table 15-4. High Speed Bus Masters

Master 0	CPU IDCODE
Master 1	CPU SYS
Master 2	SMAP
Master 3	PDCA
Master 4	USBC
Master 5	CRCCU

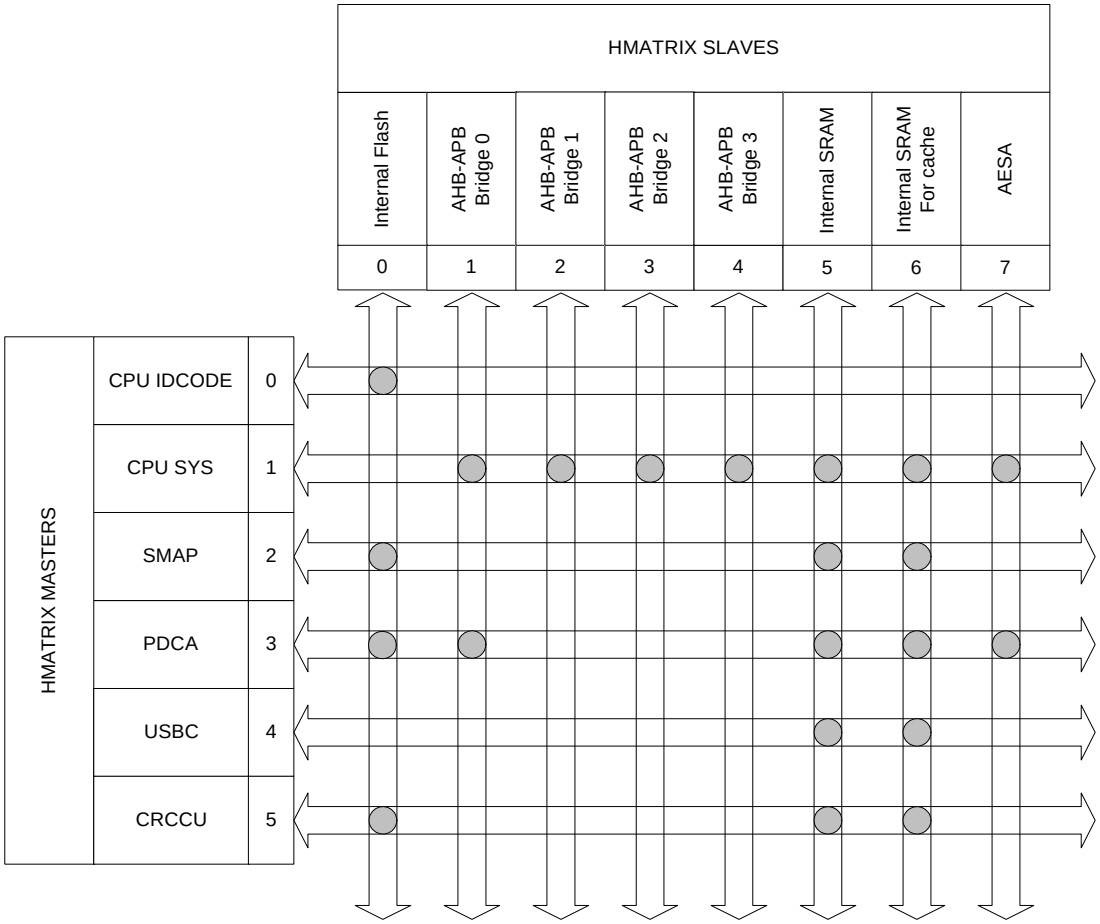
Each slave has its own arbiter, thus allowing a different arbitration per slave. The slave number in the table below can be used to index the HMATRIX control registers. For example, SCFG5 is associated with the Internal SRAM Slave Interface.

Accesses to unused areas returns an error result to the master requesting such an access.

Table 15-5. High Speed Bus Slaves

Slave 0	Internal Flash
Slave 1	AHB-APB Bridge A
Slave 2	AHB-APB Bridge B
Slave 3	AHB-APB Bridge C
Slave 4	AHB-APB Bridge D
Slave 5	Internal SRAM
Slave 6	Internal SRAM for cache
Slave 7	AESA

Figure 15-1. HMatrix Master / Slave Connections



16. Peripheral DMA Controller (PDCA)

Rev: 1.2.4.1

16.1 Features

- **Multiple channels**
- **Generates transfers between memories and peripherals such as USART and SPI**
- **Two address pointers/counters per channel allowing double buffering**
- **Optional synchronizing of data transfers with extenal peripheral events**
- **Ring buffer functionality**

16.2 Overview

The Peripheral DMA Controller (PDCA) transfers data between on-chip peripheral modules such as USART, SPI and memories (those memories may be on- and off-chip memories). Using the PDCA avoids CPU intervention for data transfers, improving the performance of the microcontroller. The PDCA can transfer data from memory to a peripheral or from a peripheral to memory.

The PDCA consists of multiple DMA channels. Each channel has:

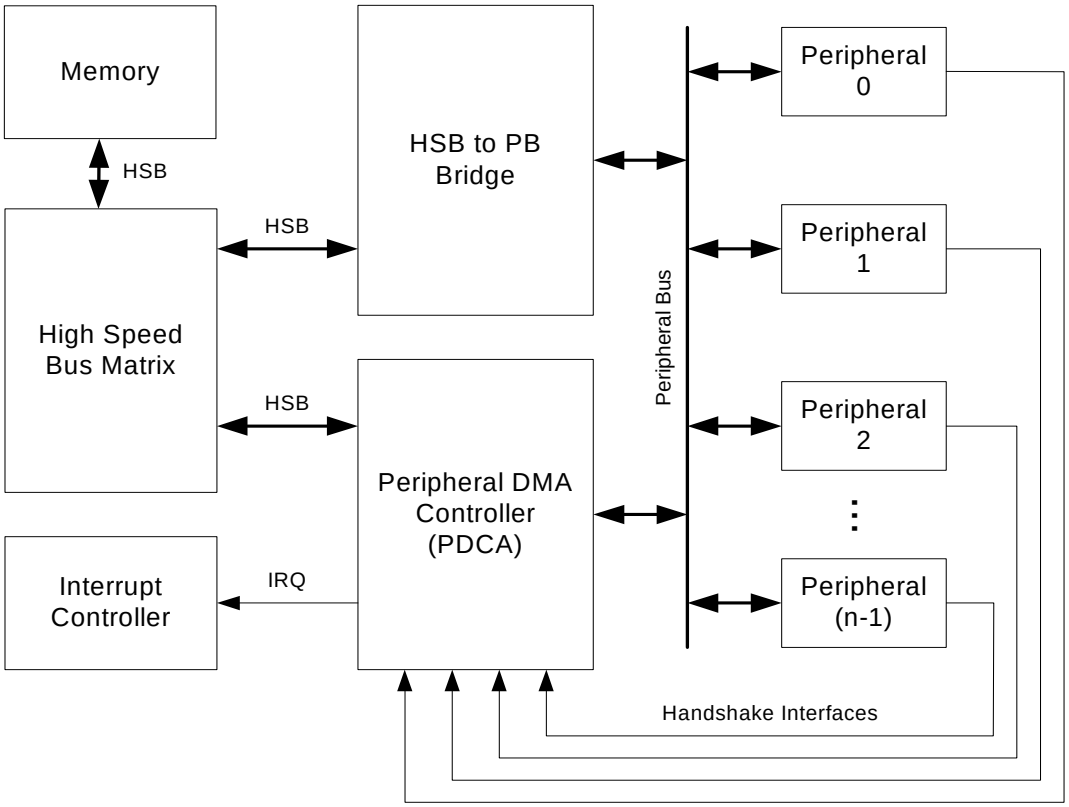
- A Peripheral Select Register
- A 32-bit memory pointer
- A 16-bit transfer counter
- A 32-bit memory pointer reload value
- A 16-bit transfer counter reload value

The PDCA communicates with the peripheral modules over a set of handshake interfaces. The peripheral signals the PDCA when it is ready to receive or transmit data. The PDCA acknowledges the request when the transmission has started.

When a transmit buffer is empty or a receive buffer is full, an optional interrupt request can be generated.

16.3 Block Diagram

Figure 16-1. PDCA Block Diagram



16.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

16.4.1 Power Management

If the CPU enters a sleep mode that disables the PDCA clocks, the PDCA will stop functioning and resume operation after the system wakes up from sleep mode.

16.4.2 Clocks

The PDCA has two bus clocks connected: One High Speed Bus clock (CLK_PDCA_HSB) and one Peripheral Bus clock (CLK_PDCA_PB). These clocks are generated by the Power Manager. The status of both clocks at reset can be known in the Power Manager section. It is recommended to disable the PDCA before disabling the clocks, to avoid freezing the PDCA in an undefined state.

16.4.3 Interrupts

The PDCA interrupt request lines are connected to the NVIC. Using the PDCA interrupts requires the NVIC to be programmed first.

16.4.4 Peripheral Events

The PDCA peripheral events are connected via the Peripheral Event System. Refer to Section 31. "Peripheral Event Controller (PEVC)" on page 845 for details.

16.5 Functional Description

16.5.1 Basic Operation

The PDCA consists of multiple independent PDCA channels, each capable of handling DMA requests in parallel. Each PDCA channels contains a set of configuration registers which must be configured to start a DMA transfer.

In this section the steps necessary to configure one PDCA channel is outlined.

The peripheral to transfer data to or from must be configured correctly in the Peripheral Select Register (PSR). This is performed by writing the Peripheral Identity (PID) value for the corresponding peripheral to the PID field in the PSR register. The PID also encodes the transfer direction, i.e. memory to peripheral or peripheral to memory. See Section 16.5.6.

The transfer size must be written to the Transfer Size field in the Mode Register (MR.SIZE). The size must match the data size produced or consumed by the selected peripheral. See Section 16.5.7.

The memory address to transfer to or from, depending on the PSR, must be written to the Memory Address Register (MAR). For each transfer the memory address is increased by either a one, two or four, depending on the size set in MR. See Section 16.5.2.

The number of data items to transfer is written to the TCR register. If the PDCA channel is enabled, a transfer will start immediately after writing a non-zero value to TCR or the reload version of TCR, TCRR. After each transfer the TCR value is decreased by one. Both MAR and TCR can be read while the PDCA channel is active to monitor the DMA progress. See Section 16.5.3.

The channel must be enabled for a transfer to start. A channel is enable by writing a one to the EN bit in the Control Register (CR).

16.5.2 Memory Pointer

Each channel has a 32-bit Memory Address Register (MAR). This register holds the memory address for the next transfer to be performed. The register is automatically updated after each transfer. The address will be increased by either one, two or four depending on the size of the DMA transfer (byte, halfword or word). The MAR can be read at any time during transfer.

16.5.3 Transfer Counter

Each channel has a 16-bit Transfer Counter Register (TCR). This register must be written with the number of transfers to be performed. The TCR register should contain the number of data items to be transferred independently of the transfer size. The TCR can be read at any time during transfer to see the number of remaining transfers.

16.5.4 Reload Registers

Both the MAR and the TCR have a reload register, respectively Memory Address Reload Register (MARR) and Transfer Counter Reload Register (TCRR). These registers provide the possibility for the PDCA to work on two memory buffers for each channel. When one buffer has completed, MAR and TCR will be reloaded with the values in MARR and TCRR. The reload logic is always enabled and will trigger if the TCR reaches zero while TCRR holds a non-zero value. After reload, the MARR and TCRR registers are cleared.

If TCR is zero when writing to TCRR, the TCR and MAR are automatically updated with the value written in TCRR and MARR.

16.5.5 Ring Buffer

When Ring Buffer mode is enabled the TCRR and MARR registers will not be cleared when TCR and MAR registers reload. This allows the PDCA to read or write to the same memory region over and over again until the transfer is actively stopped by the user. Ring Buffer mode is enabled by writing a one to the Ring Buffer bit in the Mode Register (MR.RING).

16.5.6 Peripheral Selection

The Peripheral Select Register (PSR) decides which peripheral should be connected to the PDCA channel. A peripheral is selected by writing the corresponding Peripheral Identity (PID) to the PID field in the PSR register. Writing the PID will both select the direction of the transfer (memory to peripheral or peripheral to memory), which handshake interface to use, and the address of the peripheral holding register. Refer to the Peripheral Identity (PID) table in the Module Configuration section for the peripheral PID values.

16.5.7 Transfer Size

The transfer size can be set individually for each channel to be either byte, halfword or word (8-bit, 16-bit or 32-bit respectively). Transfer size is set by writing the desired value to the Transfer Size field in the Mode Register (MR.SIZE).

When the PDCA moves data between peripherals and memory, data is automatically sized and aligned. When memory is accessed, the size specified in MR.SIZE and system alignment is used. When a peripheral register is accessed the data to be transferred is converted to a word where bit n in the data corresponds to bit n in the peripheral register. If the transfer size is byte or halfword, bits greater than 8 and16 respectively are set to zero.

Refer to the Module Configuration section for information regarding what peripheral registers are used for the different peripherals and then to the peripheral specific chapter for information about the size option available for the different registers.

16.5.8 Enabling and Disabling

Each DMA channel is enabled by writing a one to the Transfer Enable bit in the Control Register (CR.TEN) and disabled by writing a one to the Transfer Disable bit (CR.TDIS). The current status can be read from the Status Register (SR).

While the PDCA channel is enabled all DMA request will be handled as long the TCR and TCRR is not zero.

16.5.9 Interrupts

Interrupts can be enabled by writing a one to the corresponding bit in the Interrupt Enable Register (IER) and disabled by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The Interrupt Mask Register (IMR) can be read to see whether an interrupt is enabled or not. The current status of an interrupt source can be read through the Interrupt Status Register (ISR).

The PDCA has three interrupt sources:

- Reload Counter Zero - The TCRR register is zero.
- Transfer Finished - Both the TCR and TCRR registers are zero.
- Transfer Error - An error has occurred in accessing memory.

16.5.10 Priority

If more than one PDCA channel is requesting transfer at a given time, the PDCA channels are prioritized by their channel number. Channels with lower numbers have priority over channels with higher numbers, giving channel zero the highest priority.

16.5.11 Error Handling

If the Memory Address Register (MAR) is set to point to an invalid location in memory, an error will occur when the PDCA tries to perform a transfer. When an error occurs, the Transfer Error bit in the Interrupt Status Register (ISR.TERR) will be set and the DMA channel that caused the error will be stopped. In order to restart the channel, the user must program the Memory Address Register to a valid address and then write a one to the Error Clear bit in the Control Register (CR.ECLR). If the Transfer Error interrupt is enabled, an interrupt request will be generated when a transfer error occurs.

16.5.12 Peripheral Event Trigger

Peripheral events can be used to trigger PDCA channel transfers. Peripheral Event synchronizations are enabled by writing a one to the Event Trigger bit in the Mode Register (MR.ETRIG). When set, all DMA requests will be blocked until a peripheral event is received. For each peripheral event received, only one data item is transferred. If no DMA requests are pending when a peripheral event is received, the PDCA will start a transfer as soon as a peripheral event is detected. If multiple events are received while the PDCA channel is busy transferring data, an overflow condition will be signaled in the Peripheral Event System. Refer to Section 31. ”Peripheral Event Controller (PEVC)” on page 845 for more information.

16.6 User Interface

16.6.1 Memory Map Overview

Table 16-1. PDCA Register Memory Map

Address Range	Contents
0x000 - 0x03F	DMA channel 0 configuration registers
0x040 - 0x07F	DMA channel 1 configuration registers
...	...
(0x000 - 0x03F)+m*0x040	DMA channel m configuration registers
0x834	Version register

The channels are mapped as shown in Table 16-1. Each channel has a set of configuration registers, shown in Table 16-2, where *n* is the channel number.

16.6.2 Channel Memory Map

Table 16-2. PDCA Channel Configuration Registers

Offset	Register	Register Name	Access	Reset
0x000 + n*0x040	Memory Address Register	MAR	Read/Write	0x00000000
0x004 + n*0x040	Peripheral Select Register	PSR	Read/Write	- (1)
0x008 + n*0x040	Transfer Counter Register	TCR	Read/Write	0x00000000
0x00C + n*0x040	Memory Address Reload Register	MARR	Read/Write	0x00000000
0x010 + n*0x040	Transfer Counter Reload Register	TCRR	Read/Write	0x00000000
0x014 + n*0x040	Control Register	CR	Write-only	0x00000000
0x018 + n*0x040	Mode Register	MR	Read/Write	0x00000000
0x01C + n*0x040	Status Register	SR	Read-only	0x00000000
0x020 + n*0x040	Interrupt Enable Register	IER	Write-only	0x00000000
0x024 + n*0x040	Interrupt Disable Register	IDR	Write-only	0x00000000
0x028 + n*0x040	Interrupt Mask Register	IMR	Read-only	0x00000000
0x02C + n*0x040	Interrupt Status Register	ISR	Read-only	0x00000000

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

16.6.3 Version Register Memory Map

Table 16-3. PDCA Version Register Memory Map

Offset	Register	Register Name	Access	Reset
0x834	Version Register	VERSION	Read-only	- (1)

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

16.6.4 Memory Address Register

Name: MAR

Access Type: Read/Write

Offset: 0x000 + n*0x040

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
MADDR[31:24]							

23	22	21	20	19	18	17	16
MADDR[23:16]							

15	14	13	12	11	10	9	8
MADDR[15:8]							

7	6	5	4	3	2	1	0
MADDR[7:0]							

- **MADDR: Memory Address**
  Address of memory buffer. MADDR should be programmed to point to the start of the memory buffer when configuring the PDCA. During transfer, MADDR will point to the next memory location to be read/written.

16.6.5 Peripheral Select Register

Name: PSR

Access Type: Read/Write

Offset: 0x004 + n*0x040

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
PID							

- **PID: Peripheral Identifier**

  The Peripheral Identifier selects which peripheral should be connected to the DMA channel. Writing a PID will select both which handshake interface to use, the direction of the transfer and also the address of the Receive/Transfer Holding Register for the peripheral. See the Module Configuration section of PDCA for details. The width of the PID field is device specific and dependent on the number of peripheral modules in the device.

16.6.6 Transfer Counter Register

Name: TCR

Access Type: Read/Write

Offset: 0x008 + n*0x040

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
TCV[15:8]							

7	6	5	4	3	2	1	0
TCV[7:0]							

- **TCV: Transfer Counter Value**
  Number of data items to be transferred by the PDCA. TCV must be programmed with the total number of transfers to be made. During transfer, TCV contains the number of remaining transfers to be done.

16.6.7 Memory Address Reload Register

Name: MARR

Access Type: Read/Write

Offset: 0x00C + n*0x040

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
MARV[31:24]							

23	22	21	20	19	18	17	16
MARV[23:16]							

15	14	13	12	11	10	9	8
MARV[15:8]							

7	6	5	4	3	2	1	0
MARV[7:0]							

- **MARV: Memory Address Reload Value**
  Reload Value for the MAR register. This value will be loaded into MAR when TCR reaches zero if the TCRR register has a non-zero value.

16.6.8 Transfer Counter Reload Register

Name: TCRR

Access Type: Read/Write

Offset: 0x010 + n*0x040

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
TCRV[15:8]							

7	6	5	4	3	2	1	0
TCRV[7:0]							

- **TCRV: Transfer Counter Reload Value**
  Reload value for the TCR register. When TCR reaches zero, it will be reloaded with TCRV if TCRV has a positive value. If TCRV is zero, no more transfers will be performed for the channel. When TCR is reloaded, the TCRR register is cleared.

16.6.9 Control Register

Name: CR

Access Type: Write-only

Offset: 0x014 + n*0x040

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ECLR

7	6	5	4	3	2	1	0
-	-	-	-	-	-	TDIS	TEN

- **ECLR: Transfer Error Clear**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Transfer Error bit in the Status Register (SR.TERR). Clearing the SR.TERR bit will allow the channel to transmit data. The memory address must first be set to point to a valid location.
- **TDIS: Transfer Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will disable transfer for the DMA channel.
- **TEN: Transfer Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will enable transfer for the DMA channel.

16.6.10 Mode Register

Name: MR

Access Type: Read/Write

Offset: 0x018 + n*0x040

Reset Value: 0x00000000

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

- **RING: Ring Buffer**
  0:The Ring buffer functionality is disabled.
  1:The Ring buffer functionality is enabled. When enabled, the reload registers, MARR and TCRR will not be cleared after reload.
- **ETRIG: Event Trigger**
  0:Start transfer when the peripheral selected in Peripheral Select Register (PSR) requests a transfer.
  1:Start transfer only when or after a peripheral event is received.
- **SIZE: Size of Transfer**

Table 16-4. Size of Transfer

SIZE	Size of Transfer
0	Byte
1	Halfword
2	Word
3	Reserved

16.6.11 Status Register

Name: SR

Access Type: Read-only

Offset: 0x01C + n*0x040

Reset Value: 0x00000000

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

- **TEN: Transfer Enabled**
  This bit is cleared when the TDIS bit in CR is written to one.
  This bit is set when the TEN bit in CR is written to one.
  0: Transfer is disabled for the DMA channel.
  1: Transfer is enabled for the DMA channel.

16.6.12 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x020 + n*0x040

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

16.6.13 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x024 + n*0x040

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

16.6.14 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x028 + n*0x040

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	TERR	TRC	RCZ

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

16.6.15 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x02C + n*0x040

Reset Value: 0x00000000

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

- **TERR: Transfer Error**
  This bit is cleared when no transfer errors have occurred since the last write to CR.ECLR.
  This bit is set when one or more transfer errors has occurred since reset or the last write to CR.ECLR.
- **TRC: Transfer Complete**
  This bit is cleared when the TCR and/or the TCRR holds a non-zero value.
  This bit is set when both the TCR and the TCRR are zero.
- **RCZ: Reload Counter Zero**
  This bit is cleared when the TCRR holds a non-zero value.
  This bit is set when TCRR is zero.

16.6.16 PDCA Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x834

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

16.7 Module Configuration

The specific configuration for each PDCA instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 16-5. PDCA Configuration

Feature	PDCA
Number of channels	16
Number of performance monitors	0

Table 16-6. Module Clock Name

Module name	Clock Name	Description
PDCA	CLK_PDCA_HSB	Clock for the PDCA HSB interface
	CLK_PDCA_PB	Clock for the PDCA PB interface

Table 16-7. Register Reset Values

Register	Reset Value
PSR CH 0	0
PSR CH 1	1
PSR CH 2	2
PSR CH 3	3
PSR CH 4	4
PSR CH 5	5
PSR CH 6	6
PSR CH 7	7
PSR CH 8	8
PSR CH 9	9
PSR CH 10	10
PSR CH 11	11
PSR CH 12	12
PSR CH 13	13
PSR CH 14	14
PSR CH 15	15
VERSION	124

The table below defines the valid Peripheral Identifiers (PIDs). The direction is specified as observed from the memory, so RX means transfers from peripheral to memory and TX means from memory to peripheral.

Table 16-8. Peripheral Identity Values

PID	Direction	Peripheral Instance	Peripheral Register
0	RX	USART0	RHR
1	RX	USART1	RHR
2	RX	USART2	RHR
3	RX	USART3	RHR
4	RX	SPI	RDR
5	RX	TWIM0	RHR
6	RX	TWIM1	RHR
7	RX	TWIM2	RHR
8	RX	TWIM3	RHR
9	RX	TWIS0	RHR
10	RX	TWIS1	RHR
11	RX	ADCIFE	LCV
12	RX	CATB	Multiple
13	-	-	-
14	RX	IISC	RHR (CH0)
15	RX	IISC	RHR (CH1)
16	RX	PARC	RHR
17	RX	AESA	ODATA
18	TX	USART0	THR
19	TX	USART1	THR
20	TX	USART2	THR
21	TX	USART3	THR
22	TX	SPI	TDR
23	TX	TWIM0	THR
24	TX	TWIM1	THR
25	TX	TWIM2	THR
26	TX	TWIM3	THR
27	TX	TWIS0	THR
28	TX	TWIS1	THR
29	TX	ADCIFE	CDMA
30	TX	CATB	Multiple
31	TX	ABDACB	SDR0
32	TX	ABDACB	SDR1

Table 16-8. Peripheral Identity Values

PID	Direction	Peripheral Instance	Peripheral Register
33	TX	IISC	THR (CH0)
34	TX	IISC	THR (CH1)
35	TX	DACC	CDR
36	TX	AESA	IDATA
37	TX	LCDCA	ACMDR
38	TX	LCDCA	ABMDR

17. USB Device and Embedded Host Interface (USBC)

Rev: 3.1.0.19

17.1 Features

- **Compatible with the USB 2.0 specification**
- **Supports full (12Mbit/s) and low (1.5Mbit/s) speed communication**
- **Supports Embedded Host**
- **8 physical pipes/endpoints in ping-pong mode**
- **Flexible pipe/endpoint configuration and reallocation of data buffers in embedded RAM**
- **Supports an endpoint numbering range from 0 to 15, in both I/O directions**
- **Supports an infinite number of virtual pipes (alternate pipe)**
- **Up to two memory banks per pipe/endpoint**
- **Built-in DMA with multi-packet support through ping-pong mode**
- **On-chip transceivers with built-in pull-ups and pull-downs**
- **On-chip Embedded Host pad with a VBUS analog comparator**

17.2 Overview

The Universal Serial Bus interface (USBC) module complies with the Universal Serial Bus (USB) 2.0 specification supporting both the device and the embedded host mode.

Each pipe/endpoint can be configured into one of several transfer types. It can be associated with one or more memory banks (located inside the embedded system or CPU RAM) used to store the current data payload. If two banks are used (“ping-pong” mode), then one bank is read or written by the CPU (or any other AHB master) while the other is read or written by the USBC core.

Table 17-1 describes the hardware configuration of the USBC module.

Table 17-1. Description of USB pipes/endpoints

pipe/endpoint	Mnemonic	Max. size	Number of available banks	Type
0	PEP0	1023 bytes	1	Control/Isochronous/Bulk/Interrupt
1	PEP1	1023 bytes	2	Control/Isochronous/Bulk/Interrupt
2	PEP2	1023 bytes	2	Control/Isochronous/Bulk/Interrupt
...	...	...	...	...
7	PEP7	1023 bytes	2	Control/Isochronous/Bulk/Interrupt

17.3 Block Diagram

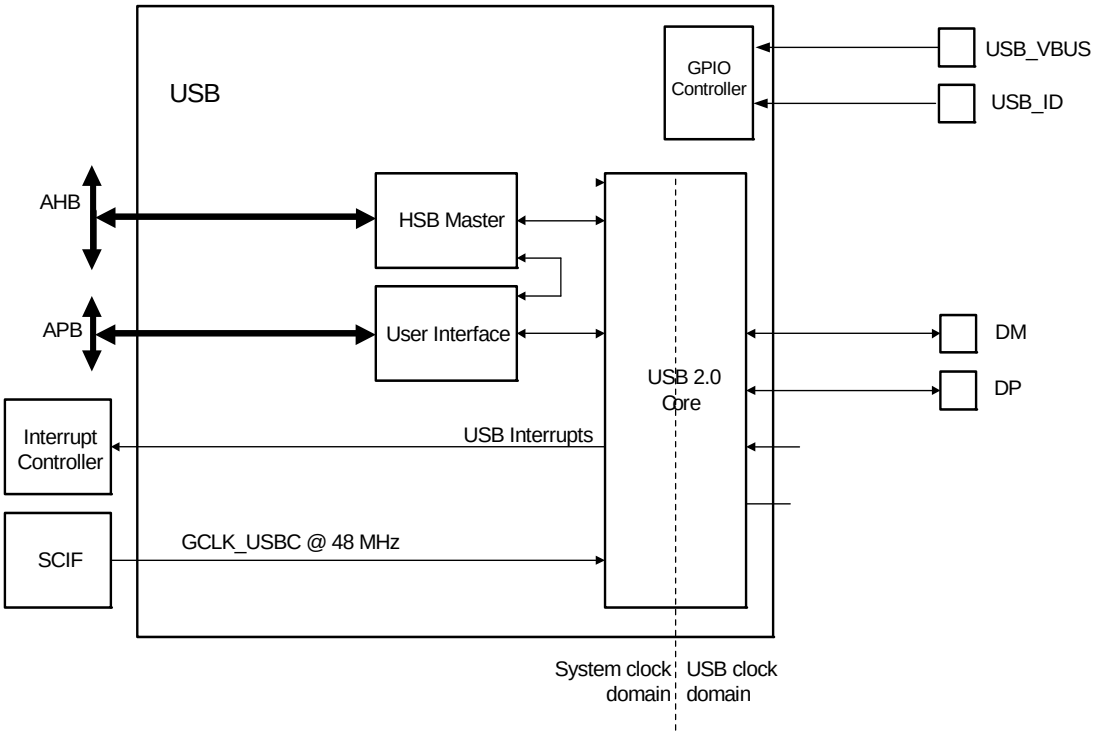
The USBC interfaces a USB link with a data flow stored in the embedded ram (CPU or AHB).

The USBC requires a 48MHz ± 0.25% reference clock, which is the USB generic clock. For more details see ”Clocks” on page 344. The 48MHz clock is used to generate either a 12MHz full-speed or a 1.5MHz low-speed bit clock from the received USB differential data, and to transmit data according to full- or low-speed USB device tolerances. Clock recovery is achieved by a digital phase-locked loop (a DPLL, not represented) in the USBC module, which complies with the USB jitter specifications.

The USBC module consists of:

- HSB master interface
- User interface
- USB Core
- Transceiver pads

Figure 17-1. USBC Block Diagram



17.4 I/O Lines Description

Table 17-2. I/O Lines Description

PIn Name	Pin Description	Type	Active Level
DM	Data -: Differential Data Line - Port	Input/Output	
DP	Data +: Differential Data Line + Port	Input/Output	

17.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

17.5.1 I/O Lines

The USBC pins may be multiplexed with the I/O Controller lines. The user must first configure the I/O Controller to assign the desired USBC pins to their peripheral functions.

The USB VBUS and ID pin lines should be connected to GPIO pins and the user should monitor this with software.

17.5.2 Power Management

If the CPU enters a Power Save Mode that disables clocks used by the USBC, the USBC will stop functioning and resume operation after the system wakes up from Power Save Mode.

17.5.3 Clocks

The USBC has two bus clocks connected: One High Speed Bus clock (CLK_USBC_AHB) and one Peripheral Bus clock (CLK_USBC_APB). These clocks are generated by the Power Manager. Both clocks are enabled at reset, and can be disabled by the Power Manager. It is recommended to disable the USBC before disabling the clocks, to avoid freezing the USBC in an undefined state.

To follow the usb data rate at 12Mbit/s in full-speed mode, the CLK_USBC_AHB clock should be at minimum 12MHz.

The 48MHz USB clock is generated by a dedicated generic clock from the SCIF module. Before using the USB, the user must ensure that the USB generic clock (GCLK_USBC) is enabled at 48MHz in the SCIF module.

17.5.4 Interrupts

The USBC interrupt request line is connected to the NVIC. Using the USBC interrupt requires the NVIC to be programmed first.

The USBC asynchronous interrupts can wake the CPU from any Power Save Mode:

- The ID Transition Interrupt (IDTI)
- The Wakeup Interrupt (WAKEUP)
- The Host Wakeup Interrupt (HWUPI)

17.6 Functional Description

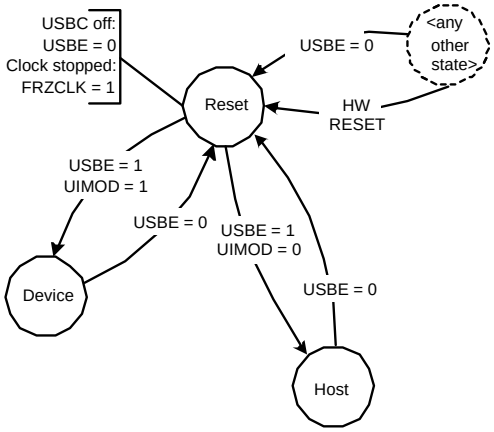
17.6.1 USB General Operation

17.6.1.1 Initialization

After a hardware reset, the USBC is disabled. When enabled, the USBC runs in either device mode or in host mode according to the ID detection.

The USBC Mode (USBCON.UIMOD) should be configured to set the USBC controller in either Host mode or Device mode.

Figure 17-2. General states



After a hardware reset, the USBC is in the Reset state. In this state:

- The module is disabled. The USBC Enable bit in the General Control register (USBCON.USBE) is reset.
- The module clock is stopped in order to minimize power consumption. The Freeze USB Clock bit in USBCON (USBCON.FRZCLK) is set.
- The USB pad is in suspend mode.
- The internal states and registers of the device and host modes are reset.
- The Freeze USB Clock (FRZCLK), USBC Enable (USBE), USBC Mode (UIMOD) in USBCON, and the Low-Speed mode bit in the Device General Control register (UDCON.LS) can be written to by software, so that the user can configure pads and speed before enabling the module. These values are only taken into account once the module has been enabled and unfrozen.

After writing a one to USBCON.USBE, the USBC enters device or host mode (according to UIMOD) in idle state.

Refer to Section 17.6.2 for the basic operation of the device mode.

Refer to Section 17.6.3 for the basic operation of the host mode.

The USBC can be disabled at any time by writing a zero to USBCON.USBE, this acts as a hardware reset, except that the FRZCLK and UIMOD bits in USBCON, and the LS bits in UDCON are not reset.

17.6.1.2 *Interrupts*

One interrupt vector is assigned to the USBC.

See Section 17.6.2.19 and Section 17.6.3.16 for further details about device and host interrupts.

See Section 17.5.4 for asynchronous interrupts.

17.6.1.3 *Frozen clock*

When the USB clock is frozen, it is still possible to access the following bits: UIMOD, FRZCLK, and USBE in the USBCON register, and LS in the UDCON register.

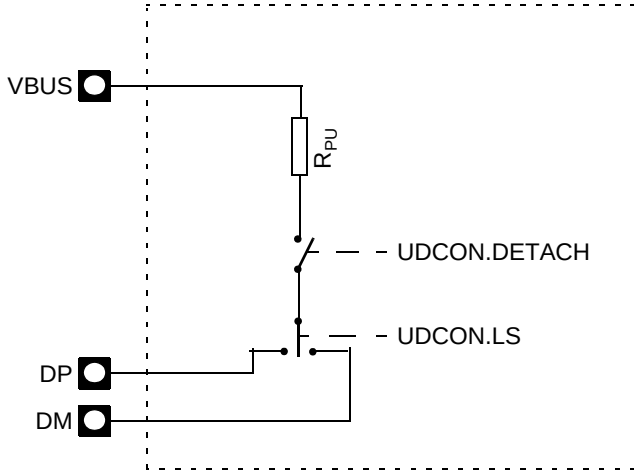
When FRZCLK is set, only the asynchronous interrupts can trigger a USB interrupt (see Section 17.5.4).

17.6.1.4 *Speed control*

- *Device mode*

When the USBC interface is in device mode, the speed selection is done by the UDCON.LS bit, connecting an internal pull-up resistor to either DP (full-speed mode) or DM (low-speed mode). The LS bit shall be written before attaching the device, which can be simulated by clearing the UDCON.DETACH bit.

Figure 17-3. Speed Selection in device mode



- *Host mode*

When the USBC interface is in host mode, internal pull-downs are enabled on both DP and DM. The interface detects the speed of the connected device and reflects this in the Speed Status field (USBSTA.SPEED).

17.6.1.5 *Data management*

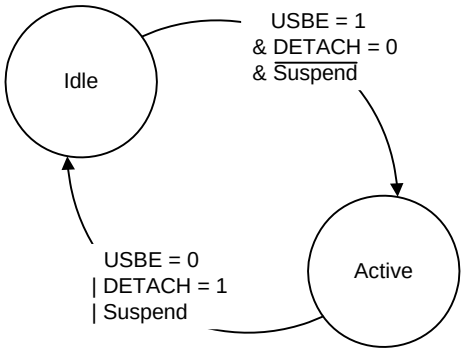
Endpoints and pipe buffers can be allocated anywhere in the embedded memory (CPU RAM or HSB RAM).

See "RAM management" on page 350.

17.6.1.6 Pad Suspend

Figure 17-4 illustrates the behavior of the USB pad in device mode.

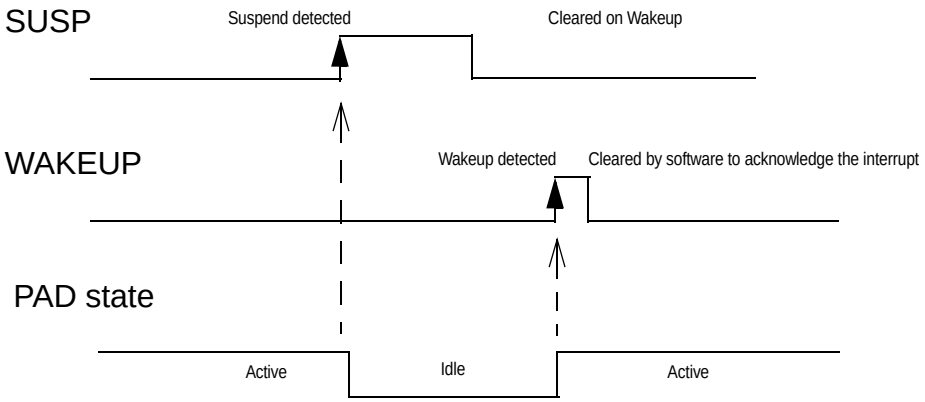
Figure 17-4. Pad Behavior



- In Idle state, the pad is in low power consumption mode.
- In Active state, the pad is working.

Figure 17-5 illustrates the pad events leading to a PAD state change.

Figure 17-5. Pad events



The Suspend Interrupt bit in the Device Global Interrupt register (UDINT.SUSP) is set when a USB Suspend state has been detected on the USB bus. This event automatically puts the USB pad in the Idle state. The detection of a non-idle event sets WAKEUP and wakes the USB pad.

The pad goes to the Idle state if the module is disabled or if UDCON.DETACH is written to one. It returns to the Active state when USBCON.USBE is written to one and DETACH is written to zero.

17.6.2 USBC Device Mode Operation

17.6.2.1 Device Enabling

In device mode, the USBC supports full- and low-speed data transfers.

Including the default control endpoint, a total of 8 endpoints are provided. They can be configured as isochronous, bulk or interrupt types, as described in Table 17-1 on page 341

After a hardware reset, the USBC device mode is in the reset state (see Section 17.6.1.1). In this state, the endpoint banks are disabled and neither DP nor DM are pulled up (DETACH is one).

DP or DM will be pulled up according to the selected speed as soon as the DETACH bit is written to zero. See “Device mode” for further details.

When the USBC is enabled (USBE is one) in device mode, it enters the Idle state, minimizing power consumption. Being in Idle state does not require the USB clocks to be activated.

The USBC device mode can be disabled or reset at any time by disabling the USBC (by writing a zero to USBE) or by enabling host mode (ID is zero).

17.6.2.2 USB reset

The USB bus reset is initiated by a connected host and managed by hardware.

When a USB reset state is detected on the USB bus, the following operations are performed by the controller:

- UDCON register is reset except for the DETACH and SPDCONF bits.
- Device Frame Number Register (UDFNUM), Endpoint n Configuration Register (UECFGn), and Endpoint n Control Register (UECONn) registers are cleared.
- The data toggle sequencing in all the endpoints are cleared.
- At the end of the reset process, the End of Reset (EORST) bit in the UDINT register is set.

17.6.2.3 Endpoint activation

When an endpoint is disabled (UERST.EPENn = 0) the data toggle sequence, Endpoint n Status Set (UESTAn), and UECONn registers will be reset. The controller ignores all transactions to this endpoint as long as it is inactive.

To complete an endpoint activation, the user should fill out the endpoint descriptor: see Figure 17-6 on page 351.

17.6.2.4 Endpoint redirection

By default, the Redirected endpoint number (UECFGn.REPNB) field is zero, and any transaction targeted to endpoint n will be associated to the physical endpoint n.

By configuring the UECFGn.REPNB field to a value m, any transaction targeted to this endpoint m will be redirected to the physical endpoint n. This allows two virtual endpoints, IN and OUT, to be configured towards the same endpoint number.

17.6.2.5 Data toggle sequence

In order to respond to a CLEAR_FEATURE USB request without disabling the endpoint, the user can clear the data toggle sequence by writing a one to the Reset Data Toggle Set bit in the Endpoint n Control Set register (UECONnSET.RSTDTS)

17.6.2.6 Busy bank enable

In order to make an endpoint bank look busy regardless of its actual state, the user can write a one to the Busy Bank Enable bit in the Endpoint n Control Register (UECONnSET.BUSY0/1ES).

If a BUSYnE bit is set, any transaction to this bank will be rejected with a NAK reply.

17.6.2.7 Address setup

The USB device address is set up according to the USB protocol.

- After all kinds of resets, the USB device address is 0.
- The host starts a SETUP transaction with a SET_ADDRESS(addr) request.
- The user writes this address to the USB Address field (UDCON.UADD), and writes a zero to the Address Enable bit (UDCON.ADDEN), resulting in the address remaining zero.
- The user sends a zero-length IN packet from the control endpoint.
- The user enables the stored USB device address by writing a one to ADDEN.

Once the USB device address is configured, the controller filters the packets to only accept those targeting the address stored in UADD.

UADD and ADDEN should not be written to simultaneously. They should be written sequentially, UADD field first.

If UADD or ADDEN is cleared, the default device address 0 is used. UADD and ADDEN are cleared:

- On a hardware reset.
- When the USBC is disabled (USBE written to zero).
- When a USB reset is detected.

17.6.2.8 Suspend and Wakeup

When an idle USB bus state has been detected for 3 ms, the controller sets the Suspend (SUSP) interrupt bit in UDINT. In this case, the transceiver is suspended, reducing power consumption.

To further reduce power consumption it is recommended to freeze the USB clock by writing a one to the Freeze USB Clock (FRZCLK) bit in USBCON when the USB bus is in suspend mode. The MCU can also enter the power save mode mode to further lower power consumption.

To recover from the suspend mode, the user shall wait for the Wakeup (WAKEUP) interrupt bit, which is set when a non-idle event is detected, and then write a zero to FRZCLK.

As the WAKEUP interrupt bit in UDINT is set when a non-idle event is detected, it can occur regardless of whether the controller is in the suspend mode or not.

17.6.2.9 Detach

The reset value of the DETACH bit located in the UDCON register, is one.

It is possible to initiate a device re-enumeration simply by writing a one and then a zero to DETACH.

DETACH acts on the pull-up connections of the DP and DM pads. See “Device mode” for further details.

17.6.2.10 Remote wakeup

The remote wakeup request (also known as upstream resume) is the only request the device may send on its own initiative. This should be preceded by a DEVICE_REMOTE_WAKEUP request from the host.

- First, the USBC must have detected a “Suspend” state on the bus, i.e. the remote wakeup request can only be sent after a SUSP interrupt has been set.
- The user may then write a one to the remote wakeup (RMWKUP) bit in UDCON to send an Upstream Resume to the host initiating the wakeup. This will automatically be done by the controller after 5ms of inactivity on the USB bus.
- When the controller sends the Upstream Resume, the Upstream Resume (UPRSM) interrupt is set and SUSP is cleared.
- RMWKUP is cleared at the end of the transmitting Upstream Resume.
- In case of a rebroadcast resume initiated by the host, the End of Resume (EORSM) interrupt is set when the rebroadcast resume is completed.

17.6.2.11 RAM management

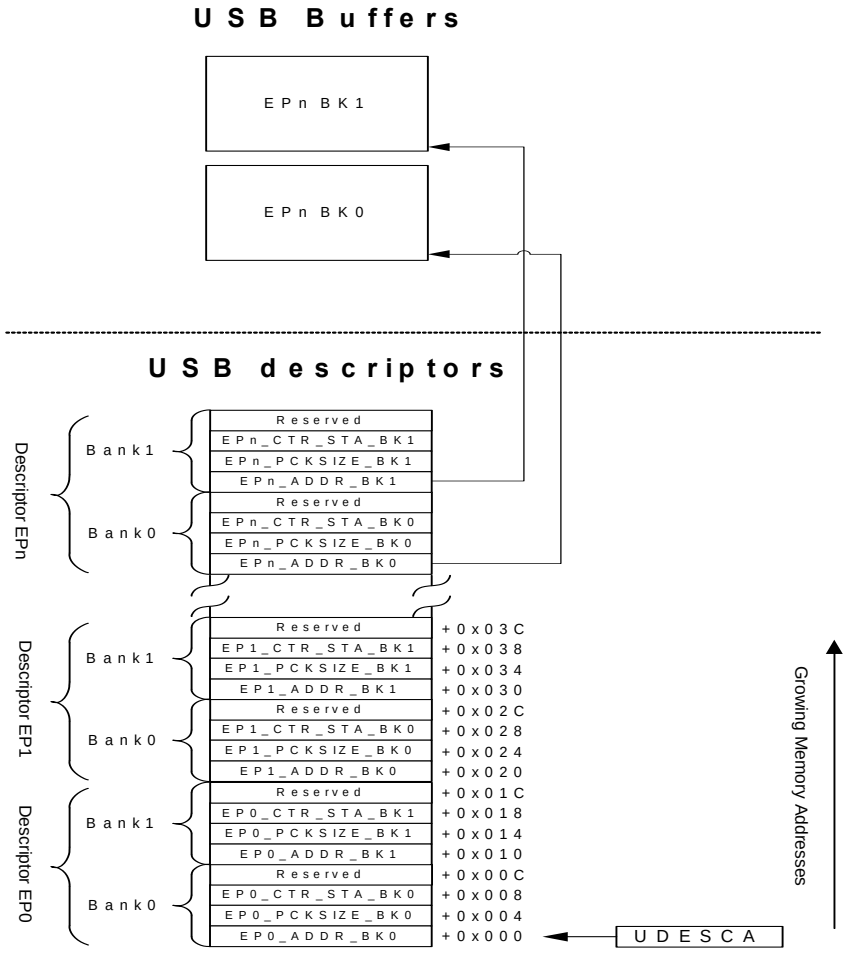
Endpoint data can be physically allocated anywhere in the embedded RAM. The USBC controller accesses these endpoints directly through the HSB master (built-in DMA).

The USBC controller reads the USBC descriptors to know where each endpoint is located. The base address of the USBC descriptor (UDESC.UDESCA) needs to be written by the user. The descriptors can also be allocated anywhere in the embedded RAM.

Before using an endpoint, the user should setup the endpoint address for each bank. Depending on the direction, the type, and the packet-mode (single or multi-packet), the user should also initialize the endpoint packet size, and the endpoint control and status fields, so that the USBC controller does not compute random values from the RAM.

When using an endpoint the user should read the UESTAX.CURRBK field to know which bank is currently being processed.

Figure 17-6. Memory organization



Each descriptor of an endpoint n consists of four words.

- The address of the endpoint and the bank used (EPn_ADDR_BK0/1).
- The packet size information for the endpoint and bank (EPn_PCKSIZE_BK0/1):

Table 17-3. EPn_PCKSIZE_BK0/1 structure

31	30:16	15	14:0
AUTO_ZLP	MULTI_PACKET_SIZE	-	BYTE_COUNT

– AUTO_ZLP: Auto zero length packet, see ”Multi packet mode for IN endpoints” on page 356.

– MULTI_PACKET_SIZE: see ”Multi packet mode and single packet mode.” on page 353.

– BYTE_COUNT: see ”Multi packet mode and single packet mode.” on page 353.

- The control and status fields for the endpoint and bank (EPn_CTR_STA_BK0/1):

Table 17-4. EPn_CTR_STA_BK0/1 structure

31:19	18	17	16	15:1	0
Status elements				Control elements	
-	UNDERF	OVERF	CRCERR	-	STALLRQ_NEXT

  - UNDERF: Underflow status for isochronous IN transfer. See "Data flow error" on page 359.
  - OVERF: Overflow status for isochronous OUT transfer. See "Data flow error" on page 359.
  - CRCERR: CRC error status for isochronous OUT transfer. See "CRC error" on page 359.
  - STALLRQ_NEXT: Stall request for the next transfer. See "STALL request" on page 352.

17.6.2.12 STALL request

For each endpoint, the STALL management is performed using:

- The STALL Request (STALLRQ) bit in UECONn is set to initiate a STALL request.
- The STALLed Interrupt (STALLEDI) bit in UESTAn is set when a STALL handshake has been sent.

To answer requests with a STALL handshake, STALLRQ has to be set by writing a one to the STALL Request Set (STALLRQS) bit. All following requests will be discarded (RXOUTI, etc. will not be set) and handshaked with a STALL until the STALLRQ bit is cleared, by receiving a new SETUP packet (for control endpoints) or by writing a one to the STALL Request Clear (STALLRQC) bit.

Each time a STALL handshake is sent, the STALLEDI bit is set by the USBC and the EPnINT interrupt is set.

The user can use the descriptor to manage STALL requests. The USBC controller reads the EPn_CTR_STA_BK0/1.STALLRQ_NEXT bit after successful transactions and if it is one the USBC controller will set UECON.STALLRQ. The STALL_NEXT bit will be cleared upon receiving a SETUP transaction and the USBC controller will then clear the STALLRQ bit.

- *Special considerations for control endpoints*

If a SETUP packet is received at a control endpoint where a STALL request is active, the Received SETUP Interrupt (RXSTPI) bit in UESTAn is set, and the STALLRQ and STALLEDI bits are cleared. It allows the SETUP to be always ACKed as required by the USB standard.

This management simplifies the enumeration process management. If a command is not supported or contains an error, the user requests a STALL and can return to the main task, waiting for the next SETUP request.

- *STALL handshake and retry mechanism*

The retry mechanism has priority over the STALL handshake. A STALL handshake is sent if the STALLRQ bit is set and if there is no retry required.

17.6.2.13 *Multi packet mode and single packet mode.*

Single packet mode is the default mode where one USB packet is managed per bank.

The multi-packet mode allows the user to manage data exceeding the maximum endpoint size (UECFGn.EPSIZE) for an endpoint bank across multiple packets without software intervention. This mode can also be coupled with the ping-pong mode.

- For an OUT endpoint, the EPn_PCKSIZE_BK0/1.MULTI_PACKET_SIZE field should be configured correctly to enable the multi-packet mode. See ”Multi packet mode for OUT endpoints” on page 358. For single packet mode, the MULTI_PACKET_SIZE should be initialized either to 0 or configured with a packet size less than EPSIZE (called a short packet), in order to allocate the exact amount of memory. In this case, any excessive words will not be written to memory.
- For an IN endpoint, the EPn_PCKSIZE_BK0/1.BYTE_COUNT field should be configured correctly to enable the multi-packet mode. See”Multi packet mode for IN endpoints” on page 356. For single packet mode, the BYTE_COUNT should be less than EPSIZE.

17.6.2.14 *Management of control endpoints*

- *Overview*

A SETUP request is always ACKed. When a new SETUP packet is received, the RXSTPI is set, but not the Received OUT Data Interrupt (RXOUTI) bit.

The FIFO Control (FIFOCON) bit in UECONn is irrelevant for control endpoints. The user should therefore never use it for these endpoints. When read, this value is always zero.

Control endpoints are managed using:

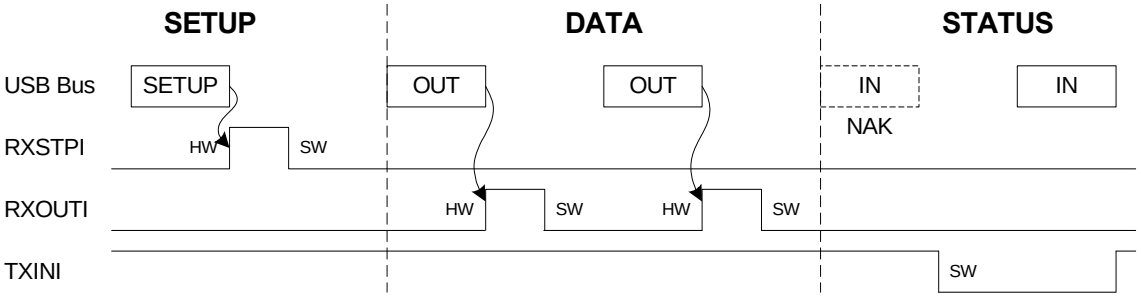
- The RXSTPI bit: is set when a new SETUP packet is received. This has to be cleared by firmware in order to acknowledge the packet and to free the bank.
- The RXOUTI bit: is set when a new OUT packet is received. This has to be cleared by firmware in order to acknowledge the packet and to free the bank.
- The Transmitted IN Data Interrupt (TXINI) bit: is set when the current bank is ready to accept a new IN packet. This has to be cleared by firmware in order to send the packet.
- The user can ignore any incoming transaction except SETUP by setting the UECONn.NOREPLY bit. This bit is automatically cleared by hardware when receiving a SETUP packet.

- *Control write*

Figure 17-7 on page 354 shows a control write transaction. During the status stage, the controller will not necessarily send a NAK on the first IN token:

- If the user knows the exact number of descriptor bytes that will be read, the status stage can be predicted, and a zero-length packet can be sent after the next IN token.
- Alternatively the bytes can be read until the NAKed IN Interrupt (NAKINI) is triggered, notifying that all bytes are sent by the host and that the transaction is now in the status stage.

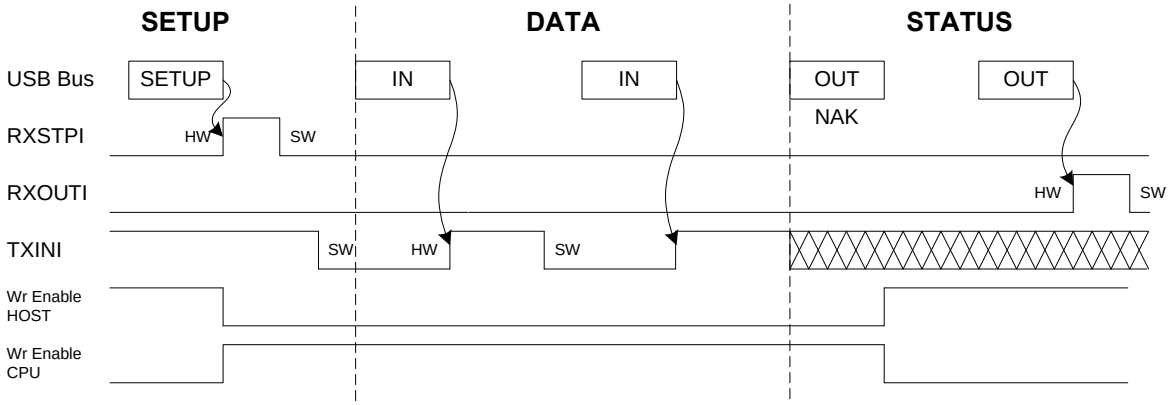
Figure 17-7. Control Write



• *Control read*

Figure 17-8 on page 354 shows a control read transaction. The USBC has to manage the simultaneous write requests from the CPU and USB host.

Figure 17-8. Control Read



A NAK handshake is always generated as the first status stage command. The UESTAn.NAKINI bit is set. It allows the user to know that the host aborts the IN data stage. As a consequence, the user should stop processing the IN data stage and should prepare to receive the OUT status stage by checking the UESTAn.RXOUTI bit.

The OUT retry is always ACKed. This OUT reception sets RXOUTI. Handle this with the following software algorithm:

```
// process the IN data stage
set TXINI
wait for RXOUTI (rising) OR TXINI (falling)
if RXOUTI is high, then process the OUT status stage
if TXINI is low, then return to process the IN data stage
```

Once the OUT status stage has been received, the USBC waits for a SETUP request. The SETUP request has priority over all other requests and will be ACKed.

17.6.2.15 Management of IN endpoints

- *Overview*

IN packets are sent by the USBC device controller upon IN requests from the host.

The endpoint and its descriptor in RAM must be pre configured (see section "RAM management" on page 350 for more details).

When the current bank is clear, the TXINI and FIFO Control (UECONn.FIFOCON) bits will be set simultaneously. This triggers an EPnINT interrupt if the Transmitted IN Data Interrupt Enable (TXINE) bit in UECONn is one.

TXINI shall be cleared by software (by writing a one to the Transmitted IN Data Interrupt Enable Clear bit in the Endpoint n Control Clear register (UECONnCLR.TXINIC)) to acknowledge the interrupt. This has no effect on the endpoint FIFO.

The user writes the IN data to the bank referenced by the EPn descriptor and allows the USBC to send the data by writing a one to the FIFO Control Clear (UECONnCLR.FIFOCONC) bit. This will also cause a switch to the next bank if the IN endpoint is composed of multiple banks. The TXINI and FIFOCON bits will be updated accordingly.

TXINI should always be cleared before clearing FIFOCON to avoid missing an TXINI event.

Figure 17-9. Example of an IN endpoint with one data bank

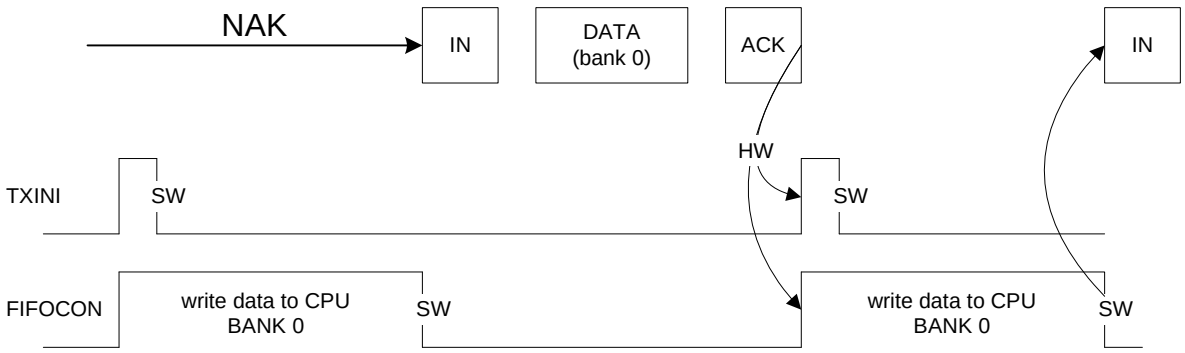
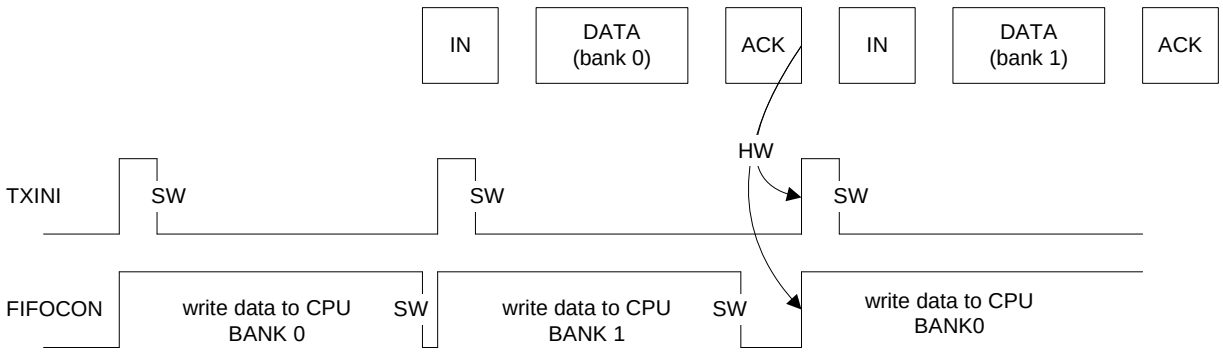


Figure 17-10. Example of an IN endpoint with two data banks



• *Detailed description*

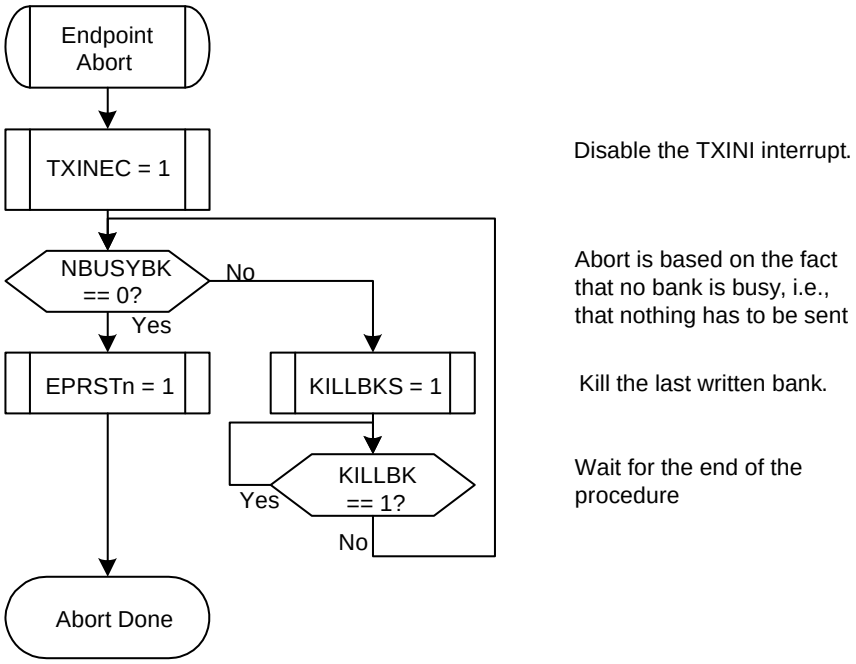
The data is written according to this sequence:

- When the bank is empty, TXINI and FIFOCON are set, which triggers an EPnINT interrupt if TXINE is one.
- The user acknowledges the interrupt by clearing TXINI.
- The user reads the UESTAX.CURRBK field to see which the current bank is.
- The user writes the data to the current bank, located in RAM as described by its descriptor: EPn_ADDR_BK0/1.
- The user should write the size of the IN packet into the USB descriptor: EPn_PCKSIZE_BK0/1.BYTE_COUNT.
- The user allows the controller to send the bank contents and switches to the next bank (if any) by clearing FIFOCON.

If the endpoint uses several banks, the current one can be written while the previous one is being read by the host. When the user clears FIFOCON, the next current bank may already be clear and TXINI is set immediately.

An “Abort” stage can be produced when a zero-length OUT packet is received during an IN stage of a control or isochronous IN transaction. The Kill IN Bank (KILLBK) bit in UECONn is used to kill the last written bank. The best way to manage this abort is to apply the algorithm represented on Figure 17-11 on page 356. See ”Endpoint n Control Register” on page 402 for more details about the KILLBK bit.

Figure 17-11. Abort Algorithm



• *Multi packet mode for IN endpoints*

In multi packet mode, the user can prepare n USB packets in the bank to be sent on a multiple IN transaction. The packet sizes will equal UECFGn.EPSIZE unless the AUTO_ZLP option is

set, or if the total byte count is not an integral multiple of EPSIZE, whereby the last packet should be short.

To enable the multi packet mode, the user should configure the endpoint descriptor (EPn_PCKSIZE_BK0/1.BYTE_COUNT) to the total size of the multi packet, which should be larger than the endpoint size (EPSIZE).

Since the EPn_PCKSIZE_BK0/1.MULTI_PACKET_SIZE is incremented (by the transmitted packet size) after each successful transaction, it should be set to zero when setting up a new multi packet transfer.

The EPn_PCKSIZE_BK0/1.MULTI_PACKET_SIZE is cleared by hardware when all the bank contents have been sent. The bank is considered as ready and the TX_IN flag is set when:

- A short packet (smaller than EPSIZE) has been transmitted.
- A packet has been successfully transmitted, the updated MULTI_PACKET_SIZE equals the BYTE_COUNT, and the AUTO_ZLP field is not set.
- An extra zero length packet has been automatically sent for the last transfer of the current bank, if BYTE_COUNT is a multiple of EPSIZE and AUTO_ZLP is set.

17.6.2.16 Management of OUT endpoints

- *Overview*

The endpoint and its descriptor in RAM must be pre configured, see section "RAM management" on page 350 for more details.

When the current bank is full, the RXOUTI and FIFO Control (UECONn.FIFOCON) bits will be set simultaneously. This triggers an EPnINT interrupt if the Received OUT Data Interrupt Enable (RXOUTE) bit in UECONn is one.

RXOUTI shall be cleared by software (by writing a one to the Received OUT Data Interrupt Clear (RXOUTIC) bit) to acknowledge the interrupt. This has no effect on the endpoint FIFO.

The user reads the OUT data from the RAM and clears the FIFOCON bit to free the bank. This will also cause a switch to the next bank if the OUT endpoint is composed of multiple banks.

RXOUTI should always be cleared before clearing FIFOCON to avoid missing an RXOUTI event.

Figure 17-12. Example of an OUT endpoint with one data bank

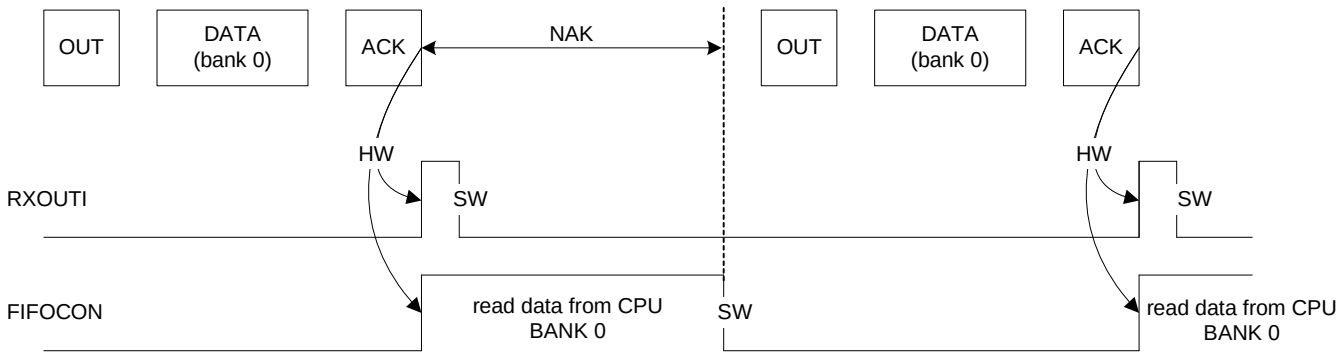
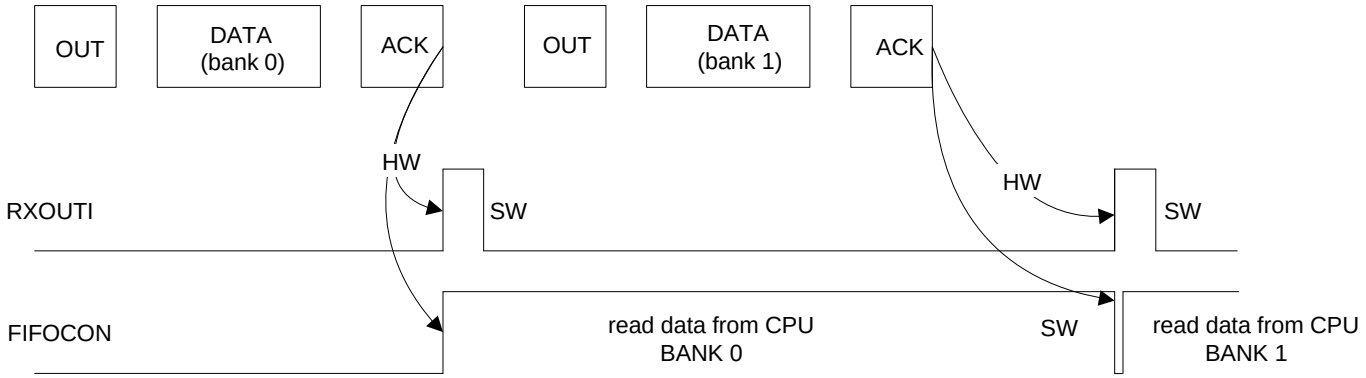


Figure 17-13. Example of an OUT endpoint with two data banks



• *Detailed description*

Before using the OUT endpoint, one should properly initialize its descriptor for each bank. See Figure 17-6 on page 351.

The data is read, according to this sequence:

- When the bank is full, RXOUTI and FIFOCON are set, which triggers an EPnINT interrupt if RXOUTE is one.
- The user acknowledges the interrupt by writing a one to RXOUTIC in order to clear RXOUTI.
- The user reads the UESTAX.CURRBK field to know the current bank number.
- The user reads the byte count of the current bank from the descriptor in RAM (EPn_PCKSIZE_BK0/1.BYTE_COUNT) to know how many bytes to read.
- The user reads the data in the current bank, located in RAM as described by its descriptor: EPn_ADDR_BK0/1.
- The user frees the bank and switches to the next bank (if any) by clearing FIFOCON.

If the endpoint uses several banks, the current one can be read while the next is being written by the host. When the user clears FIFOCON, the following bank may already be ready and RXOUTI will be immediately set.

• *Multi packet mode for OUT endpoints*

In multi packet mode, the user can extend the size of the bank allowing the storage of n USB packets in the bank.

To enable the multi packet mode, the user should configure the endpoint descriptor (EPn_PCKSIZE_BK0/1.MULTI_PACKET_SIZE) to match the size of the multi packet. This value should be a multiple of the endpoint size (UECFGn.EPSIZE).

Since the EPn_PCKSIZE_BK0/1.BYTE_COUNT is incremented (by the received packet size) after each successful transaction, it should be set to zero when setting up a new multi packet transfer.

As for single packet mode, the number of received data bytes is stored in the BYTE_CNT field.

The bank is considered as “valid” and the RX_OUT flag is set when:

- A packet has been successfully received and the updated BYTE_COUNT equals the MULTI_PACKET_SIZE.
- A short packet (smaller than EPSIZE) has been received.

17.6.2.17 Data flow error

This error exists only for isochronous IN/OUT endpoints. It sets the Errorflow Interrupt (ERRORFI) bit in UESTAn, which triggers an EPnINT interrupt if the Errorflow Interrupt Enable (ERRORFE) bit is one. The user can check the EPn_CTR_STA_BK0/1.UNDERF and OVERF bits in the endpoint descriptor to see which current bank has been affected.

- An underflow can occur during IN stage if the host attempts to read from an empty bank. A zero-length packet is then automatically sent by the USBC. The endpoint descriptor EPn_CTR_STA_BK0/1.UNDERF points out the bank from which the IN data should have originated. If a new successful transaction occurs, the UNDERF bit is overwritten to 0 only if the UESTAn.ERRORFI is cleared.
- An overflow can occur during the OUT stage if the host tries to send a packet while the bank is full. Typically this occurs when a CPU is not fast enough. The packet data is not written to the bank and is lost. The endpoint descriptor EPn_CTR_STA_BK0/1.OVERF points out which bank the OUT data was destined to. If the UESTAn.ERRORFI bit is cleared and a new transaction is successful, the OVERF bit will be overwritten to zero.

17.6.2.18 CRC error

This error exists only for isochronous OUT endpoints. It sets the CRC Error Interrupt (CRCERRI) bit in UESTAn, which triggers an EPnINT interrupt if the CRC Error Interrupt Enable (CRCERRE) bit is one.

A CRC error can occur during an isochronous OUT stage if the USBC detects a corrupted received packet. The OUT packet is stored in the bank as if no CRC error had occurred (RXOUTI is set).

The user can also check the endpoint descriptor to see which current bank is impacted by the CRC error by reading EPn_CTR_STA_BK0/1.CRCERR.

17.6.2.19 Interrupts

There are two kinds of device interrupts: processing, i.e. their generation is part of the normal processing, and exception, i.e. errors not related to CPU exceptions.

- *Global interrupts*

The processing device global interrupts are:

- The Suspend (SUSP) interrupt
- The Start of Frame (SOF) interrupt with no frame number CRC error (the Frame Number CRC Error (FNCERR) bit in the Device Frame Number (UDFNUM) register is zero)
- The End of Reset (EORST) interrupt
- The Wakeup (WAKEUP) interrupt
- The End of Resume (EORSM) interrupt
- The Upstream Resume (UPRSM) interrupt
- The Endpoint n (EPnINT) interrupt

The exception device global interrupts are:

- The Start of Frame (SOF) interrupt with a frame number CRC error (FNCERR is one)

• *Endpoint interrupts*

The processing device endpoint interrupts are:

- The Transmitted IN Data Interrupt (TXINI)
- The Received OUT Data Interrupt (RXOUTI)
- The Received SETUP Interrupt (RXSTPI)
- The Number of Busy Banks (NBUSYBK) interrupt

The exception device endpoint interrupts are:

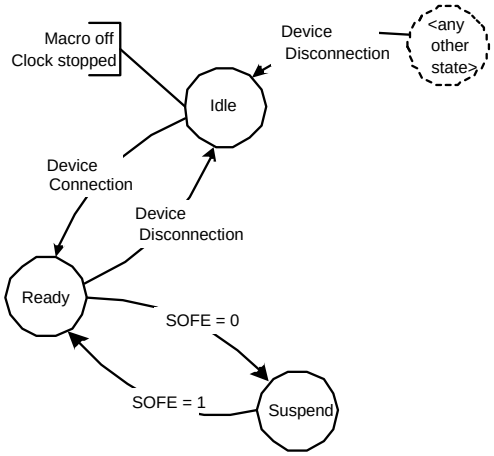
- The Errorflow Interrupt (ERRORFI)
- The NAKed OUT Interrupt (NAKOUTI)
- The NAKed IN Interrupt (NAKINI)
- The STALLed Interrupt (STALLEDI)
- The CRC Error Interrupt (CRCERRI)

17.6.3 USB Host Operation

17.6.3.1 Host Enabling

Figure 17-14 on page 361 describes the USBC host mode main states.

Figure 17-14. Host mode states



After a hardware reset, the USBC host mode is in the Reset state (see Section 17.6.1.1).

When the USBC is enabled (USBCON.USBE = 1) in host mode (USBCON.UIMOD = 0) it enters Idle state. As soon as the USBSTA.VBUSRQ bit is one, USBC waits for a device connection. Once a device is connected, the USBC enters the Ready state, which does not require the USB clock to be activated.

In host mode the USBC will suspend the USB bus by not transmitting any Start Of Frame (SOF) packets (the Start of Frame Generation Enable bit in the Host Global Interrupt register UHCON.SOFE is zero). The USBC enters the Suspend state when the USB bus is suspended, and exits when SOF generation is resumed.

17.6.3.2 Device detection

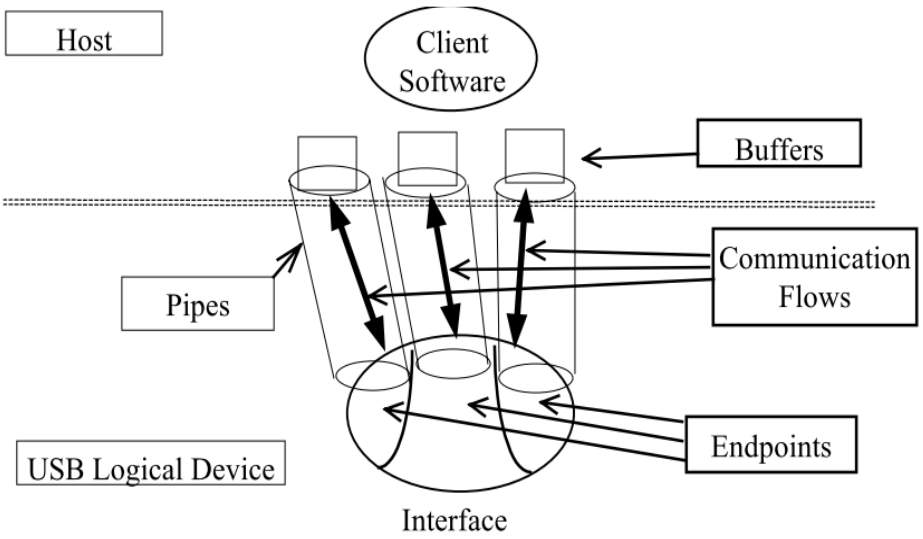
A device is detected by the USBC in host mode when DP or DM are not tied low, i.e., when a device DP or DM pull-up resistor is connected. To enable this detection, the host controller has to be notified that the VBUS is powered, which is done when USBSTA.VBUSRQ is one.

The device disconnection is detected by the host controller when both DP and DM are pulled down.

17.6.3.3 Description of pipes

In host mode, the term “pipe” is used instead of “endpoint”. A host pipe corresponds to a device endpoint, as illustrated by Figure 17-15 on page 362 from the USB specification.

Figure 17-15. USB Communication Flow



In host mode, the USBC associates a pipe to a device endpoint, according to the device configuration descriptors.

17.6.3.4 USB reset

The USBC sends a USB reset signal when the user writes a one to the Send USB Reset bit (UHCON.RESET). When the USB reset has been sent, the USB Reset Sent Interrupt bit in the Host Global Interrupt register (UHINT.RSTI) is set and all the pipes will be disabled.

If the bus was previously in a suspended state (UHCON.SOFE is zero) the USBC will switch it to the Resume state, causing the bus to asynchronously trigger the Host Wakeup Interrupt (UHINT.HWUPI). The SOFE bit will be set in order to generate SOF's immediately after the USB reset.

17.6.3.5 Pipe activation

A disabled pipe is inactive, and will be reset along with its context registers (UPCONn, UPSTAn, UPINRQn, and UPCFGn). Enabling a pipe is done by writing a one to the Pipe n Enable bit in the Pipe Enable/Reset Register (UPRST.PENn).

When starting an enumeration, the user gets the device descriptor by sending an GET_DESCRIPTOR USB request. This descriptor contains the maximal packet size of the device default control endpoint (bMaxPacketSize0) which the user should use to reconfigure the size of the default control pipe.

17.6.3.6 Address setup

Once the device has answered the first host requests with the default device address 0, the host assigns a new address to the device. The host controller has to send a USB reset to the device and a SET_ADDRESS(addr) SETUP request with the new address to be used by the device. Once this SETUP transaction is over, the user writes to the device address field in the "control and status 1 of endpoint n" word of the host's pipe n in the USB descriptor (Pn_CTR_STA1.PDADDR). All following requests by this pipe will be performed using this new address.

17.6.3.7 Remote wakeup

Writing UHCON.SOFE to zero when in host mode will cause the USBC to cease sending SOF's on the USB bus and enter the Suspend state. The USB device will enter the Suspend state 3ms later.

The device can awaken the host by sending an Upstream Resume (remote wakeup feature). When the host detects a non-idle state on the USB bus, it sets the Host Wakeup interrupt bit (UHINT.HWUPI). If the non-idle bus state corresponds to an Upstream Resume (K state), the Upstream Resume Received Interrupt bit (UHINT.RXRSMI) is set and the user has to generate a Downstream Resume within 1ms and for at least 20ms. It is required to first enter the Ready state by writing a one to UHCON.SOFEF and then writing a one to the Send USB Resume bit (UHCON.RESUME).

17.6.3.8 RAM management

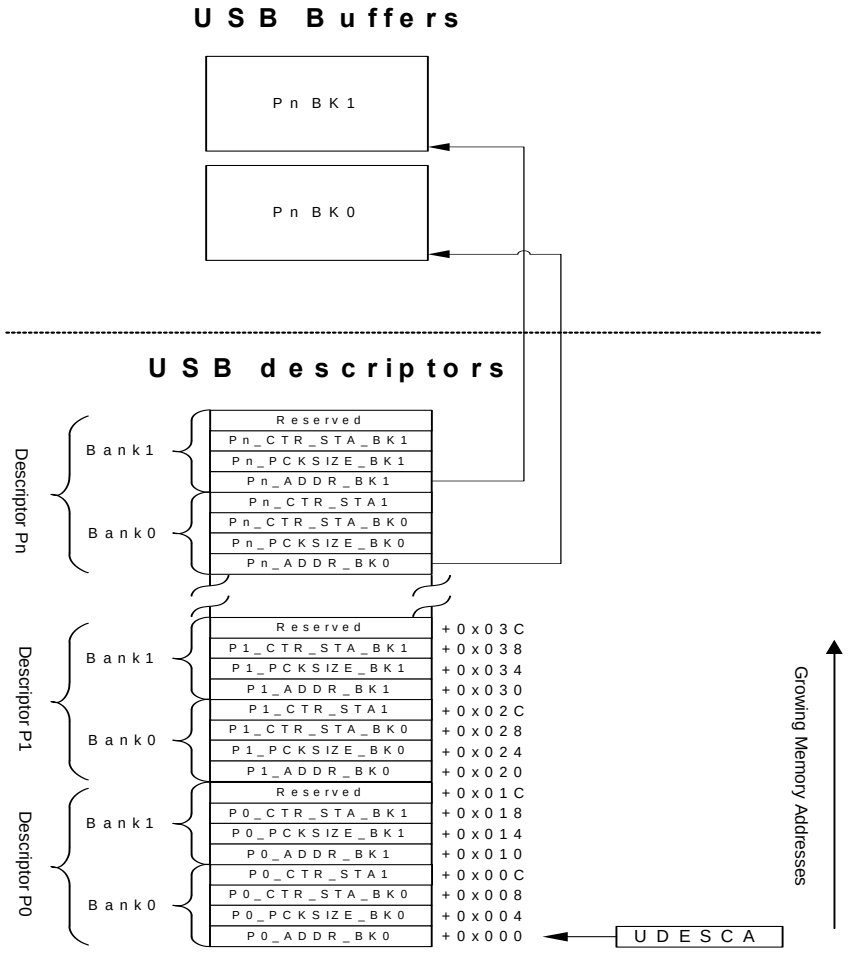
Pipe data can be physically allocated anywhere in the embedded RAM. The USBC controller accesses the pipes directly through the HSB master (built-in DMA).

The USBC controller reads the USBC descriptors to know the location of each pipe. The base address of this USBC descriptor (UDESC.UDESCA) needs to be written by the user. The descriptors can also be allocated anywhere in the embedded RAM.

Before using a pipe, the user should setup the data address for each bank. Depending on the direction, pipe type, targeted device address, targeted endpoint number, and packet-mode (single or multi-packet), the user should also initialize the pipe packet size and the pipe control and status field, so that the USB controller does not compute random values from the RAM.

When using a pipe, the user should read the UPSTAX.CURRBK field to know which bank is currently processed.

Figure 17-16. Memory organization



Each pipe n descriptor bank consists of four words.

- The address of the pipe and bank used (Pn_ADDR_BK0/1).
- The packet size information for the pipe and bank (Pn_PCKSIZE_BK0/1):

Table 17-5. Pn_PCKSIZE_BK0/1 structure

31	30:16	15	14:0
AUTO_ZLP	MULTI_PACKET_SIZE	-	BYTE_COUNT

– AUTO_ZLP: Auto zero length packet, see ”Multi packet mode for OUT pipes” on page 369.

– MULTI_PACKET_SIZE: see ”Multi packet mode and single packet mode.” on page 353.

– BYTE_COUNT: see ”Multi packet mode and single packet mode.” on page 353.

- The control and status fields for the pipe and bank (Pn_CTR_STA_BK0/1):

Table 17-6. Pn_CTR_STA_BK0/1 structure

31:19	18	17	16	15:0
Status				Control
-	UNDERF	OVERF	CRCERR	-

  – UNDERF: Underflow status for isochronous/Interrupt IN transfers. This status bit is set by hardware at the current bank (where the IN packet should have been stored). When a new successful transaction occurs this bit is overwritten to zero if UPSTAX.ERRORFI has previously been cleared by software. See "Data flow error" on page 369.

  – OVERF: Overflow status for isochronous/interrupt OUT transfers. This status bit is set by hardware at the current bank (where the OUT packet should have been loaded). When a new successful transaction occurs this bit is overwritten to zero if UPSTAX.ERRORFI has previously been cleared by software. See "Data flow error" on page 369.

  – CRCERR: CRC error status for isochronous IN transfers. See "CRC error" on page 369.

- The control and status 1 of endpoint n (Pn_CTR_STA1):

Table 17-7. Pn_CTR_STA1 structure

31:24	23:16	15:12	11:8	7	6:0
Status		Control			
-	PERSTA	PERMAX	PEPNUM	-	PDADDR

  – PERSTA: Pipe Error Status. See "PERSTA structure" table.

  – PERMAX: Should be set by the user. If the Pipe Error Counter (see Figure 17-8 on page 365) is larger than PERMAX, the UPSTAX.PERRI bit is set.

  – PEPNUM: Should be set by the user. Endpoint number for this pipe.

  – PDADDR: Should be set by the user. Device address for this pipe.

Table 17-8. PERSTA structure

23:21	20	19	18	17	16
ERCNT	CRC16ER	TOUTER	PIDER	DAPIDER	DTGLER

This field can be cleared by software. To avoid read-modify-write issues, the user should: freeze the pipe, wait until the UPSTAX.PFREEZE is one, clear the PERSTA field in memory, and then unfreeze the pipe.

  – ERCNT: Pipe Error Counter.

  – CRC16ER: Is set if a CRC16 error occurs during an isochronous IN transaction.

  – TOUTER: Is set if a Time-out error occurs during a USB transaction.

  – PIDER: Is set if a PID error occurs during a USB transaction.

  – DAPIDER: Is set if a Data PID error occurs during a USB transaction.

– DTGLER: Is set if a Data toggle error occurs during a USB transaction.

17.6.3.9 Multi packet mode and single packet mode.

See "Multi packet mode and single packet mode." on page 353 and just consider that an OUT pipe corresponds to an IN endpoint, and an IN pipe corresponds to an OUT endpoint.

17.6.3.10 Management of control pipes

A control transaction is composed of three stages:

- SETUP
- Data (IN or OUT)
- Status (OUT or IN)

The user has to change the pipe token according to each stage.

For control pipes only, the token is assigned a specific initial data toggle sequence:

- SETUP: Data0
- IN: Data1
- OUT: Data1

17.6.3.11 Management of IN pipes

• *Overview*

IN packets are sent by the USB device controller upon IN requests from the host. All the data can be read, acknowledging whether or not the bank is empty.

• *Detailed description*

The pipe and its descriptor in RAM must be pre configured.

The host can request data from the device in two modes, selected by writing to the IN Request Mode bit in the Pipe n IN Request register (UPINRQn.INMODE):

- When INMODE is written to zero, the USBC will perform INRQ IN requests before freezing the pipe.
- When INMODE is written to one, the USBC will perform IN requests as long as the pipe is not frozen by the user.

The generation of IN requests starts when the pipe is unfrozen (UPCONn.PFREEZE is zero).

When the current bank is full, the RXINI and FIFO Control (UPSTAn.FIFOCON) bits will be set simultaneously. This triggers a PnINT interrupt if the Received IN Data Interrupt Enable bit (UPCONn.RXINE) is one.

RXINI shall be cleared by software to acknowledge the interrupt. This is done by writing a one to the Received IN Data Interrupt Clear bit in the Pipe n Control Clear register (UPCONnCLR.RXINIC), which does not affect the pipe FIFO.

The user reads the byte count of the current bank from the descriptor in RAM (Pn_PCKSIZE_BK0/1.BYTE_COUNT) to know how many bytes should be read.

The user reads the IN data from the RAM and clears the FIFOCON bit to free the bank. This will also cause a switch to the next bank if the IN endpoint is composed of multiple banks. The RXINI and FIFOCON bits will be updated accordingly.

RXINI should always be cleared before clearing FIFOCON to avoid missing an RXINI event.

Figure 17-17. Example of an IN pipe with one data bank

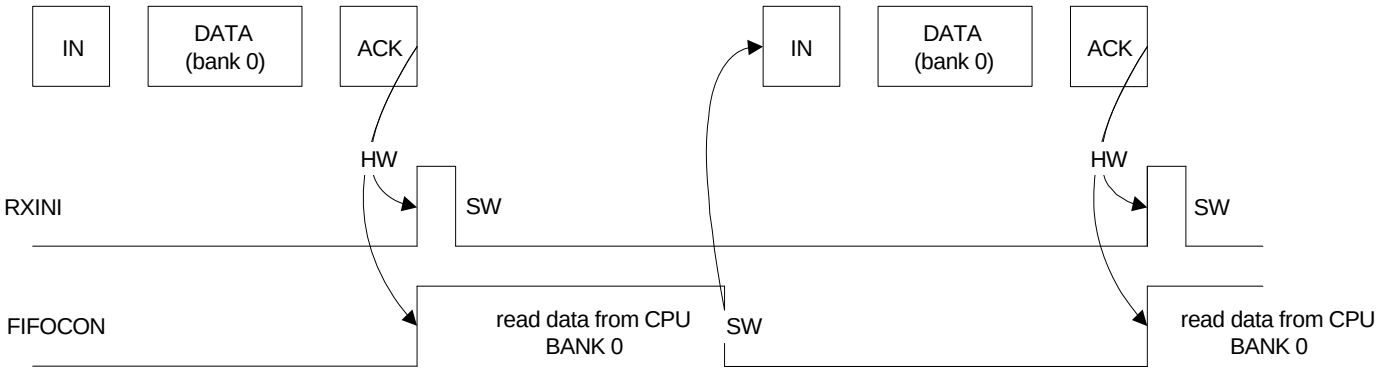
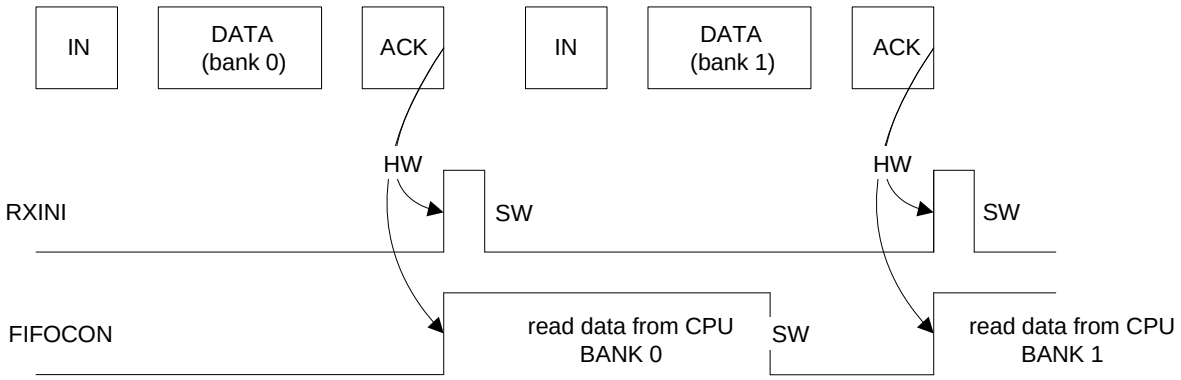


Figure 17-18. Example of an IN pipe with two data banks



- *Multi packet mode for IN pipes*

  See "Multi packet mode for OUT endpoints" on page 358 and just replace OUT endpoints with IN pipe.

17.6.3.12 *Management of OUT pipes*

- *Overview*

  OUT packets are sent by the host. All the data can be written, acknowledging whether or not the bank is full.

- *Detailed description*

  The pipe and its descriptor in RAM must be pre configured.

  When the current bank is clear, the Transmitted OUT Data Interrupt (TXOUTI) and FIFO Control (UPSTAn.FIFOCON) bits will be set simultaneously. This triggers a PnINT interrupt if the Transmitted OUT Data Interrupt Enable bit (UPCONn.TXOUTE) is one.

TXOUTI shall be cleared by software to acknowledge the interrupt. This is done by writing a one to the Transmitted OUT Data Interrupt Clear bit (UPCONnCLR.TXOUTIC), which does not affect the pipe FIFO.

The user writes the OUT data to the bank referenced to by the PEPn descriptor and allows the USBC to send the data by writing a one to the FIFO Control Clear (UPCONnCLR.FIFOCONC) bit. This will also cause a switch to the next bank if the OUT pipe is composed of multiple banks. The TXOUTI and FIFOCON bits will be updated accordingly

TXOUTI shall always be cleared before clearing FIFOCON to avoid missing an TXOUTI event.

Note that if the user decides to switch to the Suspend state (by writing a zero to UHCON.SOFE) while a bank is ready to be sent, the USBC automatically exits this state and sends the data.

Figure 17-19. Example of an OUT pipe with one data bank

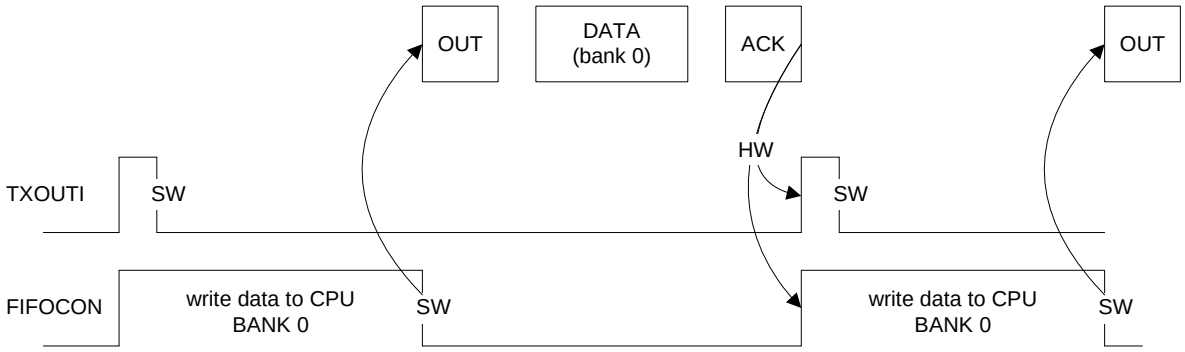


Figure 17-20. Example of an OUT pipe with two data banks and no bank switching delay

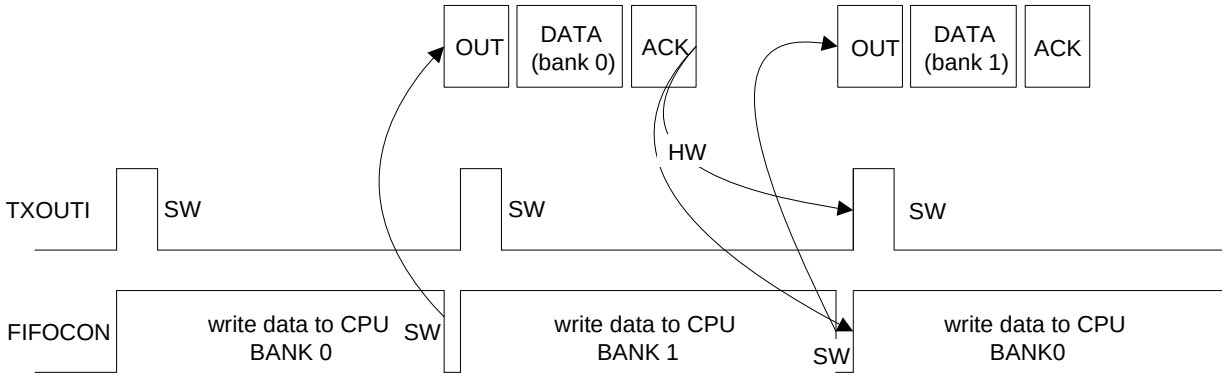
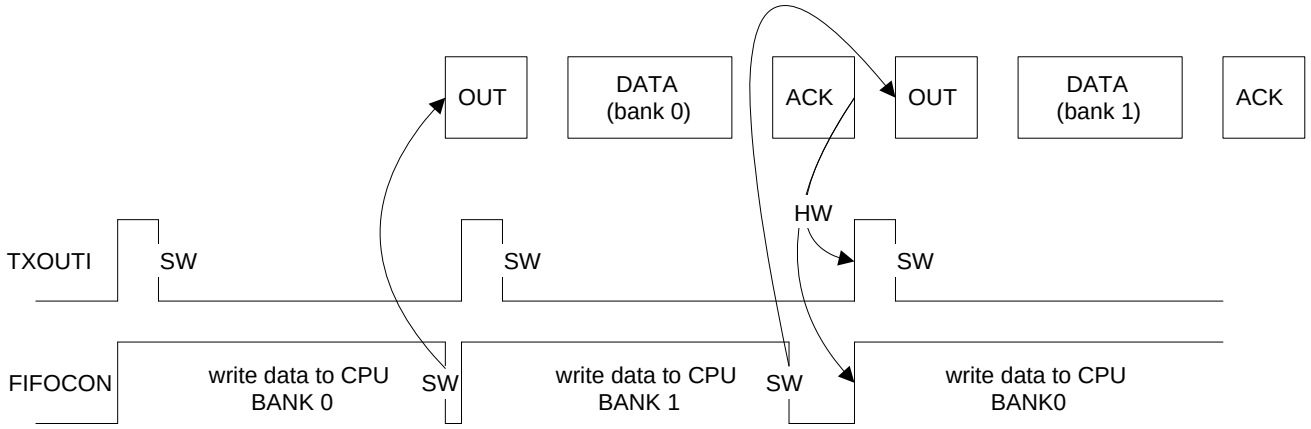


Figure 17-21. Example of an OUT pipe with two data banks and a bank switching delay



- *Multi packet mode for OUT pipes*

  See section "Multi packet mode for IN endpoints" on page 356 and just replace IN endpoints with OUT pipe.

17.6.3.13 *Alternate pipe*

The user has the possibility to run sequentially several logical pipes on the same physical pipe.

Before switching pipe, the user should save the pipe context (UPCFGn, UPCONn, UPSTAn, and the pipe descriptor table).

After switching pipe, the user should restore the pipe context, current bank number, and the current data toggle by using the UPCONn.INITDTGL and UPCONn.INITBK bits.

17.6.3.14 *Data flow error*

This error exists only for isochronous and interrupt pipes for both IN and OUT directions. It sets the Errorflow Interrupt (ERRORFI) bit in UPSTAn, which triggers an PnINT interrupt if the Errorflow Interrupt Enable (ERRORFE) bit is one. The user can check the Pn_CTR_STA_BK0/1.UNDERF and OVERF bits in the pipe descriptor to see which current bank has been affected.

- An overflow can occur during an OUT stage if the host attempts to send data from an empty bank. The pipe descriptor Pn_CTR_STA_BK0/1.OVERF points out the bank from which the OUT data should have originated. If the UPSTAn.ERRORFI bit is cleared and a new transaction is successful, the Pn_CTR_STA_BK0/1.OVERF bit will be cleared.
- An underflow can occur during an IN stage if the device tries to send a packet while the bank is full. Typically this occurs when a CPU is not fast enough. The packet data is not written to the bank and is lost. The pipe descriptor Pn_CTR_STA_BK0/1.UNDERF points out which bank the OUT data was destined to. If UPSTAn.UNDERFI is zero and a new successful transaction occurs, Pn_CTR_STA_BK0/1.UNDERF will be cleared.

17.6.3.15 *CRC error*

This error exists only for isochronous IN pipes. It sets the CRC Error Interrupt bit (CRCERRI), which triggers a PnINT interrupt if the CRC Error Interrupt Enable bit (UPCONn.CRCERRE) is one.

A CRC error can occur during the IN stage if the USBC detects a corrupted packet. The IN packet will remain stored in the bank and RXINI will be set.

The user can check the Pn_CTR_STA_BK0/1.CRCERR bit in the pipe descriptor to see which current bank has been affected.

17.6.3.16 Interrupts

There are two kinds of host interrupts: processing, i.e. their generation is part of the normal processing, and exception, i.e. errors not related to CPU exceptions.

- *Global interrupts*

The processing host global interrupts are:

- The Device Connection Interrupt (DCONNI)
- The Device Disconnection Interrupt (DDISCI)
- The USB Reset Sent Interrupt (RSTI)
- The Downstream Resume Sent Interrupt (RSMEDI)
- The Upstream Resume Received Interrupt (RXRSMI)
- The Host Start of Frame Interrupt (HSOFI)
- The Host Wakeup Interrupt (HWUPI)
- The Pipe n Interrupt (PnINT)

There is no exception host global interrupt.

- *Pipe interrupts*

The processing host pipe interrupts are:

- The Received IN Data Interrupt (RXINI)
- The Transmitted OUT Data Interrupt (TXOUTI)
- The Transmitted SETUP Interrupt (TXSTPI)
- The Number of Busy Banks (NBUSYBK) interrupt

The exception host pipe interrupts are:

- The Errorflow Interrupt (ERRORFI)
- The Pipe Error Interrupt (PERRI)
- The NAKed Interrupt (NAKEDI)
- The Received STALLed Interrupt (RXSTALLDI)
- The CRC Error Interrupt (CRCERRI)

17.7 User Interface

Table 17-9. USBC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0000	Device General Control Register	UDCON	Read/Write	0x00000100
0x0004	Device Global Interrupt Register	UDINT	Read-Only	0x00000000
0x0008	Device Global Interrupt Clear Register	UDINTCLR	Write-Only	0x00000000
0x000C	Device Global Interrupt Set Register	UDINTSET	Write-Only	0x00000000
0x0010	Device Global Interrupt Enable Register	UDINTE	Read-Only	0x00000000
0x0014	Device Global Interrupt Enable Clear Register	UDINTECLR	Write-Only	0x00000000
0x0018	Device Global Interrupt Enable Set Register	UDINTESET	Write-Only	0x00000000
0x001C	Endpoint Enable/Reset Register	UERST	Read/Write	0x00000000
0x0020	Device Frame Number Register	UDFNUM	Read-Only	0x00000000
0x0100 + n*4	Endpoint n Configuration Register	UECFGn	Read/Write	0x00000000
0x0130 + n*4	Endpoint n Status Register	UESTAn	Read-Only	0x00000100
0x0160 + n*4	Endpoint n Status Clear Register	UESTAnCLR	Write-Only	0x00000000
0x0190 + n*4	Endpoint n Status Set Register	UESTAnSET	Write-Only	0x00000000
0x01C0 + n*4	Endpoint n Control Register	UECONn	Read-Only	0x00000000
0x01F0 + n*4	Endpoint n Control Set Register	UECONnSET	Write-Only	0x00000000
0x0220 + n*4	Endpoint n Control Clear Register	UECONnCLR	Write-Only	0x00000000
0x0400	Host General Control Register	UHCON	Read/Write	0x00000000
0x0404	Host Global Interrupt Register	UHINT	Read-Only	0x00000000
0x0408	Host Global Interrupt Clear Register	UHINTCLR	Write-Only	0x00000000
0x040C	Host Global Interrupt Set Register	UHINTSET	Write-Only	0x00000000
0x0410	Host Global Interrupt Enable Register	UHINTE	Read-Only	0x00000000
0x0414	Host Global Interrupt Enable Clear Register	UHINTECLR	Write-Only	0x00000000
0x0418	Host Global Interrupt Enable Set Register	UHINTESET	Write-Only	0x00000000
0x041C	Pipe Enable/Reset Register	UPRST	Read/Write	0x00000000
0x0420	Host Frame Number Register	UHFNUM	Read/Write	0x00000000
0x0424	Host Start Of Frame Control Register	UHSOFC	Read/Write	0x00000000
0x0500 + n*4	Pipe n Configuration Register	UPCFGn	Read/Write	0x00000000
0x0530 + n*4	Pipe n Status Register	UPSTAn	Read-Only	0x00000000
0x0560 + n*4	Pipe n Status Clear Register	UPSTAnCLR	Write-Only	0x00000000
0x0590 + n*4	Pipe n Status Set Register	UPSTAnSET	Write-Only	0x00000000
0x05C0 + n*4	Pipe n Control Register	UPCONn	Read-Only	0x00000000
0x05F0 +n*4	Pipe n Control Set Register	UPCONnSET	Write-Only	0x00000000
0x0620 + n*4	Pipe n Control Clear Register	UPCONnCLR	Write-Only	0x00000000
0x0650 + n*4	Pipe n IN Request Register	UPINRQn	Read/Write	0x00000001
0x0800	General Control Register	USBCON	Read/Write	0x03004000

Table 17-9. USBC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0804	General Status Register	USBSTA	Read-Only	0x00000000
0x0808	General Status Clear Register	USBSTACLR	Write-Only	0x00000000
0x080C	General Status Set Register	USBSTASET	Write-Only	0x00000000
0x0818	IP Version Register	UVERS	Read-Only	-(1)
0x081C	IP Features Register	UFEATURES	Read-Only	-(1)
0x0820	IP PB Address Size Register	UADDRSIZE	Read-Only	-(1)
0x0824	IP Name Register 1	UNAME1	Read-Only	-(1)
0x0828	IP Name Register 2	UNAME2	Read-Only	-(1)
0x082C	USB Finite State Machine Status Register	USBFSM	Read-Only	0x00000009
0x0830	USB Descriptor address	UDESC	Read/Write	0x00000000

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

17.7.1 USB General Registers

17.7.1.1 General Control Register

Name: USBCON

Access Type: Read/Write

Offset: 0x0800

Reset Value: 0x03004000

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

- **UIMOD: USBC Mode**
  0: The module is in USB host mode.
  1: The module is in USB device mode.
  This bit can be written to even if USBE is zero or FRZCLK is one. Disabling the USBC (by writing a zero to the USBE bit) does not reset this bit.
- **USBE: USBC Enable**
  Writing a zero to this bit will disable the USBC, USB transceiver, and USB clock inputs. This will over-ride FRZCLK settings but not affect the value. Unless explicitly stated, all registers will become reset and read-only.
  Writing a one to this bit will enable the USBC.
  0: The USBC is disabled.
  1: The USBC is enabled.
  This bit can be written to even if FRZCLK is one.
- **FRZCLK: Freeze USB Clock**
  Writing a zero to this bit will enable USB clock inputs.
  Writing a one to this bit will disable USB clock inputs. The resume detection will remain active. Unless explicitly stated, all registers will become read-only.
  0: The clock inputs are enabled.
  1: The clock inputs are disabled.
  This bit can be written to even if USBE is zero.

17.7.1.2 General Status Register

Register Name: USBSTA

Access Type: Read-Only

Offset: 0x0804

Reset Value: 0x00010000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	SUSPEND
15	14	13	12	11	10	9	8
-	CLKUSABLE	SPEED		-	-	VBUSRQ	-
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	

- **SUSPEND: Suspend usb transceiver state**
  This bit is cleared when the usb transceiver is switched off.
  This bit is cleared when the usb transceiver is switched on.
- **CLKUSABLE: Generic Clock Usable**
  This bit is cleared when the USB generic clock is not usable.
  This bit is set when the USB generic clock (that should be 48MHz) is usable.
- **SPEED: Speed Status**
  This field is set according to the controller speed mode.

SPEED	Speed Status
00	full-speed mode
01	Reserved
10	low-speed mode
11	Reserved

- **VBUSRQ: VBUS Request**
  0: USBC is notified that the VBUS on the usb line is not powered.
  1: USBC is notified that the VBUS on the usb line is powered.
  This bit is cleared when USBSTACLR.VBUSRQC is written to one.
  This bit is set when USBSTASET.VBUSRQS is written to one.
  This bit should only be used in host mode.

17.7.1.3 General Status Clear Register

Register Name: USBSTACLR

Access Type: Write-Only

Offset: 0x0808

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in USBSTA.
These bits always read as zero.

17.7.1.4 General Status Set Register

Register Name: USBSTASET

Access Type: Write-Only

Offset: 0x080C

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in USBSTA.
These bits always read as zero.

17.7.1.5 Version Register

Register Name: UVERS

Access Type: Read-Only

Offset: 0x0818

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

17.7.1.6 Features Register

Register Name: UFEATURES

Access Type: Read-Only

Offset: 0x081C

Reset Value: -

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

- **EPTNBRMAX: Maximal Number of pipes/endpoints**
  This field indicates the number of hardware-implemented pipes/endpoints:

17.7.1.7 Address Size Register

Register Name: UADDRSIZE

Access Type: Read-Only

Offset: 0x0820

Reset Value: -

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

- **UADDRSIZE: IP PB Address Size**
  This field indicates the size of the PB address space reserved for the USBC IP interface.

17.7.1.8 IP Name Register 1

Register Name: UNAME1

Access Type: Read-Only

Offset: 0x0824

Reset Value: -

31	30	29	28	27	26	25	24
UNAME1[31:24]							

23	22	21	20	19	18	17	16
UNAME1[23:16]							

15	14	13	12	11	10	9	8
UNAME1[15:8]							

7	6	5	4	3	2	1	0
UNAME1[7:0]							

- **UNAME1: IP Name Part One**
  This field indicates the first part of the ASCII-encoded name of the USBC IP.

17.7.1.9 IP Name Register 2

Register Name: UNAME2

Access Type: Read-Only

Offset: 0x0828

Reset Value:

31	30	29	28	27	26	25	24
UNAME2[31:24]							

23	22	21	20	19	18	17	16
UNAME2[23:16]							

15	14	13	12	11	10	9	8
UNAME2[15:8]							

7	6	5	4	3	2	1	0
UNAME2[7:0]							

- **UNAME2: IP Name Part Two**
  This field indicates the second part of the ASCII-encoded name of the USBC IP.

17.7.1.10 Finite State Machine Status Register

Register Name: USBFSM

Access Type: Read-Only

Offset: 0x082C

Reset Value: 0x00000009

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

- **DRDSTATE: Dual Role Device State**

  This field indicates the state of the USBC.

  For Device mode it should always read 9.

DRDSTATE	Description
0	a_idle state: this is the start state for A-devices (when the ID pin is 0)
1	a_wait_vrise: In this state, the A-device waits for the voltage on VBUS to rise above the A-device VBUS Valid threshold (4.4 V).
2	a_wait_bcon: In this state, the A-device waits for the B-device to signal a connection.
3	a_host: In this state, the A-device that operates in host mode is operational.
4	a_suspend: The A-device operating as a host is in the suspend mode.
5	a_peripheral: The A-device operates as a peripheral.
6	a_wait_vfall: In this state, the A-device waits for the voltage on VBUS to drop below the A-device Session Valid threshold (1.4 V).
7	a_vbus_err: In this state, the A-device waits for recovery of the over-current condition that caused it to enter this state.
8	a_wait_discharge: In this state, the A-device waits for the data usb line to discharge (100 us).
9	b_idle: this is the start state for B-device (when the ID pin is 1). The USBC controller operates in device mode.
10	b_peripheral: In this state, the B-device acts as the peripheral.
11	b_wait_begin_hnp: In this state, the B-device is in suspend mode and waits until 3 ms before initiating the HNP protocol if requested.

DRDSTATE	Description
12	b_wait_discharge: In this state, the B-device waits for the data usb line to discharge (100 us) before becoming Host.
13	b_wait_acon: In this state, the B-device waits for the A-device to signal a connect before becoming B-Host.
14	b_host: In this state, the B-device acts as the Host.
15	b_srp_init: In this state, the B-device attempts to start a session using the SRP protocol.

17.7.1.11 USB Descriptor Address

Register Name: UDESC

Access Type: Read-Write

Offset: 0x0830

Reset Value: -

31	30	29	28	27	26	25	24
UDESCA[31:24]							

23	22	21	20	19	18	17	16
UDESCA[23:16]							

15	14	13	12	11	10	9	8
UDESCA[15:8]							

7	6	5	4	3	2	1	0
UDESCA[7:0]							

- **UDESCA: USB Descriptor Address**
  This field contains the address of the USB descriptor. The three least significant bits are always zero.

17.7.2 USB Device Registers

17.7.2.1 Device General Control Register

Register Name: UDCON

Access Type: Read/Write

Offset: 0x0000

Reset Value: 0x00000100

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	GNAK	-
15	14	13	12	11	10	9	8
-	-	-	LS	-	-	RMWKUP	DETACH
7	6	5	4	3	2	1	0
ADDEN	UADD						

- **GNAK: Global NAK**
  0: Normal mode.
  1: A NAK handshake is answered for each USB transaction regardless of the current endpoint memory bank status.
- **LS: low-speed mode force**
  0: The full-speed mode is active.
  1: The low-speed mode is active.
  This bit can be written to even if USBE is zero or FRZCLK is one. Disabling the USBC (by writing a zero to the USBE bit) does not reset this bit.
- **RMWKUP: Remote wakeup**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will send an upstream resume to the host for a remote wakeup.
  This bit is cleared when the USBC receives a USB reset or once the upstream resume has been sent.
- **DETACH: Detach**
  Writing a zero to this bit will reconnect the device.
  Writing a one to this bit will physically detach the device (disconnect internal pull-up resistor from DP and DM).
- **ADDEN: Address Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will activate the UADD field (USB address).
  This bit is cleared when a USB reset is received.
- **UADD: USB Address**
  This field contains the device address.
  This field is cleared when a USB reset is received.

17.7.2.2 Device Global Interrupt Register

Register Name: UDINT

Access Type: Read-Only

Offset: 0x0004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	EP8INT(1)	EP7INT(1)	EP6INT(1)	EP5INT(1)	EP4INT(1)
15	14	13	12	11	10	9	8
EP3INT(1)	EP2INT(1)	EP1INT(1)	EP0INT	-	-	-	-
7	6	5	4	3	2	1	0
-	UPRSM	EORSM	WAKEUP	EORST	SOF	-	SUSP

Note: 1. EPnINT bits are within the range from EP0INT to EP7INT.

- **EPnINT: Endpoint n Interrupt**
  This bit is cleared when the interrupt source is serviced.
  This bit is set when an interrupt is triggered by the endpoint n (UESTAn, UECONn). This triggers a USB interrupt if EPnINTE is one.
- **UPRSM: Upstream Resume Interrupt**
  This bit is cleared when the UDINTCLR.UPRSMC bit is written to one to acknowledge the interrupt (USB clock inputs must be enabled before).
  This bit is set when the USBC sends a resume signal called “Upstream Resume”. This triggers a USB interrupt if UPRSME is one.
- **EORSM: End of Resume Interrupt**
  This bit is cleared when the UDINTCLR.EORSMC bit is written to one to acknowledge the interrupt.
  This bit is set when the USBC detects a valid “End of Resume” signal initiated by the host. This triggers a USB interrupt if EORSME is one.
- **WAKEUP: Wakeup Interrupt**
  This bit is cleared when the UDINTCLR.WAKEUPC bit is written to one to acknowledge the interrupt (USB clock inputs must be enabled before).
  This bit is set when the USBC is reactivated by a filtered non-idle signal from the lines (not by an upstream resume). This triggers an interrupt if WAKEUPE is one.
  This interrupt is generated even if the clock is frozen by the FRZCLK bit.
- **EORST: End of Reset Interrupt**
  This bit is cleared when the UDINTCLR.EORSTC bit is written to one to acknowledge the interrupt.
  This bit is set when a USB “End of Reset” has been detected. This triggers a USB interrupt if EORSTE is one.
- **SOF: Start of Frame Interrupt**
  This bit is cleared when the UDINTCLR.SOFC bit is written to one to acknowledge the interrupt.
  This bit is set when either a USB “Start of Frame” PID (SOF) or a Low-speed keep-alive has been detected (every 1 ms). This triggers a USB interrupt if SOFE is one. The FNUM field is updated.

- **SUSP: Suspend Interrupt**
  This bit is cleared when the UDINTCLR.SUSPC bit is written to one to acknowledge the interrupt.
  This bit is set when a USB “Suspend” idle bus state has been detected for 3 frame periods (J state for 3 ms). This triggers a USB interrupt if SUSPE is one.

17.7.2.3 Device Global Interrupt Clear Register

Register Name: UDINTCLR

Access Type: Write-Only

Offset: 0x0008

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	UPRSMC	EORSMC	WAKEUPC	EORSTC	SOFC	-	SUSPC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UDINT.
These bits always read as zero.

17.7.2.4 Device Global Interrupt Set Register

Register Name: UDINTSET

Access Type: Write-Only

Offset: 0x000C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	UPRSMS	EORSMS	WAKEUPS	EORSTS	SOFS	-	SUSPS

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UDINT, which may be useful for test or debug purposes.
These bits always read as zero.

17.7.2.5 Device Global Interrupt Enable Register

Register Name: UDINTE

Access Type: Read-Only

Offset: 0x0010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	EP8INTE(1)	EP7INTE(1)	EP6INTE(1)	EP5INTE(1)	EP4INTE(1)

15	14	13	12	11	10	9	8
EP3INTE(1)	EP2INTE(1)	EP1INTE(1)	EP0INTE	-	-	-	-

7	6	5	4	3	2	1	0
-	UPRSME	EORSME	WAKEUPE	EORSTE	SOFE	-	SUSPE

Note: 1. EPnINTE bits are within the range from EP0INTE to EP7INTE.

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

A bit in this register is cleared when the corresponding bit in UDINTECLR is written to one.

A bit in this register is set when the corresponding bit in UDINTESET is written to one.

17.7.2.6 Device Global Interrupt Enable Clear Register

Register Name: UDINTECLR

Access Type: Write-Only

Offset: 0x0014

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	EP8INTEC[1]	EP7INTEC[1]	EP6INTEC[1]	EP5INTEC[1]	EP4INTEC[1]
15	14	13	12	11	10	9	8
EP3INTEC[1]	EP2INTEC[1]	EP1INTEC[1]	EP0INTEC	-	-	-	-
7	6	5	4	3	2	1	0
-	UPRSMEC	EORSMEC	WAKEUPEC	EORSTEC	SOFEC	-	SUSPEC

Note: 1. EPnINTEC bits are within the range from EP0INTEC to EP7INTEC.
Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UDINTE.
These bits always read as zero.

17.7.2.7 Device Global Interrupt Enable Set Register

Register Name: UDINTESET

Access Type: Write-Only

Offset: 0x0018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	EP8INTES(1)	EP7INTES(1)	EP6INTES(1)	EP5INTES(1)	EP4INTES(1)

15	14	13	12	11	10	9	8
EP3INTES(1)	EP2INTES(1)	EP1INTES(1)	EP0INTES	-	-	-	-

7	6	5	4	3	2	1	0
-	UPRSMES	EORSMES	WAKEUPES	EORSTES	SOFES	-	SUSPES

Note: 1. EPnINTES bits are within the range from EP0INTES to EP7INTES.
Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UDINTE.
These bits always read as zero.

17.7.2.8 Endpoint Enable/Reset Register

Register Name: UERST

Access Type: Read/Write

Offset: 0x001C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	EPEN8[1]
7	6	5	4	3	2	1	0
EPEN7[1]	EPEN6[1]	EPEN5[1]	EPEN4[1]	EPEN3[1]	EPEN2[1]	EPEN1[1]	EPEN0

- **EPENn: Endpoint n Enable**

Note: 1. EPENn bits are within the range from EPEN0 to EPEN7.

Writing a zero to this bit will disable the endpoint n (USB requests will be ignored), and resets the endpoints registers (UECFGn, UESTAn, UECONn), but not the endpoint configuration (EPBK, EPSIZE, EPDIR, EPTYPE).

Writing a one to this bit will enable the endpoint n.

0: The endpoint n is disabled.

1: The endpoint n is enabled.

17.7.2.9 Device Frame Number Register

Register Name: UDFNUM

Access Type: Read-Only

Offset: 0x0020

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
FNCERR	-	FNUM[10:5]					

7	6	5	4	3	2	1	0
FNUM[4:0]					-	-	-

- **FNCERR: Frame Number CRC Error**
  This bit is cleared upon receiving a USB reset.
  This bit is set when a corrupted frame number is received. This bit and the SOF interrupt bit are updated at the same time.
- **FNUM: Frame Number**
  This field is cleared upon receiving a USB reset.
  This field contains the 11-bit frame number information, as provided from the last SOF packet.
  FNUM is updated even if a corrupted SOF is received.

17.7.2.10 Endpoint n Configuration Register

Register Name: UECFGn, n in [0..7]

Access Type: Read/Write

Offset: 0x0100 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	REPNB			
15	14	13	12	11	10	9	8
-	-	-	EPTYPE		-	-	EPDIR
7	6	5	4	3	2	1	0
-	EPSIZE			-	EPBK	-	-

- **REPNB: Redirected endpoint number**
  This field is used to configure the redirected endpoint number.
  This field should be zero if the redirected endpoint feature is not used.
  This field should not be used for control endpoints.
  This field is cleared upon receiving a USB reset.
- **EPTYPE: Endpoint Type**
  This field selects the endpoint type:

EPTYPE		Endpoint Type
0	0	Control
0	1	Isochronous
1	0	Bulk
1	1	Interrupt

  This field is cleared upon receiving a USB reset.
- **EPDIR: Endpoint Direction**
  0: The endpoint direction is OUT.
  1: The endpoint direction is IN (nor for control endpoints).
  This bit is cleared upon receiving a USB reset.

- **EPSIZE: Endpoint Size**
  This field determines the size of each endpoint bank:

EPSIZE			Endpoint Size
0	0	0	8 bytes
0	0	1	16 bytes
0	1	0	32 bytes
0	1	1	64 bytes
1	0	0	128 bytes
1	0	1	256 bytes
1	1	0	512 bytes
1	1	1	1024 bytes

  This field is cleared upon receiving a USB reset (except for the endpoint 0).
- **EPBK: Endpoint Banks**
  This bit selects the number of banks for the endpoint:
  0: single-bank endpoint
  1: double-bank endpoint
  For control endpoints, a single-bank endpoint shall be selected.
  This field is cleared upon receiving a USB reset (except for the endpoint 0).

17.7.2.11 Endpoint n Status Register

Register Name: UESTAn, n in [0..7]

Access Type: Read-Only 0x0100

Offset: 0x0130 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	CTRLDIR	-

15	14	13	12	11	10	9	8
CURRBK		NBUSYBK		RAMACERI	-	DTSEQ	

7	6	5	4	3	2	1	0
-	STALLEDI/ CRCERRI	-	NAKINI	NAKOUTI	RXSTPI/ ERRORFI	RXOUTI	TXINI

- **CTRLDIR: Control Direction**
  Writing a zero or a one to this bit has no effect.
  This bit is cleared after a SETUP packet to indicate that the following packet is an OUT packet.
  This bit is set after a SETUP packet to indicate that the following packet is an IN packet.
- **CURRBK: Current Bank**
  This bit is set for non-control endpoints, indicating the current bank:

CURRBK		Current Bank
0	0	Bank0
0	1	Bank1
1	0	Reserved
1	1	Reserved

This field may be updated one clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

- **NBUSYBK: Number of Busy Banks**
  This field is set to indicate the number of busy banks:

NBUSYBK		Number of Busy Banks
0	0	0 (all banks free)
0	1	1
1	0	2
1	1	Reserved

For IN endpoints, this indicates the number of banks filled by the user and ready for IN transfers. When all banks are free an EPnINT interrupt will be triggered if NBUSYBKE is one.
For OUT endpoints, this indicates the number of banks filled by OUT transactions from the host. When all banks are busy an EPnINT interrupt will be triggered if NBUSYBKE is one.

- **RAMACERI: Ram Access Error Interrupt**
  This bit is cleared when the RAMACERIC bit is written to one, acknowledging the interrupt.
  This bit is set when a RAM access underflow error occurs during an IN data stage.
- **DTSEQ: Data Toggle Sequence**
  This field is set to indicate the PID of the current bank:

DTSEQ		Data Toggle Sequence
0	0	Data0
0	1	Data1
1	X	Reserved

For IN transfers, this indicates the data toggle sequence that will be used for the next packet to be sent.
For OUT transfers, this value indicates the data toggle sequence of the data received in the current bank.

- **STALLEDI: STALLed Interrupt**
  This bit is cleared when the STALLEDIC bit is written to one, acknowledging the interrupt.
  This bit is set when a STALL handshake has been sent and triggers an EPnINT interrupt if STALLEDE is one.
- **CRCERRI: CRC Error Interrupt**
  This bit is cleared when the CRCERRIC bit is written to one, acknowledging the interrupt.
  This bit is set when a CRC error has been detected in an isochronous OUT endpoint bank, and triggers an EPnINT interrupt if CRCERRE is one.
- **NAKINI: NAKed IN Interrupt**
  This bit is cleared when the NAKINIC bit is written to one, acknowledging the interrupt.
  This bit is set when a NAK handshake has been sent in response to an IN request from the host, and triggers an EPnINT interrupt if NAKINE is one.
- **NAKOUTI: NAKed OUT Interrupt**
  This bit is cleared when the NAKOUTIC bit is written to one, acknowledging the interrupt.
  This bit is set when a NAK handshake has been sent in response to an OUT request from the host, and triggers an EPnINT interrupt if NAKOUTE is one.
- **ERRORFI: Isochronous Error flow Interrupt**
  This bit is cleared when the ERRORFIC bit is written to one, acknowledging the interrupt.
  This bit is set, for isochronous IN/OUT endpoints, when an errorflow (underflow or overflow) error occurs, and triggers an EPnINT interrupt if ERRORFE is one.
  An underflow can occur during IN stage if the host attempts to read from an empty bank. A zero-length packet is then automatically sent by the USBC.
  An overflow can also occur during OUT stage if the host sends a packet while the bank is already full, resulting in the packet being lost. This is typically due to a CPU not being fast enough.
  This bit is inactive (cleared) for bulk and interrupt IN/OUT endpoints and it means RXSTPI for control endpoints.
- **RXSTPI: Received SETUP Interrupt**
  This bit is cleared when the RXSTPIC bit is written to one, acknowledging the interrupt and freeing the bank.
  This bit is set, for control endpoints, to signal that the current bank contains a new valid SETUP packet, and triggers an EPnINT interrupt if RXSTPE is one.
  This bit is inactive (cleared) for bulk and interrupt IN/OUT endpoints and it means UNDERFI for isochronous IN/OUT endpoints.
- **RXOUTI: Received OUT Data Interrupt**
  This bit is cleared when the RXOUTIC bit is written to one, acknowledging the interrupt. For control endpoints, it releases the bank. For other endpoint types, the user should clear the FIFOCON bit to free the bank. RXOUTI shall always be cleared before clearing FIFOCON to avoid missing an interrupt.

This bit is set, for control endpoints, when the current bank contains a bulk OUT packet (data or status stage). This triggers an EPnINT interrupt if RXOUTE is one.

This bit is set for isochronous, bulk and, interrupt OUT endpoints, at the same time as FIFOCON when the current bank is full. This triggers an EPnINT interrupt if RXOUTE is one.

This bit is inactive (cleared) for isochronous, bulk and interrupt IN endpoints.

- **TXINI: Transmitted IN Data Interrupt**

  This bit is cleared when the TXINIC bit is written to one, acknowledging the interrupt. For control endpoints, this will send the packet. For other endpoint types, the user should clear the FIFOCON to allow the USBC to send the data. TXINI shall always be cleared before clearing FIFOCON to avoid missing an interrupt.

  This bit is set for control endpoints, when the current bank is ready to accept a new IN packet. This triggers an EPnINT interrupt if TXINE is one.

  This bit is set for isochronous, bulk and interrupt IN endpoints, at the same time as FIFOCON when the current bank is free. This triggers an EPnINT interrupt if TXINE is one.

  This bit is inactive (cleared) for isochronous, bulk and interrupt OUT endpoints.

17.7.2.12 Endpoint n Status Clear Register

Register Name: UESTAnCLR, n in [0..7]

Access Type: Write-Only

Offset: 0x0160 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	RAMACERIC	-	-	-
7	6	5	4	3	2	1	0
-	STALLEDIC/ CRCERRIC	-	NAKINIC	NAKOUTIC	RXSTPIC/ ERRORFIC	RXOUTIC	TXINIC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UESTA.
These bits always read as zero.

17.7.2.13 Endpoint n Status Set Register

Register Name: UESTAnSET, n in [0..7]

Access Type: Write-Only

Offset: 0x0190 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	NBUSYBKS	RAMACERIS	-		-
7	6	5	4	3	2	1	0
-	STALLEDIS/ CRCERRIS	-	NAKINIS	NAKOUTIS	RXSTPIS/ ERRORFIS	RXOUTIS	TXINIS

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UESTA.
These bits always read as zero.

17.7.2.14 Endpoint n Control Register

Register Name: UECONn, n in [0..7]

Access Type: Read-Only

Offset: 0x01C0 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	BUSY1E	BUSY0E
23	22	21	20	19	18	17	16
-	-	-	-	STALLRQ	RSTDT	-	-
15	14	13	12	11	10	9	8
-	FIFOCON	KILLBK	NBUSYBKE	RAMACERE	-	-	NREPLY
7	6	5	4	3	2	1	0
-	STALLEDE/ CRCERRE	-	NAKINE	NAKOUTE	RXSTPE/ ERRORFE	RXOUTE	TXINE

- **BUSY0E: Busy Bank0 Enable**
  This bit is cleared when the BUSY0C bit is written to one.
  This bit is set when the BUSY0ES bit is written to one. This will set the bank 0 as “busy”. All transactions, except SETUP, destined to this bank will be rejected (i.e: NAK token will be answered).
- **BUSY1E: Busy Bank1 Enable**
  This bit is cleared when the BUSY1C bit is written to one.
  This bit is set when the BUSY1ES bit is written to one. This will set the bank 1 as “busy”. All transactions, except SETUP, destined to this bank will be rejected (i.e: NAK token will be answered).
- **STALLRQ: STALL Request**
  This bit is cleared when a new SETUP packet is received or when the STALLRQC bit is written to zero.
  This bit is set when the STALLRQS bit is written to one, requesting a STALL handshake to be sent to the host.
- **RSTDT: Reset Data Toggle**
  The data toggle sequence is cleared when the RSTDTS bit is written to one (i.e., Data0 data toggle sequence will be selected for the next sent (IN endpoints) or received (OUT endpoints) packet.
  This bit is always read as zero.
- **FIFOCON: FIFO Control**
  For control endpoints:
  The FIFOCON and RWALL bits are irrelevant. The software shall therefore never use them for these endpoints. When read, their value is always 0.
  For IN endpoints:
  This bit is cleared when the FIFOCONC bit is written to one, sending the FIFO data and switching to the next bank.
  This bit is set simultaneously to TXINI, when the current bank is free.
  For OUT endpoints:
  This bit is cleared when the FIFOCONC bit is written to one, freeing the current bank and switching to the next.
  This bit is set simultaneously to RXINI, when the current bank is full.

- **KILLBK: Kill IN Bank**
  This bit is cleared by hardware after the completion of the “kill packet procedure”.
  This bit is set when the KILLBKS bit is written to one, killing the last written bank.
  The user shall wait for this bit to be cleared before trying to process another IN packet.
  Caution: The bank is cleared when the “kill packet” procedure is completed by the USBC core:
  If the bank is really killed, the NBUSYBK field is decremented.
  If the bank sent instead of killed (IN transfer), the NBUSYBK field is decremented and the TXINI flag is set. This specific case can occur if an IN token comes while the user tries to kill the bank.
  Note: If two banks are ready to be sent, the above specific case will not occur, since the first bank is sent (IN transfer) while the last bank is killed.
- **NBUSYBKE: Number of Busy Banks Interrupt Enable**
  This bit is cleared when the NBUSYBKEC bit is written to zero, disabling the Number of Busy Banks interrupt (NBUSYBK).
  This bit is set when the NBUSYBKES bit is written to one, enabling the Number of Busy Banks interrupt (NBUSYBK).
- **RAMACERE: RAMACER Interrupt Enable**
  This bit is cleared when the RAMACEREC bit is written to one, disabling the RAMACER interrupt (RAMACERI).
  This bit is set when the RAMACERES bit is written to one, enabling the RAMACER interrupt (RAMACERI).
- **NREPLY: No Reply**
  This bit is cleared when the NREPLYC bit is written to one, disabling the “NO REPLY” feature, or upon receiving a SETUP packet.
  This bit is set when the NREPLYS bit is written to one, enabling the “NO_REPLY” feature. Any transaction to this endpoint will be ignored except SETUP.
- **STALLEDE: STALLed Interrupt Enable**
  This bit is cleared when the STALLEDEC bit is written to one, disabling the STALLed interrupt (STALLEDI).
  This bit is set when the STALLEDES bit is written to one, enabling the STALLed interrupt (STALLEDI).
- **CRCERRE: CRC Error Interrupt Enable**
  This bit is cleared when the CRCERREC bit is written to one, disabling the CRC Error interrupt (CRCERRI).
  This bit is set when the CRCERRES bit is written to one, enabling the CRC Error interrupt (CRCERRI).
- **NAKINE: NAKed IN Interrupt Enable**
  This bit is cleared when the NAKINEC bit is written to one, disabling the NAKed IN interrupt (NAKINI).
  This bit is set when the NAKINES bit is written to one, enabling the NAKed IN interrupt (NAKINI).
- **NAKOUTE: NAKed OUT Interrupt Enable**
  This bit is cleared when the NAKOUTEC bit is written to one, disabling the NAKed OUT interrupt (NAKOUTI).
  This bit is set when the NAKOUTES bit is written to one, enabling the NAKed OUT interrupt (NAKOUTI).
- **RXSTPE: Received SETUP Interrupt Enable**
  This bit is cleared when the RXSTPEC bit is written to one, disabling the Received SETUP interrupt (RXSTPI).
  This bit is set when the RXSTPES bit is written to one, enabling the Received SETUP interrupt (RXSTPI).
- **ERRORFE: Errorflow Interrupt Enable**
  This bit is cleared when the ERRORFEC bit is written to one, disabling the Underflow interrupt (ERRORFI).
  This bit is set when the ERRORFES bit is written to one, enabling the Underflow interrupt (ERRORFI).
- **RXOUTE: Received OUT Data Interrupt Enable**
  This bit is cleared when the RXOUTEC bit is written to one, disabling the Received OUT Data interrupt (RXOUT).
  This bit is set when the RXOUTES bit is written to one, enabling the Received OUT Data interrupt (RXOUT).
- **TXINE: Transmitted IN Data Interrupt Enable**
  This bit is cleared when the TXINEC bit is written to one, disabling the Transmitted IN Data interrupt (TXINI).
  This bit is set when the TXINES bit is written to one, enabling the Transmitted IN Data interrupt (TXINI).

17.7.2.15 Endpoint n Control Clear Register

Register Name: UECONnCLR, n in [0..7]

Access Type: Write-Only

Offset: 0x0220 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	BUSY1EC	BUSY0EC
23	22	21	20	19	18	17	16
-	-	-	-	STALLRQC	-	-	-
15	14	13	12	11	10	9	8
-	FIFOCONC	-	NBUSYBKEC	RAMACEREC	-	-	NREPLYC
7	6	5	4	3	2	1	0
-	STALLEDEC /CRCERREC	-	NAKINEC	NAKOUTEC	RXSTPEC/ ERRORFEC	RXOUTEC	TXINEC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UECONn.
These bits always read as zero.

17.7.2.16 Endpoint n Control Set Register

Register Name: UECONnSET, n in [0..7]

Access Type: Write-Only

Offset: 0x01F0 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	BUSY1ES	BUSY0ES
23	22	21	20	19	18	17	16
-	-	-	-	STALLRQS	RSTDTS	-	-
15	14	13	12	11	10	9	8
-	-	KILLBKS	NBUSYBKES	RAMACERES	-	-	NREPLYS
7	6	5	4	3	2	1	0
-	STALLEDES/ CRCERRES	-	NAKINES	NAKOUTES	RXSTPES/ ERRORFES	RXOUTES	TXINES

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UECONn.
These bits always read as zero.

17.7.3 USB Host Registers

17.7.3.1 Host General Control Register

Register Name: UHCON

Access Type: Read/Write

Offset: 0x0400

Reset Value: 0x00000000

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

-
-
- **RESUME: Send USB Resume**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will generate a USB Resume on the USB bus. This bit should only be done when the start of frame generation is enabled (SOFE bit is one).
  This bit is cleared when the USB Resume has been sent or when a USB reset is requested.
- **RESET: Send USB Reset**
  Writing a zero to this bit might be useful when a device disconnection is detected (UHINT.DDISCI is one) while a USB Reset is being sent.
  Writing a one to this bit will generate a USB Reset on the USB bus.
  This bit is cleared when the USB Reset has been sent.
- **SOFE: Start of Frame Generation Enable**
  Writing a zero to this bit will disable the SOF generation and to leave the USB bus in idle state.
  Writing a one to this bit will generate SOF on the USB bus in full speed mode and keep it alive in low speed mode.
  This bit is set when a USB reset is requested or an upstream resume interrupt is detected (UHINT.RXRSMI).

17.7.3.2 Host Global Interrupt Register

Register Name: UHINT

Access Type: Read-Only

Offset: 0x0404

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	P8INT[1]
15	14	13	12	11	10	9	8
P7INT[1]	P6INT[1]	P5INT[1]	P4INT[1]	P3INT[1]	P2INT[1]	P1INT[1]	P0INT
7	6	5	4	3	2	1	0
-	HWUPI	HSOFI	RXRSMI	RSMEDI	RSTI	DDISCI	DCONNI

Note: 1. PnINT bits are within the range from P0INT to P7INT.

- **PnINT: Pipe n Interrupt**
  This bit is cleared when the interrupt source is served.
  This bit is set when an interrupt is triggered by the endpoint n (UPSTAn). This triggers a USB interrupt if the corresponding pipe interrupt enable bit is one (UHINTE register).
- **HWUPI: Host Wakeup Interrupt**
  This bit is cleared when the HWUPIC bit is written to one.
  This bit is set when:
  - the host controller is in the suspend mode (SOFE is zero) and an upstream resume from the peripheral is detected.
  - the host controller is in the suspend mode (SOFE is zero) and a peripheral disconnection is detected.
  - the host controller is in the operationnal state (VBUSRQ is one) and a device connection is detected.
  This interrupt is generated even if the clock is frozen by the FRZCLK bit.
- **HSOFI: Host Start of Frame Interrupt**
  This bit is cleared when the HSOFIC bit is written to one.
  This bit is set when a SOF is issued by the Host controller. This triggers a USB interrupt when HSOFE is one. When using the host controller in low speed mode, this bit is also set when a keep-alive is sent.
- **RXRSMI: Upstream Resume Received Interrupt**
  This bit is set when an Upstream Resume has been received from the Device.
  This bit is cleared when the RXRSMIC is written to one.
- **RSMEDI: Downstream Resume Sent Interrupt**
  This bit is cleared when the RSMEDIC bit is written to one.
  This bit set when a Downstream Resume has been sent to the Device.
- **RSTI: USB Reset Sent Interrupt**
  This bit is cleared when the RSTIC bit is written to one.
  This bit is set when a USB Reset has been sent to the device.
- **DDISCI: Device Disconnection Interrupt**
  This bit is cleared when the DDISCIC bit is written to one.

This bit is set when the device has been removed from the USB bus.

- **DCONNI: Device Connection Interrupt**
  This bit is cleared when the DCONNIC bit is written to one.
  This bit is set when a new device has been connected to the USB bus.

17.7.3.3 Host Global Interrupt Clear Register

Register Name: UHINTCLR

Access Type: Write-Only

Offset: 0x0408

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	HWUPIC	HSOFIC	RXRSMIC	RSMEDIC	RSTIC	DDISCIC	DCONNIC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UHINT.
These bits always read as zero.

17.7.3.4 Host Global Interrupt Set Register

Register Name: UHINTSET

Access Type: Write-Only

Offset: 0x040C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-		-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	HWUPIS	HSOFIS	RXRSMIS	RSMEDIS	RSTIS	DDISCIS	DCONNIS

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UHINT.
These bits always read as zero.

17.7.3.5 Host Global Interrupt Enable Register

Register Name: UHINTE

Access Type: Read-Only

Offset: 0x0410

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	P8INTE[1]

15	14	13	12	11	10	9	8
P7INTE[1]	P6INTE[1]	P5INTE[1]	P4INTE[1]	P3INTE[1]	P2INTE[1]	P1INTE[1]	P0INTE

7	6	5	4	3	2	1	0
-	HWUPIE	HSOFIE	RXRSMIE	RSMEDIE	RSTIE	DDISCIE	DCONNIE

Note: 1. PnINTE bits are within the range from P0INTE to P7INTE.

- **PnINTE: Pipe n Interrupt Enable**
  This bit is cleared when the PnINTEC bit is written to one. This will disable the Pipe n Interrupt (PnINT).
  This bit is set when the PnINTES bit is written to one. This will enable the Pipe n Interrupt (PnINT).
- **HWUPIE: Host Wakeup Interrupt Enable**
  This bit is cleared when the HWUPIEC bit is written to one. This will disable the Host Wakeup Interrupt (HWUPI).
  This bit is set when the HWUPIES bit is written to one. This will enable the Host Wakeup Interrupt (HWUPI).
- **HSOFIE: Host Start of Frame Interrupt Enable**
  This bit is cleared when the HSOFIEC bit is written to one. This will disable the Host Start of Frame interrupt (HSOFI).
  This bit is set when the HSOFIES bit is written to one. This will enable the Host Start of Frame interrupt (HSOFI).
- **RXRSMIE: Upstream Resume Received Interrupt Enable**
  This bit is cleared when the RXRSMIEC bit is written to one. This will disable the Downstream Resume interrupt (RXRSMI).
  This bit is set when the RXRSMIES bit is written to one. This will enable the Upstream Resume Received interrupt (RXRSMI).
- **RSMEDIE: Downstream Resume Sent Interrupt Enable**
  This bit is cleared when the RSMEDIEC bit is written to one. This will disable the Downstream Resume interrupt (RSMEDI).
  This bit is set when the RSMEDIES bit is written to one. This will enable the Downstream Resume interrupt (RSMEDI).
- **RSTIE: USB Reset Sent Interrupt Enable**
  This bit is cleared when the RSTIEC bit is written to one. This will disable the USB Reset Sent interrupt (RSTI).
  This bit is set when the RSTIES bit is written to one. This will enable the USB Reset Sent interrupt (RSTI).
- **DDISCIE: Device Disconnection Interrupt Enable**
  This bit is cleared when the DDISCIEC bit is written to one. This will disable the Device Disconnection interrupt (DDISCI).
  This bit is set when the DDISCIES bit is written to one. This will enable the Device Disconnection interrupt (DDISCI).
- **DCONNIE: Device Connection Interrupt Enable**
  This bit is cleared when the DCONNIEC bit is written to one. This will disable the Device Connection interrupt (DCONNI).
  This bit is set when the DCONNIES bit is written to one. This will enable the Device Connection interrupt (DCONNI).

17.7.3.6 Host Global Interrupt Enable Clear Register

Register Name: UHINTECLR

Access Type: Write-Only

Offset: 0x0414

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	P8INTEC(1)

15	14	13	12	11	10	9	8
P7INTEC(1)	P6INTEC(1)	P5INTEC(1)	P4INTEC(1)	P3INTEC(1)	P2INTEC(1)	P1INTEC(1)	P0INTEC

7	6	5	4	3	2	1	0
-	HWUPIEC	HSOFIEC	RXRSMIEC	RSMEDIEC	RSTIEC	DDISCIEC	DCONNIEC

Note: 1. PnINTEC bits are within the range from P0INTEC to P7INTEC.

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will clear the corresponding bit in UHINTE.

These bits always read as zero.

17.7.3.7 Host Global Interrupt Enable Set Register

Register Name: UHINTESET

Access Type: Write-Only

Offset: 0x0418

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	P8INTES[1]

15	14	13	12	11	10	9	8
P7INTES[1]	P6INTES[1]	P5INTES[1]	P4INTES[1]	P3INTES[1]	P2INTES[1]	P1INTES[1]	P0INTES

7	6	5	4	3	2	1	0
-	HWUPIES	HSOFIES	RXRSMIES	RSMEDIES	RSTIES	DDISCIES	DCONNIES

Note: 1. PnINTES bits are within the range from P0INTES to P7INTES.
Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UHINT.
These bits always read as zero.

17.7.3.8 Pipe Enable/Reset Register

Register Name: UPRST

Access Type: Read/Write

Offset: 0x0041C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	PEN8[1]
7	6	5	4	3	2	1	0
PEN7[1]	PEN6[1]	PEN5[1]	PEN4[1]	PEN3[1]	PEN2[1]	PEN1[1]	PEN0

Note: 1. PENn bits are within the range from PEN0 to PEN7.

- **PENn: Pipe n Enable**

  Writing a zero to this bit will disable the pipe n, forcing the pipe to an inactive state and resetting the pipe registers (UPCFGn, UPSTAn, and UPCONn).

  Writing a one to this bit will enable the pipe n.

17.7.3.9 Host Frame Number Register

Register Name: UHFNUM

Access Type: Read/Write

Offset: 0x0420

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
FLENHIGH							

15	14	13	12	11	10	9	8
-	-	FNUM[10:5]					

7	6	5	4	3	2	1	0
FNUM[4:0]					-	-	-

- **FLENHIGH: Frame Length**
  This field contains the 8 high-order bits of the 14-bits internal frame counter (frame counter at 12 MHz, counter length is 12000 to ensure a SOF generation every 1 ms).
- **FNUM: Frame Number**
  This field contains the current SOF number.
  This field can be written by software to initialize a new frame number value. In this case, at the next SOF, the FNUM field takes its new value

17.7.3.10 Host Start Of Frame Control Register

Register Name: UHSOFC

Access Type: Read/Write

Offset: 0x0424

Reset Value: 0x00000000yes

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	FLENCE

15	14	13	12	11	10	9	8
-	-	FLENC[13:0]					
FLENC[15:0]							

7	6	5	4	3	2	1	0
FLENC[7:0]							

During a very short period just before transmitting a start of frame, this register is locked. Thus, after writing, this is recommended to check the register value, and write this register again if necessary.

This register is cleared upon sending a usb reset.

- **FLENCE: Frame Length Control Enable**

  0:At the beginning of a frame, the internal frame length down-counter is loaded to 11999 to ensure a 1 ms frame rate at 12MHz.

  0: In full speed mode, the internal frame length down-counter is loaded to 59999 to ensure a 1 ms frame rate at 60MHz.

  0: In high speed mode, the internal frame length down-counter is loaded to 74999 to ensure a 125 us micro-frame rate at 60MHz.

  1: At the begining of a frame, the internal frame length down-counter is loaded to the FLENC value.

- **FLENC: Frame Length Control**

  Write this field to configure the frame length

17.7.3.11 Pipe n Configuration Register

Register Name: UPCFGn, n in [0..7]

Access Type: Read/Write

Offset: 0x0500 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
BINTERVAL							
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	PTYPE		-	-	PTOKEN	
7	6	5	4	3	2	1	0
-	PSIZE			-	PBK	-	-

- **BINTERVAL: bInterval parameter**
  This field corresponds to the bus access period of the pipe.
  For Interrupt pipe, this field corresponds to the desired period from 1 ms to 255 ms.
  For isochronous pipe, this field corresponds to the desired period calculated as this: $2^{(BInterval)} * 1$ ms.
  For bulk or control pipe, this field corresponds to the desired period from 1 ms to 255 ms.
  This field is cleared upon sending a USB reset.
- **PTYPE: Pipe Type**
  This field contains the pipe type.
  This field is cleared upon sending a USB reset.

PTYPE		Pipe Type
0	0	Control
0	1	Isochronous
1	0	Bulk
1	1	Interrupt

- **PTOKEN: Pipe Token**
  This field contains the endpoint token.

PTOKEN	Endpoint Direction
00	SETUP
01	IN
10	OUT
11	reserved

- **PSIZE: Pipe Size**

  This field contains the size of each pipe bank.

  This field is cleared upon sending a USB reset.

PSIZE			Endpoint Size
0	0	0	8 bytes
0	0	1	16 bytes
0	1	0	32 bytes
0	1	1	64 bytes
1	0	0	128 bytes
1	0	1	256 bytes
1	1	0	512 bytes
1	1	1	1024 bytes

- **PBK: Pipe Banks**

  This bit selects the number of banks for the pipe.

  0: single-bank pipe

  1: double bank pipe

  For control endpoints, a single-bank pipe should be selected.

  This field is cleared upon sending a USB reset.

17.7.3.12 Pipe n Status Register

Register Name: UPSTAn, n in [0..7]

Access Type: Read-Only

Offset: 0x0530 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	CURRBK	NBUSYBK		-	RAMACERI	DTSEQ	

7	6	5	4	3	2	1	0
-	RXSTALLDI/ CRCERRI	ERRORFI	NAKEDI	PERRI	TXSTPI	TXOUTI	RXINI

- **CURRBK: Current Bank**

  For non-control pipe, this field indicates the number of the current bank.

  This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll this field for an interrupt.

CURRBK		Current Bank
0	0	Bank0
0	1	Bank1

- **NBUSYBK: Number of Busy Banks**

  This field indicates the number of busy bank.

  For OUT pipe, this field indicates the number of busy bank(s), filled by the user, ready for OUT transfer. When all banks are busy, this triggers an PnINT interrupt if UPCONn.NBUSYBKE is one.

  For IN pipe, this field indicates the number of busy bank(s) filled by IN transaction from the Device. When all banks are free, this triggers an PnINT interrupt if UPCONn.NBUSYBKE is one.

NBUSYBK		Number of busy bank
0	0	All banks are free.
0	1	1 busy bank
1	0	2 busy banks
1	1	reserved

- **RAMACERI: Ram Access Error Interrupt**

  This bit is cleared when the RAMACERIC bit is written to one.

  This bit is set when a RAM access underflow error occurs during IN data stage.

- **DTSEQ: Data Toggle Sequence**
  This field indicates the data PID of the current bank.
  For OUT pipes, this field indicates the data toggle of the next packet that will be sent.
  For IN pipes, this field indicates the data toggle of the received packet stored in the current bank.

DTSEQ		Data toggle sequence
0	0	Data0
0	1	Data1
1	0	reserved
1	1	reserved

- **RXSTALLDI: Received STALLed Interrupt**
  This bit is cleared when the RXSTALLDIC bit is written to one.
  This bit is set, for all endpoints (except isochronous), when a STALL handshake has been received on the current bank of the pipe. The pipe is automatically frozen. This triggers an interrupt if the RXSTALLE bit is one.
- **CRCERRI: CRC Error Interrupt**
  This bit is cleared when the CRCERRIC bit is written to one.
  This bit is set, for isochronous endpoint, when a CRC error occurs on the current bank of the pipe. This triggers an interrupt if the TXSTPE bit is one.
- **ERRORFI: Errorflow Interrupt**
  This bit is cleared when the ERRORFIC bit is written to one.
  This bit is set:
  - for isochronous and interrupt IN/OUT pipes, when an error flow occurs. This triggers an interrupt if the ERRORFIE bit is one.
  - for isochronous or interrupt OUT pipes, when a transaction underflow occurs in the current pipe. i.e, the pipe can't send the OUT data packet in time because the current bank is not ready.
  - for isochronous or interrupt IN pipes, when a transaction flow error occurs in the current pipe. i.e, the current bank of the pipe is not free when a new IN USB packet is received. This packet is not stored in the bank. For interrupt pipes, the overflowed packet is ACKed to respect the USB standard.
- **NAKEDI: NAKed Interrupt**
  This bit is cleared when the NAKEDIC bit is written to one.
  This bit is set when a NAK has been received on the current bank of the pipe. This triggers an interrupt if the NAKEDE bit is one.
- **PERRI: Pipe Error Interrupt**
  This bit is cleared when the PERRIC bit is written to one.
  This bit is set when an error occurs on the current bank of the pipe. This triggers an interrupt if the PERRE bit is set. Refers to the PERSTA structure of the pipe descriptor (Figure 17-8) to determine the source of the error.
- **TXSTPI: Transmitted SETUP Interrupt**
  This bit is cleared when the TXSTPIC bit is written to one.
  This bit is set, for Control endpoints, when the current SETUP bank is free and can be filled. This triggers an interrupt if the TXSTPE bit is one.
- **TXOUTI: Transmitted OUT Data Interrupt**
  This bit is cleared when the TXOUTIC bit is written to one.
  This bit is set when the current OUT bank is free and can be filled. This triggers an interrupt if the TXOUTE bit is one.
- **RXINI: Received IN Data Interrupt**
  This bit is cleared when the RXINIC bit is written to one.
  This bit is set when a new USB message is stored in the current bank of the pipe. This triggers an interrupt if the RXINE bit is one.

17.7.3.13 Pipe n Status Clear Register

Register Name: UPSTAnCLR, n in [0..7]

Access Type: Write-Only

Offset: 0x0560 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	RAMACERIC	-	-
7	6	5	4	3	2	1	0
-	RXSTALLDIC / CRCERRIC	ERRORFIC	NAKEDIC	PERRIC	TXSTPIC	TXOUTIC	RXINIC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UPSTAn.
These bits always read as zero.

17.7.3.14 Pipe n Status Set Register

Register Name: UPSTAnSET, n in [0..7]

Access Type: Write-Only

Offset: 0x0590 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	RAMACERIS	-	-
7	6	5	4	3	2	1	0
-	RXSTALLDIS/ CRCERRIS	ERRORFIC	NAKEDIS	PERRIS	TXSTPIS	TXOUTIS	RXINIS

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UPSTAn.
These bits always read as zero.

17.7.3.15 Pipe n Control Register

Register Name: UPCONn, n in [0..7]

Access Type: Read-Only

Offset: 0x05C0 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	INITBK	INITDTGL	PFREEZE	-
15	14	13	12	11	10	9	8
-	FIFOCON	-	NBUSYBKE	-	RAMACERE	-	-
7	6	5	4	3	2	1	0
-	RXSTALLDE /CRCERRE	ERRORFIE	NAKEDE	PERRE	TXSTPE	TXOUTE	RXINE

- **INITBK: Bank Initialization**
  This bit is always read as zero.
  If the user writes a one to the INITBKC bit, this will set the current bank to Bank0 value for the current pipe.
  If the user writes a one to the INITBKS bit, this will set the current bank to Bank1 value for the current pipe.
  This may be useful to restore a pipe to manage alternate pipes on the same physical pipe.
- **INITTGL: Data Toggle Initialization**
  This bit is always read as zero.
  If the user writes a one to the INITTGLC bit, this will set the Data toggle to Data0 value for the current pipe.
  If the user writes a one to the INITTGLS bit, this will set the Data toggle to Data1 value for the current pipe.
  This may be useful to restore a pipe to manage alternate pipes on the same physical pipe.
- **PFREEZE: Pipe Freeze**
  This bit is cleared when the PFREEZEC bit is written to one. This will enable the pipe request generation.
  This bit is set when the PFREEZES bit is written to one or when the pipe is not configured or when a STALL handshake has been received on this pipe, or when INRQ In requests have been processed, or after a pipe Enable (UPRST.PEN rising). This will freeze the pipe requests generation.
  If the PFREEZES bit is written to one while a transaction is on going on the USB bus, the transaction will be properly completed and then the PFREEZE bit will be set. UPSTAn register should be checked to know this last transaction status.
- **FIFOCON: FIFO Control**
  For OUT and SETUP pipes:
  This bit is cleared when the FIFOCONC bit is written to one. This will send the FIFO data and switch the bank.
  This bit is set when the current bank is free, at the same time than TXOUTI or TXSTPI.
  For IN pipes:
  This bit is cleared when the FIFOCONC bit is written to one. This will free the current bank and switch to the next bank.
  This bit is set when a new IN message is stored in the current bank, at the same time than RXINI.
- **NBUSYBKE: Number of Busy Banks Interrupt Enable**
  This bit is cleared when the NBUSYBKEC bit is written to one. This will disable the Transmitted IN Data interrupt (NBUSYBKE).

This bit is set when the NBUSYBKES bit is written to one.This will enable the Transmitted IN Data interrupt (NBUSYBKE).

- **RAMACERE: Ram Access Error Interrupt Enable**
  This bit is cleared when the NBUSYBKEC bit is written to one. This will disable the Transmitted IN Data interrupt (NBUSYBKE).
  This bit is set when the NBUSYBKES bit is written to one.This will enable the Transmitted IN Data interrupt (NBUSYBKE).
- **RXSTALLDE: Received STALLed Interrupt Enable**
  This bit is cleared when the RXSTALLDEC bit is written to one. This will disable the Transmitted IN Data interrupt (RXSTALLDE).
  This bit is set when the RXSTALLDES bit is written to one. This will enable the Transmitted IN Data interrupt (RXSTALLDE).
- **CRCERRE: CRC Error Interrupt Enable**
  This bit is cleared when the CRCERREC bit is written to one. This will disable the Transmitted IN Data interrupt (CRCERRE).
  This bit is set when the CRCERRES bit is written to one. This will enable the Transmitted IN Data interrupt (CRCERRE).
- **ERRORFIE: Errorflow Interrupt Enable**
  This bit is cleared when the ERRORFIEC bit is written to one. This will disable the Transmitted IN Data interrupt (OVERFIE).
  This bit is set when the ERRORFIES bit is written to one. This will enable the Transmitted IN Data interrupt (OVERFIE).
- **NAKEDE: NAKed Interrupt Enable**
  This bit is cleared when the NAKEDEC bit is written to one. This will disable the Transmitted IN Data interrupt (NAKEDE).
  This bit is set when the NAKEDES bit is written to one. This will enable the Transmitted IN Data interrupt (NAKEDE).
- **PERRE: Pipe Error Interrupt Enable**
  This bit is cleared when the PERREC bit is written to one. This will disable the Transmitted IN Data interrupt (PERRE).
  This bit is set when the PERRES bit is written to one. This will enable the Transmitted IN Data interrupt (PERRE).
- **TXSTPE: Transmitted SETUP Interrupt Enable**
  This bit is cleared when the TXSTPEC bit is written to one. This will disable the Transmitted IN Data interrupt (TXSTPE).
  This bit is set when the TXSTPES bit is written to one. This will enable the Transmitted IN Data interrupt (TXSTPE).
- **TXOUTE: Transmitted OUT Data Interrupt Enable**
  This bit is cleared when the TXOUTEC bit is written to one. This will disable the Transmitted IN Data interrupt (TXOUTE).
  This bit is set when the TXOUTES bit is written to one. This will enable the Transmitted IN Data interrupt (TXOUTE).
- **RXINE: Received IN Data Interrupt Enable**
  This bit is cleared when the RXINEC bit is written to one. This will disable the Transmitted IN Data interrupt (RXINE).
  This bit is set when the RXINES bit is written to one. This will enable the Transmitted IN Data interrupt (RXINE).

17.7.3.16 Pipe n Control Set Register

Register Name: UPCONnSET, n in [0..7]

Access Type: Write-Only

Offset: 0x05F0 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	INITBKS	INITDTGLS	PFREEZES	-
15	14	13	12	11	10	9	8
-	-	-	NBUSYBKES	-	-	-	-
7	6	5	4	3	2	1	0
-	RXSTALLDES/ CRCERRES	ERRORFIES	NAKEDES	PERRES	TXSTPES	TXOUTES	RXINES

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in UPCONn.
These bits always read as zero.

17.7.3.17 Pipe n Control Clear Register

Register Name: UPCONnCLR, n in [0..7]

Access Type: Write-Only

Offset: 0x0620 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	INITBKC	INITDTGLC	PFREEZEC	-
15	14	13	12	11	10	9	8
-	FIFOCONC	-	NBUSYBKEC	-	-	-	-
7	6	5	4	3	2	1	0
-	RXSTALLDEC /CRCERREC	ERRORFIEC	NAKEDEC	PERREC	TXSTPEC	TXOUTEC	RXINEC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in UPCONn.
These bits always read as zero.

17.7.3.18 Pipe n IN Request Register

Register Name: UPINRQn, n in [0..7]

Access Type: Read/Write

Offset: 0x0650 + (n * 0x04)

Reset Value: 0x00000001

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

- **INMODE: IN Request Mode**
  Writing a zero to this bit will perform a pre-defined number of IN requests. This number is the INRQ field.
  Writing a one to this bit will allow the USBC to perform infinite IN requests when the pipe is not frozen.
- **INRQ: IN Request Number before Freeze**
  This field contains the number of IN transactions before the USBC freezes the pipe. The USBC will perform INRQ IN requests before freezing the pipe. This counter is automatically decreased by 1 each time an IN request has been successfully performed.
  This register has no effect when the INMODE bit is 1(infinite IN requests generation till the pipe is not frozen).

17.8 Module Configuration

The specific configuration for each USBC instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 17-10. MODULE Clock Name

PB Clock Name	Description
CLK_USBC_APB	Clock for the USBC bus interface
CLK_USBC_AHB	Clock for the USBC AHB interface
CGLK	The generic clock used for the USBC is GCLK7

Table 17-11. Register Reset Values

Register	Reset Value
UVERS	0x00000310
UFEATURES	0x00000007
UADDRSIZE	0x00001000
UNAME1	0x48555342
UNAME2	0x484F5354

18. Advanced Encryption Standard (AESA)

Rev: 1.0.2.0

18.1 Features

- **Compliant with *FIPS Publication 197, Advanced Encryption Standard (AES)***
- **128-bit cryptographic key**
- **Five confidentiality modes of operation as recommended in *NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques:***
  - **Electronic Code Book**
  - **Cipher Block Chaining**
  - **Cipher Feedback**
  - **Output Feedback**
  - **Counter**
- **Short encryption and decryption time of 11 clock cycles with 128-bit cryptographic key**
- **Buffering of input and output data for non-stop processing of multiple data blocks**
- **DMA interface for multiple data block processing with minimal CPU intervention**
- **Hardware countermeasures against differential power analysis attacks**

18.2 Overview

The Advanced Encryption Standard module (AESA) is compliant with the *FIPS (Federal Information Processing Standard) Publication 197, Advanced Encryption Standard (AES)*, which specifies a symmetric block cipher that is used to encrypt and decrypt electronic data. *Encryption* is the transformation of a usable message, called the *plaintext*, into an unreadable form, called the *ciphertext*. On the other hand, *decryption* is the transformation that recovers the plaintext from the ciphertext.

AESA supports 128 bits cryptographic key size.

AESA supports all five *confidentiality modes of operation* (Electronic Code Book (ECB), Cipher Book Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Counter (CTR)) for symmetric key block cipher algorithms as recommended in the *NIST (National Institute of Standards and Technology) Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques* (see Section 18.4.2 on page 432). For the CFB mode, AESA supports data segment sizes of 8, 16, 32, 64, and 128 bits.

AESA requires 11 clock cycles to process one block (128 bits) of input data, where N_r is the number of rounds required to process one data block and is 10 when the key size is 128 bits. The relationship between the module's clock frequency and throughput (in bytes per second) is given by

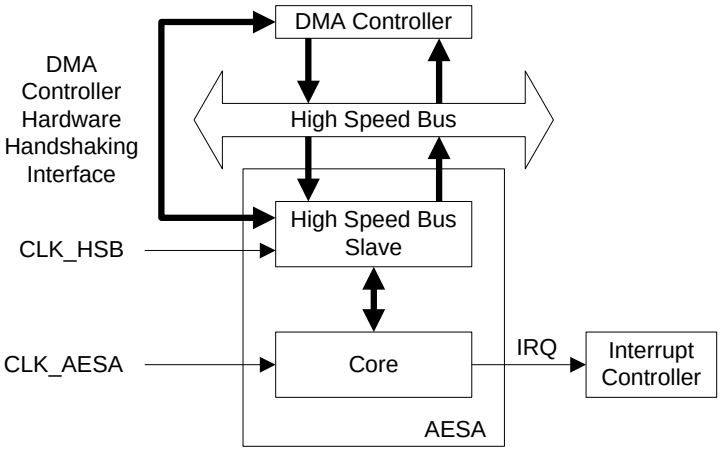
$$\text{Clock Frequency} = \left(\frac{\text{Throughput}}{16}\right) \times (11)$$

AESA is able to process multiple data blocks without stopping. This is due to the buffering of the input and output data within the module, which allows a new data block to be written to and the previous data block to be read from it while the current data block is being processed.

AESA is able to interface with a DMA controller, thus allowing the processing of multiple data blocks with minimal CPU intervention. Two channels are supported by the DMA interface - one for writing data to AESA and one for reading data from AESA.

Finally, AESA supports several hardware countermeasures that are useful for protecting data against differential power analysis attacks (Section 18.4.5 on page 434).

Figure 18-1. AESA Block Diagram



18.3 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

18.3.1 Power Management

If the CPU enters a sleep mode that disables clocks used by AESA, the module will stop functioning and resume operation after the system wakes up from sleep mode.

18.3.2 Clocks

The clock (CLK_AESA) for AESA's core operations (such as encryption and decryption) is a generic clock (GCLK). It is recommended that AESA be disabled before CLK_AESA is disabled to avoid freezing the module in an undefined state.

18.3.3 Interrupts

The AESA interrupt request line is connected to the NVIC. Using the AESA interrupt requires the NVIC to be programmed first.

18.4 Functional Description

18.4.1 Basic Programming and Operation

AESA must be enabled before it can be programmed or used. It is enabled by writing a one to the Enable Module (ENABLE) bit in the Control (CTRL) register. The module is disabled by writing a zero to the same bit.

AESA supports both the encryption and decryption of data. The desired mode of data processing is selected by programming the Encryption (ENCRYPT) bit in the MODE register.

The 128-bit key is written to the four 32-bit KEY registers. Note that access to the KEY registers is by 32-bit words only (i.e., no halfword or byte access).

The input data for processing is written to an input buffer consisting of four 32-bit registers through the Input Data (IDATA) register address. The input buffer register that is written to when the next write is performed is indicated by the Input Data Word (IDATAW) field in the Data Buffer Pointer (DATABUFPTR) register. This field is incremented by one or wrapped by hardware when a write to the IDATA register address is performed. This field can also be programmed, allowing the user direct control over which input buffer register to write to. Note that when AESA is in the CFB operation mode with the data segment size less than 128 bits, the input data must be written to the first (i.e., DATABUFPTR.IDATAW = 0) and/or second (i.e., DATABUFPTR.IDATAW = 1) input buffer registers (see Table 18-1 on page 431).

Once the input data is written to the input buffer, data processing starts automatically. After the content of the input buffer has been transferred out, the Input Buffer Ready (IBUFRDY) bit in the Status Register (SR) is set by hardware (which triggers an interrupt request if the corresponding IBUFRDY bit in the Interrupt Enable Register (IER) is programmed to '1'). This bit is cleared by hardware when new input data is written to the relevant input buffer registers.

An *initialization vector* or an initial *counter* is required as an input to the encryption and decryption processes for all confidentiality modes of operation, except the ECB operation mode (see Section 18.4.2 on page 432). The initialization vector or initial counter is written to the four 32-bit Initialization Vector (INITVECT) registers. Note that access to the INITVECT registers is by 32-bit words only (i.e., no halfword or byte access).

When data processing has completed, the Output Data Ready (ODATARDY) bit in the SR is set by hardware (which triggers an interrupt request if the corresponding ODATARDY bit in the IER is programmed to '1'). The processed output data is read out through the Output Data (ODATA) register address from the output buffer consisting of four 32-bit registers. The output buffer register that is read from when the next read is performed is indicated by the Output Data Word (ODATAW) field in the DATABUFPTR register. This field is incremented by one or wrapped by hardware when a read from the ODATA register address is performed. This field can also be programmed, giving the user direct control over which output buffer register to read from. Note that when AESA is in the CFB operation mode with the data segment size less than 128 bits, the output data must be read from the first (i.e., DATABUFPTR.ODATAW = 0) and/or second (i.e., DATABUFPTR.ODATAW = 1) output buffer registers (see Table 18-1 on page 431). The SR.ODATARDY bit is cleared by hardware after the processed data has been read from the relevant output buffer registers.

Table 18-1. Relevant Input/Output Buffer Registers for Respective Confidentiality Modes of Operation

Confidentiality Mode of Operation	Relevant Input/Output Buffer Registers
ECB	All
CBC	All
OFB	All
128-bit CFB	All
64-bit CFB	First and second
32-bit CFB	First
16-bit CFB	First
8-bit CFB	First
CTR	All

18.4.2 Confidentiality Modes of Operation

AESA supports all five confidentiality modes of operation as recommended by the *NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques*:

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Output Feedback (OFB)
- Cipher Feedback (CFB)
  - CFB8 (8-bit data segment)
  - CFB16 (16-bit data segment)
  - CFB32 (32-bit data segment)
  - CFB64 (64-bit data segment)
  - CFB128 (128-bit data segment)
- CTR: Counter

The pre-processing, post-processing, and chaining of data required in these modes of operation are automatically performed by AESA. For complete information on these modes of operation, refer to the *NIST Special Publication 800-38A.*

The desired mode of operation is selected by programming the Operation Mode (OPMODE) field in the MODE register. For the CFB mode, the desired data segment size (8, 16, 32, 64, or 128 bits) is selected by programming the Cipher Feedback Data Size (CFDS) field in the MODE register.

With the only exception of the ECB operation mode, an initialization vector or an initial counter is required as an input to the encryption and decryption processes for all confidentiality modes of operation. The initialization vector or initial counter is stored in the four 32-bit INITVECT registers. The initialization vector or initial counter is only used for processing the first 128-bit data block of a message. For this reason, it is necessary to notify AESA whenever the next data block it is going to process is the beginning of a new message. This is done by writing a one to the New Message (NEWMSG) bit in the CTRL register.

The following paragraphs on the selection of the initialization vector or counter value should be noted to avoid compromising the confidentiality of an operation mode.

For the CBC and CFB modes, the initialization vector must be unpredictable (see *NIST Special Publication 800-38A*, *Appendix C: Generation of Initialization Vectors* for recommendations for generating unpredictable initialization vectors).

For the OFB mode, the initialization vector need not be unpredictable, but it must be unique for every message ever encrypted under a given key (see *NIST Special Publication 800-38A*, *Appendix C: Generation of Initialization Vectors* for recommendations for generating unique initialization vectors).

For the CTR mode, the counter value must be unique for each input data block that is ever encrypted under a given key, across all messages. In AESA, a counter value for each input data block is automatically generated by applying the standard incrementing function on a predefined number, *m*, of LSBs of the counter value (the initial counter value is provided by the user). This number is specified in the Module Configuration section at the end of this chapter. Note that the number of data blocks (128 bits) in the message must be no more than 2^m in order for the counter value to be unique for each input data block *within the message*. The satisfaction of the

uniquess requirement for counter values *across all messages* is dependent on the choices of the initial counter values for the messages (see *NIST Special Publication 800-38A*, *Appendix B: Generation of Counter Blocks* for recommendations for choosing initial couter values).

18.4.3 DMA Interface

AESA is able to interface with a DMA controller, thus allowing the processing of multiple data blocks with minimal CPU intervention. AESA operates in the DMA mode when the DMA bit in the MODE register is programmed to '1'.

Two channels are supported by the DMA interface - an input data channel for writing input data to AESA and an output data channel for reading output data from AESA. The destination address for the input data channel is the IDATA register address, whereas the source address for the output data channel is the ODATA register address.

Note that the DMA write transactions must be performed in the ascending word order, i.e., the first write transaction is for the first word of the input data, the second write transaction is for the second word, and so on. The number of write transactions required depends on the confidentiality mode of operation that AESA is in and is the same as the number of relevant input buffer registers as shown in Table 18-1 on page 431.

Likewise, the DMA read transactions must also be performed in the ascending word order, i.e., the first read transaction is for the first word of the output data, the second read transaction is for the second word, and so on.

18.4.4 Computation of Last N_k Words of Expanded Key

The AES algorithm takes the cryptographic key provided by the user and performs a Key Expansion routine to generate an *expanded key*. The expanded key contains a total of $4(N_r + 1)$ 32-bit words, where the first N_k (4 for a 128-bit key) words are the user-provided key.

For data encryption, the expanded key is used in the forward direction, i.e., the first four words are used in the initial round of data processing, the second four words in the first round, the third four words in the second round, and so on.

On the other hand, for data decryption, the expanded key is used in the reverse direction, i.e., the last four words are used in the initial round of data processing, the last second four words in the first round, the last third four words in the second round, and so on.

To reduce gate count, AESA does not generate and store the entire expanded key prior to data processing. Instead, it computes on-the-fly the *round key* (four 32-bit words) required for the current round of data processing. In general, the round key for the current round of data processing can be computed from the N_k words of the expanded key generated in the previous rounds.

When AESA is operating in the encryption mode, the round key for the initial round of data processing is simply the user-provided key written to the KEY registers.

On the other hand, when AESA is operating in the decryption mode, the round key for the initial round of data processing is the last four words of the expanded key, which is not available unless AESA has performed at least one encryption process prior to operating in the decryption mode. In general, the last N_k words of the expanded key must be available before decryption can start.

If desired, AESA can be instructed to compute the last N_k words of the expanded key in advance by writing a one to the Decryption Key Generate (DKEYGEN) bit in the CTRL register. The computation takes N_r clock cycles.

Alternatively, the last N_k words of the expanded key can be automatically computed by AESA when a decryption process is initiated if they have not been computed in advance or have become invalid. Note that this will introduce a latency of N_r clock cycles to the first decryption process.

The last N_k words of the expanded key are stored and reused until they are invalidated by one of the following events:

- System reset
- Software reset of AESA (by writing a one to the SWRST bit in the CTRL register)
- A change to the MODE.KEYSIZE field
- A write to any of the KEY registers

Note that the last N_k words of the expanded key is automatically generated by AESA during an encryption process, so their explicit computation is not necessary if a decryption process is preceded by an encryption process using the same key.

18.4.5 Security Features

18.4.5.1 Hardware Countermeasures Against Differential Power Analysis Attacks

AESA features four types of hardware countermeasures that are useful for protecting data against differential power analysis attacks:

- Type 1: Randomly add one cycle to data processing
- Type 2: Randomly add one cycle to data processing (other version)
- Type 3: Add a random number of clock cycles to data processing, subject to a maximum of 11 clock cycles for key size of 128 bits
- Type 4: Add random spurious power consumption during data processing

By default, all countermeasures are enabled. One or more of the countermeasures can be disabled by programming the Countermeasure Type (CTYPE) field in the MODE register.

The countermeasures use random numbers generated by a deterministic random number generator embedded in AESA. The seed for the random number generator is written to the DRNGSEED register. Note that access to the DRNGSEED register is by 32-bit words only (i.e., no halfword or byte access). Note also that a new seed must be written after a change in the key size.

Note that enabling countermeasures reduces AESA's throughput. In short, the throughput is highest with all the countermeasures disabled. On the other hand, with all of the countermeasures enabled, the best protection is achieved but the throughput is worst.

18.5 User Interface

Table 18-2. AESA Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CTRL	Read/Write	0x00000000
0x04	Mode Register	MODE	Read/Write	0x000F0000
0x08	Data Buffer Pointer Register	DATABUFPTR	Read/Write	0x00000000
0x0C	Status Register	SR	Read-only	0x00010000
0x10	Interrupt Enable Register	IER	Write-only	0x00000000
0x14	Interrupt Disable Register	IDR	Write-only	0x00000000
0x18	Interrupt Mask Register	IMR	Read-only	0x00000000
0x20	Key Register 0	KEY0	Write-only	0x00000000
0x24	Key Register 1	KEY1	Write-only	0x00000000
0x28	Key Register 2	KEY2	Write-only	0x00000000
0x2C	Key Register 3	KEY3	Write-only	0x00000000
0x30	Key Register 4	KEY4	Write-only	0x00000000
0x34	Key Register 5	KEY5	Write-only	0x00000000
0x38	Key Register 6	KEY6	Write-only	0x00000000
0x3C	Key Register 7	KEY7	Write-only	0x00000000
0x40	Initialization Vector Register 0	INITVECT0	Write-only	0x00000000
0x44	Initialization Vector Register 1	INITVECT1	Write-only	0x00000000
0x48	Initialization Vector Register 2	INITVECT2	Write-only	0x00000000
0x4C	Initialization Vector Register 3	INITVECT3	Write-only	0x00000000
0x50	Input Data Register	IDATA	Write-only	-
0x60	Output Data Register	ODATA	Read-only	-
0x70	DRNG Seed Register	DRNGSEED	Write-only	0x00000000
0xF8	Parameter Register	PARAMETER	Read-only	-(1)
0xFC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value for this register is device specific. Refer to the Module Configuration section at the end of this chapter.

18.5.1 Control Register

Name: CTRL

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	SWRST
7	6	5	4	3	2	1	0
-	-	-	-	-	NEWMSG	DKEYGEN	ENABLE

- **SWRST: Software Reset**
  Writing a one to this bit resets the module.
  Writing a zero to this bit has no effect.

- **NEWMSG: New Message**
  Writing a one to this bit notifies the module that the next input data block is the beginning of a new message.
  Writing a zero to this bit has no effect.

- **DKEYGEN: Decryption Key Generate**
  Writing a one to this bit starts the computation of the last N_k words of the expanded key
  Writing a zero to this bit has no effect.

- **ENABLE: Enable Module**
  Writing a one to this bit enables the module.
  Writing a zero to this bit disables the module.

18.5.2 Mode Register

Name: MODE

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x000F0000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	CTYPE			
15	14	13	12	11	10	9	8
-	-	-	-	-	CFBS		
7	6	5	4	3	2	1	0
-	OPMODE			DMA	-	-	ENCRYPT

- **CTYPE: Countermeasure Type**

CTYPE				Description
X	X	X	0	Countermeasure type 1 is disabled
X	X	X	1	Countermeasure type 1 is enabled
X	X	0	X	Countermeasure type 2 is disabled
X	X	1	X	Countermeasure type 2 is enabled
X	0	X	X	Countermeasure type 3 is disabled
X	1	X	X	Countermeasure type 3 is enabled
0	X	X	X	Countermeasure type 4 is disabled
1	X	X	X	Countermeasure type 4 is enabled

See Section 18.4.5 on page 434 for descriptions of the various countermeasure types.
All countermeasures are enabled by default.

- **CFBS: Cipher Feedback Data Segment Size**

CFBS	Description
0	128 bits
1	64 bits
2	32 bits
3	16 bits
4	8 bits
Others	Reserved

See *NIST Special Publications 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques* for information on the CFB mode of operation.
See Section 18.4.2 on page 432 for information on operating AESA in the CFB mode.

- **OPMODE: Confidentiality Mode of Operation**

OPMODE	Description
0	Electronic Code Book
1	Cipher Block Chaining
2	Cipher Feedback
3	Output Feedback
4	Counter
Others	Reserved

See *NIST Special Publications 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques* for descriptions of the various confidentiality modes of operation.
See Section 18.4.2 on page 432 for information on operating AESA in these modes.

- **DMA: DMA Mode**

DMA	Description
0	Non-DMA mode
1	DMA mode

- **ENCRYPT: Encryption**

ENCRYPT	Description
0	Decryption
1	Encryption

18.5.3 Data Buffer Pointer Register

Name: DATABUFPTR

Access Type: Read/Write

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	ODATAW		-	-	IDATAW	

- **ODATAW: Output Data Word**
  Writing to this field changes the value of the output buffer pointer, which determines which of the four output buffer registers is read from when the next read from the ODATA register address is performed.

- **IDATAW: Input Data Word**
  Writing to this field changes the value of the input buffer pointer, which determines which of the four input buffer registers is written to when the next write to the IDATA register address is performed.

18.5.4 Status Register

Name: SR

Access Type: Read-only

Offset: 0x0C

Reset Value: 0x00010000

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

- **IBUFRDY: Input Buffer Ready**
  This bit is set when the input buffer is ready to receive input data.
  This bit is cleared when the input buffer is not ready to receive input data.

- **ODATARDY: Output Data Ready**
  This bit is set when an encryption or decryption has completed and the processed data can be read from the ODATAn.
  This bit is cleared when all ODATAn have been read.

18.5.5 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	IBUFRDY

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ODATARDY

Writing a one to a valid bit in this register sets the corresponding bit in the IMR.
Writing a zero to any bit in this register has no effect.

18.5.6 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	IBUFRDY

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ODATARDY

Writing a one to a valid bit in this register clears the corresponding bit in the IMR.
Writing a zero to any bit in this register has no effect.

18.5.7 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	IBUFRDY

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ODATARDY

1: The corresponding interrupt is enabled.
0: The corresponding interrupt is disabled.
A valid bit in this register is set when a one is written to the corresponding bit in the IER.
A valid bit in this register is cleared when a one is written to the corresponding bit in the IDR.

18.5.8 Key Registers

Name: KEYn

Access Type: Write-only

Offset: 0x20 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEYn[31:24]							

23	22	21	20	19	18	17	16
KEYn[23:16]							

15	14	13	12	11	10	9	8
KEYn[15:8]							

7	6	5	4	3	2	1	0
KEYn[7:0]							

- **KEYn: Key Word *n***
  This register stores the n^{th} word of the cryptographic key.

18.5.9 Initialization Vector Registers

Name: INITVECTn

Access Type: Write-only

Offset: 0x40 + (n * 0x04)

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
INITVECTn[31:24]							

23	22	21	20	19	18	17	16
INITVECTn[23:16]							

15	14	13	12	11	10	9	8
INITVECTn[15:8]							

7	6	5	4	3	2	1	0
INITVECTn[7:0]							

- **INITVECTn: Initialization Vector Word *n***

  This register stores the n^{th} word of the initialization vector or counter used by all confidentiality modes of operation, except the ECB operation mode:

Confidentiality Mode of Operation	Description
CBC, OFB, CFB	Initialization vector
CTR	Counter
ECB	Not used

18.5.10 Input Data Register

Name: IDATA

Access Type: Write-only

Offset: 0x50

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
IDATA[31:24]							

23	22	21	20	19	18	17	16
IDATA[23:16]							

15	14	13	12	11	10	9	8
IDATA[15:8]							

7	6	5	4	3	2	1	0
IDATA[7:0]							

- **IDATA: Input Data**
  A write to this register corresponds to a write to one of the four input buffer registers.
  The input buffer register that is written to is given by the CTRL.IDATAW field.

18.5.11 Output Data Register

Name: ODATA

Access Type: Read-only

Offset: 0x60

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
ODATA[31:24]							

23	22	21	20	19	18	17	16
ODATA[23:16]							

15	14	13	12	11	10	9	8
ODATA[15:8]							

7	6	5	4	3	2	1	0
ODATA[7:0]							

- **ODATA: Output Data**

  A read from this register corresponds to a read from one of the four output buffer registers.

  The output buffer register that is read from is given by the CTRL.ODATAW field.

18.5.12 DRNG Seed Register

Name: DRNG Seed

Access Type: Write-only

Offset: 0x70

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
SEED[31:24]							

23	22	21	20	19	18	17	16
SEED[23:16]							

15	14	13	12	11	10	9	8
SEED[15:8]							

7	6	5	4	3	2	1	0
SEED[7:0]							

- **SEED: DRNG Seed**
  A write to this register corresponds to loading a new seed into the DRNG.

18.5.13 Parameter Register

Name: Parameter

Access Type: Read-only

Offset: 0xF8

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	CTRMEAS

7	6	5	4	3	2	1	0
-	-	-	OPMODE			KEYSIZE	

- **CTRMEAS: Countermeasures**

CTRMEAS	Description
0	Countermeasures not implemented
1	Countermeasures implemented

- **OPMODE: Maximum Number of Confidentiality Modes of Operation**

OPMODE	Description
0	ECB only
1	ECB and CBC only
2	ECB, CBC, and CFB only
3	ECB, CBC, CFB, and OFB only
4	ECB, CBC, CFB, OFB, and CTR

- **KEYSIZE: Maximum Key Size**

KEYSIZE	Description
0	128-bit only
1	128- and 192-bit only
2	128-, 192-, and 256-bit only

18.5.14 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  This field is reserved.
  This field is not associated with any functionality.

- **VERSION: Version Number**
  This field stores the version number of the module.
  This field is not associated with any functionality.

18.6 Module Configuration

The specific configuration for each AESA instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 18-3. Module configuration

Feature	AESA
KeySize	128 bit only
Opmode	ECB, CBC, CFB, OFB and CTR
Countermeasure	Implemented

Table 18-4. Module clock name

Clock name	Description
CLK_AESA_AHB	Clock for the AESA AHB interface
GCLK	The generic clock used for the AESA is GCLK4

Table 18-5. Register Reset Values

Register	Reset Value
VERSION	0x00000102
PARAMETER	0x00000110

19. Asynchronous Timer (AST)

Rev: 3.1.1.1

19.1 Features

- **32-bit counter with 32-bit prescaler**
- **Clocked Source**
  - **System RC oscillator (RCSYS)**
  - **32 KHz clock (OSC32 or RC32)**
  - **APB clock**
  - **Generic clock (GCLK)**
  - **1 KHz clock (OSC32 or RC32)**
- **Operation and wakeup during backup**
- **Optional calendar mode supported**
- **Digital prescaler tuning for increased accuracy**
- **Periodic interrupt(s) and peripheral event(s) supported**
- **Alarm interrupt(s) and peripheral event(s) supported**
  - **Optional clear on alarm**

19.2 Overview

The Asynchronous Timer (AST) enables periodic interrupts and periodic peripheral events, as well as interrupts and peripheral events at a specified time in the future. The AST consists of a 32-bit prescaler which feeds a 32-bit up-counter. The prescaler can be clocked from different clock sources, including the low-power 32 kHz oscillator, which allows the AST to be used as a real-time timer with a maximum timeout of more than 100 years. Also, the PB clock or a generic clock can be used for high-speed operation, allowing the AST to be used as a general timer.

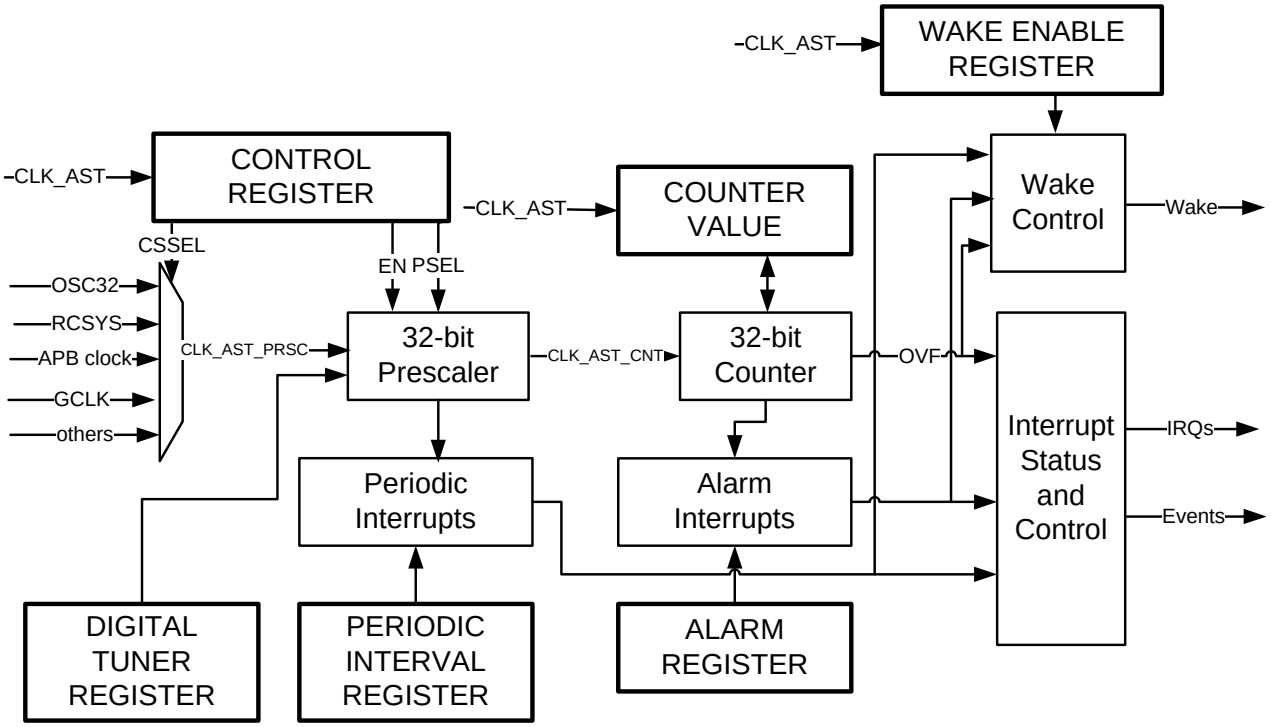
The AST can generate periodic interrupts and peripheral events from output from the prescaler, as well as alarm interrupts and peripheral events, which can trigger at any counter value. Additionally, the timer can trigger an overflow interrupt and peripheral event, and be reset on the occurrence of any alarm. This allows periodic interrupts and peripheral events at very long and accurate intervals.

To keep track of time during backup mode the AST can run while the core is powered off. This will reduce the power consumption when the system is idle. The AST can also wake up the system from backup mode using either the alarm wakeup, periodic wakeup. or overflow wakeup mechanisms.

The AST has been designed to meet the system tick and Real Time Clock requirements of most embedded operating systems.

19.3 Block Diagram

Figure 19-1. Asynchronous Timer Block Diagram



19.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

19.4.1 Power Management

When the AST is enabled, it will remain clocked as long as its selected clock source is running. It can also wake the CPU from the currently active sleep mode. Refer to Section 10. “Power Manager (PM)” on page 109 for details on the different sleep modes.

19.4.2 Clocks

The clock for the AST bus interface (CLK_AST) is generated by the Power Manager. This clock is turned on by default, and can be enabled and disabled in the Power Manager.

A number of clocks can be selected as source for the internal prescaler clock CLK_AST_PRSC. The prescaler, counter, and interrupt will function as long as this selected clock source is active. The selected clock must be enabled in the System Control Interface (SCIF).

The following clock sources are available:

- System RC oscillator (RCSYS oscillator). This oscillator is always enabled, except in some sleep modes. Refer to Section 42. “Electrical Characteristics” on page 1121 for the characteristic frequency of this oscillator.
- 32kHz clock (OSC32 or RC32). The oscillator must be enabled before use. Selection between OSC32 and RC32 is done inside the Backup Power Manager module.

- Peripheral Bus clock (PB clock). This is the clock of the peripheral bus the AST is connected to.
- Generic clock (GCLK). One of the generic clocks is connected to the AST. This clock must be enabled before use, and remains enabled in sleep modes when the PB clock is active.
- 1kHz clock from the 32kHz oscillator or 32kHz RC Oscillator (CLK_1K). The oscillator must be enabled before use. Selection between OSC32 and RC32 is done inside the Backup Power Manager module.

In backup mode only the 32kHz oscillator and the 32kHz RC Oscillator are available. The 1kHz outputs of those oscillators are also available. Refer to Section 11. “Backup Power Manager (BPM)” on page 140 for details.

19.4.3 Interrupts

The AST interrupt request lines are connected to the NVIC. Using the AST interrupts requires the NVIC to be programmed first.

19.4.4 Peripheral Events

The AST peripheral events are connected via the Peripheral Event System. Refer to Section 31. “Peripheral Event Controller (PEVC)” on page 845 for details.

19.4.5 Debug Operation

The AST prescaler and counter is not frozen during debug operation when the Core is halted, unless the bit corresponding to the AST is set in the Peripheral Debug Register (PDBG).

If the AST is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

19.5 Functional Description

19.5.1 Initialization

Before enabling the AST, the internal AST clock CLK_AST_PRSC must be enabled, following the procedure specified in Section 19.5.1.1. The Clock Source Select field in the Clock register (CLOCK.CSSEL) selects the source for this clock. The Clock Enable bit in the Clock register (CLOCK.CEN) enables the CLK_AST_PRSC.

When CLK_AST_PRSC is enabled, the AST can be enabled by writing a one to the Enable bit in the Control Register (CR.EN).

19.5.1.1 Enabling and Disabling the AST Clock

The Clock Source Selection field (CLOCK.CSSEL) and the Clock Enable bit (CLOCK.CEN) cannot be changed simultaneously. Special procedures must be followed for enabling and disabling the CLK_AST_PRSC and for changing the source for this clock.

To enable CLK_AST_PRSC:

- Write the selected value to CLOCK.CSSEL
- Wait until SR.CLKBUSY reads as zero
- Write a one to CLOCK.CEN, without changing CLOCK.CSSEL
- Wait until SR.CLKBUSY reads as zero

To disable the clock:

- Write a zero to CLOCK.CEN to disable the clock, without changing CLOCK.CSSEL

- Wait until SR.CLKBUSY reads as zero

19.5.1.2 *Changing the Source Clock*

The CLK_AST_PRSC must be disabled before switching to another source clock. The Clock Busy bit in the Status Register (SR.CLKBUSY) indicates whether the clock is busy or not. This bit is set when the CEN bit in the CLOCK register is changed, and cleared when the CLOCK register can be changed.

To change the clock:

- Write a zero to CLOCK.CEN to disable the clock, without changing CLOCK.CSSEL
- Wait until SR.CLKBUSY reads as zero
- Write the selected value to CLOCK.CSSEL
- Wait until SR.CLKBUSY reads as zero
- Write a one to CLOCK.CEN to enable the clock, without changing the CLOCK.CSSEL
- Wait until SR.CLKBUSY reads as zero

19.5.2 Basic Operation

19.5.2.1 *Prescaler*

When the AST is enabled, the 32-bit prescaler will increment on the rising edge of CLK_AST_PRSC. The prescaler value cannot be read or written, but it can be reset by writing a one to the Prescaler Clear bit in the Control Register (CR.PCLR).

The Prescaler Select field in the Control Register (CR.PSEL) selects the prescaler bit PSEL as source clock for the counter (CLK_AST_CNT). This results in a counter frequency of:

$$f_{CNT} = \frac{f_{PRSC}}{2^{PSEL+1}}$$

where f_{PRSC} is the frequency of the internal prescaler clock CLK_AST_PRSC.

19.5.2.2 *Counter Operation*

When enabled, the AST will increment on every 0-to-1 transition of the selected prescaler tapping. When the Calendar bit in the Control Register (CR.CAL) is zero, the counter operates in counter mode. It will increment until it reaches the top value of 0xFFFFFFFF, and then wrap to 0x00000000. This sets the status bit Overflow in the Status Register (SR.OVF). Optionally, the counter can also be reset when an alarm occurs (see Section 19.5.3.2 on page 457. This will also set the OVF bit.

The AST counter value can be read from or written to the Counter Value (CV) register. Note that due to synchronization, continuous reading of the CV register with the lowest prescaler setting will skip every third value. In addition, if CLK_AST_PRSC is as fast as, or faster than, the CLK_AST, the prescaler value must be 3 or higher to be able to read the CV without skipping values.

19.5.2.3 Calendar Operation

When the CAL bit in the Control Register is one, the counter operates in calendar mode. Before this mode is enabled, the prescaler should be set up to give a pulse every second. The date and time can then be read from or written to the Calendar Value (CALV) register.

Time is reported as seconds, minutes, and hours according to the 24-hour clock format. Date is the numeral date of month (starting on 1). Month is the numeral month of the year (1 = January, 2 = February, etc.). Year is a 6-bit field counting the offset from a software-defined leap year (e.g. 2000). The date is automatically compensated for leap years, assuming every year divisible by 4 is a leap year.

All peripheral events and interrupts work the same way in calendar mode as in counter mode. However, the Alarm Register (ARn) must be written in time/date format for the alarm to trigger correctly.

19.5.3 Interrupts

The AST can generate five separate interrupt requests:

- OVF: OVF
- PER: PER0
- ALARM: ALARM0
- CLKREADY
- READY

This allows the user to allocate separate handlers and priorities to the different interrupt types.

The generation of the PER interrupt is described in Section 19.5.3.1., and the generation of the ALARM interrupt is described in Section 19.5.3.2. The OVF interrupt is generated when the counter overflows, or when the alarm value is reached, if the Clear on Alarm bit in the Control Register is one. The CLKREADY interrupt is generated when SR.CLKBUSY has a 1-to-0 transition, and indicates that the clock synchronization is completed. The READY interrupt is generated when SR.BUSY has a 1-to-0 transition, and indicates that the synchronization described in Section 19.5.8 is completed.

An interrupt request will be generated if the corresponding bit in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in SR is cleared by writing a one to the corresponding bit in the Status Clear Register (SCR).

The AST interrupts can wake the CPU from any sleep mode where the source clock and the NVIC is active.

19.5.3.1 Periodic Interrupt

The AST can generate periodic interrupts. If the PERn bit in the Interrupt Mask Register (IMR) is one, the AST will generate an interrupt request on the 0-to-1 transition of the selected bit in the

prescaler when the AST is enabled. The bit is selected by the Interval Select field in the corresponding Periodic Interval Register (PIRn.INSEL), resulting in a periodic interrupt frequency of

$$f_{PA} = \frac{f_{CS}}{2^{INSEL+1}}$$

where f_{CS} is the frequency of the selected clock source.

The corresponding PERn bit in the Status Register (SR) will be set when the selected bit in the prescaler has a 0-to-1 transition.

Because of synchronization, the transfer of the INSEL value will not happen immediately. When changing/setting the INSEL value, the user must make sure that the prescaler bit number INSEL will not have a 0-to-1 transition before the INSEL value is transferred to the register. In that case, the first periodic interrupt after the change will not be triggered.

19.5.3.2 Alarm Interrupt

The AST can also generate alarm interrupts. If the ALARMn bit in IMR is one, the AST will generate an interrupt request when the counter value matches the selected alarm value, when the AST is enabled. The alarm value is selected by writing the value to the VALUE field in the corresponding Alarm Register (ARn.VALUE).

The corresponding ALARMn bit in SR will be set when the counter reaches the selected alarm value.

Because of synchronization, the transfer of the alarm value will not happen immediately. When changing/setting the alarm value, the user must make sure that the counter will not count the selected alarm value before the value is transferred to the register. In that case, the first alarm interrupt after the change will not be triggered.

If the Clear on Alarm bit in the Control Register (CR.CAn) is one, the corresponding alarm interrupt will clear the counter and set the OVF bit in the Status Register. This will generate an overflow interrupt if the OVF bit in IMR is set.

19.5.4 Peripheral Events

The AST can generate a number of peripheral events:

- OVF
- PER0
- ALARM0

The PERn peripheral event(s) is generated the same way as the PER interrupt, as described in Section 19.5.3.1. The ALARMn peripheral event(s) is generated the same way as the ALARM interrupt, as described in Section 19.5.3.2. The OVF peripheral event is generated the same way as the OVF interrupt, as described in Section 19.5.3-

The peripheral event will be generated if the corresponding bit in the Event Mask (EVM) register is set. Bits in EVM register are set by writing a one to the corresponding bit in the Event Enable

(EVE) register, and cleared by writing a one to the corresponding bit in the Event Disable (EVD) register.

19.5.5 AST wakeup

The AST can wake up the system by trigerring a PM interrupt. A wakeup can be generated when the counter overflows, when the counter reaches the selected alarm value, or when the selected prescaler bit has a 0-to-1 transition. This wakeup is propagated to the PM and a PM interrupt is generated if enabled.

The AST wakeup is enabled by writing a one to the corresponding bit in the Wake Enable Register (WER). When the CPU wakes from sleep, the wake signal must be cleared by writing a one to the corresponding bit in SCR to clear the internal wake signal to the sleep controller. If the wake signal is not cleared after waking from sleep, the next sleep instruction will have no effect because the CPU will wake immediately after this sleep instruction.

The AST wakeup can wake the CPU from any sleep mode where the source clock is active. The AST wakeup can be configured independently of the interrupt masking.

19.5.6 Backup Mode

If the AST is configured to use a clock that is available in Backup mode, the AST can be used to wake up the system from backup. Both the alarm wakeup, periodic wakeup, and overflow wakeup mechanisms can be used in this mode.

When waking up from Backup mode all control registers will have the same value as before the backup was entered, except the Interrupt Mask Register (IMR). IMR will be reset with all interrupts turned off. The software must first reconfigure the NVIC and then enable the interrupts in the AST to again receive interrupts from the AST.

The CV register will be updated with the current counter value directly after wakeup from shutdown. The SR will show the status of the AST, including the status bits set during backup operation.

When waking up the system from backup the CPU will start executing code from the reset start address.

19.5.7 Digital Tuner

The digital tuner adds the possibility to compensate for a too-slow or a too-fast input clock. The ADD bit in the Digital Tuner Register (DTR.ADD) selects if the prescaler frequency should be reduced or increased. If ADD is ‘0’, the prescaler frequency is reduced:

$$f_{TUNED} = f_0\left(1 - \frac{1}{\text{roundup}\left(\frac{256}{VALUE}\right) \cdot (2^{EXP}) + 1}\right)$$

where f_{TUNED} is the tuned frequency, f_0 is the original prescaler frequency, and VALUE and EXP are the corresponding fields to be programmed in DTR. Note that DTR.EXP must be greater than zero. Frequency tuning is disabled by programming DTR.VALUE as zero.

If ADD is ‘1’, the prescaler frequency is increased:

$$f_{TUNED} = f_0 \left(1 + \frac{1}{\text{roundup}\left(\frac{256}{VALUE}\right) \cdot (2^{EXP}) - 1} \right)$$

Note that for these formulas to be within an error of 0.01%, it is recommended that the prescaler bit that is used as the clock for the counter (selected by CR.PSEL) or to trigger the periodic interrupt (selected by PIRn.INSEL) be bit 6 or higher.

19.5.8 Synchronization

As the prescaler and counter operate asynchronously from the user interface, the AST needs a few clock cycles to synchronize the values written to the CR, CV, SCR, WER, EVE, EVD, PIRn, ARn, and DTR registers. The Busy bit in the Status Register (SR.BUSY) indicates that the synchronization is ongoing. During this time, writes to these registers will be discarded and reading will return a zero value.

Note that synchronization takes place also if the prescaler is clocked from CLK_AST.

19.6 User Interface

Table 19-1. AST Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Read/Write	0x00000000
0x04	Counter Value	CV	Read/Write	0x00000000
0x08	Status Register	SR	Read-only	0x00000000
0x0C	Status Clear Register	SCR	Write-only	0x00000000
0x10	Interrupt Enable Register	IER	Write-only	0x00000000
0x14	Interrupt Disable Register	IDR	Write-only	0x00000000
0x18	Interrupt Mask Register	IMR	Read-only	0x00000000
0x1C	Wake Enable Register	WER	Read/write	0x00000000
0x20	Alarm Register 0(2)	AR0	Read/Write	0x00000000
0x24	Alarm Register 1(2)	AR1	Read/Write	0x00000000
0x30	Periodic Interval Register 0(2)	PIR0	Read/Write	0x00000000
0x34	Periodic Interval Register 1(2)	PIR1	Read/Write	0x00000000
0x40	Clock Control Register	CLOCK	Read/Write	0x00000000
0x44	Digital Tuner Register	DTR	Read/Write	0x00000000
0x48	Event Enable	EVE	Write-only	0x00000000
0x4C	Event Disable	EVD	Write-only	0x00000000
0x50	Event Mask	EVM	Read-only	0x00000000
0x54	Calendar Value	CALV	Read/Write	0x00000000
0xF0	Parameter Register	PARAMETER	Read-only	-(1)
0xFC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

2. The number of Alarm and Periodic Interval registers are device specific. Refer to the Module Configuration section at the end of this chapter.

19.6.1 Control Register

Name: CR

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	PSEL				

15	14	13	12	11	10	9	8
-	-	-	-	-	-	CA1	CA0

7	6	5	4	3	2	1	0
-	-	-	-	-	CAL	PCLR	EN

When the SR.BUSY bit is set, writes to this register will be discarded and this register will read as zero.

- **PSEL: Prescaler Select**
  Selects prescaler bit PSEL as source clock for the counter.
- **CAn: Clear on Alarm n**
  0: The corresponding alarm will not clear the counter.
  1: The corresponding alarm will clear the counter.
- **CAL: Calendar Mode**
  0: The AST operates in counter mode.
  1: The AST operates in calendar mode.
- **PCLR: Prescaler Clear**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the prescaler.
  This bit always reads as zero.
- **EN: Enable**
  0: The AST is disabled.
  1: The AST is enabled.

19.6.2 Counter Value

Name: CV

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
VALUE[31:24]							

23	22	21	20	19	18	17	16
VALUE[23:16]							

15	14	13	12	11	10	9	8
VALUE[15:8]							

7	6	5	4	3	2	1	0
VALUE[7:0]							

When the SR.BUSY bit is set, writes to this register will be discarded and this register will read as zero.

- **VALUE: AST Value**
  The current value of the AST counter.

19.6.3 Status Register

Name: SR

Access Type: Read-only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	CLKRDY	CLKBUSY	-	-	READY	BUSY
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

- **CLKRDY: Clock Ready**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the SR.CLKBUSY bit has a 1-to-0 transition.
- **CLKBUSY: Clock Busy**
  0: The clock is ready and can be changed.
  1: CLOCK.CEN has been written and the clock is busy.
- **READY: AST Ready**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the SR.BUSY bit has a 1-to-0 transition.
- **BUSY: AST Busy**
  0: The AST accepts writes to CR, CV, SCR, WER, EVE, EVD, ARn, PIRn, and DTR.
  1: The AST is busy and will discard writes to CR, CV, SCR, WER, EVE, EVD, ARn, PIRn, and DTR.
- **PERn: Periodic n**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the selected bit in the prescaler has a 0-to-1 transition.
- **ALARMn: Alarm n**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the counter reaches the selected alarm value.
- **OVF: Overflow**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when an overflow has occurred.

19.6.4 Status Clear Register

Name: SCR

Access Type: Write-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	CLKRDY	-	-	-	READY	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

When the SR.BUSY bit is set, writes to this register will be discarded.

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

19.6.5 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	CLKRDY	-	-	-	READY	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

19.6.6 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	CLKRDY	-	-	-	READY	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

19.6.7 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	CLKRDY	-	-	-	READY	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

19.6.8 Wake Enable Register

Name: WER

Access Type: Read/Write

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

This register enables the wakeup signal from the AST.

- **PERn: Periodic n**
  0: The CPU will not wake up from sleep mode when the selected bit in the prescaler has a 0-to-1 transition.
  1: The CPU will wake up from sleep mode when the selected bit in the prescaler has a 0-to-1 transition.
- **ALARMn: Alarm n**
  0: The CPU will not wake up from sleep mode when the counter reaches the selected alarm value.
  1: The CPU will wake up from sleep mode when the counter reaches the selected alarm value.
- **OVF: Overflow**
  0: A counter overflow will not wake up the CPU from sleep mode.
  1: A counter overflow will wake up the CPU from sleep mode.

19.6.9 Alarm Register 0

Name: AR0

Access Type: Read/Write

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
VALUE[31:24]							

23	22	21	20	19	18	17	16
VALUE[23:16]							

15	14	13	12	11	10	9	8
VALUE[15:8]							

7	6	5	4	3	2	1	0
VALUE[7:0]							

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **VALUE: Alarm Value**
  When the counter reaches this value, an alarm is generated.

19.6.10 Alarm Register 1

Name: AR1

Access Type: Read/Write

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
VALUE[31:24]							

23	22	21	20	19	18	17	16
VALUE[23:16]							

15	14	13	12	11	10	9	8
VALUE[15:8]							

7	6	5	4	3	2	1	0
VALUE[7:0]							

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **VALUE: Alarm Value**
  When the counter reaches this value, an alarm is generated.

19.6.11 Periodic Interval Register 0

Name: PIR0

Access Type: Read/Write

Offset: 0x30

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	INSEL				

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **INSEL: Interval Select**
  The PER0 bit in SR will be set when the INSEL bit in the prescaler has a 0-to-1 transition.

19.6.12 Periodic Interval Register 1

Name: PIR1

Access Type: Read/Write

Offset: 0x34

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	INSEL				

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **INSEL: Interval Select**
  The PER1 bit in SR will be set when the INSEL bit in the prescaler has a 0-to-1 transition.

19.6.13 Clock Control Register

Name: CLOCK

Access Type: Read/Write

Offset: 0x40

Reset Value: 0x00000000

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

When writing to this register, follow the sequence in Section 19.5.1 on page 454.

- **CSSEL: Clock Source Selection**
  This field defines the clock source CLK_AST_PRSC for the prescaler:

Table 19-2. Clock Source Selection

CSSEL	Clock Source
0	System RC oscillator (RCSYS)
1	32kHz oscillator (OSC32 or RC32)
2	APB clock
3	Generic clock (GCLK)
4	1kHz clock from 32kHz oscillator or 32kHz RC oscillator (CLK_1K)

- **CEN: Clock Enable**
  0: CLK_AST_PRSC is disabled.
  1: CLK_AST_PRSC is enabled.

19.6.14 Digital Tuner Register

Name: DTR

Access Type: Read/Write

Offset: 0x44

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
-	-	ADD	EXP				

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **VALUE:**
  0: The frequency is unchanged.
  1-255: The frequency will be adjusted according to the formula below.

- **ADD:**
  0: The resulting frequency is $f0\left(1 - \frac{1}{roundup\left(\frac{256}{VALUE}\right) \cdot 2^{(EXP)} + 1}\right)$ for $VALUE > 0$.

  1: The resulting frequency is $f0\left(1 + \frac{1}{roundup\left(\frac{256}{VALUE}\right) \cdot 2^{(EXP)} - 1}\right)$ for $VALUE > 0$.

- **EXP:**
  The frequency will be adjusted according to the formula above.

19.6.15 Event Enable Register

Name: EVE

Access Type: Write-only

Offset: 0x48

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will set the corresponding bit in EVM.

19.6.16 Event Disable Register

Name: EVD

Access Type: Write-only

Offset: 0x4C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will clear the corresponding bit in EVM.

19.6.17 Event Mask Register

Name: EVM

Access Type: Read-only

Offset: 0x50

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	PER0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ALARM0
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF

0: The corresponding peripheral event is disabled.
1: The corresponding peripheral event is enabled.
This bit is cleared when the corresponding bit in EVD is written to one.
This bit is set when the corresponding bit in EVE is written to one.

19.6.18 Calendar Value

Name: CALV

Access Type: Read/Write

Offset: 0x54

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
YEAR						MONTH[3:2]	

23	22	21	20	19	18	17	16
MONTH[1:0]		DAY					HOUR[4]

15	14	13	12	11	10	9	8
HOUR[3:0]				MIN[5:2]			

7	6	5	4	3	2	1	0
MIN[1:0]		SEC					

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **YEAR: Year**
  Current year. The year is considered a leap year if YEAR[1:0] = 0.
- **MONTH: Month**
  1 = January
  2 = February
  ...
  12 = December
- **DAY: Day**
  Day of month, starting with 1.
- **HOUR: Hour**
  Hour of day, in 24-hour clock format.
  Legal values are 0 through 23.
- **MIN: Minute**
  Minutes, 0 through 59.
- **SEC: Second**
  Seconds, 0 through 59.

19.6.19 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0xF0

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	PER1VALUE				

23	22	21	20	19	18	17	16
-	-	-	PER0VALUE				

15	14	13	12	11	10	9	8
PIR1WA	PIR0WA	-	NUMPIR	-	-	NUMAR	

7	6	5	4	3	2	1	0
-	DTEXPVALUE					DTEXPWA	DT

This register gives the configuration used in the specific device. Also refer to the Module Configuration section.

- **PERnVALUE: Periodic Interval n Value**
  Periodic interval prescaler n tapping if PIRnWA is zero.
- **PIRnWA: Periodic Interval n Writeable**
  0: Periodic interval n prescaler tapping is a constant value. Writes to INSEL field in PIRn register will be discarded.
  1: Periodic interval n prescaler tapping is chosen by writing to INSEL field in PIRn register.
- **NUMPIR: Number of Periodic Comparators**
  0: One periodic comparator.
  1: Two periodic comparator.
- **NUMAR: Number of Alarm Comparators**
  0: Zero alarm comparators.
  1: One alarm comparator.
  2: Two alarm comparators.
- **DTREXPVALUE: Digital Tuner Exponent Value**
  Digital tuner exponent value if DTEXPWA is zero.
- **DTREXPWA: Digital Tuner Exponent Writeable**
  0: Digital tuner exponent is a constant value. Writes to EXP field in DTR will be discarded.
  1: Digital tuner exponent is chosen by writing to EXP field in DTR.
- **DT: Digital Tuner**
  0: Digital tuner not implemented.
  1: Digital tuner implemented.

19.6.20 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

19.7 Module Configuration

The specific configuration for each AST instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 19-3. AST Configuration

Feature	AST
Number of alarm comparators	1
Number of periodic comparators	1
Digital tuner	On

Table 19-4. AST Clocks

Clock Name	Description
CLK_AST	Clock for the AST bus interface
GCLK	The generic clock used for the AST is GCLK2

Table 19-5. Register Reset Values

Register	Reset Value
VERSION	0x00000311
PARAMETER	0x00004103

20. Watchdog Timer (WDT)

Rev.: 5.0.1.0

20.1 Features

- **Watchdog Timer counter with 32-bit counter**
- **Timing window watchdog**
- **Clocked from system RC oscillator or one of the 32 KHz oscillator (OSC32 or RC32)**
- **Configuration lock**
- **WDT may be enabled at reset by a fuse**

20.2 Overview

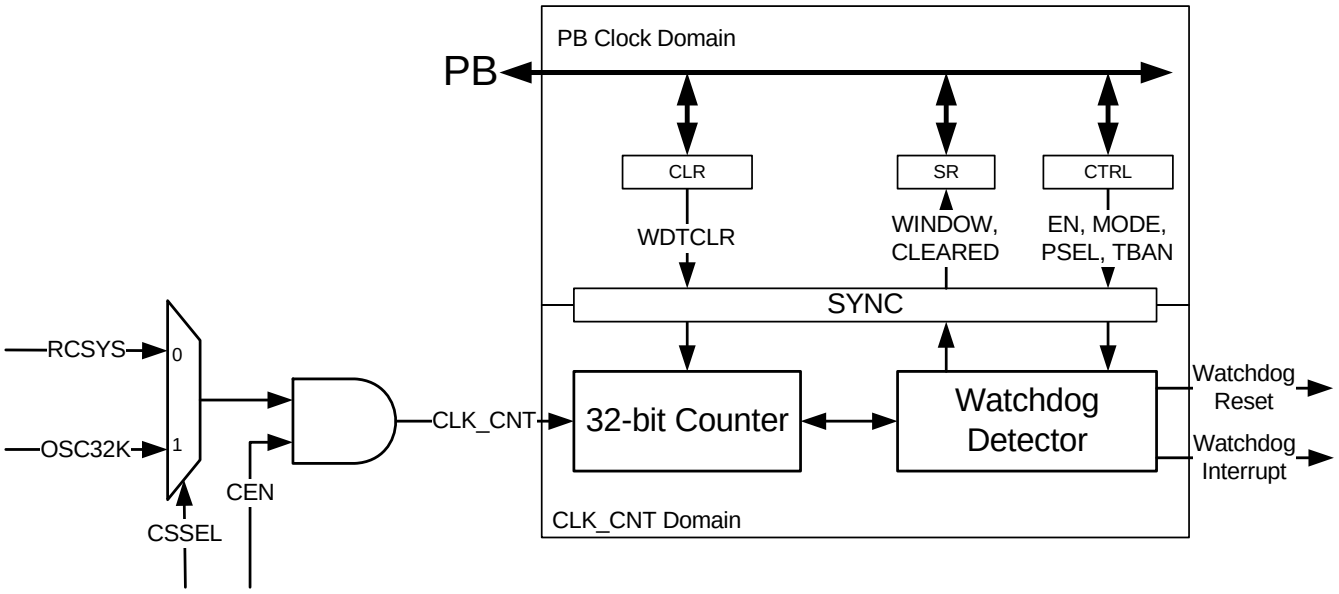
The Watchdog Timer (WDT) will reset the device unless it is periodically serviced by the software. This allows the device to recover from a condition that has caused the system to be unstable.

The WDT has an internal counter clocked from the system RC oscillator or one of the 32kHz oscillator.

The WDT counter must be periodically cleared by software to avoid a watchdog reset. If the WDT timer is not cleared correctly, the device will reset and start executing from the boot vector. If the WDT is configured in interrupt mode an interrupt request will be generated on the first timeout and a reset on the second timeout if the interrupt has not been cleared.

20.3 Block Diagram

Figure 20-1. WDT Block Diagram



20.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

20.4.1 Power Management

When the WDT is enabled, it remains clocked in all sleep modes. It is not possible to enter sleep modes where the source clock of CLK_CNT is stopped. Attempting to do so will result in the chip entering the lowest sleep mode where the source clock is running, leaving the WDT operational. Refer to Section 10. “Power Manager (PM)” on page 109 for details about sleep modes.

After a watchdog reset the WDT bit in the Reset Cause Register (RCAUSE) in the Power Manager will be set.

20.4.2 Clocks

The clock for the WDT bus interface (CLK_WDT) is generated by the Power Manager. This clock is enabled at reset, and can be disabled in the Power Manager. It is recommended to disable the WDT before disabling the clock, to avoid freezing the WDT in an undefined state.

There are two possible clock sources for the Watchdog Timer clock, CLK_CNT:

- System RC oscillator (RCSYS): This oscillator is always enabled when selected as clock source for the WDT. Refer to Section 10. “Power Manager (PM)” on page 109 for details about the RCSYS and sleep modes. Refer to Section 42. “Electrical Characteristics” on page 1121 for the characteristic frequency of this oscillator.
- 32 kHz crystal oscillator or RC oscillator (OSC32 or RC32): This oscillator has to be enabled in the Backup System Control Interface (BSCIF) before using it as clock source for the WDT. Selection between OSC32 and RC32 should be done in the Backup Power Manager.The WDT will not be able to detect if this clock is stopped.

20.4.3 Interrupt

The WDT interrupt request line is connected to the NVIC. Using the WDT interrupt requires the NVIC to be programmed first.

20.4.4 Debug Operation

The WDT counter is not frozen during debug operation, unless the Core is halted and the bit corresponding to the WDT is set in the Peripheral Debug Register (PDBG). If the WDT counter is not frozen during debug operation it will need periodically clearing to avoid a watchdog reset.

20.4.5 Fuses

The WDT can be enabled at reset. This is controlled by the WDTAUTO fuse, see Section 20.5.5 for details. Refer to the Fuse Settings section in the Flash Controller chapter for details about WDTAUTO and how to program the fuses.

20.5 Functional Description

20.5.1 Basic Mode

20.5.1.1 WDT Control Register Access

To avoid accidental disabling of the watchdog, the Control Register (CTRL) must be written twice, first with the KEY field set to 0x55, then 0xAA without changing the other bits. Failure to do so will cause the write operation to be ignored, and the value in the CTRL Register will not be changed.

20.5.1.2 Changing CLK_CNT Clock Source

After any reset, except for watchdog reset, CLK_CNT will be enabled with RCSYS as source.

To change the clock source for CLK_CNT the following steps must be taken. Note that the WDT should always be disabled before changing the CLK_CNT source:

1. Write a zero to the Clock Enable bit in the Control Register (CTRL.CEN), leaving the other bits as they are in the Control Register. This will stop CLK_CNT.

2. Read back CTRL until CTRL.CEN reads zero. The clock has now been stopped.

3. Modify the Clock Source Select bit in CTRL (CTRL.CSSEL) with your new clock selection and write it to CTRL.

4. Write a one to CTRL.CEN, leaving the other bits as they are in CTRL. This will enable the clock.

5. Read back CTRL until CTRL.CEN reads one. The clock has now been enabled.

20.5.1.3 Configuring the WDT

If the MODE bit in the Control Register (CTRL.MODE) is zero, the WDT is in basic mode. The Time Out Prescale Select (PSEL) field in CTRL (CTRL.PSEL) selects the WDT timeout period:

$$T_{timeout} = T_{psel} = 2^{(PSEL+1)} / f_{clk\_cnt}$$

20.5.1.4 Enabling the WDT

To enable the WDT write a one to the Enable bit in the Control Register (CTRL.EN). Due to internal synchronization, it will take some time for the CTRL.EN bit to read back as one.

20.5.1.5 Clearing the WDT Counter

The WDT counter must be periodically cleared within T_{psel} to avoid a watchdog reset to be issued, see Figure 20-2 on page 485. If the WDT counter is not cleared within T_{psel} a watchdog reset will be issued at the end of T_{psel}, see Figure 20-3 on page 485.

The WDT counter is cleared by writing a one to the Watchdog Clear bit in the Clear Register (CLR.WDTCLR), at any correct write to CTRL, or when the counter reaches $T_{timeout}$ and the device is reset. In basic mode, CLR.WDTCLR can be written at any time when the WDT Counter Cleared bit in the Status Register (SR.CLEARED) is one. Due to internal synchronization, clearing the WDT counter takes some time. The SR.CLEARED bit is cleared when writing to CLR.WDTCLR and set when the clearing is done. Any write to the CLR.WDTCLR bit while SR.CLEARED is zero will not clear the counter.

Writing to the CLR.WDTCLR bit has to be done in a particular sequence to be valid. The Clear Register must be written twice, first by writing 0x55 to the CLR.KEY field and CLR.WDTCLR set to one, then by writing 0xAA to CLR.KEY without changing the CLR.WDTCLR bit. Writing to the Clear Register without the correct sequence has no effect.

Figure 20-2. Basic Mode WDT Timing Diagram, Normal Operation

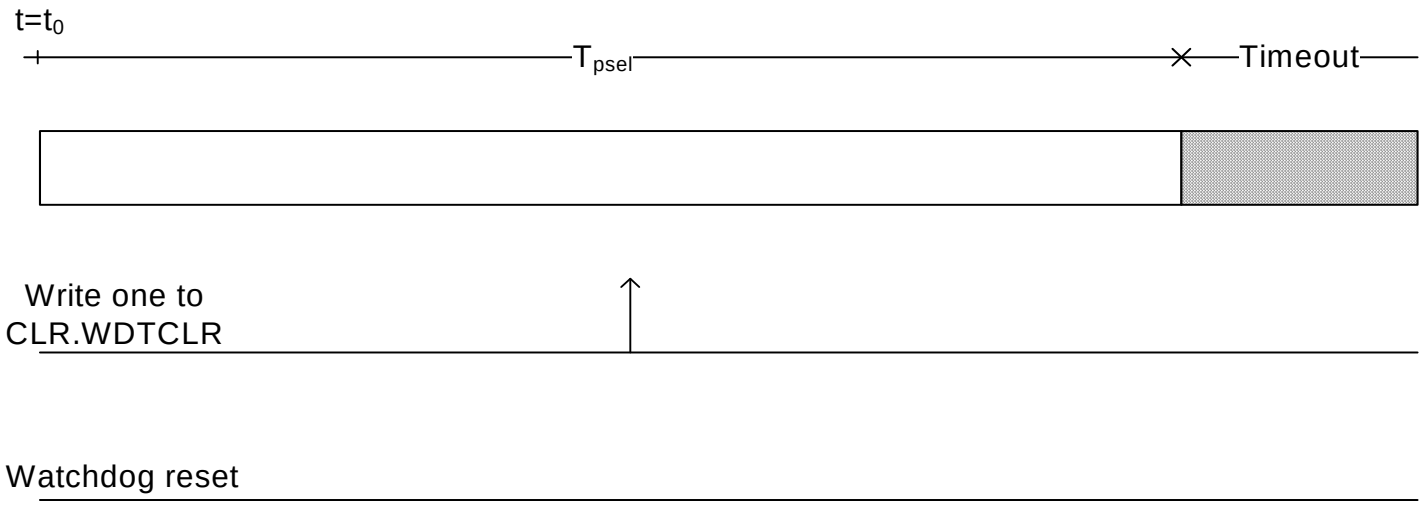
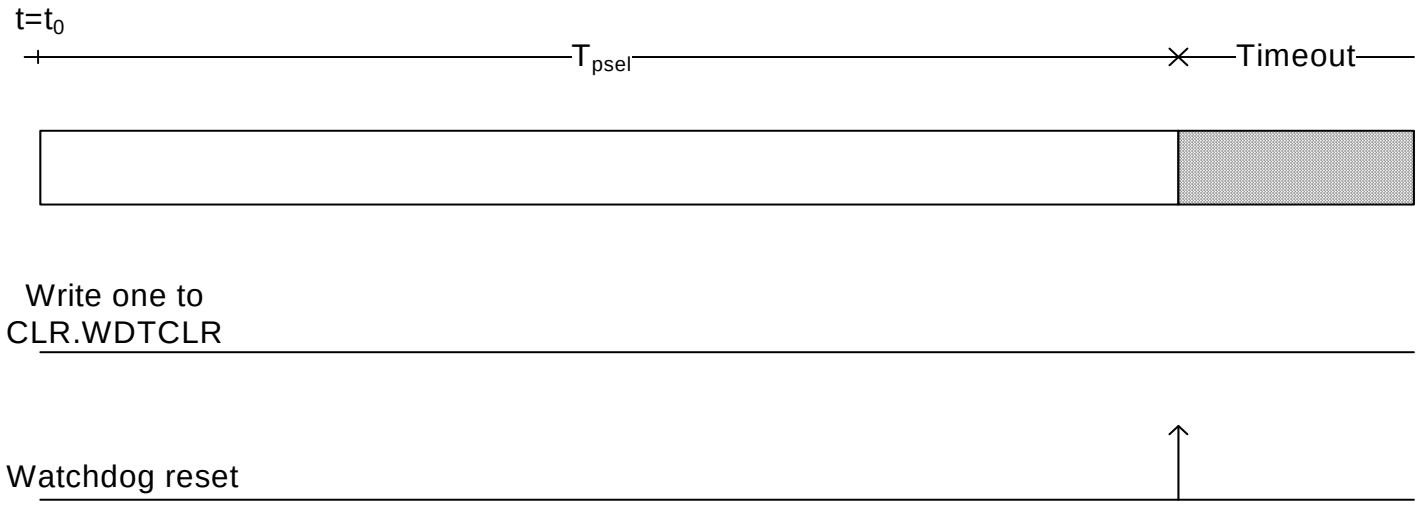


Figure 20-3. Basic Mode WDT Timing Diagram, No Clear within T_{psel}



20.5.1.6 Watchdog Reset

A watchdog reset will result in a reset and the code will start executing from the boot vector, refer to Section 10. “Power Manager (PM)” on page 109 for details. If the Disable After Reset (DAR) bit in the CTRL Register is zero, the WDT counter will restart counting from zero when the watchdog reset is released.

If the CTRL.DAR bit is one the WDT will be disabled after a watchdog reset. Only the CTRL.EN bit will be changed after the watchdog reset. However, if WDTAUTO fuse is configured to enable the WDT after a watchdog reset, and the CTRL.FCD bit is zero, writing a one to the CTRL.DAR bit will have no effect.

20.5.2 Window Mode

The window mode can protect against tight loops of runaway code. This is obtained by adding a ban period to timeout period. During the ban period clearing the WDT counter is not allowed.

If the WDT Mode (MODE) bit in the CTRL Register is one, the WDT is in window mode. Note that the CTRL.MODE bit can only be changed when the WDT is disabled (CTRL.EN=0).

The PSEL and Time Ban Prescale Select (TBAN) fields in the CTRL Register selects the WDT timeout period

$$T_{timeout} = T_{tban} + T_{psel} = (2^{(TBAN+1)} + 2^{(PSEL+1)}) / f_{clk\_cnt}$$

where T_{tban} sets the time period when clearing the WDT counter by writing to the CLR.WDTCLR bit is not allowed. Doing so will result in a watchdog reset, the device will receive a reset and the code will start executing form the boot vector, see Figure 20-5 on page 487. The WDT counter will be cleared.

Writing a one to the CLR.WDTCLR bit within the T_{psel} period will clear the WDT counter and the counter starts counting from zero ($t=t_0$), entering T_{tban}, see Figure 20-4 on page 486.

If the value in the CTRL Register is changed, the WDT counter will be cleared without a watchdog reset, regardless of if the value in the WDT counter and the TBAN value.

If the WDT counter reaches $T_{timeout}$, the counter will be cleared, the device will receive a reset and the code will start executing form the boot vector.

Figure 20-4. Window Mode WDT Timing Diagram

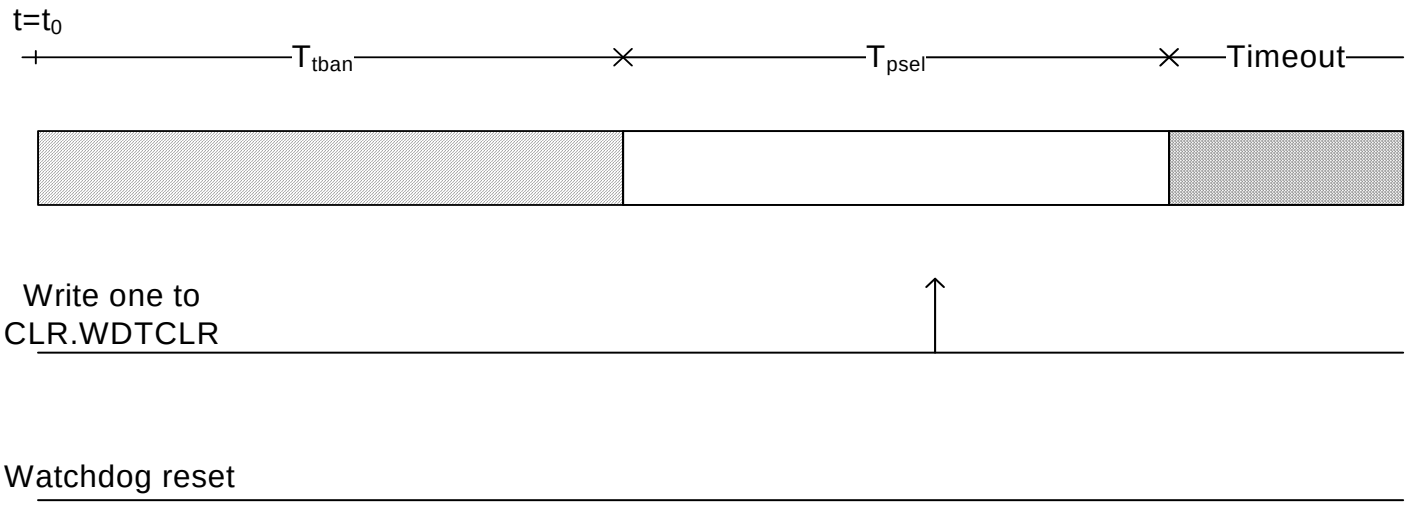
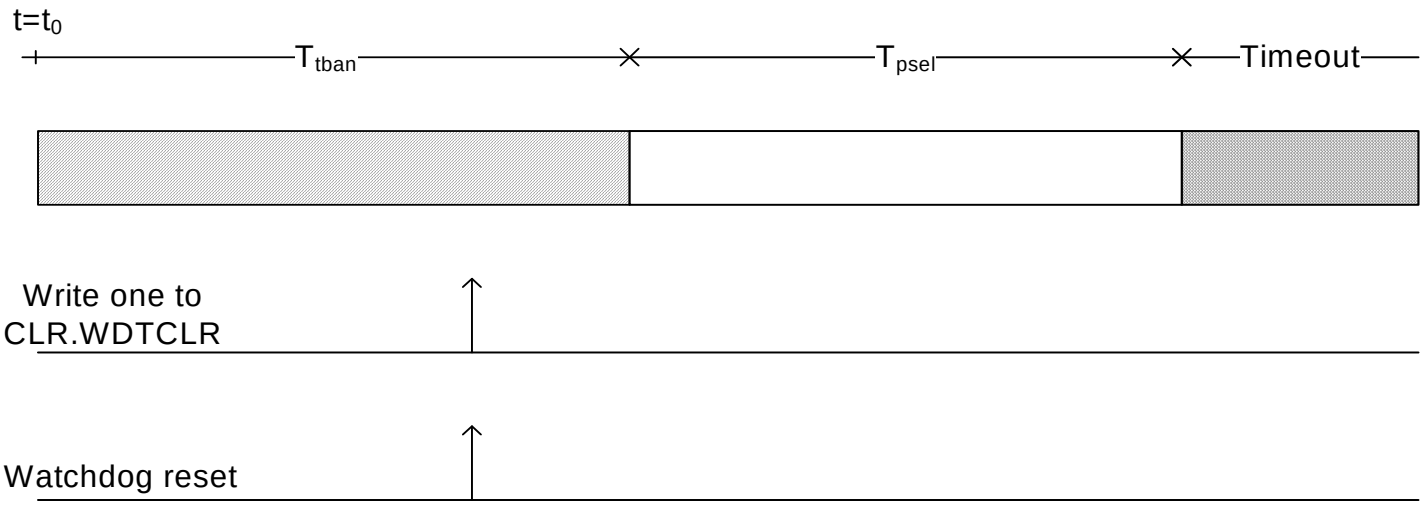


Figure 20-5. Window Mode WDT Timing Diagram, clearing within T_{tban}, resulting in watchdog reset.



20.5.3 Interrupt Mode

In interrupt mode, the WDT can generate an interrupt request when the WDT counter times out. Interrupt mode is enabled by writing a one to the Interrupt Mode bit in the CTRL register (CTRL.IM). When interrupt mode is enabled, the Watchdog Interrupt bit in the Interrupt Status Register (ISR.WINT) is set when the WDT counter times out. An interrupt request will be generated if the Watchdog Interrupt bit in the Interrupt Mask Register (IMR.WINT) is set, see Figure 20-6. IMR.WINT is set by writing a one to the Watchdog Interrupt bit in the Interrupt Enable Register (IER.WINT), and cleared by writing a one to the Watchdog Interrupt bit in the Interrupt Disable Register (IDR.WINT). The interrupt request remains active until ISR.WINT is cleared by writing a one to WINT in the Interrupt Clear Register (ICR.WINT).

If the Watchdog interrupt is not cleared before the next WDT counter timeout, a WDT reset is generated, see Figure 20-7.

Note that ISR.WINT will not be cleared by the WDT reset. If ISR.WINT is not cleared manually after a WDT reset, a new WDT reset will be issued on the first WDT counter timeout after the reset.

Interrupt mode can be enabled in both normal and window mode.

Figure 20-6. Interrupt Mode WDT Timing Diagram

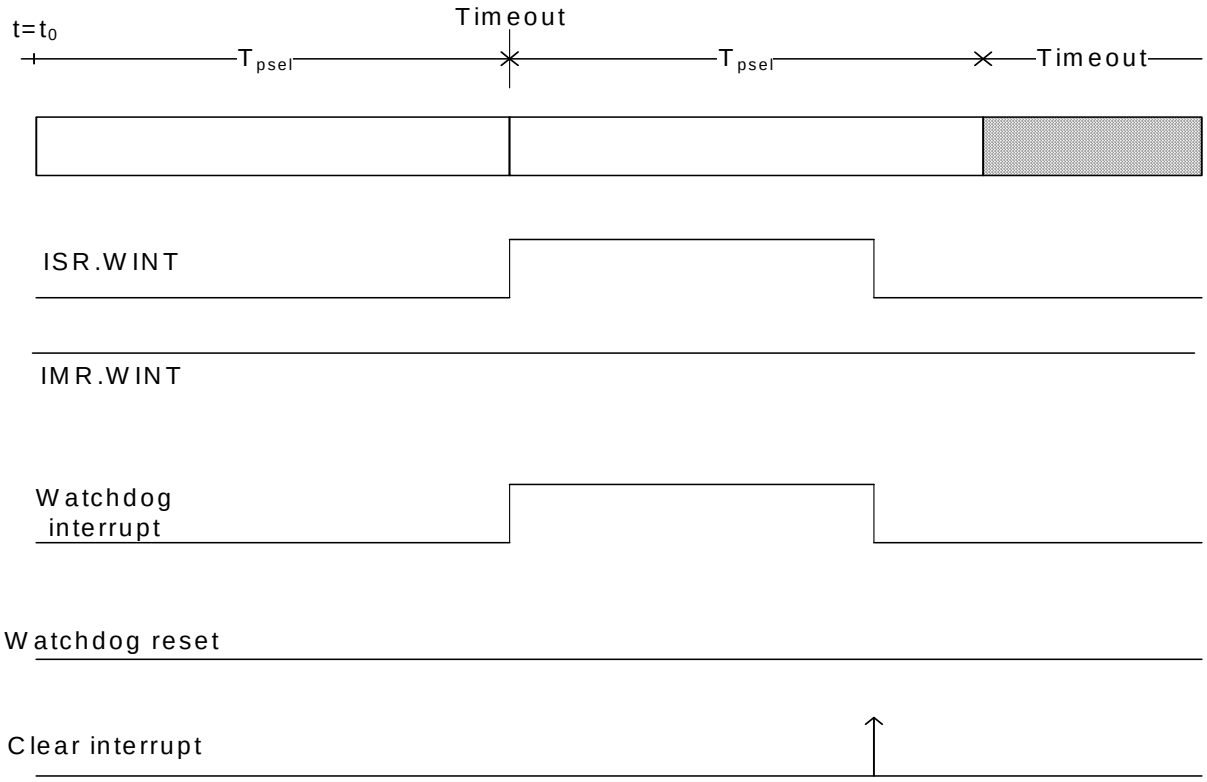
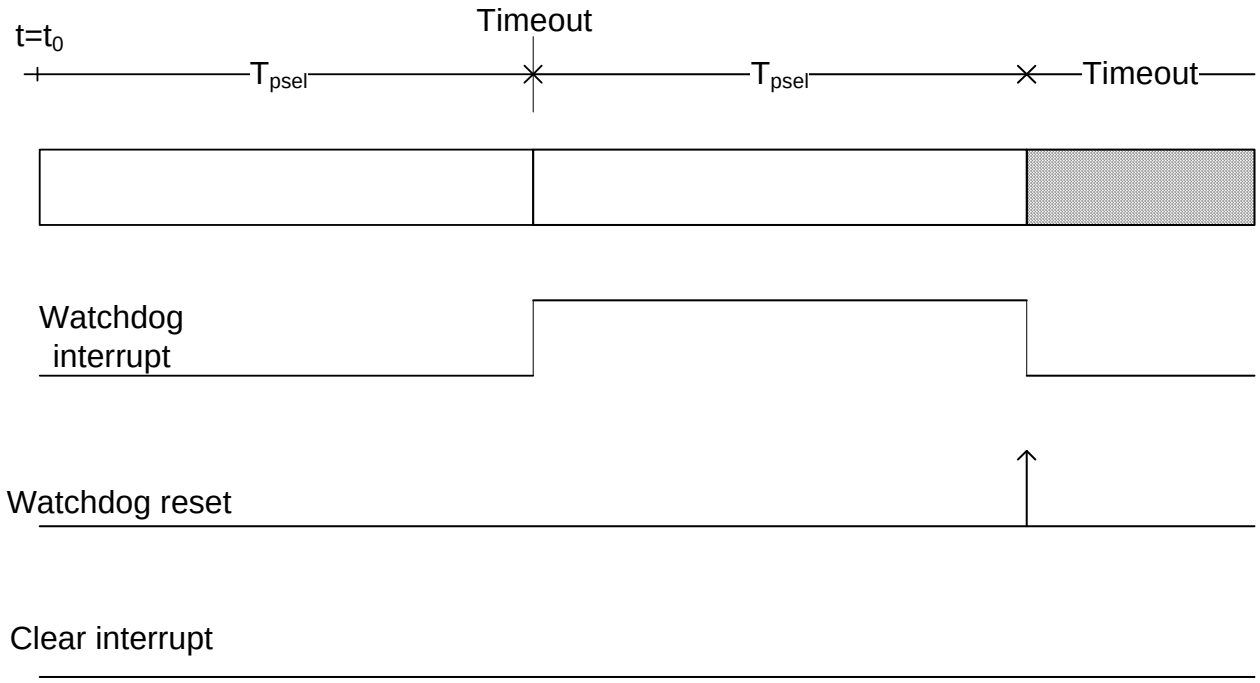


Figure 20-7. Interrupt Mode WDT Timing Diagram, not clearing the interrupt.



20.5.4 Disabling the WDT

The WDT is disabled by writing a zero to the CTRL.EN bit. When disabling the WDT no other bits in the CTRL Register should be changed until the CTRL.EN bit reads back as zero. If the CTRL.CEN bit is written to zero, the CTRL.EN bit will never read back as zero if changing the value from one to zero.

20.5.5 Flash Calibration

The WDT can be enabled at reset. This is controlled by the WDTAUTO fuse. The WDT will be set in basic mode, RCSYS is set as source for CLK_CNT, and PSEL will be set to a value giving T_{psel} above 100 ms. Refer to the Fuse Settings chapter for details about WDTAUTO and how to program the fuses.

If the Flash Calibration Done (FCD) bit in the CTRL Register is zero at a watchdog reset the flash calibration will be redone, and the CTRL.FCD bit will be set when the calibration is done. If CTRL.FCD is one at a watchdog reset, the configuration of the WDT will not be changed during flash calibration. After any other reset the flash calibration will always be done, and the CTRL.FCD bit will be set when the calibration is done.

20.5.6 Special Considerations

Care must be taken when selecting the PSEL/TBAN values so that the timeout period is greater than the startup time of the chip. Otherwise a watchdog reset will reset the chip before any code has been run. This can also be avoided by writing the CTRL.DAR bit to one when configuring the WDT.

If the Store Final Value (SFV) bit in the CTRL Register is one, the CTRL Register is locked for further write accesses. All writes to the CTRL Register will be ignored. Once the CTRL Register is locked, it can only be unlocked by a reset (e.g. POR, OCD, and WDT).

The CTRL.MODE bit can only be changed when the WDT is disabled (CTRL.EN=0).

20.5.7 Interrupts

The WDT has one interrupt:

- WINT: Watchdog Interrupt, set if WDT is in Interrupt Mode and the Watchdog timer times out.

20.6 User Interface

Table 20-1. WDT Register Memory Map

Offset	Register	Register Name	Access	Reset
0x000	Control Register	CTRL	Read/Write	0x00010080
0x004	Clear Register	CLR	Write-only	0x00000000
0x008	Status Register	SR	Read-only	0x00000003
0x00C	Interrupt Enable Register	IER	Write-only	0x00000000
0x010	Interrupt Disable Register	IDR	Write-only	0x00000000
0x014	Interrupt Mask Register	IMR	Read-only	0x00000000
0x018	Interrupt Status Register	ISR	Read-only	0x00000000
0x01C	Interrupt Clear Register	ICR	Write-only	0x00000000
0x3FC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value for this register is device specific. Refer to the Module Configuration section at the end of this chapter.

20.6.1 Control Register

Name: CTRL

Access Type: Read/Write

Offset: 0x000

Reset Value: 0x00010080

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
-	TBAN					CSSEL	CEN

15	14	13	12	11	10	9	8
-	-	-	PSEL				

7	6	5	4	3	2	1	0
FCD	-	-	IM	SFV	MODE	DAR	EN

- **KEY**
  This field must be written twice, first with key value 0x55, then 0xAA, for a write operation to be effective. This field always reads as zero.
- **TBAN: Time Ban Prescale Select**
  Counter bit TBAN is used as watchdog “banned” time frame. In this time frame clearing the WDT timer is forbidden, otherwise a watchdog reset is generated and the WDT timer is cleared.
- **CSSEL: Clock Source Select**
  0: Select the system RC oscillator (RCSYS) as clock source.
  1: Select the 32kHz crystal oscillator (OSC32K) as clock source.
- **CEN: Clock Enable**
  0: The WDT clock is disabled.
  1: The WDT clock is enabled.
- **PSEL: Time Out Prescale Select**
  Counter bit PSEL is used as watchdog timeout period.
- **IM: Interrupt Mode**
  0: Interrupt Mode is disabled.
  1: Interrupt Mode is enabled.
- **FCD: Flash Calibration Done**
  This bit is set after any reset.
  0: The flash calibration will be redone after a watchdog reset.
  1: The flash calibration will not be redone after a watchdog reset.
- **SFV: WDT Control Register Store Final Value**
  0: WDT Control Register is not locked.
  1: WDT Control Register is locked.
  Once locked, the Control Register can not be re-written, only a reset unlocks the SFV bit.

- **MODE: WDT Mode**
  0: The WDT is in basic mode, only PSEL time is used.
  1: The WDT is in window mode. Total timeout period is now TBAN+PSEL.
  Writing to this bit when the WDT is enabled has no effect.
- **DAR: WDT Disable After Reset**
  0: After a watchdog reset, the WDT will still be enabled.
  1: After a watchdog reset, the WDT will be disabled.
- **EN: WDT Enable**
  0: WDT is disabled.
  1: WDT is enabled.
  After writing to this bit the read back value will not change until the WDT is enabled/disabled. This due to internal synchronization.

20.6.2 Clear Register

Name: CLR

Access Type: Write-only

Offset: 0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	WDTCLR

When the Watchdog Timer is enabled, this Register must be periodically written within the window time frame or within the watchdog timeout period, to prevent a watchdog reset.

- **KEY**
  This field must be written twice, first with key value 0x55, then 0xAA, for a write operation to be effective.
- **WDTCLR: Watchdog Clear**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the WDT counter.

20.6.3 Status Register

Name: SR

Access Type: Read-only

Offset: 0x008

Reset Value: 0x00000003

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

- **CLEARED: WDT Counter Cleared**
  This bit is cleared when writing a one to the CLR.WDTCLR bit.
  This bit is set when clearing the WDT counter is done.
- **WINDOW: Within Window**
  This bit is cleared when the WDT counter is inside the TBAN period.
  This bit is set when the WDT counter is inside the PSEL period.

20.6.4 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x00C

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

20.6.5 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	WINT	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

20.6.6 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x014

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	WINT	-	-

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

A bit in this register is cleared when the corresponding bit in IDR is written to one.

A bit in this register is set when the corresponding bit in IER is written to one.

20.6.7 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	WINT	-	-

0: The corresponding interrupt is cleared.

1: The corresponding interrupt is pending.

This bit is cleared when the corresponding bit in ICR is written to one.

This bit is set when the corresponding interrupt occurs.

20.6.8 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x01C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	WINT	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR and the corresponding interrupt request.

20.6.9 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x3FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

20.7 Module Configuration

The specific configuration for each WDT instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 20-2. WDT Clocks

Clock Name	Description
CLK_WDT	Clock for the WDT bus interface

Table 20-3. Register Reset Values

Register	Reset Value
VERSION	0x00000501

21. External Interrupt Controller (EIC)

Rev: 3.0.2.0

21.1 Features

- **Dedicated interrupt request for each interrupt**
- **Individually maskable interrupts**
- **Interrupt on rising or falling edge**
- **Interrupt on high or low level**
- **Asynchronous interrupts for sleep modes without clock**
- **Filtering of interrupt lines**
- **Non-Maskable NMI interrupt**

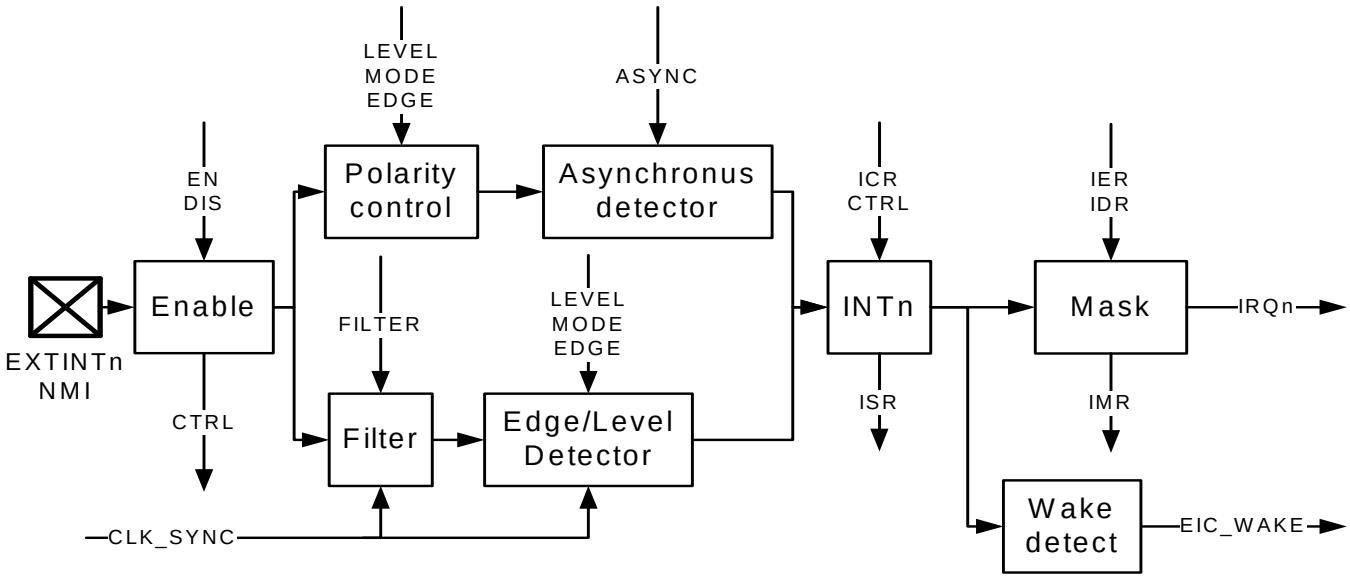
21.2 Overview

The External Interrupt Controller (EIC) allows pins to be configured as external interrupts. Each external interrupt has its own interrupt request and can be individually masked. Each external interrupt can generate an interrupt on rising or falling edge, or high or low level. Every interrupt input has a configurable filter to remove spikes from the interrupt source. Every interrupt pin can also be configured to be asynchronous in order to wake up the part from sleep modes where the CLK_SYNC clock has been disabled.

A Non-Maskable Interrupt (NMI) is also supported. This has the same properties as the other external interrupts, but is connected to the NMI request of the CPU, enabling it to interrupt any other interrupt mode.

21.3 Block Diagram

Figure 21-1. EIC Block Diagram



21.4 I/O Lines Description

Table 21-1. I/O Lines Description

Pin Name	Pin Description	Type
NMI	Non-Maskable Interrupt	Input
EXTINTn	External Interrupt	Input

21.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

21.5.1 I/O Lines

The external interrupt pins (EXTINTn and NMI) may be multiplexed with I/O Controller lines. The programmer must first program the I/O Controller to assign the desired EIC pins to their peripheral function. If I/O lines of the EIC are not used by the application, they can be used for other purposes by the I/O Controller.

It is only required to enable the EIC inputs actually in use. If an application requires two external interrupts, then only two I/O lines will be assigned to EIC inputs.

21.5.2 Power Management

All interrupts are available in all sleep modes as long as the EIC module is powered. However, in sleep modes where CLK_SYNC is stopped, the interrupt must be configured to asynchronous mode.

21.5.3 Clocks

The clock for the EIC bus interface (CLK_EIC) is generated by the Power Manager. This clock is enabled at reset, and can be disabled in the Power Manager.

The filter and synchronous edge/level detector runs on a clock which is stopped in any of the sleep modes where the system RC oscillator (RCSYS) is not running. This clock is referred to as CLK_SYNC.

21.5.4 Interrupts

The external interrupt request lines are connected to the NVIC. Using the external interrupts requires the NVIC to be programmed first.

Using the Non-Maskable Interrupt does not require the NVIC to be programmed.

21.5.5 Debug Operation

When an external debugger forces the CPU into debug mode, the EIC continues normal operation. If the EIC is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

21.6 Functional Description

21.6.1 External Interrupts

The external interrupts are not enabled by default, allowing the proper interrupt vectors to be set up by the CPU before the interrupts are enabled.

Each external interrupt INTn can be configured to produce an interrupt on rising or falling edge, or high or low level. External interrupts are configured by the MODE, EDGE, and LEVEL registers. Each interrupt has a bit INTn in each of these registers. Writing a zero to the INTn bit in the MODE register enables edge triggered interrupts, while writing a one to the bit enables level triggered interrupts.

If INTn is configured as an edge triggered interrupt, writing a zero to the INTn bit in the EDGE register will cause the interrupt to be triggered on a falling edge on EXTINTn, while writing a one to the bit will cause the interrupt to be triggered on a rising edge on EXTINTn.

If INTn is configured as a level triggered interrupt, writing a zero to the INTn bit in the LEVEL register will cause the interrupt to be triggered on a low level on EXTINTn, while writing a one to the bit will cause the interrupt to be triggered on a high level on EXTINTn.

Each interrupt has a corresponding bit in each of the interrupt control and status registers. Writing a one to the INTn bit in the Interrupt Enable Register (IER) enables the external interrupt from pin EXTINTn to propagate from the EIC to the NVIC, while writing a one to INTn bit in the Interrupt Disable Register (IDR) disables this propagation. The Interrupt Mask Register (IMR) can be read to check which interrupts are enabled. When an interrupt triggers, the corresponding bit in the Interrupt Status Register (ISR) will be set. This bit remains set until a one is written to the corresponding bit in the Interrupt Clear Register (ICR) or the interrupt is disabled.

Writing a one to the INTn bit in the Enable Register (EN) enables the external interrupt on pin EXTINTn, while writing a one to INTn bit in the Disable Register (DIS) disables the external interrupt. The Control Register (CTRL) can be read to check which interrupts are enabled. If a bit in the CTRL register is set, but the corresponding bit in IMR is not set, an interrupt will not propagate to the NVIC. However, the corresponding bit in ISR will be set, and EIC_WAKE will be set. Note that an external interrupt should not be enabled before it has been configured correctly.

If the CTRL.INTn bit is zero, the corresponding bit in ISR will always be zero. Disabling an external interrupt by writing a one to the DIS.INTn bit will clear the corresponding bit in ISR.

Refer to the Module Configuration section for the number of external interrupts.

21.6.2 Synchronization and Filtering of External Interrupts

In synchronous mode the pin value of the EXTINTn pin is synchronized to CLK_SYNC, so spikes shorter than one CLK_SYNC cycle are not guaranteed to produce an interrupt. The synchronization of the EXTINTn to CLK_SYNC will delay the propagation of the interrupt to the NVIC by two cycles of CLK_SYNC, see Figure 21-2 and Figure 21-3 for examples (FILTER off).

It is also possible to apply a filter on EXTINTn by writing a one to the INTn bit in the FILTER register. This filter is a majority voter, if the condition for an interrupt is true for more than one of the latest three cycles of CLK_SYNC the interrupt will be set. This will additionally delay the propagation of the interrupt to the NVIC by one or two cycles of CLK_SYNC, see Figure 21-2 and Figure 21-3 for examples (FILTER on).

Figure 21-2. Timing Diagram, Synchronous Interrupts, High Level or Rising Edge

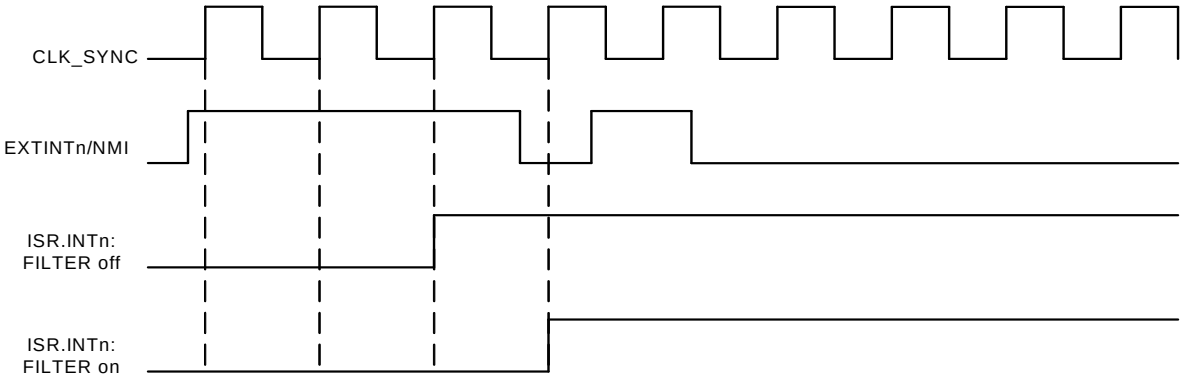
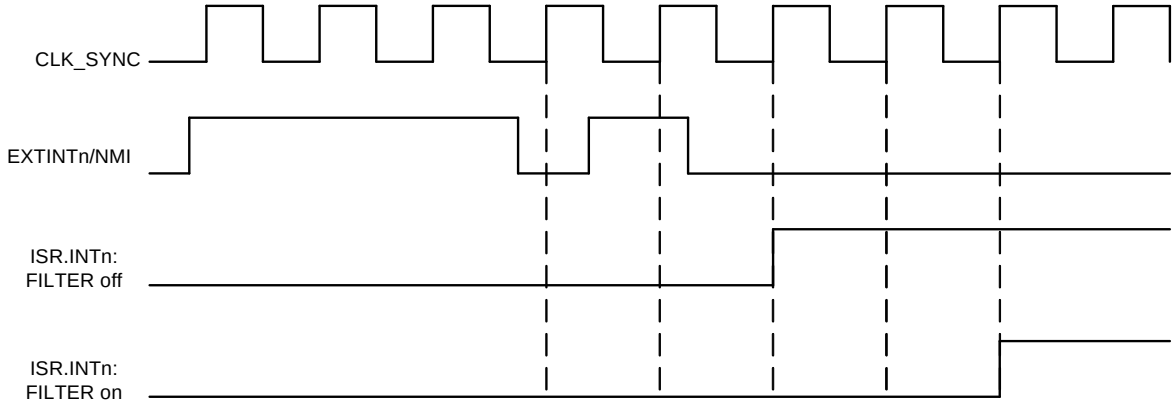


Figure 21-3. Timing Diagram, Synchronous Interrupts, Low Level or Falling Edge



21.6.3 Non-Maskable Interrupt

The NMI supports the same features as the external interrupts, and is accessed through the same registers. The description in Section 21.6.1 should be followed, accessing the NMI bit instead of the INTn bits.

The NMI is non-maskable within the CPU in the sense that it can interrupt any other execution mode. Still, as for the other external interrupts, the actual NMI input can be enabled and disabled by accessing the registers in the EIC.

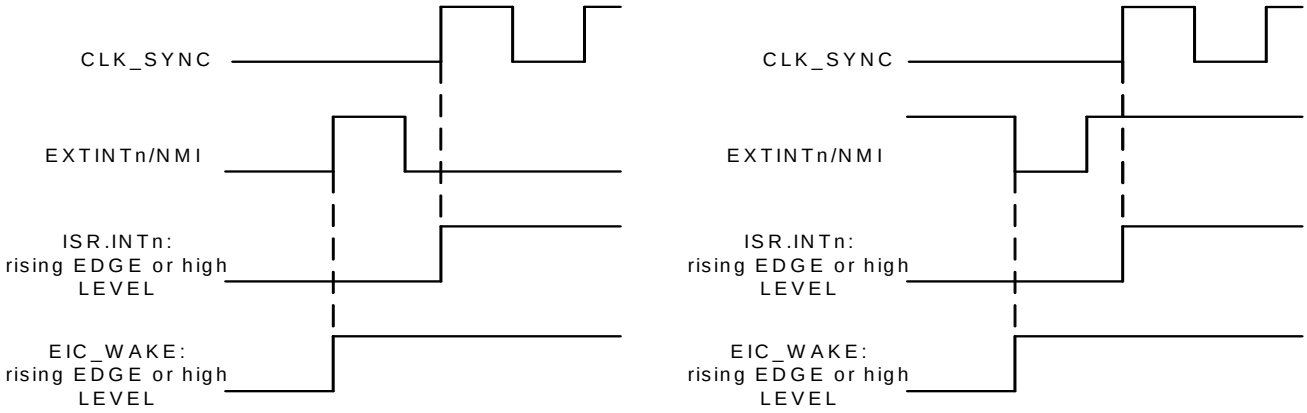
21.6.4 Asynchronous Interrupts

Each external interrupt can be made asynchronous by writing a one to INTn in the ASYNC register. This will route the interrupt signal through the asynchronous path of the module. All edge interrupts will be interpreted as level interrupts and the filter is disabled. If an interrupt is configured as edge triggered interrupt in asynchronous mode, a zero in EDGE.INTn will be interpreted as low level, and a one in EDGE.INTn will be interpreted as high level.

EIC_WAKE will be set immediately after the source triggers the interrupt, while the corresponding bit in ISR and the interrupt to the NVIC will be set on the next rising edge of CLK_SYNC. Refer to Figure 21-4 on page 506 for details.

When CLK_SYNC is stopped only asynchronous interrupts remain active, and any short spike on this interrupt will wake up the device. EIC_WAKE will restart CLK_SYNC and ISR will be updated on the first rising edge of CLK_SYNC.

Figure 21-4. Timing Diagram, Asynchronous Interrupts



21.6.5 Wakeup

The external interrupts can be used to wake up the part from power save modes if the corresponding bit in IMR is one.

21.7 User Interface

Table 21-2. EIC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x000	Interrupt Enable Register	IER	Write-only	0x00000000
0x004	Interrupt Disable Register	IDR	Write-only	0x00000000
0x008	Interrupt Mask Register	IMR	Read-only	0x00000000
0x00C	Interrupt Status Register	ISR	Read-only	0x00000000
0x010	Interrupt Clear Register	ICR	Write-only	0x00000000
0x014	Mode Register	MODE	Read/Write	0x00000000
0x018	Edge Register	EDGE	Read/Write	0x00000000
0x01C	Level Register	LEVEL	Read/Write	0x00000000
0x020	Filter Register	FILTER	Read/Write	0x00000000
0x024	Test Register	TEST	Read/Write	0x00000000
0x028	Asynchronous Register	ASYNC	Read/Write	0x00000000
0x030	Enable Register	EN	Write-only	0x00000000
0x034	Disable Register	DIS	Write-only	0x00000000
0x038	Control Register	CTRL	Read-only	0x00000000
0x3FC	Version Register	VERSION	Read-only	- (1)

Note: 1. The reset value is device specific. Refer to the Module Configuration section at the end of this chapter.

21.7.1 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the corresponding bit in IMR.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Wrting a one to this bit will set the corresponding bit in IMR.

21.7.2 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in IMR.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in IMR.

21.7.3 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x008

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The corresponding interrupt is disabled.
  1: The corresponding interrupt is enabled.
  This bit is cleared when the corresponding bit in IDR is written to one.
  This bit is set when the corresponding bit in IER is written to one.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt is disabled.
  1: The Non-Maskable Interrupt is enabled.
  This bit is cleared when the corresponding bit in IDR is written to one.
  This bit is set when the corresponding bit in IER is written to one.

21.7.4 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x00C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: An interrupt event has not occurred.
  1: An interrupt event has occurred.
  This bit is cleared by writing a one to the corresponding bit in ICR.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: An interrupt event has not occurred.
  1: An interrupt event has occurred.
  This bit is cleared by writing a one to the corresponding bit in ICR.

21.7.5 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in ISR.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in ISR.

21.7.6 Mode Register

Name: MODE

Access Type: Read/Write

Offset: 0x014

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The external interrupt is edge triggered.
  1: The external interrupt is level triggered.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt is edge triggered.
  1: The Non-Maskable Interrupt is level triggered.

21.7.7 Edge Register

Name: EDGE

Access Type: Read/Write

Offset: 0x018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The external interrupt triggers on falling edge.
  1: The external interrupt triggers on rising edge.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt triggers on falling edge.
  1: The Non-Maskable Interrupt triggers on rising edge.

21.7.8 Level Register

Name: LEVEL

Access Type: Read/Write

Offset: 0x01C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The external interrupt triggers on low level.
  1: The external interrupt triggers on high level.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt triggers on low level.
  1: The Non-Maskable Interrupt triggers on high level.

21.7.9 Filter Register

Name: FILTER

Access Type: Read/Write

Offset: 0x020

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The external interrupt is not filtered.
  1: The external interrupt is filtered.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt is not filtered.
  1: The Non-Maskable Interrupt is filtered.

21.7.10 Test Register

Name: TEST

Access Type: Read/Write

Offset: 0x024

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
TESTEN	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **TESTEN: Test Enable**
  0: This bit disables external interrupt test mode.
  1: This bit enables external interrupt test mode.
- **INTn: External Interrupt n**
  Writing a zero to this bit will set the input value to INTn to zero, if test mode is enabled.
  Writing a one to this bit will set the input value to INTn to one, if test mode is enabled.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit will set the input value to NMI to zero, if test mode is enabled.
  Writing a one to this bit will set the input value to NMI to one, if test mode is enabled.
  If TESTEN is 1, the value written to this bit will be the value to the interrupt detector and the value on the pad will be ignored.

21.7.11 Asynchronous Register

Name: ASYNC

Access Type: Read/Write

Offset: 0x028

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The external interrupt is synchronized to CLK_SYNC.
  1: The external interrupt is asynchronous.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt is synchronized to CLK_SYNC.
  1: The Non-Maskable Interrupt is asynchronous.

21.7.12 Enable Register

Name: EN

Access Type: Write-only

Offset: 0x030

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will enable the corresponding external interrupt.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will enable the Non-Maskable Interrupt.

21.7.13 Disable Register

Name: DIS

Access Type: Write-only

Offset: 0x034

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will disable the corresponding external interrupt.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will disable the Non-Maskable Interrupt.

21.7.14 Control Register

Name: CTRL

Access Type: Read-only

Offset: 0x038

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	INT30	INT29	INT28	INT27	INT26	INT25	INT24
23	22	21	20	19	18	17	16
INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
15	14	13	12	11	10	9	8
INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8
7	6	5	4	3	2	1	0
INT7	INT6	INT5	INT4	INT3	INT2	INT1	NMI

- **INTn: External Interrupt n**
  0: The corresponding external interrupt is disabled.
  1: The corresponding external interrupt is enabled.
  Refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt is disabled.
  1: The Non-Maskable Interrupt is enabled.

21.7.15 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x3FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VERSION: Version number**
  Version number of the module. No functionality associated.

21.8 Module Configuration

The specific configuration for each EIC instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 21-3. EIC Configuration

Feature	EIC
Number of external interrupts, including NMI	9

Table 21-4. EIC Clocks

Clock Name	Description
CLK_EIC	Clock for the EIC bus interface

Table 21-5. Register Reset Values

Register	Reset Value
VERSION	0x00000302

22. Frequency Meter (FREQM)

Rev: 3.1.1.1

22.1 Features

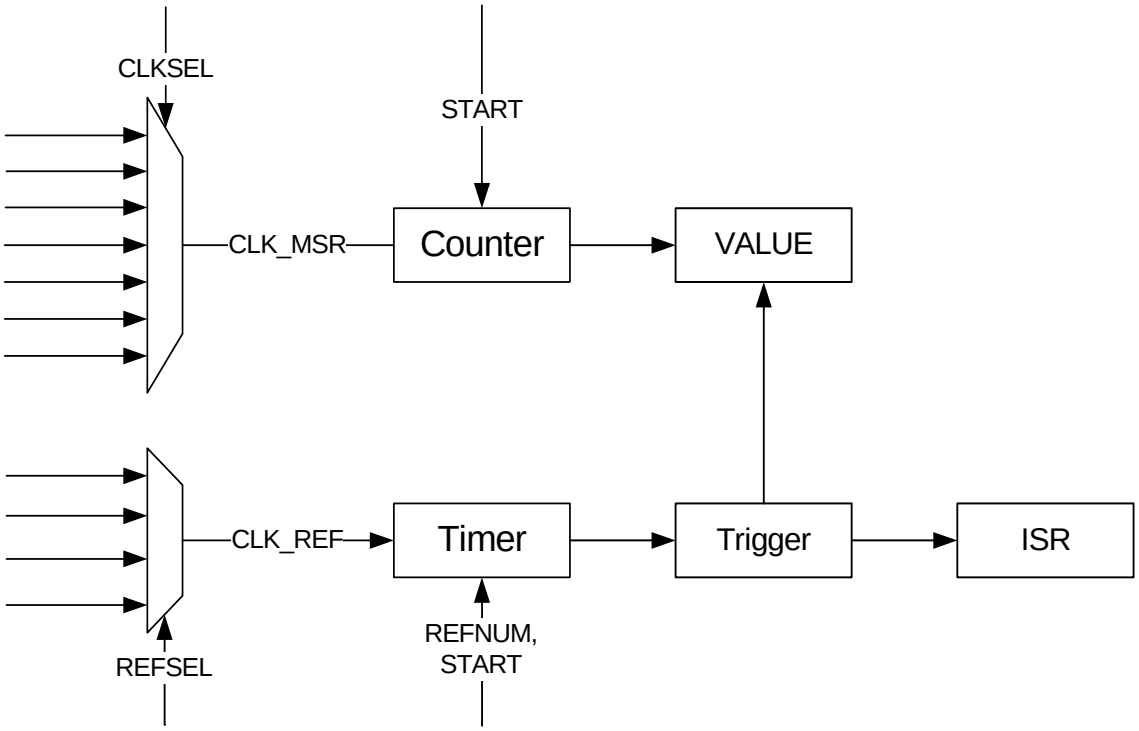
- **Accurately measures a clock frequency**
- **Selectable reference clock**
- **A selectable clock can be measured**
- **Ratio can be measured with 24-bit accuracy**

22.2 Overview

The Frequency Meter (FREQM) can be used to accurately measure the frequency of a clock by comparing it to a known reference clock.

22.3 Block Diagram

Figure 22-1. Frequency Meter Block Diagram



22.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

22.4.1 Power Management

The device can enter a sleep mode while a measurement is ongoing. However, make sure that neither CLK_MSR nor CLK_REF is stopped in the actual sleep mode.

FREQM interrupts can wake up the device from sleep modes when the measurement is done, but only from sleep modes where CLK_FREQM is running. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

22.4.2 Clocks

The clock for the FREQM bus interface (CLK_FREQM) is generated by the Power Manager. It is recommended to disable the FREQM before disabling the clock, to avoid freezing the FREQM ia an undefined state.

A set of clocks can be selected as reference (CLK_REF) and another set of clocks can be selected for measurement (CLK_MSR). Refer to the CLKSEL and REFSEL tables in the Module Configuration section for details.

22.4.3 Debug Operation

When an external debugger forces the CPU into debug mode, the FREQM continues normal operation. If the FREQM is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

22.4.4 Interrupts

The FREQM interrupt request line is connected to the internal source of the NVIC. Using the FREQM interrupt requires the NVIC to be programmed first.

22.5 Functional Description

The FREQM accuratly measures the frequency of a clock by comparing the frequency to a known frequency:

$$f_{CLK\_MSR} = (VALUE/REFNUM)*f_{CLK\_REF}$$

22.5.1 Reference Clock

The Reference Clock Selection (REFSEL) field in the Mode Register (MODE) selects the clock source for CLK_REF. The reference clock is enabled by writing a one to the Reference Clock Enable (REFCEN) bit in the Mode Register. This clock should have a known frequency.

CLK_REF needs to be disabled before switching to another clock. The RCLKBUSY bit in the Status Register (SR) indicates whether the clock is busy or not. This bit is set when the MODE.REFCEN bit is written.

To change CLK_REF:

- Write a zero to the MODE.REFCEN bit to disable the clock, without changing the other bits/fields in the Mode Register.
- Wait until the SR.RCLKBUSY bit reads as zero.
- Change the MODE.REFSEL field.
- Write a one to the MODE.REFCEN bit to enable the clock, without changing the other bits/fields in the Mode Register.
- Wait until the SR.RCLKBUSY bit reads as zero.

To enable CLK_REF:

- Write the correct value to the MODE.REFSEL field.
- Write a one to the MODE.REFCEN to enable the clock, without changing the other bits/fields in the Mode Register.

- Wait until the SR.RCLKBUSY bit reads as zero.

To disable CLK_REF:

- Write a zero to the MODE.REFCEN to disable he clock, without changing the other bits/fields in the Mode register.
- Wait until the SR.RCLKBUSY bit reads as zero.

22.5.1.1 Cautionary Note

Note that if clock selected as source for CLK_REF is stopped during a measurement, this will not be detected by the FREQM. The BUSY bit in the STATUS register will never be cleared, and the DONE interrupt will never be triggered. If the clock selected as soruce for CLK_REF is stopped, it will not be possible to change the source for the reference clock as long as the selected source is not running.

22.5.2 Measurement

In the Mode Register the Clock Source Selection (CLKSEL) field selects CLK_MSR and the Number of Reference Clock Cycles (REFNUM) field selects the duration of the measurement. The duration is given in number of CLK_REF periodes.

Writing a one to the START bit in the Control Register (CTRL) starts the measurement. The BUSY bit in SR is cleared when the measurement is done.

The result of the measurement can be read from the Value Register (VALUE). The frequency of the measured clock CLK_MSR is then:

$$f_{CLK\_MSR} = (VALUE/REFNUM)*f_{CLK\_REF}$$

22.5.3 Interrupts

The FREQM has two interrupt sources:

- DONE: A frequency measurement is done
- RCLKRDY: The reference clock is ready

These will generate an interrupt request if the corresponding bit in the Interrupt Mask Register (IMR) is set. The interrupt sources are ORed together to form one interrupt request. The FREQM will generate an interrupt request if at least one of the bits in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER) and cleared by writing a one to this bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in the Interrupt Status Register (ISR) is cleared by writing a one to this bit in the Interrupt Clear Register (ICR). Because all the interrupt sources are ORed together, the interrupt request from the FREQM will remain active until all the bits in ISR are cleared.

22.6 User Interface

Table 22-1. FREQM Register Memory Map

Offset	Register	Register Name	Access	Reset
0x000	Control Register	CTRL	Write-only	0x00000000
0x004	Mode Register	MODE	Read/Write	0x00000000
0x008	Status Register	STATUS	Read-only	0x00000000
0x00C	Value Register	VALUE	Read-only	0x00000000
0x010	Interrupt Enable Register	IER	Write-only	0x00000000
0x014	Interrupt Disable Register	IDR	Write-only	0x00000000
0x018	Interrupt Mask Register	IMR	Read-only	0x00000000
0x01C	Interrupt Status Register	ISR	Read-only	0x00000000
0x020	Interrupt Clear Register	ICR	Write-only	0x00000000
0x3FC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset value for this register is device specific. Refer to the Module Configuration section at the end of this chapter.

22.6.1 Control Register

Name: CTRL

Access Type: Write-only

Offset: 0x000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	START

- **START**

  Writing a zero to this bit has no effect.

  Writing a one to this bit will start a measurement.

22.6.2 Mode Register

Name: MODE

Access Type: Read/Write

Offset: 0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
REFCEN	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	CLKSEL				

15	14	13	12	11	10	9	8
REFNUM							

7	6	5	4	3	2	1	0
-	-	-	-	-	REFSEL		

- **REFCEN: Reference Clock Enable**
  0: The reference clock is disabled
  1: The reference clock is enabled
- **CLKSEL: Clock Source Selection**
  Selects the source for CLK_MSR. See table in Module Configuration chapter for details.
- **REFNUM: Number of Reference Clock Cycles**
  Selects the duration of a measurement, given in number of CLK_REF cycles.
- **REFSEL: Reference Clock Selection**
  Selects the source for CLK_REF. See table in Module Configuration chapter for details.

22.6.3 Status Register

Name: STATUS

Access Type: Read-only

Offset: 0x008

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	RCLKBUSY	BUSY

- **RCLKBUSY: FREQM Reference Clock Status**
  0: The FREQM ref clk is ready, so a measurement can start.
  1: The FREQM ref clk is not ready, so a measurement should not be started.
- **BUSY: FREQM Status**
  0: The Frequency Meter is idle.
  1: Frequency measurement is on-going.

22.6.4 Value Register

Name: VALUE

Access Type: Read-only

Offset: 0x00C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
VALUE[23:16]							

15	14	13	12	11	10	9	8
VALUE[15:8]							

7	6	5	4	3	2	1	0
VALUE[7:0]							

- **VALUE:**
  Result from measurement.

22.6.5 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	RCLKRDY	DONE

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

22.6.6 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x014

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	RCLKRDY	DONE

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

22.6.7 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x018

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	RCLKRDY	DONE

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

22.6.8 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x01C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	RCLKRDY	DONE

0: The corresponding interrupt is cleared.

1: The corresponding interrupt is pending.

A bit in this register is set when the corresponding bit in STATUS has a one to zero transition.

A bit in this register is cleared when the corresponding bit in ICR is written to one.

22.6.9 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x020

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	RCLKRDY	DONE

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR and the corresponding interrupt request.

22.6.10 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x3FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

22.7 Module Configuration

The specific configuration for each FREQM instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 22-2. FREQM Clock Name

Clock Name	Description
CLK_FREQM	Clock for the FREQM bus interface
CLK_MSR	Measured clock
CLK_REF	Reference clock

Table 22-3. Register Reset Values

Register	Reset Value
VERSION	0x00000311

Table 22-4. Clock Sources for CLK_MSR

CLKSEL	Clock/Oscillator	Description
0	CLK_CPU	The clock the CPU runs on
1	CLK_AHB	High Speed Bus clock
2	CLK_APBA	Peripheral Bus A clock
3	CLK_APBB	Peripheral Bus B clock
4	CLK_APBC	Peripheral Bus C clock
5	CLK_APBD	Peripheral Bus D clock
6	OSC0	Output clock from Oscillator 0
7	CLK32K	32kHz Output clock from OSC32K or RC32K
8	RCSYS	Output clock from RCSYS Oscillator
9	DFLL0	Output clock from DFLL0
11-22	GCLK0-11	Generic clock 0 through 11
23	RC80M	Output clock from RC80M
24	RCFAST	Output clock from RCFAST
25	RC1M	Output clock from VREG RC1M
26	PLL	Output clock from PLL0
27-31	Reserved	

Table 22-5. Clock Sources for CLK_REF

REFSEL	Clock/Oscillator	Description
0	RCSYS	System RC oscillator clock
1	CLK32K	32kHz Output clock from OSC32K or RC32K
2	CLK1K	1kHz Output clock from OSC32K or RC32K
3	GCLK11	Generic clock 11
4-7	Reserved	

23. General-Purpose Input/Output Controller (GPIO)

Rev: 2.1.5.5

23.1 Features

- **Configurable pin-change, rising-edge, or falling-edge interrupt**
- **Configurable peripheral event generator**
- **Glitch filter providing rejection of pulses shorter than one clock cycle**
- **Input visibility and output control**
- **Multiplexing of peripheral functions on I/O pins**
- **Programmable internal pull-up resistor**
- **Programmable internal pull-down resistor**
- **Programmable output driver strength**
- **Programmable internal input Schmitt trigger**
- **Programmable output slew rate**
-

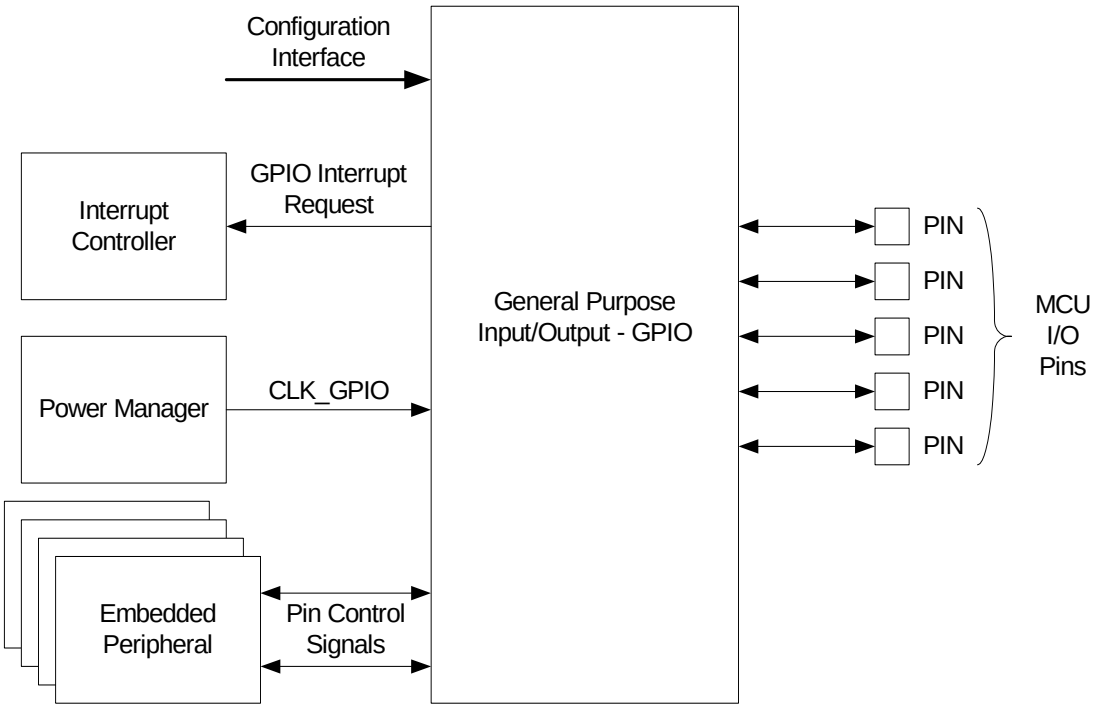
23.2 Overview

The General Purpose Input/Output Controller (GPIO) controls the I/O pins of the microcontroller. Each GPIO pin may be used as a general-purpose I/O or be assigned to a function of an embedded peripheral.

The GPIO is configured using the Peripheral Bus (PB).

23.3 Block Diagram

Figure 23-1. GPIO Block Diagram



23.4 I/O Lines Description

Pin Name	Description	Type
GPIOn	GPIO pin n	Digital

23.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

23.5.1 Power Management

If the CPU enters a sleep mode that disables clocks used by the GPIO, the GPIO will stop functioning and resume operation after the system wakes up from sleep mode.

If a peripheral function is configured for a GPIO pin, the peripheral will be able to control the GPIO pin even if the GPIO clock is stopped.

23.5.2 Clocks

The GPIO is connected to a Peripheral Bus clock (CLK_GPIO). This clock is generated by the Power Manager. CLK_GPIO is enabled at reset, and can be disabled by writing to the Power Manager. CLK_GPIO must be enabled in order to access the configuration registers of the GPIO or to use the GPIO interrupts. After configuring the GPIO, the CLK_GPIO can be disabled by writing to the Power Manager if interrupts are not used.

23.5.3 Interrupts

The GPIO interrupt request lines are connected to the NVIC. Using the GPIO interrupts requires the NVIC to be programmed first.

23.5.4 Peripheral Events

The GPIO peripheral events are connected via the Peripheral Event System. Refer to Section 31. ”Peripheral Event Controller (PEVC)” on page 845 for details.

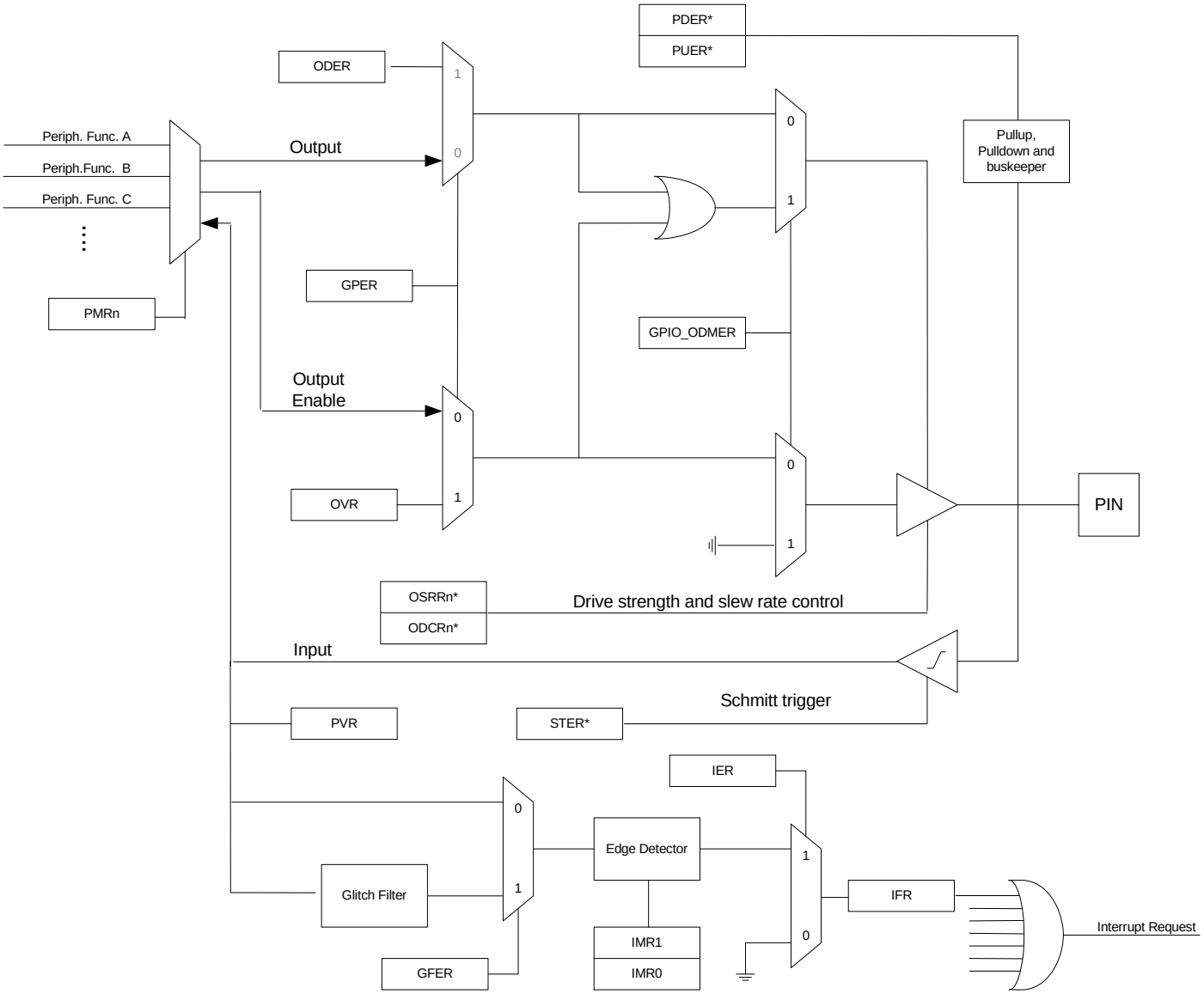
23.5.5 Debug Operation

When an external debugger forces the CPU into debug mode, the GPIO continues normal operation. If the GPIO is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

23.6 Functional Description

The GPIO controls the I/O pins of the microcontroller. The control logic associated with each pin is shown in the figure below.

Figure 23-2. Overview of the GPIO



*) Register value is overrided if a peripheral function that support this function is enabled

23.6.1 Basic Operation

23.6.1.1 Module Configuration

The GPIO user interface registers are organized into ports and each port controls 32 different GPIO pins. Most of the registers supports bit wise access operations such as set, clear and toggle in addition to the standard word access. For details regarding interface registers, refer to Section 23.7.

23.6.1.2 Available Features

The GPIO features implemented are device dependent, and not all functions are implemented on all pins. The user must refer to the Module Configuration section and the GPIO Function Multiplexing section in the Package and Pinout chapter for the device specific settings used in the ATSAM4L8/L4/L2.

Device specific settings includes:

- Number of GPIO pins
- Functions implemented on each pin
- Peripheral function(s) multiplexed on each GPIO pin
- Reset state of registers

23.6.1.3 Inputs

The level on each GPIO pin can be read through the Pin Value Register (PVR). This register indicates the level of the GPIO pins regardless of the pins being driven by the GPIO or by an external component. Note that due to power saving measures, the PVR register will only be updated when the corresponding bit in GPER is one or if an interrupt is enabled for the pin, i.e. IER is one for the corresponding pin.

23.6.1.4 Output Control

When the GPIO pin is assigned to a peripheral function, i.e. the corresponding bit in GPER is zero, the peripheral determines whether the pin is driven or not.

When the GPIO pin is controlled by the GPIO, the value of Output Driver Enable Register (ODER) determines whether the pin is driven or not. When a bit in this register is one, the corresponding GPIO pin is driven by the GPIO. When the bit is zero, the GPIO does not drive the pin.

The level driven on a GPIO pin can be determined by writing the value to the corresponding bit in the Output Value Register (OVR).

23.6.1.5 Peripheral Muxing

The GPIO allows a single GPIO pin to be shared by multiple peripheral pins and the GPIO itself. Peripheral pins sharing the same GPIO pin are arranged into peripheral functions that can be selected one at a time. Peripheral functions are configured by writing the selected function value to the Peripheral Mux Registers (PMRn). To allow a peripheral pin access to the shared GPIO pin, GPIO control must be disabled for that pin, i.e. the corresponding bit in GPER must read zero.

A peripheral function value is set by writing bit zero to PMR0 and bit one to the same index position in PMR1 and so on. In a system with 4 peripheral functions A,B,C, and D, peripheral function C for GPIO pin four is selected by writing a zero to bit four in PMR0 and a one to the same bit index in PMR1. Refer to the GPIO Function Multiplexing chapter for details regarding pin function configuration for each GPIO pin.

23.6.2 Advanced Operation

23.6.2.1 Peripheral I/O Pin Control

When a GPIO pin is assigned to a peripheral function, i.e. the corresponding bit in GPER is zero, output and output enable is controlled by the selected peripheral pin. In addition the peripheral may control some or all of the other GPIO pin functions listed in Table 23-1, if the peripheral supports those features. All pin features not controlled by the selected peripheral is controlled by the GPIO.

Refer to the Module Configuration section for details regarding implemented GPIO pin functions and to the Peripheral chapter for details regarding I/O pin function control.

Table 23-1. I/O Pin function Control

Function name	GPIO mode	Peripheral mode
Output	OVR	Peripheral
Output enable	ODER	Peripheral
Pull-up	PUER	Peripheral if supported, else GPIO
Pull-down	PDER	Peripheral if supported, else GPIO
Drive strength	ODCRn	Peripheral if supported, else GPIO
Slew rate	OSRRn	Peripheral if supported, else GPIO
Schmitt trigger	STER	Peripheral if supported, else GPIO

23.6.2.2 Pull-up Resistor, Pull-down Resistor Control

Pull-up and pull-down can be configured for each GPIO pin. Pull-up allows the pin and any connected net to be pulled up to VDD if the net is not driven. Pull-down pulls the net to GND.

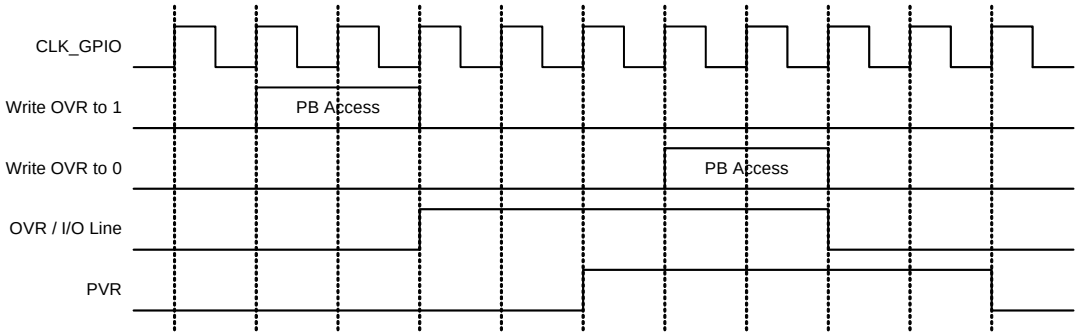
Pull-up and pull-down are useful for detecting if a pin is unconnected or if a mechanical button is pressed, for various communication protocols and to keep unconnected pins from floating.

Pull-up can be enabled and disabled by writing a one and a zero respectively to the corresponding bit in the Pull-up Enable Register (PUER). Pull-down can be enabled and disabled by writing a one and a zero respectively to the corresponding bit in the Pull-down Enable Register (PDER).

23.6.2.3 Output Pin Timings

Figure 23-3 shows the timing of the GPIO pin when writing to the Output Value Register (OVR). The same timing applies when performing a ‘set’ or ‘clear’ access, i.e. writing to OVRS or OVRC. The timing of PVR is also shown.

Figure 23-3. Output Pin Timings



23.6.2.4 *Pin Output Driver Control*

The GPIO has registers for controlling output drive properties of each pin, such as output driving capability and slew rate control.

The driving capability is controlled by the Output Driving Capability Registers (ODCRn) and the slew rate settings are controlled by the Output Slew Rate Registers (OSRRn).

For a GPIO pin with four different slew rate settings, a slew rate of two can be selected by writing a zero to the OSRR0 register at the bit position corresponding to the GPIO pin, and a one to the OSRR1 at the same bit position. The ODCRn registers are configured in the same way.

23.6.2.5 *Input Schmitt Trigger*

Each GPIO pin can be configured with an input Schmitt trigger. An input Schmitt trigger filters input signal using an hysteresis function, stopping noise from propagation into the system. The input Schmitt trigger can be enabled and disabled by writing a one and a zero respectively to the Schmitt Trigger Enable Register (STER).

23.6.2.6 *Interrupts*

The GPIO can be configured to generate an interrupt when it detects a change on a GPIO pin. Interrupts on a pin are enabled by writing a one to the corresponding bit in the Interrupt Enable Register (IER). The module can be configured to generate an interrupt whenever a pin changes value, or only on rising or falling edges. This is controlled by the Interrupt Mode Registers (IMRn). Interrupts on a pin can be enabled regardless of the GPIO pin being controlled by the GPIO or assigned to a peripheral function.

An interrupt can be generated on each GPIO pin. These interrupt generators are further grouped into groups of eight and connected to the NVIC. An interrupt request from any of the GPIO pin generators in the group will result in an interrupt request from that group to the NVIC if the corresponding bit for the GPIO pin in the IER is set. By grouping interrupt generators into groups of eight, four different interrupt handlers can be installed for each GPIO port.

The Interrupt Flag Register (IFR) can be read by software to determine which pin(s) caused the interrupt. The interrupt flag must be manually cleared by writing a zero to the corresponding bit in IFR.

GPIO interrupts will only be generated when CLK_GPIO is enabled.

23.6.2.7 *Input Glitch Filter*

Input glitch filters can be enabled on each GPIO pin. When the glitch filter is enabled, a glitch with duration of less than 1 CLK_GPIO cycle is automatically rejected, while a pulse with dura-

tion of 2 CLK_GPIO cycles or more is accepted. For pulse durations between 1 and 2 CLK_GPIO cycles, the pulse may or may not be taken into account, depending on the precise timing of its occurrence. Thus for a pulse to be guaranteed visible it must exceed 2 CLK_GPIO cycles, whereas for a glitch to be reliably filtered out, its duration must not exceed 1 CLK_GPIO cycle. The filter introduces 2 clock cycles latency.

The glitch filters are controlled by the Glitch Filter Enable Register (GFER). When a bit in GFER is one, the glitch filter on the corresponding pin is enabled. The glitch filter affects only interrupt inputs. Inputs to peripherals or the value read through PVR are not affected by the glitch filters.

23.6.2.8 *Interrupt Timings*

Figure 23-4 shows the timing for rising edge (or pin-change) interrupts when the glitch filter is disabled. For the pulse to be registered, it must be sampled at the rising edge of the clock. In this example, this is not the case for the first pulse. The second pulse is sampled on a rising edge and will trigger an interrupt request.

Figure 23-4. Interrupt Timing with Glitch Filter Disabled

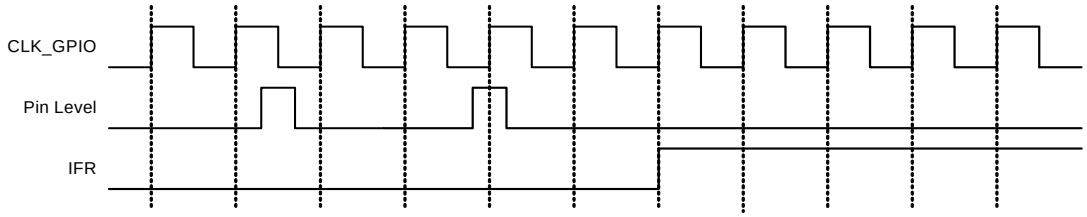
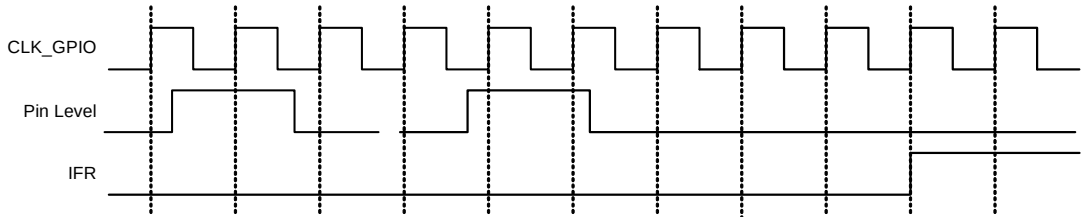


Figure 23-5 shows the timing for rising edge (or pin-change) interrupts when the glitch filter is enabled. For the pulse to be registered, it must be sampled on two subsequent rising edges. In the example, the first pulse is rejected while the second pulse is accepted and causes an interrupt request.

Figure 23-5. Interrupt Timing with Glitch Filter Enabled



23.6.2.9 *Peripheral Events*

Peripheral events allow direct peripheral to peripheral communication of specified events. See Section 31. ”Peripheral Event Controller (PEVC)” on page 845 for more information.

The GPIO can be programmed to output peripheral events whenever an interrupt condition is detected. The peripheral events configuration depends on the interrupt configuration. An event will be generated on the same condition as the interrupt (pin change, rising edge, or falling edge). The interrupt configuration is controlled by the IMR register. Peripheral event on a pin is enabled by writing a one to the corresponding bit in the Event Enable Register (EVER). The Peripheral Event trigger mode is shared with the interrupt trigger and is configured by writing to the IMR0 and IMR1 registers. Interrupt does not need to be enabled on a pin when peripheral

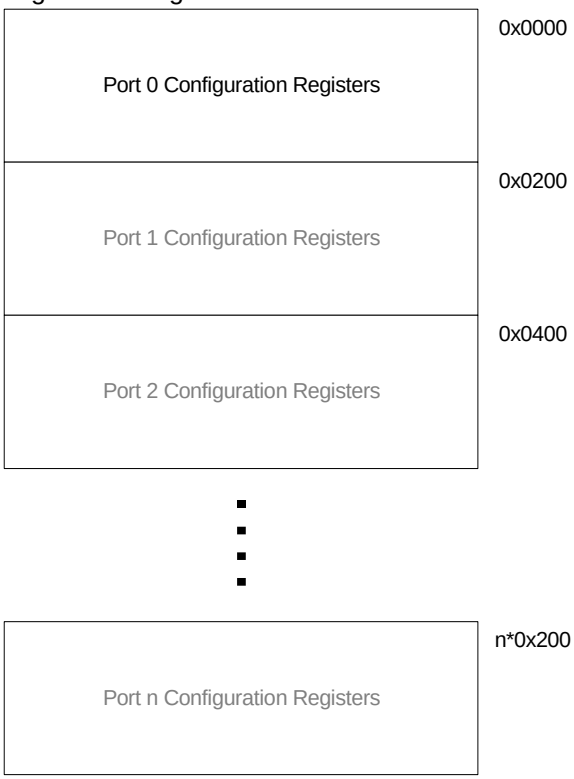
events are enabled. Peripheral Events are also affected by the Input Glitch Filter settings. See Section 23.6.2.7 for more information.

A peripheral event can be generated on each GPIO pin. Each port can then have up to 32 peripheral event generators. Groups of eight peripheral event generators in each port are ORed together to form a peripheral event line, so that each port has four peripheral event lines connected to the Peripheral Event System.

23.7 User Interface

The GPIO controller manages all the GPIO pins. The pins are managed as 32-bit ports that are configurable through a Peripheral Bus (PB) interface. Each port has a set of configuration registers. The overall memory map of the GPIO is shown below. The number of pins and hence the number of ports is product specific.

Figure 23-6. Port Configuration Registers



In the peripheral muxing table in the Package and Pinout chapter each GPIO pin has a unique number. Note that the PA, PB, PC, and PX ports do not necessarily directly correspond to the GPIO ports. To find the corresponding port and pin the following formulas can be used:

GPIO port = floor((GPIO number) / 32), example: floor((36)/32) = 1

GPIO pin = GPIO number % 32, example: 36 % 32 = 4

Table 23-2 shows the configuration registers for one port. Addresses shown are relative to the port address offset. The specific address of a configuration register is found by adding the register offset and the port offset to the GPIO start address. One bit in each of the configuration registers corresponds to a GPIO pin.

23.7.1 Access Types

Most configuration register can be accessed in four different ways. The first address location can be used to write the register directly. This address can also be used to read the register value. The following addresses facilitate three different types of write access to the register. Performing a “set” access, all bits written to one will be set. Bits written to zero will be unchanged by the operation. Performing a “clear” access, all bits written to one will be cleared. Bits written to zero will be unchanged by the operation. Finally, a toggle access will toggle the value of all bits writ-

ten to one. Again all bits written to zero remain unchanged. Note that for some registers (e.g. IFR), not all access methods are permitted.

Note that for ports with less than 32 bits, the corresponding control registers will have unused bits. This is also the case for features that are not implemented for a specific pin. Writing to an unused bit will have no effect. Reading unused bits will always return 0.

Table 23-2. GPIO Register Memory Map

Offset	Register	Function	Register Name	Access	Reset	Config. Protection	Access Protection
0x000	GPIO Enable Register	Read/Write	GPER	Read/Write	-(1)	Y	N
0x004	GPIO Enable Register	Set	GPERS	Write-only		Y	N
0x008	GPIO Enable Register	Clear	GPERC	Write-only		Y	N
0x00C	GPIO Enable Register	Toggle	GPERT	Write-only		Y	N
0x010	Peripheral Mux Register 0	Read/Write	PMR0	Read/Write	-(1)	Y	N
0x014	Peripheral Mux Register 0	Set	PMR0S	Write-only		Y	N
0x018	Peripheral Mux Register 0	Clear	PMR0C	Write-only		Y	N
0x01C	Peripheral Mux Register 0	Toggle	PMR0T	Write-only		Y	N
0x020	Peripheral Mux Register 1	Read/Write	PMR1	Read/Write	-(1)	Y	N
0x024	Peripheral Mux Register 1	Set	PMR1S	Write-only		Y	N
0x028	Peripheral Mux Register 1	Clear	PMR1C	Write-only		Y	N
0x02C	Peripheral Mux Register 1	Toggle	PMR1T	Write-only		Y	N
0x030	Peripheral Mux Register 2	Read/Write	PMR2	Read/Write	-(1)	Y	N
0x034	Peripheral Mux Register 2	Set	PMR2S	Write-only		Y	N
0x038	Peripheral Mux Register 2	Clear	PMR2C	Write-only		Y	N
0x03C	Peripheral Mux Register 2	Toggle	PMR2T	Write-only		Y	N
0x040	Output Driver Enable Register	Read/Write	ODER	Read/Write	-(1)	Y	N
0x044	Output Driver Enable Register	Set	ODERS	Write-only		Y	N
0x048	Output Driver Enable Register	Clear	ODERC	Write-only		Y	N
0x04C	Output Driver Enable Register	Toggle	ODERT	Write-only		Y	N
0x050	Output Value Register	Read/Write	OVR	Read/Write	-(1)	N	N
0x054	Output Value Register	Set	OVRS	Write-only		N	N
0x058	Output Value Register	Clear	OVRC	Write-only		N	N
0x05c	Output Value Register	Toggle	OVRT	Write-only		N	N
0x060	Pin Value Register	Read	PVR	Read-only	Depending on pin states	N	N

Table 23-2. GPIO Register Memory Map

Offset	Register	Function	Register Name	Access	Reset	Config. Protection	Access Protection
0x064	Pin Value Register	-	-	-		N	N
0x068	Pin Value Register	-	-	-		N	N
0x06c	Pin Value Register	-	-	-		N	N
0x070	Pull-up Enable Register	Read/Write	PUER	Read/Write	-(1)	Y	N
0x074	Pull-up Enable Register	Set	PUERS	Write-only		Y	N
0x078	Pull-up Enable Register	Clear	PUERC	Write-only		Y	N
0x07C	Pull-up Enable Register	Toggle	PUERT	Write-only		Y	N
0x080	Pull-down Enable Register	Read/Write	PDER	Read/Write	(1)	Y	N
0x084	Pull-down Enable Register	Set	PDERS	Write-only		Y	N
0x088	Pull-down Enable Register	Clear	PDERC	Write-only		Y	N
0x08C	Pull-down Enable Register	Toggle	PDERT	Write-only		Y	N
0x090	Interrupt Enable Register	Read/Write	IER	Read/Write	-(1)	N	N
0x094	Interrupt Enable Register	Set	IERS	Write-only		N	N
0x098	Interrupt Enable Register	Clear	IERC	Write-only		N	N
0x09C	Interrupt Enable Register	Toggle	IERT	Write-only		N	N
0x0A0	Interrupt Mode Register 0	Read/Write	IMR0	Read/Write	-(1)	N	N
0x0A4	Interrupt Mode Register 0	Set	IMR0S	Write-only		N	N
0x0A8	Interrupt Mode Register 0	Clear	IMR0C	Write-only		N	N
0x0AC	Interrupt Mode Register 0	Toggle	IMR0T	Write-only		N	N
0x0B0	Interrupt Mode Register 1	Read/Write	IMR1	Read/Write	-(1)	N	N
0x0B4	Interrupt Mode Register 1	Set	IMR1S	Write-only		N	N
0x0B8	Interrupt Mode Register 1	Clear	IMR1C	Write-only		N	N
0x0BC	Interrupt Mode Register 1	Toggle	IMR1T	Write-only		N	N
0x0C0	Glitch Filter Enable Register	Read/Write	GFER	Read/Write	-(1)	N	N
0x0C4	Glitch Filter Enable Register	Set	GFERS	Write-only		N	N
0x0C8	Glitch Filter Enable Register	Clear	GFERC	Write-only		N	N
0x0CC	Glitch Filter Enable Register	Toggle	GFERT	Write-only		N	N
0x0D0	Interrupt Flag Register	Read	IFR	Read-only	-(1)	N	N
0x0D4	Interrupt Flag Register	-	-	-		N	N
0x0D8	Interrupt Flag Register	Clear	IFRC	Write-only		N	N
0x0DC	Interrupt Flag Register	-	-	-		N	N
0x100	Output Driving Capability Register 0	Read/Write	ODCR0	Read/Write	-(1)	Y	N
0x104	Output Driving Capability Register 0	Set	ODCR0S	Write-only		Y	N
0x108	Output Driving Capability Register 0	Clear	ODCR0C	Write-only		Y	N
0x10C	Output Driving Capability Register 0	Toggle	ODCR0T	Write-only		Y	N

Table 23-2. GPIO Register Memory Map

Offset	Register	Function	Register Name	Access	Reset	Config. Protection	Access Protection
0x110	Output Driving Capability Register 1	Read	ODCR1	Read/Write	-(1)	Y	N
0x114	Output Driving Capability Register 1	Set	ODCR1S	Write-only		Y	N
0x118	Output Driving Capability Register 1	Clear	ODCR1C	Write-only		Y	N
0x11C	Output Driving Capability Register 1	Toggle	ODCR1T	Write-only		Y	N
0x130	Output Slew Rate Register 0	Read	OSRR0	Read/Write		Y	N
0x134	Output Slew Rate Register 0	Set	OSRR0S	Write-only		Y	N
0x138	Output Slew Rate Register 0	Clear	OSRR0C	Write-only		Y	N
0x13C	Output Slew Rate Register 0	Toggle	OSRR0T	Write-only		Y	N
0x160	Schmitt Trigger Enable Register	Read	STER	Read/Write	-(1)	Y	N
0x164	Schmitt Trigger Enable Register	Set	STERS	Write-only		Y	N
0x168	Schmitt Trigger Enable Register	Clear	STERC	Write-only		Y	N
0x16C	Schmitt Trigger Enable Register	Toggle	STERT	Write-only		Y	N
0x180	Event Enable Register	Read	EVER	Read/Write	-(1)	N	N
0x184	Event Enable Register	Set	EVERS	Write-only		N	N
0x188	Event Enable Register	Clear	EVERC	Write-only		N	N
0x18C	Event Enable Register	Toggle	EVERT	Write-only		N	N
0x1F8	Parameter Register	Read	PARAMETER	Read-only	-(1)	N	N
0x1FC	Version Register	Read	VERSION	Read-only	-(1)	N	N

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

23.7.2 GPIO Enable Register

Name: GPER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x000, 0x004, 0x008, 0x00C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-P31: GPIO Enable**
  0: A peripheral function controls the corresponding pin.
  1: The GPIO controls the corresponding pin.

23.7.3 Peripheral Mux Register 0

Name: PMR0

Access: Read/Write, Set, Clear, Toggle

Offset: 0x010, 0x014, 0x018, 0x01C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Peripheral Multiplexer Select bit 0**

23.7.4 Peripheral Mux Register 1

Name: PMR1

Access: Read/Write, Set, Clear, Toggle

Offset: 0x020, 0x024, 0x028, 0x02C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Peripheral Multiplexer Select bit 1**

23.7.5 Peripheral Mux Register 2

Name: PMR2

Access: Read/Write, Set, Clear, Toggle

Offset: 0x030, 0x034, 0x038, 0x03C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Peripheral Multiplexer Select bit 2**

{PMR2, PMR1, PMR0}	Selected Peripheral Function
000	A
001	B
010	C
011	D
100	E
101	F
110	G
111	H

23.7.6 Output Driver Enable Register

Name: ODER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x040, 0x044, 0x048, 0x04C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Output Driver Enable**
  0: The output driver is disabled for the corresponding pin.
  1: The output driver is enabled for the corresponding pin.

23.7.7 Output Value Register

Name: OVR

Access: Read/Write, Set, Clear, Toggle

Offset: 0x050, 0x054, 0x058, 0x05C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Output Value**
  0: The value to be driven on the GPIO pin is 0.
  1: The value to be driven on the GPIO pin is 1.

23.7.8 Pin Value Register

Name: PVR

Access: Read-only

Offset: 0x060, 0x064, 0x068, 0x06C

Reset Value: Depending on pin states

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Pin Value**
  0: The GPIO pin is at level zero.
  1: The GPIO pin is at level one.
  Note that the level of a pin can only be read when the corresponding pin in GPER is one or interrupt is enabled for the pin.

23.7.9 Pull-up Enable Register

Name: PUER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x070, 0x074, 0x078, 0x07C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Pull-up Enable**
  Writing a zero to a bit in this register will disable pull-up on the corresponding pin.
  Writing a one to a bit in this register will enable pull-up on the corresponding pin.

23.7.10 Pull-down Enable Register

Name: PDER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x080, 0x084, 0x088, 0x08C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Pull-down Enable**

{PUER, PDER}	Selected Function
00	Disabled
01	Pull-down enabled
10	Pull-up enabled
11	Buskeeper enabled

23.7.11 Interrupt Enable Register

Name: IER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x090, 0x094, 0x098, 0x09C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Interrupt Enable**
  0: Interrupt is disabled for the corresponding pin.
  1; Interrupt is enabled for the corresponding pin.

23.7.12 Interrupt Mode Register 0

Name: IMR0

Access: Read/Write, Set, Clear, Toggle

Offset: 0x0A0, 0x0A4, 0x0A8, 0x0AC

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Interrupt Mode Bit 0**

23.7.13 Interrupt Mode Register 1

Name: IMR1

Access: Read/Write, Set, Clear, Toggle

Offset: 0x0B0, 0x0B4, 0x0B8, 0x0BC

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Interrupt Mode Bit 1**

{IMR1, IMR0}	Interrupt Mode
00	Pin Change
01	Rising Edge
10	Falling Edge
11	Reserved

23.7.14 Glitch Filter Enable Register

Name: GFER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x0C0, 0x0C4, 0x0C8, 0x0CC

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Glitch Filter Enable**
  0: Glitch filter is disabled for the corresponding pin.
  1: Glitch filter is enabled for the corresponding pin.
  The value of this register should only be changed when the corresponding bit in IER is zero. Updating GFER while interrupt on the corresponding pin is enabled can cause an unintentional interrupt to be triggered.

23.7.15 Interrupt Flag Register

Name: IFR

Access: Read, Clear

Offset: 0x0D0, 0x0D8

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Interrupt Flag**
  0: No interrupt condition has been detected on the corresponding pin.
  1: An interrupt condition has been detected on the corresponding pin.
  The number of interrupt request lines depends on the number of GPIO pins on the MCU. Refer to the product specific data for details. Note also that a bit in the Interrupt Flag register is only valid if the corresponding bit in IER is one.

23.7.16 Output Driving Capability Register 0

Name: ODCR0

Access: Read/Write, Set, Clear, Toggle

Offset: 0x100, 0x104, 0x108, 0x10C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Output Driving Capability Register Bit 0**

23.7.17 Output Driving Capability Register 1

Name: ODCR1

Access: Read/Write, Set, Clear, Toggle

Offset: 0x110, 0x114, 0x118, 0x11C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Output Driving Capability Bit 1**

{ODCR1, ODCR0}	Interrupt Mode
00	Lowest drive strength
01	...
10	...
11	Highest drive strength

For the actual drive strength of the pin, refer to Section 42. ”Electrical Characteristics” on page 1121.

23.7.18 Output Slew Rate Register 0

Name: OSRR0

Access: Read/Write, Set, Clear, Toggle

Offset: 0x130, 0x134, 0x138, 0x13C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Output Slew Rate Control Enable**
  0: Slew rate control is disabled for the corresponding pin.
  1: Slew rate control is enabled for the corresponding pin.

23.7.19 Schmitt Trigger Enable Register

Name: STER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x160, 0x164, 0x168, 0x16C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Schmitt Trigger Enable**
  0: Schmitt trigger is disabled for the corresponding pin.
  1: Schmitt trigger is enabled for the corresponding pin.

23.7.20 Event Enable Register

Name: EVER

Access: Read/Write, Set, Clear, Toggle

Offset: 0x180, 0x184, 0x188, 0x18C

Reset Value: -

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Event Enable**
  0: Peripheral Event is disabled for the corresponding pin.
  1: Peripheral Event is enabled for the corresponding pin.

23.7.21 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0x1F8

Reset Value: -

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

- **PARAMETER:**
  0: The corresponding pin is not implemented in this GPIO port.
  1: The corresponding pin is implemented in this GPIO port.
  There is one PARAMETER register per GPIO port. Each bit in the Parameter Register indicates whether the corresponding GPER bit is implemented.

23.7.22 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x1FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			
15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			
7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

23.8 Module Configuration

The specific configuration for each GPIO instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 23-3. GPIO Configuration

Feature	48 pin package	64-pin package	100-pin package
Number of GPIO ports	1	2	3
Number of peripheral functions	8	8	8

Table 23-4. Implemented Pin Functions

Pin Function	Implemented	Notes
Pull-up	On all pins	Controlled by PUER or peripheral
Pull-down	On all pins	Controlled by PDER or peripheral
Slew Rate	On all pins	Controlled by OSRR or peripheral
Drive Strength	On all pins	Controlled by ODCR or peripheral
Schmitt Trigger Enable	On all pins	Controlled by STER or peripheral

Table 23-5. GPIO Clocks

Clock Name	Description
CLK_GPIO	Clock for the GPIO bus interface

The reset values for all GPIO registers are zero, with the following exceptions:

Table 23-6. Register Reset Values

Port	Register	Reset Value
0	GPER	0xFFFFFFFF
0	PMR0	0x00000008
0	PMR1	0x00000000
0	PMR2	0x00000000
0	PUER	0x00000000
0	GFER	0x00000000
0	PARAMETER	0xFFFFFFFC
0	VERSION	0x00000215
1	GPER	0x0000FFFF
1	PMR0	0x00000000
1	PMR1	0x00000000

Table 23-6. Register Reset Values

Port	Register	Reset Value
1	PMR2	0x00000000
1	PUER	0x00000000
1	GFER	0x00000000
1	PARAMETER	0x000FFFF
1	VERSION	0x00000215
2	GPER	0x00000000
2	PMR0	0x00000000
2	PMR1	0x00000000
2	PMR2	0x00000000
2	PUER	0x00000000
2	GFER	0x00000000
2	PARAMETER	0xFFFFFFFF
2	VERSION	0x00000215

24. Universal Synchronous Asynchronous Receiver Transmitter (USART)

Rev: 6.0.2.6

24.1 Features

- Configurable baud rate generator
- 5- to 9-bit full-duplex, synchronous and asynchronous, serial communication
  - 1, 1.5, or 2 stop bits in asynchronous mode, and 1 or 2 in synchronous mode
  - Parity generation and error detection, with both normal and inverted data streams
  - Framing- and overrun error detection
  - MSB- or LSB-first
  - Optional break generation and detection
  - Receiver frequency oversampling by 8 or 16 times
  - Optional RTS-CTS hardware handshaking
  - Optional DTR-DSR-DCD-RI modem signal management
  - Receiver Time-out and transmitter Timeguard
  - Optional Multidrop mode with address generation and detection
- RS485 with line driver control
- ISO7816, T=0 and T=1 protocols for Interfacing with smart cards
  - Inverted data management, NACK handling, and customizable error counter
- IrDA modulation and demodulation
  - Communication at up to 115.2 Kbit/s
- SPI Mode
  - Master or slave
  - Configurable serial clock phase and polarity
  - CLK SPI serial clock frequency up to a quarter of the CLK_USART internal clock frequency
- LIN Mode
  - Compliant with LIN 1.3 and LIN 2.0 specifications
  - Master or slave
  - Processing of Frames with up to 256 data bytes
  - Configurable response data length, optionally defined automatically by the Identifier
  - Self synchronization in slave node configuration
  - Automatic processing and verification of the “Break Field” and “Sync Field”
  - The “Break Field” is detected even if it is partially superimposed with a data byte
  - Optional, automatic identifier parity management
  - Optional, automatic checksum management
  - Supports both “Classic” and “Enhanced” checksum types
  - Full LIN error checking and reporting
  - Frame Slot Mode: the master allocates slots to scheduled frames automatically.
  - Wakeup signal generation
- Test Modes
  - Automatic echo, remote- and local loopback
- Supports two Peripheral DMA Controller channels
  - Buffer transfers without processor intervention

24.2 Overview

The Universal Synchronous Asynchronous Receiver Transmitter (USART) provides a full duplex, universal, synchronous/asynchronous serial link. Data frame format is widely configurable, including basic length, parity, and stop bit settings, maximizing standards support. The receiver implements parity-, framing-, and overrun error detection, and can handle un-fixed

frame lengths with the time-out feature. The USART supports several operating modes, providing an interface to RS485, LIN, and SPI buses, with ISO7816 T=0 and T=1 smart card slots, and infrared transceivers, and modem port connections. Communication with slow and remote devices is eased by the timeguard. Duplex multidrop communication is supported by address and data differentiation through the parity bit. The hardware handshaking feature enables an out-of-band flow control, automatically managing RTS and CTS pins. The Peripheral DMA Controller connection enables memory transactions, and the USART supports chained buffer management without processor intervention. Automatic echo, remote-, and local loopback test modes are also supported.

24.3 Block Diagram

Figure 24-1. USART Block Diagram

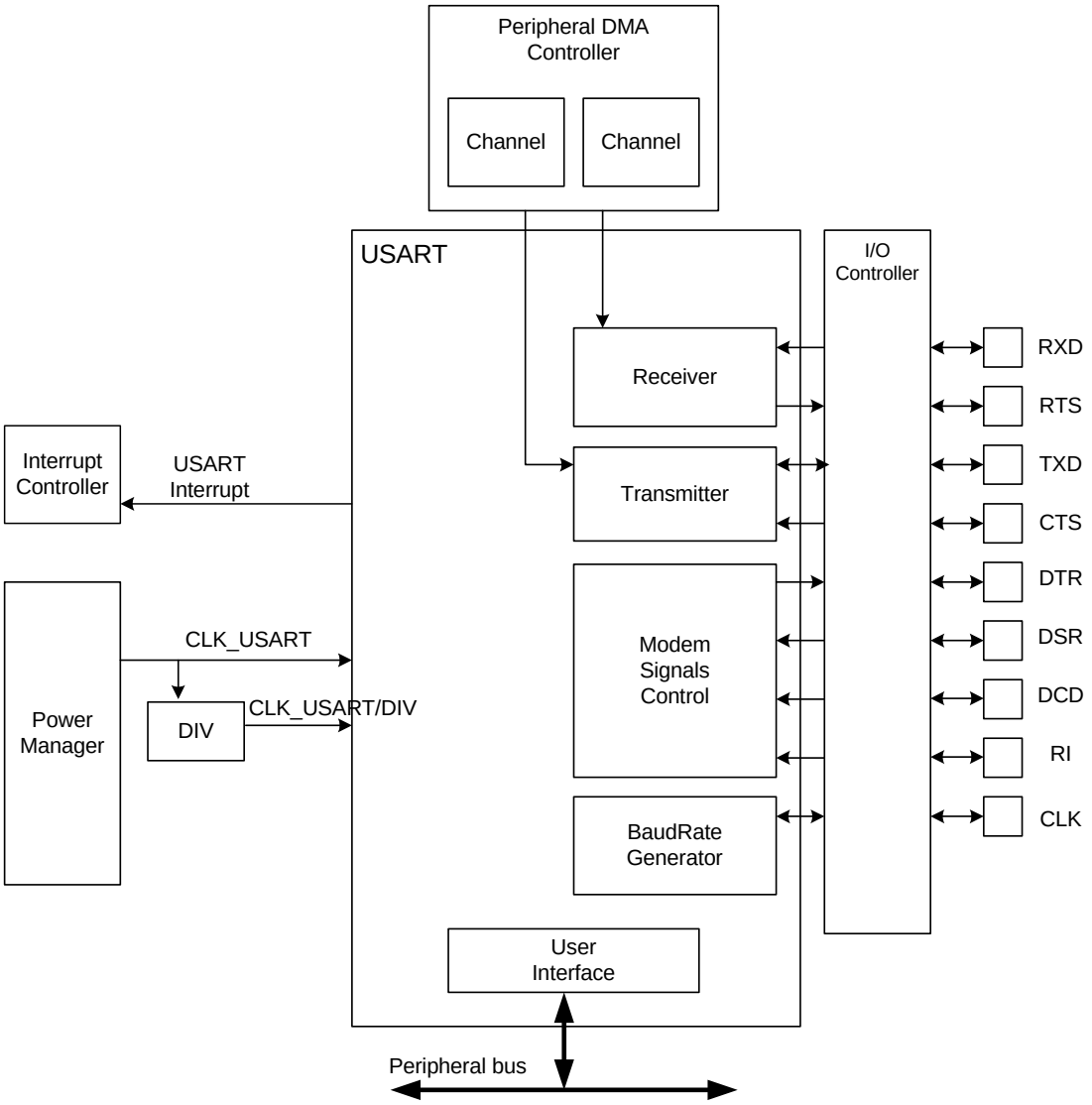


Figure 24-2. USART Block Diagram

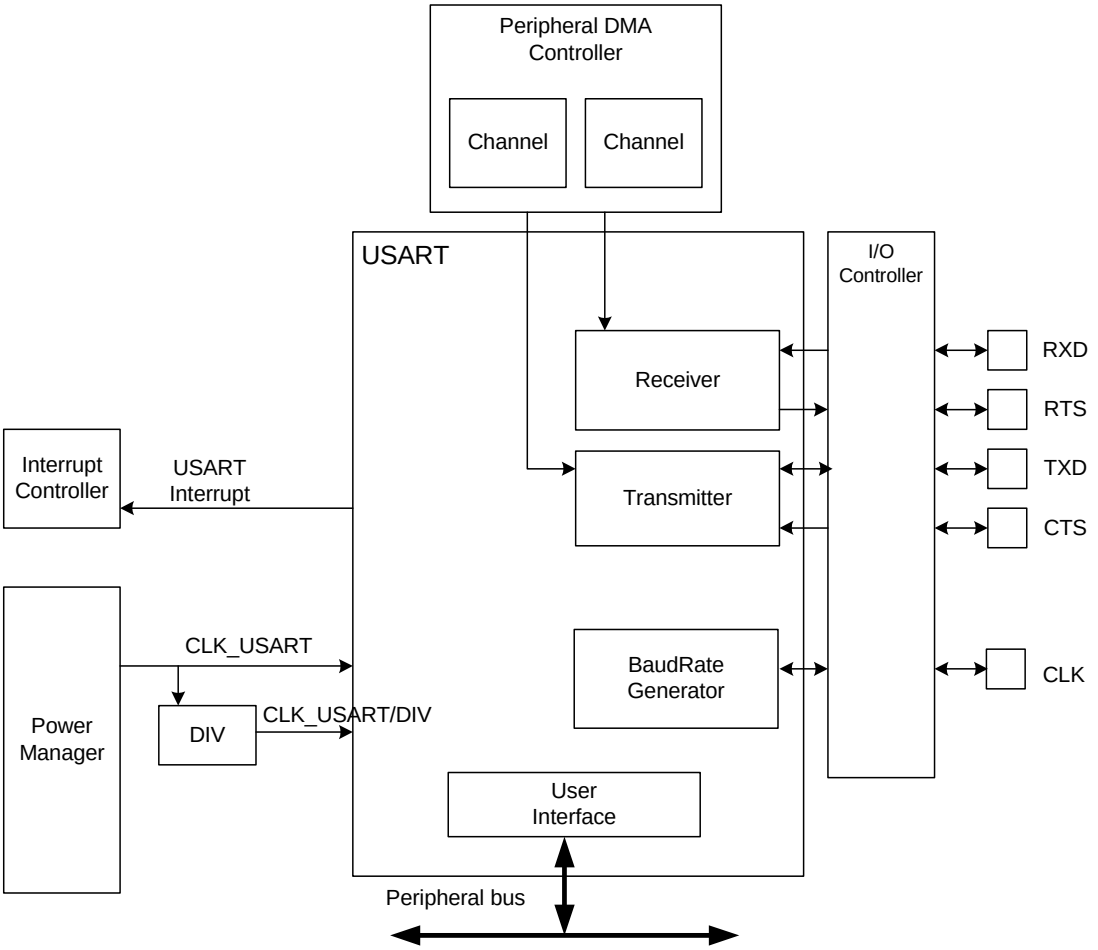


Table 24-1. SPI Operating Mode

PIN	USART	SPI Slave	SPI Master
RXD	RXD	MOSI	MISO
TXD	TXD	MISO	MOSI
RTS	RTS	–	CS
CTS	CTS	CS	–

24.4 I/O Lines Description

Table 24-2. I/O Lines Description

Name	Description	Type	Active Level
CLK	Serial Clock	I/O	
TXD	Transmit Serial Data or Master Out Slave In (MOSI) in SPI master mode or Master In Slave Out (MISO) in SPI slave mode	Output	
RXD	Receive Serial Data or Master In Slave Out (MISO) in SPI master mode or Master Out Slave In (MOSI) in SPI slave mode	Input	
RI	Ring Indicator	Input	Low
DSR	Data Set Ready	Input	Low
DCD	Data Carrier Detect	Input	Low
DTR	Data Terminal Ready	Output	Low
CTS	Clear to Send or Slave Select (NSS) in SPI slave mode	Input	Low
RTS	Request to Send or Slave Select (NSS) in SPI master mode	Output	Low

24.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

24.5.1 I/O Lines

The USART pins may be multiplexed with the I/O Controller lines. The user must first configure the I/O Controller to assign these pins to their peripheral functions. Unused I/O lines may be used for other purposes.

To prevent the TXD line from falling when the USART is disabled, the use of an internal pull-up is required. If the hardware handshaking feature or modem mode is used, the internal pull-up on RTS must also be enabled.

All the pins of the modems may or may not be implemented on the USART. On USARTs not equipped with the corresponding pins, the associated control bits and statuses have no effect on the behavior of the USART.

24.5.2 Clocks

The clock for the USART bus interface (CLK_USART) is generated by the Power Manager. It is recommended to disable the USART before disabling the clock, to avoid freezing the USART in an undefined state.

24.5.3 Interrupts

The USART interrupt request line is connected to the NVIC. Using the USART interrupt requires the NVIC to be programmed first.

24.6 Functional Description

24.6.1 USART Operating Modes

The USART can operate in several modes:

- Normal
- RS485, described in Section 24.6.5 “RS485 Mode” on page 589
- Hardware handshaking, described in Section 24.6.6 “Hardware Handshaking” on page 590
- Modem, described in Section 24.6.7 “Modem Mode” on page 591
- ISO7816, described in Section 24.6.8 “ISO7816 Mode” on page 592
- IrDA, described in Section 24.6.9 “IrDA Mode” on page 595
- LIN Master, described in Section 24.6.10 “LIN Mode” on page 597
- LIN Slave, described in Section 24.6.10 “LIN Mode” on page 597
- SPI Master, described in Section 24.6.15 “SPI Mode” on page 609
- SPI Slave, described in Section 24.6.15 “SPI Mode” on page 609

The operating mode is selected by writing to the Mode field in the “Mode Register” (MR.MODE).

Table 24-3. MR.MODE

MR.MODE	Mode of the USART
0x0	Normal
0x1	RS485
0x2	Hardware Handshaking
0x3	Modem
0x4	IS07816 Protocol: T = 0
0x6	IS07816 Protocol: T = 1
0x8	IrDA
0xA	LIN Master
0xB	LIN Slave
0xE	SPI Master
0xF	SPI Slave
Others	Reserved

In addition, Synchronous or Asynchronous mode is selected by writing to the Synchronous Mode Select bit in MR (MR.SYNC). By default, MR.MODE and MR.SYNC are both zero, and the USART operates in Normal Asynchronous mode.

24.6.2 Basic Operation

To start using the USART, the user must perform the following steps:

1. Configure the baud rate by writing to the Baud Rate Generator Register (BRGR) as described in “Baud Rate Generator” on page 587
2. Select the operating mode by writing to the relevant fields in the Mode Register (MR)
3. Enable the transmitter and/or receiver, by writing a one to CR.TXEN and/or CR.RXEN respectively

4. Check that CSR.TXRDY and/or CSR.RXRDY is one before writing to THR and/or reading from RHR respectively

24.6.2.1 *Receiver and Transmitter Control*

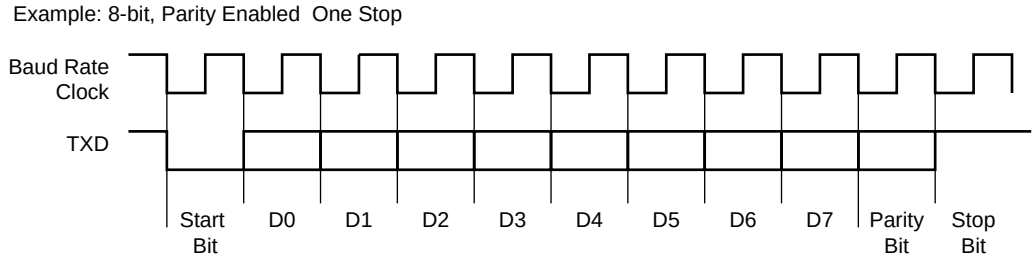
After a reset, the transceiver is disabled. The receiver/transmitter is enabled by writing a one to the Receiver Enable/Transmitter Enable bit in the Control Register (CR.RXEN/CR.TXEN) respectively. They may be enabled together and can be configured both before and after they have been enabled. The user can reset the USART receiver/transmitter at any time by writing a one to the Reset Receiver/Reset Transmitter bit (CR.RSTRX/CR.RSTTX) respectively. This software reset clears status bits and resets internal state machines, immediately halting any communication. The user interface configuration registers will retain their values.

The user can disable the receiver/transmitter by writing a one to either the Receiver Disable, or Transmitter Disable bit (CR.RXDIS, or CR.TXDIS). If the receiver is disabled during a character reception, the USART will wait for the current character to be received before disabling. If the transmitter is disabled during transmission, the USART will wait until both the current character and the character stored in the Transmitter Holding Register (THR) are transmitted before disabling. If a timeguard has been implemented it will remain functional during the transmission.

24.6.2.2 *Transmitter Operations*

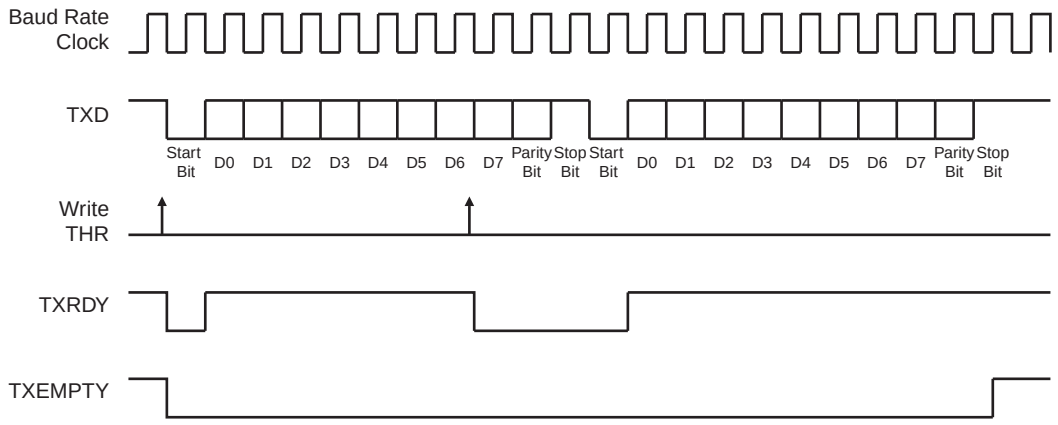
The transmitter operates equally in both Synchronous and Asynchronous operating modes (MR.SYNC). One start bit, up to 9 data bits, an optional parity bit, and up to two stop bits are successively shifted out on the TXD pin at each falling edge of the serial clock. The number of data bits is selected by the Character Length field (MR.CHRL) and the 9-bit Character Length bit in the Mode Register (MR.MODE9). Nine bits are selected by writing a one to MR.MODE9, overriding any value in MR.CHRL. The parity bit configuration is selected in the MR.PAR field. The Most Significant Bit First bit (MR.MSBF) selects which data bit to send first. The number of stop bits is selected by the MR.NBSTOP field. The 1.5 stop bit configuration is only supported in asynchronous mode.

Figure 24-3. Character Transmit



The characters are sent by writing to the Character to be Transmitted field (THR.TXCHR). The transmitter status can be read from the Transmitter Ready and Transmitter Empty bits in the Channel Status Register (CSR.TXRDY/CSR.TXEMPTY). CSR.TXRDY is set when THR is empty. CSR.TXEMPTY is set when both THR and the transmit shift register are empty (transmission complete). An interrupt request is generated if the corresponding bit in the Interrupt Mask Register (IMR) is set (IMR.TXRDY/IMR.TXEMPTY). Both CSR.TXRDY and CSR.TXEMPTY are cleared when the transmitter is disabled. CSR.TXRDY and CSR.TXEMPY can also be cleared by writing a one to the Start Break bit in CR (CR.STTBRK). Writing a character to THR while CSR.TXRDY is zero has no effect and the written character will be lost.

Figure 24-4. Transmitter Status



24.6.2.3 Asynchronous Receiver

If the USART is configured in an asynchronous operating mode (MR.SYNC is zero), the receiver will oversample the RXD input line by either 8 or 16 times the Baud Rate Clock, as selected by the Oversampling Mode bit (MR.OVER). If the line is zero for half a bit period (four or eight consecutive samples, respectively), a start bit will be assumed, and the following 8th or 16th sample will determine the logical value on the line, resulting in bit values being determined at the middle of the bit period.

The number of data bits, endianess, parity mode, and stop bits are selected by the same bits and fields as for the transmitter (MR.CHRL, MR.MODE9, MR.MSBF, MR.PAR, and MR.NBSTOP). The synchronization mechanism will only consider one stop bit, regardless of the used protocol, and when the first stop bit has been sampled, the receiver will automatically begin looking for a new start bit, enabling resynchronization even if there is a protocol mismatch. Figure 24-5 and Figure 24-6 illustrate start bit detection and character reception in asynchronous mode.

Figure 24-5. Asynchronous Start Bit Detection

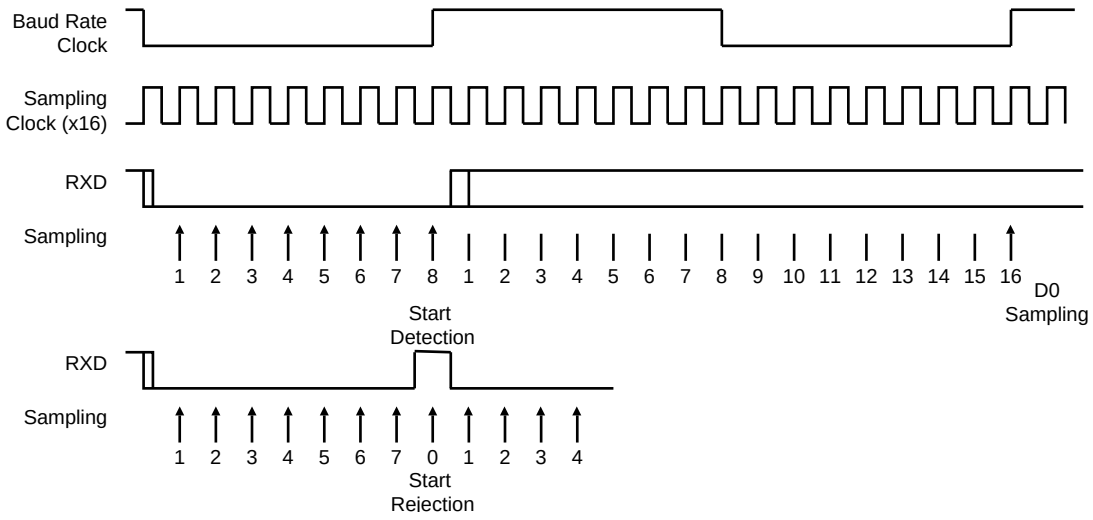
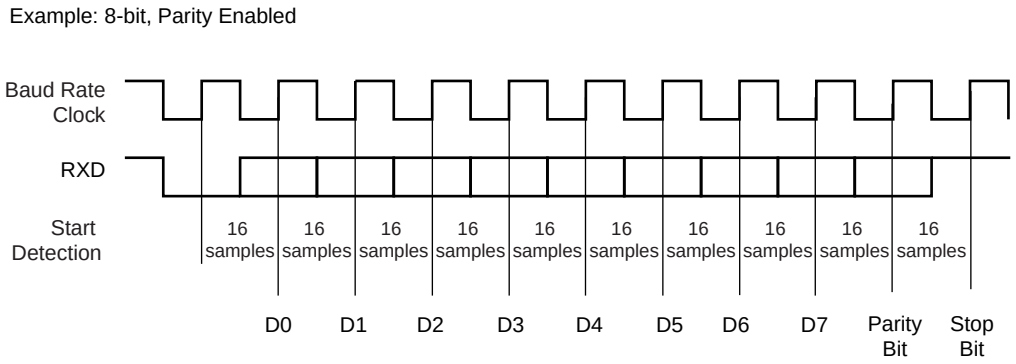


Figure 24-6. Asynchronous Mode Character Reception



24.6.2.4 *Synchronous Receiver*

In synchronous mode (MR.SYNC is one), the receiver samples the RXD signal on each rising edge of the Baud Rate Clock, as illustrated in Figure 24-7. If a low level is detected, it is considered as a start bit. Configuration bits and fields are the same as in asynchronous mode.

Figure 24-7. Synchronous Mode Character Reception

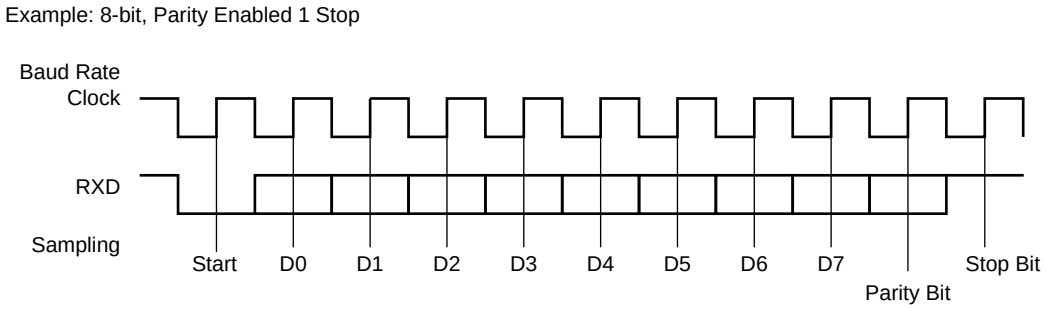
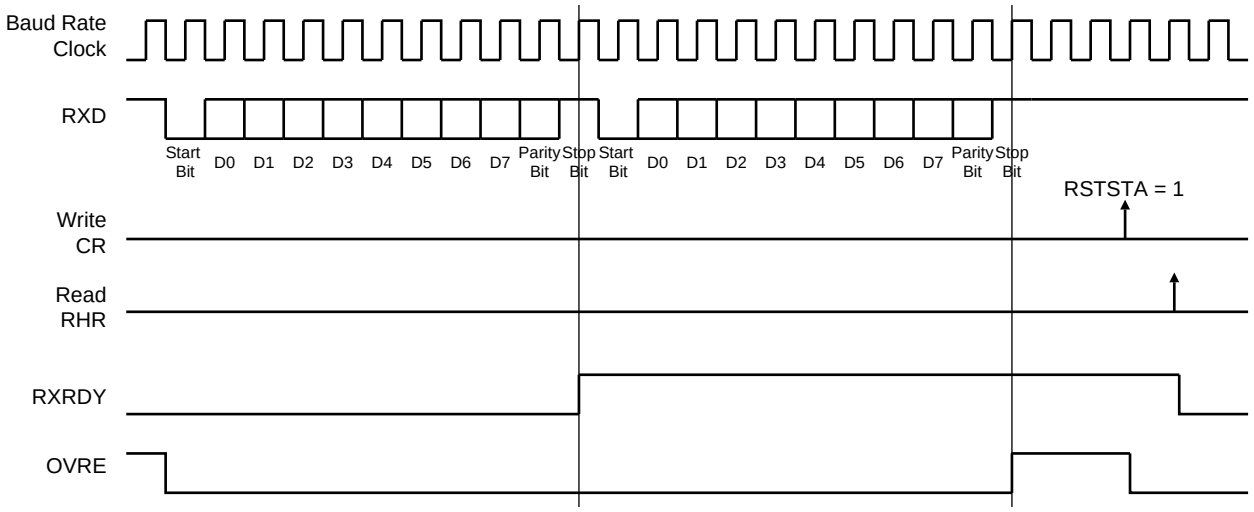


Figure 24-8. Receiver Status



24.6.2.5 *Receiver Operations*

When a character reception is completed, it is transferred to the Received Character field in the Receive Holding Register (RHR.RXCHR), and the Receiver Ready bit in the Channel Status Register (CSR.RXRDY) is set. An interrupt request is generated if the Receiver Ready bit in the

Interrupt Mask Register (IMR.RXRDY) is set. If CSR.RXRDY is already set, RHR will be overwritten and the Overrun Error bit (CSR.OVRE) is set. An interrupt request is generated if the Overrun Error bit in IMR is set. Reading RHR will clear CSR.RXRDY, and writing a one to the Reset Status bit in the Control Register (CR.RSTSTA) will clear CSR.OVRE. Refer to Figure 24-8.

24.6.3 Other Considerations

24.6.3.1 Parity

The USART supports five parity modes, selected by MR.PAR:

- Even parity
- Odd parity
- Parity forced to zero (space)
- Parity forced to one (mark)
- No parity

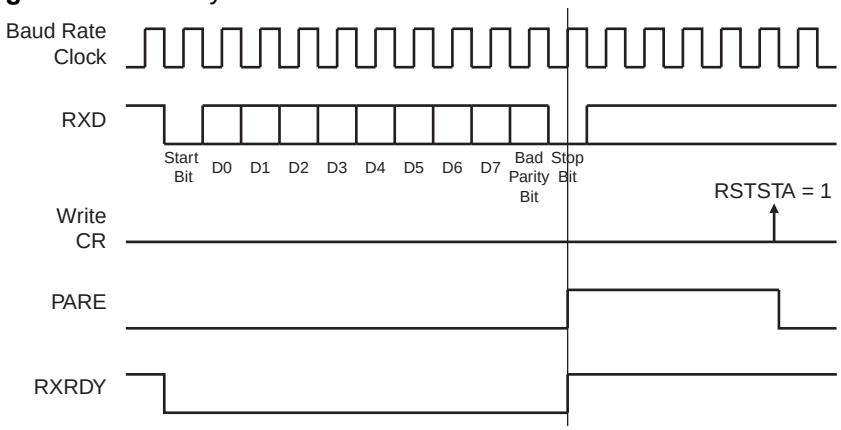
The PAR field also enables the Multidrop mode, see “Multidrop Mode” on page 584. If even parity is selected (MR.PAR is 0x0), the parity bit will be zero if there is an even number of ones in the data character, and one if there is an odd number. For odd parity the reverse applies. If space or mark parity is chosen (MR.PAR is 0x2 or 0x3, respectively), the parity bit will always be a zero or one, respectively. See Table 24-4.

Table 24-4. Parity Bit Examples

Alphanum Character	Hex	Bin	Parity Mode				
			Odd	Even	Mark	Space	None
A	0x41	0100 0001	1	0	1	0	-
V	0x56	0101 0110	1	0	1	0	-
R	0x52	0101 0010	0	1	1	0	-

The receiver will report parity errors in CSR.PARE, unless parity is disabled. An interrupt request is generated if the PARE bit in the Interrupt Mask Register is set (IMR.PARE). Writing a one to CR.RSTSTA will clear CSR.PARE. See Figure 24-9.

Figure 24-9. Parity Error



24.6.3.2 Multidrop Mode

If MR.PAR is either 0x6 or 0x7, the USART runs in Multidrop mode. This mode differentiates data and address characters. Data has the parity bit zero and addresses have a one. By writing a one to the Send Address bit (CR.SENDA) the user will cause the next character written to THR to be transmitted as an address. Receiving a character with a one as parity bit will report parity error by setting CSR.PARE. An interrupt request is generated if the PARE bit in the Interrupt Mask Register is set (IMR.PARE).

24.6.3.3 Transmitter Timeguard

The timeguard feature enables the USART to interface slow devices by inserting an idle state on the TXD line in between two characters. This idle state corresponds to a long stop bit, whose duration is selected by the Timeguard Value field in the Transmitter Timeguard Register (TTGR.TG). The transmitter will hold the TXD line high for TTGR.TG bit periods, in addition to the number of stop bits. As illustrated in Figure 24-10, the behavior of TXRDY and TXEMPTY is modified when TG has a non-zero value. If a pending character has been written to THR, the CSR.TXRDY bit will not be set until this characters start bit has been sent. CSR.TXEMPTY will remain low until the timeguard transmission has completed.

Figure 24-10. Timeguard Operation

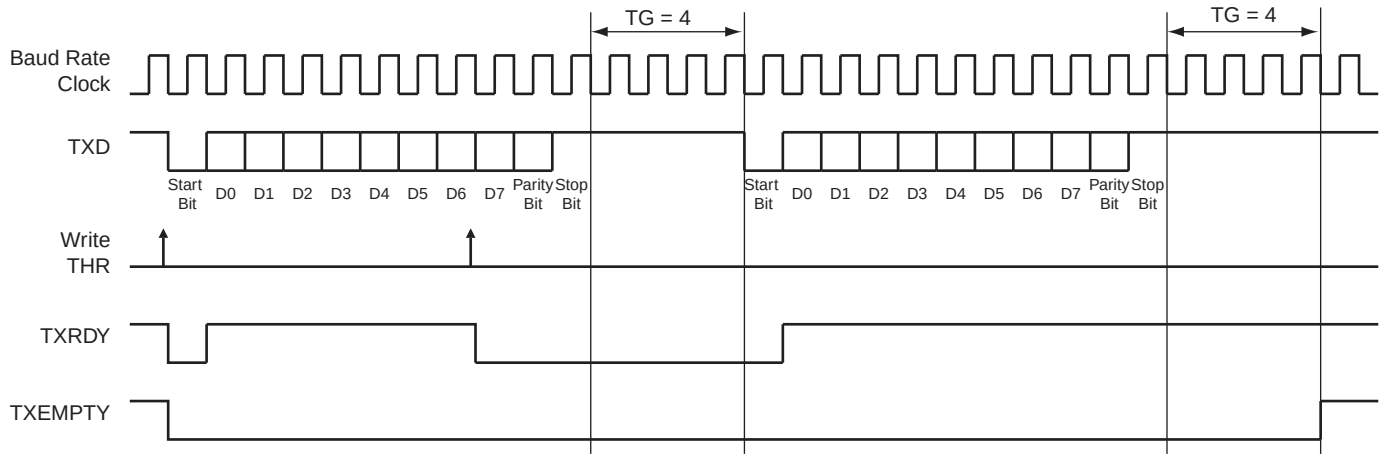


Table 24-5. Maximum Baud Rate Dependent Timeguard Durations

Baud Rate (bit/sec)	Bit time (µs)	Timeguard (ms)
1 200	833	212.50
9 600	104	26.56
14400	69.4	17.71
19200	52.1	13.28
28800	34.7	8.85
33400	29.9	7.63
56000	17.9	4.55
57600	17.4	4.43
115200	8.7	2.21

24.6.3.4 Receiver Time-out

The Time-out Value field in the Receiver Time-out Register (RTOR.TO) enables handling of variable-length frames by detection of selectable idle durations on the RXD line. The value written to TO is loaded to a decremental counter, and unless it is zero, a time-out will occur when the amount of inactive bit periods matches the initial counter value. If a time-out has not occurred, the counter will reload and restart every time a new character arrives. A time-out sets the Receiver Time-out bit in CSR (CSR.TIMEOUT). An interrupt request is generated if the Receiver Time-out bit in the Interrupt Mask Register (IMR.TIMEOUT) is set. Clearing TIMEOUT can be done in two ways:

- Writing a one to the Start Time-out bit (CR.STTTO). This also aborts count down until the next character has been received.
- Writing a one to the Reload and Start Time-out bit (CR.RETTO). This also reloads the counter and restarts count down immediately.

Figure 24-11. Receiver Time-out Block Diagram

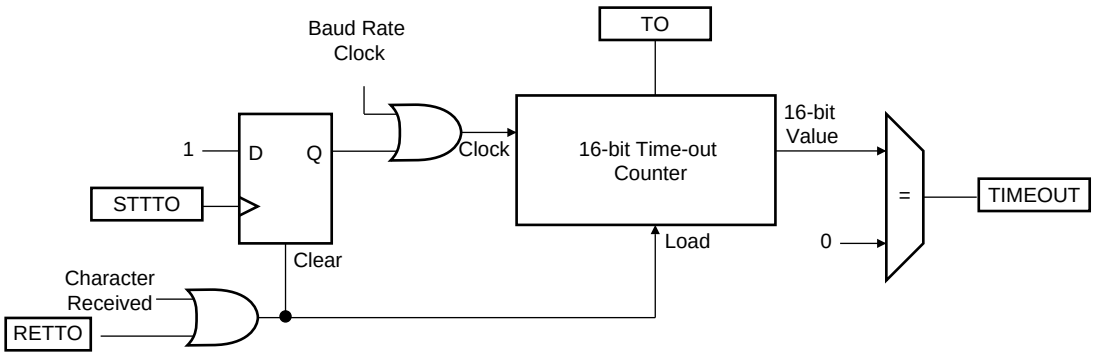


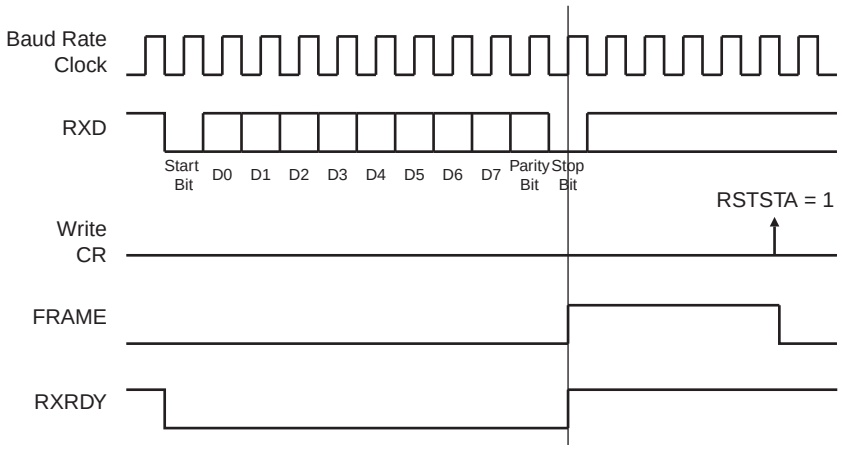
Table 24-6. Maximum Time-out Period

Baud Rate (bit/sec)	Bit Time (µs)	Time-out (ms)
600	1 667	109 225
1 200	833	54 613
2 400	417	27 306
4 800	208	13 653
9 600	104	6 827
14400	69	4 551
19200	52	3 413
28800	35	2 276
33400	30	1 962
56000	18	1 170
57600	17	1 138
200000	5	328

24.6.3.5 Framing Error

The receiver is capable of detecting framing errors. A framing error has occurred if a stop bit reads as zero. This can occur if the transmitter and receiver are not synchronized. A framing error is reported by CSR.FRAME as soon as the error is detected, at the middle of the stop bit. An interrupt request is generated if the Framing Error bit in the Interrupt Mask Register (IMR.FRAME) is set. CSR.FRAME is cleared by writing a one to CR.RSTSTA.

Figure 24-12. Framing Error Status

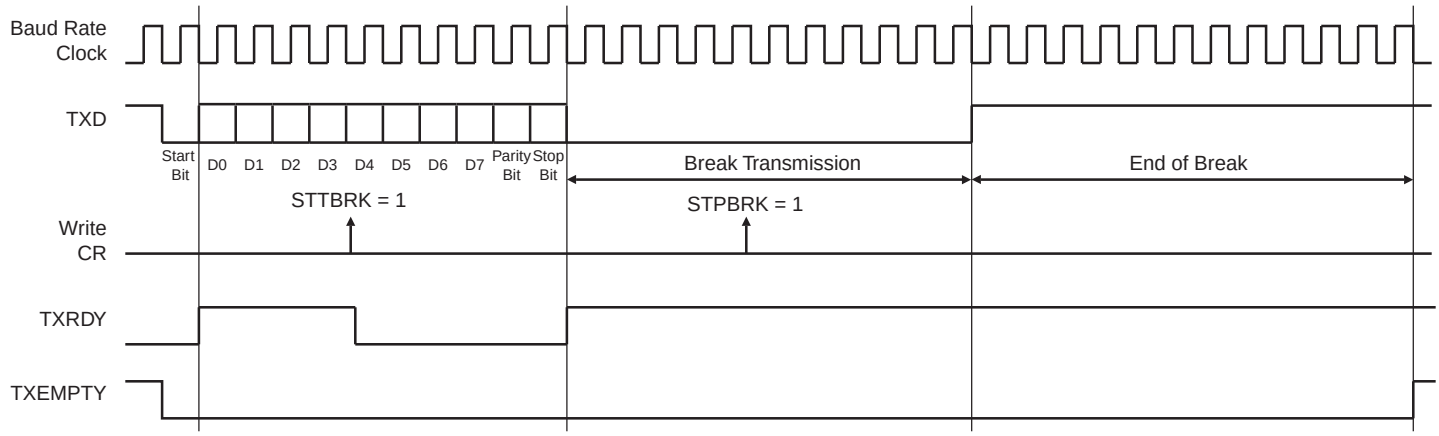


24.6.3.6 Transmit Break

When CSR.TXRDY is set, the user can request the transmitter to generate a break condition on the TXD line by writing a one to the Start Break bit (CR.STTBRK). The break is treated as a normal 0x00 character transmission, clearing CSR.TXRDY and CSR.TXEMPTY, but with zeroes for preambles, start, parity, stop, and time guard bits. Writing a one to the Stop Break bit (CR.STTBRK) will stop the generation of new break characters, and send ones for TG duration or at least 12 bit periods, ensuring that the receiver detects end of break, before resuming normal operation. Figure 24-13 illustrates CR.STTBRK and CR.STPBRK effect on the TXD line.

Writing to CR.STTBRK and CR.STPBRK simultaneously can lead to unpredictable results. Writes to THR before a pending break has started will be ignored.

Figure 24-13. Break Transmission



24.6.3.7 Receive Break

A break condition is assumed when incoming data, parity, and stop bits are zero. This corresponds to a framing error, but CSR.FRAME will remain zero while the Break Received/End of Break bit (CSR.RXBRK) is set. An interrupt request is generated if the Breadk Received/End of Break bit in the Interrupt Mask Register is set (IMR.RXBRK). Writing a one to CR.RSTSTA will clear CSR.RXBRK. An end of break will also set CSR.RXBRK, and is assumed when TX is high for at least 2/16 of a bit period in asynchronous mode, or when a high level is sampled in synchronous mode.

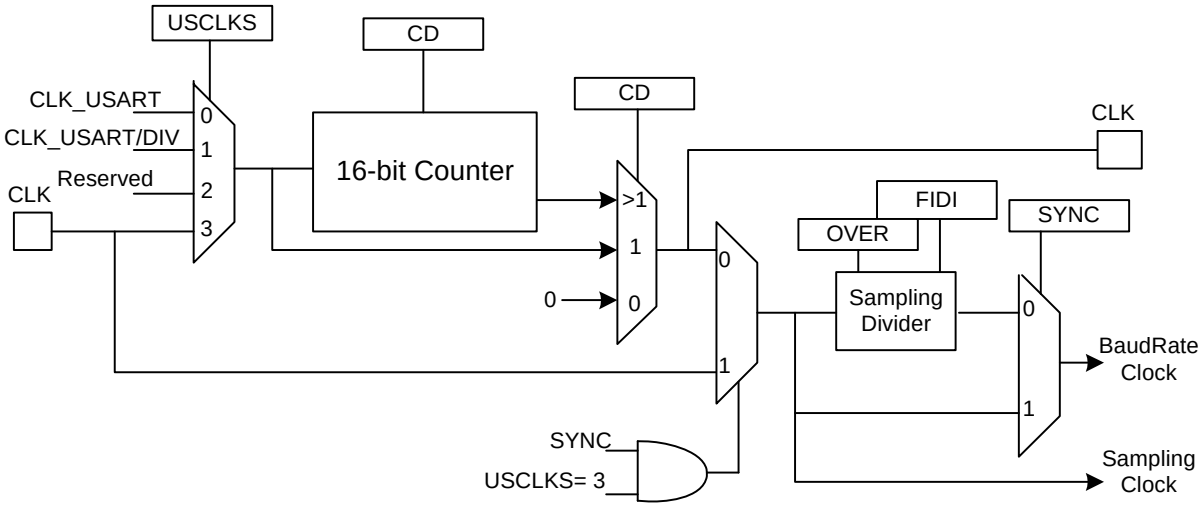
24.6.4 Baud Rate Generator

The baud rate generator provides the bit period clock named the Baud Rate Clock to both receiver and transmitter. It is based on a 16-bit divider, which is specified in the Clock Divider field in the Baud Rate Generator Register (BRGR.CD). A non-zero value enables the generator, and if BRGR.CD is one, the divider is bypassed and inactive. The Clock Selection field in the Mode Register (MR.USCLKS) selects clock source between:

- CLK_USART (internal clock, refer to Power Manager chapter for details)
- CLK_USART/DIV (a divided CLK_USART, refer to Module Configuration section)
- CLK (external clock, available on the CLK pin)

If the external clock CLK is selected, the duration of the low and high levels of the signal provided on the CLK pin must be at least 4.5 times longer than those provided by CLK_USART.

Figure 24-14. Baud Rate Generator



24.6.4.1 Baud Rate in Asynchronous Mode

If the USART is configured to operate in asynchronous mode (MR.SYNC is zero), the selected clock is divided by the BRGR.CD value before it is provided to the receiver as a sampling clock. Depending on the Oversampling Mode bit (MR.OVER) value, the clock is then divided by either 8 (MR.OVER=1), or 16 (MR.OVER=0). The baud rate is calculated with the following formula:

$$BaudRate = \frac{SelectedClock}{(8(2-OVER)CD)}$$

This gives a maximum baud rate of CLK_USART divided by 8, assuming that CLK_USART is the fastest clock available, and that MR.OVER is one.

24.6.4.2 Baud Rate Calculation Example

Table 24-7 shows calculations based on the CD field to obtain 38400 baud from different source clock frequencies. This table also shows the actual resulting baud rate and error.

Table 24-7. Baud Rate Example (OVER=0)

Source Clock (Hz)	Expected Baud Rate (bit/s)	Calculation Result	CD	Actual Baud Rate (bit/s)	Error
3 686 400	38 400	6.00	6	38 400.00	0.00%
4 915 200	38 400	8.00	8	38 400.00	0.00%
5 000 000	38 400	8.14	8	39 062.50	1.70%
7 372 800	38 400	12.00	12	38 400.00	0.00%
8 000 000	38 400	13.02	13	38 461.54	0.16%
12 000 000	38 400	19.53	20	37 500.00	2.40%
12 288 000	38 400	20.00	20	38 400.00	0.00%
14 318 180	38 400	23.30	23	38 908.10	1.31%
14 745 600	38 400	24.00	24	38 400.00	0.00%
18 432 000	38 400	30.00	30	38 400.00	0.00%
24 000 000	38 400	39.06	39	38 461.54	0.16%
24 576 000	38 400	40.00	40	38 400.00	0.00%
25 000 000	38 400	40.69	40	38 109.76	0.76%
32 000 000	38 400	52.08	52	38 461.54	0.16%
32 768 000	38 400	53.33	53	38 641.51	0.63%
33 000 000	38 400	53.71	54	38 194.44	0.54%
40 000 000	38 400	65.10	65	38 461.54	0.16%
50 000 000	38 400	81.38	81	38 580.25	0.47%
60 000 000	38 400	97.66	98	38 265.31	0.35%

The baud rate is calculated with the following formula (MR.OVER=0):

$$BaudRate = \frac{CLK\_USART}{CD \cdot 16}$$

The baud rate error is calculated with the following formula. It is not recommended to work with an error higher than 5%.

$$Error = 1 - \left(\frac{ExpectedBaudRate}{ActualBaudRate}\right)$$

24.6.4.3 Fractional Baud Rate in Asynchronous Mode

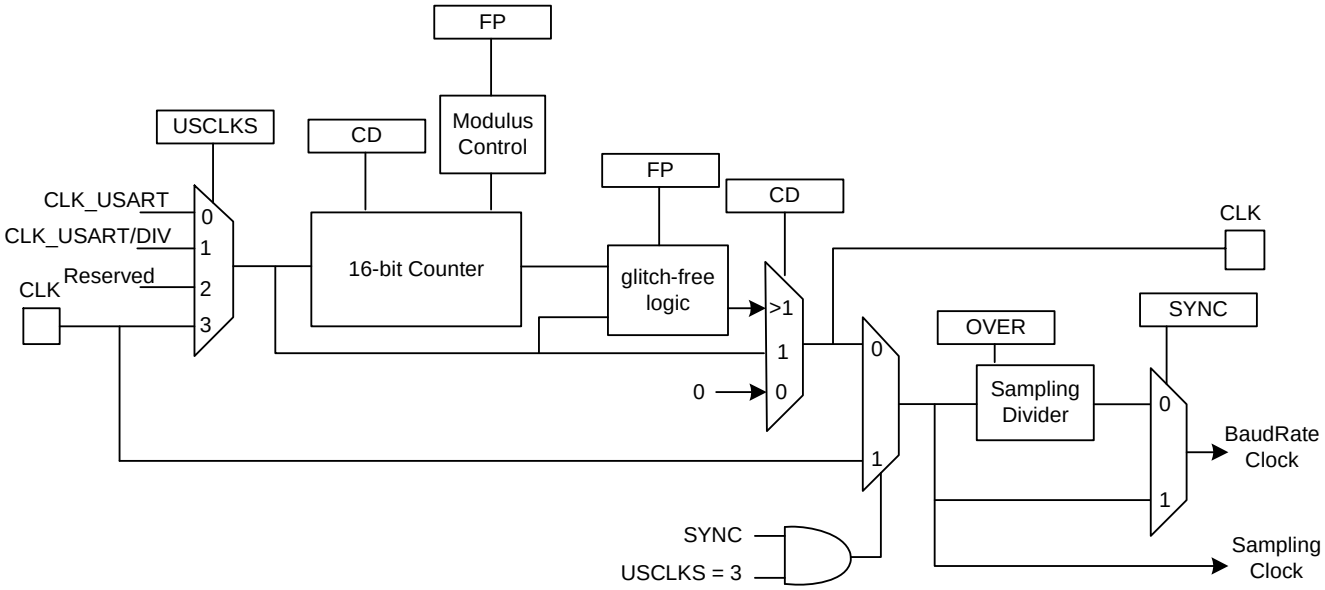
The baud rate generator has a limitation: the source frequency is always a multiple of the baud rate. An approach to this problem is to integrate a high resolution fractional N clock generator, outputting fractional multiples of the reference source clock. This fractional part is selected with

the Fractional Part field in BRGR (BRGR.FP), and is activated by giving it a non-zero value. The resolution is one eighth of CD. The resulting baud rate is calculated using the following formula:

$$BaudRate = \frac{SelectedClock}{\left(8(2-OVER)\left(CD + \frac{FP}{8}\right)\right)}$$

The modified architecture is shown in Figure 24-15.

Figure 24-15. Fractional Baud Rate Generator



24.6.4.4 *Baud Rate in Synchronous and SPI Mode*

If the USART is configured to operate in synchronous mode (MR.SYNC is one), the selected clock is divided by BRGR.CD. This does not apply when the external clock CLK is selected.

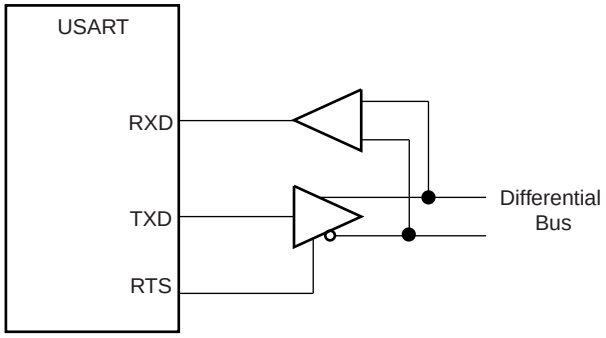
$$BaudRate = \frac{SelectedClock}{CD}$$

When CLK is selected, the frequency of the external clock must be at least 4.5 times lower than the system clock, and when either CLK or CLK_USART/DIV are selected, BRGR.CD must be even to ensure a 50/50 duty cycle. If CLK_USART is selected, the generator ensures this regardless of value.

24.6.5 RS485 Mode

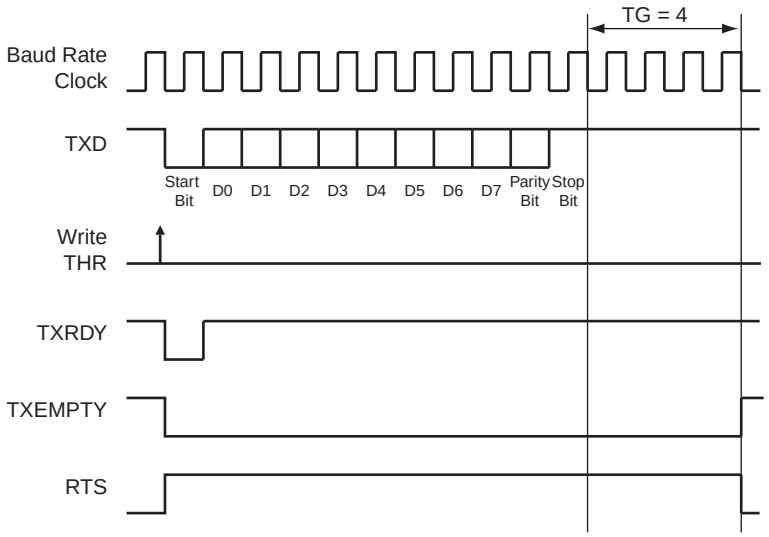
The USART features an RS485 mode, supporting line driver control. This supplements normal synchronous and asynchronous mode by driving the RTS pin high when the transmitter is operating. The RTS pin level is the inverse of the CSR.TXEMPTY value. The RS485 mode is enabled by writing 0x1 to MR.MODE. A typical connection to a RS485 bus is shown in Figure 24-16.

Figure 24-16. Typical Connection to a RS485 Bus



If a timeguard has been configured the RTS pin will remain high for the duration specified in TG, as shown in Figure 24-17.

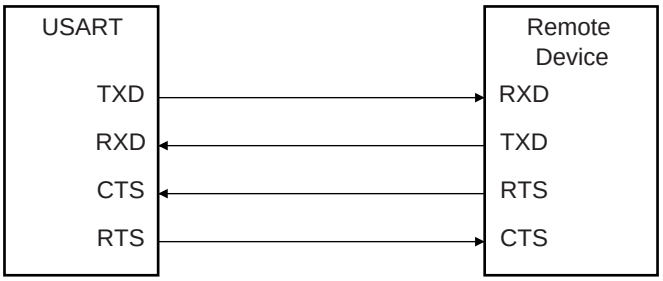
Figure 24-17. Example of RTS Drive with Timeguard Enabled



24.6.6 Hardware Handshaking

The USART features an out-of-band hardware handshaking flow control mechanism, implementable by connecting the RTS and CTS pins with the remote device, as shown in Figure 24-18.

Figure 24-18. Connection with a Remote Device for Hardware Handshaking



Writing 0x2 to the MR.MODE field configures the USART to operate in hardware handshaking mode. The receiver will drive its RTS pin high when disabled or when the Reception Buffer Full bit (CSR.RXBUFF) is set by the Buffer Full signal from the Peripheral DMA controller. If the receiver RTS pin is high, the transmitter CTS pin will also be high and only the active character transmissions will be completed. Allocating a new buffer to the DMA controller by clearing RXBUFF, will drive the RTS pin low, allowing the transmitter to resume transmission. Detected level changes on the CTS pin are reported by the CTS Input Change bit in the Channel Status Register (CSR.CTSIC). An interrupt request is generated if the Input Change bit in the Interrupt Mask Register is set. CSR.CTSIC is cleared when reading CSR.

Figure 24-19 illustrates receiver functionality, and Figure 24-20 illustrates transmitter functionality.

Figure 24-19. Receiver Behavior when Operating with Hardware Handshaking

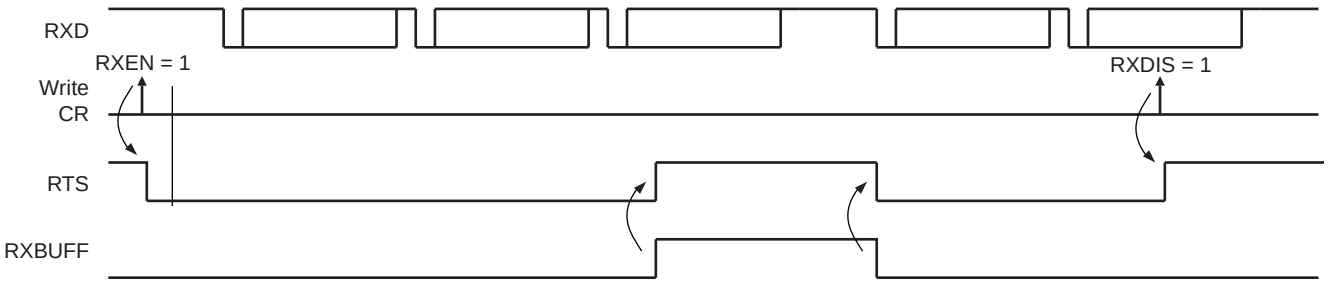
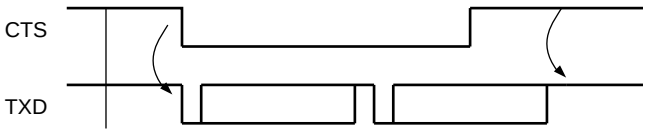


Figure 24-20. Transmitter Behavior when Operating with Hardware Handshaking



24.6.7 Modem Mode

The USART features a modem mode, supporting asynchronous communication with the following signal pins: Data Terminal Ready (DTR), Data Set Ready (DSR), Request to Send (RTS), Clear to Send (CTS), Data Carrier Detect (DCD), and Ring Indicator (RI). Modem mode is enabled by writing 0x3 to MR.MODE. The USART will behave as a Data Terminal Equipment (DTE), controlling DTR and RTS, while detecting level changes on DSR, DCD, CTS, and RI.

Table 24-8 shows USART signal pins with the corresponding standardized modem connections.

Table 24-8. Circuit References

USART Pin	V.24	CCITT	Direction
TXD	2	103	From terminal to modem
RTS	4	105	From terminal to modem
DTR	20	108.2	From terminal to modem
RXD	3	104	From modem to terminal
CTS	5	106	From terminal to modem

Table 24-8. Circuit References

USART Pin	V.24	CCITT	Direction
DSR	6	107	From terminal to modem
DCD	8	109	From terminal to modem
RI	22	125	From terminal to modem

The DTR pin is controlled by the DTR enable and disable bits in CR (CR.DTREN and CR.DTRDIS). Writing a one to CR.DTRDIS drives DTR high, and writing a one to CR.DTREN drives DTR low. The RTS pin is controlled automatically.

Detected level changes are reported by the respective Input Change bits in CSR (CSR.RIIC, CSR.DSRIC, CSR.DCDIC, and CSR.CTSIC). An interrupt request is generated if the corresponding bit in the Interrupt Mask Register is set. The Input Change bits in CSR are automatically cleared when CSR is read. When the CTS pin goes high, the USART will wait for the transmitter to complete any ongoing character transmission before automatically disabling it.

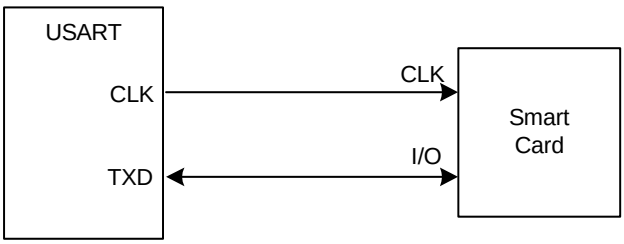
24.6.8 ISO7816 Mode

The USART features an ISO7816 compatible mode, enabling interfacing with smart cards and Security Access Modules (SAM) through an ISO7816 compliant link. T=0 and T=1 protocols, as defined in the ISO7816 standard, are supported. The ISO7816 mode is selected by writing the value 0x4 (T=0 protocol) or 0x6 (T=1 protocol) to MR.MODE.

24.6.8.1 ISO7816 Mode Overview

ISO7816 specifies half duplex communication on one bidirectional line. The baud rate is a fraction of the clock provided by the master on the CLK pin (see "Baud Rate Generator" on page 587). The USART connects to a smart card as shown in Figure 24-21. The TXD pin is bidirectional and is routed to the receiver when the transmitter is disabled. Having both receiver and transmitter enabled simultaneously may lead to unpredictable results.

Figure 24-21. USART (Master) Connected to a Smart Card



In both T=0 and T=1 modes, the character format is fixed to eight data bits, and one or two stop bits, regardless of CHRL, MODE9, and CHMODE values. Parity according to specification is even. If the inverse transmission format is used, where payload data bits are transmitted inverted on the I/O line, the user can either use odd parity and perform an XOR on data headed to THR and coming from RHR, or write a one to MR.INVDATA and let the device handle the rest.

24.6.8.2 Baud Rate in ISO 7816 Mode

The ISO7816 specification defines the bit rate with the following formula:

$$B = \frac{Di}{Fi} \times f$$

where:

- B is the bit rate
- Di is the bit-rate adjustment factor
- Fi is the clock frequency division factor
- f is the ISO7816 clock frequency (Hz)

Di is a binary value encoded on a 4-bit field, named DI, as represented in Table 24-9.

Table 24-9. Binary and Decimal Values for Di

DI field	0001	0010	0011	0100	0101	0110	1000	1001
Di (decimal)	1	2	4	8	16	32	12	20

Fi is a binary value encoded on a 4-bit field, named FI, as represented in Table 24-10.

Table 24-10. Binary and Decimal Values for Fi

FI field	0000	0001	0010	0011	0100	0101	0110	1001	1010	1011	1100	1101
Fi (decimal	372	372	558	744	1116	1488	1860	512	768	1024	1536	2048

Table 24-11 shows the resulting Fi/Di ratio, which is the ratio between the ISO7816 clock and the Baud Rate Clock.

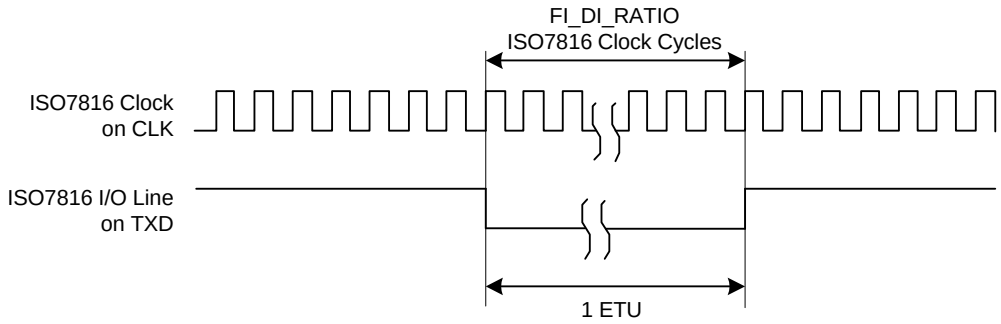
Table 24-11. Possible Values for the Fi/Di Ratio

Fi	372	558	744	1116	1488	1860	512	768	1024	1536	2048
Di=2	186	279	372	558	744	930	256	384	512	768	1024
Di=4	93	139.5	186	279	372	465	128	192	256	384	512
Di=8	46.5	69.75	93	139.5	186	232.5	64	96	128	192	256
Di=16	23.25	34.87	46.5	69.75	93	116.2	32	48	64	96	128
Di=32	11.62	17.43	23.25	34.87	46.5	58.13	16	24	32	48	64
Di=12	31	46.5	62	93	124	155	42.66	64	85.33	128	170.6
Di=20	18.6	27.9	37.2	55.8	74.4	93	25.6	38.4	51.2	76.8	102.4

The clock selected by MR.USCLKS can be output on the CLK pin to feed the smart card clock inputs. To output the clock, the user must write a one to the Clock Output Select bit in MR (MR.CLKO). The clock is divided by BRGR.CD before it is output on the CLK pin. If CLK is selected as clock source in MR.USCLKS, the clock can not be output on the CLK pin.

The selected clock is divided by the FI Over DI Ratio Value field in the FI DI Ratio Register (FIDI.FI_DI_RATIO), which can be up to 2047 in ISO7816 mode. This will be rounded off to an integral so the user has to select a FI_DI_RATIO value that comes as close as possible to the expected Fi/Di ratio. The FI_DI_RATIO reset value is 0x174 (372 in decimal) and is the most common divider between the ISO7816 clock and bit rate (Fi=372, Di=1). Figure 24-22 shows the relationship between the Elementary Time Unit (ETU), corresponding to a bit period, and the ISO 7816 clock.

Figure 24-22. Elementary Time Unit (ETU)



24.6.8.3 Protocol T=0

In T=0 protocol, a character is made up of one start bit, eight data bits, one parity bit, and a two bit period guard time. During the guard time, the line will be high if the receiver does not signal a parity error, as shown in Figure 24-23. The receiver signals a parity error, aka non-acknowledge (NACK), by pulling the line low for a bit period within the guard time, resulting in the total character length being incremented by one, see Figure 24-24. The USART will not load data to RHR if it detects a parity error, and will set PARE if it receives a NACK.

Figure 24-23. T=0 Protocol without Parity Error

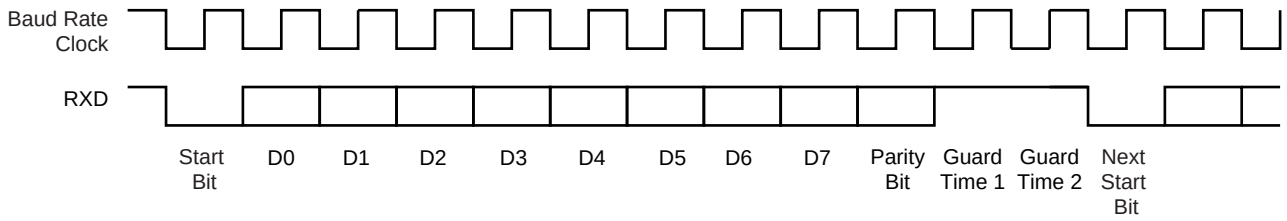
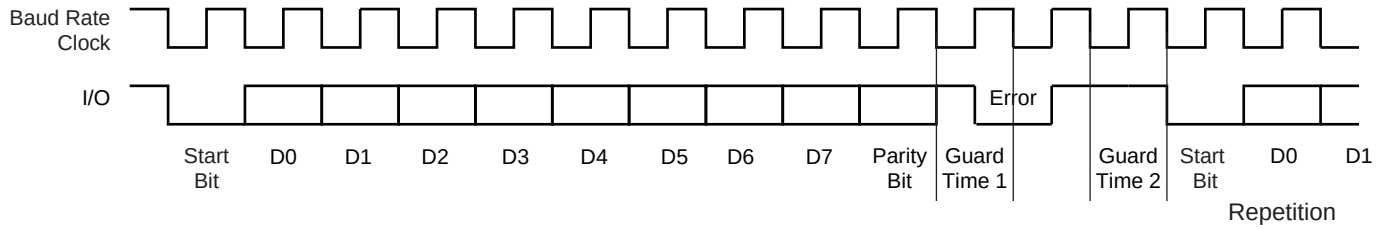


Figure 24-24. T=0 Protocol with Parity Error



24.6.8.4 Protocol T=1

In T=1 protocol, the character resembles an asynchronous format with only one stop bit. The parity is generated when transmitting and checked when receiving. Parity errors set PARE.

24.6.8.5 Receive Error Counter

The USART receiver keeps count of up to 255 errors in the Number Of Errors field in the Number of Error Register (NER.NB_ERRORS). Reading NER automatically clears NB_ERRORS.

24.6.8.6 Receive NACK Inhibit

The USART can be configured to ignore parity errors by writing a one to the Inhibit Non Acknowledge bit (MR.INACK). Erroneous characters will be treated as if they were ok, not generating a NACK, loaded to RHR, and raising RXRDY.

24.6.8.7 *Transmit Character Repetition*

The USART can be configured to automatically re-send a character if it receives a NACK. Writing a non-zero value to MR.MAX_ITERATION will enable and determine the number of consecutive re-transmissions. If the number of unsuccessful re-transmissions equals MAX_ITERATION, the iteration bit (CSR.ITER) is set. An interrupt request is generated if the ITER bit in the Interrupt Mask Register (IMR.ITER) is set. Writing a one to the Reset Iteration bit (CR.RSTIT) will clear CSR.ITER.

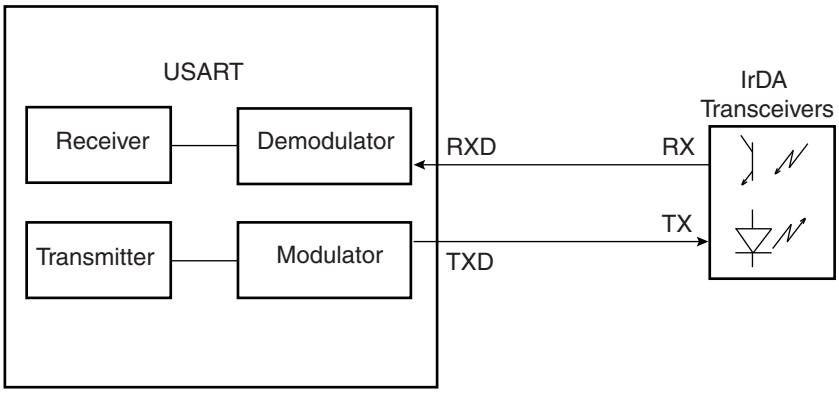
24.6.8.8 *Disable Successive Receive NACK*

The receiver can limit the number of consecutive NACKs to the value in MR.MAX_ITERATION. This is enabled by writing a one to the Disable Successive NACK bit (MR.DSNACK). If the number of NACKs is about to exceed MR.MAX_ITERATION, the character will instead be accepted as valid and CSR.ITER is set.

24.6.9 IrDA Mode

The USART features an IrDA mode, supporting asynchronous, half-duplex, point-to-point wireless communication. It embeds the modulator and demodulator, allowing for a glueless connection to the infrared transceivers, as shown in Figure 24-25. The IrDA mode is enabled by writing 0x8 to MR.MODE. This activates the IrDA specification v1.1 compliant modem. Data transfer speeds ranging from 2.4Kbit/s to 115.2Kbit/s are supported and the character format is fixed to one start bit, eight data bits, and one stop bit.

Figure 24-25. Connection to IrDA Transceivers



The receiver and the transmitter must be exclusively enabled or disabled, according to the direction of the transmission. To receive IrDA signals, the following needs to be done:

- Disable TX and enable RX.
- Configure the TXD pin as an I/O, outputting zero to avoid LED activation. Disable the internal pull-up for improved power consumption.
- Receive data.

24.6.9.1 IrDA Modulation

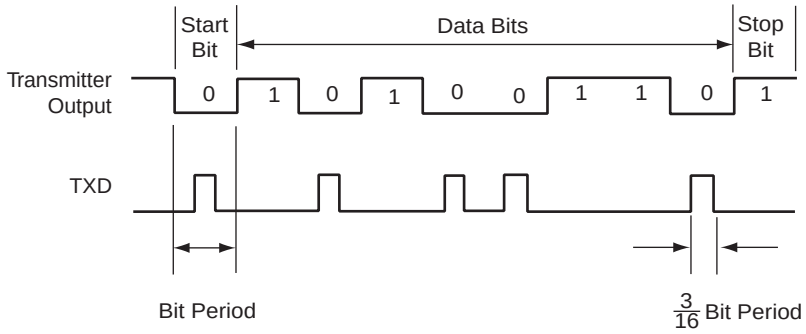
The RZI modulation scheme is used, where a zero is represented by a light pulse 3/16 of a bit period, and no pulse to represent a one. Some examples of signal pulse duration are shown in Table 24-12.

Table 24-12. IrDA Pulse Duration

Baud Rate	Pulse Duration (3/16)
2.4 Kbit/s	78.13 µs
9.6 Kbit/s	19.53 µs
19.2 Kbit/s	9.77 µs
38.4 Kbit/s	4.88 µs
57.6 Kbit/s	3.26 µs
115.2 Kbit/s	1.63 µs

Figure 24-26 shows an example of character transmission.

Figure 24-26. IrDA Modulation



24.6.9.2 IrDA Baud Rate

As the IrDA mode shares some logic with the ISO7816 mode, the FIDI.FI_DI_RATIO field must be configured correctly. See “Manchester Encoder/Decoder” on page 612. Table 24-13 shows some examples of BRGR.CD values, baud rate error, and pulse duration. Note that the maximal acceptable error rate of ±1.87% must be met.

Table 24-13. IrDA Baud Rate Error

Peripheral Clock	Baud Rate	CD	Baud Rate Error	Pulse Time
3 686 400	115 200	2	0.00%	1.63
20 000 000	115 200	11	1.38%	1.63
32 768 000	115 200	18	1.25%	1.63
40 000 000	115 200	22	1.38%	1.63
3 686 400	57 600	4	0.00%	3.26
20 000 000	57 600	22	1.38%	3.26
32 768 000	57 600	36	1.25%	3.26
40 000 000	57 600	43	0.93%	3.26

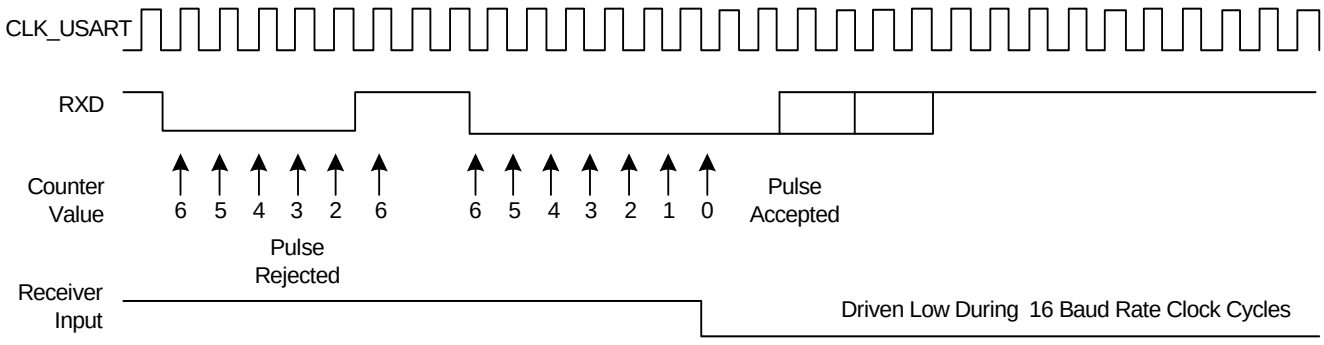
Table 24-13. IrDA Baud Rate Error (Continued)

Peripheral Clock	Baud Rate	CD	Baud Rate Error	Pulse Time
3 686 400	38 400	6	0.00%	4.88
20 000 000	38 400	33	1.38%	4.88
32 768 000	38 400	53	0.63%	4.88
40 000 000	38 400	65	0.16%	4.88
3 686 400	19 200	12	0.00%	9.77
20 000 000	19 200	65	0.16%	9.77
32 768 000	19 200	107	0.31%	9.77
40 000 000	19 200	130	0.16%	9.77
3 686 400	9 600	24	0.00%	19.53
20 000 000	9 600	130	0.16%	19.53
32 768 000	9 600	213	0.16%	19.53
40 000 000	9 600	260	0.16%	19.53
3 686 400	2 400	96	0.00%	78.13
20 000 000	2 400	521	0.03%	78.13
32 768 000	2 400	853	0.04%	78.13

24.6.9.3 IrDA Demodulator

The demodulator depends on an 8-bit down counter loaded with the value in the IRDA_Filter field in the IrDA Filter Register (IFR.IRDA_FILTER). When a falling edge on RXD is detected, the counter starts decrementing at CLK_USART speed. If a rising edge on RXD is detected , the counter stops and is reloaded with the IrD Filter value. If no rising edge has been detected when the counter reaches zero, the receiver input is pulled low during one bit period, see Figure 24-27. Writing a one to the Infrared Receive Line Filter bit (MR.FILTER), enables a noise filter that, instead of using just one sample, will choose the majority value from three consecutive samples.

Figure 24-27. IrDA Demodulator Operations



24.6.10 LIN Mode

The USART features a Local Interconnect Network (LIN) 1.3 and 2.0 compliant mode, embedding full error checking and reporting, automatic frame processing with up to 256 data bytes, customizable response data lengths, and requiring minimal CPU resources. The LIN mode is enabled by writing 0xA (master) or 0xB (slave) to MR.MODE.

24.6.10.1 Modes of Operation

Changing LIN mode after initial configuration must be followed by a transceiver software reset in order to avoid unpredictable behavior.

24.6.10.2 Receiver and Transmitter Control

See “Receiver and Transmitter Control” on page 580.

24.6.10.3 Baud Rate Configuration

The LIN nodes baud rate is configured in the Baud Rate Generator Register (BRGR), See “Baud Rate in Asynchronous Mode” on page 587. The LIN slave node copies the initial BRGR value to the LIN Baud Rate Register (LINBRR), and updates LINBRR after successful synchronization.

24.6.10.4 Character Transmission and Reception

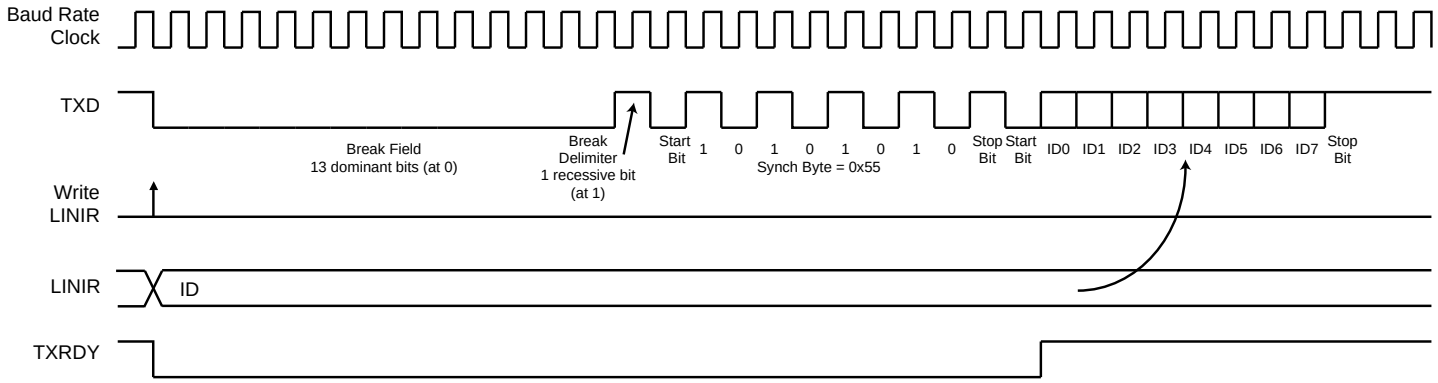
See “Transmitter Operations” on page 580, and “Receiver Operations” on page 582.

24.6.10.5 Header Transmission (Master Node Configuration)

All LIN frames start with a header sent by the master. As soon as the identifier has been written to the Identifier Character field in the LIN Identifier Register (LINIR.IDCHR), CSR.TXRDY is cleared and the header is sent. The header consists of a Break field, a Sync field, and an Identifier field. CSR.TXRDY is set when the identifier has been transferred into the transmitters shift register. An interrupt request is generated if IMR.TXRDY is set.

The Break field consists of 13 dominant bits (the break) and one recessive bit (the break delimiter). The Sync field consists of a start bit, the Sync byte (the character 0x55), and a stop bit, refer to Figure 24-30. The Identifier field contains the Identifier as written to LINIR.IDCHR. The identifier parity bits can be generated automatically (see Section 24.6.10.8).

Figure 24-28. Header Transmission



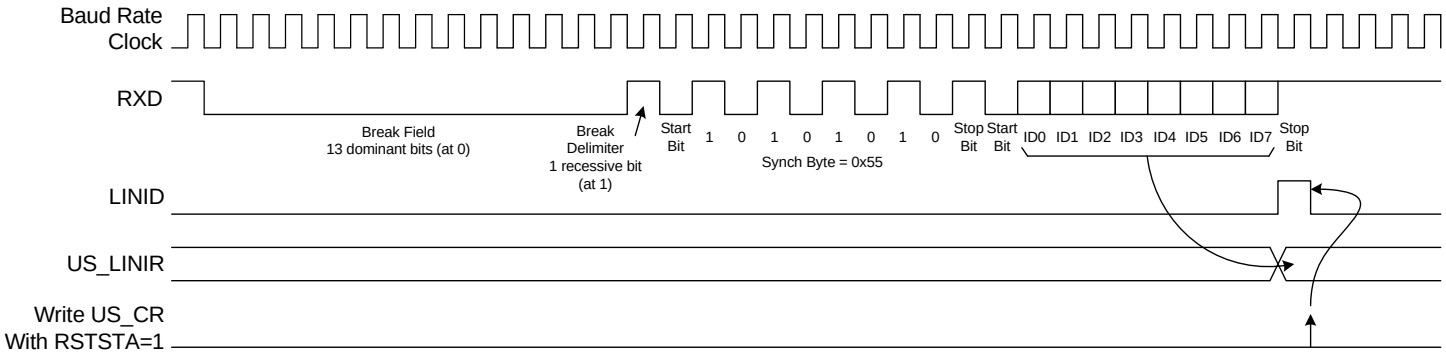
See also “Master Node Configuration” on page 604.

24.6.10.6 Header Reception (Slave Node Configuration)

The USART stays idle until it detects a break field, consisting of at least 11 consecutive dominant bits (zeroes) on the bus. A received break will set the Lin Break bit (CSR.LINBK). An interrupt request is generated if the Lin Break bit in the Interrupt Mask Register (IMR.LINBK) is set. The Sync field is used to synchronize the baud rate (see Section 24.6.10.7). IDCHR is updated and the LIN Identifier bit (CSR.LINID) is set when the Identifier has been received. An interrupt request is generated if the Lin Identifier bit in the Interrupt Mask Register (IMR.LINID) is set. The Identifier parity bits can be automatically checked (see Section 24.6.10.8). If the header

is not received within the time defined in THeader_Maximum, the Lin Header Time-out error (CSR.LINHTE) is generated (see Section 24.6.10.13). An interrupt request is generated if IMR.LINHTE is set. Writing a one to CR.RSTSTA will clear CSR.LINHTE, CSR.LINBK, and CSR.LINID.

Figure 24-29. Header Reception

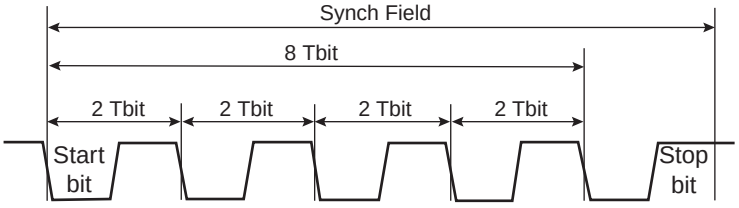


See also "Slave Node Configuration" on page 606.

24.6.10.7 *Slave Node Synchronization*

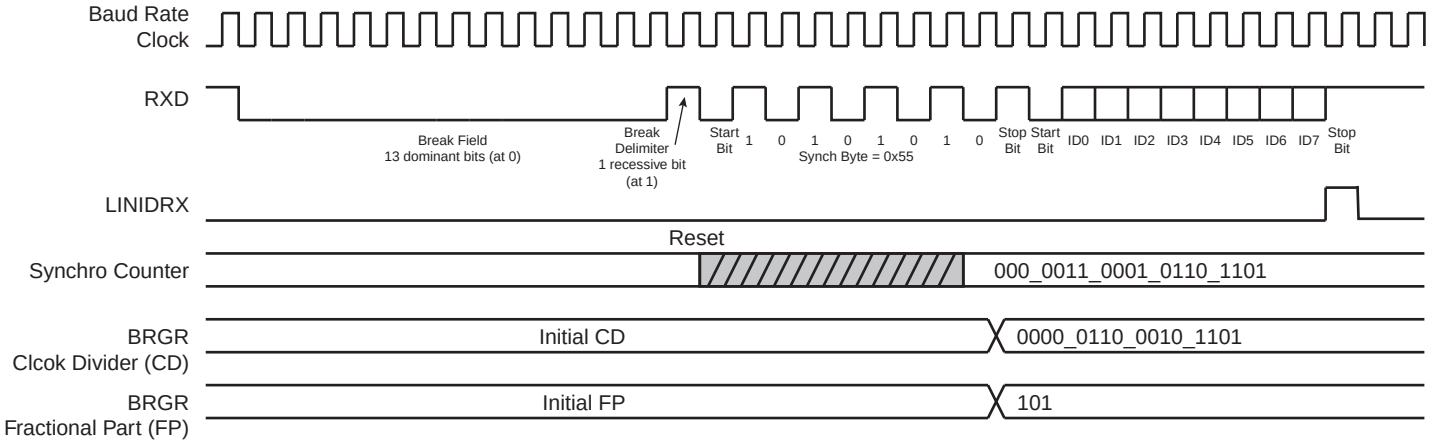
Synchronization is only done by the slave, and can be disabled by writing a one to the Synchronization Disable bit in the LIN Mode Register (LINMR.SYNCDIS). If the Sync byte is not 0x55, an Inconsistent Sync Field error is generated, and the LIN Inconsistend Sync Field Error bit in CSR (CSR.LINISFE) is set. An interrupt request is generated if the LINISFE bit in IMR is set. CSR.LINISFE is cleared by writing a one to CR.RSTSTA. The time between falling edges is measured by a 19-bit counter, driven by the sampling clock (see Section 24.6.4).

Figure 24-30. Sync Field



The counter starts when the Sync field start bit is detected, and continues for eight bit periods. If the deviation is within ±15% (see below), the 16 most significant bits (counter value divided by 8) becomes the new clock divider (LINBRR.LINCD), and the three least significant bits (the remainder) becomes the new fractional part (LINBRR.LINFP).

Figure 24-31. Slave Node Synchronization



The synchronization accuracy depends on:

- The theoretical slave node clock frequency; nominal clock frequency (F_{Nom})
- The baud rate
- The oversampling mode (OVER=0 => 16x, or OVER=1 => 8x)

The following formula is used to calculate synchronization deviation, where F_{SLAVE} is the real slave node clock frequency, and F_{TOL_UNSYNC} is the difference between F_{Nom} and F_{SLAVE}. According to the LIN specification, $F_{TOL\_UNSYNCH}$ may not exceed ±15%, and the bit rates between two nodes must be within ±2% of each other, resulting in a maximal BaudRate_deviation of ±1%. If $F_{TOL\_UNSYNCH}$ exceeds ±15%, LINBRR will not be updated with new values, and the LIN Sync Tolerance Error bit (CSR.LINSTE) is set.

$$\text{BaudRate\_deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{OVER}) + \beta] \times \text{BaudRate}}{8 \times F_{SLAVE}}\right)\%$$

$$\text{BaudRate\_deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{OVER}) + \beta] \times \text{BaudRate}}{8 \times \left(\frac{F_{TOL\_UNSYNC}}{100}\right) \text{x} F_{Nom}}\right)\%$$

$$-0{,}5 \le \alpha \le +0{,}5 \quad -1 < \beta < +1$$

Minimum nominal clock frequency with a fractional part:

$$F_{Nom}(\text{min}) = \left(100 \times \frac{[0{,}5 \times 8 \times (2 - \text{OVER}) + 1] \times \text{BaudRate}}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%}\right)\text{Hz}$$

Examples:

- Baud rate = 20 kbit/s, OVER=0 (Oversampling 16x) => F_{Nom}(min) = 2.64MHz
- Baud rate = 20 kbit/s, OVER=1 (Oversampling 8x) => F_{Nom}(min) = 1.47MHz
- Baud rate = 1 kbit/s, OVER=0 (Oversampling 16x) => F_{Nom}(min) = 132 kHz
- Baud rate = 1 kbit/s, OVER=1 (Oversampling 8x) => F_{Nom}(min) = 74 kHz

If the fractional part is not used, the synchronization accuracy is much lower. The 16 most significant bits, added with the first least significant bit, becomes the new clock divider (LINCD). The equation of the baud rate deviation is the same as above, but the constants are:

$$-4 \le \alpha \le +4 \quad -1 < \beta < +1$$

Minimum nominal clock frequency without a fractional part:

$$F_{Nom}(\min) = \left(100 \times \frac{[4 \times 8 \times (2 - OVER) + 1] \times Baudrate}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%} \right) Hz$$

Examples:

- Baud rate = 20 kbit/s, OVER=0 (Oversampling 16x) => F_{Nom}(min) = 19.12MHz
- Baud rate = 20 kbit/s, OVER=1 (Oversampling 8x) => F_{Nom}(min) = 9.71MHz
- Baud rate = 1 kbit/s, OVER=0 (Oversampling 16x) => F_{Nom}(min) = 956 kHz
- Baud rate = 1 kbit/s, OVER=1 (Oversampling 8x) => F_{Nom}(min) = 485 kHz

24.6.10.8 Identifier Parity

An identifier field consists of two sub-fields; the identifier and its parity. Bits 0 to 5 are assigned to the identifier, while bits 6 and 7 are assigned to parity. Automatic parity management is enabled by default, and can be disabled by writing a one to the Parity Disable bit in the LIN Mode register (LINMR.PARDIS).

- LINMR.PARDIS=0: During header transmission, the parity bits are computed and in the shift register they replace bits 6 and 7 from LINIR.IDCHR. During header reception, the parity bits are checked and can generate a LIN Identifier Parity Error (see Section 24.6.10.13). Bits 6 and 7 in LINIR.IDCHR read as zero when receiving.
- LINMR.PARDIS=1: During header transmission, all the bits in LINIR.IDCHR are sent on the bus. During header reception, all the bits in LINIR.IDCHR are updated with the received Identifier.

24.6.10.9 Node Action

After an identifier transaction, a LIN response mode must be selected. This is done in the Node Action field (LINMR.NACT). Below are some response modes exemplified in a small LIN cluster:

- Response, from master to slave1:

  Master: NACT=PUBLISH

  Slave1: NACT=SUBSCRIBE

  Slave2: NACT=IGNORE

- Response, from slave1 to master:

  Master: NACT=SUBSCRIBE

  Slave1: NACT=PUBLISH

  Slave2: NACT=IGNORE

- Response, from slave1 to slave2:

  Master: NACT=IGNORE

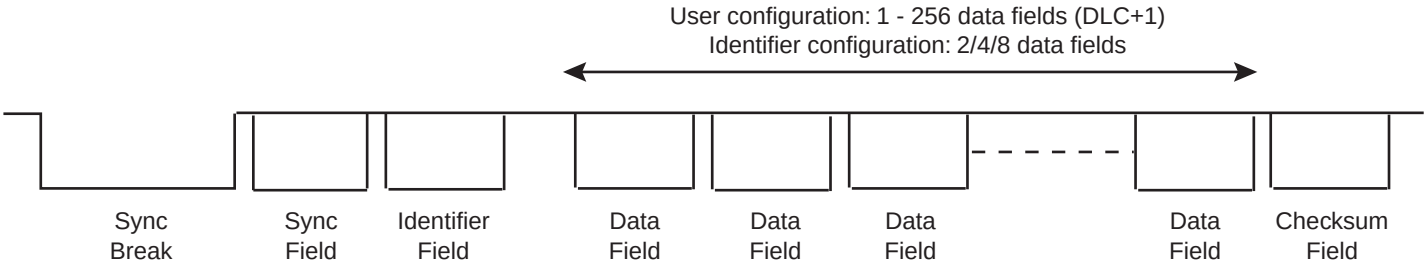
  Slave1: NACT=PUBLISH

  Slave2: NACT=SUBSCRIBE

24.6.10.10 LIN Response Data Length

The response data length is the number of data fields (bytes), excluding the checksum.

Figure 24-32. Response Data Length



The response data length can be configured, either by the user, or automatically by bits 4 and 5 in the Identifier (LINIR.IDCHR), in accordance to LIN 1.1. The user selects one of these modes by writing to the Data Length Mode bit (LINMR.DLM):

- LINMR.DLM=0: the response data length is configured by the user by writing to the 8-bit Data Length Control field (LINMR.DLC). The response data length equals DLC + 1 bytes.
- LINMR.DLM=1: the response data length is defined by the Identifier (LINIR.IDCHR) bits according to the table below.

Table 24-14. Response Data Length if DLM = 1

LINIR.IDCHR[5]	LINIR.IDCHR[4]	Response Data Length [bytes]
0	0	2
0	1	2
1	0	4
1	1	8

24.6.10.11 Checksum

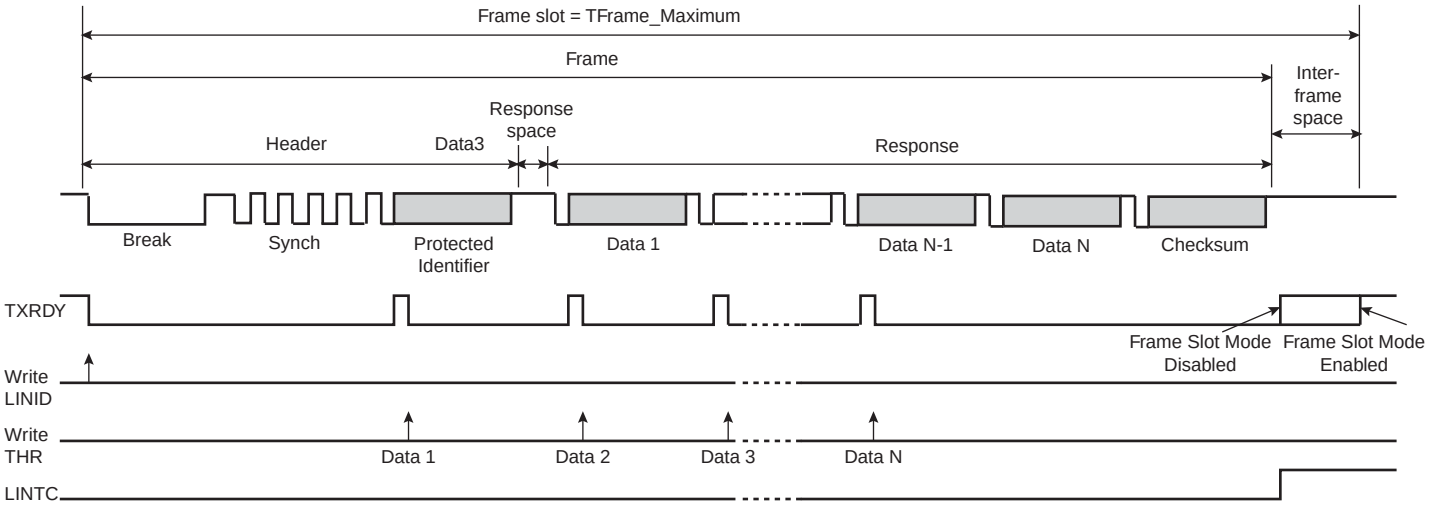
The last frame field is the checksum. It is configured by the Checksum Type (LINMR.CHKTYP), and the Checksum Disable (LINMR.CHKDIS) bits. CSR.TXRDY will not be set after the last THR data write if enabled. Writing a one to LINMR.CHKDIS will disable the automatic checksum generation/checking, and the user may send/check this last byte manually, disguised as a normal data. The checksum is an inverted 8-bit sum with carry, either:

- Over all data bytes, called a classic checksum. This is used for LIN 1.3 compliant slaves, and automatically managed when CHKDIS=0, and CHKTYP=1.
- Over all data bytes and the protected identifier, called an enhanced checksum. This is used for LIN 2.0 compliant slaves, and automatically managed when CHKDIS=0, and CHKTYP=0.

24.6.10.12 Frame Slot Mode

A LIN master can be configured to use frame slots with a pre-defined minimum length. This Frame Slot mode is enabled by default, and is disabled by writing a one to the Frame Slot Mode Disable bit (LINMR.FSDIS). The Frame Slot mode will not allow CSR.TXRDY to be set after a frame transfer until the entire frame slot duration has elapsed, in effect preventing the master from sending a new header. The LIN Transfer Complete bit (CSR.LINTC) will still be set after the checksum has been sent. An interrupt is generated if the LIN Transfer Complete bit in the Interrupt Mask Register (IMR.LINTC) is set. Writing a one to CR.RSTSTA clears CSR.LINTC.

Figure 24-33. Frame Slot Mode with Automatic Checksum



The minimum frame slot size is determined by TFrame_Maximum, and calculated below (all values in bit periods):

- THeader_Nominal = 34
- TFrame_Maximum = 1.4 x (THeader_Nominal + TResponse_Nominal + 1)

Note: The term “+1” leads to an integer result for TFrame_Max (LIN Specification 1.3)

If the Checksum is sent (CHKDIS=0):

- TResponse_Nominal = 10 x (NData + 1)
- TFrame_Maximum = 1.4 x (34 + 10 x (DLC + 1 + 1) + 1)
- TFrame_Maximum = 77 + 14 x DLC

If the Checksum is not sent (CHKDIS=1):

- TResponse_Nominal = 10 x NData
- TFrame_Maximum = 1.4 x (34 + 10 x (DLC + 1) + 1)
- TFrame_Maximum = 63 + 14 x DLC

24.6.10.13 LIN Errors

This section describes the errors generated in LIN mode, and the coresponding error bits in CSR. The error bits are cleared by writing a one to CR.RSTSTA. An interrupt request is generated if the corresponding bit in the Interrupt Mask Register (IMR) is set. This bit is set by writing a one to the corresponding bit in the Interrupt Enable Register (IER).

- Header Time-out Error (CSR.LINHTE)
  - This error is generated if the Header is not received within the time specified by THeader_Maximum.
- Sync Tolerance Error (CSR.LINSTE)
  - This error is generated if the synchronization baudrate deviation is larger than the maximum tolerance, FTol_Unsync ±15%. The synchronization procedure is aborted.
- Slave Not Responding Error (CSR.LINSNRE)

  - This error is generated if no valid message appears within the TFrame_Maximum time frame slot, while the USART is expecting a response from another node (NACT=SUBSCRIBE).
- Checksum Error (CSR.LINCE)
  - This error is generated if the received checksum is wrong. This error can only be generated if the checksum feature is enabled (CHKDIS=0).
- Identifier Parity Error (CSR.LINIPE)
  - This error is generated if the identifier parity is wrong. This error can only be generated if parity is enabled (PARDIS=0).
- Inconsistent Sync Field Error (CSR.LINISFE)
  - This error is generated in slave mode if the Sync Field character received is not 0x55. Synchronization procedure is aborted.
- Bit Error (CSR.LINBE)
  - This error is generated if the value transmitted by the USART on Tx differs from the value sampled on Rx. If a bit error is detected, the transmission is aborted at the next byte border.

24.6.11 LIN Frame Handling

24.6.11.1 Master Node Configuration

- Configure the baud rate by writing to BRGR.CD and BRGR.FP
- Configure the frame transfer by writing to the LINMR fields NACT, PARDIS, CHKDIS, CHKTYPE, DLM, FSDIS, and DLC
- Select LIN mode and master node by writing 0xA to MR.MODE
- Write a one to CR.TXEN and CR.RXEN to enable both transmitter and receiver
- Wait until CSR.TXRDY is one
- Send the header by writing to LINIR.IDCHR

The following procedure depends on the LINMR.NACT setting:

- Case 1: LINMR.NACT is 0x0 (PUBLISH, the USART transmits the response)
  - Wait until CSR.TXRDY is one
  - Send a byte by writing to THR.TXCHR
  - Repeat the two previous steps until there is no more data to send
  - Wait until CSR.LINTC is one
  - Check for LIN errors
- Case 2: LINMR.NACT is 0x1 (SUBSCRIBE, the USART receives the response)
  - Wait until CSR.RXRDY is one
  - Read RHR.RXCHR
  - Repeat the two previous steps until there is no more data to read
  - Wait until CSR.LINTC is one
  - Check for LIN errors
- Case 3: LINMR.NACT is 0x2 (IGNORE, the USART is not concerned by a response)
  - Wait until CSR.LINTC is one
  - Check for LIN errors

Figure 24-34. Master Node Configuration, LINMR.NACT is 0x0 (PUBLISH)

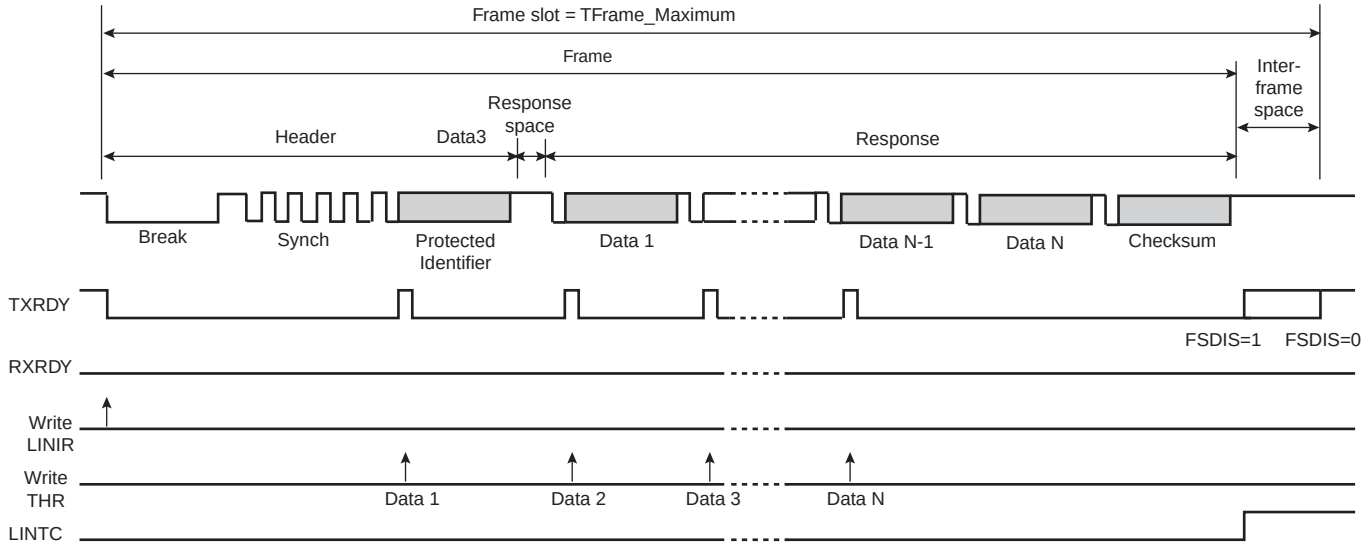


Figure 24-35. Master Node Configuration, LINMR.NACT is 0x1 (SUBSCRIBE)

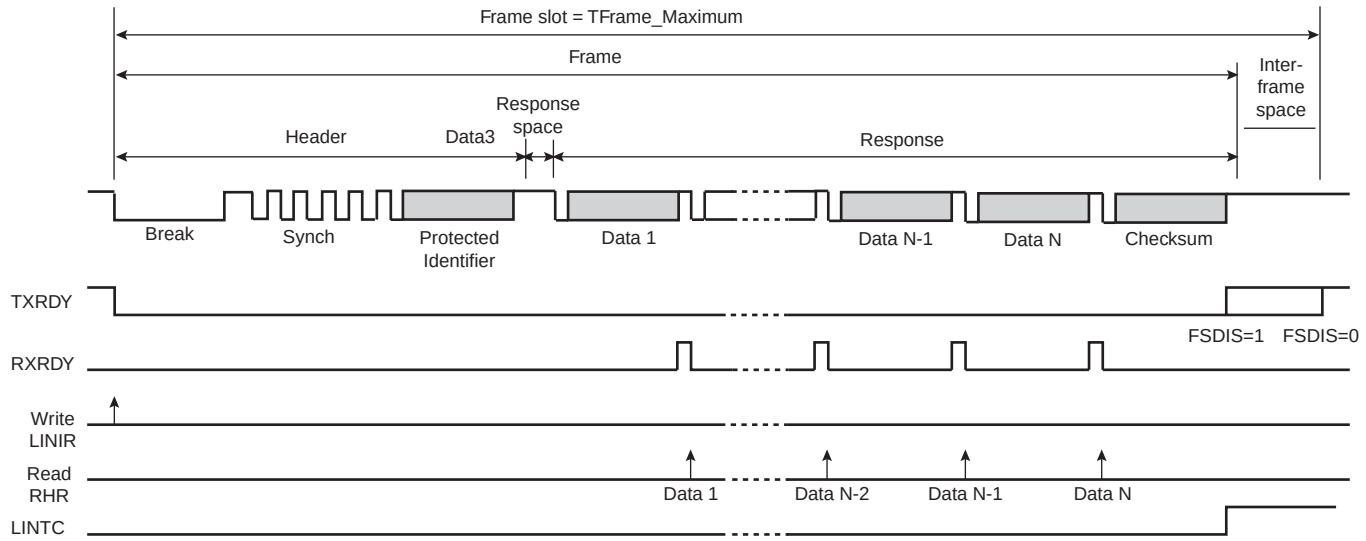
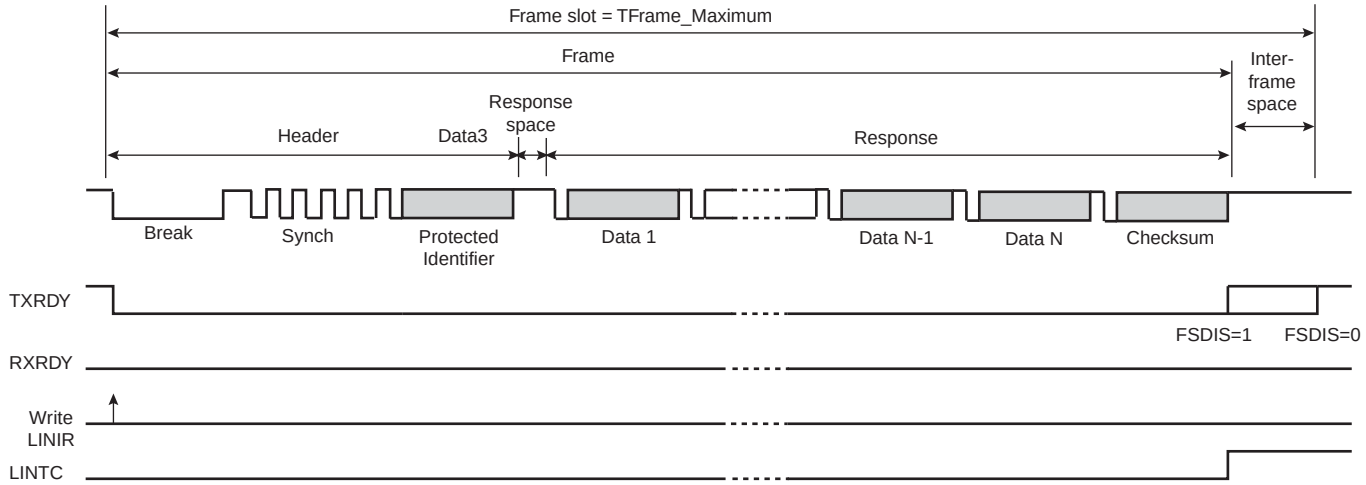


Figure 24-36. Master Node Configuration, LINMR.NACT is 0x2 (IGNORE)



24.6.11.2 Slave Node Configuration

- Configure the baud rate by writing to BRGR.CD and BRGR.FP
- Configure the frame transfer by writing to LINMR fields NACT, PARDIS, CHKDIS, CHKTYPE, DLM, and DLC
- Select LIN mode and slave node by writing 0xB to MR.MODE
- Write a one to CR.TXEN and CR.RXEN to enable both transmitter and receiver
- Wait until CSR.LINID is one
- Check for CSR.LINISFE and CSR.LINPE errors, clear errors and CSR.LINID by writing a one to CR.RSTSTA
- Read LINIR.IDCHR

IMPORTANT: If LINMR.NACT is 0x0 (PUBLISH), and this field is already correct, the LINMR register must still be written with this value in order to set CSR.TXRDY, and to request the corresponding Peripheral DMA Controller write transfer.

The different LINMR.NACT settings result in the same procedure as for the master node, see Section 24.6.11.1 “Master Node Configuration” on page 604.

Figure 24-37. Slave Node Configuration, LINMR.NACT is 0x0 (PUBLISH)

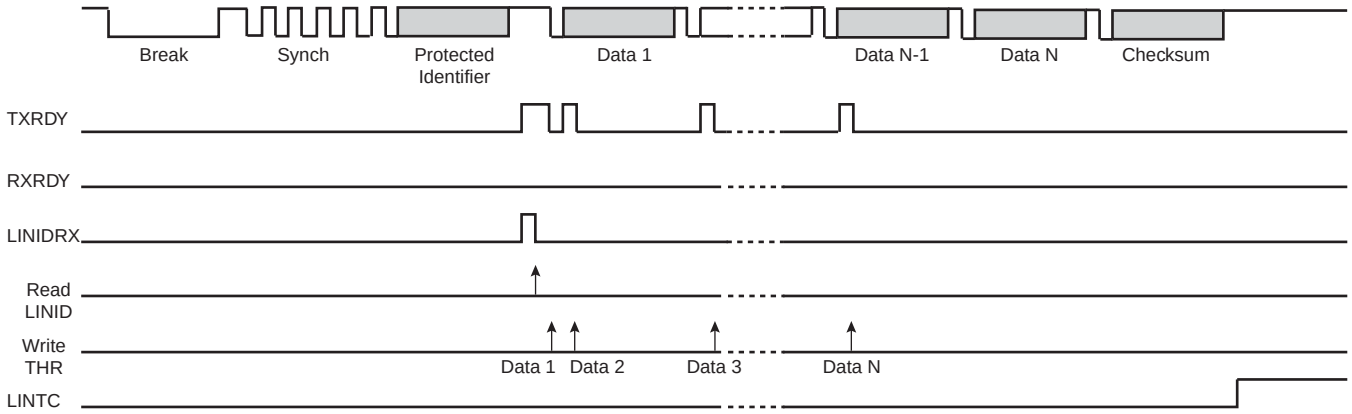


Figure 24-38. Slave Node Configuration, LINMR.NACT is 0x1 (SUBSCRIBE)

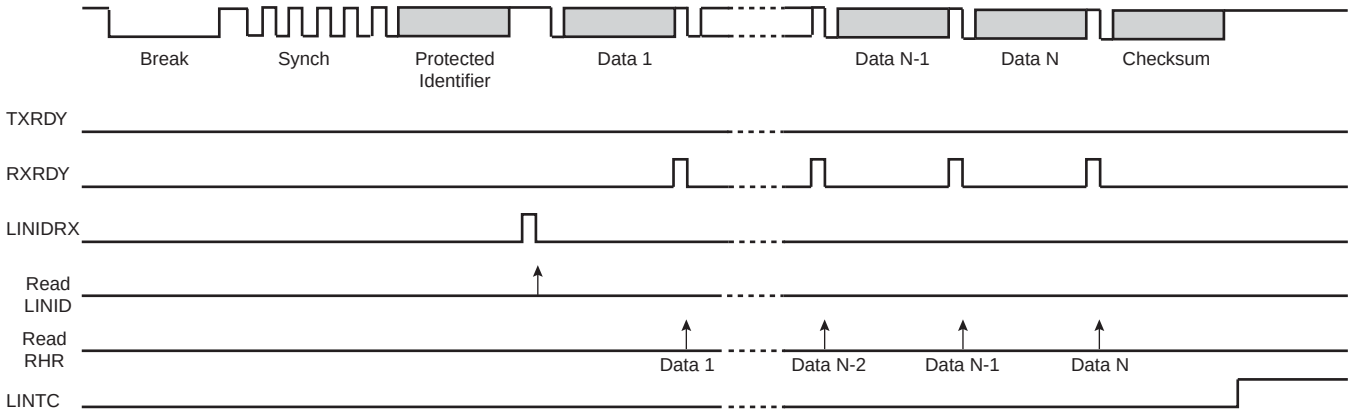
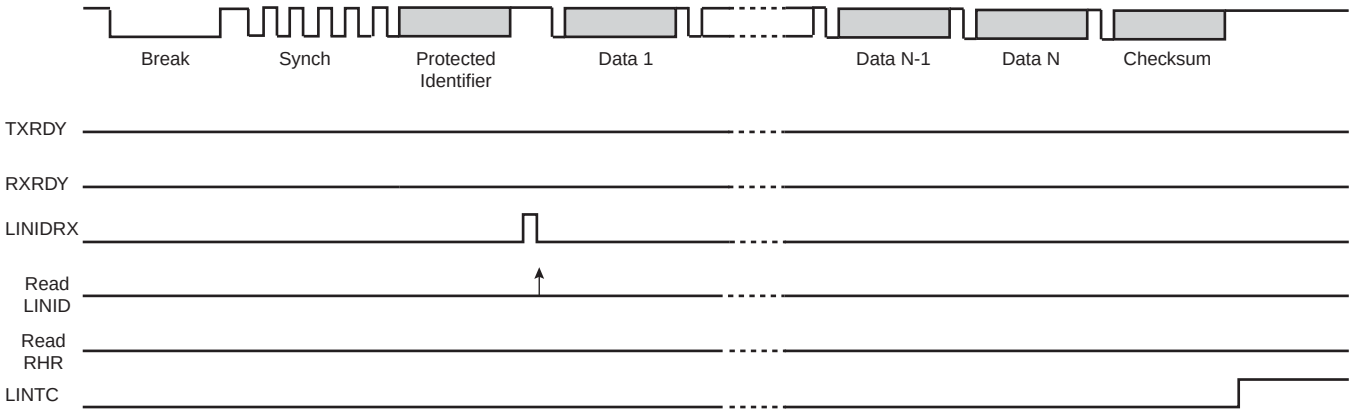


Figure 24-39. Slave Node Configuration, LINMR.NACT is 0x2 (IGNORE)



24.6.12 LIN Frame Handling With The Peripheral DMA Controller

The USART can be used together with the Peripheral DMA Controller in order to transfer data without processor intervention. The Peripheral DMA Controller uses the CSR.TXRDY and CSR.RXRDY bits to trigger one byte writes or reads. It always writes to THR, and it always reads RHR.

24.6.12.1 Master Node Configuration

The Peripheral DMA Controller Mode bit (LINMR.PDCM) allows the user to select configuration:

- LINMR.PDCM=0: LIN configuration must be written to LINMR, it is not stored in the write buffer.
- LINMR.PDCM=1: LIN configuration is written by the Peripheral DMA Controller to THR, and is stored in the write buffer. Since data transfer size is a byte, the transfer is split into two accesses. The first writes the NACT, PARDIS, CHKDIS, CHKTYP, DLM and FSDIS bits in the LINMR register, while the second writes the LINMR.DLC field. If LINMR.NACT=PUBLISH, the write buffer will also contain the Identifier.

When LINMR.NACT=SUBSCRIBE, the read buffer contains the data.

Figure 24-40. Master Node with Peripheral DMA Controller (LINMR.PDCM=0)

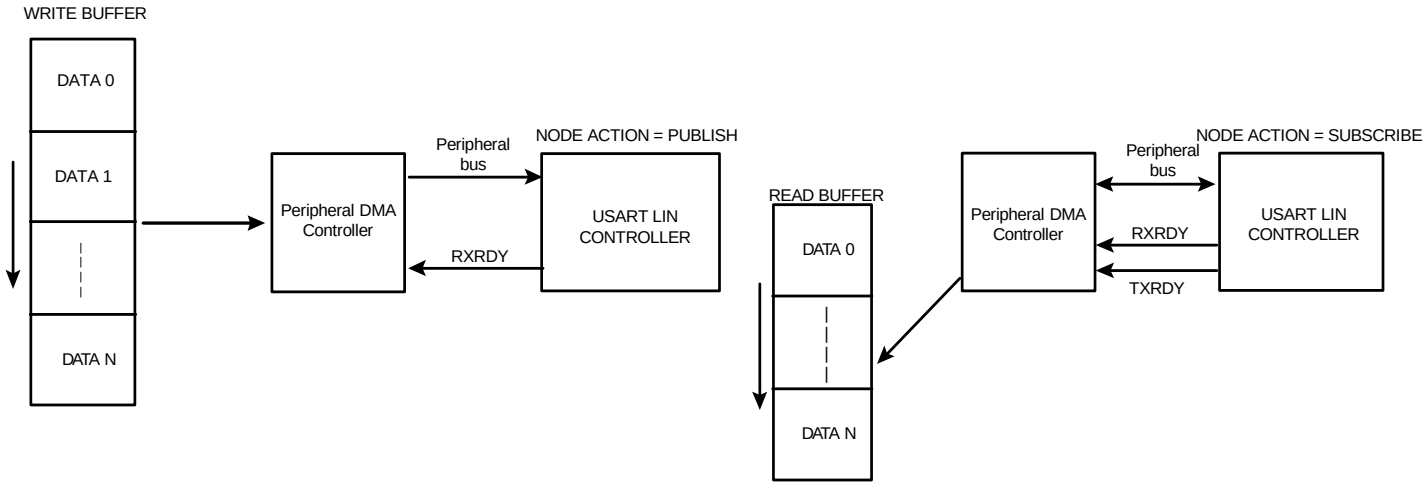
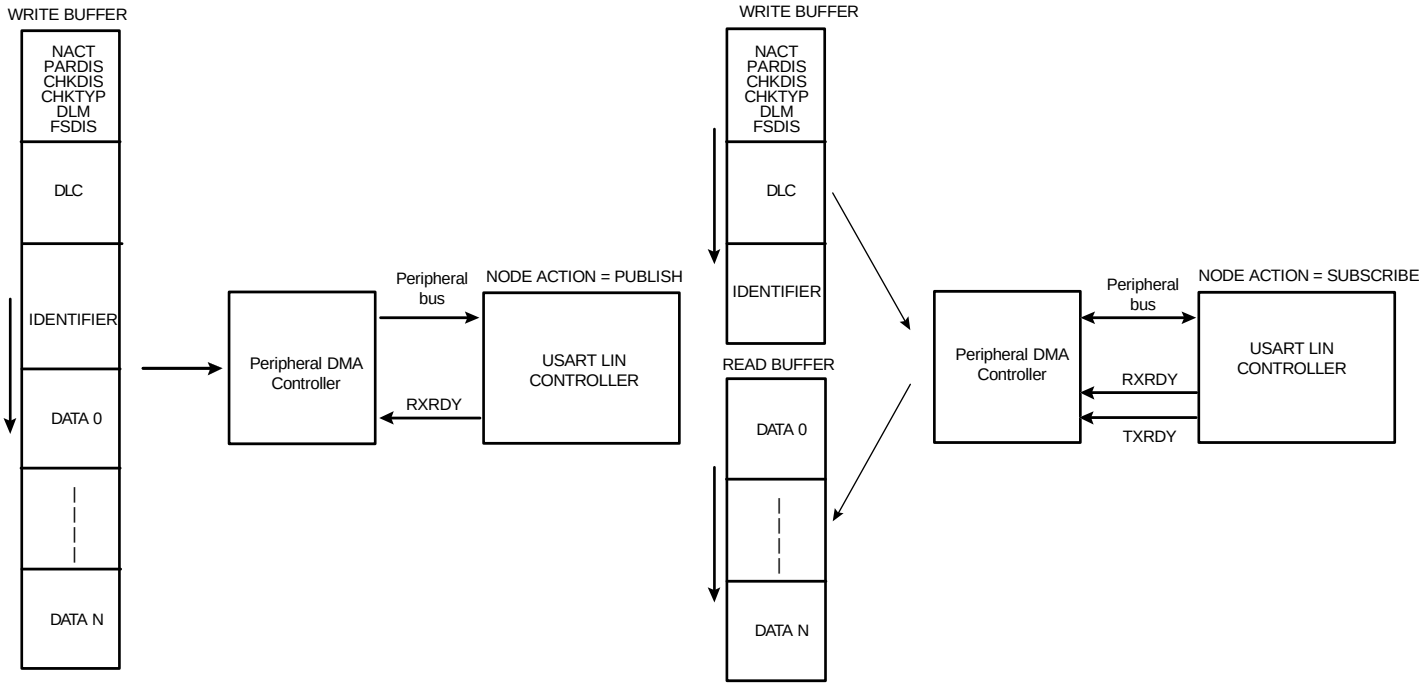


Figure 24-41. Master Node with Peripheral DMA Controller (LINMR.PDCM=1)

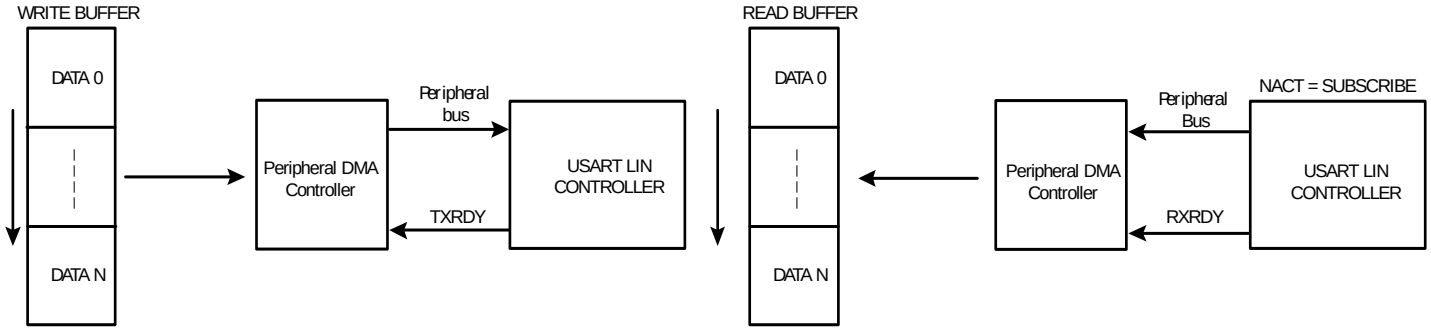


24.6.12.2 Slave Node Configuration

In this mode, the Peripheral DMA Controller transfers only data. The user reads the Identifier from LINIR, and selects LIN mode by writing to LINMR. When NACT=PUBLISH the data is in the write buffer, while the read buffer contains the data when NACT=SUBSCRIBE.

IMPORTANT: If in slave mode, LINMR.NACT is already configured correctly as PUBLISH, the LINMR register must still be written with this value in order to set CSR.TXRDY, and to request the corresponding Peripheral DMA Controller write transfer.

Figure 24-42. Slave Node with Peripheral DMA Controller



24.6.13 Wake-up Request

Any node in a sleeping LIN cluster may request a wake-up. By writing to the Wakeup Signal Type bit (LINMR.WKUPTYP), the user can choose to send either a LIN 1.3 (WKUPTYP is one) or a LIN 2.0 (WKUPTYP is zero) compliant wakeup request. Writing a one to the Send LIN Wakeup Signal bit (CR.LINWKUP), transmits a wakeup, and when completed, sets CSR.LINTC.

According to LIN 1.3, the wakeup request should be generated with the character 0x80 in order to impose eight successive dominant bits.

According to LIN 2.0, the wakeup request is issued by forcing the bus into the dominant state for 250µs to 5ms. Sending the character 0xF0 does this, regardless of baud rate.

- Baud rate max = 20 kbit/s -> one bit period = 50µs -> five bit periods = 250µs
- Baud rate min = 1 kbit/s -> one bit period = 1ms -> five bit periods = 5ms

24.6.14 Bus Idle Time-out

LIN bus inactivity should eventually cause slaves to time out and enter sleep mode. LIN 1.3 specifies this to 25000 bit periods, whilst LIN 2.0 specifies 4 seconds. For the time-out counter operation see Section 24.6.3.4 “Receiver Time-out” on page 585.

Table 24-15. Receiver Time-out Values (RTOR.TO)

LIN Specification	Baud Rate	Time-out period	TO
2.0	1 000 bit/s	4s	4 000
	2 400 bit/s		9 600
	9 600 bit/s		38 400
	19 200 bit/s		76 800
	20 000 bit/s		80 000
1.3	-	25 000 bit periods	25 000

24.6.15 SPI Mode

The USART features a Serial Peripheral Interface (SPI) link compliant mode, supporting synchronous, full-duplex communication in both master and slave mode. Writing 0xE (master) or 0xF (slave) to MR.MODE will enable this mode. An SPI in master mode controls the data flow to and from the other SPI devices, which are in slave mode. It is possible to let devices take turns being masters (aka multi-master protocol), and one master may shift data simultaneously into several slaves, but only one slave may respond at a time. A slave is selected when its slave

select (NSS) signal has been raised by the master. The USART can only generate one NSS signal, and it is possible to use standard I/O lines to address more than one slave.

24.6.15.1 Modes of Operation

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): This line supplies the data shifted from master to slave. In master mode this is connected to TXD, and in slave mode to RXD.
- Master In Slave Out (MISO): This line supplies the data shifted from slave to master. In master mode this is connected to RXD, and in slave mode to TXD.
- Serial Clock (CLK): This is controlled by the master. One period per bit transmission. In both modes this is connected to CLK.
- Slave Select (NSS): This control line allows the master to select or deselect a slave. In master mode this is connected to RTS, and in slave mode to CTS.

Changing SPI mode after initial configuration must be followed by a transceiver software reset in order to avoid unpredictable behavior.

24.6.15.2 Baud Rate

The baud rate generator operates as described in “Baud Rate in Synchronous and SPI Mode” on page 589, with the following requirements:

In SPI Master Mode:

- External clock CLK must not be selected as clock (the Clock Selection field (MR.USCLKS) must not equal 0x3).
- The USART must drive the CLK pin (MR.CLKO must be one).
- The BRGR.CD field must be at least 0x4.
- If the internal divided clock, CLK_USART/DIV, is selected (MR.USCLKS is one), the value in BRGR.CD must be even, ensuring a 50:50 duty cycle.

In SPI Slave Mode:

- The frequency of the external clock CLK must be at least four times lower than the system clock.

24.6.15.3 Data Transfer

Up to nine data bits are successively shifted out on the TXD pin at each edge. There are no start, parity, or stop bits, and MSB is always sent first. The SPI Clock Polarity (MR.CPOL), and SPI Clock Phase (MR.CPHA) bits configure CLK by selecting the edges upon which bits are shifted and sampled, resulting in four non-interoperable protocol modes, see Table 24-16. If MR.CPOL is zero, the inactive state value of CLK is logic level zero, and if MR.CPOL is one, the inactive state value of CLK is logic level one. If MR.CPHA is zero, data is changed on the leading edge of CLK, and captured on the following edge of CLK. If MR.CPHA is one, data is captured on the leading edge of CLK, and changed on the following edge of CLK. A mas-

ter/slave pair must use the same configuration, and the master must be reconfigured if it is to communicate with slaves using different configurations. See Figures 24-43 and 24-44.

Table 24-16. SPI Bus Protocol Modes

MR.CPOL	MR.CPHA	SPI Bus Protocol Mode
0	1	0
0	0	1
1	1	2
1	0	3

Figure 24-43. SPI Transfer Format (CPHA=1, 8 bits per transfer)

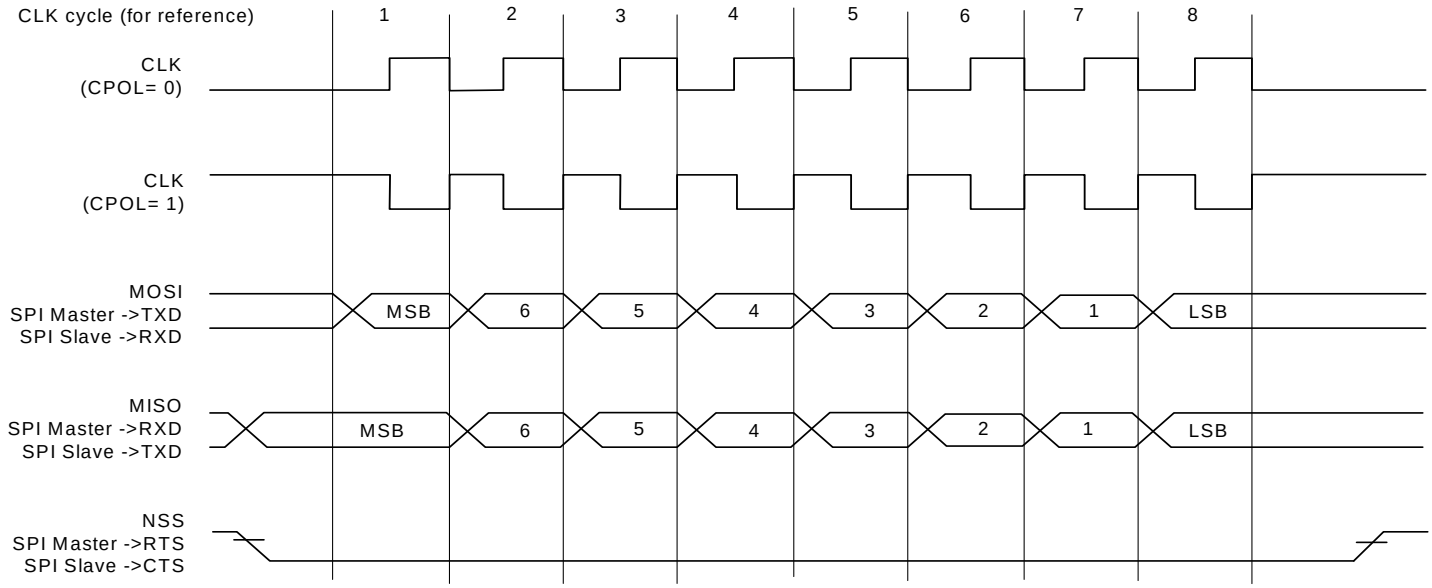
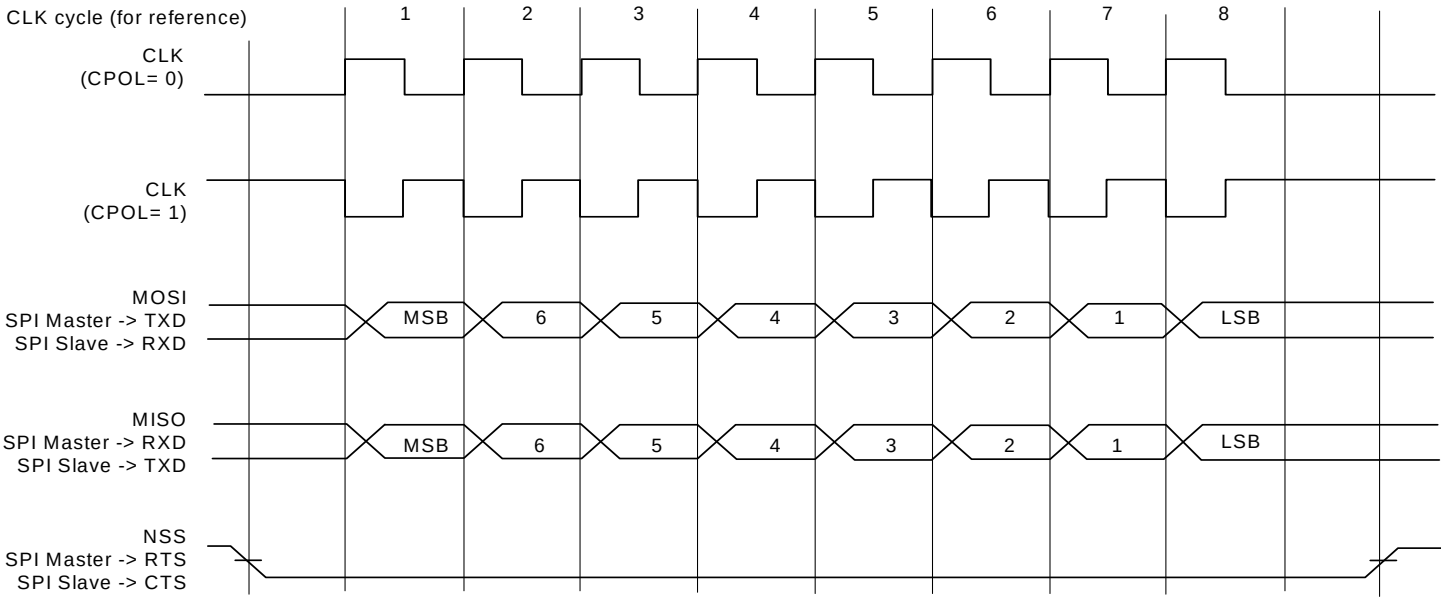


Figure 24-44. SPI Transfer Format (CPHA=0, 8 bits per transfer)



24.6.15.4 Receiver and Transmitter Control

See “Manchester Encoder” on page 612, and “Receiver Status” on page 582.

24.6.15.5 Character Transmission and Reception

When the Inhibit Non Acknowledge bit in MR (MR.INACK) is one, the SPI master will not send pending THR values until CSR.RXRDY is zero. In SPI master mode, the slave select line (NSS) is asserted low one bit period before the start of transmission, and released high one bit period after every character transmission. A delay for at least three bit periods is always inserted in between characters. In order to address slave devices supporting the Chip Select Active After Transfer (CSAAT) mode, NSS can be forced low by writing a one to the Force SPI Chip Select bit (CR.RTSEN/FCS). Releasing NSS when FCS is one is only possible by writing a one to the Release SPI Chip Select bit (CR.RTSDIS/RCS).

In SPI slave mode, a low level on NSS for at least one bit period will allow the slave to initiate a transmission or reception. The Underrun Error bit (CSR.UNRE) is set if a character must be sent while THR is empty, and TXD will be high during character transmission, as if 0xFF was being sent. An interrupt request is generated if the Underrun Error bit in the Interrupt Mask Register (IMR.UNRE) is set. If a new character is written to THR it will be sent correctly during the next transmission slot. Writing a one to CR.RSTSTA will clear CSR.UNRE. To ensure correct behavior of the receiver in SPI slave mode, the master device sending the frame must ensure a minimum delay of one bit period in between each character transmission.

24.6.15.6 Receiver Time-out

Receiver Time-outs are not possible in SPI mode as the Baud Rate Clock is only active during data transfers.

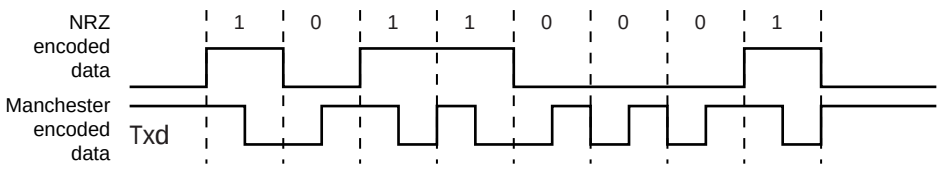
24.6.16 Manchester Encoder/Decoder

Writing a one to the Manchester Encoder/Decoder bit in the Mode Register (MR.MAN) enables the Manchester Encoder/Decoder. When the Manchester Encoder/Decoder is used, characters transmitted through the USART are encoded in Manchester II Biphase format. Depending on polarity configuration, selected by the Transmission Manchester Polarity bit in the Manchester Configuration Register (MAN.TX_MOPL), a logic level (zero or one) is transmitted as the transition from high-to-low or low-to-high during the middle of each bit period. This consumes twice the bandwidth of the simpler NRZ coding schemes, but the receiver has more error control since the expected input has a transition at every mid-bit period.

24.6.16.1 Manchester Encoder

An example of a Manchester encoded sequence is the byte 0xB1 (10110001) being encoded to 10 01 10 10 01 01 01 10, assuming default encoder polarity. Figure 24-45 illustrates this coding scheme.

Figure 24-45. NRZ to Manchester Encoding



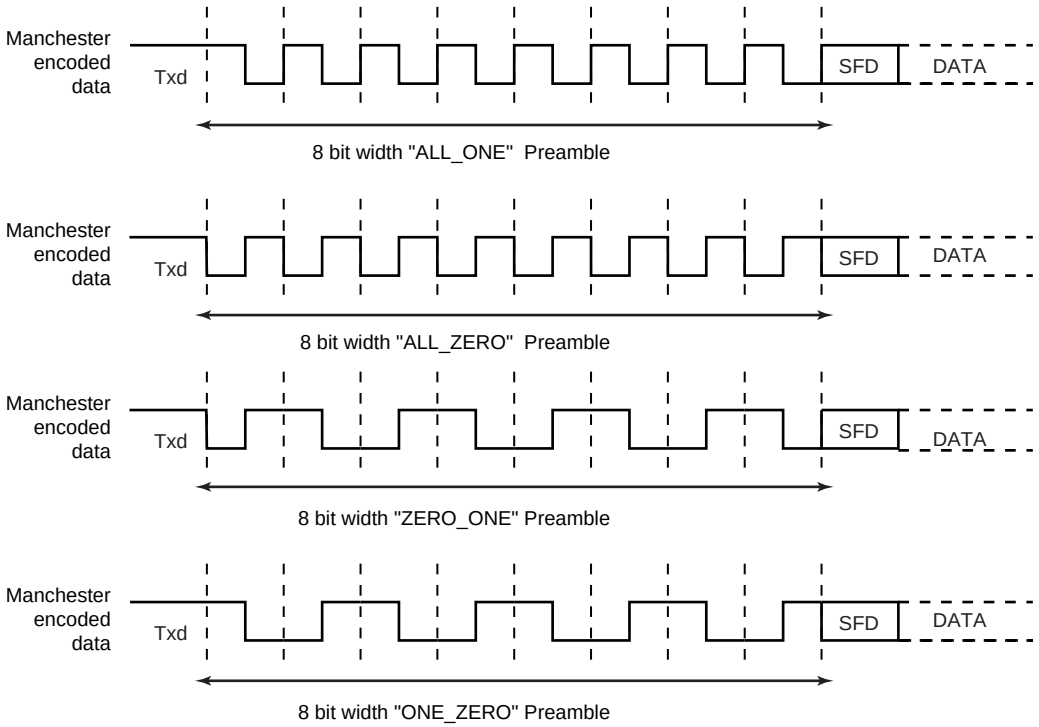
A Manchester encoded character can be preceded by both a preamble sequence and a start frame delimiter. The preamble sequence is a pre-defined pattern with a configurable length from

1 to 15 bit periods. If the preamble length is zero, the preamble waveform is not generated. The preamble length is selected by writing to the Transmitter Preamble Length field (MAN.TX_PL). The available preamble sequence patterns are:

- ALL_ONE
- ALL_ZERO
- ONE_ZERO
- ZERO_ONE

and are selected by writing to the Transmitter Preamble Pattern field (MAN.TX_PP). Figure 24-46 illustrates the supported patterns.

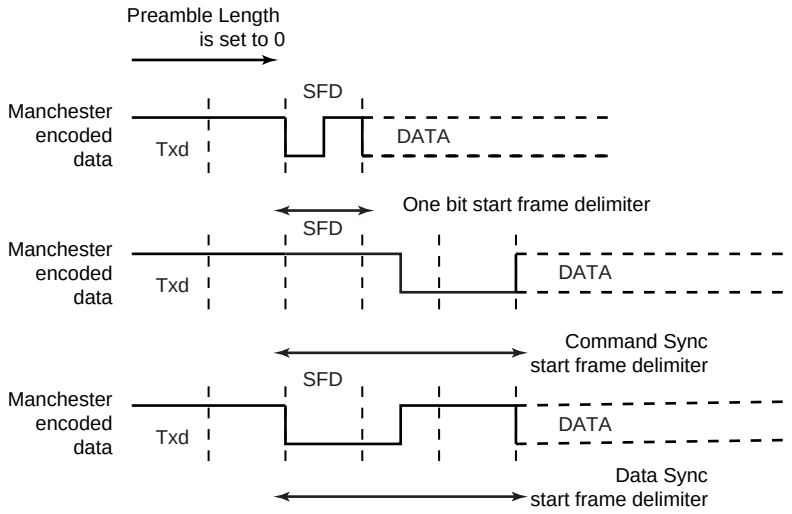
Figure 24-46. Preamble Patterns, Default Polarity Assumed



The Start Frame Delimiter Selector bit (MR.ONEBIT) configures the Manchester start bit pattern following the preamble. If MR.ONEBIT is one, a Manchester encoded zero is transmitted to indicate that a new character is about to be sent. If MR.ONEBIT is zero, a synchronization pattern is sent for the duration of three bit periods to inaugurate the new character. The sync pattern waveform by itself is an invalid Manchester encoding, since the transition only occurs at the middle of the second bit period.

The Manchester Synchronization Mode bit (MR.MODSYNC) selects sync pattern, and this also defines if the character is data (MODSYNC=0) with a zero to one transition, or a command (MODSYNC=1) with a one to zero transition. When direct memory access is used, the sync pattern can be updated on-the-fly with a modified character located in memory. To enable this mode the Variable Synchronization of Command/Data Sync Start Frame Delimiter bit (MR.VAR_SYNC) must be written to one. In this case, MODSYNC is bypassed and THR.TXSYNH selects the sync type to be included. Figure 24-47 illustrates supported patterns.

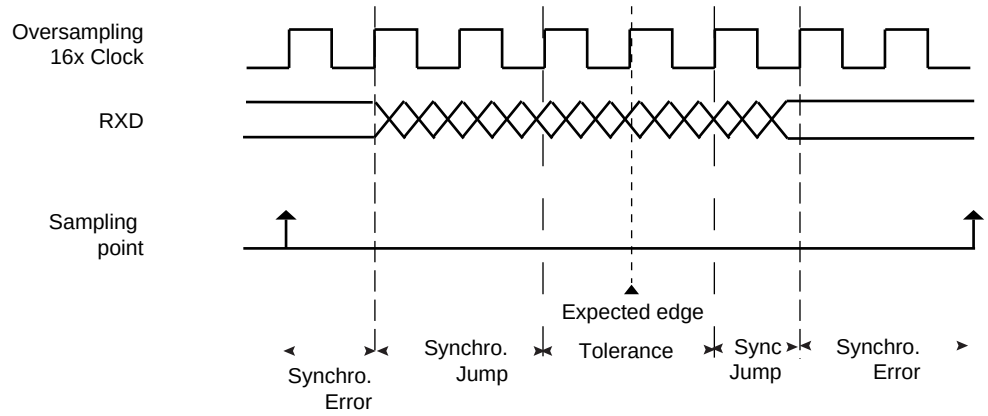
Figure 24-47. Start Frame Delimiter



Manchester Drift Compensation

The Drift Compensation bit (MAN.DRIFT) enables a hardware drift compensation and recovery system that allows for sub-optimal clock drifts without further user intervention. Drift compensation is only available in 16x oversampling mode (MR.OVER is zero). If the RXD event is one 16th clock cycle from the expected edge, it is considered as normal jitter and no corrective action will be taken. If the event is two to four 16th's early, the current period will be shortened by a 16th. If the event is two to three 16th's after the expected edge, the current period will be prolonged by a 16th.

Figure 24-48. Bit Resynchronization

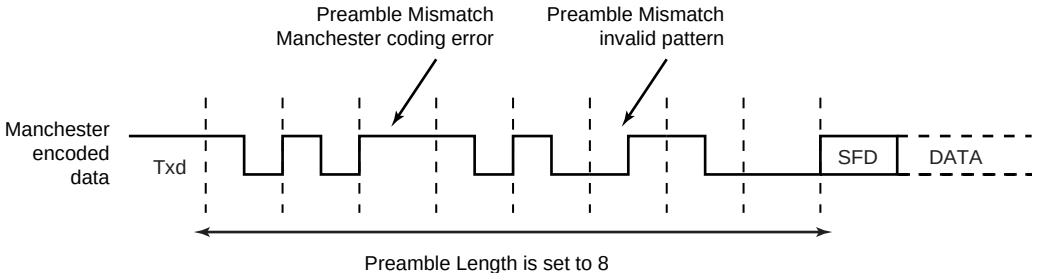


24.6.16.2 Manchester Decoder

The Manchester decoder can detect selectable preamble sequences and start frame delimiters. The Receiver Manchester Polarity bit in the "Manchester Configuration Register" (MAN.RX_MPOL) selects input stream polarity. The Receiver Preamble Length field (MAN.RX_PL) specifies the length characteristics of detectable preambles. If MAN.RX_PL is zero, the preamble pattern detection will be disabled. The Receiver Preamble Pattern field (MAN.RX_PP) selects the pattern to be detected. See Figure 24-46 for available preamble patterns. Figure 24-49 illustrates two types of Manchester preamble pattern mismatches.

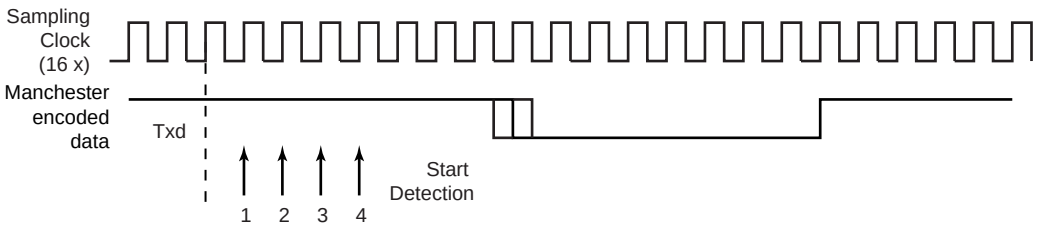
The Manchester endec uses the same Start Frame Delimiter Selector (MR.ONEBIT) for both encoder and decoder. If ONEBIT is one, only a Manchester encoded zero will be accepted as a valid start frame delimiter. If ONEBIT is zero, a data or command sync pattern will be expected. The Received Sync bit in the Receive Holding Register (RHR.RXSYNH) will be zero if the last character received is a data sync, and a one if it is a command sync.

Figure 24-49. Preamble Pattern Mismatch



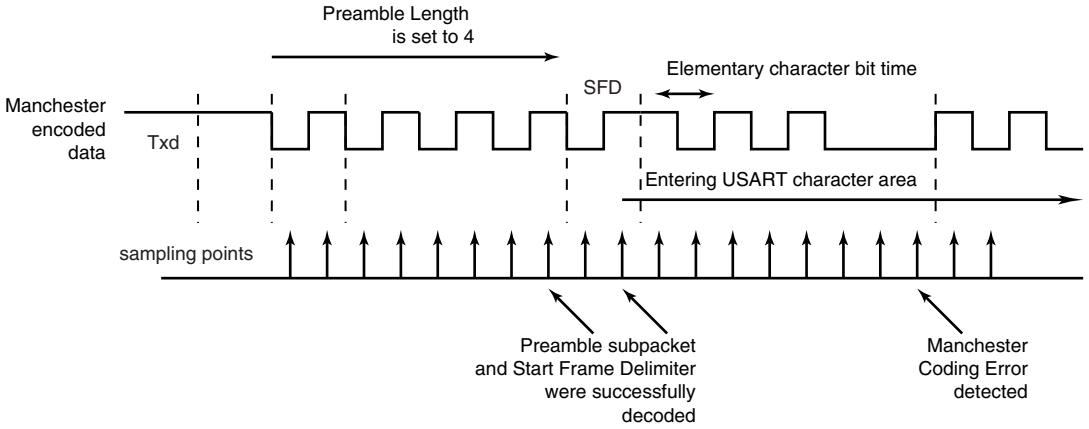
The receiver samples the RXD line in continuos bit period quarters, making the smallest time frame in which to assume a bit value three quarters. A start bit is assumed if RXD is zero during one of these quarters, see Figure 24-50.

Figure 24-50. Asynchronous Start Bit Detection



If a valid preamble pattern or start frame delimiter is detected, the receiver continues decoding with the same synchronization. If a non-valid preamble pattern or a start frame delimiter is detected, the receiver re-synchronizes at the next valid edge. When a valid start sequence has been detected, the decoded data is passed to the USART and the user will be notified of any incoming Manchester encoding violations by the Manchester Error bit (CSR.MANERR). An interrupt request is generated if one of the Manchester Error bits in the Interrupt Mask Register (IMR.MANE or IMR.MANEA) is set. CSR.MANERR is cleared by writing a one to the Reset Status bits in the Control Register (CR.RSTSTA). A violation occurs when there is no transition in the middle of a bit period. See Figure 24-51 for an illustration of a violation causing the Manchester Error bit to be set.

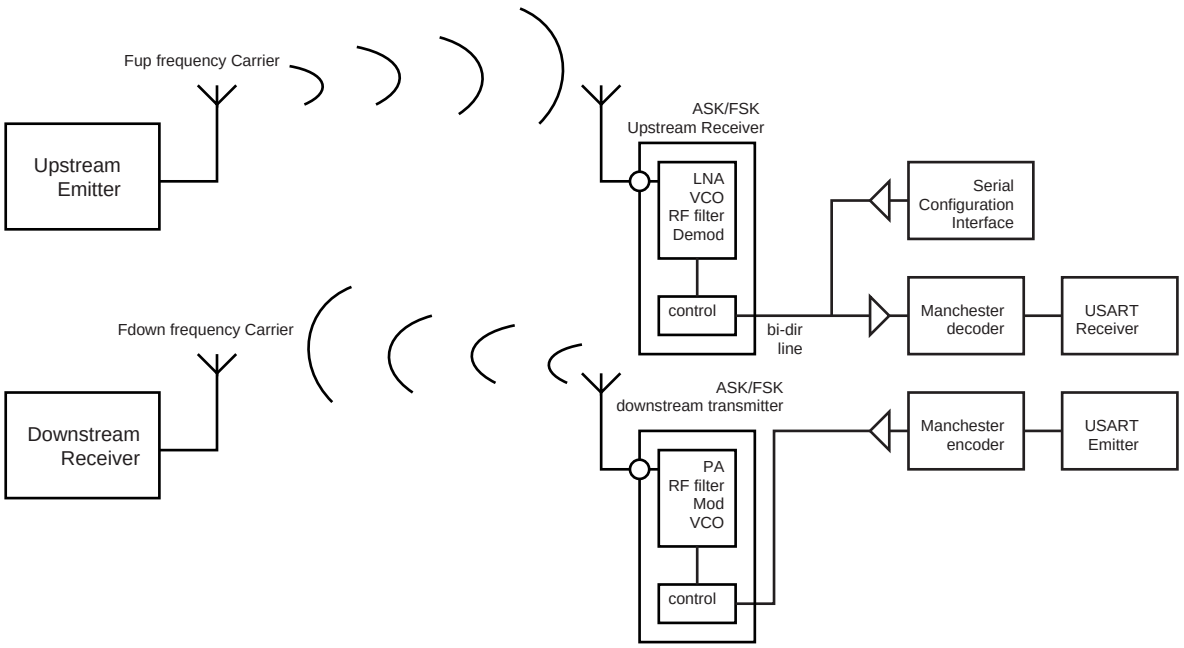
Figure 24-51. Manchester Error



24.6.16.3 Radio Interface: Manchester Endec Application

This section describes low data rate, full duplex, dual frequency, RF systems integrated with a Manchester endec, that support ASK and/or FSK modulation schemes. See Figure 24-52.

Figure 24-52. Manchester Encoded Characters RF Transmission



To transmit downstream, encoded data is sent serially to the RF modulator and then through space to the RF receiver. To receive, another frequency carrier is used and the RF demodulator does a bit-checking search for valid patterns before it switches to a receiving mode and forwards data to the decoder. Defining preambles to help distinguish between noise and valid data has to be done in conjunction with the RF module, and may sometimes be filtered away from the endec stream. Using the ASK modulation scheme, a one is transmitted as an RF signal at the downstream frequency, while a zero is transmitted as no signal. See Figure 24-53. The FSK modulation scheme uses two different frequencies to transmit data. A one is sent as a signal on one frequency, and a zero on the other. See Figure 24-54.

Figure 24-53. ASK Modulator Output

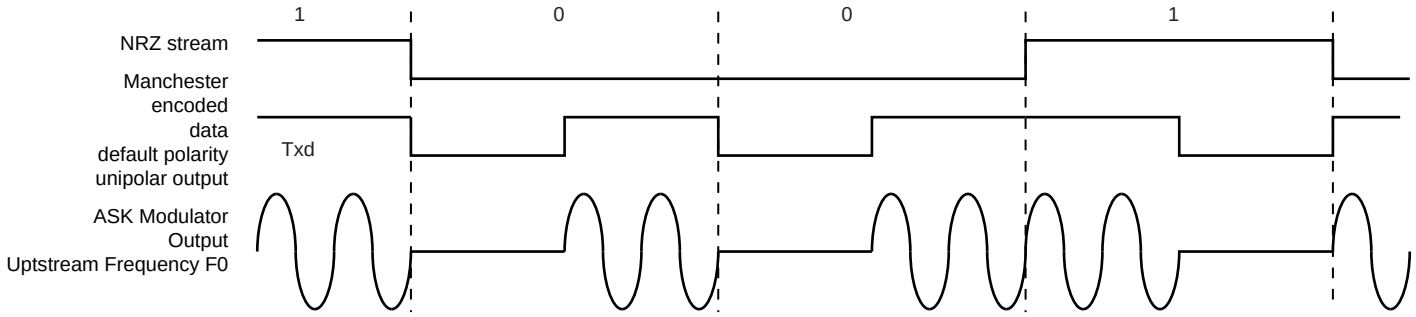
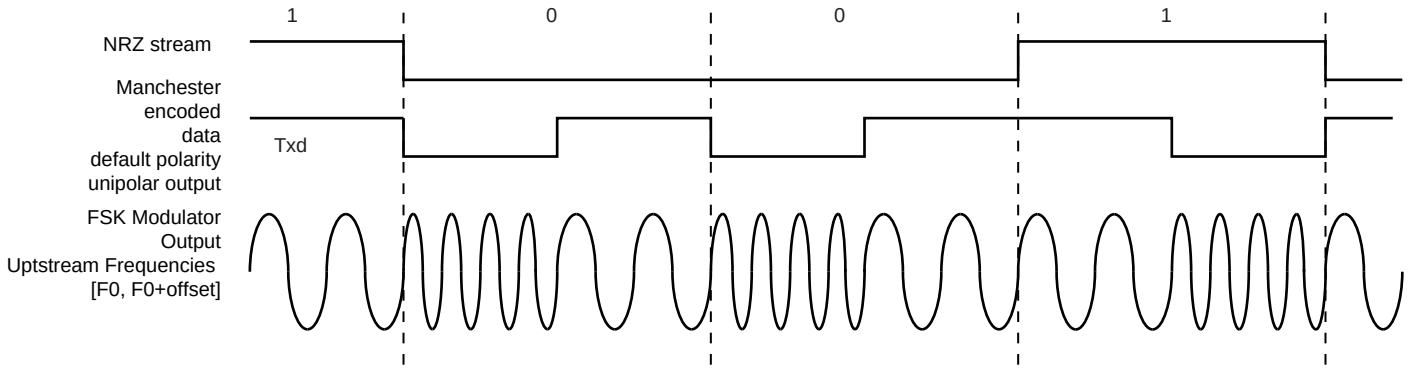


Figure 24-54. FSK Modulator Output



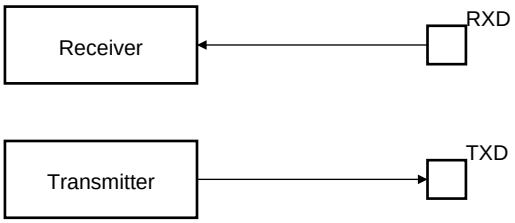
24.6.17 Test Modes

The internal loopback feature enables on-board diagnostics, and allows the USART to operate in three different test modes, with reconfigured pin functionality, as shown below.

24.6.17.1 Normal Mode

During normal operation, a receiver RXD pin is connected to a transmitter TXD pin.

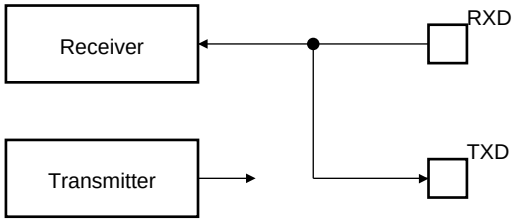
Figure 24-55. Normal Mode Configuration



24.6.17.2 Automatic Echo Mode

Automatic echo mode allows bit-by-bit retransmission. When a bit is received on the RXD pin, it is also sent to the TXD pin, as shown in Figure 24-56. Transmitter configuration has no effect.

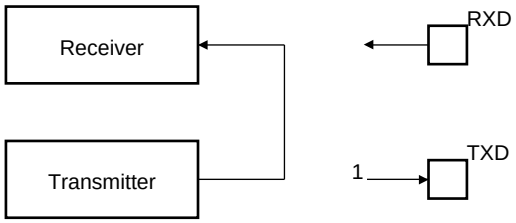
Figure 24-56. Automatic Echo Mode Configuration



24.6.17.3 Local Loopback Mode

Local loopback mode connects the output of the transmitter directly to the input of the receiver, as shown in Figure 24-57. The TXD and RXD pins are not used. The RXD pin has no effect on the receiver and the TXD pin is continuously driven high, as in idle state.

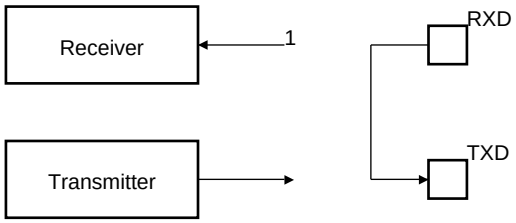
Figure 24-57. Local Loopback Mode Configuration



24.6.17.4 Remote Loopback Mode

Remote loopback mode connects the RXD pin to the TXD pin, as shown in Figure 24-58. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit retransmission.

Figure 24-58. Remote Loopback Mode Configuration



24.6.18 Interrupts

							MANEA
23	22	21	20	19	18	17	16
–	–	–	MANE	CTSIC	DCDIC–	DSRIC–	RIIC–
15	14	13	12	11	10	9	8
LINTC	LINID	NACK/LINBK	RXBUFF	–	ITER/UNRE	TXEMPTY	TIMEOUT

7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

The USART has the following interrupt sources:

- LINHTE: LIN Header Time-out Error
  - A LIN Header Time-out Error has been detected
- LINSTE: LIN Sync Tolerance Error
  - A LIN Sync Tolerance Error has been detected
- LINSNRE: LIN Slave Not Responding Error
  - A LIN Slave Not Responding Error has been detected
- LINCE: LIN Checksum Error
  - A LIN Checksum Error has been detected
- LINIPE: LIN Identifier Parity Error
  - A LIN Identifier Parity Error has been detected
- LINISFE: LIN Inconsistent Sync Field Error
  - The USART is configured as a Slave node and a LIN Inconsistent Sync Field Error has been detected since the last RSTSTA.
- LINBE: LIN Bit Error
  - A Bit Error has been detected since the last RSTSTA.
- MANERR: Manchester Error
  - At least one Manchester error has been detected since the last RSTSTA.
- CTSIC: Clear to Send Input Change Flag
  - At least one change has been detected on the CTS pin since the last CSR read.
- DCDIC: Data Carrier Detect Input Change Flag
  - A change has been detected on the DCD pin
- DSRIC: Data Set Ready Input Change Flag
  - A change has been detected on the DSR pin
- RIIC: Ring Indicator Input Change Flag
  - A change has been detected on the RI pin
- LINTC: LIN Transfer Completed
  - A LIN transfer has been completed
- LINIDR: LIN Identifier
  - A LIN Identifier has been sent (master) or received (slave)
- NACK: Non Acknowledge
  - At least one Non Acknowledge has been detected
- RXBUFF: Reception Buffer Full
  - The Buffer Full signal from the Peripheral DMA Controller channel is active.
- ITER/UNRE: Max number of Repetitions Reached or SPI Underrun Error
  - IF USART does not operate in SPI slave mode: Maximum number of repetitions has been reached since the last RSTSTA.

  - If USART operates in SPI slave mode: At least one SPI underrun error has occurred since the last RSTSTA.
- TXEMPTY: Transmitter Empty
  - There are no characters in neither THR, nor in the transmit shift register.
- TIMEOUT: Receiver Time-out
  - There has been a time-out since the last Start Time-out command.
- PARE: Parity Error
  - Either at least one parity error has been detected, or the parity bit is a one in multidrop mode, since the last RSTSTA.
- FRAME: Framing Error
  - At least one stop bit has been found as low since the last RSTSTA.
- OVRE: Overrun Error
  - At least one overrun error has occurred since the last RSTSTA.
- RXBRK: Break Received/End of Break
  - Break received or End of Break detected since the last RSTSTA.
- TXRDY: Transmitter Ready
  - There is no character in the THR.
- RXRDY: Receiver Ready
  - At least one complete character has been received and RHR has not yet been read.

An interrupt source will set a corresponding bit in the Channel Status Register (CSR). The interrupt sources will generate an interrupt request if the corresponding bit in the Interrupt Mask Register (IMR) is set. The interrupt sources are ORed together to form one interrupt request. The USART will generate an interrupt request if at least one of the bits in IMR is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in CSR is cleared. The clearing of the bits in CSR is described in “Channel Status Register” on page 631. Because all the interrupt sources are ORed together, the interrupt request from the USART will remain active until all the bits in CSR are cleared.

24.6.19 Using the Peripheral DMA Controller

24.6.20 Write Protection Registers

To prevent single software errors from corrupting USART behavior, certain address spaces can be write-protected by writing the correct Write Protect KEY and writing a one to the Write Protect Enable bit in the Write Protect Mode Register (WPMR.WPKEY and WPMR.WPEN). Disabling the write protection is done by writing the correct key to WPMR.WPKEY and a zero to WPMR.WPEN.

Write attempts to a write-protected register are detected and the Write Protect Violation Status bit in the Write Protect Status Register (WPSR.WPVS) is set. The Write Protect Violation Source field (WPSR.WPVSRC) indicates the target register. Writing the correct key to the Write Protect KEY bit (WPMR.WPKEY) clears WPSR. WPVSRC and WPSR.WPVS.

The protected registers are:

- “Mode Register” on page 625

24.7 User Interface

Table 24-17. USART Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-only	0x00000000
0x04	Mode Register	MR	Read-write	0x00000000
0x08	Interrupt Enable Register	IER	Write-only	0x00000000
0x0C	Interrupt Disable Register	IDR	Write-only	0x00000000
0x010	Interrupt Mask Register	IMR	Read-only	0x00000000
0x14	Channel Status Register	CSR	Read-only	0x00000000
0x18	Receiver Holding Register	RHR	Read-only	0x00000000
0x1C	Transmitter Holding Register	THR	Write-only	0x00000000
0x20	Baud Rate Generator Register	BRGR	Read-write	0x00000000
0x24	Receiver Time-out Register	RTOR	Read-write	0x00000000
0x28	Transmitter Timeguard Register	TTGR	Read-write	0x00000000
0x40	FI DI Ratio Register	FIDI	Read-write	0x00000174
0x44	Number of Errors Register	NER	Read-only	0x00000000
0x4C	IrDA Filter Register	IFR	Read-write	0x00000000
0x50	Manchester Configuration Register	MAN	Read-write	0x30011004
0x54	LIN Mode Register	LINMR	Read-write	0x00000000
0x58	LIN Identifier Register	LINIR	Read-write	0x00000000
0x5C	LIN Baud Rate Register	LINBR	Read-only	0x00000000
0xE4	Write Protect Mode Register	WPMR	Read-write	0x00000000
0xE8	Write Protect Status Register	WPSR	Read-only	0x00000000
0xFC	Version Register	VERSION	Read-only	-(1)

Note: 1. Values in the Version Register vary with the version of the IP block implementation.

24.7.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	LINWKUP	LINABT	RTSDIS/RCS	RTSEN/FCS	DTRDIS–	DTREN–

15	14	13	12	11	10	9	8
RETTO	RSTNACK	RSTIT	SENDA	STTTO	STPBRK	STTBRK	RSTSTA

7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- **LINWKUP: Send LIN Wakeup Signal**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will send a wakeup signal on the LIN bus.
- **LINABT: Abort LIN Transmission**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will abort the current LIN transmission.
- **RTSDIS/RCS: Request to Send Disable/Release SPI Chip Select**
  Writing a zero to this bit has no effect.
  Writing a one to this bit when USART is not in SPI master mode drives RTS high.
  Writing a one to this bit when USART is in SPI master mode releases NSS (RTS pin).
- **RTSEN/FCS: Request to Send Enable/Force SPI Chip Select**
  Writing a zero to this bit has no effect.
  Writing a one to this bit when USART is not in SPI master mode drives RTS low.
  Writing a one to this bit when USART is in SPI master mode forces NSS (RTS pin) low, even if USART is not transmitting, in order to address SPI slave devices supporting the CSAAT Mode (Chip Select Active After Transfer).
- **DTRDIS: Data Terminal Ready Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit drives DTR high.
- **DTREN: Data Terminal Ready Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit drives DTR low.
- **RETTO: Rearm Time-out**
  Writing a zero to this bit has no effect.
  Writing a one to this bit reloads the time-out counter and clears CSR.TIMEOUT.
- **RSTNACK: Reset Non Acknowledge**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears CSR.NACK.
- **RSTIT: Reset Iterations**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears CSR.ITER if ISO7816 is enabled (MR.MODE is 0x4 or 0x6)

- **SENDA: Send Address**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will in multidrop mode send the next character written to THR as an address.
- **STTTO: Start Time-out**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will abort any current time-out count down, and trigger a new count down when the next character has been received. CSR.TIMEOUT is also cleared.
- **STPBRK: Stop Break**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will stop the generation of break signal characters, and then send ones for TTGR.TG duration, or at least 12 bit periods. No effect if no break is being transmitted.
- **STTBRK: Start Break**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will start transmission of break characters when current characters present in THR and the transmit shift register have been sent. No effect if a break signal is already being generated. CSR.TXRDY and CSR.TXEMPTY will be cleared.
- **RSTSTA: Reset Status Bits**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the following bits in CSR: PARE, FRAME, OVRE, MANERR, LINHTE, LINSTE, LINBE, LINISFE, LINIPE, LINCE, LINSNRE, LINTC, LINID, UNRE, and RXBRK.
- **TXDIS: Transmitter Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the transmitter.
- **TXEN: Transmitter Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the transmitter if TXDIS is zero.
- **RXDIS: Receiver Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the receiver.
- **RXEN: Receiver Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the receiver if RXDIS is zero.
- **RSTTX: Reset Transmitter**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will reset the transmitter.
- **RSTRX: Reset Receiver**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will reset the receiver.

24.7.2 Mode Register

Name: MR

Access Type: Read-write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
ONEBIT	MODSYNC	MAN	FILTER	–	MAX_ITERATION		
23	22	21	20	19	18	17	16
INVDATA	VAR_SYNC	DSNACK	INACK	OVER	CLKO	MODE9	MSBF/CPOL
15	14	13	12	11	10	9	8
CHMODE		NBSTOP		PAR			SYNC/CPHA
7	6	5	4	3	2	1	0
CHRL		USCLKS		MODE			

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **ONEBIT: Start Frame Delimiter Selector**
  0: The start frame delimiter is a command or data sync, as defined by MODSYNC.
  1: The start frame delimiter is a normal start bit, as defined by MODSYNC.
- **MODSYNC: Manchester Synchronization Mode**
  0: The manchester start bit is either a 0-to-1 transition, or a data sync.
  1: The manchester start bit is either a 1-to-0 transition, or a command sync.
- **MAN: Manchester Encoder/Decoder Enable**
  0: Manchester endec is disabled.
  1: Manchester endec is enabled.
- **FILTER: Infrared Receive Line Filter**
  0: The USART does not filter the receive line.
  1: The USART filters the receive line by doing three consecutive samples and uses the majority value.
- **MAX_ITERATION**
  This field determines the number of acceptable consecutive NACKs when in protocol T=0.
- **INVDATA: Inverted Data**
  0: The TXD and RXD transmissions equal the values written to THR and read from RHR. Normal mode of operation.
  1: The TXD and RXD transmissions equal the xor'ed values written to THR and read from RHR. Inverted mode of operation, parity is accounted for.
- **VAR_SYNC: Variable Synchronization of Command/Data Sync Start Frame Delimiter**
  0: Sync pattern according to MODSYNC.
  1: Sync pattern according to THR.TXSYNH.
- **DSNACK: Disable Successive NACK**
  0: NACKs are handled as normal, unless disabled by INACK.
  1: The receiver restricts the amount of consecutive NACKs by MAX_ITERATION value. If MAX_ITERATION=0 no NACK will be issued and the first erroneous message is accepted as a valid character, setting CSR.ITER.
- **INACK: Inhibit Non Acknowledge**
  0: The NACK is generated. In SPI master mode, transmission is as usual.
  1: The NACK is not generated. In SPI master mode, data in THR will not be sent until RXRDY is zero.
- **OVER: Oversampling Mode**
  0: Oversampling at 16 times the baud rate.

1: Oversampling at 8 times the baud rate.

- **CLKO: Clock Output Select**
  0: The USART does not drive the CLK pin.
  1: The USART drives the CLK pin unless USCLKS selects the external clock.
- **MODE9: 9-bit Character Length**
  0: CHRL defines character length.
  1: 9-bit character length.
- **MSBF/CPOL: Bit Order or SPI Clock Polarity**
  If USART does not operate in SPI Mode:
  MSBF=0: Least Significant Bit is sent/received first.
  MSBF=1: Most Significant Bit is sent/received first.
  If USART operates in SPI Mode, CPOL is used with CPHA to produce the required clock/data relationship between devices.
  CPOL=0: The inactive state value of CLK is logic level zero.
  CPOL=1: The inactive state value of CLK is logic level one.
- **CHMODE: Channel Mode**

Table 24-18.

CHMODE		Mode Description
0	0	Normal Mode
0	1	Automatic Echo. Receiver input is connected to the TXD pin.
1	0	Local Loopback. Transmitter output is connected to the Receiver input.
1	1	Remote Loopback. RXD pin is internally connected to the TXD pin.

- **NBSTOP: Number of Stop Bits**

Table 24-19.

NBSTOP		Asynchronous (SYNC=0)	Synchronous (SYNC=1)
0	0	1 stop bit	1 stop bit
0	1	1.5 stop bits	Reserved
1	0	2 stop bits	2 stop bits
1	1	Reserved	Reserved

- **PAR: Parity Type**

Table 24-20.

PAR			Parity Type
0	0	0	Even parity
0	0	1	Odd parity
0	1	0	Parity forced to 0 (Space)
0	1	1	Parity forced to 1 (Mark)
1	0	x	No parity
1	1	x	Multidrop mode

- **SYNC/CPHA: Synchronous Mode Select or SPI Clock Phase**
  If USART does not operate in SPI Mode (MR.MODE is not equal to 0xE or 0xF):
  SYNC = 0: USART operates in Asynchronous mode.

SYNC = 1: USART operates in Synchronous mode.

If USART operates in SPI Mode, CPHA determines which edge of CLK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

CPHA = 0: Data is changed on the leading edge of CLK and captured on the following edge of CLK.

CPHA = 1: Data is captured on the leading edge of CLK and changed on the following edge of CLK.

- **CHRL: Character Length.**

Table 24-21.

CHRL		Character Length
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

- **USCLKS: Clock Selection**

Table 24-22.

USCLKS		Selected Clock
0	0	CLK_USART
0	1	CLK_USART/DIV(1)
1	0	Reserved
1	1	CLK

Note: 1. The value of DIV is device dependent. Refer to the Module Configuration section at the end of this chapter.

- **MODE**

Table 24-23.

MODE				Mode of the USART
0	0	0	0	Normal
0	0	0	1	RS485
0	0	1	0	Hardware Handshaking
0	0	1	1	Modem
0	1	0	0	IS07816 Protocol: T = 0
0	1	1	0	IS07816 Protocol: T = 1
1	0	0	0	IrDA
1	0	1	0	LIN Master
1	0	1	1	LIN Slave
1	1	1	0	SPI Master
1	1	1	1	SPI Slave
Others				Reserved

24.7.3 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	MANEA
23	22	21	20	19	18	17	16
–	–	–	MANE	CTSIC	DCDIC–	DSRIC–	RIIC–
15	14	13	12	11	10	9	8
LINTC	LINID	NACK/LINBK	RXBUFF	–	ITER/UNRE	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will set the corresponding bit in IMR.

- **LINHTE: LIN Header Time-out Error**
- **LINSTE: LIN Sync Tolerance Error**
- **LINSNRE: LIN Slave Not Responding Error**
- **LINCE: LIN Checksum Error**
- **LINIPE: LIN Identifier Parity Error**
- **LINISFE: LIN Inconsistent Sync Field Error**
- **LINBE: LIN Bit Error**
- **MANEA/MANE: Manchester Error**
- **CTSIC: Clear to Send Input Change Flag**
- **DCDIC: Data Carrier Detect Input Change Flag**
- **DSRIC: Data Set Ready Input Change Flag**
- **RIIC: Ring Indicator Input Change Flag**
- **LINTC: LIN Transfer Completed**
- **LINIDR: LIN Identifier**
- **NACK: Non Acknowledge**
- **RXBUFF: Reception Buffer Full**
- **ITER/UNRE: Max number of Repetitions Reached or SPI Underrun Error**
- **TXEMPTY: Transmitter Empty**
- **TIMEOUT: Receiver Time-out**
- **PARE: Parity Error**
- **FRAME: Framing Error**
- **OVRE: Overrun Error**
- **RXBRK: Break Received/End of Break**
- **TXRDY: Transmitter Ready**
- **RXRDY: Receiver Ready**

For backward compatibility the MANE bit has been duplicated to the MANEA bit position. Writing either one or the other has the same effect. The corresponding bit in CSR and the corresponding interrupt request are named MANERR.

24.7.4 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	MANEA
23	22	21	20	19	18	17	16
–	–	–	MANE	CTSIC	DCDIC–	DSRIC–	RIIC–
15	14	13	12	11	10	9	8
LINTC	LINID	NACK/LINBK	RXBUFF	–	ITER/UNRE	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will clear the corresponding bit in IMR.

- **LINHTE: LIN Header Time-out Error**
- **LINSTE: LIN Sync Tolerance Error**
- **LINSNRE: LIN Slave Not Responding Error**
- **LINCE: LIN Checksum Error**
- **LINIPE: LIN Identifier Parity Error**
- **LINISFE: LIN Inconsistent Sync Field Error**
- **LINBE: LIN Bit Error**
- **MANEA/MANE: Manchester Error**
- **CTSIC: Clear to Send Input Change Flag**
- **DCDIC: Data Carrier Detect Input Change Flag**
- **DSRIC: Data Set Ready Input Change Flag**
- **RIIC: Ring Indicator Input Change Flag**
- **LINTC: LIN Transfer Completed**
- **LINIDR: LIN Identifier**
- **NACK: Non Acknowledge**
- **RXBUFF: Reception Buffer Full**
- **ITER/UNRE: Max number of Repetitions Reached or SPI Underrun Error**
- **TXEMPTY: Transmitter Empty**
- **TIMEOUT: Receiver Time-out**
- **PARE: Parity Error**
- **FRAME: Framing Error**
- **OVRE: Overrun Error**
- **RXBRK: Break Received/End of Break**
- **TXRDY: Transmitter Ready**
- **RXRDY: Receiver Ready**

For backward compatibility the MANE bit has been duplicated to the MANEA bit position. Writing either one or the other has the same effect. The corresponding bit in CSR and the corresponding interrupt request are named MANERR.

24.7.5 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	MANEA
23	22	21	20	19	18	17	16
–	–	–	MANE	CTSIC	DCDIC–	DSRIC–	RIIC–
15	14	13	12	11	10	9	8
LINTC	LINID	NACK/LINBK	RXBUFF	–	ITER/UNRE	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

A bit in this register is cleared when the corresponding bit in IDR is written to one.

A bit in this register is set when the corresponding bit in IER is written to one.

- **LINHTE: LIN Header Time-out Error**
- **LINSTE: LIN Sync Tolerance Error**
- **LINSNRE: LIN Slave Not Responding Error**
- **LINCE: LIN Checksum Error**
- **LINIPE: LIN Identifier Parity Error**
- **LINISFE: LIN Inconsistent Sync Field Error**
- **LINBE: LIN Bit Error**
- **MANEA/MANE: Manchester Error**
- **CTSIC: Clear to Send Input Change Flag**
- **DCDIC: Data Carrier Detect Input Change Flag**
- **DSRIC: Data Set Ready Input Change Flag**
- **RIIC: Ring Indicator Input Change Flag**
- **LINTC: LIN Transfer Completed**
- **LINIDR: LIN Identifier**
- **NACK: Non Acknowledge**
- **RXBUFF: Reception Buffer Full**
- **ITER/UNRE: Max number of Repetitions Reached or SPI Underrun Error**
- **TXEMPTY: Transmitter Empty**
- **TIMEOUT: Receiver Time-out**
- **PARE: Parity Error**
- **FRAME: Framing Error**
- **OVRE: Overrun Error**
- **RXBRK: Break Received/End of Break**
- **TXRDY: Transmitter Ready**
- **RXRDY: Receiver Ready**

For backward compatibility the MANE bit has been duplicated to the MANEA bit position. Reading either one or the other has the same effect. The corresponding bit in CSR and the corresponding interrupt request are named MANERR.

24.7.6 Channel Status Register

Name: CSR

Access Type: Read-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	MANERR
23	22	21	20	19	18	17	16
CTS/LINBLS	DCD–	DSR–	RI–	CTSIC	DCDIC–	DSRIC–	RIIC–
15	14	13	12	11	10	9	8
LINTC	LINID	NACK/LINBK	RXBUFF	–	ITER/UNRE	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

- **LINHTE: LIN Header Time-out Error**
  0: No LIN Header Time-out Error has been detected since the last RSTSTA.
  1: A LIN Header Time-out Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **LINSTE: LIN Sync Tolerance Error**
  0: No LIN Sync Tolerance Error has been detected since the last RSTSTA.
  1: A LIN Sync Tolerance Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **LINSNRE: LIN Slave Not Responding Error**
  0: No LIN Slave Not Responding Error has been detected since the last RSTSTA.
  1: A LIN Slave Not Responding Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **LINCE: LIN Checksum Error**
  0: No LIN Checksum Error has been detected since the last RSTSTA.
  1: A LIN Checksum Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **LINIPE: LIN Identifier Parity Error**
  0: No LIN Identifier Parity Error has been detected since the last RSTSTA.
  1: A LIN Identifier Parity Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **LINISFE: LIN Inconsistent Sync Field Error**
  0: No LIN Inconsistent Sync Field Error has been detected since the last RSTSTA
  1: The USART is configured as a Slave node and a LIN Inconsistent Sync Field Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **LINBE: LIN Bit Error**
  0: No Bit Error has been detected since the last RSTSTA.
  1: A Bit Error has been detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **MANERR: Manchester Error**
  0: No Manchester error has been detected since the last RSTSTA.
  1: At least one Manchester error has been detected since the last RSTSTA.

- **CTS/LINBLS: Image of CTS Input, or LIN Bus Line Status (when in LIN mode)**
  0: CTS or LIN Bus Line is low.
  1: CTS or LIN Bus Line is high.
- **DCD: Image of DCD Input**
  0: DCD is low.
  1: DCD is high.
- **DSR: Image of DSR Input**
  0: DSR is low.
  1: DSR is high.
- **RI: Image of RI Input**
  0: RI is low.
  1: RI is high.
- **CTSIC: Clear to Send Input Change Flag**
  0: No change has been detected on the CTS pin since the last CSR read.
  1: At least one change has been detected on the CTS pin since the last CSR read.
  This bit is cleared when reading CSR.
- **DCDIC: Data Carrier Detect Input Change Flag**
  0: No change has been detected on the DCD pin since the last CSR read.
  1: At least one change has been detected on the DCD pin since the last CSR read.
  This bit is cleared when reading CSR.
- **DSRIC: Data Set Ready Input Change Flag**
  0: No change has been detected on the DSR pin since the last CSR read.
  1: At least one change has been detected on the DSR pin since the last CSR read.
  This bit is cleared when reading CSR.
- **RIIC: Ring Indicator Input Change Flag**
  0: No change has been detected on the RI pin since the last CSR read.
  1: At least one change has been detected on the RI pin since the last CSR read.
  This bit is cleared when reading CSR.
- **LINTC: LIN Transfer Completed**
  0: The USART is either idle or a LIN transfer is ongoing.
  1: A LIN transfer has been completed since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA:
- **LINID: LIN Identifier**
  0: No LIN Identifier has been sent or received.
  1: A LIN Identifier has been sent (master) or received (slave), since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA:
- **NACK: Non Acknowledge**
  0: No Non Acknowledge has been detected since the last RSTNACK.
  1: At least one Non Acknowledge has been detected since the last RSTNACK.
  This bit is cleared by writing a one to CR.RSTNACK.
- **RXBUFF: Reception Buffer Full**
  0: The Buffer Full signal from the Peripheral DMA Controller channel is inactive.
  1: The Buffer Full signal from the Peripheral DMA Controller channel is active.
- **ITER/UNRE: Max Number of Repetitions Reached or SPI Underrun Error**
  If USART operates in SPI Slave Mode:
  UNRE = 0: No SPI underrun error has occurred since the last RSTSTA.
  UNRE = 1: At least one SPI underrun error has occurred since the last RSTSTA.
  If USART does not operate in SPI Slave Mode, no functionality is associated to UNRE. The bit will behave as ITER if the USART is in ISO7816 mode:
  ITER = 0: Maximum number of repetitions has not been reached since the last RSTSTA.
  ITER = 1: Maximum number of repetitions has been reached since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.

- **TXEMPTY: Transmitter Empty**
  0: The transmitter is either disabled or there are characters in THR, or in the transmit shift register.
  1: There are no characters in neither THR, nor in the transmit shift register.
  This bit is cleared by writing a one to CR.STTBRK.
- **TIMEOUT: Receiver Time-out**
  0: There has not been a time-out since the last Start Time-out command (CR.STTTO), or RTOR.TO is zero.
  1: There has been a time-out since the last Start Time-out command.
  This bit is cleared by writing a one to CR.STTTO or CR.RETTO.
- **PARE: Parity Error**
  0: Either no parity error has been detected, or the parity bit is a zero in multidrop mode, since the last RSTSTA.
  1: Either at least one parity error has been detected, or the parity bit is a one in multidrop mode, since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **FRAME: Framing Error**
  0: No stop bit has been found as low since the last RSTSTA.
  1: At least one stop bit has been found as low since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **OVRE: Overrun Error**
  0: No overrun error has occurred since the last RSTSTA.
  1: At least one overrun error has occurred since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **RXBRK: Break Received/End of Break**
  0: No Break received or End of Break detected since the last RSTSTA.
  1: Break received or End of Break detected since the last RSTSTA.
  This bit is cleared by writing a one to CR.RSTSTA.
- **TXRDY: Transmitter Ready**
  0: The transmitter is either disabled, or a character in THR is waiting to be transferred to the transmit shift register, or an STTBRK command has been requested. As soon as the transmitter is enabled, TXRDY is set.
  1: There is no character in the THR.
  This bit is cleared by writing a one to CR.STTBRK.
- **RXRDY: Receiver Ready**
  0: The receiver is either disabled, or no complete character has been received since the last read of RHR. If characters were being received when the receiver was disabled, RXRDY is set when the receiver is enabled.
  1: At least one complete character has been received and RHR has not yet been read.
  This bit is cleared when the Receive Holding Register (RHR) is read.

24.7.7 Receiver Holding Register

Name: RHR

Access Type: Read-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RXSYNH	–	–	–	–	–	–	RXCHR[8]

7	6	5	4	3	2	1	0
RXCHR[7:0]							

Reading this register will clear the CSR.RXRDY bit.

- **RXSYNH: Received Sync**
  0: Last character received is a data sync.
  1: Last character received is a command sync.
- **RXCHR: Received Character**
  Last received character.

24.7.8 Transmitter Holding Register

Name: THR

Access Type: Write-only

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TXSYNH	–	–	–	–	–	–	TXCHR[8]

7	6	5	4	3	2	1	0
TXCHR[7:0]							

- **TXSYNH: Sync Field to be transmitted**
  0: If MR.VARSYNC is one, the next character sent is encoded as data, and the start frame delimiter is a data sync.
  1: If MR.VARSYNC is one, the next character sent is encoded as a command, and the start frame delimiter is a command sync.
- **TXCHR: Character to be Transmitted**
  If TXRDY is zero this field contains the next character to be transmitted.

24.7.9 Baud Rate Generator Register

Name: BRGR

Access Type: Read-write

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	FP		

15	14	13	12	11	10	9	8
CD[15:8]							

7	6	5	4	3	2	1	0
CD[7:0]							

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **FP: Fractional Part**
  0: Fractional divider is disabled.
  1 - 7: Baud rate resolution, defined by FP x 1/8.
- **CD: Clock Divider**

Table 24-24. Baud Rate in Asynchronous Mode (MR.SYNC is 0)

CD	OVER = 0	OVER = 1
0	Baud Rate Clock Disabled	
1 to 65535	$\text{Baud Rate} = \frac{\text{Selected Clock}}{16 \cdot \text{CD}}$	$\text{Baud Rate} = \frac{\text{Selected Clock}}{8 \cdot \text{CD}}$

Table 24-25. Baud Rate in Synchronous Mode (MR.SYNC is 1) and SPI Mode(MR.MODE is 0xE or 0xF)

CD	Baud Rate
0	Baud Rate Clock Disabled
1 to 65535	$\text{Baud Rate} = \frac{\text{Selected Clock}}{\text{CD}}$

Table 24-26. Baud Rate in ISO7816 Mode

CD	Baud Rate
0	Baud Rate Clock Disabled
1 to 65535	$\text{Baud Rate} = \frac{\text{Selected Clock}}{\text{FI\_DI\_RATIO} \cdot \text{CD}}$

24.7.10 Receiver Time-out Register

Name: RTOR

Access Type: Read-write

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	TO[16]

15	14	13	12	11	10	9	8
TO[15:8]							

7	6	5	4	3	2	1	0
TO[7:0]							

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **TO: Time-out Value**
  0: The receiver Time-out is disabled.
  1 - 131071: The receiver Time-out is enabled and the time-out delay is TO x bit period.
  Note that the size of the TO counter is device dependent, refer to the Module Configuration section.

24.7.11 Transmitter Timeguard Register

Name: TTGR

Access Type: Read-write

Offset: 0x28

Reset Value: 0x00000000

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

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **TG: Timeguard Value**
  0: The transmitter Timeguard is disabled.
  1 - 255: The transmitter timeguard is enabled and the timeguard delay is TG bit periods.

24.7.12 FI DI Ratio Register

Name: FIDI

Access Type: Read-write

Offset: 0x40

Reset Value: 0x00000174

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	FI_DI_RATIO[10:8]		
7	6	5	4	3	2	1	0
FI_DI_RATIO[7:0]							

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **FI_DI_RATIO: FI Over DI Ratio Value**
  0: If ISO7816 mode is selected, the baud rate generator does not generate a signal.
  1 - 2047: If ISO7816 mode is selected, the baud rate is the clock provided on CLK divided by FI_DI_RATIO.

24.7.13 Number of Errors Register

Name: NER

Access Type: Read-only

Offset: 0x44

Reset Value: **0x00000000**

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

- **NB_ERRORS: Number of Errors**
  Total number of errors that occurred during an ISO7816 transfer. This register is automatically cleared when read.

24.7.14 IrDA Filter Register

Name: IFR

Access Type: Read-write

Offset: 0x4C

Reset Value: 0x00000000

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

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **IRDA_FILTER: IrDA Filter**
  Configures the IrDA demodulator filter.

24.7.15 Manchester Configuration Register

Name: MAN

Access Type: Read-write

Offset: 0x50

Reset Value: 0x30011004

31	30	29	28	27	26	25	24
–	DRIFT	1	RX_MPOL	–	–	RX_PP	
23	22	21	20	19	18	17	16
–	–	–	–	RX_PL			
15	14	13	12	11	10	9	8
–	–	–	TX_MPOL	–	–	TX_PP	
7	6	5	4	3	2	1	0
–	–	–	–	TX_PL			

This register can only be written if write protection is disabled in the "Write Protect Mode Register" (WPMR.WPEN is zero).

- **DRIFT: Drift cCompensation**
  0: The USART can not recover from a clock drift.
  1: The USART can recover from clock drift (only available in 16x oversampling mode).
- **RX_MPOL: Receiver Manchester Polarity**
  0: Zeroes are encoded as zero-to-one transitions, and ones are encoded as a one-to-zero transitions.
  1: Zeroes are encoded as one-to-zero transitions, and ones are encoded as a zero-to-one transitions.
- **RX_PP: Receiver Preamble Pattern detected**

Table 24-27.

RX_PP		Preamble Pattern default polarity assumed (RX_MPOL field not set)
0	0	ALL_ONE
0	1	ALL_ZERO
1	0	ZERO_ONE
1	1	ONE_ZERO

- **RX_PL: Receiver Preamble Length**
  0: The receiver preamble pattern detection is disabled.
  1 - 15: The detected preamble length is RX_PL bit periods.
- **TX_MPOL: Transmitter Manchester Polarity**
  0: Zeroes are encoded as zero-to-one transitions, and ones are encoded as a one-to-zero transitions.
  1: Zeroes are encoded as one-to-zero transitions, and ones are encoded as a zero-to-one transitions.

- **TX_PP: Transmitter Preamble Pattern**

Table 24-28.

TX_PP		Preamble Pattern default polarity assumed (TX_MPOL field not set)
0	0	ALL_ONE
0	1	ALL_ZERO
1	0	ZERO_ONE
1	1	ONE_ZERO

- **TX_PL: Transmitter Preamble Length**

  0: The transmitter preamble pattern generation is disabled.

  1 - 15: The preamble length is TX_PL bit periods.

24.7.16 LIN Mode Register

Name: LINMR

Access Type: Read-write

Offset: 0x54

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	SYNCDIS	PDCM

15	14	13	12	11	10	9	8
DLC							

7	6	5	4	3	2	1	0
WKUPTYP	FSDIS	DLM	CHKTYP	CHKDIS	PARDIS	NACT	

- **SYNCDIS: Synchronization Disable**
  0: LIN slave synchronization is enabled.
  1: LIN slave synchronization is disabled.
- **PDCM: Peripheral DMA Controller Mode**
  0: The LIN mode register is not written by the Peripheral DMA Controller.
  1: The LIN mode register, except for this bit, is written by the Peripheral DMA Controller.
- **DLC: Data Length Control**
  0 - 255: If DLM=0 this field defines the response data length to DLC+1 bytes.
- **WKUPTYP: Wakeup Signal Type**
  0: Writing a one to CR.LINWKUP will send a LIN 2.0 wakeup signal.
  1: Writing a one to CR.LINWKUP will send a LIN 1.3 wakeup signal.
- **FSDIS: Frame Slot Mode Disable**
  0: The Frame Slot mode is enabled.
  1: The Frame Slot mode is disabled.
- **DLM: Data Length Mode**
  0: The response data length is defined by DLC.
  1: The response data length is defined by bits 4 and 5 of the Identifier (LINIR.IDCHR).
- **CHKTYP: Checksum Type**
  0: LIN 2.0 “Enhanced” checksum
  1: LIN 1.3 “Classic” checksum
- **CHKDIS: Checksum Disable**
  0: Checksum is automatically computed and sent when master, and checked when slave.
  1: Checksum is not computed and sent, nor checked.
- **PARDIS: Parity Disable**
  0: Identifier parity is automatically computed and sent when master, and checked when slave.
  1: Identifier parity is not computed and sent, nor checked.

- **NACT: LIN Node Action**

Table 24-29.

NACT		Mode Description
0	0	PUBLISH: The USART transmits the response.
0	1	SUBSCRIBE: The USART receives the response.
1	0	IGNORE: The USART does not transmit and does not receive the response.
1	1	Reserved

24.7.17 LIN Identifier Register

Name: LINIR

Access Type: Read-write or Read-only

Offset: 0x58

Reset Value: 0x00000000

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

- **IDCHR: Identifier Character**

  If USART is in LIN master mode, the IDCHR field is read-write, and its value is the Identifier character to be transmitted.

  If USART is in LIN slave mode, the IDCHR field is read-only, and its value is the last received Identifier character.

24.7.18 LIN Baud Rate Register

Name: LINBRR

Access Type: Read-only

Offset: 0x5C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	LINFP		

15	14	13	12	11	10	9	8
LINCD							

7	6	5	4	3	2	1	0
LINCD							

- **LINFP: LIN Fractional Part after Synchronization**
- **LINCD: LIN Clock Divider after Synchronization**

24.7.19 Write Protect Mode Register

Register Name: WPMR

Access Type: Read-write

Offset: 0xE4

Reset Value: See Table 24-17

31	30	29	28	27	26	25	24
WPKEY[23:16]							
23	22	21	20	19	18	17	16
WPKEY[15:8]							
15	14	13	12	11	10	9	8
WPKEY[7:0]							
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	WPEN

- **WPKEY: Write Protect KEY**
  Has to be written to 0x555341 (“USA” in ASCII) in order to successfully write WPEN. This bit always reads as zero. Writing the correct key to this field clears WPSR.WPVSRC and WPSR.WPVS.
- **WPEN: Write Protect Enable**
  0: Write protection disabled.
  1: Write protection enabled.

Protects the registers:

- “Mode Register” on page 625
- “Baud Rate Generator Register” on page 636
- “Receiver Time-out Register” on page 638
- “Transmitter Timeguard Register” on page 639
- “FI DI Ratio Register” on page 640
- “IrDA Filter Register” on page 642
- “Manchester Configuration Register” on page 643

24.7.20 Write Protect Status Register

Register Name: WPSR

Access Type: Read-only

Offset: 0xE8

Reset Value: See Table 24-17

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
WPVSRC[15:8]							
15	14	13	12	11	10	9	8
WPVSRC[7:0]							
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	WPVS

- **WPVSRC: Write Protect Violation Source**
  If WPVS is one, this field indicates which write-protected register was unsuccessfully written to, either by address offset or code.
- **WPVS: Write Protect Violation Status**
  0: No write protect violation has occurred since the last WPSR read.
  1: A write protect violation has occurred since the last WPSR read.

Note: Reading WPSR automatically clears all fields. Writing the correct key to WPSR.WPKEY clears all fields.

24.7.21 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	MFN			
15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			
7	6	5	4	3	2	1	0
VERSION[7:0]							

- **MFN**
  Reserved. No functionality associated.
- **VERSION**
  Version of the module. No functionality associated.

24.8 Module Configuration

The specific configuration for each USART instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to the Power Manager chapter for details.

Table 24-30. Module configuration

Feature	USART0	USART1	USART2	USART3
SPI Logic	Implemented	Implemented	Implemented	Implemented
LIN Logic	Implemented	Not Implemented	Not Implemented	Not Implemented
Manchester Logic	Implemented	Not Implemented	Not Implemented	Not Implemented
IRDA Logic	Implemented	Not Implemented	Not Implemented	Not Implemented
RS485 Logic	Implemented	Not Implemented	Not Implemented	Not Implemented
Fractional Baudrate	Implemented	Implemented	Implemented	Implemented
ISO7816	Implemented	Not Implemented	Not Implemented	Not Implemented
DIV value for divided CLK_USART	8	8	8	8
Receiver Time-out Counter Size (Size of the RTOR.TO field)	17-bits	8-bits	8-bits	8-bits

Table 24-31. Module clock name

Module name	Clock name
USART0	**CLK_USART0**
USART1	**CLK_USART1**
USART2	**CLK_USART2**
USART3	**CLK_USART3**

Table 24-32. Register Reset Values

Register	Reset Value
VERSION	0x00000602

25. Picopower UART (PICOUART)

Rev: 1.0.1.0

25.1 Features

- **Ultra low power UART RX line with fixed format :**
  - **9600 bauds**
  - **1 start bit**
  - **8-bit data**
  - **No parity bit**
  - **1 stop bit**
- **Use very low power 32kHz or 1kHz reference clock (Internal RC or Crystal)**
- **Event system support and device wake-up source**
  - **start bit detection**
  - **full frame reception**
  - **character recognition**
- **All low power modes supported included backup mode**

25.2 Overview

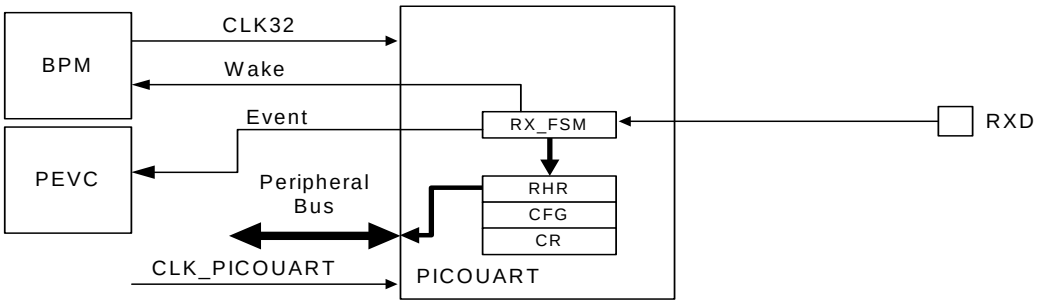
The Picopower UART peripheral provides a minimalistic UART RX line feature in any of the low power modes including backup mode. It allows the device to be waken up from any of low-power modes by monitoring the RX line. It also able to generate an event to the Peripheral Event Controller (PEVC).

The PICOUART is usually associated with another USART in the system, sharing the same RXD pin. The PICOUART is then used to wake-up the device and the USART provides the full UART RX/TX feature when the device is waken up

The number of PICOUART modules implemented is device specific. Refer to the Module Configuration section for details.

25.3 Block Diagram

Figure 25-1. PICOUART Block Diagram



25.4 I/O Lines Description

Table 25-1. I/O Lines Description

Pin Name	Pin Description	Type
RXD	Receive Serial Data	Input

25.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

25.5.1 I/O Lines

The PICOUART pin is directly connected to the RXD pin, meaning the user doesn't need to configure the I/O Controller to give control of the pin to the PICOUART.

25.5.2 Power Management

PICOUART remains active when the system enters a Power Save Mode that disables its CLK_PICOUART clock.

25.5.3 Clocks

The configuration clock for PICOUART (CLK_PICOUART) is generated by the Power Manager. It can be enabled or disabled either manually through the user interface of the Power Manager or automatically when the system enters a Power Save Mode that disables the clocks to the peripheral bus modules. The running clock for PICOUART (CLK32) is generated by the Backup Power Manager and is necessary for the PICOUART to operate. The source of this clock can be either the RC32K or the OSC32K, which must be enabled using the Backup System Control Interface.

25.5.4 Backup Power Management

The PICOUART wake-up line is connected to the Backup Power Manager. It allows to wake-up the device from the current Power Save Mode. The user must first configure the Backup Power Manager to enable the wake-up source accordingly.

25.5.5 Peripheral Events

The PICOUART peripheral events are connected via the Peripheral Event System. Refer to the Peripheral Event System chapter for details.

25.6 Functional Description

25.6.1 Reception Operation

PICOUART Reception is enabled by writing a one to the RX Enable bit in the Control Register (CR.RXEN). PICOUART Reception is disabled by writing a one to the RX Disable bit in the Control Register (CR.RXDIS). Reception is made by sampling the RXD pin on the CLK32 reference clock. The decoding of the frame format is hardly coded and limited to the following configuration:

- 9600 bauds if CLK32 is equal to 32,768 kHz
- One start bit
- Eight-bit data with LSB first

- One Stop bit
- No parity bit

A frame is valid when it matches exactly with this configuration. Valid received byte is extracted from the frame and stored in the Receive Holding Register (RHR.CDATA). When a valid data byte is stored, the Data Ready bit is set to one in the Status Register (SR.DRDY). DRDY is set to zero when a new start bit is detected.

There is no overrun detection, meaning that RHR value is valid only when DRDY is equal to one.

25.6.2 Wakeup from Power Save Mode

PICOUART is able to wakeup the device from any Power Save Mode including Backup Mode. The use should first enable the wakeup source by programming the Configuration register (CFG.SOURCE) and select the source action to be a wake-up by programming the Configuration register (CFG.ACTION) .

There are three wakeup sources:

- Start bit detection: As soon as a start bit is detected , the PICOUART wakes up the device and normal PICOUART operations are perfomed, leading to DRDY set to one if a valid frame is finally received.
- Full frame reception: when a valid frame is received, the valid received byte is stored in RHR and DRDY is set to one. Then the device is waken up
- Character recognition: when a valid frame is received, the valid received byte is stored in RHR and DRDY is set to one. If the received byte match the data stored in CFG.MATCH, then the device is waken u.p

Only one wakeup source can be enabled at a time.

25.6.3 Peripheral Event

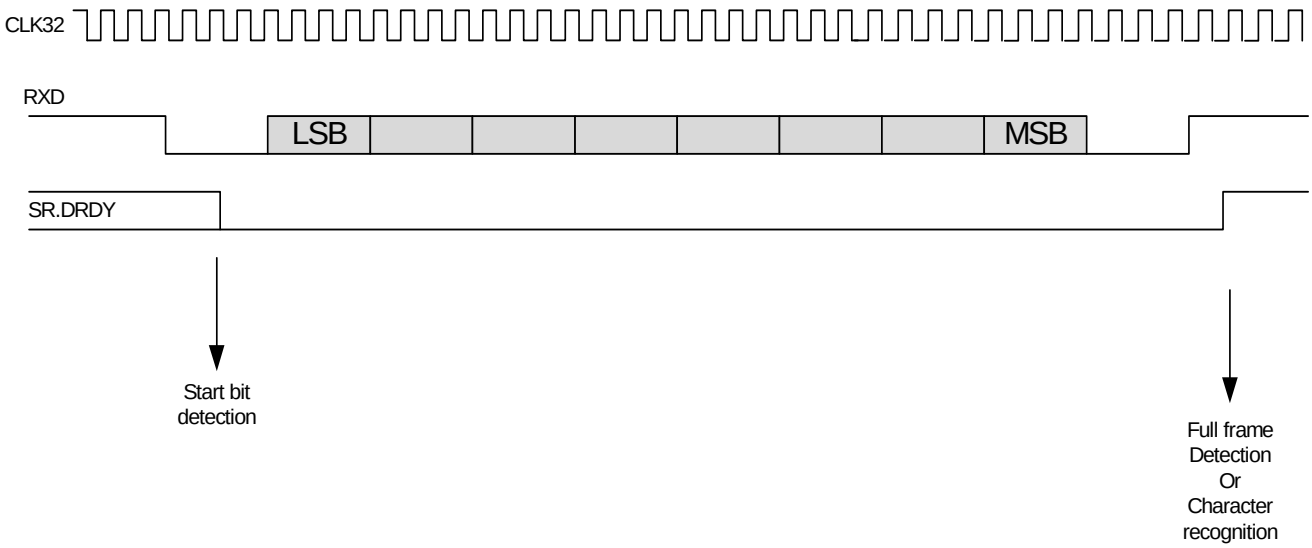
PICOUART is able to generate an event to the PEVC. The use should first enable the event source by programming the Configuration register (CFG.SOURCE) and select the source action to be a wake-up by programming the Configuration register (CFG.ACTION) .

There are three event sources:

- Start bit detection: As soon as a start bit is detected , the PICOUART generates an event and normal PICOUART operations are perfomed, leading to DRDY set to one if a valid frame is finally received.
- Full frame reception: when a valid frame is received, the valid received byte is stored in RHR and DRDY is set to one. Then the event is generated.
- Character recognition: when a valid frame is received, the valid received byte is stored in RHR and DRDY is set to one. If the received byte match the data stored in CFG.MATCH, then the event is generated.

Only one even source can be enabled at a time.

Figure 25-2. Frame Valid Waveform



25.7 User Interface

Table 25-2. PICOUART Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-only	0x00000000
0x04	Configuration Register	CFG	Read/Write	0x00000000
0x08	Status Register	SR	Read-only	0x00000000
0x0C	Receive Holding Register	RHR	Read	0x00000000
0x20	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

25.7.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: 0x00000000

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

- **DIS: Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables PICOUART.
  This bit always reads as zero.
- **EN: Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables PICOUART.
  This bit always reads as zero.

25.7.2 Configuration Register

Name: CFG

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
MATCH							

7	6	5	4	3	2	1	0
-	-	-	-	-	ACTION	SOURCE	

To avoid unexpected behavior CFG must be written when PICOUART is disabled.

- **MATCH: Data Match**
  Data used in characterer recognition, only used when SOURCE = 11.
- **ACTION: Action to perform**
  Action to perform when SOURCE is not equal to 00.
  0: Wakeup the device.
  1: Generate an event.
- **SOURCE: Source Enable Mode**

SOURCE		Mode
0	0	Wake up and event disable
0	1	Wake up or event enable on start bit detection
1	0	Wake up or event enable on full frame reception
1	1	Wake up or event enable on character recognition

25.7.3 Status Register

Name: SR

Access Type: Read-only

Offset: 0x08

Reset Value: 0x00000000

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

- **DRDY: Data Ready**
  0: No data is ready in RHR.
  1: A new data is ready.
  This bit is cleared when a new start bit is detected.
  This bit is set when a valid frame is received.
- **EN: Enable Status**
  0: PICOUART is disabled.
  1: PICOUART is enabled.

25.7.4 Receive Holding Register

Name: RHR

Access Type: Read/Write

Offset: 0x0C

Reset Value: 0x00000000

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

- **CDATA: Received Data**

25.7.5 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x20

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

25.8 Module Configuration

The specific configuration for each PICOUART instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 25-3. PICOUART Clocks

Module Name	Clock Name	Description
PICOUART	CLK_PICOUART	Peripheral Clock for PICOUART

Table 25-4. PICOUART/USART association

Module Name	USART Name	Description
PICOUART	USART0	USART instance associated with PICOUART

Table 25-5. PICOUART RXD Pin

Module Name	RXD Pin number	Description
PICOUART	PB00	RXD pin number

Table 25-6. Register Reset Values

Register	Reset Value
VERSION	0x00000101

26. Serial Peripheral Interface (SPI)

Rev: 2.1.1.3

26.1 Features

- **Compatible with an embedded 32-bit microcontroller**
- **Supports communication with serial external devices**
  - **Four chip selects with external decoder support allow communication with up to 15 peripherals**
  - **Serial memories, such as DataFlash and 3-wire EEPROMs**
  - **Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and Sensors**
  - **External co-processors**
- **Master or Slave Serial Peripheral Bus Interface**
  - **4 - to 16-bit programmable data length per chip select**
  - **Programmable phase and polarity per chip select**
  - **Programmable transfer delays between consecutive transfers and between clock and data per chip select**
  - **Programmable delay between consecutive transfers**
  - **Selectable mode fault detection**
- **Connection to Peripheral DMA Controller channel capabilities optimizes data transfers**
  - **One channel for the receiver, one channel for the transmitter**
  - **Next buffer support**
  - **Four character FIFO in reception**

26.2 Overview

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master"' which controls the data flow, while the other devices act as “slaves'' which have data shifted into and out by the master. Different CPUs can take turn being masters (Multiple Master Protocol opposite to Single Master Protocol where one CPU is always the master while all of the others are always slaves) and one master may simultaneously shift data into multiple slaves. However, only one slave may drive its output to write data back to the master at any given time.

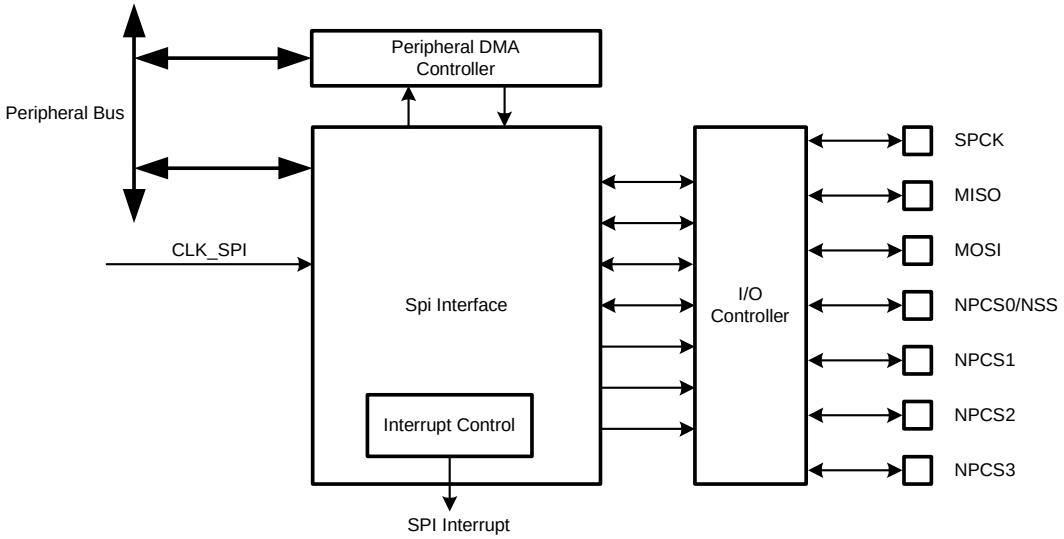
A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): this data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO): this data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK): this control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates; the SPCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): this control line allows slaves to be turned on and off by hardware.

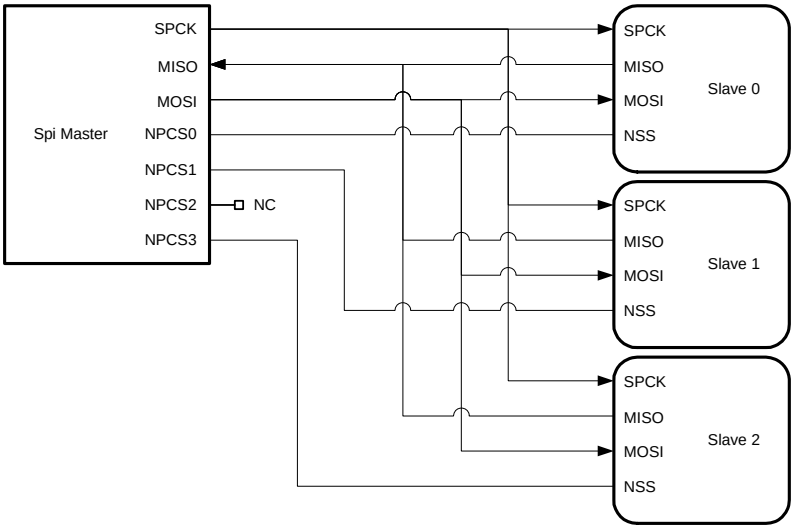
26.3 Block Diagram

Figure 26-1. SPI Block Diagram



26.4 Application Block Diagram

Figure 26-2. Application Block Diagram: Single Master/Multiple Slave Implementation



26.5 I/O Lines Description

Table 26-1. I/O Lines Description

Pin Name	Pin Description	Type	
		Master	Slave
MISO	Master In Slave Out	Input	Output
MOSI	Master Out Slave In	Output	Input
SPCK	Serial Clock	Output	Input
NPCS1-NPCS3	Peripheral Chip Selects	Output	Unused
NPCS0/NSS	Peripheral Chip Select/Slave Select	Output	Input

26.6 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

26.6.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with I/O lines. The user must first configure the I/O Controller to assign the SPI pins to their peripheral functions.

26.6.2 Clocks

The clock for the SPI bus interface (CLK_SPI) is generated by the Power Manager. This clock is disabled at reset, and can be enabled in the Power Manager. It is recommended to disable the SPI before disabling the clock, to avoid freezing the SPI in an undefined state.

26.6.3 Interrupts

The SPI interrupt request line is connected to the NVIC. Using the SPI interrupt requires the NVIC to be programmed first.

26.7 Functional Description

26.7.1 Modes of Operation

The SPI operates in master mode or in slave mode.

Operation in master mode is configured by writing a one to the Master/Slave Mode bit in the Mode Register (MR.MSTR). The pins NPCS0 to NPCS3 are all configured as outputs, the SPCK pin is driven, the MISO line is wired on the receiver input and the MOSI line driven as an output by the transmitter.

If the MR.MSTR bit is written to zero, the SPI operates in slave mode. The MISO line is driven by the transmitter output, the MOSI line is wired on the receiver input, the SPCK pin is driven by the transmitter to synchronize the receiver. The NPCS0 pin becomes an input, and is used as a Slave Select signal (NSS). The pins NPCS1 to NPCS3 are not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operations. The baud rate generator is activated only in master mode.

26.7.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is configured with the Clock Polarity bit in the Chip Select Registers (CSRn.CPOL). The clock phase is configured with the Clock Phase bit in the CSRn registers (CSRn.NCPHA). These two bits determine the edges of the clock signal on which data is driven and sampled. Each of the two bits has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are used and fixed in different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

Table 26-2 on page 667 shows the four modes and corresponding parameter settings.

Table 26-2. SPI modes

SPI Mode	CPOL	NCPHA
0	0	1
1	0	0
2	1	1
3	1	0

Figure 26-3 on page 667 and Figure 26-4 on page 668 show examples of data transfers.

Figure 26-3. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)

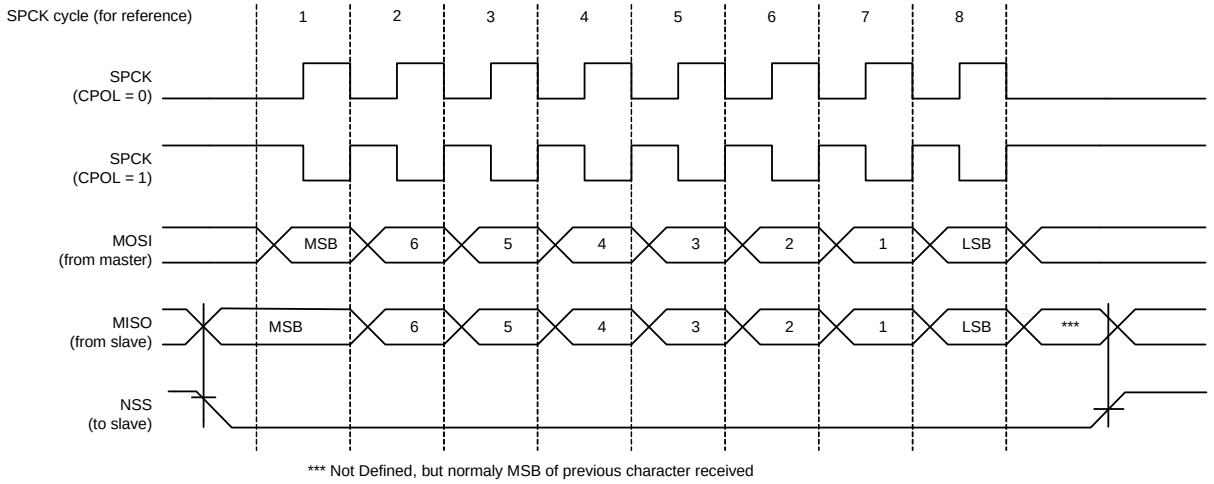
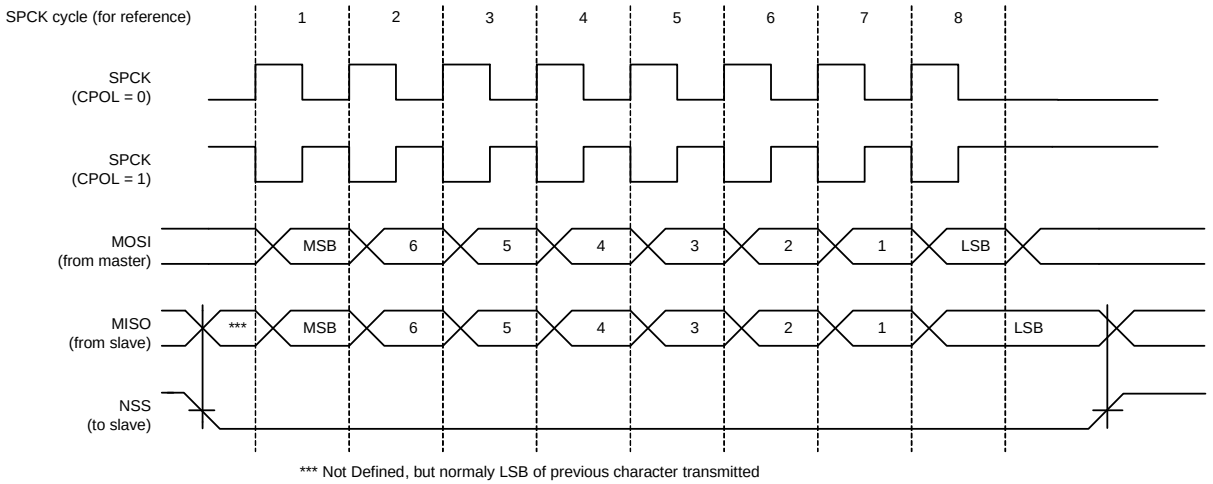


Figure 26-4. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)



26.7.3 Master Mode Operations

When configured in master mode, the SPI uses the internal programmable baud rate generator as clock source. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (TDR) and the Receive Data Register (RDR), and a single Shift Register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer begins when the processor writes to the TDR register. The written data is immediately transferred in the Shift Register and transfer on the SPI bus starts. While the data in the Shift Register is shifted on the MOSI line, the MISO line is sampled and shifted in the Shift Register. Transmission cannot occur without reception.

Before writing to the TDR, the Peripheral Chip Select field in TDR (TDR.PCS) must be written in order to select a slave.

If new data is written to TDR during the transfer, it stays in it until the current transfer is completed. Then, the received data is transferred from the Shift Register to RDR, the data in TDR is loaded in the Shift Register and a new transfer starts.

The transfer of a data written in TDR in the Shift Register is indicated by the Transmit Data Register Empty bit in the Status Register (SR.TDRE). When new data is written in TDR, this bit is cleared. The SR.TDRE bit is used to trigger the Transmit Peripheral DMA Controller channel.

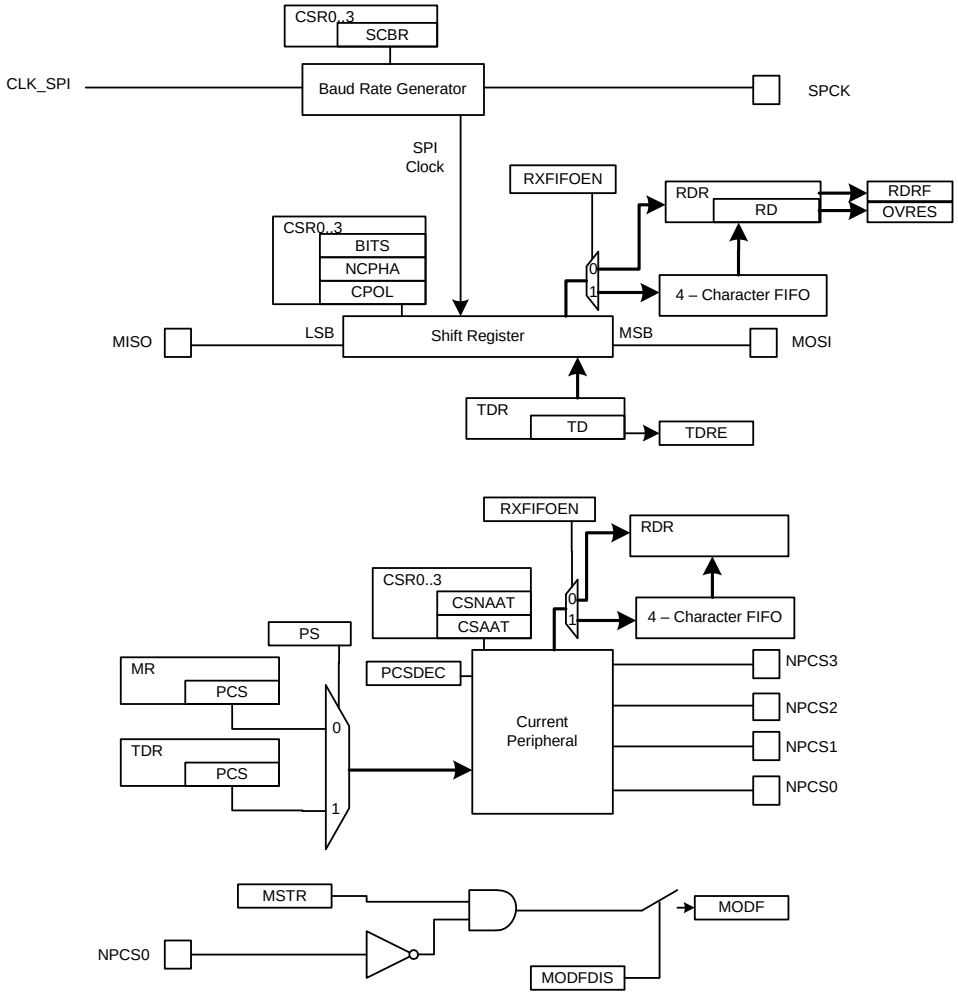
The end of transfer is indicated by the Transmission Registers Empty bit in the SR register (SR.TXEMPTY). If a transfer delay (CSRn.DLYBCT) is greater than zero for the last transfer, SR.TXEMPTY is set after the completion of said delay. The CLK_SPI can be switched off at this time.

During reception, received data are transferred from the Shift Register to the reception FIFO. The FIFO can contain up to 4 characters (both Receive Data and Peripheral Chip Select fields). While a character of the FIFO is unread, the Receive Data Register Full bit in SR remains high (SR.RDRF). Characters are read through the RDR register. If the four characters stored in the FIFO are not read and if a new character is stored, this sets the Overrun Error Status bit in the SR register (SR.OVRES). The procedure to follow in such a case is described in Section 26.7.3.8.

Figure 26-5 on page 669shows a block diagram of the SPI when operating in master mode. Figure 26-6 on page 670 shows a flow chart describing how transfers are handled.

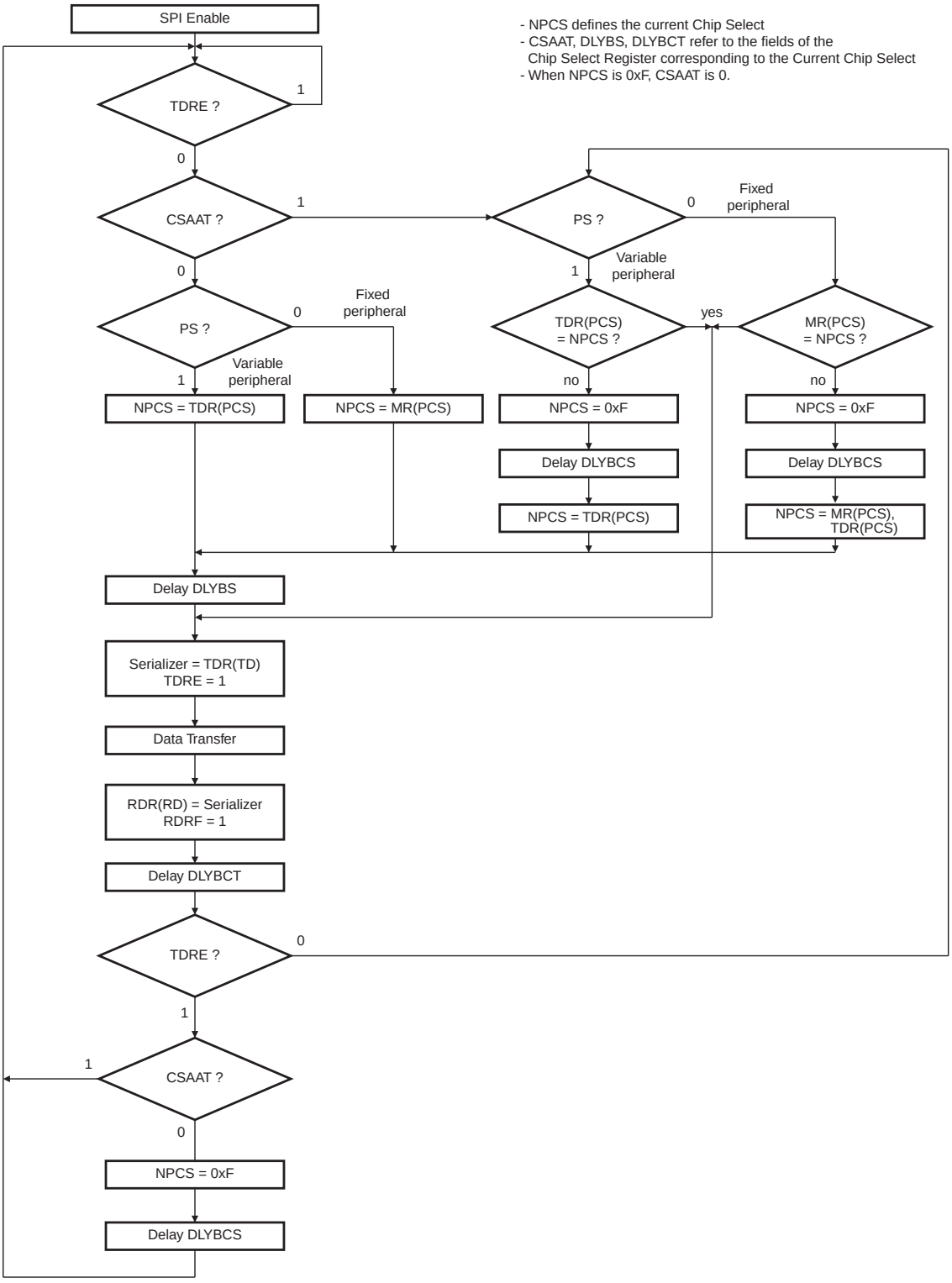
26.7.3.1 Master Mode Block Diagram

Figure 26-5. Master Mode Block Diagram



26.7.3.2 Master Mode Flow Diagram

Figure 26-6. Master Mode Flow Diagram



26.7.3.3 Clock Generation

The SPI Baud rate clock is generated by dividing the CLK_SPI , by a value between 1 and 255.

This allows a maximum operating baud rate at up to CLK_SPI and a minimum operating baud rate of CLK_SPI divided by 255.

Writing the Serial Clock Baud Rate field in the CSRn registers (CSRn.SCBR) to zero is forbidden. Triggering a transfer while CSRn.SCBR is zero can lead to unpredictable results.

At reset, CSRn.SCBR is zero and the user has to configure it at a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be configured in the CSRn.SCBR field. This allows the SPI to automatically adapt the baud rate for each interfaced peripheral without reprogramming.

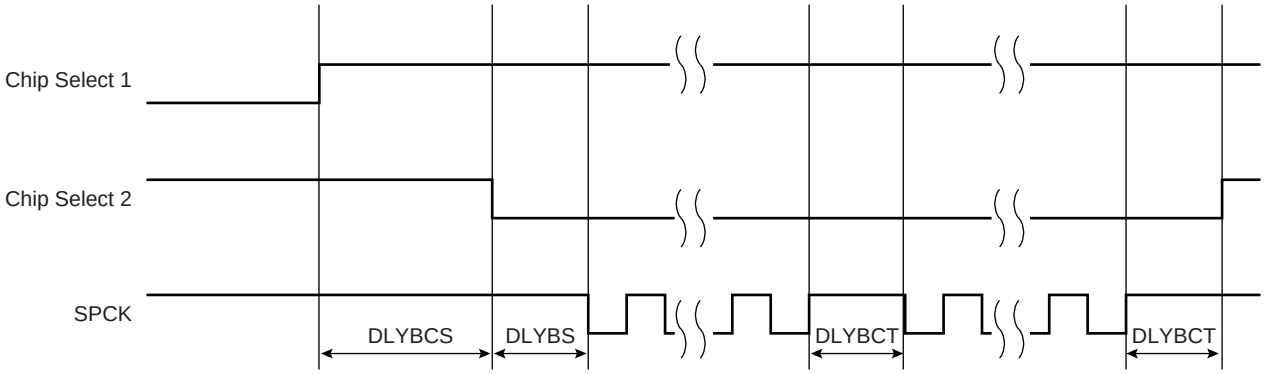
26.7.3.4 Transfer Delays

Figure 26-7 on page 671 shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be configured to modify the transfer waveforms:

- The delay between chip selects, programmable only once for all the chip selects by writing to the Delay Between Chip Selects field in the MR register (MR.DLYBCS). Allows insertion of a delay between release of one chip select and before assertion of a new one.
- The delay before SPCK, independently programmable for each chip select by writing the Delay Before SPCK field in the CSRn registers (CSRn.DLYBS). Allows the start of SPCK to be delayed after the chip select has been asserted.
- The delay between consecutive transfers, independently programmable for each chip select by writing the Delay Between Consecutive Transfers field in the CSRn registers (CSRn.DLYBCT). Allows insertion of a delay between two transfers occurring on the same chip select

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

Figure 26-7. Programmable Delays



26.7.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS3 signals. By default, all the NPCS signals are high before and after each transfer.

The peripheral selection can be performed in two different ways:

- Fixed Peripheral Select: SPI exchanges data with only one peripheral
- Variable Peripheral Select: Data can be exchanged with more than one peripheral

Fixed Peripheral Select is activated by writing a zero to the Peripheral Select bit in MR (MR.PS). In this case, the current peripheral is defined by the MR.PCS field and the TDR.PCS field has no effect.

Variable Peripheral Select is activated by writing a one to the MR.PS bit . The TDR.PCS field is used to select the current peripheral. This means that the peripheral selection can be defined for each new data.

The Fixed Peripheral Selection allows buffer transfers with a single peripheral. Using the Peripheral DMA Controller is an optimal means, as the size of the data transfer between the memory and the SPI is either 4 bits or 16 bits. However, changing the peripheral selection requires the Mode Register to be reprogrammed.

The Variable Peripheral Selection allows buffer transfers with multiple peripherals without reprogramming the MR register. Data written to TDR is 32-bits wide and defines the real data to be transmitted and the peripheral it is destined to. Using the Peripheral DMA Controller in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs, however the SPI still controls the number of bits (8 to16) to be transferred through MISO and MOSI lines with the CSRn registers. This is not the optimal means in term of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

26.7.3.6 Peripheral Chip Select Decoding

The user can configure the SPI to operate with up to 15 peripherals by decoding the four Chip Select lines, NPCS0 to NPCS3 with an external logic. This can be enabled by writing a one to the Chip Select Decode bit in the MR register (MR.PCSDEC).

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e. driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

When operating with decoding, the SPI directly outputs the value defined by the PCS field of either the MR register or the TDR register (depending on PS).

As the SPI sets a default value of 0xF on the chip select lines (i.e. all chip select lines at one) when not processing any transfer, only 15 peripherals can be decoded.

The SPI has only four Chip Select Registers, not 15. As a result, when decoding is activated, each chip select defines the characteristics of up to four peripherals. As an example, the CRS0 register defines the characteristics of the externally decoded peripherals 0 to 3, corresponding to the PCS values 0x0 to 0x3. Thus, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 3, 4 to 7, 8 to 11 and 12 to 14.

26.7.3.7 Peripheral Deselection

When operating normally, as soon as the transfer of the last data written in TDR is completed, the NPCS lines all rise. This might lead to runtime error if the processor is too long in responding

to an interrupt, and thus might lead to difficulties for interfacing with some serial peripherals requiring the chip select line to remain active during a full set of transfers.

To facilitate interfacing with such devices, the CSRn registers can be configured with the Chip Select Active After Transfer bit written to one (CSRn.CSAAT) . This allows the chip select lines to remain in their current state (low = active) until transfer to another peripheral is required.

When the CSRn.CSAAT bit is written to qero, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the SR.TDRE bit rises as soon as the content of the TDR is transferred into the internal shifter. When this bit is detected the TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not de-asserted between the two transfers. This might lead to difficulties for interfacing with some serial peripherals requiring the chip select to be de-asserted after each transfer. To facilitate interfacing with such devices, the CSRn registers can be configured with the Chip Select Not Active After Transfer bit (CSRn.CSNAAT) written to one. This allows to de-assert systematically the chip select lines during a time DLYBCS. (The value of the CSRn.CSNAAT bit is taken into account only if the CSRn.CSAAT bit is written to zero for the same Chip Select).

Figure 26-8 on page 674 shows different peripheral deselection cases and the effect of the CSRn.CSAAT and CSRn.CSNAAT bits.

26.7.3.8 FIFO Management

A FIFO has been implemented in Reception FIFO (both in master and in slave mode), in order to be able to store up to 4 characters without causing an overrun error. If an attempt is made to store a fifth character, an overrun error rises. If such an event occurs, the FIFO must be flushed. There are two ways to Flush the FIFO:

- By performing four read accesses of the RDR (the data read must be ignored)
- By writing a one to the Flush Fifo Command bit in the CR register (CR.FLUSHFIFO).

After that, the SPI is able to receive new data.

Figure 26-8. Peripheral Deselection

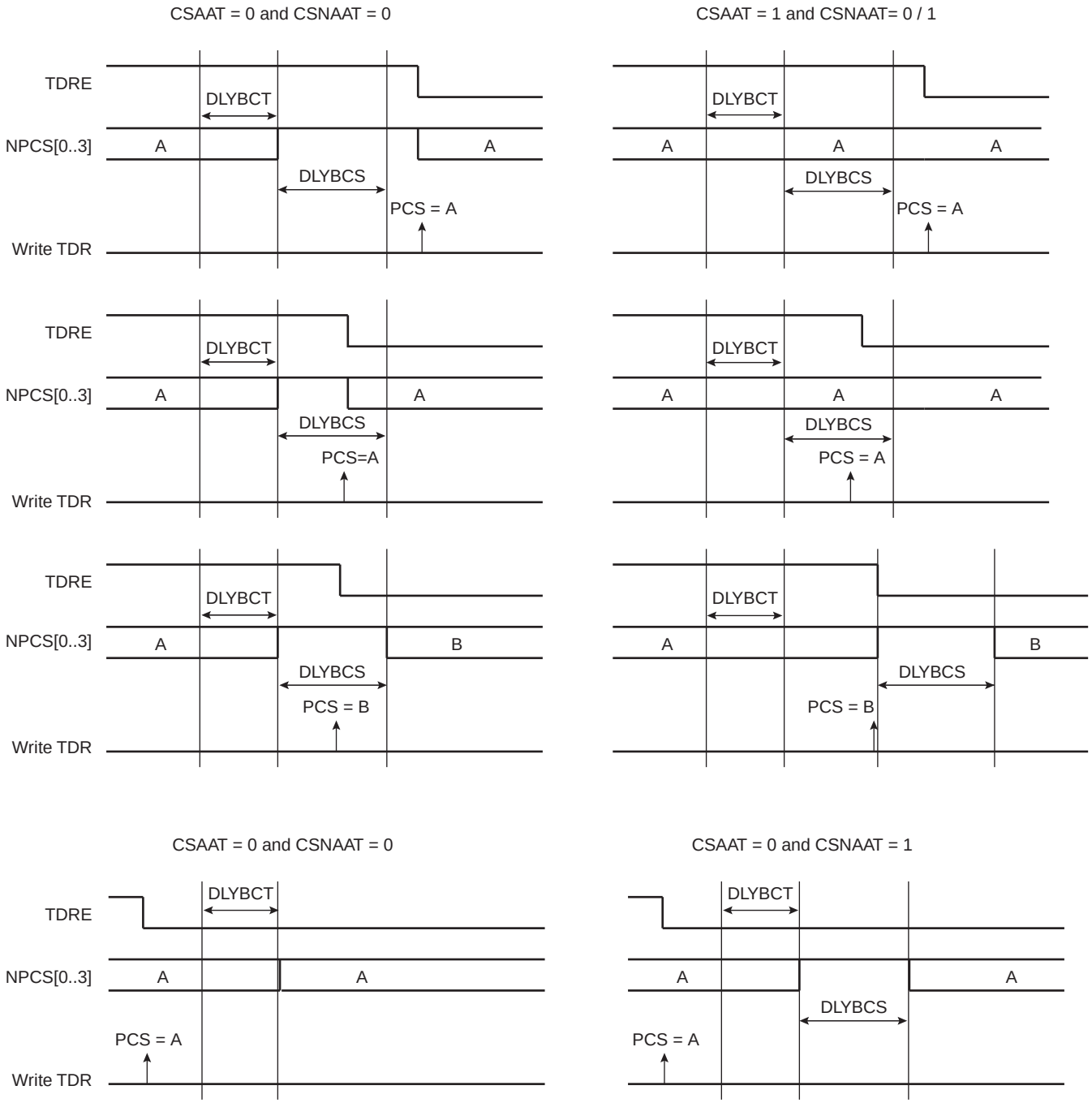


Figure 26-8 on page 674 shows different peripheral deselection cases and the effect of the CSRn.CSAAT and CSRn.CSNAAT bits.

26.7.3.9 Mode Fault Detection

The SPI is capable of detecting a mode fault when it is configured in master mode and NPCS0, MOSI, MISO, and SPCK are configured as open drain through the I/O Controller with either internal or external pullup resistors. If the I/O Controller does not have open-drain capability, mode fault detection **must** be disabled by writing a one to the Mode Fault Detection bit in the MR

register (MR.MODFDIS). In systems with open-drain I/O lines, a mode fault is detected when a low level is driven by an external master on the NPCS0/NSS signal.

When a mode fault is detected, the Mode Fault Error bit in the SR (SR.MODF) is set until the SR is read and the SPI is automatically disabled until re-enabled by writing a one to the SPI Enable bit in the CR register (CR.SPIEN).

By default, the mode fault detection circuitry is enabled. The user can disable mode fault detection by writing a one to the Mode Fault Detection bit in the MR register (MR.MODFDIS).

26.7.4 SPI Slave Mode

When operating in slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits for NSS to go active before receiving the serial clock from an external master. When NSS falls, the clock is validated on the serializer, which processes the number of bits defined by the Bits Per Transfer field of the Chip Select Register 0 (CSR0.BITS). These bits are processed following a phase and a polarity defined respectively by the CSR0.NCPHA and CSR0.CPOL bits. Note that the BITS, CPOL, and NCPHA bits of the other Chip Select Registers have no effect when the SPI is configured in Slave Mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

When all the bits are processed, the received data is transferred in the Receive Data Register and the SR.RDRF bit rises. If the RDR register has not been read before new data is received, the SR.OVRES bit is set. Data is loaded in RDR even if this flag is set. The user has to read the SR register to clear the SR.OVRES bit.

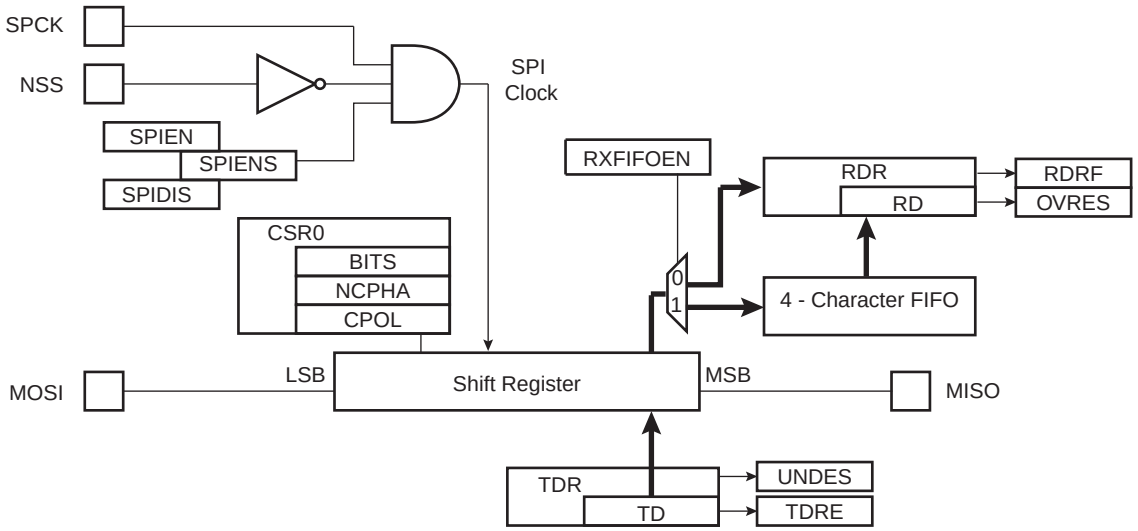
When a transfer starts, the data shifted out is the data present in the Shift Register. If no data has been written in the TDR register, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the Shift Register resets to zero.

When a first data is written in TDR, it is transferred immediately in the Shift Register and the SR.TDRE bit rises. If new data is written, it remains in TDR until a transfer occurs, i.e. NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in TDR is transferred in the Shift Register and the SR.TDRE bit rises. This enables frequent updates of critical variables with single transfers.

Then, a new data is loaded in the Shift Register from the TDR. In case no character is ready to be transmitted, i.e. no character has been written in TDR since the last load from TDR to the Shift Register, the Shift Register is not modified and the last received character is retransmitted. In this case the Underrun Error Status bit is set in SR (SR.UNDES).

Figure 26-9 on page 676 shows a block diagram of the SPI when operating in slave mode.

Figure 26-9. Slave Mode Functional Block Diagram



26.8 User Interface

Table 26-3. SPI Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-only	0x00000000
0x04	Mode Register	MR	Read/Write	0x00000000
0x08	Receive Data Register	RDR	Read-only	0x00000000
0x0C	Transmit Data Register	TDR	Write-only	0x00000000
0x10	Status Register	SR	Read-only	0x00000000
0x14	Interrupt Enable Register	IER	Write-only	0x00000000
0x18	Interrupt Disable Register	IDR	Write-only	0x00000000
0x1C	Interrupt Mask Register	IMR	Read-only	0x00000000
0x30	Chip Select Register 0	CSR0	Read/Write	0x00000000
0x34	Chip Select Register 1	CSR1	Read/Write	0x00000000
0x38	Chip Select Register 2	CSR2	Read/Write	0x00000000
0x3C	Chip Select Register 3	CSR3	Read/Write	0x00000000
0xE4	Write Protection Control Register	WPCR	Read/Write	0X00000000
0xE8	Write Protection Status Register	WPSR	Read-only	0x00000000
0xF8	Features Register	FEATURES	Read-only	- (1)
0xFC	Version Register	VERSION	Read-only	- (1)

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

26.8.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	LASTXFER
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	FLUSHFIFO
7	6	5	4	3	2	1	0
SWRST	-	-	-	-	-	SPIDIS	SPIEN

- **LASTXFER: Last Transfer**
  1: The current NPCS will be deasserted after the character written in TD has been transferred. When CSRn.CSAAT is one, this allows to close the communication with the current serial peripheral by raising the corresponding NPCS line as soon as TD transfer has completed.
  0: Writing a zero to this bit has no effect.
- **FLUSHFIFO: Flush Fifo Command**
  1: If The FIFO Mode is enabled (MR.FIFOEN written to one) and if an overrun error has been detected, this command allows to empty the FIFO.
  0: Writing a zero to this bit has no effect.
- **SWRST: SPI Software Reset**
  1: Writing a one to this bit will reset the SPI. A software-triggered hardware reset of the SPI interface is performed. The SPI is in slave mode after software reset. Peripheral DMA Controller channels are not affected by software reset.
  0: Writing a zero to this bit has no effect.
- **SPIDIS: SPI Disable**
  1: Writing a one to this bit will disable the SPI. As soon as SPIDIS is written to one, the SPI finishes its transfer, all pins are set in input mode and no data is received or transmitted. If a transfer is in progress, the transfer is finished before the SPI is disabled. If both SPIEN and SPIDIS are equal to one when the CR register is written, the SPI is disabled.
  0: Writing a zero to this bit has no effect.
- **SPIEN: SPI Enable**
  1: Writing a one to this bit will enable the SPI to transfer and receive data.
  0: Writing a zero to this bit has no effect.

26.8.2 Mode Register

Name: MR

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DLYBCS							
23	22	21	20	19	18	17	16
-	-	-	-	PCS			
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
LLB	RXFIFOEN	-	MODFDIS	-	PCSDEC	PS	MSTR

- **DLYBCS: Delay Between Chip Selects**
  This field defines the delay from NPCS inactive to the activation of another NPCS. The DLYBCS time guarantees non-overlapping chip selects and solves bus contentions in case of peripherals having long data float times.
  If DLYBCS is less than or equal to six, six CLK_SPI periods will be inserted by default.
  Otherwise, the following equation determines the delay:

  $$\text{Delay Between Chip Selects} = \frac{DLYBCS}{CLKSPI}$$

- **PCS: Peripheral Chip Select**
  This field is only used if Fixed Peripheral Select is active (PS = 0).
  If PCSDEC = 0:
  PCS = xxx0NPCS[3:0] = 1110
  PCS = xx01NPCS[3:0] = 1101
  PCS = x011NPCS[3:0] = 1011
  PCS = 0111NPCS[3:0] = 0111
  PCS = 1111forbidden (no peripheral is selected)
  (x = don't care)
  If PCSDEC = 1:
  NPCS[3:0] output signals = PCS.
- **LLB: Local Loopback Enable**
  1: Local loopback path enabled. LLB controls the local loopback on the data serializer for testing in master mode only (MISO is internally connected on MOSI).
  0: Local loopback path disabled.
- **RXFIFOEN: FIFO in Reception Enable**
  1: The FIFO is used in reception (four characters can be stored in the SPI).

0: The FIFO is not used in reception (only one character can be stored in the SPI).

- **MODFDIS: Mode Fault Detection**
  1: Mode fault detection is disabled. If the I/O controller does not have open-drain capability, mode fault detection **must** be disabled for proper operation of the SPI.
  0: Mode fault detection is enabled.
- **PCSDEC: Chip Select Decode**
  0: The chip selects are directly connected to a peripheral device.
  1: The four chip select lines are connected to a 4- to 16-bit decoder.
  When PCSDEC equals one, up to 15 Chip Select signals can be generated with the four lines using an external 4- to 16-bit decoder. The CSRn registers define the characteristics of the 15 chip selects according to the following rules:
  CSR0 defines peripheral chip select signals 0 to 3.
  CSR1 defines peripheral chip select signals 4 to 7.
  CSR2 defines peripheral chip select signals 8 to 11.
  CSR3 defines peripheral chip select signals 12 to 14.
- **PS: Peripheral Select**
  1: Variable Peripheral Select.
  0: Fixed Peripheral Select.
- **MSTR: Master/Slave Mode**
  1: SPI is in master mode.
  0: SPI is in slave mode.

26.8.3 Receive Data Register

Name: RDR

Access Type: Read-only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
RD[15:8]							

7	6	5	4	3	2	1	0
RD[7:0]							

- **RD: Receive Data**
  Data received by the SPI Interface is stored in this register right-justified. Unused bits read zero.

26.8.4 Transmit Data Register

Name: TDR

Access Type: Write-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	LASTXFER

23	22	21	20	19	18	17	16
-	-	-	-	PCS			

15	14	13	12	11	10	9	8
TD[15:8]							

7	6	5	4	3	2	1	0
TD[7:0]							

- **LASTXFER: Last Transfer**

  1: The current NPCS will be deasserted after the character written in TD has been transferred. When CSRn.CSAAT is one, this allows to close the communication with the current serial peripheral by raising the corresponding NPCS line as soon as TD transfer has completed.
  0: Writing a zero to this bit has no effect.

  This field is only used if Variable Peripheral Select is active (MR.PS = 1).
- **PCS: Peripheral Chip Select**

  If PCSDEC = 0:
  PCS = xxx0NPCS[3:0] = 1110
  PCS = xx01NPCS[3:0] = 1101
  PCS = x011NPCS[3:0] = 1011
  PCS = 0111NPCS[3:0] = 0111
  PCS = 1111forbidden (no peripheral is selected)
  (x = don’t care)
  If PCSDEC = 1:
  NPCS[3:0] output signals = PCS

  This field is only used if Variable Peripheral Select is active (MR.PS = 1).
- **TD: Transmit Data**

  Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the TDR register in a right-justified format.

26.8.5 Status Register

Name: SR

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	SPIENS
15	14	13	12	11	10	9	8
-	-	-	-	-	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
-	-	-	-	OVRES	MODF	TDRE	RDRF

- **SPIENS: SPI Enable Status**
  1: This bit is set when the SPI is enabled.
  0: This bit is cleared when the SPI is disabled.
- **UNDES: Underrun Error Status (Slave Mode Only)**
  1: This bit is set when a transfer begins whereas no data has been loaded in the TDR register.
  0: This bit is cleared when the SR register is read.
- **TXEMPTY: Transmission Registers Empty**
  1: This bit is set when TDR and internal shifter are empty. If a transfer delay has been defined, TXEMPTY is set after the completion of such delay.
  0: This bit is cleared as soon as data is written in TDR.
- **NSSR: NSS Rising**
  1: A rising edge occurred on NSS pin since last read.
  0: This bit is cleared when the SR register is read.
- **OVRES: Overrun Error Status**
  1: This bit is set when an overrun has occurred. An overrun occurs when RDR is loaded at least twice from the serializer since the last read of the RDR.
  0: This bit is cleared when the SR register is read.
- **MODF: Mode Fault Error**
  1: This bit is set when a Mode Fault occurred.
  0: This bit is cleared when the SR register is read.
- **TDRE: Transmit Data Register Empty**
  1: This bit is set when the last data written in the TDR register has been transferred to the serializer.
  0: This bit is cleared when data has been written to TDR and not yet transferred to the serializer.
  TDRE equals zero when the SPI is disabled or at reset. The SPI enable command sets this bit to one.
- **RDRF: Receive Data Register Full**
  1: Data has been received and the received data has been transferred from the serializer to RDR since the last read of RDR.
  0: No data has been received since the last read of RDR

26.8.6 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
-	-	-	-	OVRES	MODF	TDRE	RDRF

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

26.8.7 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	UNDES	TXEMPTY	NSSR

7	6	5	4	3	2	1	0
-	-	-	-	OVRES	MODF	TDRE	RDRF

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

26.8.8 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
-	-	-	-	OVRES	MODF	TDRE	RDRF

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

26.8.9 Chip Select Register 0

Name: CSR0

Access Type: Read/Write

Offset: 0x30

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DLYBCT							

23	22	21	20	19	18	17	16
DLYBS							

15	14	13	12	11	10	9	8
SCBR							

7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

- **DLYBCT: Delay Between Consecutive Transfers**

  This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

  When DLYBCT equals zero, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

  Otherwise, the following equation determines the delay:

  $$\text{Delay Between Consecutive Transfers} = \frac{32 \times DLYBCT}{CLKSPI}$$

- **DLYBS: Delay Before SPCK**

  This field defines the delay from NPCS valid to the first valid SPCK transition.

  When DLYBS equals zero, the NPCS valid to SPCK transition is 1/2 the SPCK clock period.

  Otherwise, the following equations determine the delay:

  $$\text{Delay Before SPCK} = \frac{DLYBS}{CLKSPI}$$

- **SCBR: Serial Clock Baud Rate**

  In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the CLK_SPI. The Baud rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK baud rate:

  $$\text{SPCK Baudrate} = \frac{CLKSPI}{SCBR}$$

  Writing the SCBR field to zero is forbidden. Triggering a transfer while SCBR is zero can lead to unpredictable results.

  At reset, SCBR is zero and the user has to write it to a valid value before performing the first transfer.

  If a clock divider (SCBRn) field is set to one and the other SCBR fields differ from one, access on CSn is correct but no correct access will be possible on other CS.

- **BITS: Bits Per Transfer**
  The BITS field determines the number of data bits transferred. Reserved values should not be used.

BITS	Bits Per Transfer
0000	8
0001	9
0010	10
0011	11
0100	12
0101	13
0110	14
0111	15
1000	16
1001	4
1010	5
1011	6
1100	7
1101	Reserved
1110	Reserved
1111	Reserved

- **CSAAT: Chip Select Active After Transfer**
  1: The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.
  0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.
- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**
  0: The Peripheral Chip Select does not rise between two transfers if the TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.
  1: The Peripheral Chip Select rises systematically between each transfer performed on the same slave for a minimal duration of:

  $\frac{DLYBCS}{CLKSPI}$ (if DLYBCT field is different from 0)

  $\frac{DLYBCS + 1}{CLK\overline{SPI}}$ (if DLYBCT field equals 0)
- **NCPHA: Clock Phase**
  1: Data is captured after the leading (inactive-to-active) edge of SPCK and changed on the trailing (active-to-inactive) edge of SPCK.
  0: Data is changed on the leading (inactive-to-active) edge of SPCK and captured after the trailing (active-to-inactive) edge of SPCK.
  NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.
- **CPOL: Clock Polarity**
  1: The inactive state value of SPCK is logic level one.
  0: The inactive state value of SPCK is logic level zero.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

26.8.10 Chip Select Register 1

Name: CSR1

Access Type: Read/Write

Offset: 0x34

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DLYBCT							

23	22	21	20	19	18	17	16
DLYBS							

15	14	13	12	11	10	9	8
SCBR							

7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

- **DLYBCT: Delay Between Consecutive Transfers**

  This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

  When DLYBCT equals zero, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

  Otherwise, the following equation determines the delay:

  $$\text{Delay Between Consecutive Transfers} = \frac{32 \times DLYBCT}{CLKSPI}$$

- **DLYBS: Delay Before SPCK**

  This field defines the delay from NPCS valid to the first valid SPCK transition.

  When DLYBS equals zero, the NPCS valid to SPCK transition is 1/2 the SPCK clock period.

  Otherwise, the following equations determine the delay:

  $$\text{Delay Before SPCK} = \frac{DLYBS}{CLKSPI}$$

- **SCBR: Serial Clock Baud Rate**

  In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the CLK_SPI. The Baud rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK baud rate:

  $$\text{SPCK Baudrate} = \frac{CLKSPI}{SCBR}$$

  Writing the SCBR field to zero is forbidden. Triggering a transfer while SCBR is zero can lead to unpredictable results.

  At reset, SCBR is zero and the user has to write it to a valid value before performing the first transfer.

  If a clock divider (SCBRn) field is set to one and the other SCBR fields differ from one, access on CSn is correct but no correct access will be possible on other CS.

- **BITS: Bits Per Transfer**
  The BITS field determines the number of data bits transferred. Reserved values should not be used.

BITS	Bits Per Transfer
0000	8
0001	9
0010	10
0011	11
0100	12
0101	13
0110	14
0111	15
1000	16
1001	4
1010	5
1011	6
1100	7
1101	Reserved
1110	Reserved
1111	Reserved

- **CSAAT: Chip Select Active After Transfer**
  1: The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.
  0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.
- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**
  0: The Peripheral Chip Select does not rise between two transfers if the TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.
  1: The Peripheral Chip Select rises systematically between each transfer performed on the same slave for a minimal duration of:

  $\frac{DLYBCS}{CLKSPI}$ (if DLYBCT field is different from 0)

  $\frac{DLYBCS + 1}{CLK\overline{SPI}}$ (if DLYBCT field equals 0)
- **NCPHA: Clock Phase**
  1: Data is captured after the leading (inactive-to-active) edge of SPCK and changed on the trailing (active-to-inactive) edge of SPCK.
  0: Data is changed on the leading (inactive-to-active) edge of SPCK and captured after the trailing (active-to-inactive) edge of SPCK.
  NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.
- **CPOL: Clock Polarity**
  1: The inactive state value of SPCK is logic level one.
  0: The inactive state value of SPCK is logic level zero.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

26.8.11 Chip Select Register 2

Name: CSR2

Access Type: Read/Write

Offset: 0x38

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DLYBCT							
23	22	21	20	19	18	17	16
DLYBS							
15	14	13	12	11	10	9	8
SCBR							
7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

- **DLYBCT: Delay Between Consecutive Transfers**

  This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

  When DLYBCT equals zero, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

  Otherwise, the following equation determines the delay:

  $$\text{Delay Between Consecutive Transfers} = \frac{32 \times DLYBCT}{CLKSPI}$$

- **DLYBS: Delay Before SPCK**

  This field defines the delay from NPCS valid to the first valid SPCK transition.

  When DLYBS equals zero, the NPCS valid to SPCK transition is 1/2 the SPCK clock period.

  Otherwise, the following equations determine the delay:

  $$\text{Delay Before SPCK} = \frac{DLYBS}{CLKSPI}$$

- **SCBR: Serial Clock Baud Rate**

  In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the CLK_SPI. The Baud rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK baud rate:

  $$\text{SPCK Baudrate} = \frac{CLKSPI}{SCBR}$$

  Writing the SCBR field to zero is forbidden. Triggering a transfer while SCBR is zero can lead to unpredictable results.

  At reset, SCBR is zero and the user has to write it to a valid value before performing the first transfer.

  If a clock divider (SCBRn) field is set to one and the other SCBR fields differ from one, access on CSn is correct but no correct access will be possible on other CS.

- **BITS: Bits Per Transfer**
  The BITS field determines the number of data bits transferred. Reserved values should not be used.

BITS	Bits Per Transfer
0000	8
0001	9
0010	10
0011	11
0100	12
0101	13
0110	14
0111	15
1000	16
1001	4
1010	5
1011	6
1100	7
1101	Reserved
1110	Reserved
1111	Reserved

- **CSAAT: Chip Select Active After Transfer**
  1: The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.
  0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.
- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**
  0: The Peripheral Chip Select does not rise between two transfers if the TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.
  1: The Peripheral Chip Select rises systematically between each transfer performed on the same slave for a minimal duration of:

  $\frac{DLYBCS}{CLKSPI}$ (if DLYBCT field is different from 0)

  $\frac{DLYBCS + 1}{CLK\overline{SPI}}$ (if DLYBCT field equals 0)
- **NCPHA: Clock Phase**
  1: Data is captured after the leading (inactive-to-active) edge of SPCK and changed on the trailing (active-to-inactive) edge of SPCK.
  0: Data is changed on the leading (inactive-to-active) edge of SPCK and captured after the trailing (active-to-inactive) edge of SPCK.
  NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.
- **CPOL: Clock Polarity**
  1: The inactive state value of SPCK is logic level one.
  0: The inactive state value of SPCK is logic level zero.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

26.8.12 Chip Select Register 3

Name: CSR3

Access Type: Read/Write

Offset: 0x3C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DLYBCT							

23	22	21	20	19	18	17	16
DLYBS							

15	14	13	12	11	10	9	8
SCBR							

7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

- **DLYBCT: Delay Between Consecutive Transfers**

  This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

  When DLYBCT equals zero, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

  Otherwise, the following equation determines the delay:

  $$\text{Delay Between Consecutive Transfers} = \frac{32 \times DLYBCT}{CLKSPI}$$

- **DLYBS: Delay Before SPCK**

  This field defines the delay from NPCS valid to the first valid SPCK transition.

  When DLYBS equals zero, the NPCS valid to SPCK transition is 1/2 the SPCK clock period.

  Otherwise, the following equations determine the delay:

  $$\text{Delay Before SPCK} = \frac{DLYBS}{CLKSPI}$$

- **SCBR: Serial Clock Baud Rate**

  In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the CLK_SPI. The Baud rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK baud rate:

  $$\text{SPCK Baudrate} = \frac{CLKSPI}{SCBR}$$

  Writing the SCBR field to zero is forbidden. Triggering a transfer while SCBR is zero can lead to unpredictable results.

  At reset, SCBR is zero and the user has to write it to a valid value before performing the first transfer.

  If a clock divider (SCBRn) field is set to one and the other SCBR fields differ from one, access on CSn is correct but no correct access will be possible on other CS.

- **BITS: Bits Per Transfer**
  The BITS field determines the number of data bits transferred. Reserved values should not be used.

BITS	Bits Per Transfer
0000	8
0001	9
0010	10
0011	11
0100	12
0101	13
0110	14
0111	15
1000	16
1001	4
1010	5
1011	6
1100	7
1101	Reserved
1110	Reserved
1111	Reserved

- **CSAAT: Chip Select Active After Transfer**
  1: The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.
  0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.
- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**
  0: The Peripheral Chip Select does not rise between two transfers if the TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.
  1: The Peripheral Chip Select rises systematically between each transfer performed on the same slave for a minimal duration of:

  $\frac{DLYBCS}{CLKSPI}$ (if DLYBCT field is different from 0)

  $\frac{DLYBCS + 1}{CLK\overline{SPI}}$ (if DLYBCT field equals 0)
- **NCPHA: Clock Phase**
  1: Data is captured after the leading (inactive-to-active) edge of SPCK and changed on the trailing (active-to-inactive) edge of SPCK.
  0: Data is changed on the leading (inactive-to-active) edge of SPCK and captured after the trailing (active-to-inactive) edge of SPCK.
  NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.
- **CPOL: Clock Polarity**
  1: The inactive state value of SPCK is logic level one.
  0: The inactive state value of SPCK is logic level zero.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

26.8.13 Write Protection Control Register

Register Name: WPCR

Access Type: Read-write

Offset: 0xE4

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
SPIWPKEY[23:16]							
23	22	21	20	19	18	17	16
SPIWPKEY[15:8]							
15	14	13	12	11	10	9	8
SPIWPKEY[7:0]							
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	SPIWPEN

- **SPIWPKEY: SPI Write Protection Key Password**

  If a value is written in SPIWPEN, the value is taken into account only if SPIWPKEY is written with “SPI” (SPI written in ASCII Code, i.e. 0x535049 in hexadecimal).

- **SPIWPEN: SPI Write Protection Enable**

  1: The Write Protection is Enabled.

  0: The Write Protection is Disabled.

26.8.14 Write Protection Status Register

Register Name: WPSR

Access Type: Read-only

Offset: 0xE8

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
SPIWPVSRC							

7	6	5	4	3	2	1	0
-	-	-	-	-	SPIWPVS		

- **SPIWPVSRC: SPI Write Protection Violation Source**
  This Field indicates the Peripheral Bus Offset of the register concerned by the violation (MR or CSRx)
- **SPIWPVS: SPI Write Protection Violation Status**

SPIWPVS value	Violation Type
1	The Write Protection has blocked a Write access to a protected register (since the last read).
2	Software Reset has been performed while Write Protection was enabled (since the last read or since the last write access on MR, IER, IDR or CSRx).
3	Both Write Protection violation and software reset with Write Protection enabled have occurred since the last read.
4	Write accesses have been detected on MR (while a chip select was active) or on CSRi (while the Chip Select “i” was active) since the last read.
5	The Write Protection has blocked a Write access to a protected register and write accesses have been detected on MR (while a chip select was active) or on CSRi (while the Chip Select “i” was active) since the last read.
6	Software Reset has been performed while Write Protection was enabled (since the last read or since the last write access on MR, IER, IDR or CSRx) and some write accesses have been detected on MR (while a chip select was active) or on CSRi (while the Chip Select “i” was active) since the last read.
7	- The Write Protection has blocked a Write access to a protected register. and - Software Reset has been performed while Write Protection was enabled. and - Write accesses have been detected on MR (while a chip select was active) or on CSRi (while the Chip Select “i” was active) since the last read.

26.8.15 Features Register

Register Name: FEATURES

Access Type: Read-only

Offset: 0xF8

Reset Value: –

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	SWIMPL	FIFORIMPL	BRPBHSB	CSNAATIMPL	EXTDEC

15	14	13	12	11	10	9	8
LENNCONF							LENCONF

7	6	5	4	3	2	1	0
PHZNCONF	PHCONF	PPNCONF	PCONF	NCS			

- **SWIMPL: Spurious Write Protection Implemented**
  - 0: Spurious write protection is not implemented.
  - 1: Spurious write protection is implemented.
- **FIFORIMPL: FIFO in Reception Implemented**
  - 0: FIFO in reception is not implemented.
  - 1: FIFO in reception is implemented.
- **BRPBHSB: Bridge Type is PB to HSB**
  - 0: Bridge type is not PB to HSB.
  - 1: Bridge type is PB to HSB.
- **CSNAATIMPL: CSNAAT Features Implemented**
  - 0: CSNAAT (Chip select not active after transfer) features are not implemented.
  - 1: CSNAAT features are implemented.
- **EXTDEC: External Decoder True**
  - 0: External decoder capability is not implemented.
  - 1: External decoder capability is implemented.
- **LENNCONF: Character Length if not Configurable**
  - If the character length is not configurable, this field specifies the fixed character length.
- **LENCONF: Character Length Configurable**
  - 0: The character length is not configurable.
  - 1: The character length is configurable.
- **PHZNCONF: Phase is Zero if Phase not Configurable**
  - 0: If phase is not configurable, phase is non-zero.
  - 1: If phase is not configurable, phase is zero.
- **PHCONF: Phase Configurable**
  - 0: Phase is not configurable.
  - 1: Phase is configurable.

- **PPNCONF: Polarity Positive if Polarity not Configurable**
  0: If polarity is not configurable, polarity is negative.
  1: If polarity is not configurable, polarity is positive.
- **PCONF: Polarity Configurable**
  0: Polarity is not configurable.
  1: Polarity is configurable.
- **NCS: Number of Chip Selects**
  This field indicates the number of chip selects implemented.

26.8.16 Version Register

Register Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: –

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	MFN			

15	14	13	12	11	10	9	8
				VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **MFN**

  Reserved. No functionality associated.

- **VERSION**

  Version number of the module. No functionality associated.

26.9 Module Configuration

The specific configuration for each SPI instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 26-4. SPI Clock Name

Module Name	Clock Name	Description
SPI	CLK_SPI	Clock for the SPI bus interface

Table 26-5. Register Reset Values

Register	Reset Value
FEATURES	0x001F0154
VERSION	0x00000211

27. Two-wire Master Interface (TWIM)

Rev: 1.2.0.1

27.1 Features

- **Compatible with I²C standard**
  - **Multi-master support**
  - **Transfer speeds up to 3.4 Mbit/s**
  - **7- and 10-bit and General Call addressing**
- **Compatible with SMBus standard**
  - **Hardware Packet Error Checking (CRC) generation and verification with ACK control**
  - **25 ms clock low timeout delay**
  - **10 ms master cumulative clock low extend time**
  - **25 ms slave cumulative clock low extend time**
- **Compatible with PMBus**
- **Compatible with Atmel Two-wire Interface Serial Memories**
- **DMA interface for reducing CPU load**
- **Arbitrary transfer lengths, including 0 data bytes**
- **Optional clock stretching if transmit or receive buffers not ready for data transfer**

27.2 Overview

The Atmel Two-wire Master Interface (TWIM) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 3.4 Mbit/s, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus serial EEPROM and I²C compatible device such as a real time clock (RTC), dot matrix/graphic LCD controller, and temperature sensor, to name a few. The TWIM is always a bus master and can transfer sequential or single bytes. Multiple master capability is supported. Arbitration of the bus is performed internally and relinquishes the bus automatically if the bus arbitration is lost.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.Table 27-1 lists the compatibility level of the Atmel Two-wire Interface in Master Mode and a full I²C compatible device.

Table 27-1. Atmel TWIM Compatibility with I²C Standard

I²C Standard	Atmel TWIM
Standard-mode (100 kbit/s)	Supported
Fast-mode (400 kbit/s)	Supported
Fast-mode Plus (1 Mbit/s)	Supported
High-speed-mode (3.4 Mbit/s)	Supported
7- or 10-bits Slave Addressing	Supported
START BYTE(1)	Not Supported
Repeated Start (Sr) Condition	Supported
ACK and NACK Management	Supported
Slope Control and Input Filtering (Fast mode)	Supported
Clock Stretching	Supported

Note: 1. START + b000000001 + Ack + Sr

Table 27-2 lists the compatibility level of the Atmel Two-wire Master Interface and a full SMBus compatible master.

Table 27-2. Atmel TWIM Compatibility with SMBus Standard

SMBus Standard	Atmel TWIM
Bus Timeouts	Supported
Address Resolution Protocol	Supported
Host Functionality	Supported
Packet Error Checking	Supported

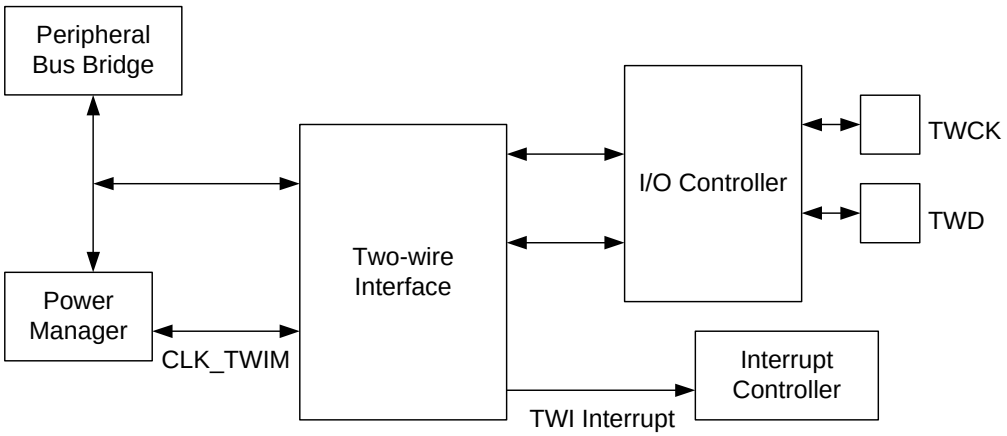
27.3 List of Abbreviations

Table 27-3. Abbreviations

Abbreviation	Description
TWI	Two-wire Interface
A	Acknowledge
NA	Non Acknowledge
P	Stop
S	Start
Sr	Repeated Start
SADR	Slave Address
ADR	Any address except SADR
R	Read
W	Write

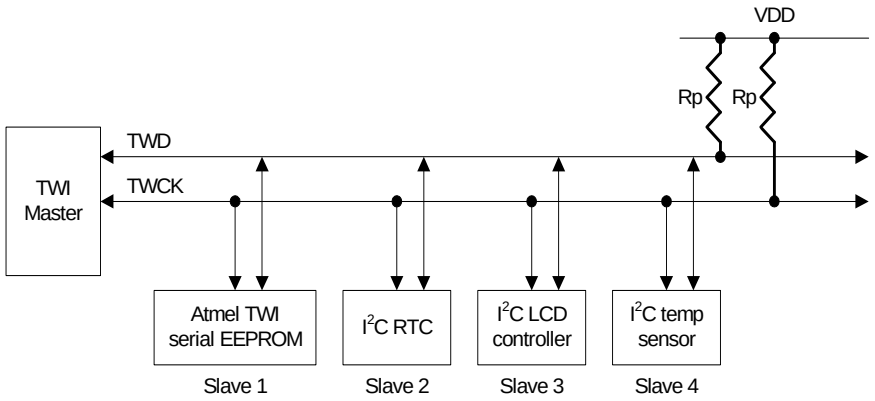
27.4 Block Diagram

Figure 27-1. Block Diagram



27.5 Application Block Diagram

Figure 27-2. Application Block Diagram



Rp: pull-up value as given by the I2C Standard

27.6 I/O Lines Description

Table 27-4. I/O Lines Description

Pin Name	Pin Description	Type
TWD	Two-wire Serial Data	Input/Output
TWCK	Two-wire Serial Clock	Input/Output

27.7 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

27.7.1 I/O Lines

TWD and TWCK are bidirectional lines, connected to a positive supply voltage via a current source or pull-up resistor (see Figure 27-4 on page 709). When the bus is free, both lines are high. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

The TWD and TWCK pins may be multiplexed with I/O Controller lines. To enable the TWIM, the user must perform the following steps:

- Program the I/O Controller to:
  - Dedicate TWD, TWCK as peripheral lines.
  - Define TWD, TWCK as open-drain.

27.7.2 Power Management

If the CPU enters a sleep mode that disables clocks used by the TWIM, the TWIM will stop functioning and resume operation after the system wakes up from sleep mode.

27.7.3 Clocks

The clock for the TWIM bus interface (CLK_TWIM) is generated by the Power Manager. It is recommended to disable the TWIM before disabling the clock, to avoid freezing the TWIM in an undefined state.

27.7.4 DMA

The TWIM DMA handshake interface is connected to the Peripheral DMA Controller. Using the TWIM DMA functionality requires the Peripheral DMA Controller to be programmed after setting up the TWIM.

27.7.5 Interrupts

The TWIM interrupt request lines are connected to the NVIC. Using the TWIM interrupts requires the NVIC to be programmed first.

27.7.6 Debug Operation

When an external debugger forces the CPU into debug mode, the TWIM continues normal operation. If the TWIM is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

27.8 Functional Description

27.8.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (see Figure 27-4).

Each transfer begins with a START condition and terminates with a STOP condition (see Figure 27-4).

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines a STOP condition.

Figure 27-3. START and STOP Conditions

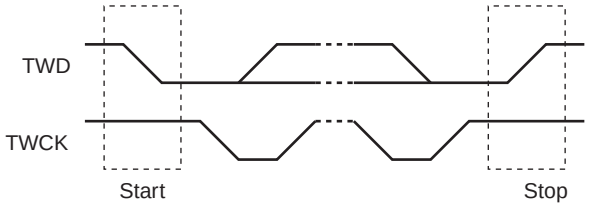
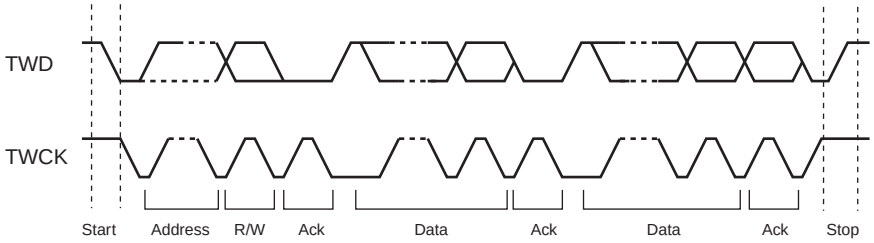


Figure 27-4. Transfer Format



27.8.2 Operation

The TWIM has two modes of operation:

- Master transmitter mode
- Master receiver mode

The master is the device which starts and stops a transfer and generates the TWCK clock. These modes are described in the following chapters.

27.8.2.1 High-speed-mode

After reset and initialization, the TWIM is in Standard-mode, Fast-mode, or Fast-mode Plus (collectively referred to as the F/S-mode). For the TWIM to enter High-speed-mode (HS-mode), the user must write a one to the HS-mode (HS) bit and write a unique 3-bit code to the HS-mode Master Code field (HSMCODE) in the Command Register (CMDR) or/and the Next Command Register (NCMDR). This instructs the TWIM to initiate a HS-mode transfer, when the bus is free, by transmitting (in F/S-mode) the START condition followed by a unique 8-bit HS-mode master code (0x00001XXX), which is formed by prefixing CMDR.HSMCODE with 0b00001. This is then followed by the not-acknowledge bit (NA) on the bus, after which HS-mode transfer commences. In summary, the conditions for initiating HS-mode transfer are:

1. START condition (S)
2. 8-bit master code (0000 1XXX)
3. Not-acknowledge bit (NA)

HS-mode master codes are reserved 8-bit codes and serve two purposes:

1. It allows arbitration and synchronization between competing masters at F/S-mode speeds, resulting in one winning master.
2. It indicates the beginning of an HS-mode transfer.

If the TWIM remains as the active master after the transmission of the master code and the NA bit, it releases the TWCK line and waits for the line to be pulled up to HIGH, during which it does the following:

1. Adapt the TWD and TWCK input filters to the spike suppression requirement in HS-mode.
2. Adapt the TWD and TWCK output stages to the slope control requirements in HS-mode.

Once the TWCK line is HIGH, the TWIM operates in HS-mode and only switches back to F/S-mode after a STOP condition. If an acknowledge bit (A) is erroneously placed on the bus after the transmission of the master code, the TWIM sets the HSMCACK bit in the Status Register (SR) and transmits the STOP condition on the bus.

With regard to the slope control of the TWD and TWCK outputs, the user can control the rise and fall times of the TWCK output in F/S- and HS-mode by writing the Clock Drive Strength HIGH/LOW (CLDRIVEH/L) and Clock Slew Limit (CLSLEW) fields of the Slew Rate Register (SRR) and HS-mode Slew Rate Register (HSSRR), respectively. Likewise, the fall times of the TWD output in F/S- and HS-mode can be controlled by writing the Data Drive Strength LOW (DADRIVEL) and Data Slew Limit (DASLEW) fields of SRR and HSSRR, respectively. Refer to Section 42. ”Electrical Characteristics” on page 1121 for appropriates values of these register fields.

Note that the fall times of the TWD output are also controlled by the corresponding register fields in the Two-wire Slave Interface (TWIS) module. In order to correctly control the slew rate of the TWD output, the user must either

1. Write the relevant register fields in the TWIM with appropriate values and leave those in TWIS as zeros, or vice versa; or
2. Write the relevant register fields in both the TWIM and the TWIS with the same values.

During HS-mode transfer, the TWIM enables and controls a current-source pull-up circuit at the output stage of its TWCK signal (if it is the active master) to shorten the rise time of the signal. The current-source pull-up circuit is temporarily disabled by the TWIM after a REPEATED START condition and after each acknowledge bit (A) and not-acknowledge bit (NA), thus enabling other devices connected to the bus to delay the serial transfer by stretching the LOW

period of the TWCK signal. The TWIM enables its current-source pull-up circuit again when all devices have released the TWCK line and the line reaches a HIGH level.

27.8.2.2 Clock Generation

When the TWIM is in F/S-mode, theClock Waveform Generator Register (CWGR) is used to control the waveform of the TWCK clock. CWGR must be written so that the desired TWI bus timings are generated. CWGR describes bus timings as a function of cycles of a prescaled clock. The clock prescaling can be selected through the Clock Prescaler field in CWGR (CWGR.EXP).

$$f_{PRESCALER} = \frac{f_{CLK\_TWIM}}{2^{(EXP+1)}}$$

CWGR has the following fields:

LOW: Prescaled clock cycles in clock low count. Used to time T_{LOW} and T_{BUF}.

HIGH: Prescaled clock cycles in clock high count. Used to time T_{HIGH}.

STASTO: Prescaled clock cycles in clock high count. Used to time T_{HD_STA}, T_{SU_STA}, T_{SU_STO}.

DATA: Prescaled clock cycles for data setup and hold count. Used to time T_{HD_DAT}, T_{SU_DAT}.

EXP: Specifies the clock prescaler setting.

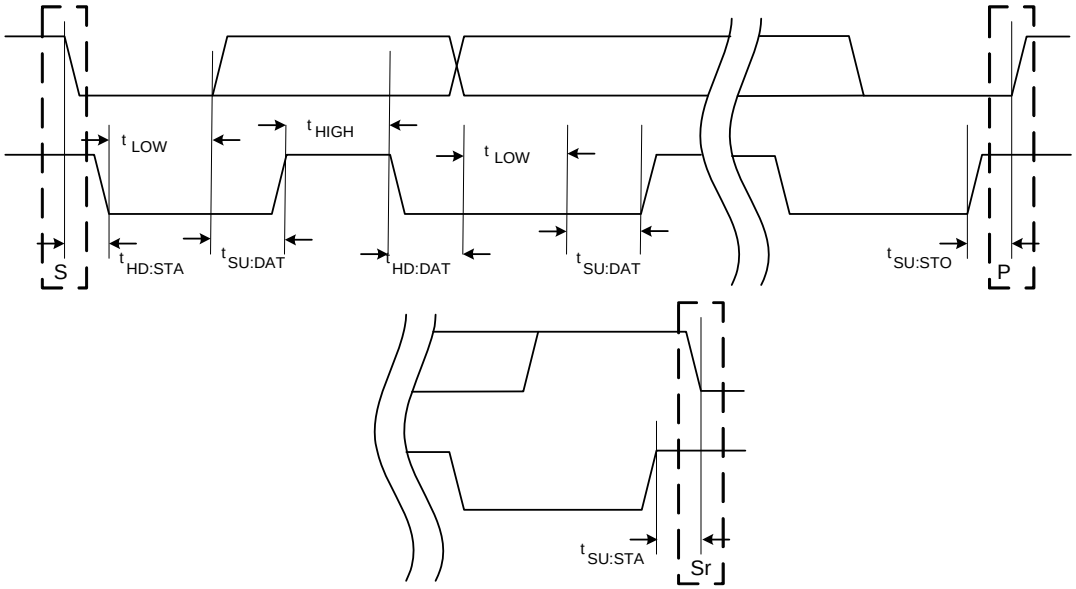
Note that the total clock low time generated is the sum of $T_{HD\_DAT} + T_{SU\_DAT} + T_{LOW}$.

Any slave or other bus master taking part in the transfer may extend the TWCK low period at any time.

The TWIM hardware monitors the state of the TWCK line as required by the I²C specification. The clock generation counters are started when a high/low level is detected on the TWCK line, not when the TWIM hardware releases/drives the TWCK line. This means that the CWGR settings alone do not determine the TWCK frequency. The CWGR settings determine the clock low time and the clock high time, but the TWCK rise and fall times are determined by the external circuitry (capacitive load, etc.).

When the TWIM is in HS-mode, the HS-mode Clock Waveform Generator Register (HSCWGR), instead of CWGR, is used to generate the TWCK signal. HSCWGR has the same fields as CWGR serving the same purposes.

Figure 27-5. Bus Timing Diagram



27.8.2.3 Setting up and Performing a Transfer

Operation of the TWIM is mainly controlled by the Control Register (CR) and the Command Register (CMDR). TWIM status is provided in the Status Register (SR). The following list presents the main steps in a typical communication:

1. Before any transfers can be performed, bus timings must be configured by writing to the Clock Waveform Generator Register (CWGR) and, if HS-mode is supported, the HS-mode Clock Waveform Generator Register (HSCWGR). If operating in SMBus mode, the SMBus Timing Register (SMBTR) register must also be configured.
2. If the Peripheral DMA Controller is to be used for the transfers, it must be set up.
3. CMDR or NCMDR must be written with a value describing the transfer to be performed.

The interrupt system can be set up to give interrupt requests on specific events or error conditions in the SR, for example when the transfer is complete or if arbitration is lost. The Interrupt Enable Register (IER) and Interrupt Disable Register (IDR) can be written to specify which bits in the SR will generate interrupt requests.

The SR.BUSFREE bit is set when activity is completed on the two-wire bus. The SR.CRDY bit is set when CMDR and/or NCMDR is ready to receive one or more commands.

The controller will refuse to start a new transfer while ANAK, DNAK,, ARBLST, or HSMCACK in the Status Register (SR) is one. This is necessary to avoid a race when the software issues a continuation of the current transfer at the same time as one of these errors happen. Also, if ANAK or DNAK occurs, a STOP condition is sent automatically. The user will have to restart the transmission by clearing the error bits in SR after resolving the cause for the NACK.

After a data or address NACK from the slave, a STOP will be transmitted automatically. Note that the VALID bit in CMDR is NOT cleared in this case. If this transfer is to be discarded, the VALID bit can be cleared manually allowing any command in NCMDR to be copied into CMDR.

When a data or address NACK is returned by the slave while the master is transmitting, it is possible that new data has already been written to the THR register. This data will be transferred out as the first data byte of the next transfer. If this behavior is to be avoided, the safest approach is to perform a software reset of the TWIM.

27.8.3 Master Transmitter Mode

A START condition is transmitted and master transmitter mode is initiated when the bus is free and CMDR has been written with START=1 and READ=0. START and SADR+W will then be transmitted. During the address acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to acknowledge the address. The master polls the data line during this clock pulse and sets the Address Not Acknowledged bit (ANAK**)** in the Status Register if no slave acknowledges the address.

After the address phase, the following is repeated:

while (NBYTES>0)

1. Wait until THR contains a valid data byte, stretching low period of TWCK. SR.TXRDY indicates the state of THR. Software or the Peripheral DMA Controller must write the data byte to THR.
2. Transmit this data byte
3. Decrement NBYTES
4. If (NBYTES==0) and STOP=1, transmit STOP condition

Writing CMDR with START=STOP=1 and NBYTES=0 will generate a transmission with no data bytes, ie START, SADR+W, STOP.

TWI transfers require the slave to acknowledge each received data byte. During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets the Data Acknowledge bit (DNACK) in the Status Register if the slave does not acknowledge the data byte. As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable Register (IER).

TXRDY is used as Transmit Ready for the Peripheral DMA Controller transmit channel.

The end of a command is marked when the TWIM sets the SR.CCOMP bit. See Figure 27-6 and Figure 27-7.

Figure 27-6. Master Write with One Data Byte

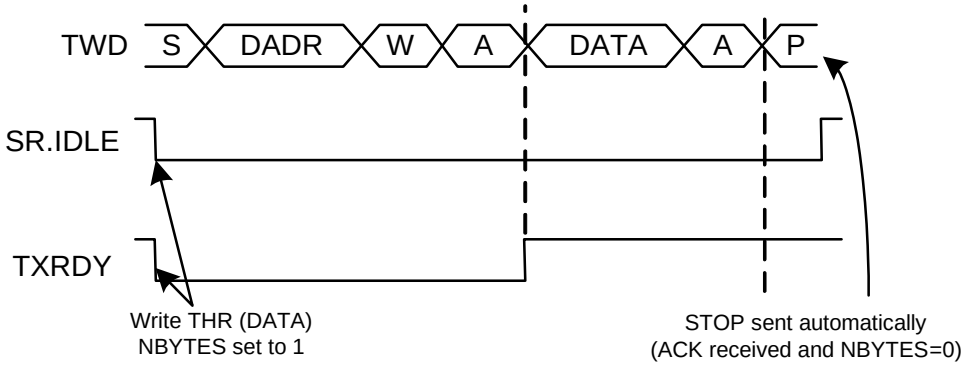
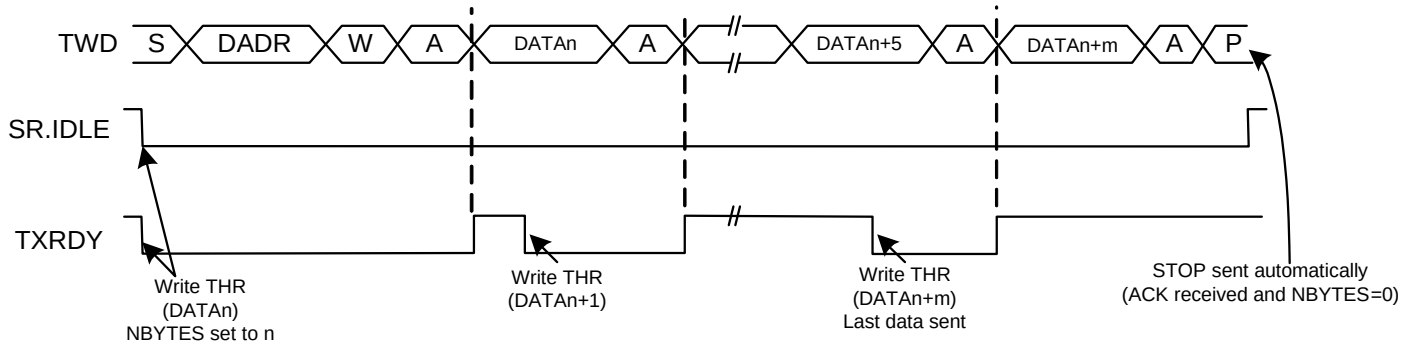


Figure 27-7. Master Write with Multiple Data Bytes



27.8.4 Master Receiver Mode

A START condition is transmitted and master receiver mode is initiated when the bus is free and CMDR has been written with START=1 and READ=1. START and SADR+R will then be transmitted. During the address acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to acknowledge the address. The master polls the data line during this clock pulse and sets the Address Not Acknowledged bit (ANAK) in the Status Register if no slave acknowledges the address.

After the address phase, the following is repeated:

while (NBYTES>0)

1. Wait until RHR is empty, stretching low period of TWCK. SR.RXRDY indicates the state of RHR. Software or the Peripheral DMA Controller must read any data byte present in RHR.
2. Release TWCK generating a clock that the slave uses to transmit a data byte.
3. Place the received data byte in RHR, set RXRDY.
4. If NBYTES=0, generate a NAK after the data byte, otherwise generate an ACK.
5. Decrement NBYTES
6. If (NBYTES==0) and STOP=1, transmit STOP condition.

Writing CMDR with START=STOP=1 and NBYTES=0 will generate a transmission with no data bytes, ie START, DADR+R, STOP

The TWI transfers require the master to acknowledge each received data byte. During the acknowledge clock pulse (9th pulse), the slave releases the data line (HIGH), enabling the master to pull it down in order to generate the acknowledge. All data bytes except the last are acknowledged by the master. Not acknowledging the last byte informs the slave that the transfer is finished.

RXRDY is used as Receive Ready for the Peripheral DMA Controller receive channel.

Figure 27-8. Master Read with One Data Byte

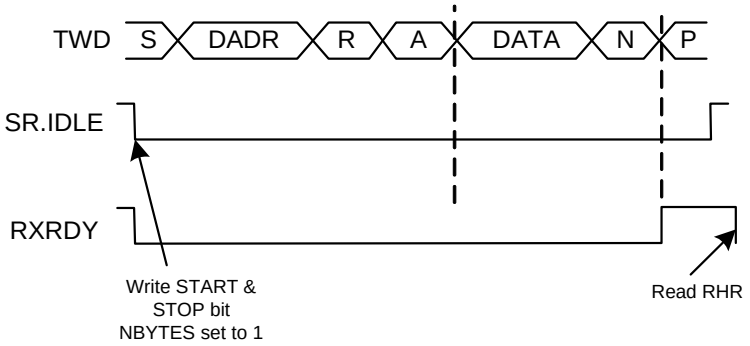
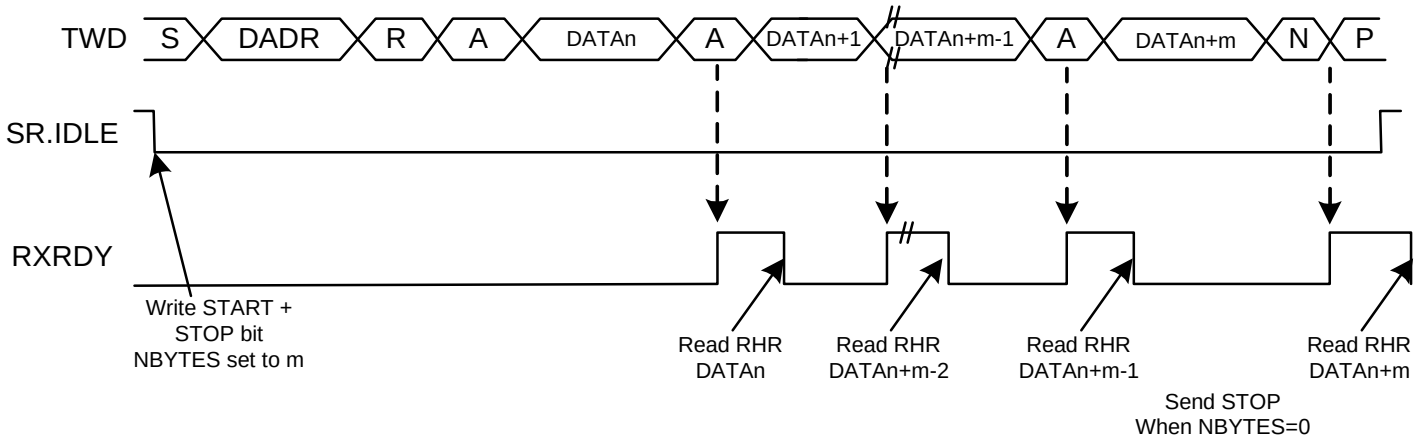


Figure 27-9. Master Read with Multiple Data Bytes



27.8.5 Using the Peripheral DMA Controller

The use of the Peripheral DMA Controller significantly reduces the CPU load. The user can set up ring buffers for the Peripheral DMA Controller, containing data to transmit or free buffer space to place received data.

To assure correct behavior, respect the following programming sequences:

27.8.5.1 Data Transmit with the Peripheral DMA Controller

1. Initialize the transmit Peripheral DMA Controller (memory pointers, size, etc.).
2. Configure the TWIM (ADR, NBYTES, etc.).
3. Start the transfer by enabling the Peripheral DMA Controller to transmit.
4. Wait for the Peripheral DMA Controller end-of-transmit flag.
5. Disable the Peripheral DMA Controller.

27.8.5.2 Data Receive with the Peripheral DMA Controller

1. Initialize the receive Peripheral DMA Controller (memory pointers, size, etc.).
2. Configure the TWIM (ADR, NBYTES, etc.).
3. Start the transfer by enabling the Peripheral DMA Controller to receive.
4. Wait for the Peripheral DMA Controller end-of-receive flag.
5. Disable the Peripheral DMA Controller.

27.8.6 Multi-master Mode

More than one master may access the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a STOP. The SR.ARBLST flag will be set. When the STOP is detected, the master who lost arbitration may reinitiate the data transfer.

Arbitration is illustrated in Figure 27-11.

If the user starts a transfer and if the bus is busy, the TWIM automatically waits for a STOP condition on the bus before initiating the transfer (see Figure 27-10).

Note: The state of the bus (busy or free) is not indicated in the user interface.

Figure 27-10. User Sends Data While the Bus is Busy

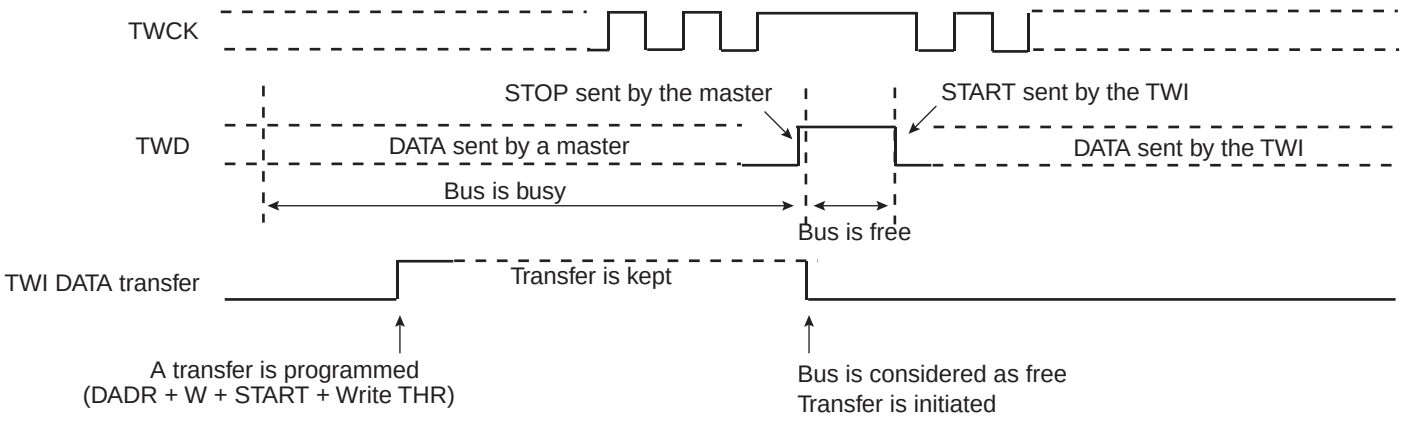
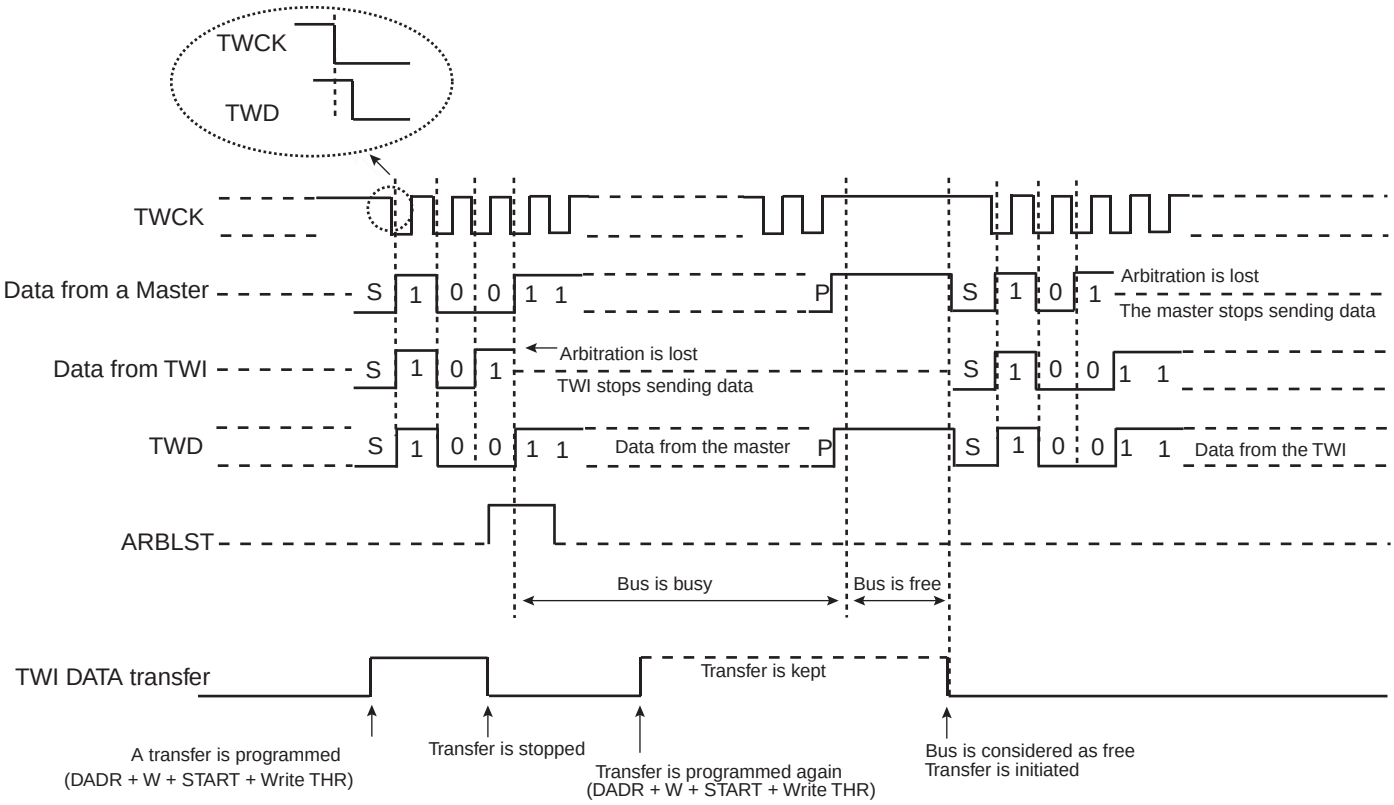


Figure 27-11. Arbitration Cases



27.8.7 Combined Transfers

CMDR and NCMDR may be used to generate longer sequences of connected transfers, since generation of START and/or STOP conditions is programmable on a per-command basis.

Writing NCMDR with START=1 when the previous transfer was written with STOP=0 will cause a REPEATED START on the bus. The ability to generate such connected transfers allows arbitrary transfer lengths, since it is legal to write CMDR with both START=0 and STOP=0. If this is done in master receiver mode, the CMDR.ACKLAST bit must also be controlled.

As for single data transfers, the TXRDY and RXRDY bits in the Status Register indicates when data to transmit can be written to THR, or when received data can be read from RHR. Transfer of data to THR and from RHR can also be done automatically by DMA, see Section 27.8.5

27.8.7.1 Write Followed by Write

Consider the following transfer:

START, DADR+W, DATA+A, DATA+A, REPSTART, DADR+W, DATA+A, DATA+A, STOP.

To generate this transfer:

1. Write CMDR with START=1, STOP=0, DADR, NBYTES=2 and READ=0.
2. Write NCMDR with START=1, STOP=1, DADR, NBYTES=2 and READ=0.
3. Wait until SR.TXRDY==1, then write first data byte to transfer to THR.
4. Wait until SR.TXRDY==1, then write second data byte to transfer to THR.
5. Wait until SR.TXRDY==1, then write third data byte to transfer to THR.

6. Wait until SR.TXRDY==1, then write fourth data byte to transfer to THR.

27.8.7.2 *Read Followed by Read*

Consider the following transfer:

START, DADR+R, DATA+A, DATA+NA, REPSTART, DADR+R, DATA+A, DATA+NA, STOP.

To generate this transfer:

1. Write CMDR with START=1, STOP=0, DADR, NBYTES=2 and READ=1.
2. Write NCMDR with START=1, STOP=1, DADR, NBYTES=2 and READ=1.
3. Wait until SR.RXRDY==1, then read first data byte received from RHR.
4. Wait until SR.RXRDY==1, then read second data byte received from RHR.
5. Wait until SR.RXRDY==1, then read third data byte received from RHR.
6. Wait until SR.RXRDY==1, then read fourth data byte received from RHR.

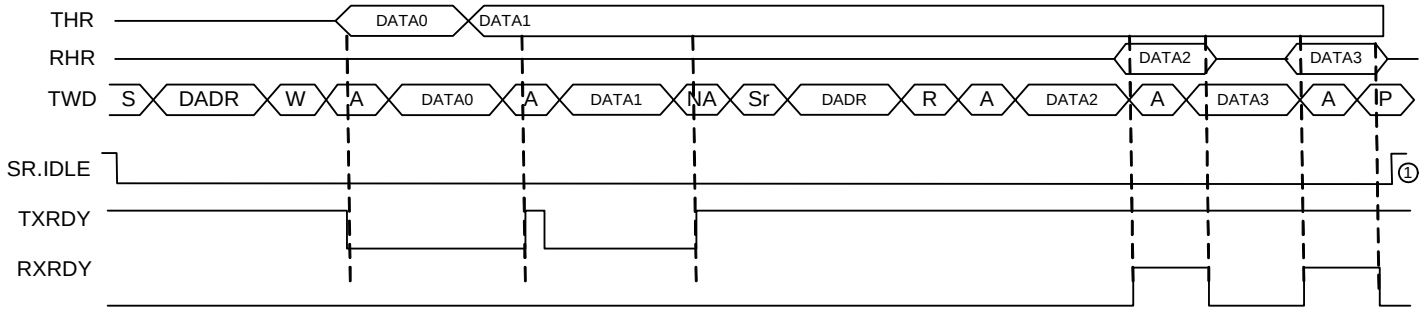
If combining several transfers, without any STOP or REPEATED START between them, remember to write a one to the ACKLAST bit in CMDR to keep from ending each of the partial transfers with a NACK.

27.8.7.3 *Write Followed by Read*

Consider the following transfer:

START, DADR+W, DATA+A, DATA+A, REPSTART, DADR+R, DATA+A, DATA+NA, STOP.

Figure 27-12. Combining a Write and Read Transfer



To generate this transfer:

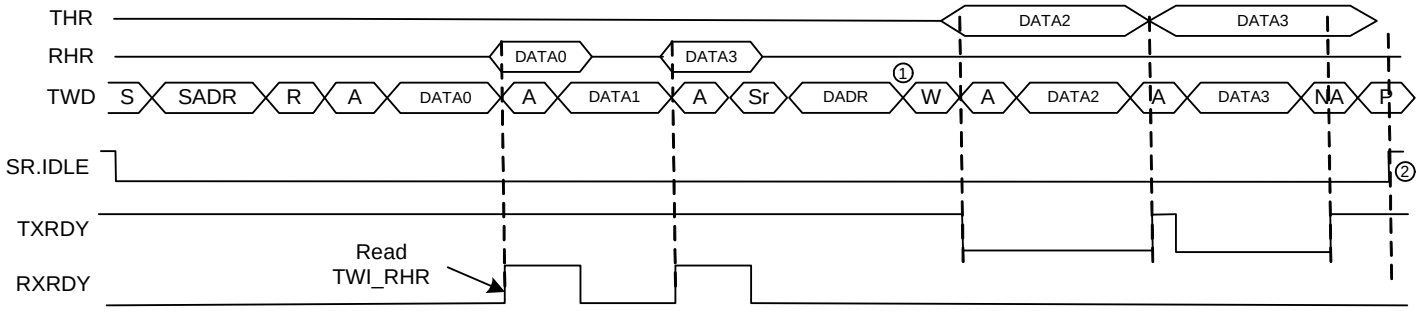
1. Write CMDR with START=1, STOP=0, DADR, NBYTES=2 and READ=0.
2. Write NCMDR with START=1, STOP=1, DADR, NBYTES=2 and READ=1.
3. Wait until SR.TXRDY==1, then write first data byte to transfer to THR.
4. Wait until SR.TXRDY==1, then write second data byte to transfer to THR.
5. Wait until SR.RXRDY==1, then read first data byte received from RHR.
6. Wait until SR.RXRDY==1, then read second data byte received from RHR.

27.8.7.4 *Read Followed by Write*

Consider the following transfer:

START, DADR+R, DATA+A, DATA+NA, REPSTART, DADR+W, DATA+A, DATA+A, STOP.

Figure 27-13. Combining a Read and Write Transfer



To generate this transfer:

1. Write CMDR with START=1, STOP=0, DADR, NBYTES=2 and READ=1.
2. Write NCMDR with START=1, STOP=1, DADR, NBYTES=2 and READ=0.
3. Wait until SR.RXRDY==1, then read first data byte received from RHR.
4. Wait until SR.RXRDY==1, then read second data byte received from RHR.
5. Wait until SR.TXRDY==1, then write first data byte to transfer to THR.
6. Wait until SR.TXRDY==1, then write second data byte to transfer to THR.

27.8.8 Ten Bit Addressing

Writing a one to CMDR.TENBIT enables 10-bit addressing in hardware. Performing transfers with 10-bit addressing is similar to transfers with 7-bit addresses, except that bits 9:7 of CMDR.SADR must be written appropriately.

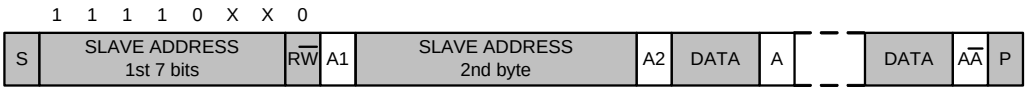
In Figure 27-14 and Figure 27-15, the grey boxes represent signals driven by the master, the white boxes are driven by the slave.

27.8.8.1 Master Transmitter

To perform a master transmitter transfer:

1. Write CMDR with TENBIT=1, REPSAME=0, READ=0, START=1, STOP=1 and the desired address and NBYTES value.

Figure 27-14. A Write Transfer with 10-bit Addressing



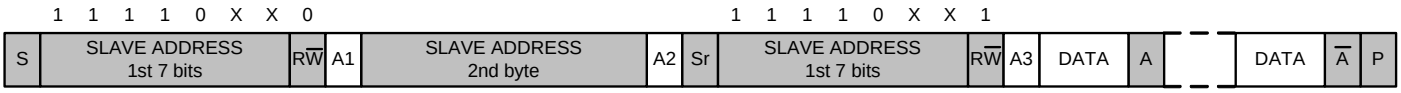
27.8.8.2 Master Receiver

When using master receiver mode with 10-bit addressing, CMDR.REPSAME must also be controlled. CMDR.REPSAME must be written to one when the address phase of the transfer should consist of only 1 address byte (the 11110xx byte) and not 2 address bytes. The I²C standard specifies that such addressing is required when addressing a slave for reads using 10-bit addressing.

To perform a master receiver transfer:

1. Write CMDR with TENBIT=1, REPSAME=0, READ=0, START=1, STOP=0, NBYTES=0 and the desired address.
2. Write NCMDR with TENBIT=1, REPSAME=1, READ=1, START=1, STOP=1 and the desired address and NBYTES value.

Figure 27-15. A Read Transfer with 10-bit Addressing



27.8.9 SMBus Mode

SMBus mode is enabled and disabled by writing to the SMEN and SMDIS bits in CR. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be written into SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses have been reserved for protocol handling, such as Alert Response Address (ARA) and Host Header (HH) Address.

27.8.9.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to CMDR.PECEN enables automatic PEC handling in the current transfer. Transfers with and without PEC can freely be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers will be correct.

In master transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. The DNAK bit in SR reflects the state of the last received ACK/NACK value. Some slaves may not be able to check the received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and some other mechanism must be implemented to verify that the transmission was received correctly.

In master receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and SR.PECERR is set. In master receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

The PEC byte is automatically inserted in a master transmitter transmission if PEC is enabled when NBYTES reaches zero. The PEC byte is identified in a master receiver transmission if PEC is enabled when NBYTES reaches zero. NBYTES must therefore be written with the total number of data bytes in the transmission, including the PEC byte.

In combined transfers, the PECEN bit should only be written to one in the last of the combined transfers. Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

This transfer is generated by writing two commands to the command registers. The first command is a write with NBYTES=1 and PECEN=0, and the second is a read with NBYTES=2 and PECEN=1.

Writing a one to the STOP bit in CR will place a STOP condition on the bus after the current byte. No PEC byte will be sent in this case.

27.8.9.2 Timeouts

The TLOWS and TLOWM fields in SMBTR configure the SMBus timeout values. If a timeout occurs, the master will transmit a STOP condition and leave the bus. The SR.TOUT bit is set.

27.8.10 Identifying Bus Events

This chapter lists the different bus events, and how they affect bits in the TWIM registers. This is intended to help writing drivers for the TWIM.

Table 27-5. Bus Events

Event	Effect
Master transmitter has sent a data byte	SR.THR is cleared.
Master receiver has received a data byte	SR.RHR is set.
Start+Sadr sent, no ack received from slave	SR.ANAK is set. SR.CCOMP not set. CMDR.VALID remains set. STOP automatically transmitted on bus.
Data byte sent to slave, no ack received from slave	SR.DNAK is set. SR.CCOMP not set. CMDR.VALID remains set. STOP automatically transmitted on bus.
Arbitration lost	SR.ARBLST is set. SR.CCOMP not set. CMDR.VALID remains set. TWCK and TWD immediately released to a pulled-up state.
SMBus timeout received	SR.SMBTOUT is set. SR.CCOMP not set. CMDR.VALID remains set. STOP automatically transmitted on bus.

Table 27-5. Bus Events

Event	Effect
Master transmitter receives SMBus PEC Error	SR.DNAK is set. SR.CCOMP not set. CMDR.VALID remains set. STOP automatically transmitted on bus.
Master receiver discovers SMBus PEC Error	SR.PECERR is set. SR.CCOMP not set. CMDR.VALID remains set. STOP automatically transmitted on bus.
CR.STOP is written by user	SR.STOP is set. SR.CCOMP set. CMDR.VALID remains set. STOP transmitted on bus after current byte transfer has finished.

27.9 User Interface

Table 27-6. TWIM Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-only	0x00000000
0x04	Clock Waveform Generator Register	CWGR	Read/Write	0x00000000
0x08	SMBus Timing Register	SMBTR	Read/Write	0x00000000
0x0C	Command Register	CMDR	Read/Write	0x00000000
0x10	Next Command Register	NCMDR	Read/Write	0x00000000
0x14	Receive Holding Register	RHR	Read-only	0x00000000
0x18	Transmit Holding Register	THR	Write-only	0x00000000
0x1C	Status Register	SR	Read-only	0x00000002
0x20	Interrupt Enable Register	IER	Write-only	0x00000000
0x24	Interrupt Disable Register	IDR	Write-only	0x00000000
0x28	Interrupt Mask Register	IMR	Read-only	0x00000000
0x2C	Status Clear Register	SCR	Write-only	0x00000000
0x30	Parameter Register	PR	Read-only	-(1)
0x34	Version Register	VR	Read-only	-(1)
0x38	HS-mode Clock Waveform Generator	HSCWGR	Read/Write	0x00000000
0x3C	Slew Rate Register	SRR	Read/Write	0x00000000
0x40	HS-mode Slew Rate Register	HSSRR	Read/Write	0x00000000

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

27.9.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	STOP

7	6	5	4	3	2	1	0
SWRST	-	SMDIS	SMEN	-	-	MDIS	MEN

- **STOP: Stop the Current Transfer**
  Writing a one to this bit terminates the current transfer, sending a STOP condition after the shifter has become idle. If there are additional pending transfers, they will have to be explicitly restarted by software after the STOP condition has been successfully sent.
  Writing a zero to this bit has no effect.
- **SWRST: Software Reset**
  If the TWIM master interface is enabled, writing a one to this bit resets the TWIM. All transfers are halted immediately, possibly violating the bus semantics.
  If the TWIM master interface is not enabled, it must first be enabled before writing a one to this bit.
  Writing a zero to this bit has no effect.
- **SMDIS: SMBus Disable**
  Writing a one to this bit disables SMBus mode.
  Writing a zero to this bit has no effect.
- **SMEN: SMBus Enable**
  Writing a one to this bit enables SMBus mode.
  Writing a zero to this bit has no effect.
- **MDIS: Master Disable**
  Writing a one to this bit disables the master interface.
  Writing a zero to this bit has no effect.
- **MEN: Master Enable**
  Writing a one to this bit enables the master interface.
  Writing a zero to this bit has no effect.

27.9.2 Clock Waveform Generator Register

Name: CWGR

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	EXP			DATA			

23	22	21	20	19	18	17	16
STASTO							

15	14	13	12	11	10	9	8
HIGH							

7	6	5	4	3	2	1	0
LOW							

- **EXP: Clock Prescaler**
  Used to specify how to prescale the TWCK clock. Counters are prescaled according to the following formula
  $f_{PRESCALER} = \frac{f_{CLK\_TWIM}}{2^{(EXP+1)}}$

- **DATA: Data Setup and Hold Cycles**
  Clock cycles for data setup and hold count. Prescaled by CWGR.EXP. Used to time T_{HD_DAT}, T_{SU_DAT}.
- **STASTO: START and STOP Cycles**
  Clock cycles in clock high count. Prescaled by CWGR.EXP. Used to time T_{HD_STA}, T_{SU_STA}, T_{SU_STO}
- **HIGH: Clock High Cycles**
  Clock cycles in clock high count. Prescaled by CWGR.EXP. Used to time T_{HIGH}.
- **LOW: Clock Low Cycles**
  Clock cycles in clock low count. Prescaled by CWGR.EXP. Used to time T_{LOW}, T_{BUF}.

27.9.3 SMBus Timing Register

Name: SMBTR

Access Type: Read/Write

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
EXP				-	-	-	-

23	22	21	20	19	18	17	16
THMAX							

15	14	13	12	11	10	9	8
TLOWM							

7	6	5	4	3	2	1	0
TLOWS							

- **EXP: SMBus Timeout Clock Prescaler**
  Used to specify how to prescale the TIM and TLOWM counters in SMBTR. Counters are prescaled according to the following formula

$$f_{prescaled,\,SMBus} = \frac{f_{CLKTWIM}}{2^{(EXP+1)}}$$

- **THMAX: Clock High Maximum Cycles**
  Clock cycles in clock high maximum count. Prescaled by SMBTR.EXP. Used for bus free detection. Used to time $T_{HIGH:MAX}$.
  NOTE: Uses the prescaler specified by CWGR, NOT the prescaler specified by SMBTR.
- **TLOWM: Master Clock Stretch Maximum Cycles**
  Clock cycles in master maximum clock stretch count. Prescaled by SMBTR.EXP. Used to time $T_{LOW:MEXT}$
- **TLOWS: Slave Clock Stretch Maximum Cycles**
  Clock cycles in slave maximum clock stretch count. Prescaled by SMBTR.EXP. Used to time $T_{LOW:SEXT}$.

27.9.4 Command Register

Name: CMDR

Access Type: Read/Write

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	HSMCODE			-	HS	ACKLAST	PECEN

23	22	21	20	19	18	17	16
NBYTES							

15	14	13	12	11	10	9	8
VALID	STOP	START	REPSAME	TENBIT	SADR[9:7]		

7	6	5	4	3	2	1	0
SADR[6:0]							READ

- **HSMCODE: HS-mode Master Code**
  3-bit code to be prefixed with 0b00001 to form a unique 8-bit HS-mode master code (0000 1XXX).
- **HS: HS-mode**
  0: Causes the TWIM not to initiate HS-mode transfer.
  1: Causes the TWIM to initiate HS-mode transfer.
- **ACKLAST: ACK Last Master RX Byte**
  0: Causes the last byte in master receive mode (when NBYTES has reached 0) to be NACKed. This is the standard way of ending a master receiver transfer.
  1: Causes the last byte in master receive mode (when NBYTES has reached 0) to be ACKed. Used for performing linked transfers in master receiver mode with no STOP or REPEATED START between the subtransfers. This is needed when more than 255 bytes are to be received in one single transmission.
- **PECEN: Packet Error Checking Enable**
  0: Causes the transfer not to use PEC byte verification. The PEC LFSR is still updated for every bit transmitted or received. Must be used if SMBus mode is disabled.
  1: Causes the transfer to use PEC. PEC byte generation (if master transmitter) or PEC byte verification (if master receiver) will be performed.
- **NBYTES: Number of Data Bytes in Transfer**
  The number of data bytes in the transfer. After the specified number of bytes have been transferred, a STOP condition is transmitted if CMDR.STOP is one. In SMBus mode, if PEC is used, NBYTES includes the PEC byte, i.e. there are NBYTES-1 data bytes and a PEC byte.
- **VALID: CMDR Valid**
  0: Indicates that CMDR does not contain a valid command.
  1: Indicates that CMDR contains a valid command. This bit is cleared when the command is finished.
- **STOP: Send STOP Condition**
  0: Do not transmit a STOP condition after the data bytes have been transmitted.
  1: Transmit a STOP condition after the data bytes have been transmitted.
- **START: Send START Condition**
  0: The transfer in CMDR should not commence with a START or REPEATED START condition.

1: The transfer in CMDR should commence with a START or REPEATED START condition. If the bus is free when the command is executed, a START condition is used. If the bus is busy, a REPEATED START is used.

- **REPSAME: Transfer is to Same Address as Previous Address**
  Only used in 10-bit addressing mode, always write to 0 in 7-bit addressing mode.
  Write this bit to one if the command in CMDR performs a repeated start to the same slave address as addressed in the previous transfer in order to enter master receiver mode.
  Write this bit to zero otherwise.
- **TENBIT: Ten Bit Addressing Mode**
  0: Use 7-bit addressing mode.
  1: Use 10-bit addressing mode. Must not be used when the TWIM is in SMBus mode.
- **SADR: Slave Address**
  Address of the slave involved in the transfer. Bits 9-7 are don’t care if 7-bit addressing is used.
- **READ: Transfer Direction**
  0: Allow the master to transmit data.
  1: Allow the master to receive data.

27.9.5 Next Command Register

Name: NCMDR

Access Type: Read/Write

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	HSMCODE			-	HS	ACKLAST	PECEN

23	22	21	20	19	18	17	16
NBYTES							

15	14	13	12	11	10	9	8
VALID	STOP	START	REPSAME	TENBIT	SADR[9:7]		

7	6	5	4	3	2	1	0
SADR[6:0]							READ

This register is identical to CMDR. When the VALID bit in CMDR becomes 0, the content of NCMDR is copied into CMDR, clearing the VALID bit in NCMDR. If the VALID bit in CMDR is cleared when NCMDR is written, the content is copied immediately.

27.9.6 Receive Holding Register

Name: RHR

Access Type: Read-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
RXDATA							

- **RXDATA: Received Data**
  When the RXRDY bit in the Status Register (SR) is one, this field contains a byte received from the TWI bus.

27.9.7 Transmit Holding Register

Name: THR

Access Type: Write-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
TXDATA							

- **TXDATA: Data to Transmit**
  Write data to be transferred on the TWI bus here.

27.9.8 Status Register

Name: SR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000002

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	HSMCACK	MENB

15	14	13	12	11	10	9	8
-	STOP	PECERR	TOUT	-	ARBLST	DNAK	ANAK

7	6	5	4	3	2	1	0
-	-	BUSFREE	IDLE	CCOMP	CRDY	TXRDY	RXRDY

- **HSMCACK: ACK in HS-mode Master Code Phase Received**
  This bit is one when an ACK is erroneously received during a HS-mode master code phase.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **MENB: Master Interface Enable**
  0: Master interface is disabled.
  1: Master interface is enabled.
- **STOP: Stop Request Accepted**
  This bit is one when a STOP request caused by writing a one to CR.STOP has been accepted, and transfer has stopped.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **PECERR: PEC Error**
  This bit is one when a SMBus PEC error occurred.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **TOUT: Timeout**
  This bit is one when a SMBus timeout occurred.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **ARBLST: Arbitration Lost**
  This bit is one when the actual state of the SDA line did not correspond to the data driven onto it, indicating a higher-priority transmission in progress by a different master.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **DNAK: NAK in Data Phase Received**
  This bit is one when no ACK was received form slave during data transmission.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **ANAK: NAK in Address Phase Received**
  This bit is one when no ACK was received from slave during address phase
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **BUSFREE: Two-wire Bus is Free**
  This bit is one when activity has completed on the two-wire bus.
  Otherwise, this bit is cleared.

- **IDLE: Master Interface is Idle**
  This bit is one when no command is in progress, and no command waiting to be issued.
  Otherwise, this bit is cleared.
- **CCOMP: Command Complete**
  This bit is one when the current command has completed successfully.
  This bit is zero if the command failed due to conditions such as a NAK receved from slave.
  This bit is cleared by writing 1 to the corresponding bit in the Status Clear Register (SCR).
- **CRDY: Ready for More Commands**
  This bit is one when CMDR and/or NCMDR is ready to receive one or more commands.
  This bit is cleared when this is no longer true.
- **TXRDY: THR Data Ready**
  This bit is one when THR is ready for one or more data bytes.
  This bit is cleared when this is no longer true (i.e. THR is full or transmission has stopped).
- **RXRDY: RHR Data Ready**
  This bit is one when RX data are ready to be read from RHR.
  This bit is cleared when this is no longer true.

27.9.9 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	HSMCACK	-

15	14	13	12	11	10	9	8
-	STOP	PECERR	TOUT	-	ARBLST	DNAK	ANAK

7	6	5	4	3	2	1	0
-	-	BUSFREE	IDLE	CCOMP	CRDY	TXRDY	RXRDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR

27.9.10 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	HSMCACK	-

15	14	13	12	11	10	9	8
-	STOP	PECERR	TOUT	-	ARBLST	DNAK	ANAK

7	6	5	4	3	2	1	0
-	-	BUSFREE	IDLE	CCOMP	CRDY	TXRDY	RXRDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR

27.9.11 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x28

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	HSMCACK	-

15	14	13	12	11	10	9	8
-	STOP	PECERR	TOUT	-	ARBLST	DNAK	ANAK

7	6	5	4	3	2	1	0
-	-	BUSFREE	IDLE	CCOMP	CRDY	TXRDY	RXRDY

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
This bit is cleared when the corresponding bit in IDR is written to one.
This bit is set when the corresponding bit in IER is written to one.

27.9.12 Status Clear Register

Name: SCR

Access Type : Write-only

Offset: 0x2C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	HSMCACK	-

15	14	13	12	11	10	9	8
-	STOP	PECERR	TOUT	-	ARBLST	DNAK	ANAK

7	6	5	4	3	2	1	0
-	-	-	-	CCOMP	-	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

27.9.13 Parameter Register

Name: PR

Access Type: Read-only

Offset: 0x30

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	HS

- **HS: HS-mode**
  0: High-speed-mode is not supported.
  1: High-speed-mode is supported.

27.9.14 Version Register

Name: VR

Access Type: Read-only

Offset: 0x34

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION [11:8]			

7	6	5	4	3	2	1	0
VERSION [7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

27.9.15 HS-mode Clock Waveform Generator Register

Name: HSCWGR

Access Type: Read/Write

Offset: 0x38

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	EXP			DATA			

23	22	21	20	19	18	17	16
STASTO							

15	14	13	12	11	10	9	8
HIGH							

7	6	5	4	3	2	1	0
LOW							

This register is identical to CWGR. It is used to generate the TWCK signal during HS-mode transfer.

27.9.16 Slew Rate Register

Name: SRR

Access Type: Read/Write

Offset: 0x3C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	FILTER		-	-	CLSLEW	

23	22	21	20	19	18	17	16
-	-	-	-	-	CLDRIVEL		

15	14	13	12	11	10	9	8
-	-	-	-	-	-	DASLEW	

7	6	5	4	3	2	1	0
-	-	-	-	-	DADRIVEL		

- **FILTER: Input Spike Filter Control**

FILTER Value	Function
0	Reserved
1	Reserved
2	Standard- or Fast-mode
3	Fast-mode plus

- **CLSLEW: Clock Slew Limit**
  Selects the slew limit of the TWCK output buffer in F/S-mode.
- **CLDRIVEL: Clock Drive Strength LOW**
  Selects the pull-down drive strength of the TWCK output buffer in F/S-mode.
- **DASLEW: Data Slew Limit**
  Selects the slew limit of the TWD output buffer in F/S-mode.
- **DADRIVEL: Data Drive Strength LOW**
  Selects the pull-down drive strength of the TWD output buffer in F/S-mode.

Refer to Section 42. ”Electrical Characteristics” on page 1121 **for appropriates values of these register fields.**

27.9.17 HS-mode Slew Rate Register

Name: HSSRR

Access Type: Read/Write

Offset: 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	FILTER		-	-	CLSLEW	

23	22	21	20	19	18	17	16
-	-	CLDRIVEH		-	CLDRIVEL		

15	14	13	12	11	10	9	8
-	-	-	-	-	-	DASLEW	

7	6	5	4	3	2	1	0
-					DADRIVEL		

- **FILTER: Input Spike Filter Control**
  0: Reserved
  1: HS-mode
  2: Reserved
  3: Reserved
- **CLSLEW: Clock Slew Limit**
  Selects the slew limit of the TWCK output buffer in HS-mode.
- **CLDRIVEH: Clock Drive Strength HIGH**
  Selects the pull-up drive strength of the TWCK output buffer In HS-mode.
- **CLDRIVEL: Clock Drive Strength LOW**
  Selects the pull-down drive strength of the TWCK output buffer in HS-mode.
- **DASLEW: Data Slew Limit**
  Selects the slew limit of the TWD output buffer in HS-mode.
- **DADRIVEL: Data Drive Strength LOW**
  Selects the pull-down drive strength of the TWD output buffer in HS-mode.

Refer to Section 42. ”Electrical Characteristics” on page 1121 **for appropriates values of these register fields.**

27.10 Module Configuration

The specific configuration for each TWIM instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 27-7. Module Clock Name

Module Name	Clock Name	Description
TWIM0	CLK_TWIM0	Clock for the TWIM0 bus interface
TWIM1	CLK_TWIM1	Clock for the TWIM1 bus interface
TWIM2	CLK_TWIM2	Clock for the TWIM2 bus interface
TWIM3	CLK_TWIM3	Clock for the TWIM3 bus interface

Note : TWI2 and TWI3 are master only. TWI0 and TWI1 are master and slave

Table 27-8. Register Reset Values

Register	Reset Value
VERSION	0x00000120
PARAMETER	0x00000001

28. Two-wire Slave Interface (TWIS)

Rev: 1.4.0.1

28.1 Features

- **Compatible with I²C standard**
  - **Transfer speeds up to 3.4 Mbit/s**
  - **7 and 10-bit and General Call addressing**
- **Compatible with SMBus standard**
  - **Hardware Packet Error Checking (CRC) generation and verification with ACK response**
  - **25 ms clock low timeout delay**
  - **25 ms slave cumulative clock low extend time**
- **Compatible with PMBus**
- **DMA interface for reducing CPU load**
- **Arbitrary transfer lengths, including 0 data bytes**
- **Optional clock stretching if transmit or receive buffers not ready for data transfer**
- **32-bit Peripheral Bus interface for configuration of the interface**

28.2 Overview

The Atmel Two-wire Slave Interface (TWIS) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 3.4 Mbit/s, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus, I²C, or SMBus-compatible master. The TWIS is always a bus slave and can transfer sequential or single bytes.

Below, Table 28-1 lists the compatibility level of the Atmel Two-wire Slave Interface and a full I²C compatible device.

Table 28-1. Atmel TWIS Compatibility with I²C Standard

I²C Standard	Atmel TWIS
Standard-mode (100 kbit/s)	Supported
Fast-mode (400 kbit/s)	Supported
High-speed-mode (3.4 Mbit/s)	Supported
7 or 10 bits Slave Addressing	Supported
START BYTE(1)	Not Supported
Repeated Start (Sr) Condition	Supported
ACK and NAK Management	Supported
Slope control and input filtering (Fast mode)	Supported
Clock stretching	Supported

Note: 1. START + b000000001 + Ack + Sr

Below, Table 28-2 lists the compatibility level of the Atmel Two-wire Slave Interface and a full SMBus compatible device.

Table 28-2. Atmel TWIS Compatibility with SMBus Standard

SMBus Standard	Atmel TWIS
Bus Timeouts	Supported
Address Resolution Protocol	Supported
Packet Error Checking	Supported

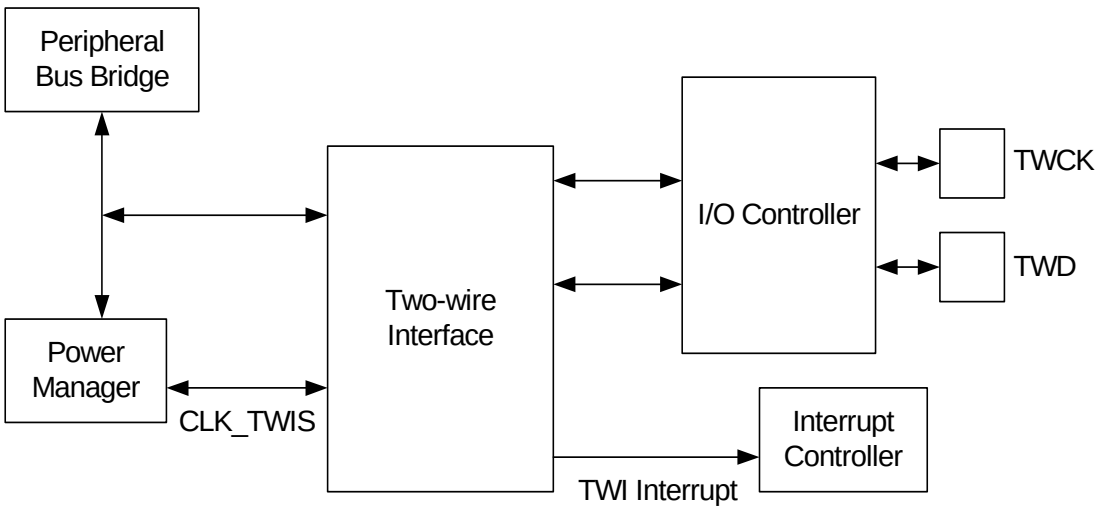
28.3 List of Abbreviations

Table 28-3. Abbreviations

Abbreviation	Description
TWI	Two-wire Interface
A	Acknowledge
NA	Non Acknowledge
P	Stop
S	Start
Sr	Repeated Start
SADR	Slave Address
ADR	Any address except SADR
R	Read
W	Write

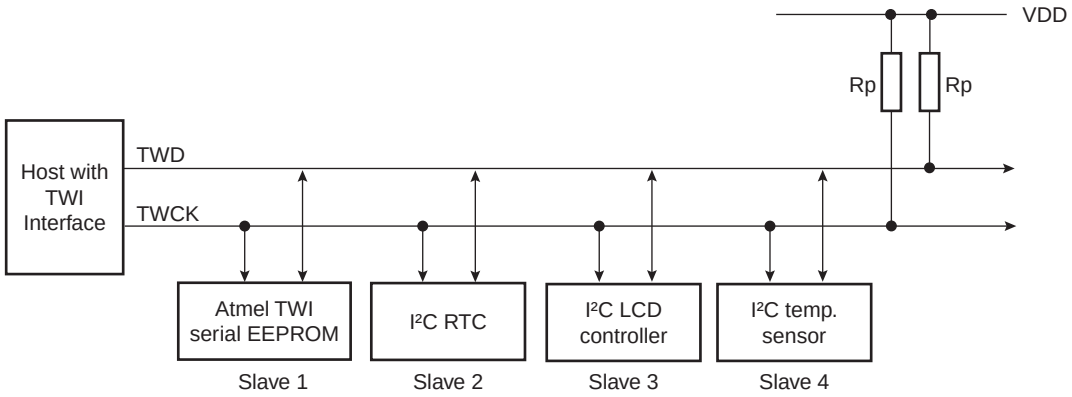
28.4 Block Diagram

Figure 28-1. Block Diagram



28.5 Application Block Diagram

Figure 28-2. Application Block Diagram



Rp: Pull up value as given by the I²C Standard

28.6 I/O Lines Description

Table 28-4. I/O Lines Description

Pin Name	Pin Description	Type
TWD	Two-wire Serial Data	Input/Output
TWCK	Two-wire Serial Clock	Input/Output

28.7 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

28.7.1 I/O Lines

TWD and TWCK are bidirectional lines, connected to a positive supply voltage via a current source or pull-up resistor (see Figure 28-5 on page 748). When the bus is free, both lines are high. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

TWD and TWCK pins may be multiplexed with I/O Controller lines. To enable the TWIS, the user must perform the following steps:

- Program the I/O Controller to:
  - Dedicate TWD, TWCKas peripheral lines.
  - Define TWD, TWCKas open-drain.

28.7.2 Power Management

If the CPU enters a sleep mode that disables clocks used by the TWIS, the TWIS will stop functioning and resume operation after the system wakes up from sleep mode. The TWIS is able to wake the system from sleep mode upon address match, see Section 28.8.9 on page 755.

28.7.3 Clocks

The clock for the TWIS bus interface (CLK_TWIS) is generated by the Power Manager. It is recommended to disable the TWIS before disabling the clock, to avoid freezing the TWIS in an undefined state.

28.7.4 DMA

The TWIS DMA handshake interface is connected to the Peripheral DMA Controller. Using the TWIS DMA functionality requires the Peripheral DMA Controller to be programmed after setting up the TWIS.

28.7.5 Interrupts

The TWIS interrupt request lines are connected to the NVIC. Using the TWIS interrupts requires the NVIC to be programmed first.

28.7.6 Debug Operation

When an external debugger forces the CPU into debug mode, the TWIS continues normal operation. If the TWIS is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

28.8 Functional Description

28.8.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (see Figure 28-4 on page 747).

Each transfer begins with a START condition and terminates with a STOP condition (see Figure 28-3).

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines a STOP condition.

Figure 28-3. START and STOP Conditions

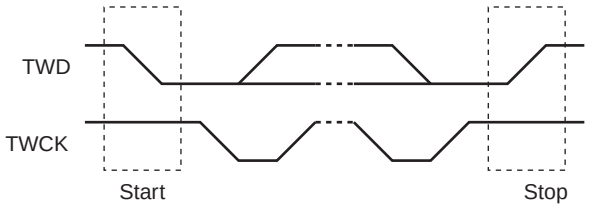
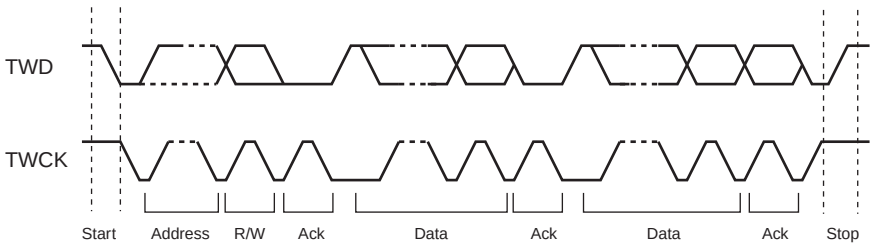


Figure 28-4. Transfer Format



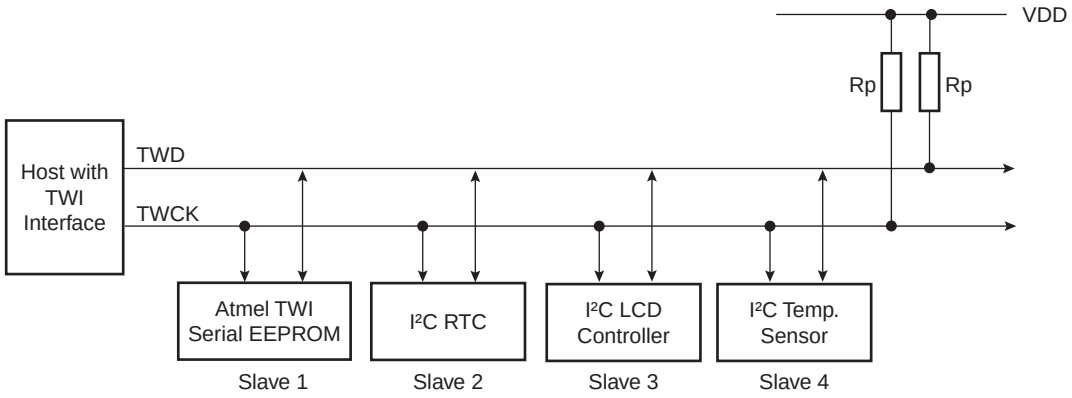
28.8.2 Operation

The TWIS has two modes of operation:

- Slave transmitter mode
- Slave receiver mode

A master is a device which starts and stops a transfer and generates the TWCK clock. A slave is assigned an address and responds to requests from the master. These modes are described in the following chapters.

Figure 28-5. Typical Application Block Diagram



Rp: Pull up value as given by the I²C Standard

28.8.3 High-speed-mode

After reset and initialization, the TWIS is either in Standard-mode, Fast-mode, or Fast-mode Plus (collectively referred to as the F/S-mode). The TWIS automatically enters High-speed-mode (HS-mode) after it detects the following conditions (all of which are in F/S-mode) on the bus:

1. START condition (S)
2. 8-bit master code (0000 1XXX)
3. Not-acknowledge bit (NA)

After the TWIS has detected the above conditions and before the commencement of HS-mode transfer, it does the following:

1. Adapts the TWD and TWCK input filters to the spike suppression requirement in HS-mode.
2. Adapts the TWD output stage to the slope control requirement in HS-mode.

The TWIS automatically returns to F/S-mode after it detects a STOP condition on the bus.

With regard to the slope control of the TWD output, the user can control the fall times of the TWD output in F/S- and HS-mode by writing to the Data Drive Strength LOW (DADRIVEL) and Data Slew Limit (DASLEW) fields in the Slew Rate Register (SRR) and HS-mode Slew Rate Register (HSSRR), respectively. Refer to Section 42. ”Electrical Characteristics” on page 1121 for appropriate values of these register fields.

Note that the fall times of the TWD output are also controlled by the corresponding register fields in the Two-wire Master Interface (TWIM) module. In order to correctly control the slew rate of the TWD output, the user must either

- Write to the relevant register fields in the TWIS with appropriate values and leave those in TWIM as zeros, or vice versa; or
- Write to the relevant register fields in both the TWIM and the TWIS with the same values.

28.8.3.1 Bus Timing

The Timing Register (TR) is used to control the timing of bus signals driven by the TWIS. TR describes bus timings as a function of cycles of the prescaled CLK_TWIS. The clock prescaling can be selected through TR.EXP.

$$f_{PRESCALED} = \frac{f_{CLK\_TWIS}}{2^{(EXP+1)}}$$

TR has the following fields:

TLOWS: Prescaled clock cycles used to time SMBUS timeout $T_{LOW:SEXT}$.

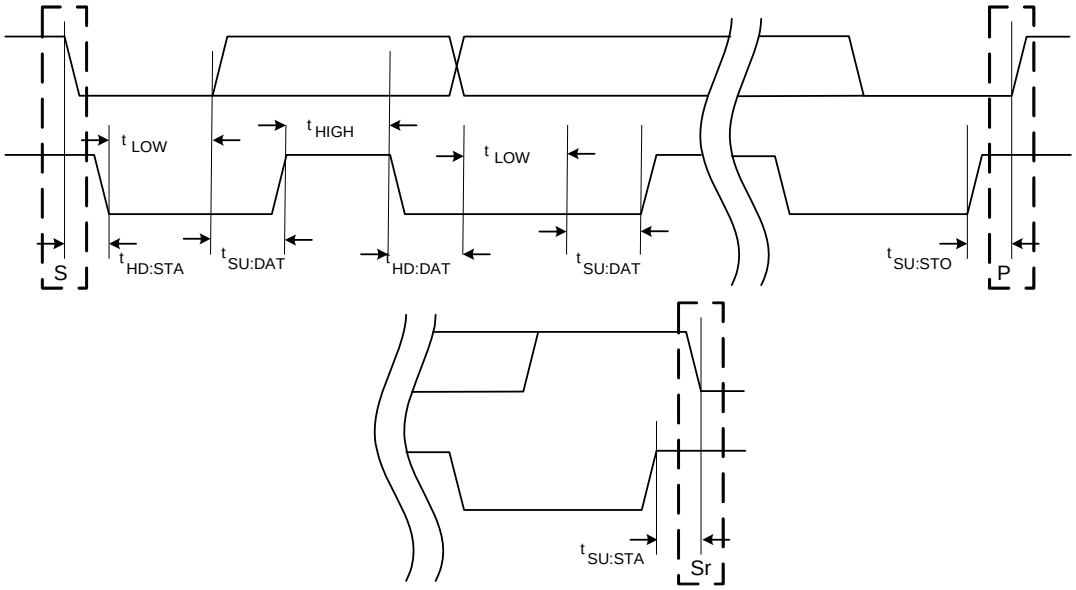
TTOUT: Prescaled clock cycles used to time SMBUS timeout $T_{TIMEOUT}$.

SUDAT: Non-prescaled clock cycles for data setup and hold count. Used to time T_{SU_DAT}.

EXP: Specifies the clock prescaler setting used for the SMBUS timeouts.

When the TWIS is in HS-mode, the data hold count is set by writing to the HDDAT field in the HS-mode Timing Register (HSTR).

Figure 28-6. Bus Timing Diagram



28.8.3.2 Setting Up and Performing a Transfer

Operation of the TWIS is mainly controlled by the Control Register (CR). The following list presents the main steps in a typical communication:

1. Before any transfers can be performed, bus timings must be configured by writing to the Timing Register (TR) and, if HS-mode transfer is supported, the HS-mode Timing Register (HSTR).
2. If the Peripheral DMA Controller is to be used for the transfers, it must be set up.
3. The Control Register (CR) must be configured with information such as the slave address, SMBus mode, Packet Error Checking (PEC), number of bytes to transfer, and which addresses to match.

The interrupt system can be set up to generate interrupt request on specific events or error conditions, for example when a byte has been received.

The NBYTES register is only used in SMBus mode, when PEC is enabled. In I²C mode or in SMBus mode when PEC is disabled, the NBYTES register is not used, and should be written to zero. NBYTES is updated by hardware, so in order to avoid hazards, software updates of NBYTES can only be done through writes to the NBYTES register.

28.8.3.3 Address Matching

The TWIS can be set up to match several different addresses. More than one address match may be enabled simultaneously, allowing the TWIS to be assigned to several addresses. The address matching phase is initiated after a START or REPEATED START condition. When the TWIS receives an address that generates an address match, an ACK is automatically returned to the master.

In I²C mode:

- The address in CR.ADR is checked for address match if CR.SMATCH is one.
- The General Call address is checked for address match if CR.GCMATCH is one.

In SMBus mode:

- The address in CR.ADR is checked for address match if CR.SMATCH is one.
- The Alert Response Address is checked for address match if CR.SMAL is one.
- The Default Address is checked for address match if CR.SMDA is one.
- The Host Header Address is checked for address match if CR.SMHH is one.

28.8.3.4 Clock Stretching

Any slave or bus master taking part in a transfer may extend the TWCK low period at any time. The TWIS may extend the TWCK low period after each byte transfer if CR.STREN is one and:

- Module is in slave transmitter mode, data should be transmitted, but THR is empty, or
- Module is in slave receiver mode, a byte has been received and placed into the internal shifter, but the Receive Holding Register (RHR) is full, or
- Stretch-on-address-match bit CR.SOAM=1 and slave was addressed. Bus clock remains stretched until all address match bits in the Status Register (SR) have been cleared.

If CR.STREN is zero and:

- Module is in slave transmitter mode, data should be transmitted but THR is empty: Transmit the value present in THR (the last transmitted byte or reset value), and set SR.URUN.

- Module is in slave receiver mode, a byte has been received and placed into the internal shifter, but RHR is full: Discard the received byte and set SR.ORUN.

28.8.3.5 Bus Errors

If a bus error (misplaced START or STOP) condition is detected, the SR.BUSERR bit is set and the TWIS waits for a new START condition.

28.8.4 Slave Transmitter Mode

If the TWIS matches an address in which the R/$\overline{W}$ bit in the TWI address phase transfer is set, it will enter slave transmitter mode and set the SR.TRA bit (note that SR.TRA is set one CLK_TWIS cycle after the relevant address match bit in the same register is set).

After the address phase, the following actions are performed:

1. If SMBus mode and PEC is used, NBYTES must be set up with the number of bytes to transmit. This is necessary in order to know when to transmit the PEC byte. NBYTES can also be used to count the number of bytes received if using DMA.
2. Byte to transmit depends on I²C/SMBus mode and CR.PEC:
   – If in I²C mode or CR.PEC is zero or NBYTES is non-zero: The TWIS waits until THR contains a valid data byte, possibly stretching the low period of TWCK. After THR contains a valid data byte, the data byte is transferred to a shifter, and then SR.TXRDY is changed to one because the THR is empty again.
   – SMBus mode and CR.PEC is one: If NBYTES is zero, the generated PEC byte is automatically transmitted instead of a data byte from THR. TWCK will not be stretched by the TWIS.
3. The data byte in the shifter is transmitted.
4. NBYTES is updated. If CR.CUP is one, NBYTES is incremented, otherwise NBYTES is decremented.
5. After each data byte has been transmitted, the master transmits an ACK (Acknowledge) or NAK (Not Acknowledge) bit. If a NAK bit is received by the TWIS, the SR.NAK bit is set. Note that this is done two CLK_TWIS cycles after TWCK has been sampled by the TWIS to be HIGH (see Figure 28-9). The NAK indicates that the transfer is finished, and the TWIS will wait for a STOP or REPEATED START. If an ACK bit is received, the SR.NAK bit remains LOW. The ACK indicates that more data should be transmitted, jump to step 2. At the end of the ACK/NAK clock cycle, the Byte Transfer Finished (SR.BTF) bit is set. Note that this is done two CLK_TWIS cycles after TWCK has been sampled by the TWIS to be LOW (see Figure 28-9). Also note that in the event that SR.NAK bit is set, it must not be cleared before the SR.BTF bit is set to ensure correct TWIS behavior.
6. If STOP is received, SR.TCOMP and SR.STO will be set.
7. If REPEATED START is received, SR.REP will be set.

The TWI transfers require the receiver to acknowledge each received data byte. During the acknowledge clock pulse (9th pulse), the slave releases the data line (HIGH), enabling the master to pull it down in order to generate the acknowledge. The slave polls the data line during this clock pulse and sets the NAK bit in SR if the master does not acknowledge the data byte. A NAK means that the master does not wish to receive additional data bytes. As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable Register (IER).

SR.TXRDY is used as Transmit Ready for the Peripheral DMA Controller transmit channel.

The end of the complete transfer is marked by the SR.TCOMP bit changing from zero to one. See Figure 28-7 and Figure 28-8.

Figure 28-7. Slave Transmitter with One Data Byte

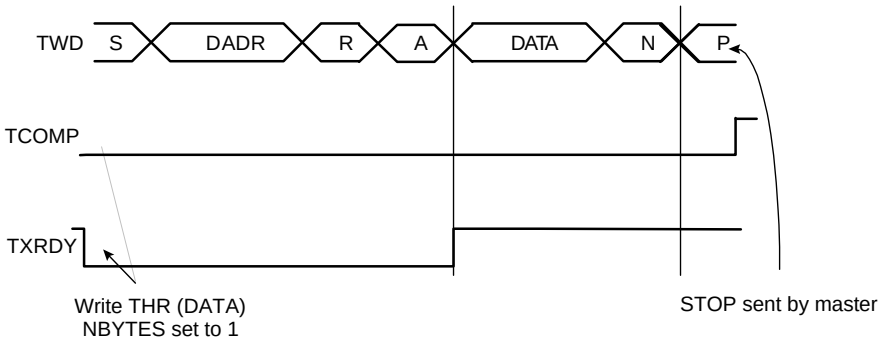


Figure 28-8. Slave Transmitter with Multiple Data Bytes

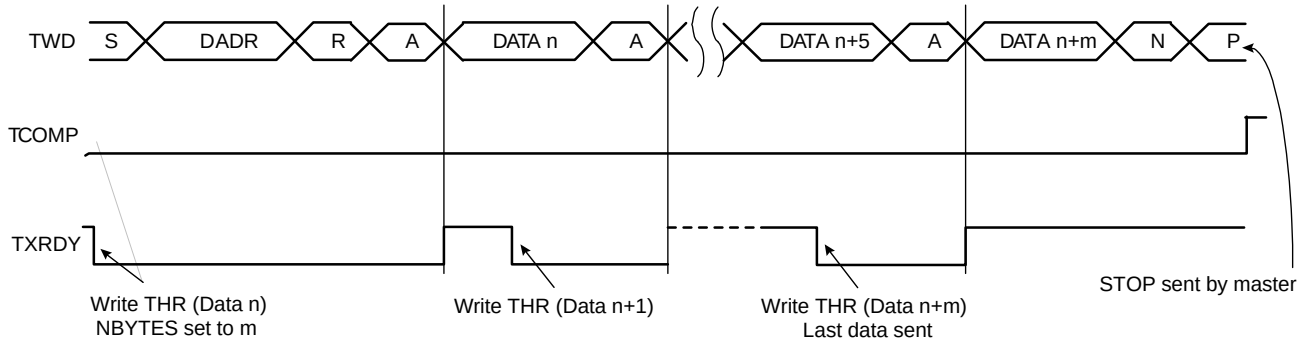
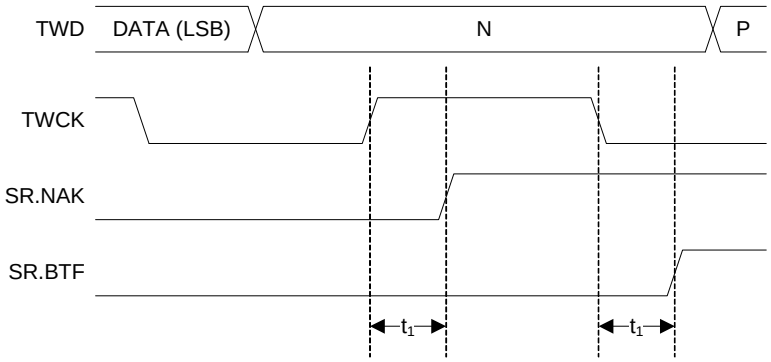


Figure 28-9. Timing Relationship between TWCK, SR.NAK, and SR.BTF



t_1: (CLK_TWIS period) x 2

28.8.5 Slave Receiver Mode

If the TWIS matches an address in which the R/$\overline{W}$ bit in the TWI address phase transfer is cleared, it will enter slave receiver mode and clear SR.TRA (note that SR.TRA is cleared one CLK_TWIS cycle after the relevant address match bit in the same register is set).

After the address phase, the following is repeated:

1. If SMBus mode and PEC is used, NBYTES must be set up with the number of bytes to receive. This is necessary in order to know which of the received bytes is the PEC byte. NBYTES can also be used to count the number of bytes received if using DMA.
2. Receive a byte. Set SR.BTF when done.
3. Update NBYTES. If CR.CUP is written to one, NBYTES is incremented, otherwise NBYTES is decremented. NBYTES is usually configured to count downwards if PEC is used.
4. After a data byte has been received, the slave transmits an ACK or NAK bit. For ordinary data bytes, the CR.ACK field controls if an ACK or NAK should be returned. If PEC is enabled and the last byte received was a PEC byte (indicated by NBYTES equal to zero), The TWIS will automatically return an ACK if the PEC value was correct, otherwise a NAK will be returned.
5. If STOP is received, SR.TCOMP will be set.
6. If REPEATED START is received, SR.REP will be set.

The TWI transfers require the receiver to acknowledge each received data byte. During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse.

The SR.RXRDY bit indicates that a data byte is available in the RHR. The RXRDY bit is also used as Receive Ready for the Peripheral DMA Controller receive channel.

Figure 28-10. Slave Receiver with One Data Byte

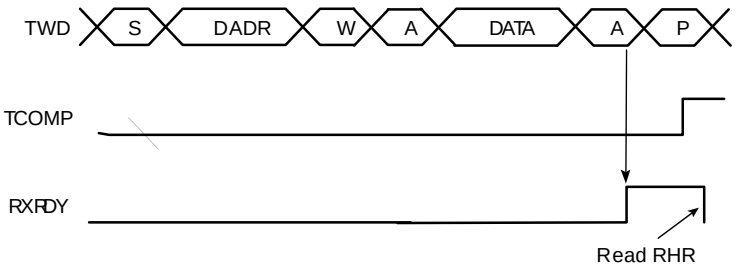
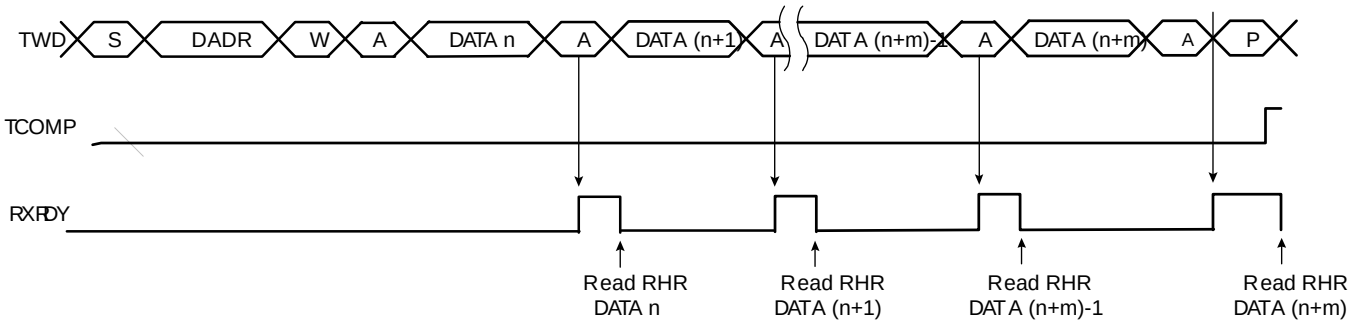


Figure 28-11. Slave Receiver with Multiple Data Bytes



28.8.6 Interactive ACKing Received Data Bytes

When implementing a register interface over TWI, it may sometimes be necessary or just useful to report reads and writes to invalid register addresses by sending a NAK to the host. To be able to do this, one must first receive the register address from the TWI bus, and then tell the TWIS

whether to ACK or NAK it. In normal operation of the TWIS, this is not possible because the controller will automatically ACK the byte at about the same time as the RXRDY bit changes from zero to one. Writing a one to the Stretch on Data Byte Received bit (CR.SODR) will stretch the clock allowing the user to update CR.ACK bit before returning the desired value. After the last bit in the data byte is received, the TWI bus clock is stretched, the received data byte is transferred to the RHR register, and SR.BTF is set. At this time, the user can examine the received byte and write the desired ACK or NACK value to CR.ACK. When the user clears SR.BTF, the desired ACK value is transferred on the TWI bus. This makes it possible to look at the byte received, determine if it is valid, and then decide to ACK or NAK it.

28.8.7 Using the Peripheral DMA Controller

The use of the Peripheral DMA Controller significantly reduces the CPU load. The user can set up ring buffers for the Peripheral DMA Controller, containing data to transmit or free buffer space to place received data. By initializing NBYTES to zero before a transfer, and writing a one to CR.CUP, NBYTES is incremented by one each time a data has been transmitted or received. This allows the user to detect how much data was actually transferred by the DMA system.

To assure correct behavior, respect the following programming sequences:

28.8.7.1 Data Transmit with the Peripheral DMA Controller

1. Initialize the transmit Peripheral DMA Controller (memory pointers, size, etc.).
2. Configure the TWIS (ADR, NBYTES, etc.).
3. Start the transfer by enabling the Peripheral DMA Controller to transmit.
4. Wait for the Peripheral DMA Controller end-of-transmit flag.
5. Disable the Peripheral DMA Controller.

28.8.7.2 Data Receive with the Peripheral DMA Controller

1. Initialize the receive Peripheral DMA Controller (memory pointers, size - 1, etc.).
2. Configure the TWIS (ADR, NBYTES, etc.).
3. Start the transfer by enabling the Peripheral DMA Controller to receive.
4. Wait for the Peripheral DMA Controller end-of-receive flag.
5. Disable the Peripheral DMA Controller.

28.8.8 SMBus Mode

SMBus mode is enabled by writing a one to the SMBus Mode Enable (SMEN) bit in CR. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be written to TR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses have been reserved for protocol handling, such as Alert Response Address (ARA) and Host Header (HH) Address. Address matching on these addresses can be enabled by configuring CR appropriately.

28.8.8.1 Packet Error Checking (PEC)

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to the Packet Error Checking Enable (PECEN) bit in CR enables automatic PEC handling in the

current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on following linked transfers will be correct.

In slave receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NAK value. The SR.SMBPECERR bit is set automatically if a PEC error occurred.

In slave transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

The PEC byte is automatically inserted in a slave transmitter transmission if PEC enabled when NBYTES reaches zero. The PEC byte is identified in a slave receiver transmission if PEC enabled when NBYTES reaches zero. NBYTES must therefore be set to the total number of data bytes in the transmission, including the PEC byte.

28.8.8.2 Timeouts

The Timing Register (TR) configures the SMBus timeout values. If a timeout occurs, the slave will leave the bus. The SR.SMBTOUT bit is also set.

28.8.9 Wakeup from Sleep Modes by TWI Address Match

The TWIS is able to wake the device up from a sleep mode upon an address match, including sleep modes where CLK_TWIS is stopped. After detecting the START condition on the bus, The TWIS will stretch TWCK until CLK_TWIS has started. The time required for starting CLK_TWIS depends on which sleep mode the device is in. After CLK_TWIS has started, the TWIS releases its TWCK stretching and receives one byte of data on the bus. At this time, only a limited part of the device, including the TWIS, receives a clock, thus saving power. If the received byte is a master code, the TWIS enters HS-mode. The TWIS goes on to receive the slave address. If the address phase causes a TWIS address match, the entire device is wakened and normal TWIS address matching actions are performed. Normal TWI transfer then follows. If the TWIS is not addressed, CLK_TWIS is automatically stopped and the device immediately returns to its original sleep mode. If the TWIS is in HS-mode, it remains so until it detects a STOP condition on the bus, after which it switches back to F/S-mode.

28.8.10 Identifying Bus Events

This chapter lists the different bus events, and how these affects the bits in the TWIS registers. This is intended to help writing drivers for the TWIS.

Table 28-5. Bus Events

Event	Effect
Slave transmitter has sent a data byte	SR.THR is cleared. SR.BTF is set. The value of the ACK bit sent immediately after the data byte is given by CR.ACK.
Slave receiver has received a data byte	SR.RHR is set. SR.BTF is set. SR.NAK updated according to value of ACK bit received from master.
Start+Sadr on bus, but address is to another slave	None.
Start+Sadr on bus, current slave is addressed, but address match enable bit in CR is not set	None.
Start+Sadr on bus, current slave is addressed, corresponding address match enable bit in CR set	Correct address match bit in SR is set. SR.TRA updated according to transfer direction (updating is done one CLK_TWIS cycle after address match bit is set) Slave enters appropriate transfer direction mode and data transfer can commence.
Start+Sadr on bus, current slave is addressed, corresponding address match enable bit in CR set, SR.STREN and SR.SOAM are set.	Correct address match bit in SR is set. SR.TRA updated according to transfer direction (updating is done one CLK_TWIS cycle after address match bit is set). Slave stretches TWCK immediately after transmitting the address ACK bit. TWCK remains stretched until all address match bits in SR have been cleared. Slave enters appropriate transfer direction mode and data transfer can commence.
Repeated Start received after being addressed	SR.REP set. SR.TCOMP unchanged.
Stop received after being addressed	SR.STO set. SR.TCOMP set.
Start, Repeated Start, or Stop received in illegal position on bus	SR.BUSERR set. SR.STO and SR.TCOMP may or may not be set depending on the exact position of an illegal stop.
Data is to be received in slave receiver mode, SR.STREN is set, and RHR is full	TWCK is stretched until RHR has been read.
Data is to be transmitted in slave receiver mode, SR.STREN is set, and THR is empty	TWCK is stretched until THR has been written.

Table 28-5. Bus Events

Event	Effect
Data is to be received in slave receiver mode, SR.STREN is cleared, and RHR is full	TWCK is not stretched, read data is discarded. SR.ORUN is set.
Data is to be transmitted in slave receiver mode, SR.STREN is cleared, and THR is empty	TWCK is not stretched, previous contents of THR is written to bus. SR.URUN is set.
SMBus timeout received	SR.SMBTOUT is set. TWCK and TWD are immediately released.
Slave transmitter in SMBus PEC mode has transmitted a PEC byte, that was not identical to the PEC calculated by the master receiver.	Master receiver will transmit a NAK as usual after the last byte of a master receiver transfer. Master receiver will retry the transfer at a later time.
Slave receiver discovers SMBus PEC Error	SR.SMBPECERR is set. NAK returned after the data byte.

28.9 User Interface

Table 28-6. TWIS Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Read/Write	0x00000000
0x04	NBYTES Register	NBYTES	Read/Write	0x00000000
0x08	Timing Register	TR	Read/Write	0x00000000
0x0C	Receive Holding Register	RHR	Read-only	0x00000000
0x10	Transmit Holding Register	THR	Write-only	0x00000000
0x14	Packet Error Check Register	PECR	Read-only	0x00000000
0x18	Status Register	SR	Read-only	0x00000002
0x1C	Interrupt Enable Register	IER	Write-only	0x00000000
0x20	Interrupt Disable Register	IDR	Write-only	0x00000000
0x24	Interrupt Mask Register	IMR	Read-only	0x00000000
0x28	Status Clear Register	SCR	Write-only	0x00000000
0x2C	Parameter Register	PR	Read-only	-(1)
0x30	Version Register	VR	Read-only	-(1)
0x34	HS-mode Timing Register	HSTR	Read/Write	0x00000000
0x38	Slew Rate Register	SRR	Read/Write	0x00000000
0x3C	HS-mode Slew Rate Register	HSSRR	Read/Write	0x00000000

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

28.9.1 Control Register

Name: CR

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	TENBIT	ADR[9:8]	

23	22	21	20	19	18	17	16
ADR[7:0]							

15	14	13	12	11	10	9	8
SODR	SOAM	CUP	ACK	PECEN	SMHH	SMDA	-

7	6	5	4	3	2	1	0
SWRST	-	-	STREN	GCMATCH	SMATCH	SMEN	SEN

- **TENBIT: Ten Bit Address Match**
  0: Disables Ten Bit Address Match.
  1: Enables Ten Bit Address Match.
- **ADR: Slave Address**
  Slave address used in slave address match. Bits 9:0 are used if in 10-bit mode, bits 6:0 otherwise.
- **SODR: Stretch Clock on Data Byte Reception**
  0: Does not stretch bus clock immediately before ACKing a received data byte.
  1: Stretches bus clock immediately before ACKing a received data byte.
- **SOAM: Stretch Clock on Address Match**
  0: Does not stretch bus clock after address match.
  1: Stretches bus clock after address match.
- **CUP: NBYTES Count Up**
  0: Causes NBYTES to count down (decrement) per byte transferred.
  1: Causes NBYTES to count up (increment) per byte transferred.
- **ACK: Slave Receiver Data Phase ACK Value**
  0: Causes a low value to be returned in the ACK cycle of the data phase in slave receiver mode.
  1: Causes a high value to be returned in the ACK cycle of the data phase in slave receiver mode.
- **PECEN: Packet Error Checking Enable**
  0: Disables SMBus PEC (CRC) generation and check.
  1: Enables SMBus PEC (CRC) generation and check.
- **SMHH: SMBus Host Header**
  0: Causes the TWIS not to acknowledge the SMBus Host Header.
  1: Causes the TWIS to acknowledge the SMBus Host Header.
- **SMDA: SMBus Default Address**
  0: Causes the TWIS not to acknowledge the SMBus Default Address.
  1: Causes the TWIS to acknowledge the SMBus Default Address.
- **SWRST: Software Reset**
  This bit will always read as 0.
  Writing a zero to this bit has no effect.

Writing a one to this bit resets the TWIS.

- **STREN: Clock Stretch Enable**
  0: Disables clock stretching if RHR/THR buffer full/empty. May cause over/underrun.
  1: Enables clock stretching if RHR/THR buffer full/empty.
- **GCMATCH: General Call Address Match**
  0: Causes the TWIS not to acknowledge the General Call Address.
  1: Causes the TWIS to acknowledge the General Call Address.
- **SMATCH: Slave Address Match**
  0: Causes the TWIS not to acknowledge the Slave Address.
  1: Causes the TWIS to acknowledge the Slave Address.
- **SMEN: SMBus Mode Enable**
  0: Disables SMBus mode.
  1: Enables SMBus mode.
- **SEN: Slave Enable**
  0: Disables the slave interface.
  1: Enables the slave interface.

28.9.2 NBYTES Register

Name: NBYTES

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
NBYTES							

- **NBYTES: Number of Bytes to Transfer**
  Writing to this field updates the NBYTES counter. The field can also be read to learn the progress of the transfer. NBYTES can be incremented or decremented automatically by hardware.

28.9.3 Timing Register

Name: TR

Access Type: Read/Write

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
EXP				-	-	-	-

23	22	21	20	19	18	17	16
SUDAT							

15	14	13	12	11	10	9	8
TTOUT							

7	6	5	4	3	2	1	0
TLOWS							

- **EXP: Clock Prescaler**
  Used to specify how to prescale the SMBus TLOWS counter. The counter is prescaled according to the following formula:

$$f_{PRESCALED} = \frac{f_{CLK\_TWIS}}{2^{(EXP+1)}}$$

- **SUDAT: Data Setup Cycles**
  Non-prescaled clock cycles for data setup count. Used to time T_{SU_DAT}. Data is driven SUDAT cycles after TWCK low detected. This timing is used for timing the ACK/NAK bits, and any data bits driven in slave transmitter mode.
- **TTOUT: SMBus $T_{TIMEOUT}$ Cycles**
  Prescaled clock cycles used to time SMBus $T_{TIMEOUT}$.
- **TLOWS: SMBus $T_{LOW:SEXT}$ Cycles**
  Prescaled clock cycles used to time SMBus $T_{LOW:SEXT}$.

28.9.4 Receive Holding Register

Name: RHR

Access Type: Read-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
RXDATA							

- **RXDATA: Received Data Byte**
  When the RXRDY bit in the Status Register (SR) is one, this field contains a byte received from the TWI bus.

28.9.5 Transmit Holding Register

Name: THR

Access Type: Write-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
TXDATA							

- **TXDATA: Data Byte to Transmit**
  Write data to be transferred on the TWI bus here.

28.9.6 Packet Error Check Register

Name: PECR

Access Type: Read-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
PEC							

- **PEC: Calculated PEC Value**
  The calculated PEC value. Updated automatically by hardware after each byte has been transferred. Reset by hardware after a STOP condition. Provided if the user manually wishes to control when the PEC byte is transmitted, or wishes to access the PEC value for other reasons. In ordinary operation, the PEC handling is done automatically by hardware.

28.9.7 Status Register

Name: SR

Access Type: Read-only

Offset: 0x18

Reset Value: 0x000000002

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
BTF	REP	STO	SMBDAM	SMBHHM	-	GCM	SAM

15	14	13	12	11	10	9	8
-	BUSERR	SMBPECERR	SMBTOUT	-	-	-	NAK

7	6	5	4	3	2	1	0
ORUN	URUN	TRA	-	TCOMP	SEN	TXRDY	RXRDY

- **BTF: Byte Transfer Finished**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when byte transfer has completed.
- **REP: Repeated Start Received**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when a REPEATED START condition is received.
- **STO: Stop Received**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the STOP condition is received.
- **SMBDAM: SMBus Default Address Match**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the received address matched the SMBus Default Address.
- **SMBHHM: SMBus Host Header Address Match**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the received address matched the SMBus Host Header Address.
- **GCM: General Call Match**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the received address matched the General Call Address.
- **SAM: Slave Address Match**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when the received address matched the Slave Address.
- **BUSERR: Bus Error**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when a misplaced START or STOP condition has occurred.
- **SMBPECERR: SMBus PEC Error**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when a SMBus PEC error has occurred.

- **SMBTOUT: SMBus Timeout**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when a SMBus timeout has occurred.
- **NAK: NAK Received**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when a NAK was received from the master during slave transmitter operation.
- **ORUN: Overrun**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when an overrun has occurred in slave receiver mode. Can only occur if CR.STREN is zero.
- **URUN: Underrun**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when an underrun has occurred in slave transmitter mode. Can only occur if CR.STREN is zero.
- **TRA: Transmitter Mode**
  0: The slave is in slave receiver mode.
  1: The slave is in slave transmitter mode.
- **TCOMP: Transmission Complete**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when transmission is complete. Set after receiving a STOP after being addressed.
- **SEN: Slave Enabled**
  0: The slave interface is disabled.
  1: The slave interface is enabled.
- **TXRDY: TX Buffer Ready**
  0: The TX buffer is full and should not be written to.
  1: The TX buffer is empty, and can accept new data.
- **RXRDY: RX Buffer Ready**
  0: No RX data ready in RHR.
  1: RX data is ready to be read from RHR.

28.9.8 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
BTF	REP	STO	SMBDAM	SMBHHM	-	GCM	SAM
15	14	13	12	11	10	9	8
-	BUSERR	SMBPECERR	SMBTOUT	-	-	-	NAK
7	6	5	4	3	2	1	0
ORUN	URUN	-	-	TCOMP	-	TXRDY	RXRDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will write a one to the corresponding bit in IMR.

28.9.9 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
BTF	REP	STO	SMBDAM	SMBHHM	-	GCM	SAM

15	14	13	12	11	10	9	8
-	BUSERR	SMBPECERR	SMBTOUT	-	-	-	NAK

7	6	5	4	3	2	1	0
ORUN	URUN	-	-	TCOMP	-	TXRDY	RXRDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

28.9.10 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
BTF	REP	STO	SMBDAM	SMBHHM	-	GCM	SAM
15	14	13	12	11	10	9	8
-	BUSERR	SMBPECERR	SMBTOUT	-	-	-	NAK
7	6	5	4	3	2	1	0
ORUN	URUN	-	-	TCOMP	-	TXRDY	RXRDY

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
This bit is cleared when the corresponding bit in IDR is written to one.
This bit is set when the corresponding bit in IER is written to one.

28.9.11 Status Clear Register

Name: SCR

Access Type: Write-only

Offset: 0x28

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
BTF	REP	STO	SMBDAM	SMBHHM	-	GCM	SAM

15	14	13	12	11	10	9	8
-	BUSERR	SMBPECERR	SMBTOUT	-	-	-	NAK

7	6	5	4	3	2	1	0
ORUN	URUN	-	-	TCOMP	-	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

28.9.12 Parameter Register

Name: PR

Access Type: Read-only

Offset: 0x2C

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	HS

- **HS: HS-mode**
  0: High-speed-mode is not supported.
  1: High-speed-mode is supported.

28.9.13 Version Register

Name: VR

Access Type: Read-only

Offset: 0x30

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION [11:8]			

7	6	5	4	3	2	1	0
VERSION [7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

28.9.14 HS-mode Timing Register

Name: HSTR

Access Type: Read/Write

Offset: 0x34

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
HDDAT							

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

- **HDDAT: Data Hold Cycles**

  Non-prescaled clock cycles for data hold count when the TWIS is in HS-mode. Used to time T_{HD_DAT}. Data is driven HDDAT cycles after a LOW on TWCK is detected. This timing is used for timing the ACK/NAK bits, and any data bits driven in slave transmitter mode.

28.9.15 Slew Rate Register

Name: SRR

Access Type: Read/Write

Offset: 0x38

Reset Value: 0x00000000

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

- **FILTER: Input Spike Filter Control**

FILTER Value	Function
0	Reserved
1	Reserved
2	Standard- or Fast-mode
3	Fast-mode Plus

- **DASLEW: Data Slew Limit**
  Selects the slew limit of the TWD output buffer in F/S-mode.
- **DADRIVEL: Data Drive Strength LOW**
  Selects the pull-down drive strength of the TWD output buffer in F/S-mode.

Refer to Section 42. ”Electrical Characteristics” on page 1121 **for appropriates values of these register fields.**

28.9.16 HS-mode Slew Rate Register

Name: HSSRR

Access Type: Read/Write

Offset: 0x3C

Reset Value: 0x00000000

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

- **FILTER: Input Spike Filter Control**

FILTER Value	Function
0	Reserved
1	HS-mode
2	Reserved
3	Reserved

- **DASLEW: Data Slew Limit**
  Selects the slew limit of the TWD output buffer in HS-mode.
- **DADRIVEL: Data Drive Strength LOW**
  Selects the pull-down drive strength of the TWD output buffer in HS-mode.

Refer to Section 42. ”Electrical Characteristics” on page 1121 **for appropriates values of these register fields.**

28.10 Module Configuration

The specific configuration for each TWIS instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 28-7. Module Clock Name

Module Name	Clock Name	Description
TWIS0	CLK_TWIS0	Clock for the TWIS0 bus interface
TWIS1	CLK_TWIS1	Clock for the TWIS1 bus interface

Note : TWI2 and TWI3 are master only. TWI0 and TWI1 are master and slave

Table 28-8. Register Reset Values

Register	Reset Value
VERSION	0x00000140
PARAMETER	0x00000001

29. Inter-IC Sound Controller (IISC)

Rev: 1.0.0.0

29.1 Features

- **Compliant with Inter-IC Sound (I^2S) bus specification**
- **Master, slave, and controller modes:**
  - **Slave: data received/transmitted**
  - **Master: data received/transmitted and clocks generated**
  - **Controller: clocks generated**
- **Individual enable and disable of receiver, transmitter, and clocks**
- **Configurable clock generator common to receiver and transmitter:**
  - **Suitable for a wide range of sample frequencies (fs), including 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, and 192 kHz**
  - **16 fs to 1024 fs Master Clock generated for external oversampling ADCs**
- **Several data formats supported:**
  - **32-, 24-, 20-, 18-, 16-, and 8-bit mono or stereo format**
  - **16- and 8-bit compact stereo format, with left and right samples packed in the same word to reduce data transfers**
- **DMA interfaces for receiver and transmitter to reduce processor overhead:**
  - **Either one DMA channel for both audio channels, or**
  - **One DMA channel per audio channel**
- **Smart holding registers management to avoid audio channels mix after overrun or underrun**

29.2 Overview

The Inter-IC Sound Controller (IISC) provides a 5-wire, bidirectional, synchronous, digital audio link with external audio devices: ISDI, ISDO, IWS, ISCK, and IMCK pins.

This controller is compliant with the Inter-IC Sound (I^2S) bus specification.

The IISC consists of a Receiver, a Transmitter, and a common Clock Generator, that can be enabled separately, to provide Master, Slave, or Controller modes with Receiver, Transmitter, or both active.

Peripheral DMA channels, separate for the Receiver and for the Transmitter, allow a continuous high bitrate data transfer without processor intervention to the following:

- Audio CODECs in Master, Slave, or Controller mode
- Stereo DAC or ADC through dedicated I^2S serial interface

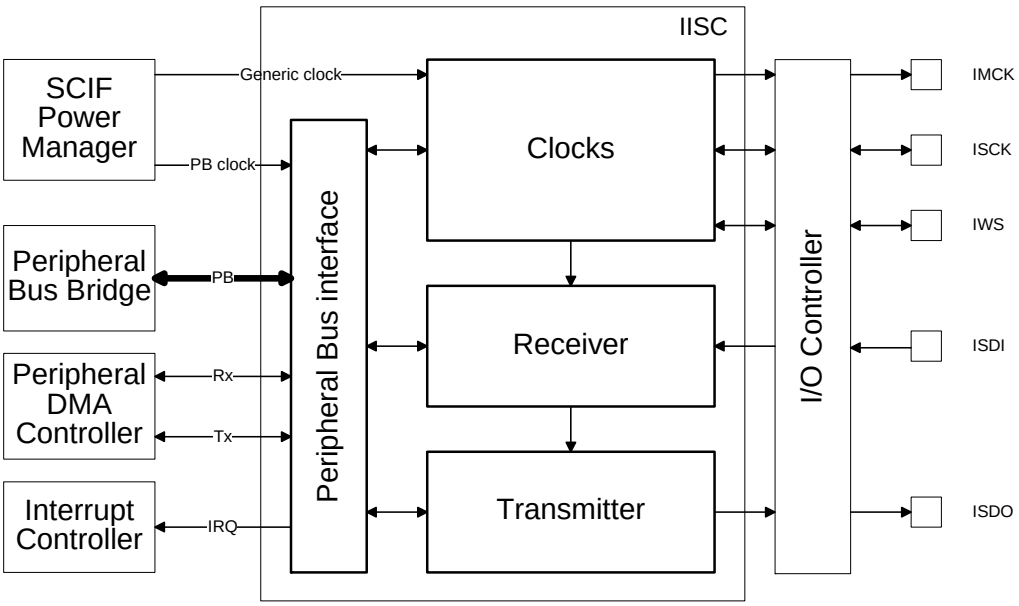
The IISC can use either a single DMA channel for both audio channels or one DMA channel per audio channel.

The 8- and 16-bit compact stereo format allows reducing the required DMA bandwidth by transferring the left and right samples within the same data word.

In Master Mode, the IISC allows outputting a 16 fs to 1024 fs Master Clock, in order to provide an oversampling clock to an external audio codec or digital signal processor (DSP).

29.3 Block Diagram

Figure 29-1. IISC Block Diagram



29.4 I/O Lines Description

Table 29-1. I/O Lines Description

Pin Name	Pin Description	Type
IMCK	Master Clock	Output
ISCK	Serial Clock	Input/Output
IWS	I^2S Word Select	Input/Output
ISDI	Serial Data Input	Input
ISDO	Serial Data Output	Output

29.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

29.5.1 I/O lines

The IISC pins may be multiplexed with I/O Controller lines. The user must first program the I/O Controller to assign the desired IISC pins to their peripheral function. If the IISC I/O lines are not used by the application, they can be used for other purposes by the I/O Controller. It is required to enable only the IISC inputs and outputs actually in use.

29.5.2 Power Management

If the CPU enters a sleep mode that disables clocks used by the IISC, the IISC will stop functioning and resume operation after the system wakes up from sleep mode.

29.5.3 Clocks

The clock for the IISC bus interface (CLK_IISC) is generated by the Power Manager. It is recommended to disable the IISC before disabling the clock, to avoid freezing the IISC in an undefined state.

One of the generic clocks is connected to the IISC. The generic clock (GCLK_IISC) can be set to a wide range of frequencies and clock sources. The GCLK_IISC must be enabled and configured before use. Refer to the module configuration section for details on the GCLK_IISC used for the IISC. The frequency for this clock has to be set as described in Table.

29.5.4 DMA

The IISC DMA handshake interfaces are connected to the Peripheral DMA Controller. Using the IISC DMA functionality requires the Peripheral DMA Controller to be programmed first.

29.5.5 Interrupts

The IISC interrupt line is connected to the Interrupt Controller. Using the IISC interrupt requires the Interrupt Controller to be programmed first.

29.5.6 Debug Operation

When an external debugger forces the CPU into debug mode, the IISC continues normal operation. If this module is configured in a way that requires it to be periodically serviced by the CPU through interrupt requests or similar, improper operation or data loss may result during debugging.

29.6 Functional Description

29.6.1 Initialization

The IISC features a Receiver, a Transmitter, and, for Master and Controller modes, a Clock Generator. Receiver and Transmitter share the same Serial Clock and Word Select.

Before enabling the IISC, the chosen configuration must be written to the Mode Register (MR). The IMCKMODE, MODE, and DATALENGTH fields in the MR register must be written. If the IMCKMODE field is written as one, then the IMCKFS field should be written with the chosen ratio, as described in Section 29.6.5 “Serial Clock and Word Select Generation” on page 782.

Once the Mode Register has been written, the IISC Clock Generator, Receiver, and Transmitter can be enabled by writing a one to the CKEN, RXEN, and TXEN bits in the Control Register (CR). The Clock Generator can be enabled alone, in Controller Mode, to output clocks to the IMCK, ISCK, and IWS pins. The Clock Generator must also be enabled if the Receiver or the Transmitter is enabled.

The Clock Generator, Receiver, and Transmitter can be disabled independently by writing a one to CR.CXDIS, CR.RXDIS and/or CR.TXDIS respectively. Once requested to stop, they will only stop when the transmission of the pending frame transmission will be completed.

29.6.2 Basic Operation

The Receiver can be operated by reading the Receiver Holding Register (RHR), whenever the Receive Ready (RXRDY) bit in the Status Register (SR) is set. Successive values read from RHR will correspond to the samples from the left and right audio channels for the successive frames.

The Transmitter can be operated by writing to the Transmitter Holding Register (RHR), whenever the Transmit Ready (TXRDY) bit in the Status Register (SR) is set. Successive values written to THR should correspond to the samples from the left and right audio channels for the successive frames.

The Receive Ready and Transmit Ready bits can be polled by reading the Status Register.

The IISC processor load can be reduced by enabling interrupt-driven operation. The RXRDY and/or TXRDY interrupt requests can be enabled by writing a one to the corresponding bit in the Interrupt Enable Register (IER). The interrupt service routine associated to the IISC interrupt request will then be executed whenever the Receive Ready or the Transmit Ready status bit is set.

29.6.3 Master, Controller, and Slave Modes

In Master and Controller modes, the IISC provides the Master Clock, the Serial Clock and the Word Select. IMCK, ISCK, and IWS pins are outputs.

In Controller mode, the IISC Receiver and Transmitter are disabled. Only the clocks are enabled and used by an external receiver and/or transmitter.

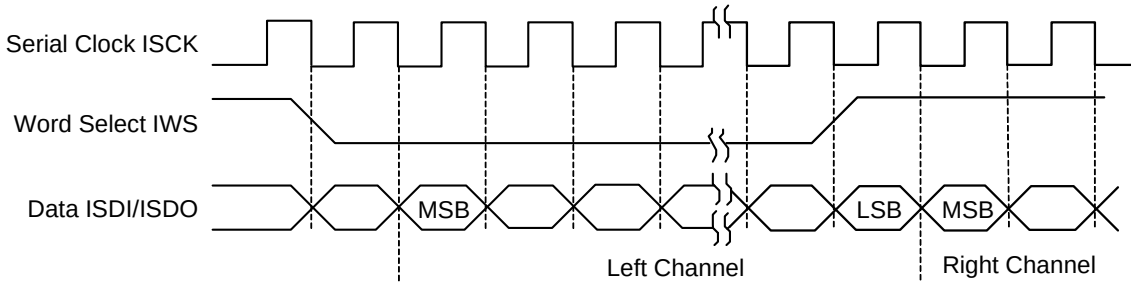
In Slave mode, the IISC receives the Serial Clock and the Word Select from an external master. ISCK and IWS pins are inputs.

The mode is selected by writing the MODE field of the Mode Register (MR). Since the MODE field changes the direction of the IWS and ISCK pins, the Mode Register should only be written when the IISC is stopped, in order to avoid unwanted glitches on the IWS and ISCK pins.

29.6.4 I^2S Reception and Transmission Sequence

As specified in the I^2S protocol, data bits are left-adjusted in the Word Select time slot, with the MSB transmitted first, starting one clock period after the transition on the Word Select line.

Figure 29-2. I^2S Reception and Transmission Sequence



Data bits are sent on the falling edge of the Serial Clock and sampled on the rising edge of the Serial Clock. The Word Select line indicates the channel in transmission, a low level for the left channel and a high level for the right channel.

The length of transmitted words can be chosen among 8, 16, 18, 20, 24, and 32 bits by writing the MR.DATALENGTH field.

If the time slot allows for more data bits than written in the MR.DATALENGTH field, zeroes are appended to the transmitted data word or extra received bits are discarded. If the time slot allows for less data bits than written, the extra bits to be transmitted are not sent or the missing bits are set to zero in the received data word.

29.6.5 Serial Clock and Word Select Generation

The generation of clocks in the IISC is described in Figure 29-3 on page 783.

In Slave mode, the Serial Clock and Word Select Clock are driven by an external master. ISCK and IWS pins are inputs and no generic clock is required by the IISC.

In Master mode, the user can configure the Master Clock, Serial Clock, and Word Select Clock through the Mode Register (MR). IMCK, ISCK, and IWS pins are outputs and a generic clock is used to derive the IISC clocks.

Audio codecs connected to the IISC pins may require a Master Clock signal with a frequency multiple of the audio sample frequency (fs), such as 256fs. When the IISC is in Master mode, writing a one to MR.IMCKMODE will output GCLK_IISC as Master Clock to the IMCK pin, and will divide GCLK_IISC to create the internal bit clock, output on the ISCK pin. The clock division factor is defined by writing to MR.IMCKFS and MR.DATALENGTH, as described "IMCKFS: Master Clock to fs Ratio" on page 789.

The Master Clock (IMCK) frequency is 16*(IMCKFS+1) times the sample frequency (fs), i.e. IWS frequency. The Serial Clock (ISCK) frequency is 2*Slot Length times the sample frequency (fs), where Slot Length is defined in Table 29-2 on page 782.

Table 29-2. Slot Length

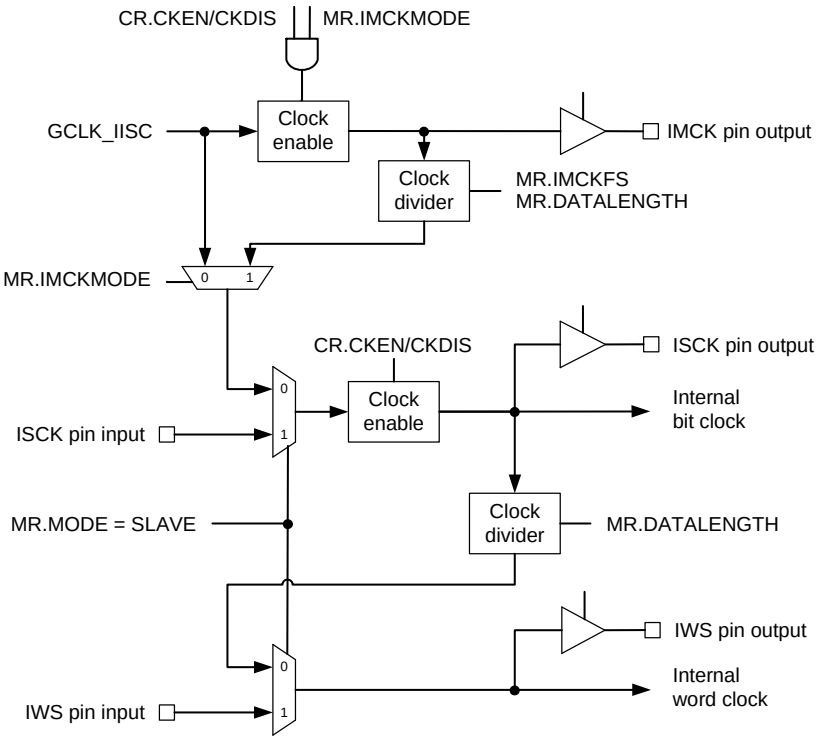
MR.DATALENGTH	Word Length	Slot Length
0	32 bits	32
1	24 bits	32 if MR.IWS24 is zero 24 if MR.IWS24 is one
2	20 bits	
3	18 bits	
4	16 bits	16
5	16 bits compact stereo	
6	8 bits	8
7	8 bits compact stereo	

Warning: MR.IMCKMODE should only be written as one if the Master Clock frequency is strictly higher than the Serial Clock.

If a Master Clock output is not required, the GCLK_IISC generic clock is used as ISCK, by writing a zero to MR.IMCKMODE. Alternatively, if the frequency of the generic clock used is a multiple of the required ISCK frequency, the IMCK to ISCK divider can be used with the ratio defined by writing the MR.IMCKFS field.

The IWS pin is used as Word Select as described in Section 29.6.4 "I2S Reception and Transmission Sequence" on page 781.

Figure 29-3. IISC Clocks Generation



29.6.6 Mono

When the Transmit Mono (TXMONO) in the Mode Register is set, data written to the left channel is duplicated to the right output channel.

When the Receive Mono (RXMONO) in the Mode Register is set, data received from the left channel is duplicated to the right channel.

29.6.7 Holding Registers

The IISC user interface includes a Receive Holding Register (RHR) and a Transmit Holding Register (THR). RHR and THR are used to access audio samples for both audio channels.

When a new data word is available in the RHR register, the Receive Ready bit (RXRDY) in the Status Register (SR) is set. Reading the RHR register will clear this bit.

A receive overrun condition occurs if a new data word becomes available before the previous data word has been read from the RHR register. Then, the Receive Overrun bit in the Status Register will be set and bit i of the RXORCH field in the Status Register is set, where i is the current receive channel number.

When the THR register is empty, the Transmit Ready bit (TXRDY) in the Status Register (SR) is set. Writing into the THR register will clear this bit.

A transmit underrun condition occurs if a new data word needs to be transmitted before it has been written to the THR register. Then, the Transmit Underrun bit in the Status Register will be set and bit i of the TXORCH field in the Status Register is set, where i is the current transmit channel number. If the TXSAME bit in the Mode Register is zero, then a zero data word is transmitted in case of underrun. If MR.TXSAME is one, then the previous data word for the current transmit channel number is transmitted.

Data words are right-justified in the RHR and THR registers. For 16-bit compact stereo, the left sample uses bits 15 through 0 and the right sample uses bits 31 through 16 of the same data word. For 8-bit compact stereo, the left sample uses bits 7 through 0 and the right sample uses bits 15 through 8 of the same data word.

29.6.8 DMA Operation

The Receiver and the Transmitter can each be connected either to one single Peripheral DMA channel or to one Peripheral DMA channel per data channel. This is selected by writing to the MR.RXDMA and MR.TXDMA bits. If a single Peripheral DMA channel is selected, all data samples use IISC Receiver or Transmitter DMA channel 0.

The Peripheral DMA reads from the RHR register and writes to the RHR register for both audio channels, successively.

The Peripheral DMA transfers may use 32-bit word, 16-bit halfword, or 8-bit byte according to the value of the MR.DATALENGTH field.

29.6.9 Loop-back Mode

For debugging purposes, the IISC can be configured to loop back the Transmitter to the Receiver. Writing a one to the MR.LOOP bit will internally connect ISDO to ISDI, so that the transmitted data is also received. Writing a zero to MR.LOOP will restore the normal behavior with independent Receiver and Transmitter. As for other changes to the Receiver or Transmitter configuration, the IISC Receiver and Transmitter must be disabled before writing to the MR register to update MR.LOOP.

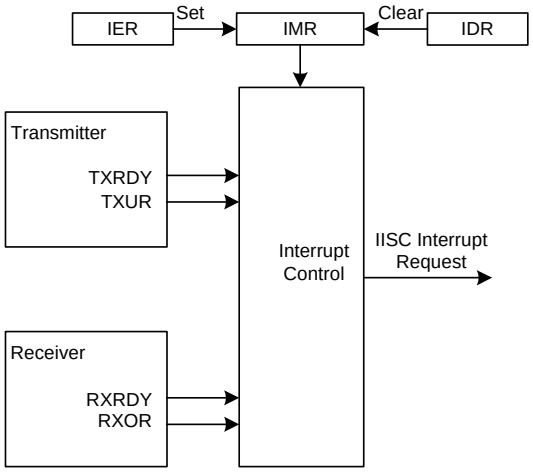
29.6.10 Interrupts

An IISC interrupt request can be triggered whenever one or several of the following bits are set in the Status Register (SR): Receive Ready (RXRDY), Receive Overrun (RXOR), Transmit Ready (TXRDY), or Transmit Underrun (TXOR).

The interrupt request will be generated if the corresponding bit in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in SR is cleared by writing a one the corresponding bit in the Status Clear Register (SCR).

For debugging purposes, interrupt requests can be simulated by writing a one to the corresponding bit in the Status Set Register (SSR).

Figure 29-4. Interrupt Block Diagram



29.7 IISC Application Examples

The IISC can support several serial communication modes used in audio or high-speed serial links. Some standard applications are shown in the following figures. All serial link applications supported by the IISC are not listed here.

Figure 29-5. Audio Application Block Diagram

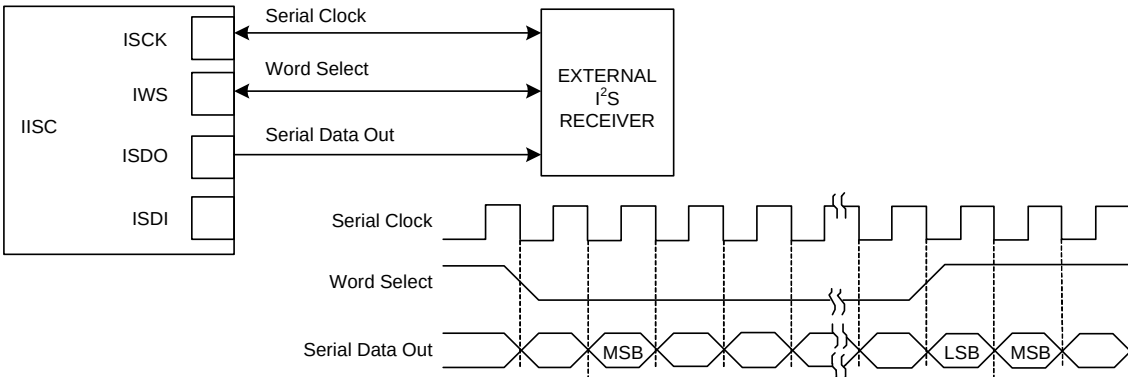


Figure 29-6. Codec Application Block Diagram

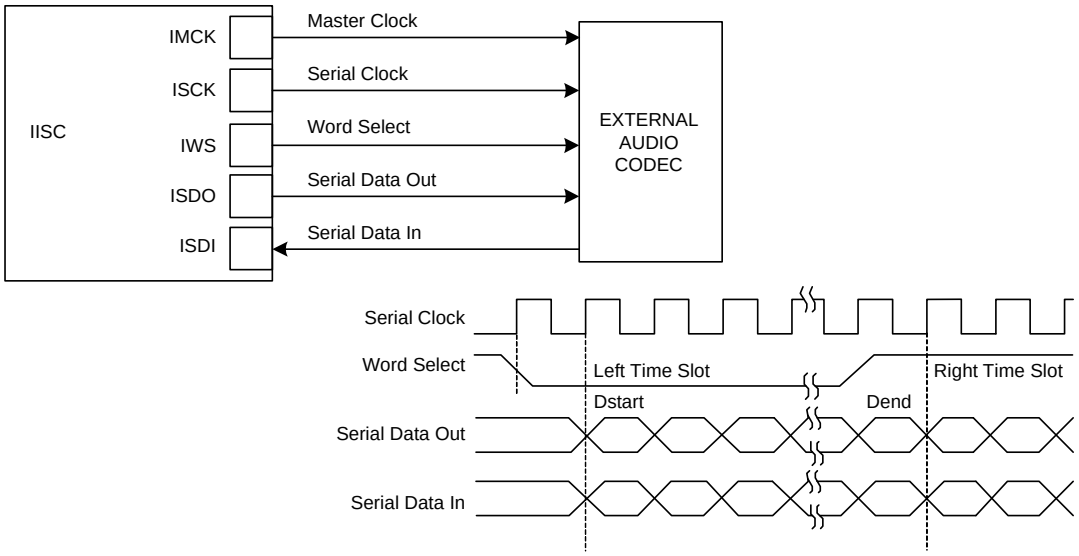
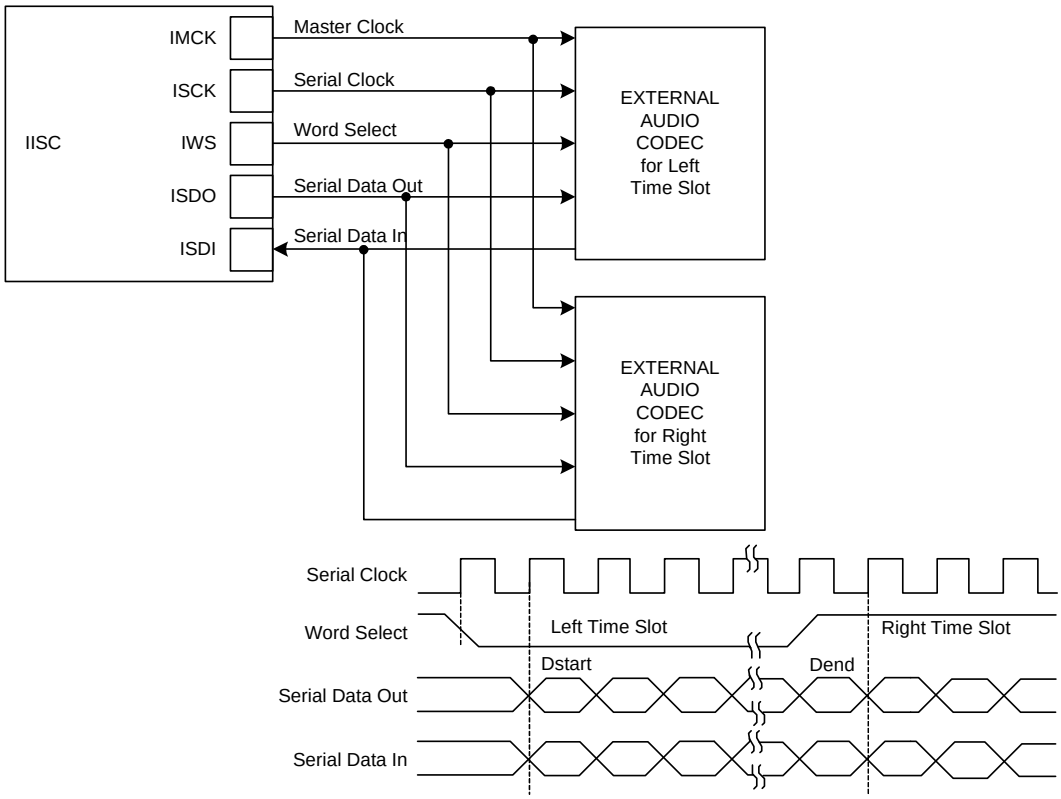


Figure 29-7. Time Slot Application Block Diagram



29.8 User Interface

Table 29-3. IISC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-only	0x00000000
0x04	Mode Register	MR	Read/Write	0x00000000
0x08	Status Register	SR	Read-only	0x00000000
0x0C	Status Clear Register	SCR	Write-only	0x00000000
0x10	Status Set Register	SSR	Write-only	0x00000000
0x14	Interrupt Enable Register	IER	Write-only	0x00000000
0x18	Interrupt Disable Register	IDR	Write-only	0x00000000
0x1C	Interrupt Mask Register	IMR	Read-only	0x00000000
0x20	Receiver Holding Register	RHR	Read-only	0x00000000
0x24	Transmitter Holding Register	THR	Write-only	0x00000000
0x28	Version Register	VERSION	Read-only	-(1)
0x2C	Parameter Register	PARAMETER	Read-only	-(1)

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

29.8.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
SWRST	-	TXDIS	TXEN	CKDIS	CKEN	RXDIS	RXEN

The Control Register should only be written to enable the IISC after the chosen configuration has been written to the Mode Register, in order to avoid unwanted glitches on the IWS, ISCK, and ISDO outputs. The proper sequence is to write the MR register, then write the CR register to enable the IISC, or to disable the IISC before writing a new value into MR.

- **SWRST: Software Reset**
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all the registers in the module. The module will be disabled after the reset.
  This bit always reads as zero.
- **TXDIS: Transmitter Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the IISC Transmitter. SR.TXEN will be cleared when the Transmitter is effectively stopped.
- **TXEN: Transmitter Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the IISC Transmitter, if TXDIS is not one. SR.TXEN will be set when the Transmitter is effectively started.
- **CKDIS: Clocks Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the IISC clocks generation.
- **CKEN: Clocks Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the IISC clocks generation, if CKDIS is not one.
- **RXDIS: Receiver Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the IISC Receiver. SR.TXEN will be cleared when the Transmitter is effectively stopped.
- **RXEN: Receiver Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the IISC Receiver, if RXDIS is not one. SR.RXEN will be set when the Receiver is effectively started.

29.8.2 Mode Register

Name: MR

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
IWS24	IMCKMODE	IMCKFS					
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	TXSAME	TXDMA	TXMONO		RXLOOP	RXDMA	RXMONO
7	6	5	4	3	2	1	0
-	-	-	DATALENGTH			-	MODE

The Mode Register should only be written when the IISC is stopped, in order to avoid unwanted glitches on the IWS, ISCK, and ISDO outputs. The proper sequence is to write the MR register, then write the CR register to enable the IISC, or to disable the IISC before writing a new value into MR.

- **IWS24: IWS TDM Slot Width**
  0: IWS slot is 32-bit wide for DATALENGTH=18/20/24-bit.
  1: IWS slot is 24-bit wide for DATALENGTH=18/20/24-bit.
  Refer to Table 29-2, “Slot Length,” on page 782.
- **IMCKMODE: Master Clock Mode**
  0: No Master Clock generated (generic clock is used as ISCK output).
  1: Master Clock generated (generic clock is used as IMCK output).
  Warning: if IMCK frequency is the same as ISCK, IMCKMODE should not be written as one. Refer to Section 29.6.5 “Serial Clock and Word Select Generation” on page 782 and Table 29-2, “Slot Length,” on page 782.
- **IMCKFS: Master Clock to fs Ratio**
  Master Clock frequency is 16*(IMCKFS+1) times the sample rate, i.e. IWS frequency:

Table 29-4. Master Clock to Sample Frequency (fs) Ratio

fs Ratio	IMCKFS
16 fs	0
32 fs	1
48fs	2
64 fs	3
96fs	5
128 fs	7
192fs	11
256 fs	15

Table 29-4. Master Clock to Sample Frequency (fs) Ratio

fs Ratio	IMCKFS
384 fs	23
512 fs	31
768 fs	47
1024 fs	63

- **TXSAME: Transmit Data when Underrun**
  0: Zero sample transmitted when underrun.
  1: Previous sample transmitted when underrun
- **TXDMA: Single or multiple DMA Channels for Transmitter**
  0: Transmitter uses a single DMA channel for both audio channels.
  1: Transmitter uses one DMA channel per audio channel.
- **TXMONO: Transmit Mono**
  0: Stereo.
  1: Mono, with left audio samples duplicated to right audio channel by the IISC.
- **RXLOOP: Loop-back Test Mode**
  0: Normal mode.
  1: ISDO output of IISC is internally connected to ISDI input.
- **RXMONO: Receive Mono**
  0: Stereo.
  1: Mono, with left audio samples duplicated to right audio channel by the IISC.
- **RXDMA: Single or multiple DMA Channels for Receiver**
  0: Receiver uses a single DMA channel for both audio channels.
  1: Receiver uses one DMA channel per audio channel.
- **DATALENGTH: Data Word Length**

Table 29-5. Data Word Length

DATALENGTH	Word Length	Comments
0	32 bits	
1	24 bits	
2	20 bits	
3	18 bits	
4	16 bits	
5	16 bits compact stereo	Left sample in bits 15 through 0 and right sample in bits 31 through 16 of the same word
6	8 bits	
7	8 bits compact stereo	Left sample in bits 7 through 0 and right sample in bits 15 through 8 of the same word

- **MODE: Mode**

Table 29-6. Mode

MODE		Comments
0	SLAVE	ISCK and IWS pin inputs used as Bit Clock and Word Select/Frame Sync.
1	MASTER	Bit Clock and Word Select/Frame Sync generated by IISC from GCLK_IISC and output to ISCK and IWS pins. GCLK_IISC is output as Master Clock on IMCK if MR.IMCKMODE is one.

29.8.3 Status Register

Name: SR

Access Type: Read-only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	TXURCH		-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	RXORCH	

7	6	5	4	3	2	1	0
-	TXUR	TXRDY	TXEN	-	RXOR	RXRDY	RXEN

- **TXURCH: Transmit Underrun Channel**
  This field is cleared when SCR.TXUR is written to one.
  Bit i of this field is set when a transmit underrun error occurred in channel i (i=0 for first channel of the frame).
- **RXORCH: Receive Overrun Channel**
  This field is cleared when SCR.RXOR is written to one.
  Bit i of this field is set when a receive overrun error occurred in channel i (i=0 for first channel of the frame).
- **TXUR: Transmit Underrun**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when an underrun error occurs on the THR register or when the corresponding bit in SSR is written to one.
- **TXRDY: Transmit Ready**
  This bit is cleared when data is written to THR.
  This bit is set when the THR register is empty and can be written with new data to be transmitted.
- **TXEN: Transmitter Enabled**
  This bit is cleared when the Transmitter is effectively disabled, following a CR.TXDIS or CR.SWRST request.
  This bit is set when the Transmitter is effectively enabled, following a CR.TXEN request.
- **RXOR: Receive Overrun**
  This bit is cleared when the corresponding bit in SCR is written to one.
  This bit is set when an overrun error occurs on the RHR register or when the corresponding bit in SSR is written to one.
- **RXRDY: Receive Ready**
  This bit is cleared when the RHR register is read.
  This bit is set when received data is present in the RHR register.
- **RXEN: Receiver Enabled**
  This bit is cleared when the Receiver is effectively disabled, following a CR.RXDIS or CR.SWRST request.
  This bit is set when the Receiver is effectively enabled, following a CR.RXEN request.

29.8.4 Status Clear Register

Name: SCR

Access Type: Write-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	TXURCH		-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	RXORCH	

7	6	5	4	3	2	1	0
-	TXUR	-	-	-	RXOR	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

29.8.5 Status Set Register

Name: SSR

Access Type: Write-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	TXURCH		-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	RXORCH	

7	6	5	4	3	2	1	0
-	TXUR	-	-	-	RXOR	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in SR.

29.8.6 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	TXUR	TXRDY	-	-	RXOR	RXRDY	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

29.8.7 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	TXUR	TXRDY	-	-	RXOR	RXRDY	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

29.8.8 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	TXUR	TXRDY	-	-	RXOR	RXRDY	-

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

29.8.9 Receive Holding Register

Name: RHR

Access Type: Read-only

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
RHR[31:24]							

23	22	21	20	19	18	17	16
RHR[23:16]							

15	14	13	12	11	10	9	8
RHR[15:8]							

7	6	5	4	3	2	1	0
RHR[7:0]							

- **RHR: Received Word**
  This field is set by hardware to the last received data word. If MR.DATALENGTH specifies less than 32 bits, data shall be right-justified into the RHR field.

29.8.10 Transmit Holding Register

Name: THR

Access Type: Write-only

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
THR[31:24]							

23	22	21	20	19	18	17	16
THR[23:16]							

15	14	13	12	11	10	9	8
THR[15:8]							

7	6	5	4	3	2	1	0
THR[7:0]							

- **THR: Data Word to Be Transmitted**
  Next data word to be transmitted after the current word if TXRDY is not set. If MR.DATALENGTH specifies less than 32 bits, data shall be right-justified into the THR field.

29.8.11 Module Version

Name: VERSION

Access Type: Read-only

Offset: 0x28

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

29.8.12 Module Parameters

Name: PARAMETER

Access Type: Read-only

Offset: 0x2C

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

Reserved. No functionality associated.

29.9 Module Configuration

The specific configuration for each IISC instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 29-7. IISC Clocks

Clock Name	Description
CLK_IISC	Clock for the IISC bus interface
GCLK	The generic clock used for the IISC is GCLK6

Table 29-8. Register Reset Values

Register	Reset Value
VERSION	0x00000100

30. Timer/Counter (TC)

Rev: 4.0.2.0

30.1 Features

- **Three 16-bit Timer Counter channels**
- **A wide range of functions including:**
  - **Frequency measurement**
  - **Event counting**
  - **Interval measurement**
  - **Pulse generation**
  - **Delay timing**
  - **Pulse width modulation**
  - **Up/down capabilities**
  - **2-bit gray up/down count for stepper motor**
- **Each channel is user-configurable and contains:**
  - **Three external clock inputs**
  - **Five internal clock inputs**
  - **Two multi-purpose input/output signals**
- **Internal interrupt signal**
- **Two global registers that act on all three TC channels**
- **Configuration registers can be write protected**
- **Peripheral event input on all A/B lines in capture mode**

30.2 Overview

The Timer Counter (TC) includes three identical 16-bit Timer Counter channels.

Each channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing, and pulse width modulation.

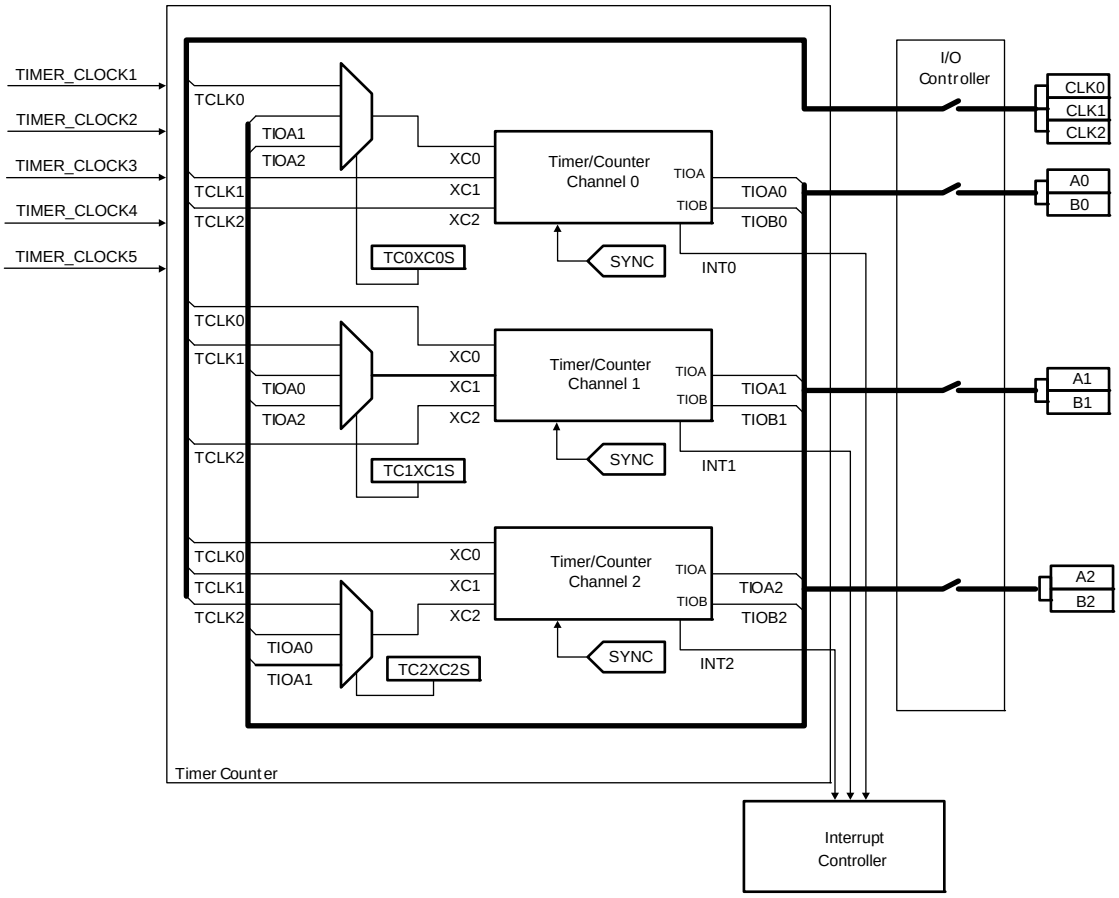
Each channel has three external clock inputs, five internal clock inputs, and two multi-purpose input/output signals which can be configured by the user. Each channel drives an internal interrupt signal which can be programmed to generate processor interrupts.

The TC block has two global registers which act upon all three TC channels.

The Block Control Register (BCR) allows the three channels to be started simultaneously with the same instruction.

The Block Mode Register (BMR) defines the external clock inputs for each channel, allowing them to be chained.

30.3 Block Diagram



30.4 I/O Lines Description

Table 30-1. I/O Lines Description

Pin Name	Description	Type
CLK0-CLK2	External Clock Input	Input
A0-A2	I/O Line A	Input/Output
B0-B2	I/O Line B	Input/Output

30.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

30.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with I/O lines. The user must first program the I/O Controller to assign the TC pins to their peripheral functions.

When using the TIOA/TIOB lines as inputs the user must make sure that no peripheral events are generated on the line. Refer to Section 31. “Peripheral Event Controller (PEVC)” on page 845 for details.

30.5.2 Power Management

If the CPU enters a sleep mode that disables clocks used by the TC, the TC will stop functioning and resume operation after the system wakes up from sleep mode.

30.5.3 Clocks

The clock for the TC bus interface (CLK_TC) is generated by the Power Manager. This clock is disabled at reset, and can be enabled in the Power Manager. It is recommended to disable the TC before disabling the clock, to avoid freezing the TC in an undefined state.

30.5.4 Interrupts

The TC interrupt request line is connected to the NVIC. Using the TC interrupt requires the NVIC to be programmed first.

30.5.5 Peripheral Events

The TC peripheral events are connected via the Peripheral Event System. Refer to Section 31. “Peripheral Event Controller (PEVC)” on page 845 for details.

30.5.6 Debug Operation

The Timer Counter clocks are frozen during debug operation, unless the OCD system keeps peripherals running in debug operation.

30.6 Functional Description

30.6.1 TC Description

The three channels of the Timer Counter are independent and identical in operation. The registers for channel programming are listed in Figure 30-3 on page 819.

30.6.1.1 Channel I/O Signals

As described in Figure 30.3 on page 803, each Channel has the following I/O signals.

Table 30-2. Channel I/O Signals Description

Block/Channel	Signal Name	Description
Channel Signal	XC0, XC1, XC2	External Clock Inputs
	TIOA	Capture mode: Timer Counter Input Waveform mode: Timer Counter Output
	TIOB	Capture mode: Timer Counter Input Waveform mode: Timer Counter Input/Output
	INT	Interrupt Signal Output
	SYNC	Synchronization Input Signal

30.6.1.2 16-bit counter

Each channel is organized around a 16-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value 0xFFFF and

passes to 0x0000, an overflow occurs and the Counter Overflow Status bit in the Channel n Status Register (SRn.COVFS) is set.

The current value of the counter is accessible in real time by reading the Channel n Counter Value Register (CVn). The counter can be reset by a trigger. In this case, the counter value passes to 0x0000 on the next valid edge of the selected clock.

30.6.1.3 Clock selection

At block level, input clock signals of each channel can either be connected to the external inputs TCLK0, TCLK1 or TCLK2, or be connected to the configurable I/O signals A0, A1 or A2 for chaining by writing to the BMR register. See Figure 30-1 on page 806.

Each channel can independently select an internal or external clock source for its counter:

- Internal clock signals: TIMER_CLOCK1, TIMER_CLOCK2, TIMER_CLOCK3, TIMER_CLOCK4, TIMER_CLOCK5. See the Module Configuration Chapter for details about the connection of these clock sources.
- External clock signals: XC0, XC1 or XC2. See the Module Configuration Chapter for details about the connection of these clock sources.

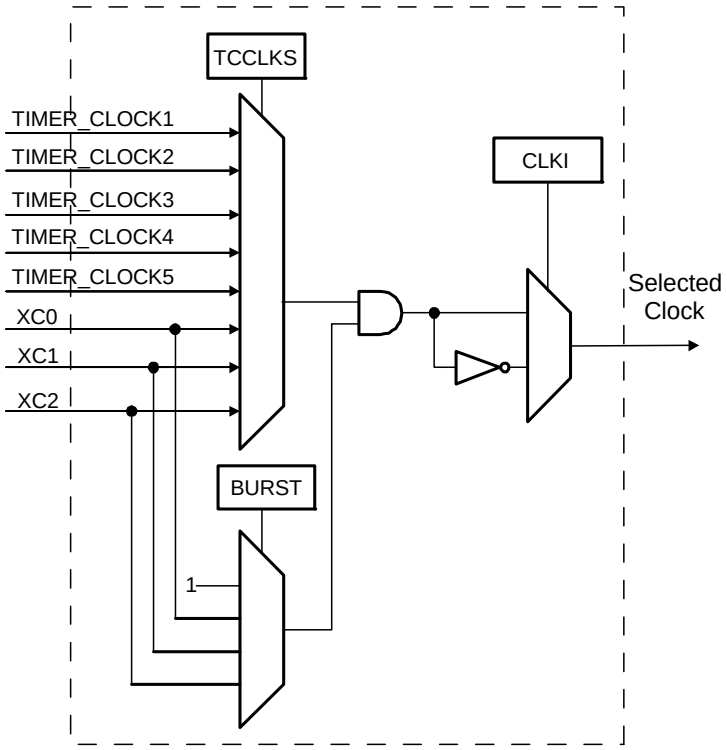
This selection is made by the Clock Selection field in the Channel n Mode Register (CMRn.TCCLKS).

The selected clock can be inverted with the Clock Invert bit in CMRn (CMRn.CLKI). This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The Burst Signal Selection field in the CMRn register (CMRn.BURST) defines this signal.

Note: In all cases, if an external clock is used, the duration of each of its levels must be longer than the CLK_TC period. The external clock frequency must be at least 2.5 times lower than the CLK_TC.

Figure 30-1. Clock Selection

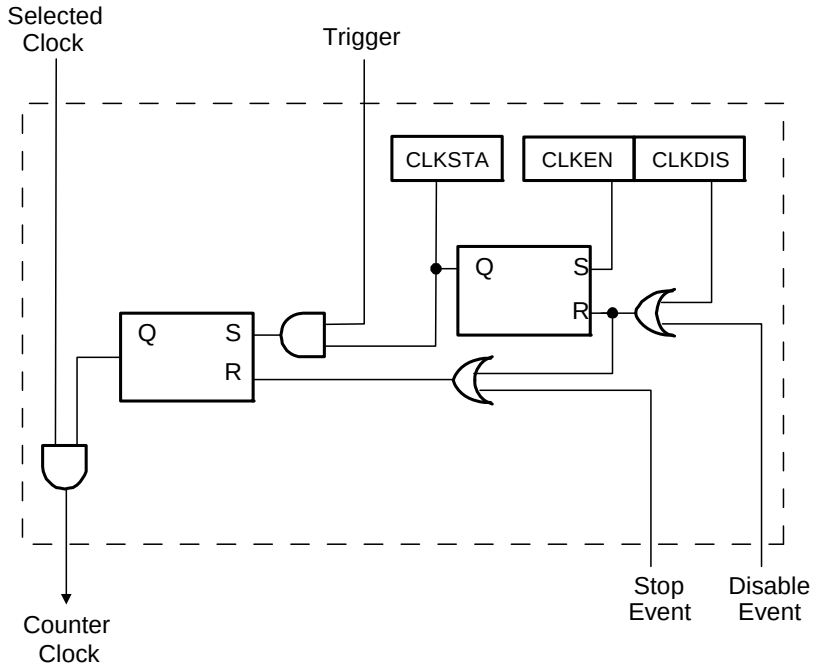


30.6.1.4 Clock control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped. See Figure 30-2 on page 807.

- The clock can be enabled or disabled by the user by writing to the Counter Clock Enable/Disable Command bits in the Channel n Clock Control Register (CCRn.CLKEN and CCRn.CLKDIS). In Capture mode it can be disabled by an RB load event if the Counter Clock Disable with RB Loading bit in CMRn is written to one (CMRn.LDBDIS). In Waveform mode, it can be disabled by an RC Compare event if the Counter Clock Disable with RC Compare bit in CMRn is written to one (CMRn.CPCDIS). When disabled, the start or the stop actions have no effect: only a CLKEN command in CCRn can re-enable the clock. When the clock is enabled, the Clock Enabling Status bit is set in SRn (SRn.CLKSTA).
- The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. In Capture mode the clock can be stopped by an RB load event if the Counter Clock Stopped with RB Loading bit in CMRn is written to one (CMRn.LDBSTOP). In Waveform mode it can be stopped by an RC compare event if the Counter Clock Stopped with RC Compare bit in CMRn is written to one (CMRn.CPCSTOP). The start and the stop commands have effect only if the clock is enabled.

Figure 30-2. Clock Control



30.6.1.5 TC operating modes

Each channel can independently operate in two different modes:

- Capture mode provides measurement on signals.
- Waveform mode provides wave generation.

The TC operating mode selection is done by writing to the Wave bit in the CCRn register (CCRn.WAVE).

In Capture mode, TIOA and TIOB are configured as inputs.

In Waveform mode, TIOA is always configured to be an output and TIOB is an output if it is not selected to be the external trigger.

30.6.1.6 Trigger

A trigger resets the counter and starts the counter clock. Three types of triggers are common to both modes, and a fourth external trigger is available to each mode.

The following triggers are common to both modes:

- Software Trigger: each channel has a software trigger, available by writing a one to the Software Trigger Command bit in CCRn (CCRn.SWTRG).
- SYNC: each channel has a synchronization signal SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing a one to the Synchro Command bit in the BCR register (BCR.SYNC).
- Compare RC Trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if the RC Compare Trigger Enable bit in CMRn (CMRn.CPCTRG) is written to one.

The channel can also be configured to have an external trigger. In Capture mode, the external trigger signal can be selected between TIOA and TIOB. In Waveform mode, an external event can be programmed to be one of the following signals: TIOB, XC0, XC1, or XC2. This external event can then be programmed to perform a trigger by writing a one to the External Event Trigger Enable bit in CMRn (CMRn.ENETRG).

If an external trigger is used, the duration of the pulses must be longer than the CLK_TC period in order to be detected.

Regardless of the trigger used, it will be taken into account at the following active edge of the selected clock. This means that the counter value can be read differently from zero just after a trigger, especially when a low frequency signal is selected as the clock.

30.6.1.7 Peripheral events on TIOA/TIOB inputs

The TIOA*/TIOB* input lines are ored internally with peripheral events from the Peripheral Event System. To capture using events the user must ensure that the corresponding pin functions for the TIOA*/TIOB* line are disabled. When capturing on the external TIOA*/TIOB* pin the user must ensure that no peripheral events are generated on this pin.

30.6.2 Capture Operating Mode

This mode is entered by writing a zero to the CMRn.WAVE bit.

Capture mode allows the TC channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOA and TIOB signals which are considered as inputs.

Figure 30-3 on page 809 shows the configuration of the TC channel when programmed in Capture mode.

30.6.2.1 Capture registers A and B

Registers A and B (RA and RB) are used as capture registers. This means that they can be loaded with the counter value when a programmable event occurs on the signal TIOA.

The RA Loading Selection field in CMRn (CMRn.LDRA) defines the TIOA edge for the loading of the RA register, and the RB Loading Selection field in CMRn (CMRn.LDRB) defines the TIOA edge for the loading of the RB register.

RA is loaded only if it has not been loaded since the last trigger or if RB has been loaded since the last loading of RA.

RB is loaded only if RA has been loaded since the last trigger or the last loading of RB.

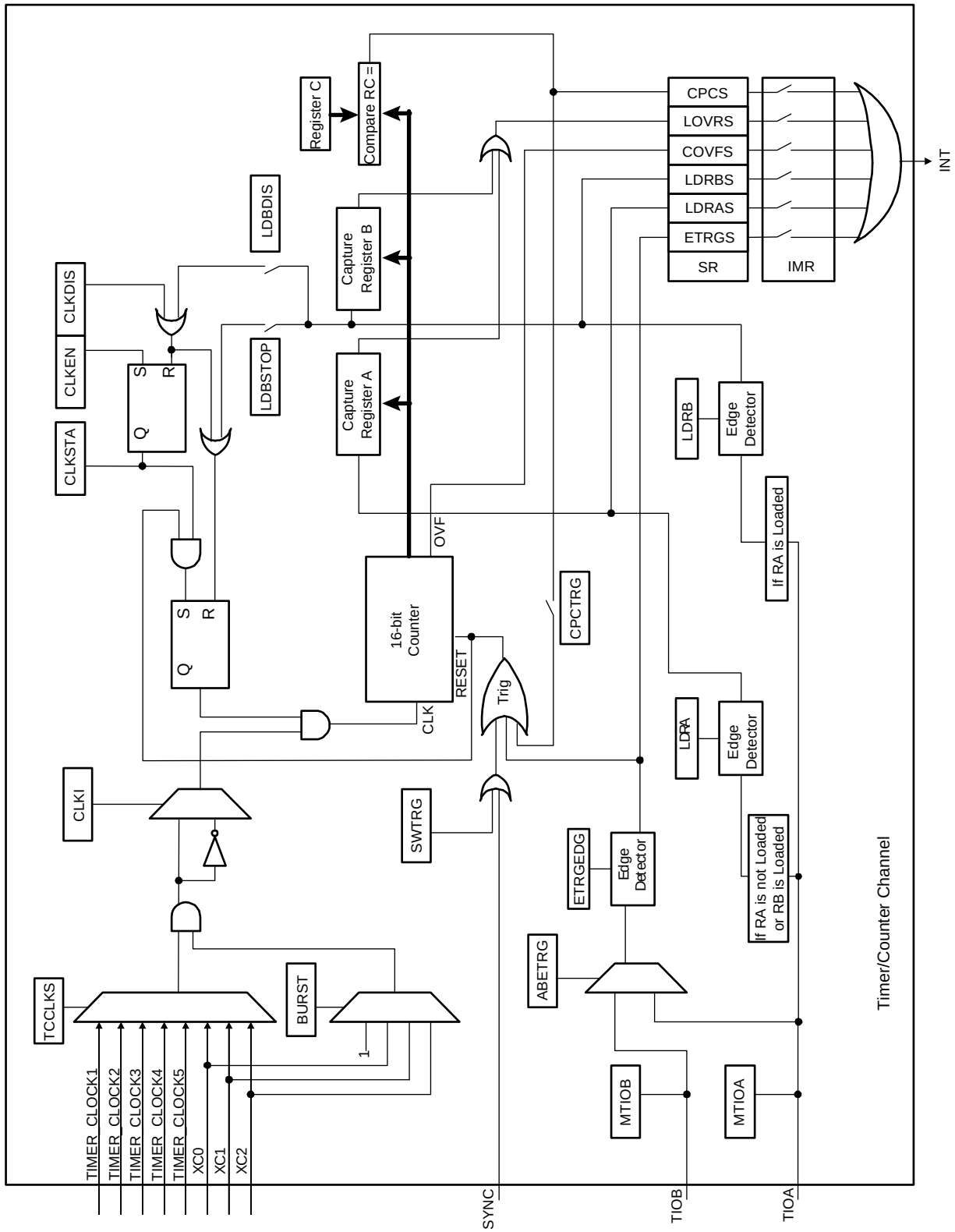
Loading RA or RB before the read of the last value loaded sets the Load Overrun Status bit in SRn (SRn.LOVRS). In this case, the old value is overwritten.

30.6.2.2 Trigger conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

The TIOA or TIOB External Trigger Selection bit in CMRn (CMRn.ABETRG) selects TIOA or TIOB input signal as an external trigger. The External Trigger Edge Selection bit in CMRn (CMRn.ETREDG) defines the edge (rising, falling or both) detected to generate an external trigger. If CMRn.ETRGEDG is zero (none), the external trigger is disabled.

Figure 30-3. Capture Mode



30.6.3 Waveform Operating Mode

Waveform operating mode is entered by writing a one to the CMRn.WAVE bit.

In Waveform operating mode the TC channel generates one or two PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOA is configured as an output and TIOB is defined as an output if it is not used as an external event.

Figure 30-4 on page 811 shows the configuration of the TC channel when programmed in Waveform operating mode.

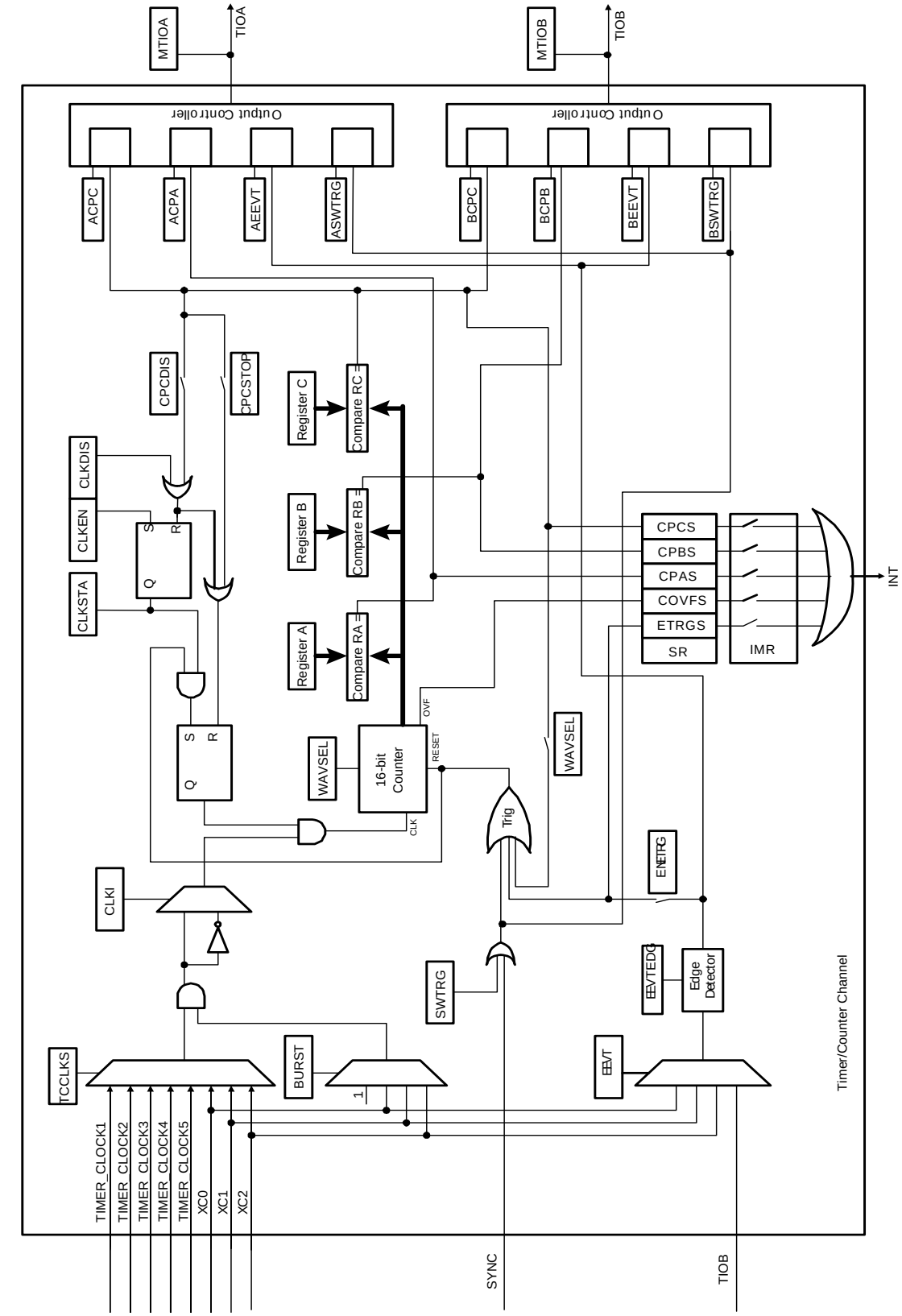
30.6.3.1 Waveform selection

Depending on the Waveform Selection field in CMRn (CMRn.WAVSEL), the behavior of CVn varies.

With any selection, RA, RB and RC can all be used as compare registers.

RA Compare is used to control the TIOA output, RB Compare is used to control the TIOB output (if correctly configured) and RC Compare is used to control TIOA and/or TIOB outputs.

Figure 30-4. Waveform Mode



30.6.3.2 WAVSEL = 0

When CMRn.WAVSEL is zero, the value of CVn is incremented from 0 to 0xFFFF. Once 0xFFFF has been reached, the value of CVn is reset. Incrementation of CVn starts again and the cycle continues. See Figure 30-5 on page 812.

An external event trigger or a software trigger can reset the value of CVn. It is important to note that the trigger may occur at any time. See Figure 30-6 on page 813.

RC Compare cannot be programmed to generate a trigger in this configuration. At the same time, RC Compare can stop the counter clock (CMRn.CPCSTOP = 1) and/or disable the counter clock (CMRn.CPCDIS = 1).

Figure 30-5. WAVSEL= 0 Without Trigger

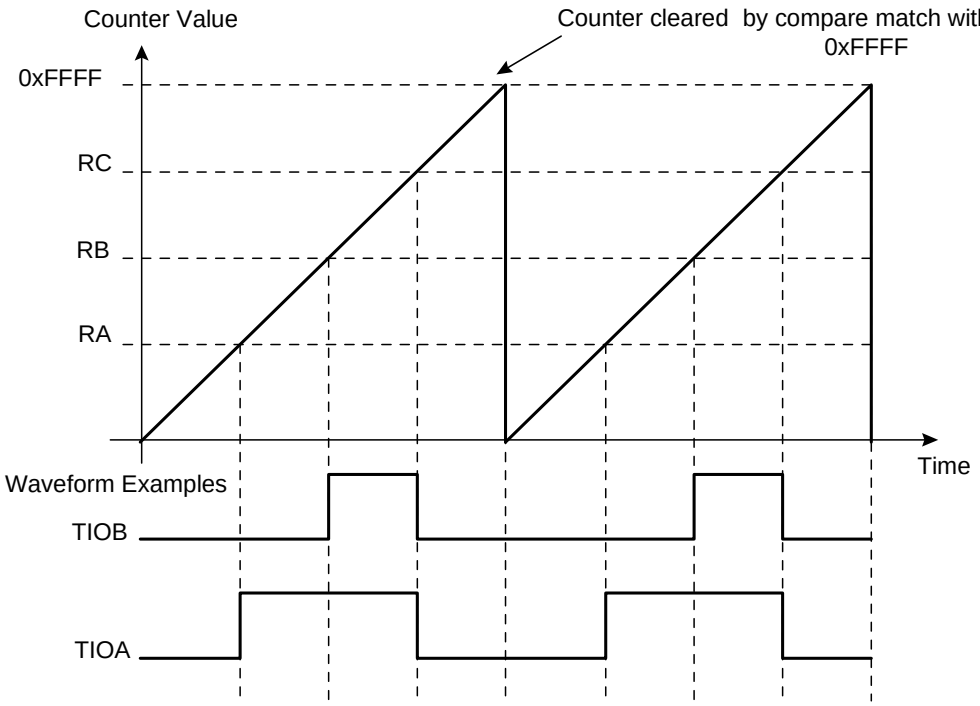
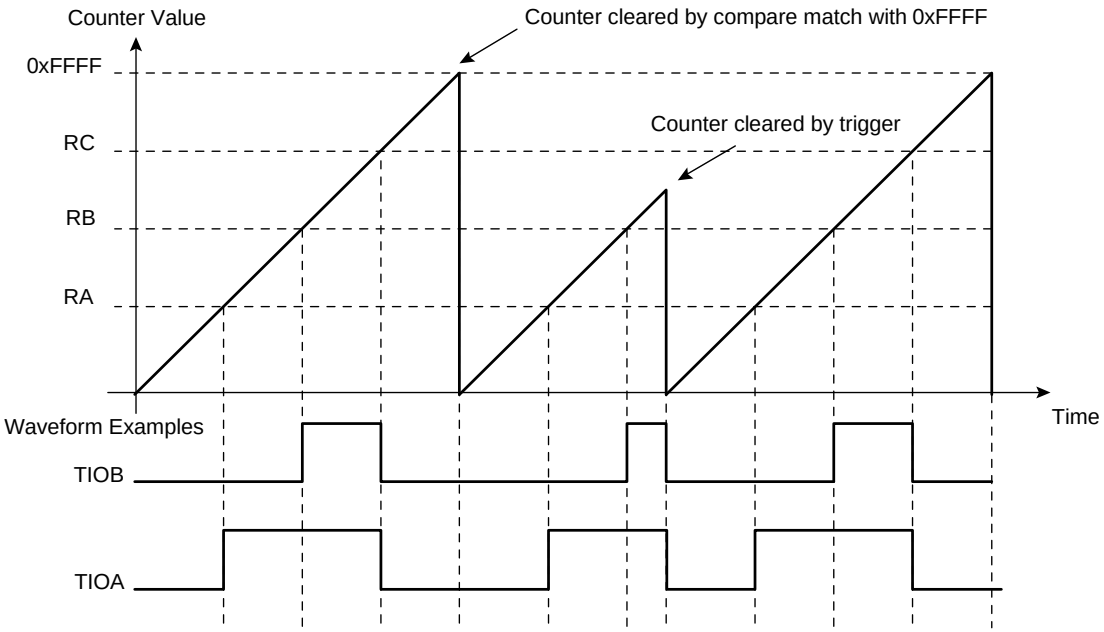


Figure 30-6. WAVSEL= 0 With Trigger



30.6.3.3 WAVSEL = 2

When CMRn.WAVSEL is two, the value of CVn is incremented from zero to the value of RC, then automatically reset on a RC Compare. Once the value of CVn has been reset, it is then incremented and so on. See Figure 30-7 on page 814.

It is important to note that CVn can be reset at any time by an external event or a software trigger if both are programmed correctly. See Figure 30-8 on page 814.

In addition, RC Compare can stop the counter clock (CMRn.CPCSTOP) and/or disable the counter clock (CMRn.CPCDIS = 1).

Figure 30-7. WAVSEL = 2 Without Trigger

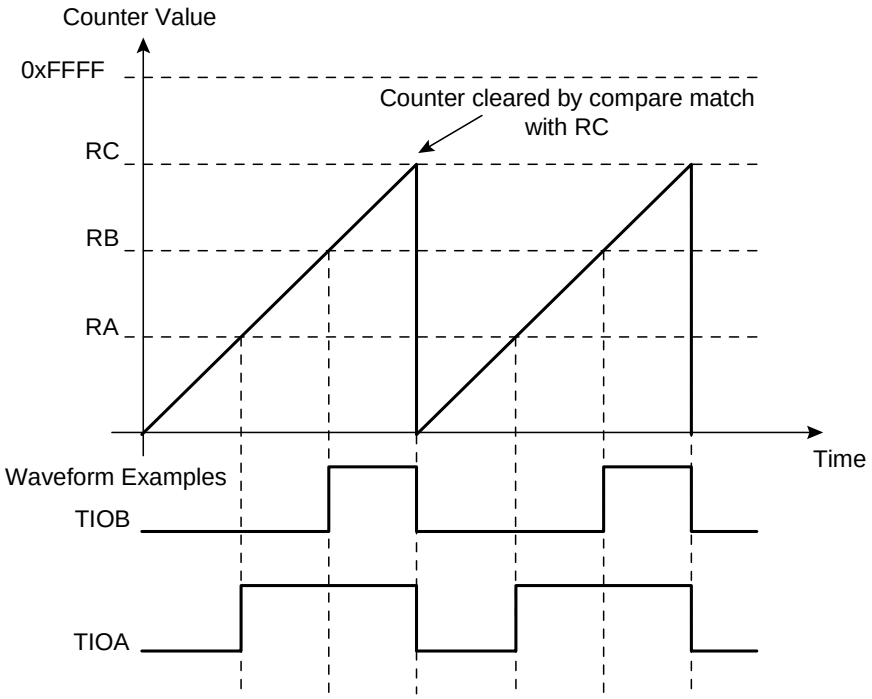
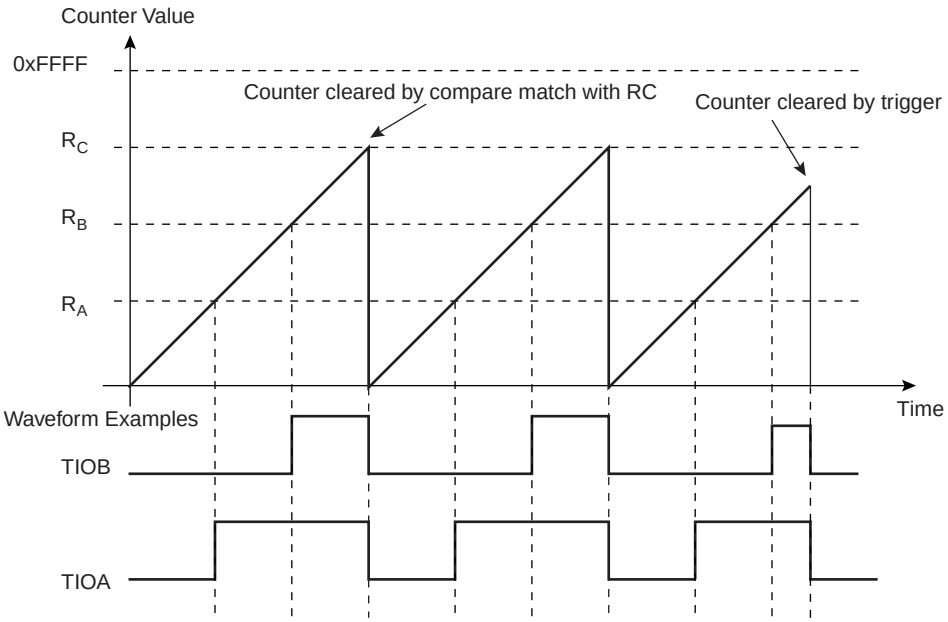


Figure 30-8. WAVSEL = 2 With Trigger



30.6.3.4 WAVSEL = 1

When CMRn.WAVSEL is one, the value of CVn is incremented from 0 to 0xFFFF. Once 0xFFFF is reached, the value of CVn is decremented to 0, then re-incremented to 0xFFFF and so on. See Figure 30-9 on page 815.

A trigger such as an external event or a software trigger can modify CVn at any time. If a trigger occurs while CVn is incrementing, CVn then decrements. If a trigger is received while CVn is decrementing, CVn then increments. See Figure 30-10 on page 815.

RC Compare cannot be programmed to generate a trigger in this configuration.

At the same time, RC Compare can stop the counter clock (CMRn.CPCSTOP = 1) and/or disable the counter clock (CMRn.CPCDIS = 1).

Figure 30-9. WAVSEL = 1 Without Trigger

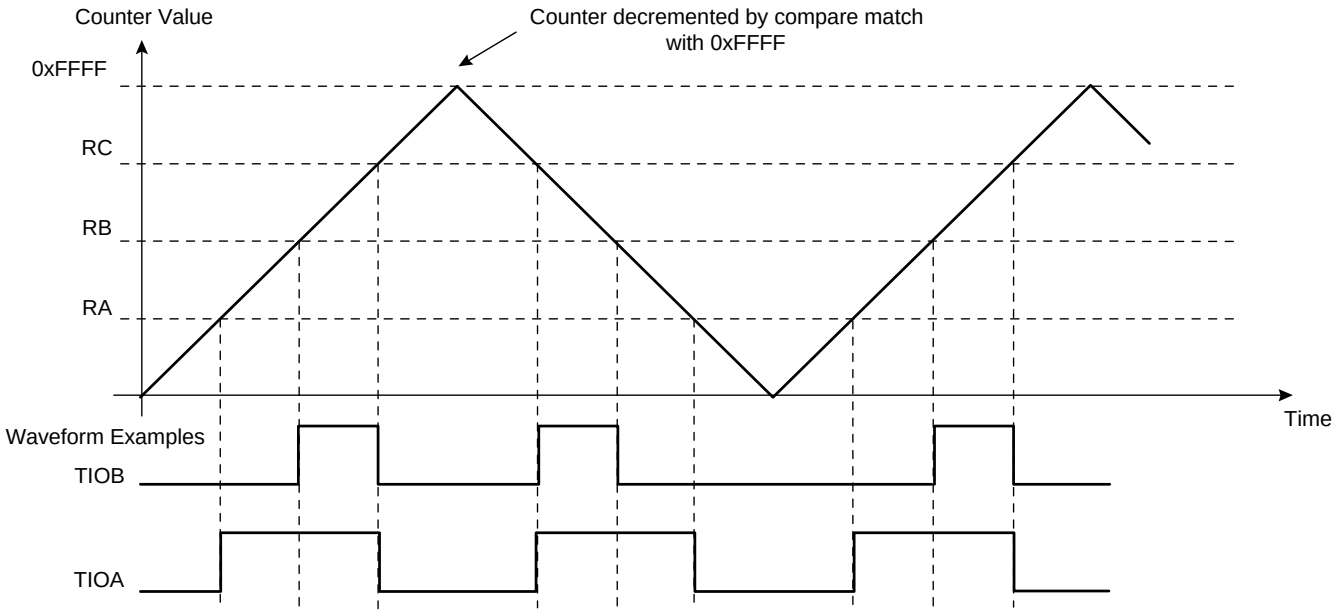
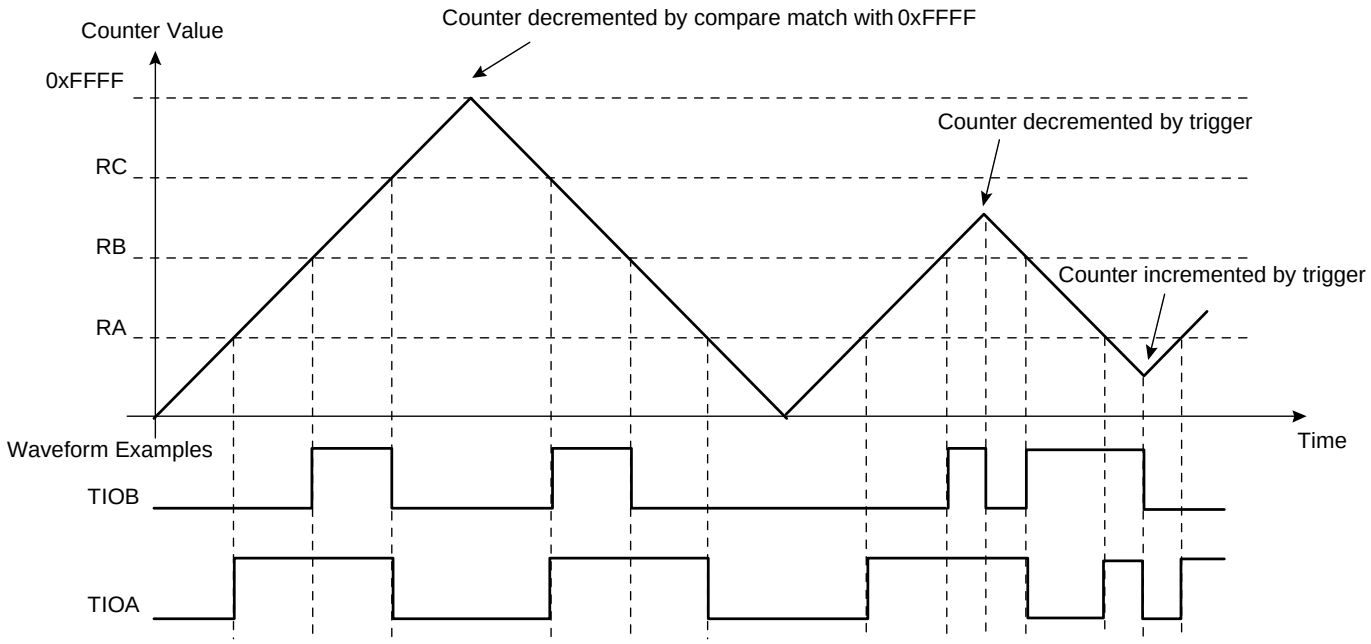


Figure 30-10. WAVSEL = 1 With Trigger



30.6.3.5 WAVSEL = 3

When CMRn.WAVSEL is three, the value of CVn is incremented from zero to RC. Once RC is reached, the value of CVn is decremented to zero, then re-incremented to RC and so on. See Figure 30-11 on page 816.

A trigger such as an external event or a software trigger can modify CVn at any time. If a trigger occurs while CVn is incrementing, CVn then decrements. If a trigger is received while CVn is decrementing, CVn then increments. See Figure 30-12 on page 817.

RC Compare can stop the counter clock (CMRn.CPCSTOP = 1) and/or disable the counter clock (CMRn.CPCDIS = 1).

Figure 30-11. WAVSEL = 3 Without Trigger

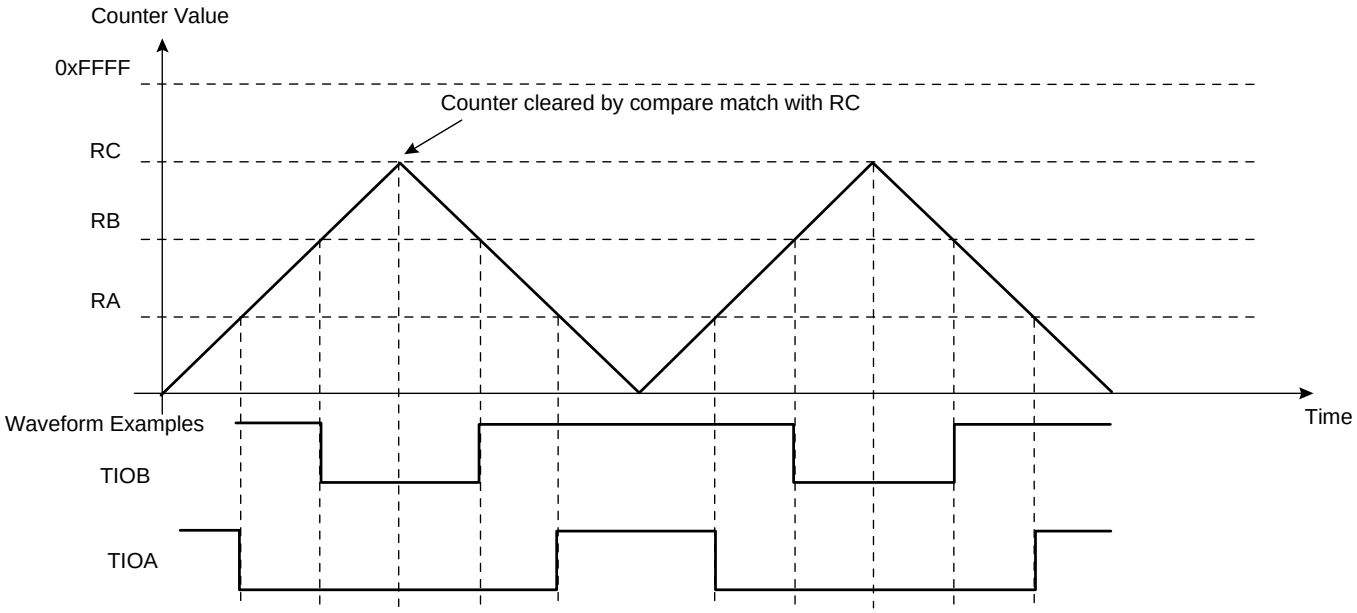
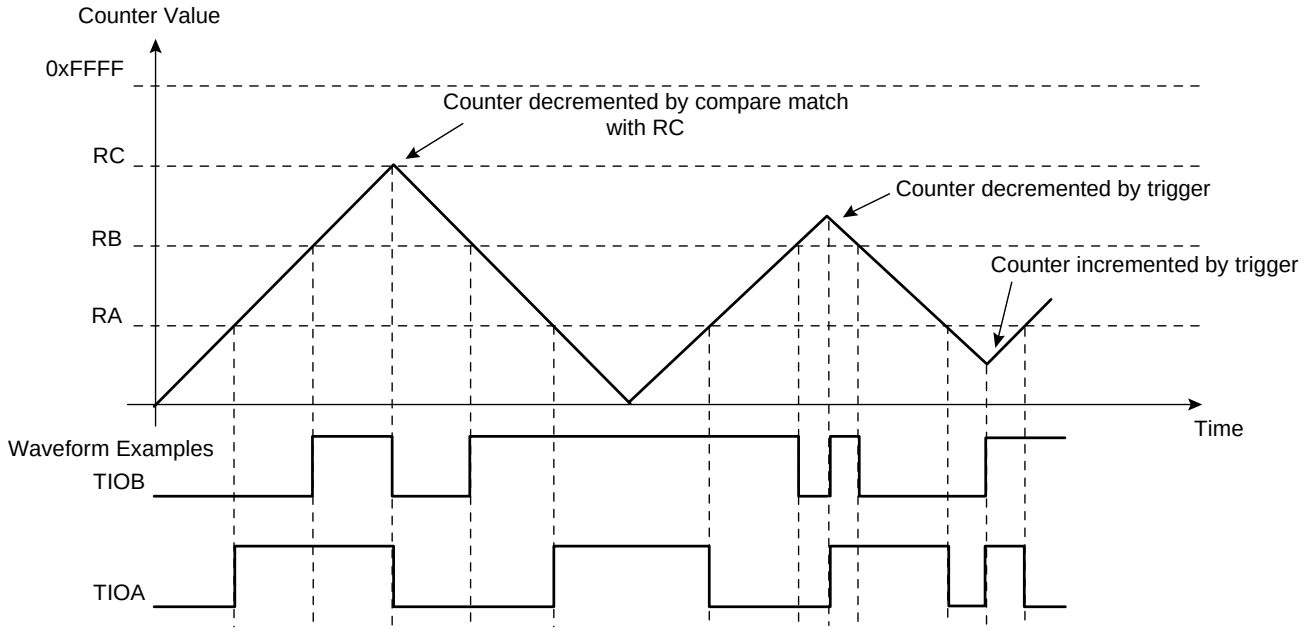


Figure 30-12. WAVSEL = 3 With Trigger



30.6.3.6 External event/trigger conditions

An external event can be programmed to be detected on one of the clock sources (XC0, XC1, XC2) or TIOB. The external event selected can then be used as a trigger.

The External Event Selection field in CMRn (CMRn.EEVT) selects the external trigger. The External Event Edge Selection field in CMRn (CMRn.EEVTEDG) defines the trigger edge for each of the possible external triggers (rising, falling or both). If CMRn.EEVTEDG is written to zero, no external event is defined.

If TIOB is defined as an external event signal (CMRn.EEVT = 0), TIOB is no longer used as an output and the compare register B is not used to generate waveforms and subsequently no IRQs. In this case the TC channel can only generate a waveform on TIOA.

When an external event is defined, it can be used as a trigger by writing a one to the CMRn.ENETRG bit.

As in Capture mode, the SYNC signal and the software trigger are also available as triggers. RC Compare can also be used as a trigger depending on the CMRn.WAVSEL field.

30.6.3.7 Output controller

The output controller defines the output level changes on TIOA and TIOB following an event. TIOB control is used only if TIOB is defined as output (not as an external event).

The following events control TIOA and TIOB:

- software trigger
- external event
- RC compare

RA compare controls TIOA and RB compare controls TIOB. Each of these events can be programmed to set, clear or toggle the output as defined in the following fields in CMRn:

- RC Compare Effect on TIOB (CMRn.BCPC)

- RB Compare Effect on TIOB (CMRn.BCPB)
- RC Compare Effect on TIOA (CMRn.ACPC)
- RA Compare Effect on TIOA (CMRn.ACPA)

30.7 2-bit Gray Up/Down Counter for Stepper Motor

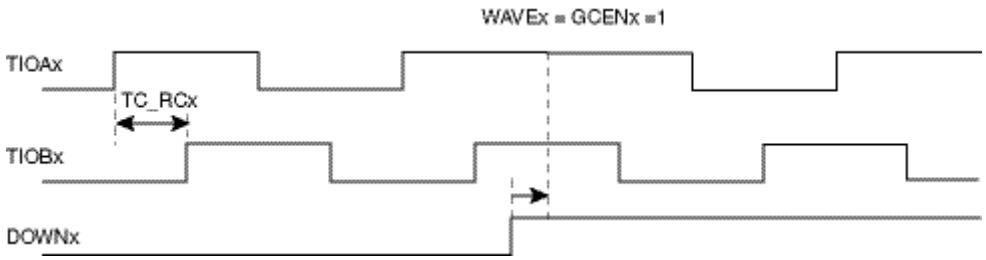
Each channel can be independently configured to generate a 2-bit gray count waveform on corresponding TIOA, TIOB outputs by means of GCEN bit in SMMRx registers.

Up or Down count can be defined by writing bit DOWN in SMMRx registers.

It is mandatory to configure the channel in WAVE mode in CMR register.

The period of the counters can be programmed on RCx registers.

Figure 30-13. 2-bit Gray Up/Down Counter.



30.8 Write Protection System

In order to bring security to the Timer Counter, a write protection system has been implemented.

The write protection mode prevent the write of BMR, FMR, CMRx, SMMRx, RAx, RBx, RCx registers. When this mode is enabled and one of the protected registers write, the register write request canceled.

Due to the nature of the write protection feature, enabling and disabling the write protection mode requires the use of a security code. Thus when enabling or disabling the write protection mode the WPKEY field of the WPMR register must be filled with the “TIM” ASCII code (corresponding to 0x54494D) otherwise the register write will be canceled.

30.9 User Interface

Table 30-3. TC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Channel 0 Control Register	CCR0	Write-only	0x00000000
0x04	Channel 0 Mode Register	CMR0	Read/Write	0x00000000
0x08	Ch 0 Stepper Motor Mode Register	SMMR0	Read/Write	0x00000000
0x10	Channel 0 Counter Value	CV0	Read-only	0x00000000
0x14	Channel 0 Register A	RA0	Read/Write(1)	0x00000000
0x18	Channel 0 Register B	RB0	Read/Write(1)	0x00000000
0x1C	Channel 0 Register C	RC0	Read/Write	0x00000000
0x20	Channel 0 Status Register	SR0	Read-only	0x00000000
0x24	Interrupt Enable Register	IER0	Write-only	0x00000000
0x28	Channel 0 Interrupt Disable Register	IDR0	Write-only	0x00000000
0x2C	Channel 0 Interrupt Mask Register	IMR0	Read-only	0x00000000
0x40	Channel 1 Control Register	CCR1	Write-only	0x00000000
0x44	Channel 1 Mode Register	CMR1	Read/Write	0x00000000
0x48	Ch 1 Stepper Motor Mode Register	SMMR1	Read/Write	0x00000000
0x50	Channel 1 Counter Value	CV1	Read-only	0x00000000
0x54	Channel 1 Register A	RA1	Read/Write(1)	0x00000000
0x58	Channel 1 Register B	RB1	Read/Write(1)	0x00000000
0x5C	Channel 1 Register C	RC1	Read/Write	0x00000000
0x60	Channel 1 Status Register	SR1	Read-only	0x00000000
0x64	Channel 1 Interrupt Enable Register	IER1	Write-only	0x00000000
0x68	Channel 1 Interrupt Disable Register	IDR1	Write-only	0x00000000
0x6C	Channel 1 Interrupt Mask Register	IMR1	Read-only	0x00000000
0x80	Channel 2 Control Register	CCR2	Write-only	0x00000000
0x84	Channel 2 Mode Register	CMR2	Read/Write	0x00000000
0x88	Ch 2 Stepper Motor Mode Register	SMMR2	Read/Write	0x00000000
0x90	Channel 2 Counter Value	CV2	Read-only	0x00000000
0x94	Channel 2 Register A	RA2	Read/Write(1)	0x00000000
0x98	Channel 2 Register B	RB2	Read/Write(1)	0x00000000
0x9C	Channel 2 Register C	RC2	Read/Write	0x00000000
0xA0	Channel 2 Status Register	SR2	Read-only	0x00000000
0xA4	Channel 2 Interrupt Enable Register	IER2	Write-only	0x00000000
0xA8	Channel 2 Interrupt Disable Register	IDR2	Write-only	0x00000000
0xAC	Channel 2 Interrupt Mask Register	IMR2	Read-only	0x00000000
0xC0	Block Control Register	BCR	Write-only	0x00000000
0xC4	Block Mode Register	BMR	Read/Write	0x00000000

Table 30-3. TC Register Memory Map

Offset	Register	Register Name	Access	Reset
0xE4	Write Protect Mode Register	WPMR	Read/Write	0x00000000
0xF8	Features Register	FEATURES	Read-only	-(2)
0xFC	Version Register	VERSION	Read-only	-(2)

Notes: 1. Read-only if CMRn.WAVE is zero.

2. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

30.9.1 Channel Control Register

Name: CCR

Access Type: Write-only

Offset: 0x00 + n * 0x40

Reset Value: 0x00000000

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

- **SWTRG: Software Trigger Command**
  1: Writing a one to this bit will perform a software trigger: the counter is reset and the clock is started.
  0: Writing a zero to this bit has no effect.
- **CLKDIS: Counter Clock Disable Command**
  1: Writing a one to this bit will disable the clock.
  0: Writing a zero to this bit has no effect.
- **CLKEN: Counter Clock Enable Command**
  1: Writing a one to this bit will enable the clock if CLKDIS is not one.
  0: Writing a zero to this bit has no effect.

30.9.2 Channel Mode Register: Capture Mode

Name: CMR

Access Type: Read/Write

Offset: 0x04 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	LDRB		LDRA	

15	14	13	12	11	10	9	8
WAVE	CPCTRG	-	-	-	ABETRG	ETRGEDG	

7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **LDRB: RB Loading Selection**

LDRB	Edge
0	none
1	rising edge of TIOA
2	falling edge of TIOA
3	each edge of TIOA

- **LDRA: RA Loading Selection**

LDRA	Edge
0	none
1	rising edge of TIOA
2	falling edge of TIOA
3	each edge of TIOA

- **WAVE**
  1: Capture mode is disabled (Waveform mode is enabled).
  0: Capture mode is enabled.
- **CPCTRG: RC Compare Trigger Enable**
  1: RC Compare resets the counter and starts the counter clock.
  0: RC Compare has no effect on the counter and its clock.

- **ABETRG: TIOA or TIOB External Trigger Selection**
  1: TIOA is used as an external trigger.
  0: TIOB is used as an external trigger.
- **ETRGEDG: External Trigger Edge Selection**

ETRGEDG	Edge
0	none
1	rising edge
2	falling edge
3	each edge

- **LDBDIS: Counter Clock Disable with RB Loading**
  1: Counter clock is disabled when RB loading occurs.
  0: Counter clock is not disabled when RB loading occurs.
- **LDBSTOP: Counter Clock Stopped with RB Loading**
  1: Counter clock is stopped when RB loading occurs.
  0: Counter clock is not stopped when RB loading occurs.
- **BURST: Burst Signal Selection**

BURST	Burst Signal Selection
0	The clock is not gated by an external signal
1	XC0 is ANDed with the selected clock
2	XC1 is ANDed with the selected clock
3	XC2 is ANDed with the selected clock

- **CLKI: Clock Invert**
  1: The counter is incremented on falling edge of the clock.
  0: The counter is incremented on rising edge of the clock.
- **TCCLKS: Clock Selection**

TCCLKS	Clock Selected
0	TIMER_CLOCK1
1	TIMER_CLOCK2
2	TIMER_CLOCK3
3	TIMER_CLOCK4
4	TIMER_CLOCK5
5	XC0
6	XC1
7	XC2

30.9.3 Channel Mode Register: Waveform Mode

Name: CMR

Access Type: Read/Write

Offset: 0x04 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
BSWTRG		BEEVT		BCPC		BCPB	

23	22	21	20	19	18	17	16
ASWTRG		AEEVT		ACPC		ACPA	

15	14	13	12	11	10	9	8
WAVE	WAVSEL		ENETRG	EEVT		EEVTEDG	

7	6	5	4	3	2	1	0
CPCDIS	CPCSTOP	BURST		CLKI	TCCLKS		

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **BSWTRG: Software Trigger Effect on TIOB**

BSWTRG	Effect
0	none
1	set
2	clear
3	toggle

- **BEEVT: External Event Effect on TIOB**

BEEVT	Effect
0	none
1	set
2	clear
3	toggle

- **BCPC: RC Compare Effect on TIOB**

BCPC	Effect
0	none
1	set
2	clear
3	toggle

- **BCPB: RB Compare Effect on TIOB**

BCPB	Effect
0	none
1	set
2	clear
3	toggle

- **ASWTRG: Software Trigger Effect on TIOA**

ASWTRG	Effect
0	none
1	set
2	clear
3	toggle

- **AEEVT: External Event Effect on TIOA**

AEEVT	Effect
0	none
1	set
2	clear
3	toggle

- **ACPC: RC Compare Effect on TIOA**

ACPC	Effect
0	none
1	set
2	clear
3	toggle

- **ACPA: RA Compare Effect on TIOA**

ACPA	Effect
0	none
1	set
2	clear
3	toggle

- **WAVE**
  1: Waveform mode is enabled.
  0: Waveform mode is disabled (Capture mode is enabled).
- **WAVSEL: Waveform Selection**

WAVSEL	Effect
0	UP mode without automatic trigger on RC Compare
1	UPDOWN mode without automatic trigger on RC Compare
2	UP mode with automatic trigger on RC Compare
3	UPDOWN mode with automatic trigger on RC Compare

- **ENETRG: External Event Trigger Enable**
  1: The external event resets the counter and starts the counter clock.
  0: The external event has no effect on the counter and its clock. In this case, the selected external event only controls the TIOA output.
- **EEVT: External Event Selection**

EEVT	Signal selected as external event	TIOB Direction
0	TIOB	input(1)
1	XC0	output
2	XC1	output
3	XC2	output

Note: 1. *If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms and subsequently no IRQs*.

- **EEVTEDG: External Event Edge Selection**

EEVTEDG	Edge
0	none
1	rising edge
2	falling edge
3	each edge

- **CPCDIS: Counter Clock Disable with RC Compare**
  1: Counter clock is disabled when counter reaches RC.
  0: Counter clock is not disabled when counter reaches RC.

- **CPCSTOP: Counter Clock Stopped with RC Compare**
  1: Counter clock is stopped when counter reaches RC.
  0: Counter clock is not stopped when counter reaches RC.
- **BURST: Burst Signal Selection**

BURST	Burst Signal Selection
0	The clock is not gated by an external signal.
1	XC0 is ANDed with the selected clock.
2	XC1 is ANDed with the selected clock.
3	XC2 is ANDed with the selected clock.

- **CLKI: Clock Invert**
  1: Counter is incremented on falling edge of the clock.
  0: Counter is incremented on rising edge of the clock.
- **TCCLKS: Clock Selection**

TCCLKS	Clock Selected
0	TIMER_CLOCK1
1	TIMER_CLOCK2
2	TIMER_CLOCK3
3	TIMER_CLOCK4
4	TIMER_CLOCK5
5	XC0
6	XC1
7	XC2

30.9.4 Stepper Motor Mode Register

Name: SMCR

Access Type: Read/Write

Offset: 0x08 + n * 0x40

Reset Value: 0x00000000

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

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **DOWN: Down Count**
  0: Up counter.
  1: Down counter.
- **GCEN: Gray Count Enable**
  0: TIOAx and TIOBx are driven by internal counter of channel x.
  1: TIOAx and TIOBx are driven by a 2-bit gray counter.

30.9.5 Channel Counter Value Register

Name: CV

Access Type: Read-only

Offset: 0x10 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
CV[15:8]							

7	6	5	4	3	2	1	0
CV[7:0]							

- **CV: Counter Value**
  CV contains the counter value in real time.

30.9.6 Channel Register A

Name: RA

Access Type: Read-only if CMRn.WAVE = 0, Read/Write if CMRn.WAVE = 1

Offset: 0x14 + n * 0X40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
RA[15:8]							

7	6	5	4	3	2	1	0
RA[7:0]							

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **RA: Register A**
  RA contains the Register A value in real time.

30.9.7 Channel Register B

Name: RB

Access Type: Read-only if CMRn.WAVE = 0, Read/Write if CMRn.WAVE = 1

Offset: 0x18 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
RB[15:8]							
7	6	5	4	3	2	1	0
RB[7:0]							

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **RB: Register B**
  RB contains the Register B value in real time.

30.9.8 Channel Register C

Name: RC

Access Type: Read/Write

Offset: 0x1C + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
RC[15:8]							

7	6	5	4	3	2	1	0
RC[7:0]							

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **RC: Register C**
  RC contains the Register C value in real time.

30.9.9 Channel Status Register

Name: SR

Access Type: Read-only

Offset: 0x20 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	MTIOB	MTIOA	CLKSTA
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

Note: Reading the Status Register will also clear the interrupt bit for the corresponding interrupts.

- **MTIOB: TIOB Mirror**
  1: TIOB is high. If CMRn.WAVE is zero, this means that TIOB pin is high. If CMRn.WAVE is one, this means that TIOB is driven high.
  0: TIOB is low. If CMRn.WAVE is zero, this means that TIOB pin is low. If CMRn.WAVE is one, this means that TIOB is driven low.
- **MTIOA: TIOA Mirror**
  1: TIOA is high. If CMRn.WAVE is zero, this means that TIOA pin is high. If CMRn.WAVE is one, this means that TIOA is driven high.
  0: TIOA is low. If CMRn.WAVE is zero, this means that TIOA pin is low. If CMRn.WAVE is one, this means that TIOA is driven low.
- **CLKSTA: Clock Enabling Status**
  1: This bit is set when the clock is enabled.
  0: This bit is cleared when the clock is disabled.
- **ETRGS: External Trigger Status**
  1: This bit is set when an external trigger has occurred.
  0: This bit is cleared when the SR register is read.
- **LDRBS: RB Loading Status**
  1: This bit is set when an RB Load has occurred and CMRn.WAVE is zero.
  0: This bit is cleared when the SR register is read.
- **LDRAS: RA Loading Status**
  1: This bit is set when an RA Load has occurred and CMRn.WAVE is zero.
  0: This bit is cleared when the SR register is read.
- **CPCS: RC Compare Status**
  1: This bit is set when an RC Compare has occurred.
  0: This bit is cleared when the SR register is read.

- **CPBS: RB Compare Status**
  1: This bit is set when an RB Compare has occurred and CMRn.WAVE is one.
  0: This bit is cleared when the SR register is read.
- **CPAS: RA Compare Status**
  1: This bit is set when an RA Compare has occurred and CMRn.WAVE is one.
  0: This bit is cleared when the SR register is read.
- **LOVRS: Load Overrun Status**
  1: This bit is set when RA or RB have been loaded at least twice without any read of the corresponding register and CMRn.WAVE is zero.
  0: This bit is cleared when the SR register is read.
- **COVFS: Counter Overflow Status**
  1: This bit is set when a counter overflow has occurred.
  0: This bit is cleared when the SR register is read.

30.9.10 Channel Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x24 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

30.9.11 Channel Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x28 + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

30.9.12 Channel Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x2C + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

30.9.13 Block Control Register

Name: BCR

Access Type: Write-only

Offset: 0xC0

Reset Value: 0x00000000

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

- **SYNC: Synchro Command**
  1: Writing a one to this bit asserts the SYNC signal which generates a software trigger simultaneously for each of the channels.
  0: Writing a zero to this bit has no effect.

30.9.14 Block Mode Register

Name: BMR

Access Type: Read/Write

Offset: 0xC4

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	TC2XC2S		TC1XC1S		TC0XC0S	

This register can only be written if write protect is disabled (WPMR.WPEN is zero).

- **TC2XC2S: External Clock Signal 2 Selection**

TC2XC2S	Signal Connected to XC2
0	TCLK2
1	none
2	TIOA0
3	TIOA1

- **TC1XC1S: External Clock Signal 1 Selection**

TC1XC1S	Signal Connected to XC1
0	TCLK1
1	none
2	TIOA0
3	TIOA2

- **TC0XC0S: External Clock Signal 0 Selection**

TC0XC0S	Signal Connected to XC0
0	TCLK0
1	none
2	TIOA1
3	TIOA2

30.9.15 Write Protect Mode Register

Name: WPMR

Access Type: Read/write

Offset: 0xE4

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	WPEN

- **WPKEY: Write Protect Key**
  Valid key is “TIM” in ASCII (0x54494D in hexadecimal).
- **WPEN: Write Protect Enable**
  1: Writing a one to this bit will enable write protection (WPKEY must be set).
  0: Writing a zero to this bit will disable write protection (WPKEY must be set).
  Protected registers:
  Channel Mode Register: Capture Mode
  Channel Mode Register: Waveform Mode
  Stepper Motor Mode Register
  Register A
  Register B
  Register C
  Block Mode Register

30.9.16 Features Register

Name: FEATURES

Access Type: Read-only

Offset: 0xF8

Reset Value: -

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

- **BRPBHSB: Bridge type is PB to HSB**
  1: Bridge type is PB to HSB.
  0: Bridge type is not PB to HSB.
- **UPDNIMPL: Up/down is implemented**
  1: Up/down counter capability is implemented.
  0: Up/down counter capability is not implemented.
- **CTRSIZE: Counter size**
  This field indicates the size of the counter in bits.

30.9.17 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

30.10 Module Configuration

The specific configuration for each Timer/Counter instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 30-4. TC Bus Interface Clocks

Module name	Clock Name	Description
TC0	CLK_TC0	Clock for the TC0 bus interface
TC1	CLK_TC1	Clock for the TC1 bus interface

Each Timer/Counter channel can independently select an internal or external clock source for its counter:

Table 30-5. Timer/Counter Clock Connections

Module	Source	Name	Connection
TC0	Internal	TIMER_CLOCK1	Generic Clock number 5
		TIMER_CLOCK2	PBA Clock / 2
		TIMER_CLOCK3	PBA Clock / 8
		TIMER_CLOCK4	PBA Clock / 32
		TIMER_CLOCK5	PBA Clock / 128
	External	XC0	PA14, PB13
		XC1	PA15, PB14
		XC2	PA16, PB15
TC1	Internal	TIMER_CLOCK1	Generic Clock number 8
		TIMER_CLOCK2	PBA Clock / 2
		TIMER_CLOCK3	PBA Clock / 8
		TIMER_CLOCK4	PBA Clock / 32
		TIMER_CLOCK5	PBA Clock / 128
	External	XC0	PC06, PC21
		XC1	PC07, PC22
		XC2	PC08, PC23

Table 30-6. Register Reset Values

Register	Reset Value
VERSION	0x00000402

31. Peripheral Event Controller (PEVC)

Rev: 2.0.0.0

31.1 Features

- **Direct peripheral to peripheral communication system**
- **Allows peripherals to receive, react to, and send peripheral events without CPU intervention**
- **Cycle deterministic event communication**
- **SleepWalking™ and asynchronous interrupts for peripheral operation in Power Save Modes**

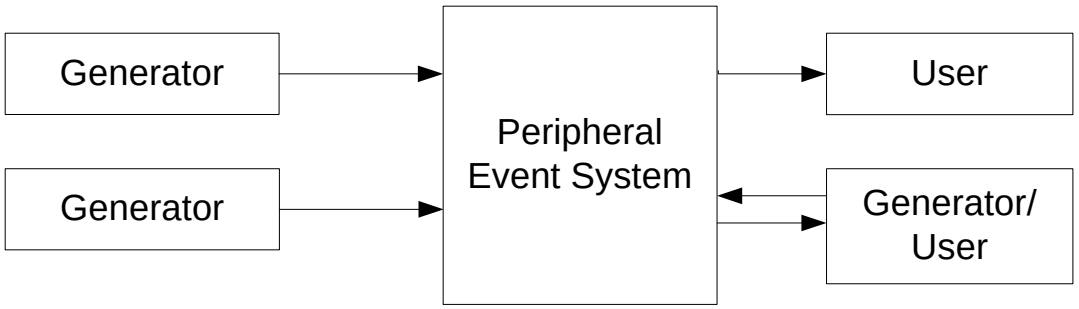
31.2 Overview

Several peripheral modules can be configured to emit or respond to signals known as peripheral events. The exact condition to trigger a peripheral event, or the action taken upon receiving a peripheral event, is specific to each module. Peripherals that respond to events are called users and peripherals that emit events are called generators. A module may be both a generator and user.

The peripheral event generators and users are interconnected by a network known as the Peripheral Event System. The Peripheral Event Controller (PEVC) controls the interconnection parameters, such as generator-to-user multiplexing and peripheral event enable/disable.

The Peripheral Event System allows low latency peripheral-to-peripheral signalling without CPU intervention, and without consuming system resources such as bus or RAM bandwidth. This offloads the CPU and system resources compared to a traditional interrupt-based software driven system.

Figure 31-1. Peripheral Event System Overview



31.3 Block Diagram

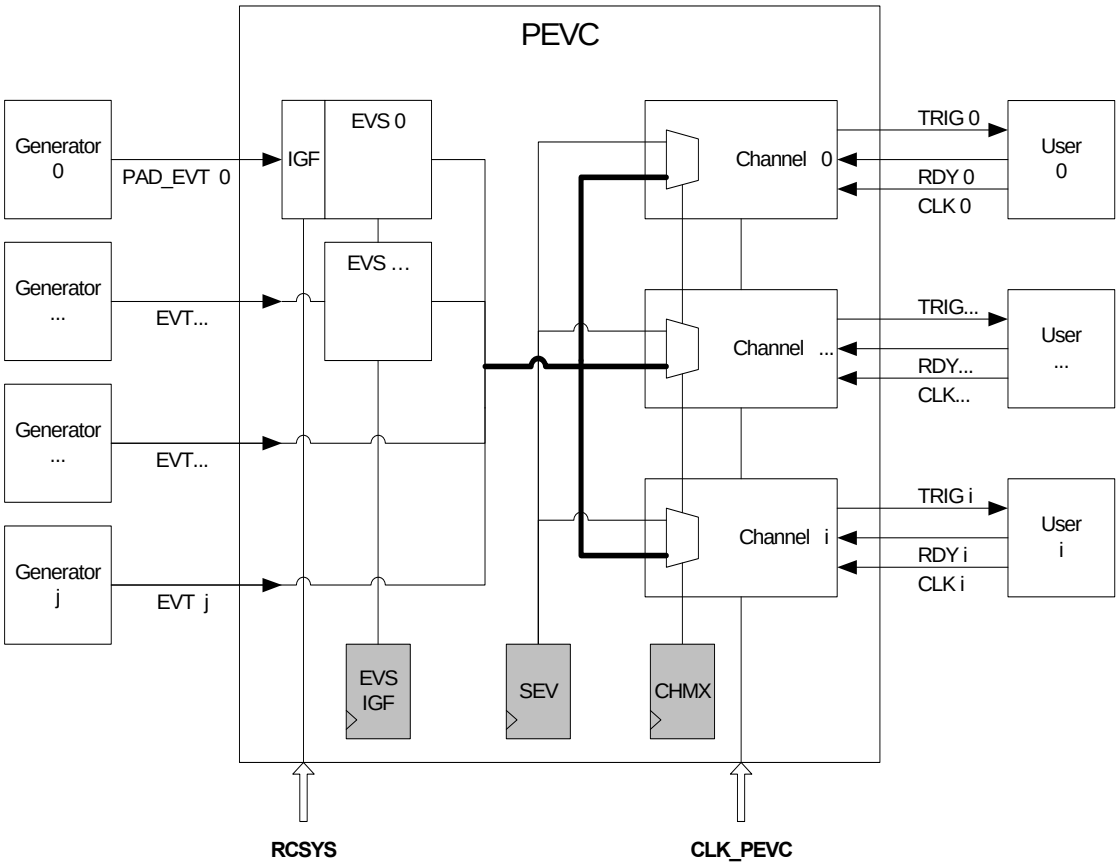
The main building blocks of the PEVC are:

- Channels: One channel per user, to propagate events and follow-up the user status
- Event Shapers (EVS): Instantiated for some generators, in case synchronisation and/or edge detection is needed prior to peripheral event propagation
- Input Glitch Filters (IGF): Present specifically for I/O inputs, to filter the incoming signal prior to going through EVS and Channel

To help distinguish the different signalling stages, following naming conventions are used:

- Generators generate events
- PEVC multiplexes these incoming events
- PEVC outputs triggers to users

Figure 31-2. PEVC Block Diagram



The maximum number of generators and Event Shapers supported by the PEVC is 64.

The maximum number of channels and users supported by the PEVC is 32.

EVS and IGF implementation are device-specific.

Refer to the Module Configuration section at the end of this chapter for the device-specific configuration.

31.4 I/O Lines Description

Table 31-1. I/O Lines Description

Pin Name	Pin Description	Type
PAD_EVT[n]	External Event Inputs	Input

31.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

31.5.1 I/O Lines

Multiplexed I/O lines can be used as event generators. To generate a peripheral event from an external source the source pin must be configured as an input pin by the I/O Controller. It is also possible to trigger a peripheral event by driving these pins from registers in the I/O Controller, or another peripheral output connected to the same pin.

31.5.2 Power Management and Low Power Operation

As the peripheral events do not require CPU intervention, they are available in Idle mode. They are also available in deeper Power Save Modes if both the generator and user remain clocked in that mode.

In deeper Power Save Modes, certain events can be issued even when the system clock is stopped, and revive unclocked user peripherals. The clock will be restarted for this module only, without waking the system from Power Save Mode. The clock remains active only as long as required by the triggered function, before being switched off again, and the system remains in the original sleep mode. The CPU and system will only be woken up if the user peripheral generates an interrupt as a result of the operation. This concept is known as SleepWalking™ and is described in further detail in Section 10. ”Power Manager (PM)” on page 109. Note that asynchronous peripheral events may be associated with a delay due to the need to restart the system clock source if this has been stopped in the sleep mode.

Refer to the Module Configuration section at the end of this chapter for the device-specific configuration, especially on which peripheral event generators and users can rely on SleepWalking™.

The table and diagram below represent the timed behavior of SleepWalking™ operations:

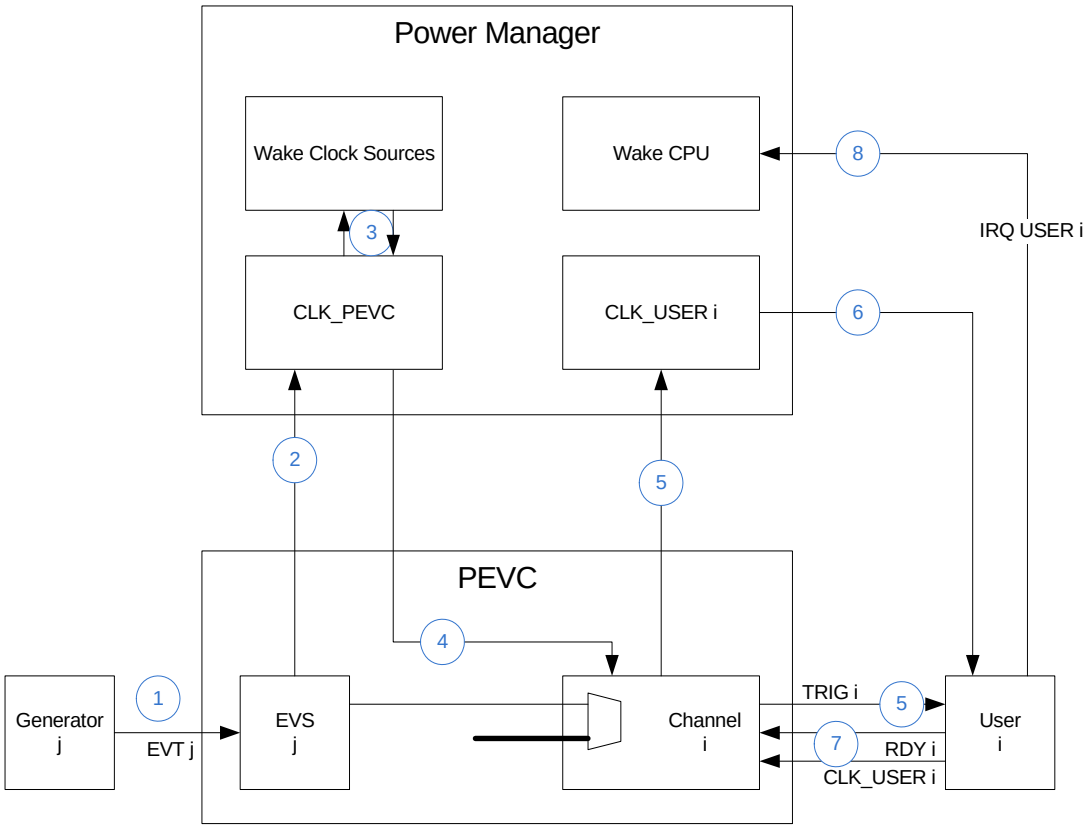
Table 31-2. SleepWalking™ on Peripheral Events

Phase	Operation
1	while in a Power Save Mode where peripheral clocks are asleep, asynchronous Generator j sends Peripheral Event to PEVC
2	incoming Event is detected by EVS j , and CLK_PEVC requested to PM
3	upon reception of clock request, PM wakes clock sources if required
4	CLK_PEVC wakes up

Table 31-2. SleepWalking™ on Peripheral Events

Phase	Operation
5	incoming Event j propagates to Channel i: • CLK_USER i request to PM • TRIG i forwarded to User i
6	CLK_USER i wakes up
7	User i handles the incoming Peripheral Event and signals it to PEVC
8	depending on the Peripheral's operation, a CPU wake request occurs in case of Interrupt processing
9	alternatively, the Peripheral may simply go back to sleep, in which case PM will silently shut off CLK_USER i and CLK_PEVC

Figure 31-3. SleepWalking™ on Peripheral Events



31.5.3 Clocks

The PEVC has two clocks connected: one Peripheral Bus clock (CLK_PEVC) and the system RC oscillator clock (CLK_RCSYS). These clocks are generated by the Power Manager.

CLK_PEVC is required for event propagation and Peripheral Bus operations.

CLK_RCSYS is used for glitch filtering in Event Shapers. It is required for event propagation in case glitch filtering is turned on for a given generator.

31.5.4 Interrupts

PEVC can generate an interrupt request in case of trigger generation or trigger overrun. The PEVC interrupt request lines are connected to the NVIC. Using the PEVC interrupts requires the NVIC to be programmed first.

31.5.5 Debug Operation

PEVC is not frozen during debug operation when the Core is halted, unless the bit corresponding to the PEVC is set in the Peripheral Debug Register (PDBG). Refer to the On-Chip Debug chapter for details.

31.6 Functional Description

31.6.1 PEVC Channel Operation

PEVC routes incoming events to users by means of one channel per user. Channels operate in parallel, allowing multiple users to listen to the same generator.

31.6.1.1 Channel Setup

The Channel Multiplexer Register (CHMXi) is written to allocate a generator to a given channel. The Event Multiplexer field (EVMX) selects between the different generators, while the Software Event Multiplexer bit (SMX) selects Software Events.

The channel is then enabled by writing a one to the appropriate bit in the Channel Enable Register (CHER). It is disabled by writing a one to the appropriate bit in the Channel Disable Register (CHDR).

To safely configure a channel, user software must:

- disable the channel by writing a one to CHDR
- configure CHMXi
- enable the channel by writing a one to CHER

31.6.1.2 Channel Operation

When the channel is enabled, the user signals its busy/ready state to the channel, to determine how an incoming event will be handled:

- If the user is ready, an incoming event is forwarded. The corresponding Trigger Status Register (TRISR) bit is set to one allowing an interrupt to be generated for tracking PEVC operations.
- If the user is busy (because of a previous event, or for some other cause), the new event is not forwarded. The corresponding Overrun Status Register (OVSR) flag is set allowing an interrupt to be generated.

The Busy Register (BUSY) is used to determine the current activity of a channel/user. A busy status has one of two causes:

- A peripheral event is being relayed by the channel and handled by the user,
- No event relayed, but user is not ready (e.g. not initialized, or handling some other request).

31.6.1.3 Software Event

A Software Event can be initiated by writing to the Software Event Register (SEV). This is intended for application debugging.

The channel must first be configured by writing a one to the Software Event Multiplexer bit (SMX) of CHMXi.

Writing a one to the appropriate bit of SEV will then trigger a Software Event on the channel.

31.6.2 Event Shaper (EVS) Operation

PEVC contains Event Shapers (EVS) for certain types of generators:

- External inputs
- General-purpose waveforms like timer outputs or Generic Clocks

Refer to the Module Configuration section at the end of this chapter for the device-specific configuration of Event Shapers and Input Glitch Filters.

Each Event Shaper is responsible of shaping one input, prior to going through a PEVC channel:

- Optionally apply input-glich-filtering
- Synchronize incoming events
- Request CLK_PEVC when in deeper sleep modes

Write a one to the EN field of the corresponding Event Shaper Register (EVS) to enable operation.

31.6.2.1 Input Glitch Filter (IGF)

Input Glitch Filtering can be turned on or off by writing to the Input Glitch Filter fields (IGFR and IGFF) of the corresponding Event Shaper Register (EVS).

When IGF is on, the incoming event is sampled periodically. The sampling clock is divided from CLK_RCSYS by the value of the Input Glitch Filter Divider Register (IGFDR). IGF will filter out spikes and propagate only incoming events that respect one of the following two conditions:

- rise event: 2 samples low, followed by 0+ changes, followed by 2 samples high
- fall event: 2 samples high, followed by 0+ changes, followed by 2 samples low

CLK_RCSYS must be enabled to use Input Glitch Filtering.

31.6.3 Event Propagation Latency

Once a channel is setup, incoming peripheral events are relayed by hardware. Event progation latency is therefore cycle deterministic. However, its value depends on the exact settings that apply to a given channel.

When the channel multiplexer CHMXi.EVMX selects a generator without Event Shaper, event propagation latency is 0 cycle. Software event is a particular case of 0 cycle propagation.

When the channel multiplexer CHMXi.EVMX selects a generator with Event Shaper, event propagation latency depends on Input Glitch Filter setting EVSj.IGF and SleepWalking™:

- IGF off: event propagation latency is less or equal to 2 CLK_PEVC cycles
- IGF on: event propagation latency is less or equal to $3 * 2^{IGFDR} *$ CLK_RCSYS cycles
- in case of SleepWalking™ operation, an additional delay incurs if the PM needs to revive the clocks sources. In that particular case of Low-Power operation, propagation latency is longer and cannot be predicted.

Table 31-3. Event Propagation Latency

Generator CHMXi.EVMX	Input Glitch Filter EVSj.IGF	Latency	Clock
Software event	-	0	-
Generator without Event Shaper	-	0	-
Generator with Event Shaper	Off	2	CLK_PEVC
Generator with Event Shaper	On	$3 * 2^{IGFDR}$	CLK_RCSYS
Asynchronous Generator in SleepWalking™ operation	On or Off	unpredicted	-

Refer to the Module Configuration section at the end of this chapter for the list of generators implementing Event Shapers.

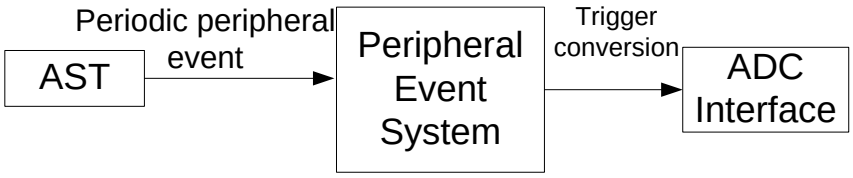
31.7 Application Example

This application example shows how the Peripheral Event System can be used to program the ADC Interface to perform ADC conversions at selected intervals.

One of the possible ADC conversion trigger is a peripheral event trigger, allowing the Peripheral Event System to synchronize conversion with some configured peripheral event source. One particular peripheral event source can be an AST peripheral event, among other types of peripheral events. The Peripheral Event System can then be used to set up the ADC Interface to sample an analog signal at regular intervals generated by the AST.

The user must enable peripheral events in the AST and in the ADC Interface. Refer to the corresponding chapters for how this is accomplished. Next, the AST will generate peripheral events periodically, and the Peripheral Event System will route the peripheral events to the ADC Interface, which will perform ADC conversions at the selected intervals.

Figure 31-4. Application Example



Since the AST peripheral event is an asynchronous event, the description above will also work in Power Save Modes where the ADC clock is stopped. In this case, the ADC clock (and clock source, if needed) will be restarted during the ADC conversion. After the conversion, the ADC clock and clock source will return to the sleep state, unless the ADC generates an interrupt, which in turn will wake up the system. Using asynchronous events thus allows ADC operation in much lower power states than would otherwise be possible.

31.8 User Interface

Table 31-4. PEVC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x000	Channel Status Register	CHSR	Read-only	0x00000000
0x004	Channel Enable Register	CHER	Write-only	-
0x008	Channel Disable Register	CHDR	Write-only	-
0x010	Software Event	SEV	Write-only	-
0x014	Channel / User Busy	BUSY	Read-only	-(1)
0x020	Trigger Interrupt Mask Enable Register	TRIER	Write-only	-
0x024	Trigger Interrupt Mask Disable Register	TRIDR	Write-only	-
0x028	Trigger Interrupt Mask Register	TRIMR	Read-only	0x00000000
0x030	Trigger Status Register	TRSR	Read-only	0x00000000
0x034	Trigger Status Clear Register	TRSCR	Write-only	-
0x040	Overrun Interrupt Mask Enable Register	OVIER	Write-only	-
0x044	Overrun Interrupt Mask Disable Register	OVIDR	Write-only	-
0x048	Overrun Interrupt Mask Register	OVIMR	Read-only	0x00000000
0x050	Overrun Status Register	OVSR	Read-only	0x00000000
0x054	Overrun Status Clear Register	OVSCR	Write-only	-
0x100	Channel Multiplexer 0	CHMX0	Read/Write	0x00000000
0x100 + i*4	Channel Multiplexer i	CHMXi	Read/Write	0x00000000
0x17C	Channel Multiplexer 31	CHMX31	Read/Write	0x00000000
0x200	Event Shaper 0	EVS0	Read/Write	0x00000000
0x200 + j*4	Event Shaper j	EVSj	Read/Write	0x00000000
0x2FC	Event Shaper 63	EVS63	Read/Write	0x00000000
0x300	Input Glitch Filter Divider Register	IGFDR	Read/Write	0x00000000
0x3F8	Parameter	PARAMETER	Read-only	-(1)
0x3FC	Version	VERSION	Read-only	-(1)

Notes: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

31.8.1 Channel Status Register

Name: CHSR

Access Type: Read-only

Offset: 0x000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
CHS							

23	22	21	20	19	18	17	16
CHS							

15	14	13	12	11	10	9	8
CHS							

7	6	5	4	3	2	1	0
CHS							

- **CHS: Channel Status**
  0: The corresponding channel is disabled.
  1: The corresponding channel is enabled.
  This bit is cleared when the corresponding bit in CHDR is written to one.
  This bit is set when the corresponding bit in CHER is written to one.

31.8.2 Channel Enable Register

Name: CHER

Access Type: Write-only

Offset: 0x004

Reset Value: -

31	30	29	28	27	26	25	24
CHE							

23	22	21	20	19	18	17	16
CHE							

15	14	13	12	11	10	9	8
CHE							

7	6	5	4	3	2	1	0
CHE							

- **CHE: Channel Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the corresponding bit in CHSR.

31.8.3 Channel Disable Register

Name: CHDR

Access Type: Write-only

Offset: 0x008

Reset Value: -

31	30	29	28	27	26	25	24
CHD							

23	22	21	20	19	18	17	16
CHD							

15	14	13	12	11	10	9	8
CHD							

7	6	5	4	3	2	1	0
CHD							

- **CHD: Channel Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in CHSR.

31.8.4 Software Event Register

Name: SEV

Access Type: Write-only

Offset: 0x010

Reset Value: -

31	30	29	28	27	26	25	24
SEV							

23	22	21	20	19	18	17	16
SEV							

15	14	13	12	11	10	9	8
SEV							

7	6	5	4	3	2	1	0
SEV							

- **SEV: Software Event**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will trigger a Software Event for the corresponding channel.

31.8.5 Channel / User Busy

Name: BUSY

Access Type: Read-only

Offset: 0x014

Reset Value: -

31	30	29	28	27	26	25	24
BUSY							

23	22	21	20	19	18	17	16
BUSY							

15	14	13	12	11	10	9	8
BUSY							

7	6	5	4	3	2	1	0
BUSY							

- **BUSY: Channel Status**
  0: The corresponding channel and user are idle.
  1: The corresponding channel and user are busy.

31.8.6 Trigger Interrupt Enable Register

Name: TRIER

Access Type: Write-only

Offset: 0x020

Reset Value: -

31	30	29	28	27	26	25	24
TRIE							

23	22	21	20	19	18	17	16
TRIE							

15	14	13	12	11	10	9	8
TRIE							

7	6	5	4	3	2	1	0
TRIE							

- **TRIE: Trigger Interrupt Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the corresponding bit in TRIMR.

31.8.7 Trigger Interrupt Disable Register

Name: TRIDR

Access Type: Write-only

Offset: 0x024

Reset Value: -

31	30	29	28	27	26	25	24
TRID							

23	22	21	20	19	18	17	16
TRID							

15	14	13	12	11	10	9	8
TRID							

7	6	5	4	3	2	1	0
TRID							

- **TRID: Trigger Interrupt Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in IMR.

31.8.8 Trigger Interrupt Mask Register

Name: TRIMR

Access Type: Read-only

Offset: 0x028

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
TRIM							

23	22	21	20	19	18	17	16
TRIM							

15	14	13	12	11	10	9	8
TRIM							

7	6	5	4	3	2	1	0
TRIM							

- **TRIM: Trigger Interrupt Mask**
  0: The corresponding interrupt is disabled.
  1: The corresponding interrupt is enabled.
  This bit is cleared when the corresponding bit in TRIDR is written to one.
  This bit is set when the corresponding bit in TRIER is written to one.

31.8.9 Trigger Status Register

Name: TRSR

Access Type: Read-only

Offset: 0x030

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
TRS							

23	22	21	20	19	18	17	16
TRS							

15	14	13	12	11	10	9	8
TRS							

7	6	5	4	3	2	1	0
TRS							

- **TRS: Trigger Interrupt Status**
  0: An interrupt event has not occurred
  1: An interrupt event has occurred
  This bit is cleared by writing a one to the corresponding bit in TRSCR.

31.8.10 Trigger Status Clear Register

Name: TRSCR

Access Type: Write-only

Offset: 0x034

Reset Value: -

31	30	29	28	27	26	25	24
TRSC							

23	22	21	20	19	18	17	16
TRSC							

15	14	13	12	11	10	9	8
TRSC							

7	6	5	4	3	2	1	0
TRSC							

- **TRSC: Trigger Interrupt Status Clear**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in TRSR.

31.8.11 Overrun Interrupt Enable Register

Name: OVIER

Access Type: Write-only

Offset: 0x040

Reset Value: -

31	30	29	28	27	26	25	24
OVIE							

23	22	21	20	19	18	17	16
OVIE							

15	14	13	12	11	10	9	8
OVIE							

7	6	5	4	3	2	1	0
OVIE							

- **OVIE: Overrun Interrupt Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the corresponding bit in OVIMR.

31.8.12 Overrun Interrupt Disable Register

Name: OVIDR

Access Type: Write-only

Offset: 0x044

Reset Value: -

31	30	29	28	27	26	25	24
OVID							

23	22	21	20	19	18	17	16
OVID							

15	14	13	12	11	10	9	8
OVID							

7	6	5	4	3	2	1	0
OVID							

- **OVID: Overrun Interrupt Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in IMR.

31.8.13 Overrun Interrupt Mask Register

Name: OVIMR

Access Type: Read-only

Offset: 0x048

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
OVIM							

23	22	21	20	19	18	17	16
OVIM							

15	14	13	12	11	10	9	8
OVIM							

7	6	5	4	3	2	1	0
OVIM							

- **OVIM: Overrun Interrupt Mask**
  0: The corresponding interrupt is disabled.
  1: The corresponding interrupt is enabled.
  This bit is cleared when the corresponding bit in OVIDR is written to one.
  This bit is set when the corresponding bit in OVIER is written to one.

31.8.14 Overrun Status Register

Name: OVSR

Access Type: Read-only

Offset: 0x050

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
OVS							

23	22	21	20	19	18	17	16
OVS							

15	14	13	12	11	10	9	8
OVS							

7	6	5	4	3	2	1	0
OVS							

- **OVS: Overrun Interrupt Status**
  0: An interrupt event has not occurred
  1: An interrupt event has occurred
  This bit is cleared by writing a one to the corresponding bit in OVSCR.

31.8.15 Overrun Status Clear Register

Name: OVSCR

Access Type: Write-only

Offset: 0x054

Reset Value: -

31	30	29	28	27	26	25	24
OVSC							

23	22	21	20	19	18	17	16
OVSC							

15	14	13	12	11	10	9	8
OVSC							

7	6	5	4	3	2	1	0
OVSC							

- **OVSC: Overrun Interrupt Status Clear**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in OVSR.

31.8.16 Channel Multiplexer Register

Name: CHMXi

Access Type: Read/Write

Offset: 0x100 + i*0x004

Reset Value: 0x00000000

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

- **SMX: Software Event Multiplexer**
  0: The Software Event is not selected. Event / generator is selected by EVMX.
  1: The Software Event is selected. EVMX is not considered.
- **EVMX: Event Multiplexer**
  Select input event / generator.

SMX	EVMX	Channel Input
1	Any	Software Event
0	0x00	EVT0
0	0x01	EVT1
0	0xj	EVTj
0	> TRIGOUT	None

31.8.17 Event Shaper Register

Name: EVSj

Access Type: Read/Write

Offset: 0x200 + j*0x004

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	IGFON	IGFF	IGFR
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	EN

- **IGFON: Input Glitch Filter Status**
  0: Input Glitch Filter is off.
  1: Input Glitch Filter is on.
  Only present when IGF is used. Refer to the Module Configuration section at the end of this chapter.
  To enable Input Glitch Filtering, EN bit must be set to 1, as well as one of IGFF or IGFR.
  This bit is Read-only, and can be used to detect that configuration is effective. It incurs some delay compared to the writing of IGFF, IGFR, or EN.
- **IGFF: Input Glitch Filter Fall**
  0: No event detection on falling edge.
  1: Event detection through Input Glitch Fiilter on falling edge.
  Only present when IGF is used. Refer to the Module Configuration section at the end of this chapter.
  To enable Input Glitch Filtering, EN bit must also be set to 1.
  Both IGFF and IGFR can be combined.
- **IGFR: Input Glitch Filter Rise**
  0: No event detection on rising edge.
  1: Event detection through Input Glitch Fiilter on rising edge.
  Only present when IGF is used. Refer to the Module Configuration section at the end of this chapter.
  To enable Input Glitch Filtering, EN bit must also be set to 1.
  Both IGFF and IGFR can be combined.
- **EN: Event Shaper Enable**
  0: Event Shaper is off.
  1: Event Shaper is on.
  To enable Input Glitch Filtering, IGFR and/or IGFF must also be set to 1.

31.8.18 Input Glitch Filter Divider Register

Name: IGFDR

Access Type: Read/Write

Offset: 0x300

Reset Value: 0x00000000

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

- **IGFDR: Input Glitch Filter Divider**
  Selects prescaler division ratio for the system RC clock used for glitch filtering.

IGFDR	Division Ratio
0x0	1
0x1	2
0x2	4
0xn	2^n
0xF	32768

31.8.19 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0x3F8

Reset Value: -

31	30	29	28	27	26	25	24
TRIGOUT							

23	22	21	20	19	18	17	16
EVIN							

15	14	13	12	11	10	9	8
EVS_COUNT							

7	6	5	4	3	2	1	0
IGF_COUNT							

- **TRIGOUT: Number of Trigger Outputs / Channels / Users**
  Number of trigger outputs / channels implemented. No functionality associated.
- **EVIN: Number of Event Inputs / Generators**
  Number of event inputs. No functionality associated.
- **EVS_COUNT: Number of Event Shapers**
  Number of Event Shapers implemented. No functionality associated.
- **IGF_COUNT: Number of Input Glitch Filters**
  Number of Input Glitch Filters implemented. No functionality associated.

31.8.20 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x3FC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Variant number of the module. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

31.9 Module Configuration

The specific configuration for each PEVC instance is listed in the following tables.

The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 31-5. Module Clock Name

Module name	Clock name
PEVC	CLK_PEVC

The module Register Reset Values are listed below.

Table 31-6. Register Reset Values

Register	Reset Value
VERSION	0x00000200
PARAMETER	0x131F1204
BUSY	0x0002401F

The following table defines generators and input events connected to the Peripheral Event System. It also specifies whether Event Shaper and Input Glitch Filter are implemented for this generator, and if SleepWalking is available.

Table 31-7. Generators

CHMXn.EVMX	Generator - input event	IGF	EVS	SleepWalking
0	PAD_EVT 0 - change on input pin	Yes	Yes	Yes
1	PAD_EVT 1 - change on input pin	Yes	Yes	Yes
2	PAD_EVT 2 - change on input pin	Yes	Yes	Yes
3	PAD_EVT 3 - change on input pin	Yes	Yes	Yes
4	GCLK 8 - rising edge		Yes	
5	GCLK 9 - rising edge		Yes	
6	AST - alarm event 0		Yes	Yes
7	Reserved			
8	AST - periodic event 0		Yes	Yes
9	Reserved			
10	AST - overflow event		Yes	Yes
11	ACIFC - AC0 VINP>VINN			
12	ACIFC - AC1 VINP>VINN			
13	ACIFC - AC2 VINP>VINN			
14	ACIFC - AC3 VINP>VINN			
15	ACIFC - AC0 VINP<VINN			
16	ACIFC - AC1 VINP<VINN			
17	ACIFC - AC2 VINP<VINN			
18	ACIFC - AC3 VINP<VINN			

Table 31-7. Generators

CHMXn.EVMX	Generator - input event	IGF	EVS	SleepWalking
19	ACIFC - AC0-AC1 window			
20	ACIFC - AC2-AC3 window			
21	TC0 - A0 waveform mode rising edge		Yes	
22	TC0 - A1 waveform mode rising edge		Yes	
23	TC0 - A2 waveform mode rising edge		Yes	
24	TC0 - B0 waveform mode rising edge		Yes	
25	TC0 - B1 waveform mode rising edge		Yes	
26	TC0 - B2 waveform mode rising edge		Yes	
27	ADC - window match			
28	ADC - end of conversion			
29	VREGIFG - stop switching ready			
30	PICOUART - character reception		Yes	Yes

The following table defines users connected to the Peripheral Event System, and their corresponding triggered action. It also specifies wheter SleepWalking is available for each User.

Table 31-8. Users

Channel Number	User - trigerred action	SleepWalking
0	PDCA - channel 0 transfer one word	
1	PDCA - channel 1transfer one word	
2	PDCA - channel 2 transfer one word	
3	PDCA - channel 3 transfer one word	
4	ADC - start one conversion	Yes
5	DAC - start one conversion	
6	CATB - trigger one autonomous touch sensing	Yes
7	Reserved	
8	TC1 - A0 capture	
9	TC1 - A1 capture	
10	TC1 - A2 capture	
11	TC1 - B0 capture	
12	TC1 - B1 capture	
13	TC1 - B2 capture	
14	ACIFC - trigger one comparison	Yes
15	PARC - parrallel capture start	
16	PARC - parrallel capture stop	
17	VREGIFG - stop switching request	Yes
18	VREGIFG - stop switching disable	Yes

32. Audio Bit Stream DAC (ABDACB)

Rev: 1.0.0.0

32.1 Features

- **16 bit digital stereo DAC**
- **Oversampling D/A conversion architecture**
  - **Adjustable oversampling ratio**
  - **3rd order Sigma-Delta D/A converters**
- **Digital bitstream output**
- **Parallel interface**
- **Connects to DMA for background transfer without CPU intervention**
- **Supported sampling frequencies**
  - **8000 Hz, 11025 Hz, 12000 Hz, 16000 Hz, 22050 Hz, 24000 Hz, 32000 Hz, 44100 Hz, and 48000 Hz**
- **Supported data formats**
  - **32-, 24-, 20-, 18-, 16-, and 8-bit stereo format**
  - **16- and 8-bit compact stereo format, with left and right sample packed in the same word to reduce data transfers**
- **Common mode offset control**
- **Volume control**

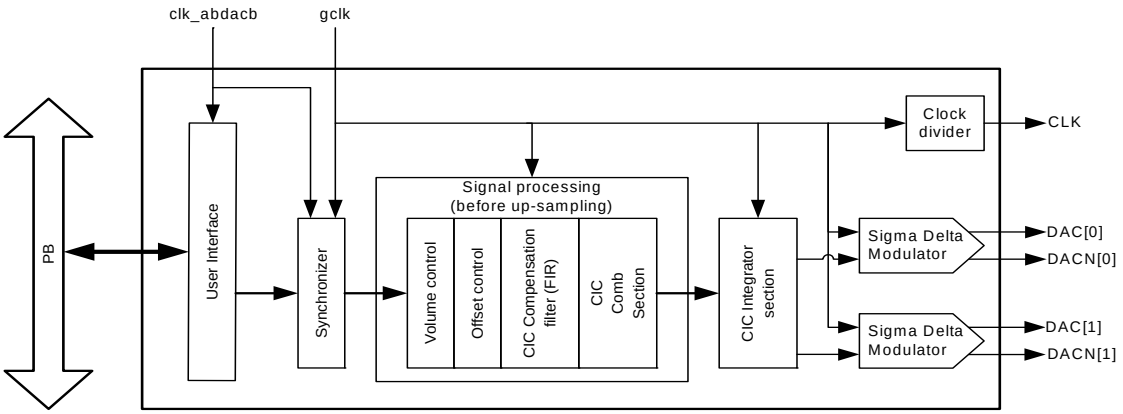
32.2 Overview

The Audio Bitstream DAC (ABDACB) converts a 16-bit sample value to a digital bitstream with an average value proportional to the sample value. Two channels are supported making the Audio Bitstream DAC particularly suitable for stereo audio. Each channel has a pair of complementary digital outputs, DAC and DACN, which can be connected to an external high input impedance amplifier.

The Audio Bitstream DAC is made up of several signal processing blocks and a 3rd order Sigma Delta D/A converter for each channel. The Sigma Delta modulator converts the parallel data to a bitstream, while the signal processing blocks perform volume control, offset control, upsampling, and filtering to compensate for the upsampling process. The upsampling is performed by a Cascaded Integrator-Comb (CIC) filter, and the compensation filter is a Finite Impulse Response (FIR) CIC compensation filter.

32.3 Block Diagram

Figure 32-1. ABDACB Block Diagram



32.4 I/O Lines Description

Table 32-1. I/O Lines Description

Pin Name	Pin Description	Type
DAC[0]	Output for channel 0	Output
DACN[0]	Inverted output for channel 0	Output
DAC[1]	Output for channel 1	Output
DACN[1]	Inverted output for channel 1	Output
CLK	Clock output for DAC	Output

32.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

32.5.1 I/O lines

The output pins used for the output bitstream from the Audio Bitstream DAC may be multiplexed with I/O Controller lines.

Before using the Audio Bitstream DAC, the I/O Controller must be configured in order for the Audio Bitstream DAC I/O lines to be in Audio Bitstream DAC peripheral mode.

32.5.2 Clocks

The clock for the ABDACB bus interface (CLK_ABDACB) is generated by the Power Manager. It is recommended to disable the ABDACB before disabling the clock, to avoid freezing the ABDACB in an undefined state. Before using the Audio Bitstream DAC, the user must ensure that the Audio Bitstream DAC clock is enabled in the Power Manager.

The Audio Bitstream DAC requires a separate clock for the D/A conversion. This clock is provided by a generic clock which has to be set up in the System Control Interface (SCIF). The frequency for this clock has to be set as described in Table 32-3 on page 887. It is important that this clock is accurate and has low jitter. Incorrect frequency will result in too fast or too slow playback (frequency shift), and too high jitter will add noise to the D/A conversion. For best performance one should trade frequency accuracy (within some limits) for low jitter to obtain the best performance as jitter will have large impact on the quality of the converted signal.

32.5.3 DMA

The ABDACB is connected to the Peripheral DMA controller. Using DMA to transfer data samples requires the Peripheral DMA controller to be programmed before enabling the ABDACB.

32.5.4 Interrupts

The ABDACB interrupt request line is connected to the NVIC. Using the ABDACB interrupt requires the NVIC to be programmed first.

32.6 Functional Description

32.6.1 Construction

The Audio Bitstream DAC is divided into several parts, the user interface, the signal processing blocks, and the Sigma Delta modulator blocks. See Figure 32-1 on page 877. The user interface is used to configure the signal processing blocks and to input new data samples to the converter.The signal processing blocks manages volume control, offset control, and upsampling. The Sigma Delta blocks converts the parallel data to1-bit bitstreams.

32.6.1.1 CIC Interpolation Filter

The interpolation filter in the system is a Cascaded Integrator-Comb (CIC) interpolation filter which interpolates from F_S to {125, 128, 136}xF_S depending on the control settings. This filter is a 4th order CIC filter, and the basic building blocks of the filter is a comb part and an integrator part. Since the CIC interpolator has a sinc-function frequency response it is compensated by a linear phase CIC compensation filter to make the passband response more flat in the range 0-20kHz, see figure Figure 32-4 on page 883. The frequency response of this type of interpolator has the first zero at the input sampling frequency. This means that the first repeated specters created by the upsampling process will not be fully rejected and the output signal will contain signals from these repeated specters. See Figure 32-6 on page 884.

Since the human ear can not hear frequencies above 20kHz, we should not be affected by this when the sample rate is above 40kHz, but digital measurement equipment will be affected. This need to be accounted for when doing measurements on the system to prevent aliasing and incorrect measurement results.

32.6.1.2 Sigma Delta Modulator

The Sigma Delta modulator is a 3rd order modulator consisting of three differentiators (delta blocks), three integrators (sigma blocks), and a one bit quantizer. The purpose of the integrators is to shape the noise, so that the noise is reduced in the audio passband and increased at the higher frequencies, where it can be filtered out by an analog low-pass filter. To be able to filter out all the noise at high frequencies the analog low-pass filter must be one order larger than the Sigma Delta modulator.

32.6.1.3 Recreating the Analog Signal

Since the DAC and DACN outputs from the ABDAC are digital square wave signals, they have to be passed through a low pass filter to recreate the analog signal. This also means that noise on the IO voltage will couple through to the analog signal. To remove some of the IO noise the ABDAC can output a clock signal, CLK, which can be used to resample the DAC and DACN signals on external Flip-Flops powered by a clean supply.

32.6.2 Initialization

Before enabling the ABDACB the correct configuration must be applied to the Control Register (CR). Configuring the Alternative Upsampling Ratio bit (CR.ALTUPR), Common Mode Offset Control bit (CR.CMOC), and the Sampling Frequency field (CR.FS) according to the sampling rate of the data that is converted and the type of amplifier the outputs are connected to is required to get the correct behavior of the system. When the correct configuration is applied the ABDACB can be enabled by writing a one to the Enable bit in the Control Register (CR.EN). The module is disabled by writing a zero to the Enable bit. The module should be disabled before entering sleep modes to ensure that the outputs are not left in an undesired state.

32.6.3 Basic operation

To convert audio data to a digital bitstream the user must first initialize the ABDACB as described in Section 32.6.2. When the ABDACB is initialized and enabled it will indicate that it is ready to receive new data by setting the Transmit Ready bit in the Status Register (SR.TXRDY). When the TXRDY bit is set in the Status Register the user has to write new samples to Sample Data Register 0 (SDR0) and Sample Data Register 1 (SDR1). If the Mono Mode (MONO) bit in the Control Register (CR) is set, or one of the compact stereo formats are used by configuring the Data Word Format (DATAFORMAT) in the Control Register, only SDR0 has to be written. Failing to write to the sample data registers will result in an underrun indicated by the Transmit Underrun (TXUR) bit in the Status Register (SR.TXUR). When new samples are written to the sample data registers the TXRDY bit will be cleared.

To increase performance of the system an interrupt handler or DMA transfer can be used to write new samples to the sample data registers. See Section 32.6.10 for details on DMA, and Section 32.6.11 for details on interrupt.

32.6.4 Data Format

The input data type is two's complement. The Audio Bitstream DAC can be configured to accept different audio formats. The format must be configured in the Data Word Format field in the Control Register. In regular operation data for the two channels are written to the sample data registers SDR0 and SDR1. If the data format field specifies a format using less than 32 bits, data must be written right-justified in SDR0 and SDR1. Sign extension into the unused bits is not necessary. Only the 16 most significant bits in the data will be used by the ABDACB. For data formats larger than 16 bits the least significant bits are ignored. For 8-bit data formats the 8 bits will be used as the most significant bits in the 16-bit samples, the additional bits will be zeros.

The ABDACB also supports compact data formats for 16- and 8-bit samples. For 16-bit samples the sample for channel 0 must be written to bits 15 through 0 and the sample for channel 1 must be written to bits 31 through 16 in SDR0. For 8-bit samples the sample for channel 0 must be written to bits 7 through 0 and the sample for channel 1 must be written to bits 15 through 8 in SDR0. SDR1 is not used in this mode. See Table 32-5 on page 889.

32.6.5 Data Swapping

When the Swap Channels (SWAP) bit in the Control Register (CR.SWAP) is one, writing to the Sample Data Register 0 (SDR0) will put the data in Sample Data Register 1 (SDR1). Writing SDR1 will put the data in SDR0. If one of the two compact stereo formats is used the lower and upper halfword of SDR0 will be swapped when writing to SDR0.

32.6.6 Common Mode Offset Control

When the Common Mode Offset Control (CMOC) bit in the Control Register is one the input data will get a DC value applied to it and the amplitude will be scaled. This will make the common mode offset of the two corresponding outputs, DAC and DACN, to move away from each other so that the output signals are not overlapping. The result is that the two signals can be applied to a differential analog filter, and the difference will always be a positive value, removing the need for a negative voltage supply for the filter. The cost of doing this a 3dB loss in dynamic range. On the left side of Figure 32-2 one can see the filtered output from the DAC and DACN pins when a sine wave is played when CR.CMOC is zero. The waveform on the right side shows the output of the differential filter when the two outputs on the left side are used as inputs to the differential filter. Figure 32-3 show the corresponding outputs when CR.CMOC is one.

Figure 32-2. Output signals with CMOC=0

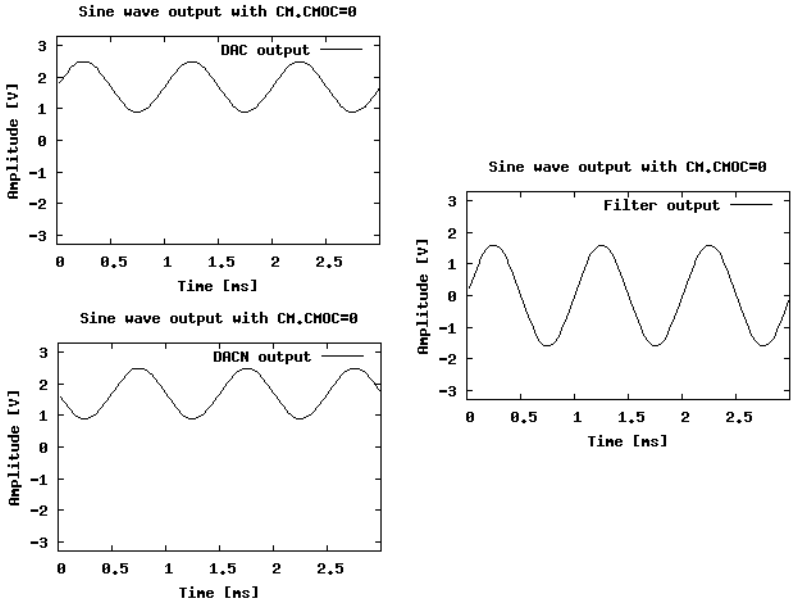
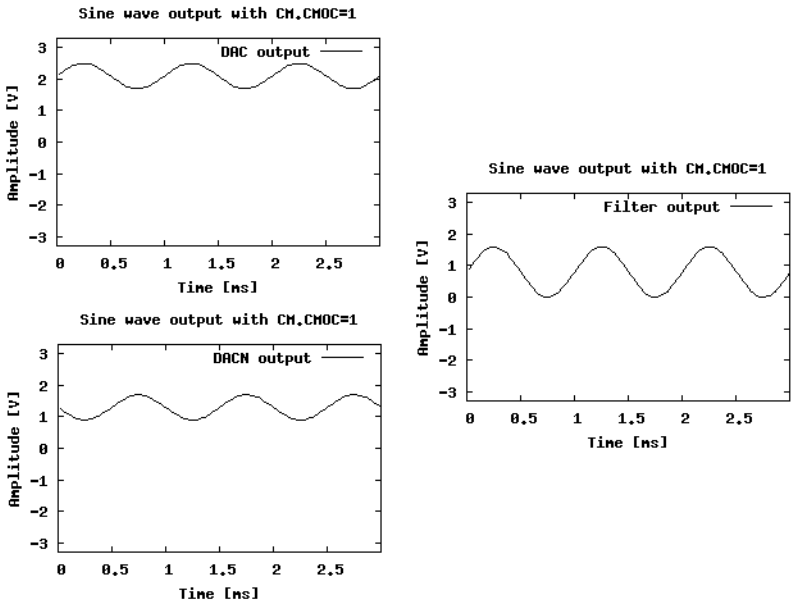


Figure 32-3. Output signals with CMOC=1



32.6.7 Volume Control

The Audio Bitstream DAC have two volume control registers, Volume Control Register 0 (VCR0) and Volume Control Register 1 (VCR1), that can be used to adjust the volume for the corresponding channel. The volume control is linear and will only scale each sample according to the value in the Volume Control (VOLUME) field in the volume control registers. The register also has a Mute bit (MUTE) which can be used to mute the corresponding channel. The filtered out-

put of the DAC pins will have a voltage given by the following equation, given that it is configured to run at the default upsampling ratio of 128:

$$V_{OUT} = \left(\frac{1}{2} - \frac{33}{128} \cdot \frac{SDR}{2^{15}} \cdot \frac{VOLUME}{2^{15}-1}\right) \cdot V_{VDDIO}$$

If one want to get coherence between the sign of the input data and the output voltage one can use the DATAN outputs or invert the sign of the input data by software.

32.6.8 Mono

When the Mono bit (MONO) in the Control Register is set, data written to SDR0 will be used for both output channels. If one of the compact stereo formats are used only the data written to the part of SDR0 that corresponds with channel 0 is used.

32.6.9 Alternative Upsampling Ratio

The digital filters and Sigma Delta modulators requires its own clock to perform the conversion at the correct speed, and this clock is provided by a generic clock in the SCIF. The frequency of this clock depends on the input sample rate and the upsampling ratio which is controlled by the Alternative Upsampling Ratio bit (ALTUPR) in the Control Register.

The ABDACB supports three upsampling ratios, 125, 128, and 136. The default setting is a ratio of 128, and is used when CR.ALTUPR is zero. Using this ratio gives a clock frequency requirement that is common for audio products. In some cases one may want to use other clock frequencies that already are available in the system. By writing a one to CR.ALTUPR a upsampling ratio of 125 or 136 is used depending on the configuration of the Sampling Frequency field in the Control Register. Refer to Table 32-3 for required clock frequency and settings.

The required clock frequency of the generic clock can be calculated from the following equation:

$$GCLK[Hz] = F_S \cdot R \cdot 8$$

R is the upsampling ratio of the converter. If CR.ALTUPR is zero the upsampling ratio is 128. If CR.ALTUPR is one, R will change to 125 when CR.FS is configured for 8kHz, 12kHz, 16kHz, 24kHz, 32kHz, and 48kHz. For the other configurations of CR.FS, 11.025kHz, 22.050kHz, and 44.100kHz, it will change to 136.

32.6.10 DMA operation

The Audio Bitstream DAC is connected to the Peripheral DMA Controller. The Peripheral DMA Controller can be programmed to automatically transfer samples to the Sample Data Registers (SDR0 and SDR1) when the Audio Bitstream DAC is ready for new samples. Two DMA channels are used, one for each sample data register. If the Mono Mode bit in the Control Register (CR.MONO) is one, or one of the compact stereo formats is used, only the DMA channel connected to SDR0 will be used. When using DMA only the Control Register needs to be written in the Audio Bitstream DAC. This enables the Audio Bitstream DAC to operate without any CPU intervention such as polling the Status Register (SR) or using interrupts. See the Peripheral DMA Controller documentation for details on how to setup Peripheral DMA transfers.

32.6.11 Interrupts

The ABDACB requires new data samples at a rate of F_S. The interrupt status bits are used to indicate when the system is ready to receive new samples. The Transmit Ready Interrupt Status bit in the Status Register (SR.TXRDY) will be set whenever the ABDACB is ready to receive a new sample. A new sample value must be written to the sample data registers (SDR0 and

SDR1) before $1/F_S$ second, or an underrun will occur, as indicated by the Underrun Interrupt bit in SR (SR.TXUR). The interrupt bits in SR are cleared by writing a one to the corresponding bit in the Status Clear Register (SCR).

32.6.12 Frequency Response

Figure Figure 32-4 to Figure 32-7 show the frequency response for the system. The sampling frequency used is 48kHz, but the response will be the same for other sampling frequencies, always having the first zero at F_S.

Figure 32-4. Passband Frequency Response

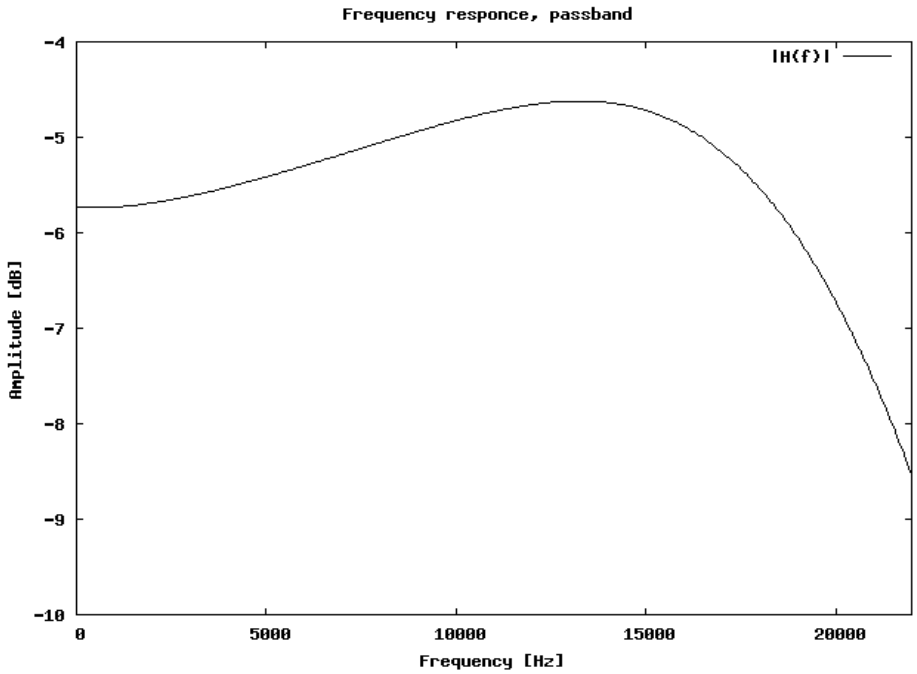


Figure 32-5. Frequency Response up to Sampling Frequency

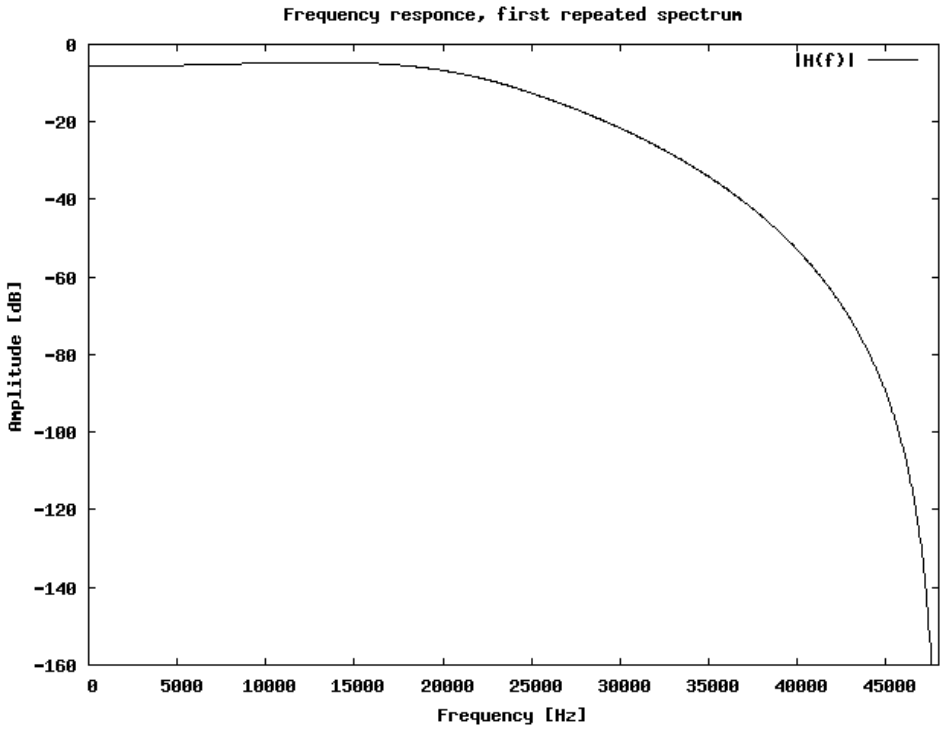


Figure 32-6. Frequency Response up to 3x Sampling Frequency

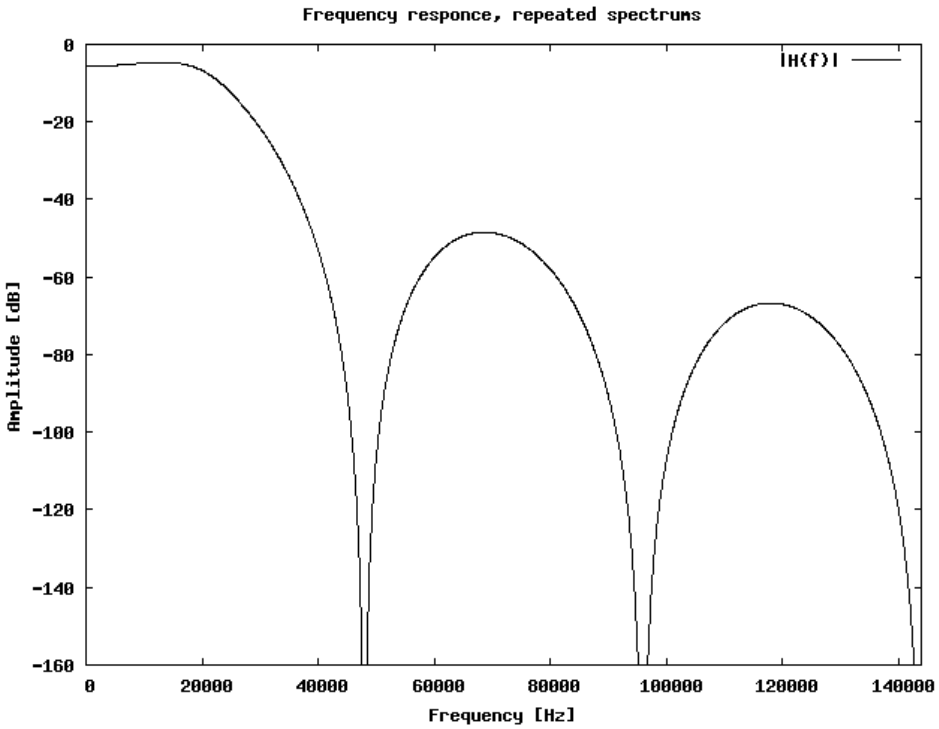
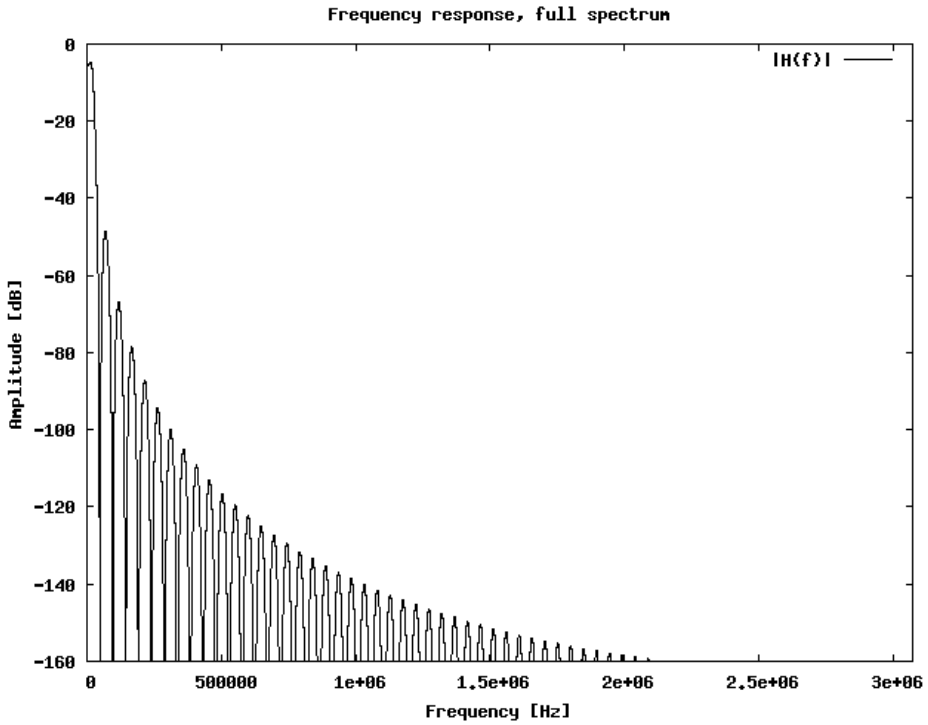


Figure 32-7. Frequency Response up to 128x Sampling Frequency



32.7 User Interface

Table 32-2. ABDACB Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Read/Write	0x00000000
0x04	Sample Data Register 0	SDR0	Read/Write	0x00000000
0x08	Sample Data Register 1	SDR1	Read/Write	0x00000000
0x0C	Volume Control Register 0	VCR0	Read/Write	0x00000000
0x10	Volume Control Register 1	VCR1	Read/Write	0x00000000
0x14	Interrupt Enable Register	IER	Write-only	0x00000000
0x18	Interrupt Disable Register	IDR	Write-only	0x00000000
0x1C	Interrupt Mask Register	IMR	Read-only	0x00000000
0x20	Status Register	SR	Read-only	0x00000000
0x24	Status Clear Register	SCR	Write-only	0x00000000
0x28	Parameter Register	PARAMETER	Read-only	-(1)
0x2C	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

32.7.1 Control Register

Name: CR

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	FS			

23	22	21	20	19	18	17	16
-	-	-	-	-	DATAFORMAT		

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
SWRST	-	MONO	CMOC	ALTUPR	-	SWAP	EN

- **FS: Sampling Frequency**

  Must be set to the matching data sampling frequency, see Table 32-3.

Table 32-3. Generic Clock Requirements

CR.FS	Description	GCLK (CR.ALTUPR=1)	GCLK (CR.ALTUPR=0)
0	8000Hz sampling frequency	8.0MHz	8.1920MHz
1	11025Hz sampling frequency	12.0MHz(1)	11.2896MHz
2	12000Hz sampling frequency	12.0MHz	12.2880MHz
3	16000Hz sampling frequency	16.0MHz	16.3840MHz
4	22050Hz sampling frequency	24.0MHz(1)	22.5792MHz
5	24000Hz sampling frequency	24.0MHz	24.5760MHz
6	32000Hz sampling frequency	32.0MHz	32.7680MHz
7	44100Hz sampling frequency	48.0MHz(1)	45.1584MHz
8	48000Hz sampling frequency	48.0MHz	49.1520MHz
Other	Reserved	-	-

Note: 1. The actual clock requirement are 11.9952MHz, 23.9904MHz, and 47.9808MHz, but this is very close to the suggested clock frequencies, and will only result in a very small frequency shift. This need to be accounted for during testing if comparing to a reference signal.

Notes: 1.

- **DATAFORMAT: Data Word Format**

Table 32-4. Data Word Format

DATAFORMAT	Word length	Comment
0	32 bits	
1	24 bits	
2	20 bits	
3	18 bits	
4	16 bits	
5	16 bits compact stereo	Channel 1 sample in bits 31 through 16, channel 0 sample in bits 15 through 0 in SDR0
6	8 bits	
7	8 bits compact stereo	Channel 1 sample in bits 15 through 8, channel 0 sample in bits 7through 0 in SDR0

- **SWRST: Software Reset**
  Writing a zero to this bit does not have any effect.
  Writing a one to this bit will reset the ABDACB as if a hardware reset was done.
- **MONO: Mono Mode**
  0: Mono mode is disabled.
  1: Mono mode is enabled.
- **CMOC: Common Mode Offset Control**
  0: Common mode adjustment is disabled.
  1: Common mode adjustment is enabled.
- **ALTUPR: Alternative Upsampling Ratio**
  0: Alternative upsampling is disabled.
  1: Alternative upsampling is enabled.
- **SWAP: Swap Channels**
  0: Channel swap is disabled.
  1: Channel swap is enabled.
- **EN: Enable**
  0: The ABDACB is disabled.
  1: The ABDACB is enabled.

32.7.2 Sample Data Register 0

Name: SDR0

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DATA[31:24]							

23	22	21	20	19	18	17	16
DATA[23:16]							

15	14	13	12	11	10	9	8
DATA[15:8]							

7	6	5	4	3	2	1	0
DATA[7:0]							

- **DATA: Sample Data**
  Sample Data for channel 0 in two's complement format. Data must be right-justified, see Table 32-5.

Table 32-5. Sample Data Register Formats

Data Format	SDR0	SDR1	Comment
32 bits	CH0 sample in DATA[31:0]	CH1 sample in DATA[31:0]	
24 bits	CH0 sample in DATA[23:0]	CH1 sample in DATA[23:0]	Remaining bits are ignored.
20 bits	CH0 sample in DATA[19:0]	CH1 sample in DATA[19:0]	Remaining bits are ignored.
18 bits	CH0 sample in DATA[17:0]	CH1 sample in DATA[17:0]	Remaining bits are ignored.
16 bits	CH0 sample in DATA[15:0]	CH1 sample in DATA[15:0]	Remaining bits are ignored.
16 bits compact stereo	CH0 sample in DATA[15:0] CH1 sample in DATA[31:16]	Not used	
8 bits	CH0 sample in DATA[7:0]	CH1 sample in DATA[7:0]	Remaining bits are ignored.
8 bits compact stereo	CH0 sample in DATA[7:0] CH1 sample in DATA[15:8]	Not used	Remaining bits are ignored.

32.7.3 Sample Data Register 1

Name: SDR1

Access Type: Read/Write

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DATA[31:24]							

23	22	21	20	19	18	17	16
DATA[23:16]							

15	14	13	12	11	10	9	8
DATA[15:8]							

7	6	5	4	3	2	1	0
DATA[7:0]							

- **DATA: Sample Data**
  Sample Data for channel 1 in two's complement format. Data must be right-justified, see Table 32-5 on page 889.

32.7.4 Volume Control Register 0

Name: VCR0

Access Type: Read/Write

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
MUTE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	VOLUME[14:8]						
7	6	5	4	3	2	1	0
VOLUME[7:0]							

- **MUTE: Mute**
  0: Channel 0 is not muted.
  1: Channel 0 is muted.
- **VOLUME: Volume Control**
  15-bit value adjusting the volume for channel 0.

32.7.5 Volume Control Register 1

Name: VCR1

Access Type: Read/Write

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
MUTE	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	VOLUME[14:8]						
7	6	5	4	3	2	1	0
VOLUME[7:0]							

- **MUTE: Mute**
  0: Channel 1 is not muted.
  1: Channel 1 is muted.
- **VOLUME: Volume Control**
  15-bit value adjusting the volume for channel 1.

32.7.6 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x14

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

32.7.7 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x18

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

32.7.8 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000000

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

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

A bit in this register is cleared when the corresponding bit in IDR is written to one.

A bit in this register is set when the corresponding bit in IER is written to one.

32.7.9 Status Register

Name: SR

Access Type: Read-only

Offset: 0x20

Reset Value: 0x00000000

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

- **TXUR: Transmit Underrun**
  This bit is cleared when no underrun has occurred since the last time this bit was cleared (by reset or by writing to SCR).
  This bit is set when at least one underrun has occurred since the last time this bit was cleared (by reset or by writing to SCR).
- **TXRDY: Transmit Ready**
  This bit is cleared when the ABDACB is not ready to receive a new data in SDR.
  This bit is set when the ABDACB is ready to receive a new data in SDR.
- **BUSY: ABDACB Busy**
  This bit is set when the ABDACB is busy doing a data transfer between clock domains. CR, SDR0, and SDR1 can not be written during this time.

32.7.10 Status Clear Register

Name: SCR

Access Type: Write-only

Offset: 0x24

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

32.7.11 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0x28

Reset Value: 0x00000000

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

Reserved. No functionality associated.

32.7.12 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x2C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			
15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			
7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

32.8 Module Configuration

The specific configuration for each ABDACB instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 32-6. ABDACB Clocks

Clock Name	Description
CLK_ABDACB	Clock for the ABDACB bus interface
GCLK	The generic clock used for the ABDACB is GCLK6

Table 32-7. Register Reset Values

Register	Reset Value
VERSION	0x00000100
PARAMETER	0x00000000

33. Digital to Analog Converter Controller (DACC)

Rev. 1.1.1.0

33.1 Features

- **10-bit Resolution**
- **Hardware Trigger**
  - **External Trigger Pins**
  - **Peripheral Event**
- **PDCA Support**
- **DACC Timings Configuration**
- **Internal FIFO for flexibility and efficiency**

33.2 Overview

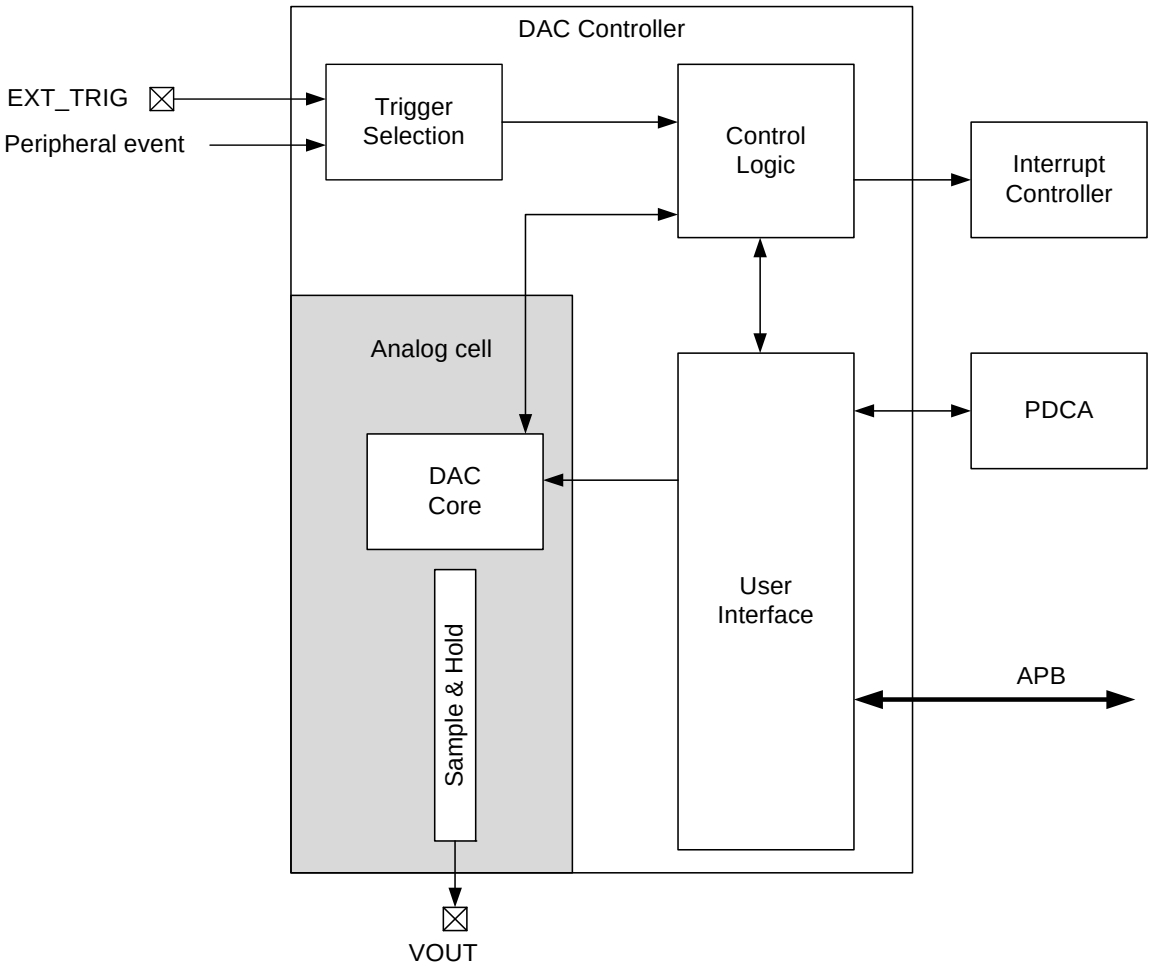
The Digital-to-Analog Converter Controller (DACC) supports 10-bit resolution. Data to be converted are sent in a common register. External triggers, through ext_trig pins, and internal triggers (events) are configurable.

The DACC connects with a PDCA channel reducing and processor intervention.

User can configure DACC timings, such as Startup Time and Internal Trigger Period.

33.3 Block Diagram

Figure 33-1. Digital-to-Analog Converter Controller Block Diagram



33.4 Signal Description

Table 33-1. DACC Pin Description

Pin Name	Description
VOUT	Analog output
EXT_TRIG	External trigger

33.5 Product Dependencies

33.5.1 Clocks

DACC uses bus interface clock (CLK_DACC) which is generated by the Power Manager. It is recommended to disable the DACC before disabling the clock, to avoid freezing the DACC in an undefined state.

33.5.2 Interrupt Sources

The DACC interrupt line is connected on one of the internal sources of the NVIC. Using the DACC interrupt requires the NVIC to be programmed first.

33.6 Functional Description

33.6.1 Digital-to-Analog Conversion

DACC uses the APB clock (CLK_DACC) to perform conversions.

DAC is enabled by writing a one to DACEN in Mode Register (MR). Once enabled, DAC is ready to operate after a startup time (see electrical characteristics). User must therefore configure startup time by writing new value in MR.STARTUP, according to APB clcok frequency. Startup time is therefore (MR.STARTUP+1) * APB clock period.

Once a conversion is started, DAC requires setup time before providing analog result on the analog pin VOUT (see electrical characteristics).

33.6.2 Conversion FIFO

To provide flexibility and high efficiency, a 4 half-word FIFO is used to handle the data to convert.

As long as the Transmit Ready bit (TXRDY) in the Interrupt Status Register (ISR) is set to one the DAC Controller accepts new conversion requests by writing data to the Conversion Data Register (CDR). Data which cannot be converted immediately are stored in the FIFO.

When the FIFO is full or the DACC is not ready to accept conversion requests, TXRDY is low. Writing to CDR while TXRDY is low corrupts FIFO data.

User can configure transfer data size. When bit WORD in MR is set to one transfer size is 16 bits and when WORD is set to one transfer size is 32 bits. In 16 bits transfer size, data to convert is CDR[9:0] and in 32 bits transfer size, data to convert are CDR[9:0] (first data to convert) and CDR[25:16] (second data to convert).

33.6.3 Conversion Triggers

Internal trigger mode is selected by writing a zero to Trigger Enable (MR.TRGEN). In internal trigger mode, conversion starts as soon as DAC is enabled, data is written in the Conversion Data Register (CDR) and internal trigger event occurs. The internal trigger frequency is configurable through the MR.CLKDIV and must not exceed the maximum frequency allowed by the DAC. Trigger period is therefore CLKDIV * APB clock period.

External trigger mode is selected by writing a one to MR.TRGEN. The external event source is configured by writing a zero to Trigger Select (MR.TRGSEL) to select external pin EXT_TRG and a one to select the peripheral event (from PEVC). With external pin source, DACC waits for a rising edge to begin conversion. With peripheral event source, DACC waits for event (configured by PEVC) to begin conversion.

Figure 33-2. Internal Trigger

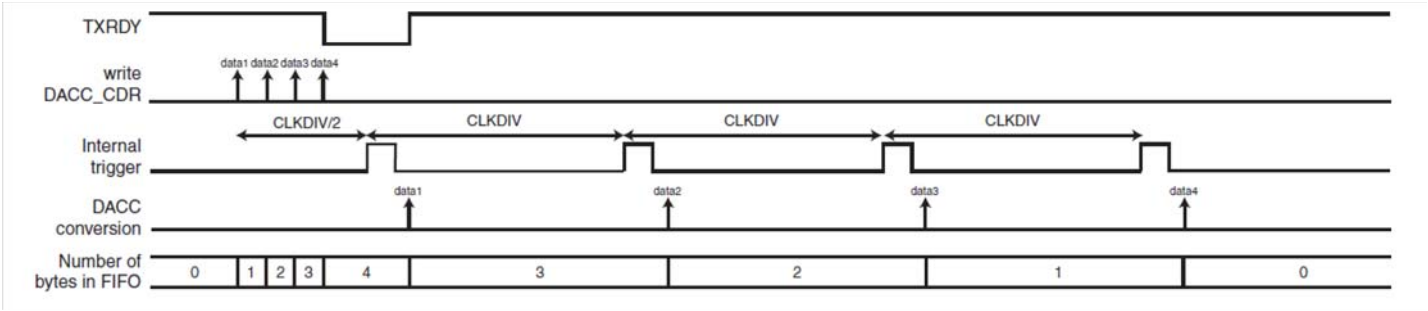
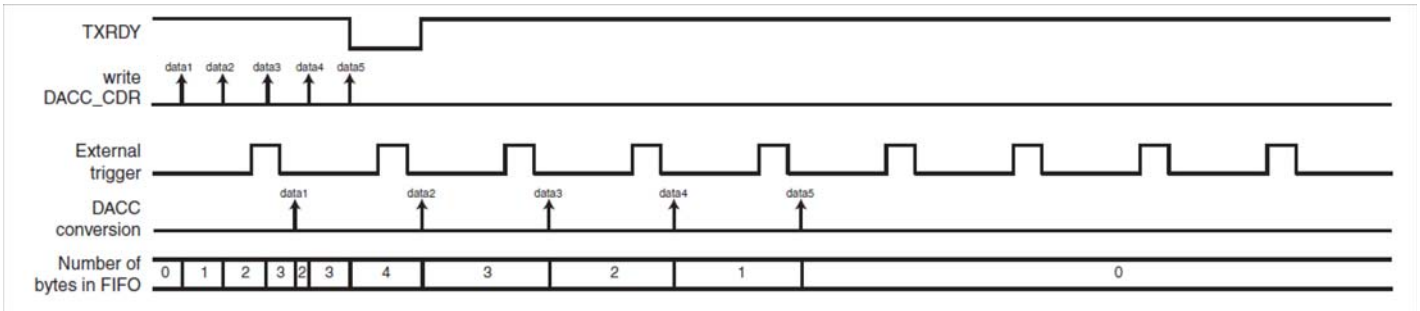


Figure 33-3. External Trigger



33.6.4 Write Protection Registers

In order to provide security to the DACC, a write protection mode is implemented. This mode is enabled by writing a one in Write Protect Enable (WPE) in the Write Protect Mode Register (WPMR) and disabled by writing a zero. Writing to WPMR requires to write a security key in Write Protect Key (WPMR.WPKEY). The value is “DAC” in ASCII, corresponding to 0x444143.

The write protection mode prevents the write of Mode Register. When this mode is enabled and the protected registers is written, an error is generated in the Write Protect Status Register (WPSR) and the register write request is canceled. When a write protection error occurs, the Write Protection Error bit (WPROTERR) is set to one.

33.7 User Interface

Table 33-2. DACC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-only	–
0x04	Mode Register	MR	Read-write	0x00000000
0x08	Conversion Data Register	CDR	Write-only	0x00000000
0x0C	Interrupt Enable Register	IER	Write-only	–
0x10	Interrupt Disable Register	IDR	Write-only	–
0x14	Interrupt Mask Register	IMR	Read-only	0x00000000
0x18	Interrupt Status Register	ISR	Read-only	0x00000000
0xE4	Write Protect Mode register	WPMR	Read-write	0x00000000
0xE8	Write Protect Status register	WPSR	Read-only	0x00000000
0xFC	Version Register	VERSION	Read-only	0x-(1)

Note: 1. The reset value is device specific. Refer to the Module Configuration section at the end of this chapter.

33.7.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: -

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

- **SWRST: Software Reset**
  Writing a one to this bit resets the DACC.
  Writing a zero to this bit has no effect.

33.7.2 Mode Register

Name: MR

Access: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
CLKDIV							
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
STARTUP							
7	6	5	4	3	2	1	0
–	–	WORD	DACEN	TRGSEL			TRGEN

This register can only be written if the WPEN bit is cleared in Write Protect Mode Register.

- **CLKDIV: Clock Divider for Internal Trigger**
  Trigger period is CLKDIV * APB clock period.
- **STARTUP: Startup Time Selection**
  Starup time is (STARTUP + 1) * APB clock period.
- **WORD: Word Transfer**

WORD	Selected Resolution
0	Half-Word transfer (16 bits)
1	Word Transfer (32 bits)

- **DACEN: DAC Enable**
  Writing a one to this bit enables the DAC.
  Writing a zero to this bit disables the DAC.
- **TRGSEL: Trigger Selection**

TRGSEL			Selected TRGSEL
0	0	0	external trigger
0	0	1	peripheral event
0	1	0	Reserved
0	1	1	Reserved
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

- **TRGEN: Trigger Enable**
  0: Internal trigger mode.
  1: External trigger mode.

33.7.3 Conversion Data Register

Name: CDR

Access: Write-only

Offset: 0x08

Reset Value: 0x00000000

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

- **DATA: Data to Convert**
  When the MR.WORD is zero, only DATA[15:0] is used for conversion else DATA[31:0] is used to write 2 data for conversion.

33.7.4 Interrupt Enable Register

Name: IER

Access: Write-only

Offset: 0x0C

Reset Value: -

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

33.7.5 Interrupt Disable Register

Name: IDR

Access: Write-only

Offset: 0x10

Reset Value: -

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

33.7.6 Interrupt Mask Register

Name: IMR

Access: Read-only

Offset: 0x14

Reset Value: 0x00000000

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

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

A bit in this register is cleared when the corresponding bit in IDR is written to one.

A bit in this register is set when the corresponding bit in IER is written to one.

33.7.7 Interrupt Status Register

Name: ISR

Access: Read-only

Offset: 0x18

Reset Value: 0x00000000

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

- **TXRDY: Transmit Ready**
  0 = DACC is not ready to accept new conversion requests.
  1 = DACC is ready to accept new conversion requests.

33.7.8 Write Protect Mode Register

Name: WPMR

Access: Read/Write

Offset: 0xE4

Reset Value: 0x00000000

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

- **WPKEY: Write Protect KEY**
  This security code is needed to set/reset the WPEN bit.
  Must be filled with “DAC” ASCII code (0x444143).
- **WPEN: Write Protect Enable**
  0 = Disables the Write Protect.
  1 = Enables the Write Protect.

33.7.9 Write Protect Status Register

Name: WPSR

Access: Read-only

Offset: 0xE8

Reset Value: 0x00000000

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

- **WPROTADDR: Write Protection Error Address**
  Address of the register write request which generated the error.
- **WPROTERR: Write Protection Error**
  0: no error.
  1: write protection error.

33.7.10 Version Register

Name: VERSION

Access: Read-only

Offset: 0xFC

Reset Value: -

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

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Reserved. No functionality associated.

33.8 Module Configuration

The specific configuration for each DACC instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 33-3. ABDACB Clocks

Clock Name	Description
CLK_DACC	Clock for the DACC bus interface

Table 33-4. Register Reset Values

Register	Reset Value
VERSION	0x00000111

34. Capacitive Touch Module (CATB)

Rev: 1.0.0.0

34.1 Features

- **Low-power capacitive proximity and touch detection**
- **Touch filtering without CPU intervention**
- **Capacitance measurement on multiple sensors for button/slider/wheel support**
- **Autonomous interrupt generation on touch and out-of-touch events**
- **Autocalibration provides automatic adaptation to changing operating conditions**
- **Peripheral-event-triggered acquisition for autonomous sleep operation and synchronization**
- **Spread-spectrum operation for EMI compatibility**
- **Differential or single-ended sensing operation for measurement of both mutual and self capacitance**
- **Support for internal and external discharge resistors**

34.2 Overview

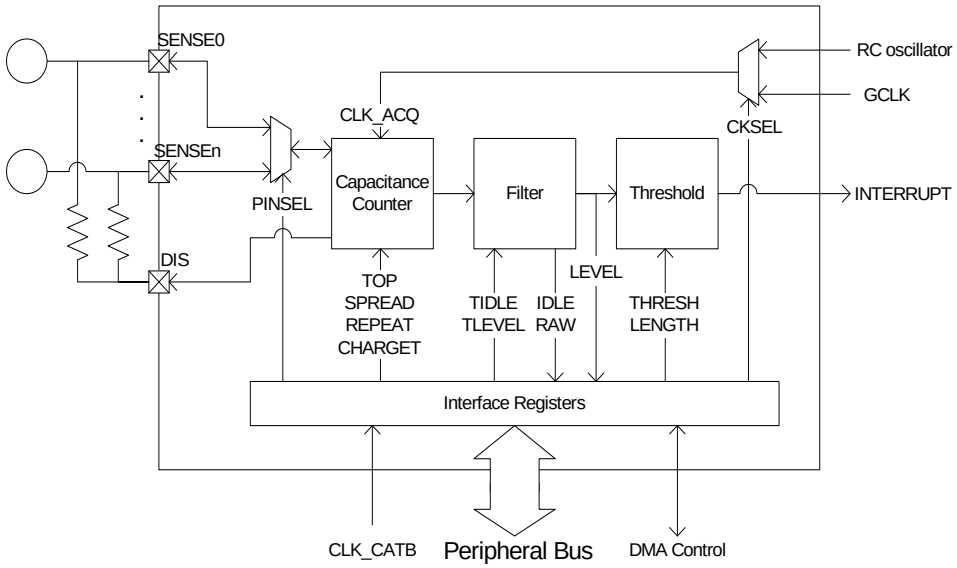
The Capacitive Touch Module (CATB) performs acquisition, filtering, and detection of capacitive touch sensors. The capacitive touch sensors use no mechanical components, and therefore demand less maintenance in the user application. The CATB can operate in different capacitance discharge modes:

- Single-ended, with one pin per sensor
- Differential, with two pins per sensor
- External discharge resistors, with an extra pin (DIS) in single-ended mode
- Internal discharge resistors

Using DMA, the CATB can sense on multiple sensor pins. Refer to the Module Configuration section for the number *n* of supported sensors. The CATB can report measured capacitance values to software algorithms for the implementation of buttons/sliders/wheels, and the user can configure thresholds triggering interrupts.

34.3 Block Diagram

Figure 34-1. CATB Block Diagram



34.4 I/O Lines Description

Table 34-1. I/O Lines Description

Pin Name	Pin Description	Type
SENSE[n:0]	Capacitive sense line	Input/Output
DIS	Capacitive discharge line	Output

34.5 Product Dependencies

In order to use the CATB module, other parts of the system must be configured correctly, as described below.

34.5.1 I/O Lines

The CATB pins may be multiplexed with the I/O Controller lines. The user must first configure the I/O Controller to assign the desired CATB pins to their peripheral functions.

34.5.2 Power Management

If the CPU enters a sleep mode that disables clocks used by the CATB, it will stop functioning and the CATB must be reinitialized to resume operation after the system wakes up from sleep mode. Ongoing measurements will be invalid. The CATB can automatically request clocks when using the Peripheral Event System. The CATB is able to wake the system from sleep mode using interrupts.

34.5.3 Clocks

The clock for the CATB bus interface (CLK_CATB) is generated by the Power Manager (PM). This clock is enabled at reset, and can be disabled in the PM. It is recommended to disable the CATB before disabling this clock, in order to avoid freezing the CATB in an undefined state.

The CATB also depends on the acquisition clock (CLK_ACQ). This clock can be configured to be either a dedicated generic clock (GCLK), or a dedicated RC oscillator. The GCLK must be enabled in the System Control Interface (SCIF) before the CATB can be used. The RC oscillator is automatically enabled when needed. The RC oscillator and GCLK used is specified in the Module Configuration section.

34.5.4 Interrupts

The CATB interrupt request line is connected to the NVIC. Using the CATB interrupt requires the NVIC to be configured first.

34.5.5 Direct Memory Access

The CATB can use DMA to switch its configuration when sensing multiple sensors. Two Peripheral DMA channels must be configured: One for transferring configuration and state data into the CATB, and one for storing the updated values back to RAM.

34.5.6 Debug Operation

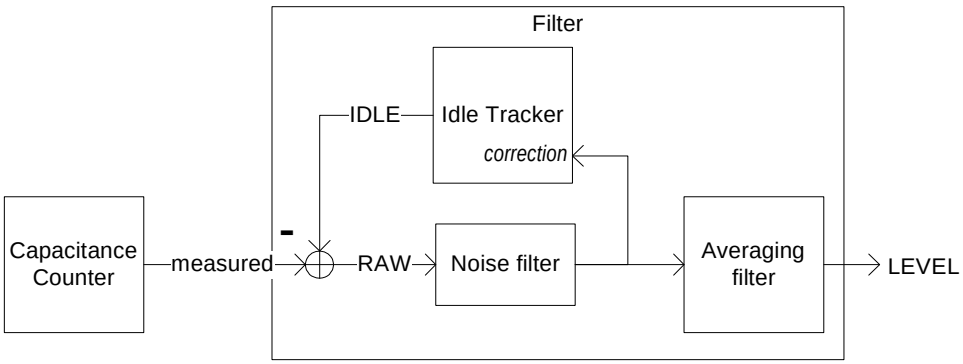
When an external debugger forces the CPU into debug mode, the CATB continues normal operation. If the CATB is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging. Care must be taken when accessing the DMA register when the CATB is in DMA mode, as accessing this register can be misinterpreted as DMA handshakes by the CATB.

34.6 Functional Description

34.6.1 Principle of Operation

The CATB module detects touch by measuring change in sensor capacitance. The capacitance is measured by charging the sensor to a potential, and then timing the discharge through a resistor, yielding a measurement reflecting the current RC value. The Capacitance Counter block in Figure 34-1 and Figure 34-2 performs the charge/discharge cycling, and reports measured values to the filter.

Figure 34-2. CATB Signal Filter



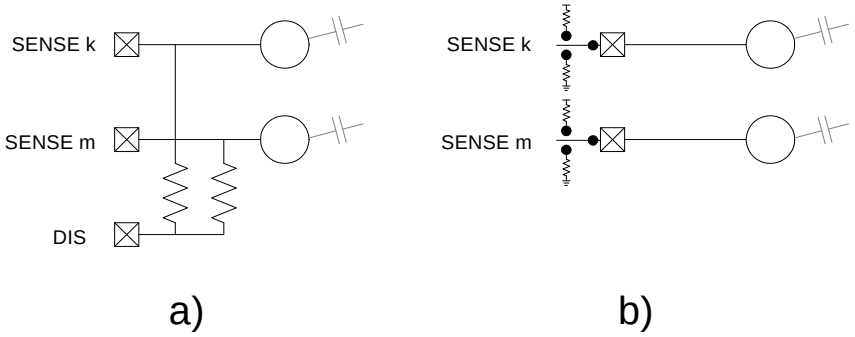
As shown in Figure 34-2, the filter receives the measured values, subtracts a bias (the IDLE value), and filters away noise before using this as a correction value for the Idle Tracker. The filtered signal is further smoothed before yielding the filtered LEVEL signal.

The application can record and process the acquired LEVEL values in order to implement advanced touch-detection algorithms, or use the configurable thresholds to generate interrupts.

Capacitive changes can correspond to touch, proximity, out-of-touch, or events such as change of power supply (ripple or ground shift). Fast changing environmental parameters can also affect the touch measurements.

While the CATB can only measure and hold configuration and state for one sensor at the time, the Peripheral DMA Controller can be used with a buffer in RAM to rotate sensor configuration for the CATB. 20 bytes of RAM are then needed per sensor.

Figure 34-3. CATB Sensor Connections.



The CATB can operate in single-ended or differential mode. In single-ended mode, one pin and one conductive area per sensor is required (Figure 34-3 a and b), selected in the PINSEL register. When using external resistors in single-ended mode, a common discharge pin must be used (Figure 34-3 a).

34.6.2 Basic Operation

Before configuring the CATB, the CATB interface must be enabled by writing a one to the Access Enable bit in the Control Register (CR.EN). This enables writes to the CATB registers. During the CATB operation, the CATB interface can be disabled by writing a zero to CR.EN to save power and avoid accidental writes to the configuration registers.

The CATB is by default set up in single-ended mode with external resistors. The default source for the acquisition clock is the dedicated RC oscillator. Refer to the Module Configuration Sectoin for configuration of oscillators.

The sample rate is selected by writing to the Counter Top Value field in the Count Control Register (CNTCR.TOP) where

$$\text{sample rate} = \frac{1}{\text{sample period}} = \frac{f_{CLK\_ACQ}}{\alpha \cdot TOP \cdot REPEAT}$$

For single-ended mode, α is 2, and for differential mode, 4. Note that CNTCR.TOP must be written to a value such that the sample period is larger than the discharge time of the sensor, determined by the product of the sensor capacitance and the discharge resistance.

Before touch can be captured, the Idle Smoothing Factor and Relative Level Smoothing Factor time durations in the Filter Timing Register (TIMING.TIDLE and TIMING.TLEVEL) must be chosen to match the CNTCR sample rate settings. TIMING.TIDLE and TIMING.TLEVEL are fractional positive values less than one and determine how many seconds the filters should

average over. They should equal

$$\frac{2}{T \cdot \text{sample rate} + 1}$$

with typical T values being 5s for TIMING.TIDLE, and 0.1s for TIMING.TLEVEL.

Because the Idle Tracker is used to bias the measured values around zero, IDLE needs to be initialized with an appropriate value in order to avoid saturation of the filter. This value can either be obtained experimentally and stored in the application, or by writing a one to the Initialize Idle bit (CR.IIDLE) when the sensor is not touched, which causes IDLE to be loaded with the next measured value. CR.IIDLE is automatically cleared by hardware when this is done.

The CATB can perform acquisition on an array of selectable SENSE pins. The user selects pins in the Pin Select Register (PINSEL). SENSE[m] is selected by writing the value *m* to PINSEL.

To enable CATB touch acquisition, the configuration registers should be loaded with application and sensor specific settings. Writing a one to the Start Operation bit (CR.RUN) starts conversion and touch data is available when bits in the Interrupt Status Register are set, optionally generating interrupts. Do not change settings during CATB operation (CR.RUN is one).

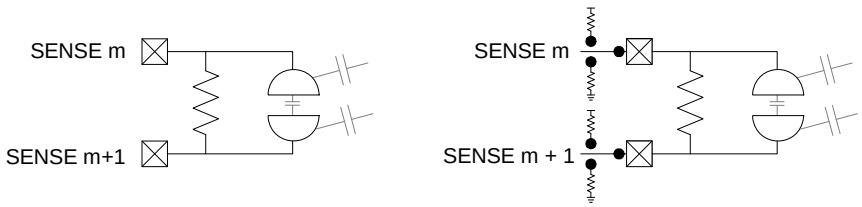
When an acquisition is done and a new sample is ready, the Sample Ready bit in the Interrupt Status register (ISR.SAMPLE) is set. Note that an overflow can occur if the CATB sample rate is higher than the CPU's capability to read the data. The CATB will not report such overflows, and the user should check and verify the expected data rate with a timer.

34.6.3 Differential Mode

The CATB can be configured to work in differential mode, where sensing is done on two conductive sensors. In this mode, both the mutual capacitance between the two sensors and the common capacitance is measured. In single-ended mode, only the common capacitance is measured.

The differential mode is enabled by writing a one to the Differential Mode bit in the Control Register (CR.DIFF). In differential mode, two pins and two conductive areas per sensor are required (Figure 34-3 a and b). Two pins will be used, SENSE[m] and SENSE[m+1], selected by writing m to PINSEL.

Figure 34-4. CATB Sensor Connection, Differential Mode



a) b)

34.6.4 Thresholding

In order for the CATB to autonomously report touch, a threshold value must be written to the Threshold Register (THRESH). The threshold value consists of an integer part

(THRESH.RTHRESH), and a fractional part (THRESH.FTHRESH). The user should select a threshold value that corresponds to application typical sensor signal strengths. The threshold values are compared to the LEVEL register values, and the In Touch bit in the Interrupt Status Register (ISR.INTCH) is set when the value in the LEVEL register is larger than the value in the THRESH register, while the Out of Touch bit (ISR.OUTTCH) is set when LEVEL becomes less than *half* of THRESH, in effect creating a hysteresis between in- and out-of-touch events.

During long touches, the value in the IDLE register will slowly increase and eventually cause a false out-of-touch event. When the touch is released, the measured value will become smaller than IDLE, causing a negative response on LEVEL. Writing a one to the Threshold Direction bit in THRESH (THRESH.DIR) enables negative thresholding, causing the CATB to compare against -THRESH. During negative thresholding, ISR.INTCH is set when LEVEL goes *below* the negative THRESH value, and ISR.OUTTCH is set when LEVEL goes *above* the negative THRESH/2 value.

For better noise immunity, a threshold filter can be configured in the Threshold Length field in THRESH (THRESH.LENGTH). The threshold filter requires a number of samples to be over (or under, if negative thresholding is used) the threshold before the CATB reports an in- or out-of-touch event.

34.6.5 Acquisition Clock and Timing

The discharge time of the capacitive sensor over the resistor is measured against CLK_ACQ. The source for this clock is selected with the Clock Select bit in the Control Register (CR.CKSEL), and must remain unchanged during CATB operation. The RC oscillator is available on demand, and the GCLK must be enabled before enabling the CATB. Refer to the Module Configuration section for the configuration of the generic clock and the RC oscillator.

Before each discharge measurement, the sensor is charged towards the I/O voltage or ground. Writing to the Charge Time field in CR (CR.CHARGET) selects the charge time duration, if larger charge times are needed, e.g. for large capacitive loads.

If the measured discharge time is too small, the user can either increase the charge time by using a larger resistance value for the discharge resistor, or configure the CATB to repeat and accumulate measurements by writing to the Repeat Measurements field in CNTCR (CNTCR.REPEAT). Writing a non-zero value to CNTCR.REPEAT will cause the acquisition block to repeat the acquisition REPEAT+1 times, summing the individual measurements before sending the sum to the filter block.

34.6.6 Filter Algorithm

The filter in the CATB consists of three main parts: A median3 filter, an Idle tracker, and an averaging filter. The data in the CATB is represented in a fixed-point format consisting of an integer part and a fractional part. Not all of the LSB's of the fractional bits might be implemented. Refer to the Module Configuration section for the number of implemented fractional bits.

The filter in the CATB implements the following recurrence equations:

$$RAW_n = measured_n - IDLE_n$$

$$median_n = median3(RAW_{n-2}, RAW_{n-1}, RAW_n)$$

$$IDLE_{n+1} = IDLE_n + TIDLE \cdot median_n$$

$$LEVEL_{n+1} = LEVEL_n + TLEVEL \cdot (median_n - LEVEL_n)$$

The RAW_n and RAW_{n-1} values can be observed in the RAWA and RAWB fields respectively in the RAW register. These values can be logged, and the user can simulate the rest of the filter for tuning and debugging purposes.

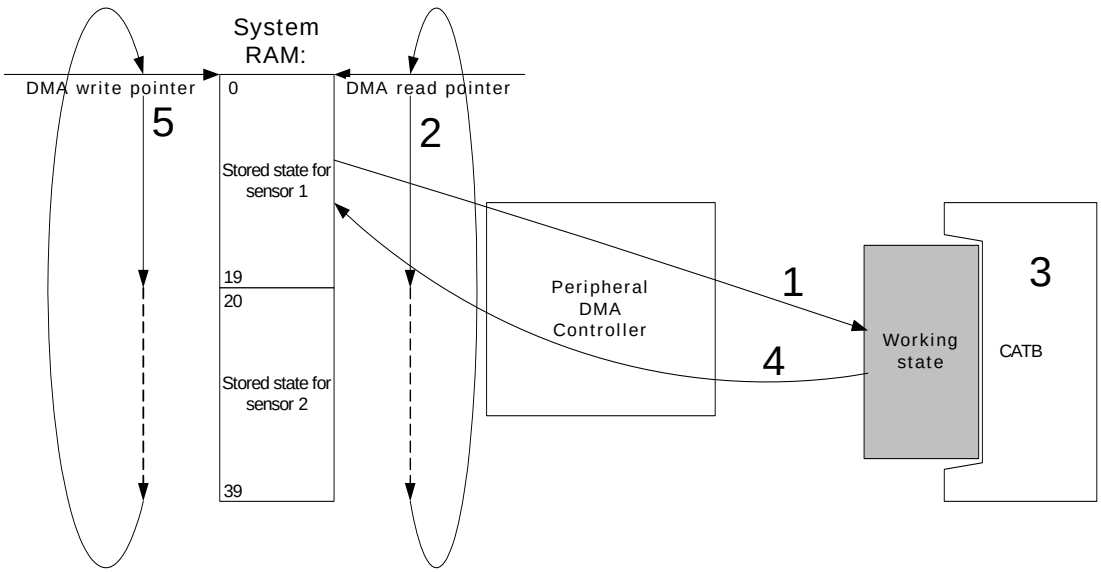
The Idle tracker must be set up using TIMING.TIDLE to function as a slow-moving average to be subtracted from the measured values to remove the constant bias. This biases the signal around zero, with fast changes in capacitance shown as deviations from zero.

The averaging filter, which outputs values to LEVEL, must be set up using TIMING.TLEVEL to have a faster response than the Idle tracker, in order to function as a smoothing filter on the median-filtered signal. The thresholds are compared to the LEVEL value to decide whether or not a touch event should be generated.

34.6.7 Multiple Sensors and DMA Operation

The CATB can only hold state and configuration for one sensor at the time. In order to use the CATB with multiple sensors, sensor data can be stored in RAM using 5 x 32-bit words (20 bytes) per sensor (refer to Table 34-2). The Peripheral DMA Controller must be used to transfer state and configuration for the different sensors back and forth between the CATB and system memory. DMA mode of the CATB is enabled by writing a one to the DMA Enable bit in the Control Register (CR.DMAEN). When DMA mode of the CATB is enabled, the CATB automatically generates the required handshakes to the DMA controller for transferring data back and forth with the correct timing.

Figure 34-5. Transferring of Configuration and State in Multisensor Mode



As shown in Figure 34-5, configuration and state data is transferred between the system memory and the CATB using DMA in the following sequence:

1. The configuration and state data for sensor 1 is read from memory and written to the CATB
2. The DMA read pointer is incremented, and the next read from memory will read the data for sensor 2
3. The CATB performs an acquisition using the parameters for sensor 1

4. The updated data is written back to memory, overwriting the previous configuration and state
5. The DMA write pointer is incremented, and the next write to memory will write the data for sensor 2

Using the ring buffer feature of the DMA controller, the system will go cyclically through the sensors set up in memory.

Detected touch events for different sensor configurations are reported in the In-touch Status bits in the In-Touch Registers (INTCHn.INTCHm), and in the Out-of-Touch Status bits in the Out-of-Touch Registers (OUTTCHn.OUTTCHm). When bits in these registers are set, the ISR.INTCH or ISR.OUTTCH bits are also set. The switched data contains the following:

- Timing information (as described in the TIMING register)
- Current IDLE and LEVEL values
- Threshold configuration (as described in the THRESH register)
- Current Touch State (TCHSTATE); zero being out-of touch, selecting THRESH for monitoring, and one being in-touch, selecting THRESH/2 for monitoring
- Status Bit Select (STATUSSEL), selecting INTCHn.INTCHm, and OUTTCHn.OUTTCHm
- Internal states for the DSP (should be initialized to zero); Length Count (LENGTH_COUNT), counting samples beyond thresholds, and Comparator State (CMPSTATE), a filter cache

Table 34-2. Layout of Data in Memory for One Sensor Configuration

Byte/Bit	7	6	5	4	3	2	1	0
0	-	LENGTH_COUNT[4:3]			THRESH.LENGTH[4:0]			
1	THRESH.DIR	LENGTH_COUNT[2:0]			THRESH.RTHRESH[7:4]			
2	THRESH.RTHRESH[3:0]				THRESH.FTHRESH[11:8]			
3	THRESH.FTHRESH[7:0]							
4	-	-	-	-	IDLE.RIDLE[15:12]			
5	IDLE.RIDLE[11:4]							
6	IDLE.RIDLE[3:0]				IDLE.FIDLE[11:8]			
7	IDLE.FIDLE[7:0]							
8	-	-	-	-	-	-	-	-
9	-	-	-	-	LEVEL.RLEVEL[7:4]			
10	LEVEL.RLEVEL[3:0]				LEVEL.FLEVEL[11:8]			
11	LEVEL.FLEVEL[7:0]							
12	CMPSTATE	TCHSTATE	-	-	TIMING.TIDLE[11:8]			
13	TIMING.TIDLE[7:0]							
14	-	-	-	-	TIMING.TLEVEL[11:8]			
15	TIMING.TLEVEL[7:0]							
16	RAW.RAWB[7:0]							
17	RAW.RAWA[7:0]							
18	STATUSSEL[7:0]							
19	PINSEL.PINSEL[7:0]							

- **LENGTH_COUNT:** Number of samples over threshold
  - This value is used to track how many samples that have crossed the threshold (THRESH.RTHRESH and THRESH.FTHRESH), and is compared to THRESH.LENGTH. This field should be initialized to zero.
- **THRESH.LENGTH:** Threshold length
- **THRESH.DIR:** Threshold direction
- **THRESH.RTHRESH:** Threshold, integer part
- **THRESH.FTHRESH:** Threshold, fractional part
  - Refer to corresponding fields in Section 34.7.7 “Threshold Register” on page 936.
- **IDLE.RIDLE:** Sensor Idle Value, integer part
- **IDLE.FIDLE:** Sensor Idle Value, fractional part
  - Refer to corresponding fields in Section 34.7.3 “Sensor Idle Level” on page 932.
- **LEVEL.RLEVEL:** Sensor Relative Level, integer part
- **LEVEL.FLEVEL:** Sensor Relative Level, fractional part
  - Refer to corresponding fields in Section 34.7.4 “Sensor Relative Level” on page 933
- **CMPSTATE:** Comparator state
  - Internal state used by noise filter. This field should be initialized to zero.
- **TCHSTATE:** Touch state
  - State bit used for hysteresis.
    0: The sensor is out of touch.
    1: The sensor is in touch
- **TIMING.TIDLE:** Timing constant for the Idle value
- **TIMING.TLEVEL:** Timing constant for the relative level
  - Refer to corresponding fields in Section 34.7.6 “Filter Timing Register” on page 935.
- **RAW.RAWB:** Sensor raw value B
- **RAW.RAWA:** Sensor raw value A
  - Refer to corresponding fields in Section 34.7.5 “Sensor Raw Value” on page 934.
- **STATUSSEL:** Selection of status bit
  - The CATB only has one common interrupt for in- and out-of-touch events respectively. To assist the software to quickly determine which sensor caused a interrupt, additional bits in the In- and Out-of-Touch Status registers (INTCH and OUTTCH) can be updated. The value in this field select which bit that should be updated. Refer to the Module Configuration section for details on how many such bits that are implemented.
- **PINSEL.PINSEL:** Pin selection
  - Refer to corresponding field in Section 34.7.8 “Pin Selection Register” on page 937.

To enable DMA transfers, the user needs to setup the following:

1. Write the individual sensor configurations into an array in RAM, according to the layout in Table 34-2.
2. Enable the Peripheral DMA Controller in circular mode, pointing it to the RAM array. Two channels are needed: One for writing to the CATB, and one for reading from the CATB.

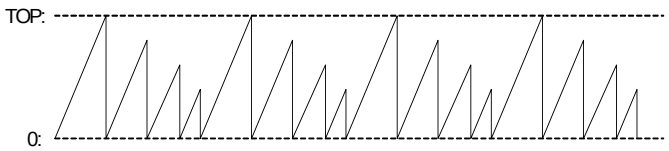
3. Configure common setup for all sensors (such as CNTCR.TOP, CNTCR.CHARGET, CNTCR.SPREAD).
4. Write a one to the DMA Enable (CR.DMAEN) and the CR.RUN bit.

Having a separate STATUSSEL and PINSEL allows the user to set up e.g. multiple thresholds on the same pin. This can be used for having multi-level threshold (proximity and touch), or for having different signs for the threshold, to capture both decrease and increase in capacitance on the same sensor.

34.6.8 Spread-Spectrum Operation

In order to reduce Electromagnetic Interference (EMI), a spread-spectrum mode of operation can be accomplished by adding jitter to the sampling frequency. The spread-spectrum operation also reduces the probability of periodic interference on the measurements.

Figure 34-6. Counter in Spread-Spectrum Operation



Writing a non-zero deviation value to the Spread Spectrum field (CNTCR.SPREAD) will cause the effective TOP value to alternate in a sawtooth pattern, with 16 different values ranging from $TOP - (16 \cdot 2^{SPREAD-1})$ to TOP. Figure 34-6 shows in principle how the CATB would vary the period if the spread spectrum generator varied between 4 different values instead of 16.

34.6.9 Interrupts

The CATB has three interrupt sources:

- SAMPLE - Sample Ready
  - New measurement sample ready since last write to ICR.SAMPLE.
- INTCH - In-Touch
  - In-touch event detected
- OUTTCH - Out-of-Touch
  - Out-of-touch event detected

Each interrupt source has a corresponding bit in the Interrupt Status Register (ISR).

An interrupt request will be generated if the bit in the Interrupt Status Register and the corresponding bit in the Interrupt Mask Register (IMR) are set. The interrupt sources are ORed together to form one interrupt request. The CATB will generate an interrupt request if at least one of the corresponding bits in IMR is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in ISR is cleared by writing a one to the corresponding bit in the Status Clear Register (SCR). Because all the interrupt sources are ORed together, the interrupt request from the CATB will remain active until all the bits in ISR are cleared.

34.6.10 Peripheral Event Triggered Operation

During normal operation, the CATB performs sensor sampling continuously. In the peripheral-event-triggered mode of operation, the CATB can receive periodic peripheral events, which trig-

ger one acquisition per event. The peripheral-event-triggered operation is enabled by writing a one to the Event Triggered Operation bit in the Control Register (CR.ETRIG). In this mode, the CATB will start the required clocks for performing an acquisition, and can thereby operate in all sleep modes where the peripheral event generator is running.

Using the peripheral-event-triggered mode allows the CATB to be used as a low-power wake-up source for the CPU, as interrupts can be generated on in-touch or out-of-touch events. It is not possible to use multiple sensors in sleep modes where the Peripheral DMA Controller is disabled.

Writing a one to the Event Triggered Operation bit in the Control Register (CR.ETRIG) will disable the continuous sampling and only sample when the CATB receives a peripheral event. The number of samples per peripheral event can be configured by writing to the Number of Event Samples field in the Control Register (CR.ESAMPLES). One peripheral event causes CR.ESAMPLES+1 samples to be acquired. If a monitored threshold is passed, the CATB can wake the device from sleep.

When the CATB receives a peripheral event during sleep mode, it automatically requests both the CLK_CATB and the CLK_ACQ clock. Note that it is not possible to use the peripheral-event-triggered operation with the GCLK as a source for CLK_ACQ in sleep modes where the GCLK is disabled. Refer to Section 10. “Power Manager (PM)” on page 109 for details. If no touch is detected after an acquisition, CLK_CATB and CLK_ACQ will be stopped, returning the device to low power mode.

34.7 User Interface

Table 34-3. CATB Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Read/Write	0x00000000
0x04	Counter Control Register	CNTCR	Read/Write	0x00000000
0x08	Sensor Idle Level	IDLE	Read/Write	0x00000000
0x0C	Sensor Relative Level	LEVEL	Read-only	0x00000000
0x10	Sensor Raw Value	RAW	Read-only	0x00000000
0x14	Filter Timing Register	TIMING	Read/Write	0x00000000
0x18	Threshold Register	THRESH	Read/Write	0x00000000
0x1C	Pin Selection Register	PINSEL	Read/Write	0x00000000
0x20	Direct Memory Access Register	DMA	Read/Write	0x00000000
0x24	Interrupt Status Register	ISR	Read-only	0x00000000
0x28	Interrupt Enable Register	IER	Write-only	0x00000000
0x2C	Interrupt Disable Register	IDR	Write-only	0x00000000
0x30	Interrupt Mask Register	IMR	Read-only	0x00000000
0x34	Status Clear Register	SCR	Write-only	0x00000000
0x40+4n	In-Touch Status Register n	INTCHn	Read-only	0x00000000
0x50+4n	In-Touch Status Clear Register n	INTCHCLRn	Write-only	0x00000000
0x60+4n	Out-of-Touch Status Register n	OUTTCHn	Read-only	0x00000000
0x70+4n	Out-of-Touch Status Clear Register n	OUTTCHCLRn	Write-only	0x00000000
0xF8	Parameter Register	PARAMETER	Read-only	-(1)
0xFC	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

34.7.1 Control Register

Name: CR

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
SWRST	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	CHARGET			
15	14	13	12	11	10	9	8
-	ESAMPLES						
7	6	5	4	3	2	1	0
DMAEN	DIFF	CKSEL	INTRES	ETRIG	IIDLE	RUN	EN

- **SWRST: Software Reset**
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets the CATB. The CATB will be disabled after the reset.
  This bit always reads as zero.
- **CHARGET: Charge Time**
  The measured capacitance is charged for 2^(CHARGET) clock cycles of the acquisition clock (selected with CKSEL) before each discharge measurement. This field must not be changed when the CATB is operating (CR.RUN is one).
- **ESAMPLES: Number of Event Samples**
  When the CATB is driven by the Peripheral Event System, the amount of samples per event equal ESAMPLES +1.
- **DMAEN: DMA Enable**
  0: DMA is not used. Single sensor mode of operation.
  1: DMA is being used for multi sensor operation. Sensor configuration data is rotated with data from RAM.
- **DIFF: Differential Mode**
  0: Normal acquisition mode. One pin is used per sensor (plus the shared discharge pin (DIS) if INTRES is zero.
  1: Differential mode. Two pins are used per sensor.
  Note: This bit must not be changed during operation (RUN is one).
- **CKSEL: Clock Select**
  0: The RC oscillator is selected as acquisition clock and is used as timing source for the measurements.
  1: The GCLK is selected as acquisition clock and is used as the timing source for measurements.
  Refer to the Module Configuration section for configuration of clocks.
  Note: This bit must not be changed during operation (RUN is one).
- **INTRES: Internal Resistors**
  0: External resistors are used between DIS and SENSE pins, or between sensor-pads if DIFF=1.
  1: Internal resistors are used.
  Note: This bit must not be changed during operation (RUN=1).

- **ETRIG: Event Triggered Operation**
  0: Normal continuous acquisition mode of operation.
  1: CATB acquisitions are triggered by the Peripheral Event System.
- **IIDLE: Initialize Idle Value**
  0: The IDLE register will only be updated by the CATB, or if written to.
  1: The IDLE register is set to the next measured value.
  This bit always reads as zero.
- **RUN: Start Operation**
  Writing a zero to this bit will stop the CATB.
  Writing a one to this bit will usually start acquisitions immediately. If CR.DMAEN is one, the CATB will wait until the first DMA transfer. If CR.ETRIG is one, the CATB will wait for a peripheral event.
  0: CATB is stopped.
  1: CATB is running.
- **EN: Access Enable**
  0: The CATB interface is disabled, register configuration settings can not be changed.
  1: The CATB interface is enabled, register configuration settings can be changed.

34.7.2 Counter Control Register

Name: CNTCR

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	REPEAT			SPREAD			

23	22	21	20	19	18	17	16
TOP[23:16]							

15	14	13	12	11	10	9	8
TOP[15:8]							

7	6	5	4	3	2	1	0
TOP[7:0]							

Note: This register must not be written to during CATB operation (CR.RUN is one).

- **REPEAT: Repeat Measurements**
  Measurements are repeated REPEAT+1 times in the acqusition block. The individual measurements are summed before being sent to the filter.
- **SPREAD: Spread Spectrum**
  0: Normal mode of operation. Spread spectrum mode of operation is not used.
  1-15: Spread-spectrum mode of operation. The effective TOP value deviation ranges from $TOP - (16 \cdot 2^{SPREAD-1})$ to TOP.
- **TOP: Counter Top Value**
  Determines sample rate.

34.7.3 Sensor Idle Level

Name: IDLE

Access Type: Read/Write

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	RIDLE[15:12]			

23	22	21	20	19	18	17	16
RIDLE[11:4]							

15	14	13	12	11	10	9	8
RIDLE[3:0]				FIDLE[11:8]			

7	6	5	4	3	2	1	0
FIDLE[7:0]							

This register value is subtracted from initial measurement data in order to determine the relative capacitance. The IDLE value is updated as a moving exponential weighted average of previous measurements, with the time-constant given in TIMING.TIDLE. This register value is updated when ISR.SAMPLE is set.

- **RIDLE: Integer Sensor Idle**
  This field contains the integer part of the measured idle-signal value.
- **FIDLE: Fractional Sensor Idle**
  This field contains the fractional part of the measured idle-signal value. Note that some of the LSB's of the fractional part might not be implemented. Refer to the Module Configuration section.

34.7.4 Sensor Relative Level

Name: LEVEL

Access Type: Read-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	RLEVEL[7:4]			

15	14	13	12	11	10	9	8
RLEVEL[3:0]				FLEVEL[11:8]			

7	6	5	4	3	2	1	0
FLEVEL[7:0]							

This register contains the latest sensor measurement value. It has been filtered, subtracted from the IDLE register values, and smoothened. This value is compared to the THRESH register values to determine whether the sensor is in touch or not. Filter smoothening is determined by the TIMING.TLEVEL value. When this register is updated, ISR.SAMPLE is set by the CATB.

- **RLEVEL: Integer Sensor Level**
  This field contains the integer part of the measured sensor discharge value.
- **FLEVEL: Fractional Sensor Level**
  This field contains the fractional part of the measured sensor discharge value. Note that some of the LSB's of the fractional part might not be implemented. Refer to the Module Configuration section.

34.7.5 Sensor Raw Value

Name: RAW

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

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

- **RAWB: Last Sensor Raw Value**
  The unfiltered counter value from last acquisition.
- **RAWA: Current Sensor Raw Value**
  The unfiltered counter value from last acquisition, with IDLE subtracted.

34.7.6 Filter Timing Register

Name: TIMING

Access Type: Read/Write

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	TIDLE[11:8]			

23	22	21	20	19	18	17	16
TIDLE[7:0]							

15	14	13	12	11	10	9	8
-	-	-	-	TLEVEL[11:8]			

7	6	5	4	3	2	1	0
TLEVEL[7:0]							

- **TIDLE: Idle Smoothening**
  This field determines how fast Idle tracker should track the measured samples. Note that some of the LSB's of this field might not be implemented. Refer to the Module Configuration section.
- **TLEVEL: Relative Level Smoothening**
  The field determines the smoothening factor to the averaging filter. Note that some of the LSB's of this field might not be implemented. Refer to the Module Configuration section.

34.7.7 Threshold Register

Name: THRESH

Access Type: Read/Write

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	LENGTH				

23	22	21	20	19	18	17	16
DIR	-	-	-	RTHRESH[7:4]			

15	14	13	12	11	10	9	8
RTHRESH[3:0]				FTHRESH[11:8]			

7	6	5	4	3	2	1	0
FTHRESH[7:0]							

- **LENGTH: Threshold Length**
  The amount of successive samples that have to be beyond the threshold before a in- or out-of-touch event is detected.
- **DIR: Threshold Direction**
  0: Touch is detected when LEVEL goes above threshold value.
  1: Touch is detected when LEVEL goes below threshold value.
- **RTHRESH: Integer Threshold Value**
  The integer part of the threshold value. Note that some of the LSB's of this field might not be implemented. Refer to the Module Configuration section.
- **FTHRESH: Fractional Threshold Value**
  The fractional part of the threshold value. Note that some of the LSB's of this field might not be implemented. Refer to the Module Configuration section.

34.7.8 Pin Selection Register

Name: PINSEL

Access Type: Read/Write

Offset: 0x1C

Reset Value: 0x00000000

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

The number of available bits in this register is device dependent. Refer to the Module Configuration section for details.

- **PINSEL: Pin Select**
  The pin selected for touch acquisition. In differential mode (CR.DIFF is one), two pins are selected when writing *m* to PINSEL: SENSE[m] and SENSE[m+1].

34.7.9 Direct Memory Access Register

Name: DMA

Access Type: Read/Write

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DMA[31:24]							

23	22	21	20	19	18	17	16
DMA[23:16]							

15	14	13	12	11	10	9	8
DMA[15:8]							

7	6	5	4	3	2	1	0
DMA[7:0]							

Note: This register is used for DMA transfers between the CATB and RAM. This register must not be read or written by the user.

34.7.10 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	OUTTCH	INTCH	SAMPLE

Bits in this register are cleared by writing a one to the corresponding bit in the Status Clear Register (SCR).

- **OUTTCH: Out-of-Touch**
  0: No out-of-touch event detected.
  1: Out-of-touch event detected.
- **INTCH: In-touch**
  0: No in-touch event detected.
  1: In-touch event detected.
- **SAMPLE: Sample Ready**
  0: No new measurement sample ready since last write to IDR.SAMPLE.
  1: New measurement sample ready since last write to SCR.SAMPLE. The values in the LEVEL, RAW, and IDLE registers have been updated.

34.7.11 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x28

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

34.7.12 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x2C

Reset Value: 0x00000000

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

34.7.13 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x30

Reset Value: 0x00000000

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

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

34.7.14 Status Clear Register

Name: SCR

Access Type: Write-only

Offset: 0x34

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	OUTTCH	INTCH	SAMPLE

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR and the corresponding interrupt request.

34.7.15 In-Touch Status Register i

Name: INTCHi

Access Type: Read-only

Offset: 0x40+4i

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
INTCH31	INTCH30	INTCH29	INTCH28	INTCH27	INTCH26	INTCH25	INTCH24
23	22	21	20	19	18	17	16
INTCH23	INTCH22	INTCH21	INTCH20	INTCH19	INTCH18	INTCH17	INTCH16
15	14	13	12	11	10	9	8
INTCH15	INTCH14	INTCH13	INTCH12	INTCH11	INTCH10	INTCH9	INTCH8
7	6	5	4	3	2	1	0
INTCH7	INTCH6	INTCH5	INTCH4	INTCH3	INTCH2	INTCH1	INTCH0

- **INTCHj: In-Touch**

  These bits provides status when the CATB operates in DMA mode. Each block of configuration data (see Table 34-2), can choose in the STATUSSEL field which bit in this register that should be set when an in-touch event is detected. A STATUSSEL value of 0 will choose the INTCH0 bit in the INTCH0 register (i ==0 and j==0), while a STATUSSEL of 42 will choose the INTTCH10 bit in the INTCH1 register (i ==1 and j==10). Refer to the Module Configuration section for details of the number of bits implemented in theese registers.

  0: DMA configuration with STATUSSEL set to i*32+j has not registered an in-touch event.

  1: DMA configuration with STATUSSEL set to i*32+j has registered an in-touch event.

34.7.16 In-Touch Status Clear Register n

Name: INTCHCLRn

Access Type: Write-only

Offset: 0x50+4n

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
INTCHCLR31	INTCHCLR30	INTCHCLR29	INTCHCLR28	INTCHCLR27	INTCHCLR26	INTCHCLR25	INTCHCLR24
23	22	21	20	19	18	17	16
INTCHCLR23	INTCHCLR22	INTCHCLR21	INTCHCLR20	INTCHCLR19	INTCHCLR18	INTCHCLR17	INTCHCLR16
15	14	13	12	11	10	9	8
INTCHCLR15	INTCHCLR14	INTCHCLR13	INTCHCLR12	INTCHCLR11	INTCHCLR10	INTCHCLR9	INTCHCLR8
7	6	5	4	3	2	1	0
INTCHCLR7	INTCHCLR6	INTCHCLR5	INTCHCLR4	INTCHCLR3	INTCHCLR2	INTCHCLR1	INTCHCLR0

- **INTCHCLRm: In-Touch Clear**
  Writing a one to a bit in this register will clear the corresponding bit in INTCHn.INTCHm.

34.7.17 Out-of-Touch Status Register i

Name: OUTTCHi

Access Type: Read-only

Offset: 0x60+4i

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
OUTTCH31	OUTTCH30	OUTTCH29	OUTTCH28	OUTTCH27	OUTTCH26	OUTTCH25	OUTTCH24
23	22	21	20	19	18	17	16
OUTTCH23	OUTTCH22	OUTTCH21	OUTTCH20	OUTTCH19	OUTTCH18	OUTTCH17	OUTTCH16
15	14	13	12	11	10	9	8
OUTTCH15	OUTTCH14	OUTTCH13	OUTTCH12	OUTTCH11	OUTTCH10	OUTTCH9	OUTTCH8
7	6	5	4	3	2	1	0
OUTTCH7	OUTTCH6	OUTTCH5	OUTTCH4	OUTTCH3	OUTTCH2	OUTTCH1	OUTTCH0

- **OUTTCHj: Out-of-Touch**

  These bits provides status when the CATB operates in DMA mode. Each block of configuration data (see Table 34-2), can choose in the STATUSSEL field which bit in this register that should be set when an out-of-touch event is detected. A STATUSSEL value of 0 will choose the OUTTCH0 bit in the OUTTCH0 register (i ==0 and j==0), while a STATUSSEL of 42 will choose the OUTTCH10 bit in the OUTTCH1 register (i ==1 and j==10). Refer to the Module Configuration section for details of the number of bits implemented in theese registers.

  0: DMA configuration with STATUSSEL set to i*32+j has not registered an out-of-touch event.

  1: DMA configuration with STATUSSEL set to i*32+j has registered an out-of-touch event.

34.7.18 Out of Touch Status Clear Register n

Name: OUTTCHCLRn

Access Type: Write-only

Offset: 0x70+4n

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
OUTTCHCLR31	OUTTCHCLR30	OUTTCHCLR29	OUTTCHCLR28	OUTTCHCLR27	OUTTCHCLR26	OUTTCHCLR25	OUTTCHCLR24
23	22	21	20	19	18	17	16
OUTTCHCLR23	OUTTCHCLR22	OUTTCHCLR21	OUTTCHCLR20	OUTTCHCLR19	OUTTCHCLR18	OUTTCHCLR17	OUTTCHCLR16
15	14	13	12	11	10	9	8
OUTTCHCLR15	OUTTCHCLR14	OUTTCHCLR13	OUTTCHCLR12	OUTTCHCLR11	OUTTCHCLR10	OUTTCHCLR9	OUTTCHCLR8
7	6	5	4	3	2	1	0
OUTTCHCLR7	OUTTCHCLR6	OUTTCHCLR5	OUTTCHCLR4	OUTTCHCLR3	OUTTCHCLR2	OUTTCHCLR1	OUTTCHCLR0

- **OUTTCHCLRm: Out of Touch**
  Writing a one to a bit in this register will clear the corresponding bit in OUTTCHn.OUTTCHm.

34.7.19 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0xF8

Reset Value: -

31	30	29	28	27	26	25	24
	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	FRACTIONAL			

15	14	13	12	11	10	9	8
NSTATUS							

7	6	5	4	3	2	1	0
NPINS							

- **FRACTIONAL: Number of Fractional bits**
  The number of implemented fractional bits.
- **NSTATUS: Number of Status bits**
  The number of implemented status bits.
- **NPINS: Number of Pins**
  The number of connected pins.

34.7.20 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			
15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			
7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**

  Reserved. No functionality associated.

- **VERSION: Version Number**

  Version number of the module. No functionality associated.

34.8 Module Configuration

The specific configuration for each CATB instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 34-4. CATB Configuration

Feature	CATB
Number of sensors/connected pins	32
Number of status bits in STATUSSEL	32
Number of fractional bits implemented in IDLE, LEVEL, TIMING, and THRESH.	10
Number of available bits in the PINSEL register	5

Table 34-5. CATB Clocks

Clock Name	Description
CLK_CATB	Clock for the CATB bus interface
RC Oscillator	Source clock for the CLK_ACQ
GCLK	Source clock for the CLK_ACQ. The generic clock used for the CATB is GCLK3

Table 34-6. Register Reset Values

Register	Reset Value
VERSION	0x00000100
PARAMETER	0x000A2020

35. True Random Number Generator (TRNG)

Rev: 1.0.3.0

35.1 Features

- **Passed NIST Special Publication 800-22 Tests Suite**
- **Passed Diehard Random Tests Suite**
- **Provides a 32-bit Random Number Every 84 Clock Cycles**

35.2 Overview

The True Random Number Generator provides 32-bit random numbers.

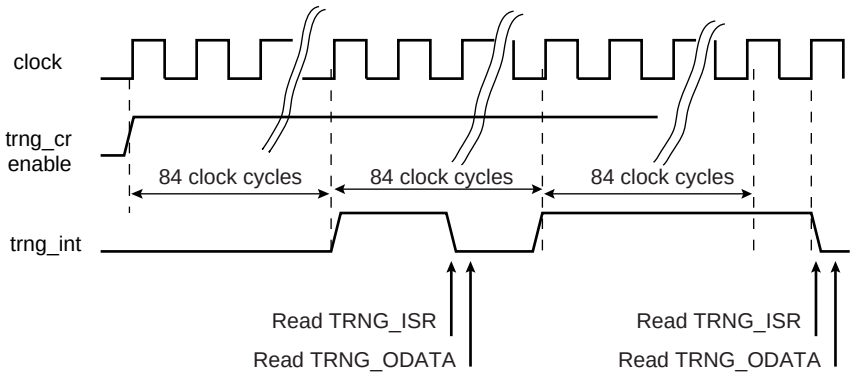
35.3 Functional Description

The True Random Number Generator (TRNG) passes the American *NIST Special Publication 800-22 and Diehard Random Tests Suites.*

As soon as the TRNG is enabled (writing the ENABLE bit in the CR register with the correct KEY), the generator provides one 32-bit value every 84 clock cycles. An Interrupt can be enabled through the IER register (respectively disabled in IDR). This interrupt is set when a new random value is available and is cleared when the Interrupt Status Register is read (ISR). The bit DATRDY in ISR is set when the random data is ready to be read out on the 32-bit output data register (ODATA).

The user should check that the DATRDY bit is one before readin the ODATA register when a 32-bit random value is required.

Figure 35-1. TRNG Data Generation Sequence



35.4 User Interface

Table 35-1. TRNG Register Memory Map

Offset	Register	Name	Access	Reset
0x00	Control Register	CR	Write-only	–
0x10	Interrupt Enable Register	IER	Write-only	–
0x14	Interrupt Disable Register	IDR	Write-only	–
0x18	Interrupt Mask Register	IMR	Read-only	0x0000_0000
0x1C	Interrupt Status Register	ISR	Read-only	0x0000_0000
0x50	Output Data Register	ODATA	Read-only	0x0000_0000
0xFC	Version Register	VERSION	Read-only	–[1]

Notes: 1. Values in the Version Register vary with the version of the IP block implementation.

35.4.1 Control Register

Name: CR

Access Type: Write-only

Offset: 0x00

Reset Value: –

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

- **KEY: Security Key**

KEY = 0x524e47 (RNG in ASCII)

This field must be written to 0x524E47 for a write operation to be effective.

- **ENABLE: Enables the TRNG to provide random values**

Writing a zero to this bit disables the TRNG.

Writing a one to this bit enables the TRNG.

35.4.2 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x10

Reset Value: –

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR

35.4.3 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x14

Reset Value: –

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR

35.4.4 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x18

Reset Value: 0x00000000

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

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
This bit is cleared when the corresponding bit in IDR is written to one.
This bit is set when the corresponding bit in IER is written to one.

35.4.5 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x34

Reset Value: 0x00000000

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

- **DATRDY: Data Ready**

0 = Output data is not valid or TRNG is disabled.

1 = New Random value is completed.

DATRDY is cleared when this register is read.

35.4.6 Output Data Register

Name: ODATA

Access Type: Read-only

Offset: 0x34

Reset Value: 0x00000000

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

- **ODATA: Output Data**

The 32-bit Output Data register contains the 32-bit random data.

35.4.7 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

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

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**

Version number of the module. No functionality associated.

35.5 Module Configuration

The specific configuration for each TRNG instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 35-2. TRNG Clock Name

Module Name	Clock Name	Description
TRNG	CLK_TRNG	Peripheral clock for TRNG

Table 35-3. Register Reset Values

Register	Reset Value
VERSION	0x00000103

36. Glue Logic Controller (GLOC)

Rev: 1.0.2.0

36.1 Features

- **Glue logic for general purpose PCB design**
- **Programmable lookup table**
- **Up to four inputs supported per lookup table**
- **Optional filtering of output**

36.2 Overview

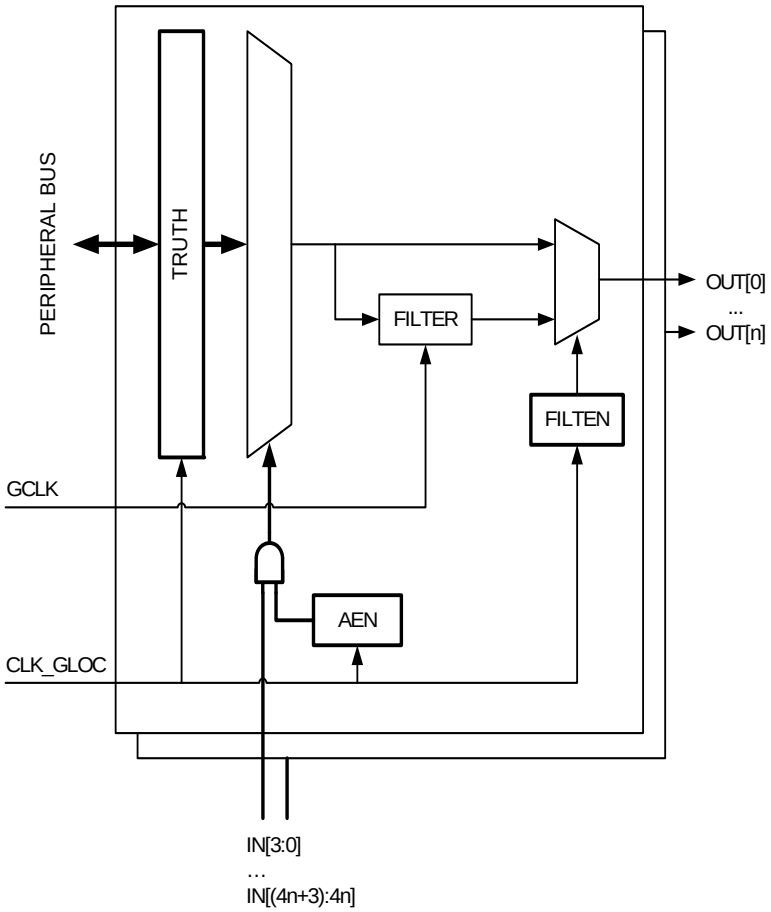
The Glue Logic Controller (GLOC) contains programmable logic which can be connected to the device pins. This allows the user to eliminate logic gates for simple glue logic functions on the PCB.

The GLOC consists of a number of lookup table (LUT) units. Each LUT can generate an output as a user programmable logic expression with four inputs. Inputs can be individually masked.

The output can be combinatorially generated from the inputs, or filtered to remove spikes.

36.3 Block Diagram

Figure 36-1. GLOC Block Diagram



36.4 I/O Lines Description

Table 36-1. I/O Lines Description

Pin Name	Pin Description	Type
IN0-INm	Inputs to lookup tables	Input
OUT0-OUTn	Output from lookup tables	Output

Each LUT have 4 inputs and one output. The inputs and outputs for the LUTs are mapped sequentially to the inputs and outputs. This means that LUT0 is connected to IN0 to IN3 and OUT0. LUT1 is connected to IN4 to IN7 and OUT1. In general, LUTn is connected to IN[4n] to IN[4n+3] and OUTn.

36.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

36.5.1 I/O Lines

The pins used for interfacing the GLOC may be multiplexed with I/O Controller lines. The programmer must first program the I/O Controller to assign the desired GLOC pins to their peripheral function. If I/O lines of the GLOC are not used by the application, they can be used for other purposes by the I/O Controller.

It is only required to enable the GLOC inputs and outputs in use. Pullups for pins configured to be used by the GLOC will be disabled.

36.5.2 Clocks

The clock for the GLOC bus interface (CLK_GLOC) is generated by the Power Manager. This clock is disabled at reset, and can be enabled in the Power Manager. It is recommended to disable the GLOC before disabling the clock, to avoid freezing the module in an undefined state.

Additionally, the GLOC depends on a dedicated Generic Clock (GCLK). The GCLK can be set to a wide range of frequencies and clock sources, and must be enabled by the System Control Interface (SCIF) before the GLOC filter can be used.

36.5.3 Debug Operation

When an external debugger forces the CPU into debug mode, the GLOC continues normal operation.

36.6 Functional Description

36.6.1 Enabling the Lookup Table Inputs

Since the inputs to each lookup table (LUT) unit can be multiplexed with other peripherals, each input must be explicitly enabled by writing a one to the corresponding enable bit (AEN) in the corresponding Control Register (CR).

If no inputs are enabled, the output OUTn will be the least significant bit in the TRUTHn register.

36.6.2 Configuring the Lookup Table

The lookup table in each LUT unit can generate any logic expression OUT as a function of up to four inputs, IN[3:0]. The truth table for the expression is written to the TRUTH register for the LUT. Table 36-2 shows the truth table for LUT0. The truth table for LUTn is written to TRUTHn, and the corresponding input and outputs will be IN[4n] to IN[4n+3] and OUTn.

Table 36-2. Truth Table for the Lookup Table in LUT0

IN[3]	IN[2]	IN[1]	IN[0]	OUT[0]
0	0	0	0	TRUTH0[0]
0	0	0	1	TRUTH0[1]
0	0	1	0	TRUTH0[2]
0	0	1	1	TRUTH0[3]
0	1	0	0	TRUTH0[4]
0	1	0	1	TRUTH0[5]
0	1	1	0	TRUTH0[6]
0	1	1	1	TRUTH0[7]
1	0	0	0	TRUTH0[8]
1	0	0	1	TRUTH0[9]
1	0	1	0	TRUTH0[10]
1	0	1	1	TRUTH0[11]
1	1	0	0	TRUTH0[12]
1	1	0	1	TRUTH0[13]
1	1	1	0	TRUTH0[14]
1	1	1	1	TRUTH0[15]

36.6.3 Output Filter

By default, the output OUTn is a combinatorial function of the inputs IN[4n] to IN[4n+3]. This may cause some short glitches to occur when the inputs change value.

It is also possible to clock the output through a filter to remove glitches. This requires that the corresponding generic clock (GCLK) has been enabled before use. The filter can then be enabled by writing a one to the Filter Enable (FILTEN) bit in CRn. The OUTn output will be delayed by three to four GCLK cycles when the filter is enabled.

36.7 User Interface

Table 36-3. GLOC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00+n*0x08	Control Register n	CRn	Read/Write	0x00000000
0x04+n*0x08	Truth Table Register n	TRUTHn	Read/Write	0x00000000
0x38	Parameter Register	PARAMETER	Read-only	- (1)
0x3C	Version Register	VERSION	Read-only	- (1)

Note: 1. The reset values are device specific. Refer to the Module Configuration section at the end of this chapter.

36.7.1 Control Register n

Name: CRn

Access Type: Read/Write

Offset: 0x00+n*0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
FILTEN	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	AEN			

- **FILTEN: Filter Enable**
  1: The output is glitch filtered.
  0: The output is not glitch filtered.
- **AEN: Enable IN Inputs**
  Input IN[n] is enabled when AEN[n] is one.
  Input IN[n] is disabled when AEN[n] is zero, and will not affect the OUT value.

36.7.2 Truth Table Register n

Name: TRUTHn

Access Type: Read/Write

Offset: 0x04+n*0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
TRUTH[15:8]							

7	6	5	4	3	2	1	0
TRUTH[7:0]							

- **TRUTH: Truth Table Value**

  This value defines the output OUT as a function of inputs IN:
  OUT = TRUTH[IN]

36.7.3 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0x38

Reset Value: -

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

- **LUTS: Lookup Table Units Implemented**
  This field contains the number of lookup table units implemented in this device.

36.7.4 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x3C

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			
15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			
7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

36.8 Module Configuration

The specific configuration for each GLOC instance is listed in the following tables.The GLOC bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 36-4. GLOC Configuration

Feature	GLOC
Number of LUT units	2

Table 36-5. GLOC Clocks

Clock Name	Description
CLK_GLOC	Clock for the GLOC bus interface
GCLK	The generic clock used for the GLOC is GCLK5

Table 36-6. Register Reset Values

Register	Reset Value
VERSION	0x00000102
PARAMETER	0x00000002

37. Analog Comparator Interface (ACIFC)

Rev: 1.0.0.0

37.1 Features

- **Controls an array of Analog Comparators**
- **Low power option**
  - **Single shot mode support**
- **Window mode**
  - **Detect inside/outside window**
  - **Detect above/below window**
- **Interrupt**
  - **On comparator result rising edge, falling edge, toggle, comparison done**
  - **Inside window, outside window, toggle**
  - **When startup time is over**
- **Peripheral events**
  - **Comparators triggered by incoming peripheral events**
  - **Configurable generated peripheral events**

37.2 Overview

The Analog Comparator Interface (ACIFC) controls a number of Analog Comparators (AC) with identical behavior. Each Analog Comparator compares two voltages and gives a compare output depending on this comparison.

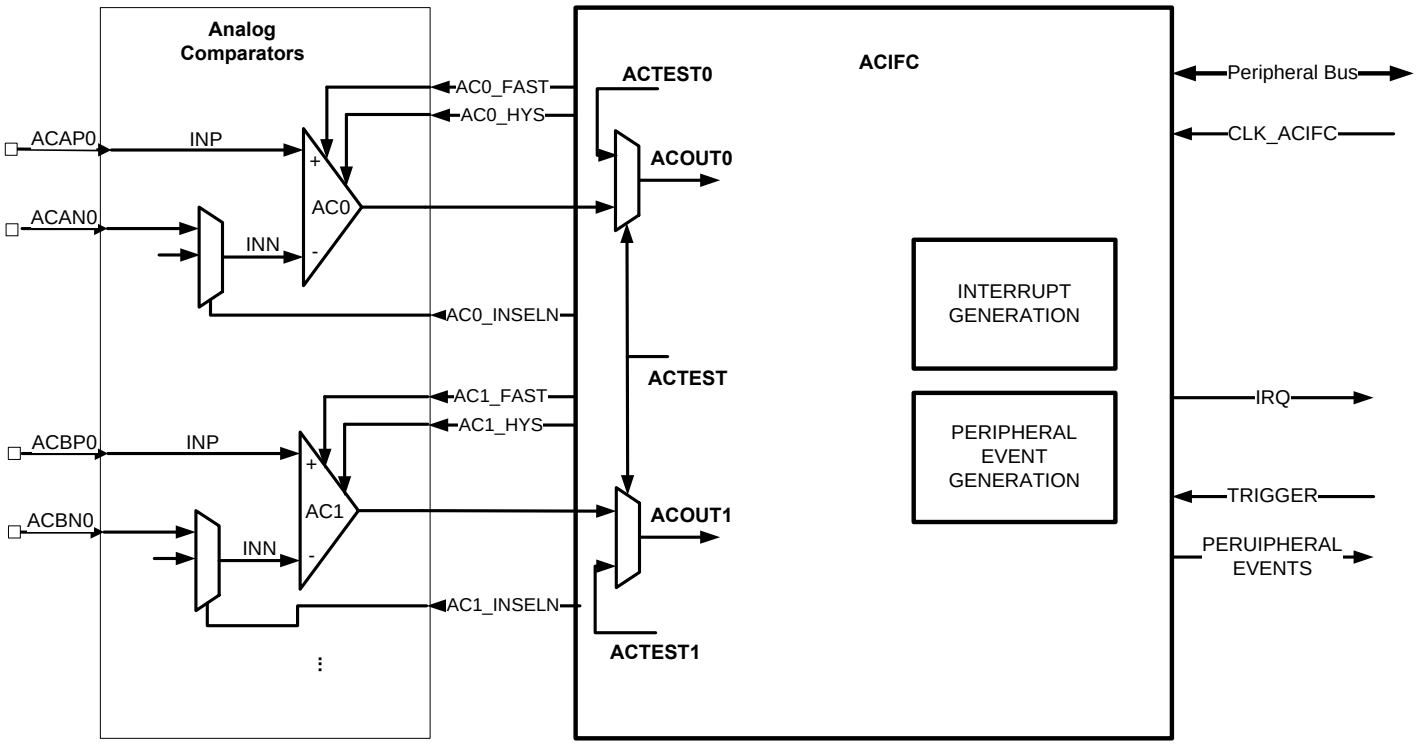
A specific AC is referred to as ACx where x is any number from 0 to n and n is the index of last AC module (refer to the Module Configuration section at the end of this chapter for details).

The ACIFC can be configured in normal mode using each comparator independently or in window mode using defined comparator pairs (ACx and ACx+1) to observe a window.

The number of Analog Comparators implemented is device specific. Refer to the Module Configuration section for details.

37.3 Block Diagram

Figure 37-1. ACIFC Block Diagram



37.4 I/O Lines Description

Table 37-1. I/O Line Description

Pin Name	Pin Description	Type
ACAPx	Positive reference pin for Analog Comparator x	Analog
ACANx	Negative reference pin for Analog Comparator x	Analog
ACBPx	Positive reference pin for Analog Comparator x	Analog
ACBNx	Negative reference pin for Analog Comparator x	Analog

Table 37-2. Pin Mapping

Channel	Pins (Normal Mode)	Window pair	Pins (Window Mode)
0	ACAP0, ACAN0	0	ACAP0, ACBN0
1	ACBP0, ACBN0		
2	ACAP1, ACAN1	1	ACAP1, ACBN1
3	ACBP1, ACBN1		

Table 37-2. Pin Mapping

Channel	Pins (Normal Mode)	Window pair	Pins (Window Mode)
4	ACAP2, ACAN2	2	ACAP2, ACBN2
5	ACBP2, ACBN2		
6	ACAP3, ACAN3	3	ACAP3, ACBN3
7	ACBP3, ACBN3		

37.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

37.5.1 I/O Lines

The ACIFC pins are multiplexed with other peripherals. The user must first configure the I/O Controller to give control of the pins to the ACIFC.

37.5.2 Power Management

ACIFC stops functioning when the system enters a sleep mode that disables its clock. However, ACIFC can resume its operation if the system enters the SleepWalking mode and the ACIFC clock is started by the Peripheral Event System (see Section 10. “Power Manager (PM)” on page 109 for details on the SleepWalking mode). During this time, if ACIFC generates an interrupt, the system will wake up from sleep mode and normal system operation will resume.

37.5.3 Clocks

The clock for ACIFC (CLK_ACIFC) is generated by the Power Manager. It can be disabled either manually through the user interface of the Power Manager or automatically when the system enters a sleep mode that disables the clocks to the peripheral bus modules.

37.5.4 Interrupts

The ACIFC interrupt request line is connected to the NVIC. Using the ACIFC interrupt requires the NVIC to be configured first.

37.5.5 Peripheral Events

The ACIFC peripheral events are connected via the Peripheral Event System. Refer to Section 31. “Peripheral Event Controller (PEVC)” on page 845 for details.

37.5.6 Debug Operation

When an external debugger forces the CPU into debug mode, the ACIFC continues normal operation. If the ACIFC is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

37.6 Functional Description

The ACIFC is enabled by writing a one to the Control Register Enable bit (CTRL.EN). Additionally, the comparators must be individually enabled byselecting a measurement mode by writing to the MODE field in the ACx Configuration Register (CONFx.MODE).

The results from the individual comparators can either be used directly (normal mode), or the results from two comparators can be grouped to generate a comparison window (window mode). All comparators does not have to be in the same mode, some comparators may be in normal

mode, while others are in window mode. There are restrictions on which ACs can be grouped in a window mode, see Section 37.6.5.

37.6.1 Analog Comparator Operation

Each AC can be in one of four measurement modes, determined by CONFx.MODE:

- No Measurement
- Continuous Measurement mode (CM)
- User Triggered Single Measurement mode (UT)
- Peripheral Event Triggered Single Measurement mode (ET)

An AC is disabled between measurements if the Always On bit in the corresponding CONFx register (CONFx.ALWAYSON) is zero. This is the default value after reset. This reduces the AC power dissipation between measurements, but the AC output will not be available until after an AC startup time. If the ACx Ready bit in the Status Register (SR.ACRDYx) is one, the output of ACx is ready. In window mode the result is available when both the comparator outputs are ready (SR.ACRDYx=1 and SR.ACRDYx+1=1).

An AC is always enabled (i.e., during and between measurements) if the corresponding CONFx.ALWAYSON bit is one. This allows a measurement on AC to be made quickly after a measurement is triggered, without waiting for the AC startup time (except for the first startup after reset).

37.6.1.1 Continuous Measurement Mode

In the CM mode, AC is performing comparisons continuously. Therefore the result of the latest comparison is always available in the ACx Current Comparison Status bit in the Status Register (SR.ACCSx). Comparisons result are updated on every positive edge of CLK_ACIFC.

CM is enabled by writing CONFx.MODE to one. When the corresponding AC is ready, SR.ACRDYx is set and comparison status is reflected in SR.ACCSx.

Corresponding peripheral events and interrupts are generated if enabled. New comparisons are performed continuously until the CONFx.MODE field is written to zero.

37.6.1.2 User Triggered Single Measurement Mode

In the UT mode, the user starts a single comparison by writing a one to the User Start Single Comparison bit (CTRL.USTART).

This mode is enabled by writing CONFx.MODE to 2. Once the corresponding AC is ready, user can start a single comparison then SR.ACCSx is updated accordingly. CTRL.USTART is cleared automatically by hardware when the single comparison has been done.

Corresponding peripheral events and interrupts are generated if enabled.

37.6.1.3 Peripheral Event Triggered Single Measurement Mode

This mode is enabled by writing CONFx.MODE to 3 and the Peripheral Event Trigger Enable bit in CTRL (CTRL.EVENTEN) to one. The ET mode is similar to the UT mode, the difference is that a peripheral event from another hardware module causes the hardware to automatically set the Peripheral Event Start Single Comparison bit (CTRL.ESTART). Once the corresponding AC is ready, an event starts a single comparison then SR.ACCSx is updated accordingly. CTRL.ESTART is cleared automatically by hardware when the single comparison has been done.

Corresponding peripheral events and interrupts are generated if enabled. No new comparisons will be performed.

37.6.1.4 Selecting Comparator Inputs

Each Analog Comparator has one positive (INP) and one negative (INN) input. The positive input is fed from an external input pin (ACAPx, ACBPx).

There are several sources for negative input including external input pin ACA/BNx. The Negative Input Select field (CONFx.INSELN) selects the source for the negative input. See *CONFx.INSELN: Negative Input Select* table in Module Configuration section.

Note that in window mode, the negative input ACANx and positive input ACBPx+1 must have the same source, as shown in Figure 37-2.

37.6.2 Interrupt Generation

The ACIFC has several interrupt sources. The status of each interrupt source can be read from the Interrupt Status Register (ISR). An interrupt request will be generated if a bit in ISR and the corresponding bit in the Interrupt Mask Register (IMR) are set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in ISR is cleared by writing a one to the corresponding bit in the Interrupt Status Clear Register (ICR).

The interrupt sources are ORed together to form one interrupt request. The ACIFC will generate an interrupt request if at least one the bits in IMR is set. Because all the interrupt sources are ORed togheter, the interrupt request from the ACIFC will remain active utnil all the bits in ISR are cleared.

37.6.3 Peripheral Event Generation

The ACIFC can be set up so that certain comparison results notify other parts of the device via the Peripheral Event system. Refer to Section 37.6.4.3 and Section 37.6.5.3 for information on which comparison results can generate peripheral events, and how to configure the ACIFC to achieve this.

Zero or one peripheral event will be generated per comparison.

37.6.4 Normal Mode

In normal mode all Analog Comparators are operating independently.

37.6.4.1 Normal Mode Output

Each Analog Comparator generates one output ACOUT according to the input voltages on INP (AC positive input) and INN (AC negative input):

- ACOUT = 1 if $V_{INP} > V_{INN}$
- ACOUT = 0 if $V_{INP} < V_{INN}$
- ACOUT = 0 if the AC output is not available (SR.ACRDY = 0)

The output can optionally be filtered, as described in Section 37.6.6.

37.6.4.2 Normal Mode Interrupt

There are two types of interrupt for each AC:

- Startup time interrupt (bit ISR.SUTINTx), each AC can generate an interrupt when the startup time is over.
- AC output based interrupt (bit ISR.ACINTx). In normal mode, the following interrupt sources are available:
  - When $V_{INP} > V_{INN}$
  - When $V_{INP} < V_{INN}$
  - On toggle of the AC output (ACOUT)
  - When comparison has been done

The user selects the interrupt source by writing to the Interrupt Settings field in the ACx Configuration Register (CONFx.IS).

37.6.4.3 Normal Mode Peripheral Events

The ACIFC can generate peripheral events according to the configuration of CONFx.EVENN and CONFx.EVENP.

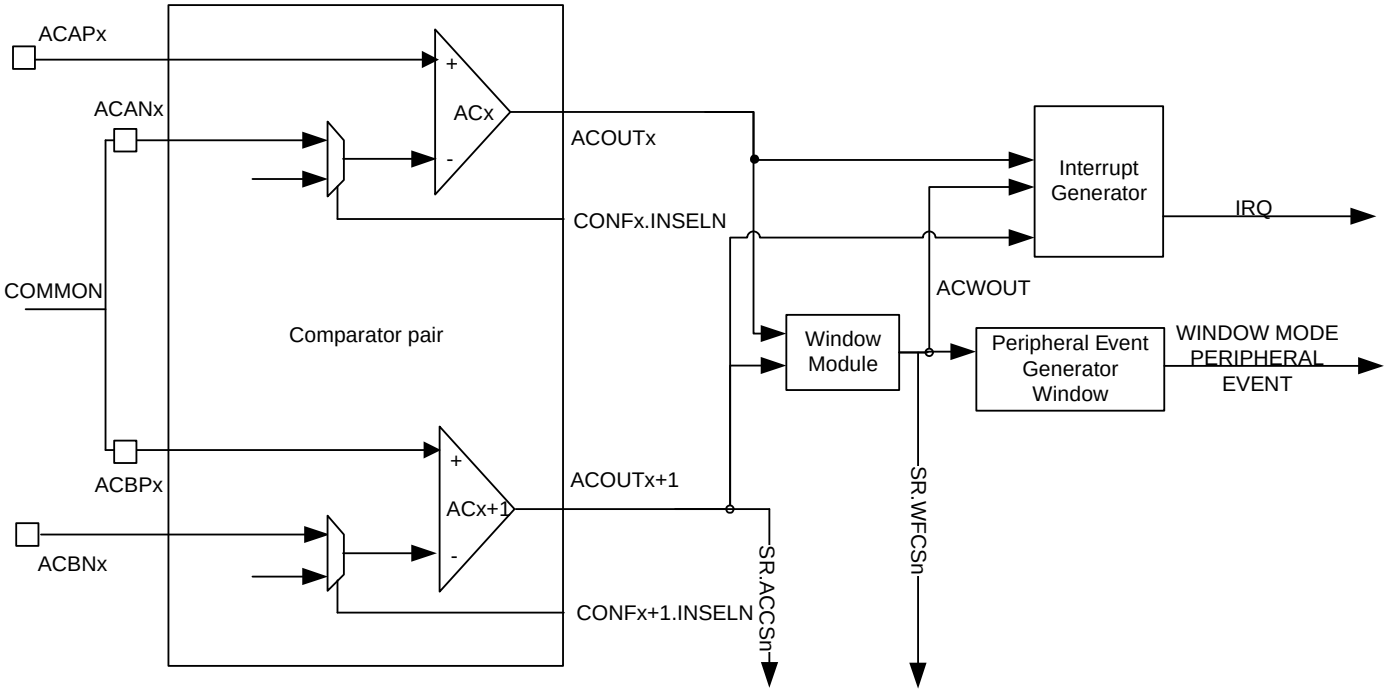
- When $V_{INP} > V_{INN}$ or
- When $V_{INP} < V_{INN}$

37.6.5 Window Mode

Window mode is enabled by writing a one to the Window Mode Enable bit in the Window Configuration Register (CONFWx.WFEN). Two ACs, ACx and ACx+1, (an even and the following odd build up a pair) are grouped if window mode is enabled by writing CONFWx.WFEN.

The sources of the negative input of ACx (even) and the positive input of ACx+1 (odd) must be connected together externally. The Negative Input Select field in the ACx Configuration Register (CONFx.INSELN) must select the corresponding external input pin (ACANx) as the source to the negative input of ACx. The negative input of ACx+1 can still be selected independently by CONFx+1.INSELN.

Figure 37-2. Analog Comparator Interface in Window mode



37.6.5.1 Window Mode Output

When operating in window mode, each AC generates the same ACOUT outputs as in normal mode, see Section 37.6.4.1.

Additionally, the ACIFC generates a window mode signal (ACWOUT) according to the common input voltage to be compared:

- ACWOUT = 1 if the common input voltage is inside the window, $V_{ACBNx+1} < V_{common} < V_{ACAPx}$
- ACWOUT = 0 if the common input voltage is outside the window, $V_{common} < V_{ACANx+1}$ or $V_{common} > V_{ACAPx}$
- ACWOUT = 0 if the window mode output is not available (SR.ACRDYx=0 or SR.ACRDYx+1=0)

The Windowx Mode Current Status bit in SR (SR.WFCSx) gives the comparison status of the window pair.

37.6.5.2 Window Mode Interrupts

When operating in window mode, each AC can generate the same interrupt as in normal mode, see Section 37.6.4.2.

Additionally, when ACs operate in window mode, a third type of interrupt based on ACWOUT can be generated (bit ISR.WFINTx).

Writing to Window Mode Interrupt Settings in Windowx Mode Configuration Register (CONFWx.WIS) selects an interrupt source:

- As soon as the common input voltage is inside the window.
- As soon as the common input voltage is outside the window.

- On toggle of the window compare output (ACWOUT).
- When the comparison in both ACs in the window mode is ready.
- When the common input voltage enters the window (i.e., rising-edge of ACWOUT).
- When the common input voltage leaves the window (i.e., falling-edge of ACWOUT).

37.6.5.3 Window Mode Peripheral Events

When operating in window mode, each AC can generate the same peripheral events as in normal mode, see Section 37.6.4.3.

Additionally, user enables generation of window peripheral events by writing a one to the Window Peripheral Event Enable bit (CONFWx.WFEN).

Writing to Window Mode Peripheral Event Selection Source (CONFWx.WEVSRC) selects the peripheral events source:

- As soon as the common input voltage is inside the window.
- As soon as the common input voltage is outside the window.
- On toggle of the window compare output (ACWOUT)
- Whenever a comparison is ready and the common input voltage is inside the window.
- Whenever a comparison is ready and the common input voltage is outside the window.
- When the comparison of both ACs in the window mode is ready.

37.6.6 Analog Hysteresis Control

The user can select the hysteresis voltage used by each AC to reduce noise in its output. As shown in Figure 37-3, the output of the AC switches from zero to one when the differential voltage V_D (= V_{INP} - V_{INN}) is greater than (hysteresis voltage / 2) and switches from one to zero when V_D is less than (hysteresis voltage / 2). The hysteresis voltage for each AC is selected by writing to the Hysteresis Voltage field in the CONFx register (CONFx.HYS), as shown in Table 37-3.

Figure 37-3. Hysteresis Voltage

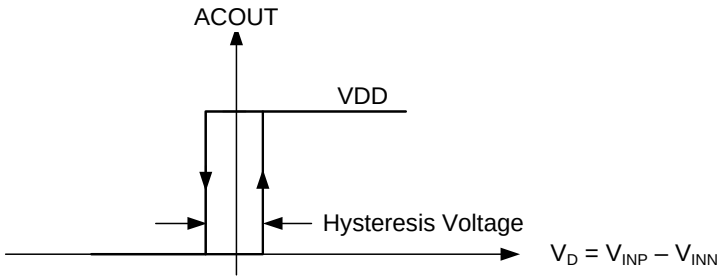


Table 37-3. Hysteresis Voltage Selection

CONFx.HYS	Hysteresis Voltage (mV)
0	0
1	25
2	50
3	75

37.6.7 Power Dissipation and Speed Trade-off

ACIFC is able to control whether an AC operates in fast or low-power mode, allowing a trade-off between its active power dissipation and speed. The desired mode is selected by writing to the FAST bit in the CONFx register (CONFx.FAST). When CONFx.FAST is zero (default), the corresponding AC operates in low-power mode. When CONFx.FAST is one, it operates in fast mode. In low-power mode, the AC dissipates significantly less active power than when it is in fast mode, but has a longer start-up time and propagation delay. Refer to Section 42. “Electrical Characteristics” on page 1121 for details.

37.6.8 Wake Up from Sleep Modes by Interrupt

An AC is able to wake up the system from a sleep mode by its interrupt, including sleep modes where CLK_ACIFC is stopped. For an AC to wake up the system, the user must follow this procedure before the system enters a sleep mode:

- Set AC in the Continuous Measurement mode (CONFx.MODE).
- Set the AC’s interrupt setting (CONFx.IS =2) so that it generates an interrupt on the toggling of the corresponding AC output.
- Wait for AC to be ready (SR.ACRDYx = 1).

Once ACIFC detects a transition on the relevant AC output, it sends a request to Power Manager for CLK_ACIFC to restart (the corresponding bit in the Power Manager Peripheral Power Control Register (PPCR) must be set) and sets the corresponding interrupt status bit in ISR after CLK_ACIFC has restarted. Note that in order for the interrupt from ACIFC to wake up the system, the ACIFC bit in the Power Manager Asynchronous Wake Up Enable Register (AWEN) must be set.

37.7 Peripheral Event Triggers

Peripheral events from other modules can trigger comparisons in the ACIFC. All ACs that are configured in peripheral event triggered single measurement mode will be started simultaneously when a peripheral event is received. ACs operating in continuous measurement mode or user triggered single measurement mode are unaffected by the received peripheral event. Software can still operate these ACs independently of ACs in peripheral event triggered single measurement mode.

A peripheral event will trigger one or more comparisons, in normal or window mode.

37.8 AC Test mode

By writing a one to the Analog Comparator Test Mode bit (CR.ACTEST), the outputs from the ACs are overridden by the value in the Test Register (TR), see Figure 37-1. This can be useful for software tests.

37.9 User Interface

Table 37-4. ACIFC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CTRL	Read/Write	0x00000000
0x04	Status Register	SR	Read-only	0x00000000
0x10	Interrupt Enable Register	IER	Write-only	0x00000000
0x14	Interrupt Disable Register	IDR	Write-only	0x00000000
0x18	Interrupt Mask Register	IMR	Read-only	0x00000000
0x1C	Interrupt Status Register	ISR	Read-only	0x00000000
0x20	Interrupt Status Clear Register	ICR	Write-only	0x00000000
0x24	Test Register	TR	Read/Write	0x00000000
0x30	Parameter Register	PARAMETER	Read-only	-(1)
0x34	Version Register	VERSION	Read-only	-(1)
0x80	Window0 Configuration Register(2)	CONFW0	Read/Write	0x00000000
0x84	Window1 Configuration Register(2)	CONFW1	Read/Write	0x00000000
0x88	Window2 Configuration Register(2)	CONFW2	Read/Write	0x00000000
0x8C	Window3 Configuration Register(2)	CONFW3	Read/Write	0x00000000
0xD0	AC0 Configuration Register(2)	CONF0	Read/Write	0x00000000
0xD4	AC1 Configuration Register(2)	CONF1	Read/Write	0x00000000
0xD8	AC2 Configuration Register(2)	CONF2	Read/Write	0x00000000
0xDC	AC3 Configuration Register(2)	CONF3	Read/Write	0x00000000
0xE0	AC4 Configuration Register(2)	CONF4	Read/Write	0x00000000
0xE4	AC5 Configuration Register(2)	CONF5	Read/Write	0x00000000
0xE8	AC6 Configuration Register(2)	CONF6	Read/Write	0x00000000
0xEC	AC7 Configuration Register(2)	CONF7	Read/Write	0x00000000

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.
2. Number of AC and AC pair is device specific therefore associated register may not exist.

37.9.1 Control Register

Name: CTRL

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
ACTEST	-	ESTART	USTART	-	-	EVENTEN	EN

- **ACTEST: Analog Comparator Test Mode**
  0: The Analog Comparator outputs are connected to the logic in ACIFC.
  1: The Analog Comparator outputs are bypassed with the AC Test Register.
- **ESTART: Peripheral Event Start Single Comparison**
  Writing a zero to this bit has no effect.
  Writing a one to this bit starts a comparison and can be used for test purposes.
  This bit is cleared when comparison is done.
  This bit is set when an enabled peripheral event is received.
- **USTART: User Start Single Comparison**
  Writing a zero to this bit has no effect.
  Writing a one to this bit starts a single comparison.
  This bit is cleared when comparison is done.
- **EVENTEN: Peripheral Event Trigger Enable**
  0: A peripheral event will not trigger a comparison.
  1: Enable comparison triggered by a peripheral event.
- **EN: ACIFC Enable**
  0: The ACIFC is disabled.
  1: The ACIFC is enabled.

37.9.2 Status Register

Name: SR

Access Type: Read-only

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	WFCS3	WFCS2	WFCS1	WFCS0
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
ACRDY7	ACCS7	ACRDY6	ACCS6	ACRDY5	ACCS5	ACRDY4	ACCS4
7	6	5	4	3	2	1	0
ACRDY3	ACCS3	ACRDY2	ACCS2	ACRDY1	ACCS1	ACRDY0	ACCS0

- **WFCSx: Windowx Mode Current Status**
  This bit is cleared when the common input voltage is outside the window.
  This bit is set when the common input voltage is inside the window.
- **ACRDYx: ACx Ready**
  This bit is cleared when the AC output (ACOUT) is not ready.
  This bit is set when the AC output (ACOUT) is ready, AC is enabled and its startup time is over.
- **ACCSx: ACx Current Comparison Status**
  This bit is cleared when V_{INP} is currently lower than V_{INN}
  This bit is set when V_{INP} is currently greater than V_{INN}.

37.9.3 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	WFINT3	WFINT2	WFINT1	WFINT0
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
SUTINT7	ACINT7	SUTINT6	ACINT6	SUTINT5	ACINT5	SUTINT4	ACINT4
7	6	5	4	3	2	1	0
SUTINT3	ACINT3	SUTINT2	ACINT2	SUTINT1	ACINT1	SUTINT0	ACINT0

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

37.9.4 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	WFINT3	WFINT2	WFINT1	WFINT0
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
SUTINT7	ACINT7	SUTINT6	ACINT6	SUTINT5	ACINT5	SUTINT4	ACINT4
7	6	5	4	3	2	1	0
SUTINT3	ACINT3	SUTINT2	ACINT2	SUTINT1	ACINT1	SUTINT0	ACINT0

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

37.9.5 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	WFINT3	WFINT2	WFINT1	WFINT0
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
SUTINT7	ACINT7	SUTINT6	ACINT6	SUTINT5	ACINT5	SUTINT4	ACINT4
7	6	5	4	3	2	1	0
SUTINT3	ACINT3	SUTINT2	ACINT2	SUTINT1	ACINT1	SUTINT0	ACINT0

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

37.9.6 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	WFINT3	WFINT2	WFINT1	WFINT0
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
SUTINT7	ACINT7	SUTINT6	ACINT6	SUTINT5	ACINT5	SUTINT4	ACINT4
7	6	5	4	3	2	1	0
SUTINT3	ACINT3	SUTINT2	ACINT2	SUTINT1	ACINT1	SUTINT0	ACINT0

- **WFINTx: Windowx Mode Interrupt Status**
  0: No Window mode interrupt is pending.
  1: Window mode interrupt is pending.
  This bit is cleared when the corresponding bit in ICR is written to one.
  This bit is set when the corresponding ACs pair operating in window mode generates an interrupt (see CONFWx to select the source).
- **SUTINTx: ACx Startup Time Interrupt Status**
  0: No Startup Time Interrupt is pending.
  1: Startup Time Interrupt is pending.
  This bit is cleared when the corresponding bit in ICR is written to one.
  This bit is set when the startup time of the corresponding AC is over.
- **ACINTx: ACx Interrupt Status**
  0: No normal mode interrupt is pending.
  1: Normal mode interrupt is pending.
  This bit is cleared when the corresponding bit in ICR is written to one.
  This bit is set when the corresponding AC generates an interrupt (see CONFx to select the source).

37.9.7 Interrupt Status Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	WFINT3	WFINT2	WFINT1	WFINT0
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
SUTINT7	ACINT7	SUTINT6	ACINT6	SUTINT5	ACINT5	SUTINT4	ACINT4
7	6	5	4	3	2	1	0
SUTINT3	ACINT3	SUTINT2	ACINT2	SUTINT1	ACINT1	SUTINT0	ACINT0

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in ISR and the corresponding interrupt request.

37.9.8 Test Register

Name: TR

Access Type: Read/Write

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
ACTEST7	ACTEST6	ACTEST5	ACTEST4	ACTEST3	ACTEST2	ACTEST1	ACTEST0

- **ACTESTx: ACx Output Override Value**
  0: Normal operating mode.
  1: If CTRL.ACTEST is one, ACx output is the bit value ACTESTx.

37.9.9 Parameter Register

Name: PARAMETER

Access Type: Read-only

Offset: 0x30

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	WIMPL3	WIMPL2	WIMPL1	WIMPL0
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
ACIMPL7	ACIMPL6	ACIMPL5	ACIMPL4	ACIMPL3	ACIMPL2	ACIMPL1	ACIMPL0

- **WIMPLx: Window x Mode Implemented**
  0: Window x mode is not implemented.
  1: Window x mode is implemented.
- **ACIMPLx: Analog Comparator x Implemented**
  0: Analog Comparator x is not implemented.
  1: Analog Comparator x is implemented.

37.9.10 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x34

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

37.9.11 Windowx Configuration Register

Name: CONFWx

Access Type: Read/Write

Offset: 0x80,0x84,0x88,0x8C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	WFEN
15	14	13	12	11	10	9	8
-	-	-	-	WEVEN	WEVSRC		
7	6	5	4	3	2	1	0
-	-	-	-	-	WIS		

- **WFEN: Window Mode Enable**
  0: Window mode is disabled.
  1: Window mode is enabled.
- **WEVEN: Window Peripheral Event Enable**
  0: Peripheral event from ACWOUT is disabled.
  1: Peripheral event from ACWOUT is enabled.

- **WEVSRC: Peripheral Event Source Selection for Window Mode**

WEVSRC			Peripheral Source Selection
0	0	0	ACWOUT rising edge
0	0	1	ACWOUT falling edge
0	1	0	ACWOUT rising or falling edge
0	1	1	Inside window
1	0	0	Outside window
1	0	1	Measure done
1	1	0	Reserved
1	1	1	Reserved

- **WIS: Window Mode Interrupt Settings**

WIS			Interrupt Settings
0	0	0	Window interrupt as soon as the common input voltage is inside the window
0	0	1	Window interrupt as soon as the common input voltage is outside the window
0	1	0	Window interrupt on toggle of ACWOUT
0	1	1	Window interrupt when evaluation of common input voltage is done
1	0	0	Window interrupt when the common input voltage enters the window (i.e., rising-edge of ACWOUT)
1	0	1	Window interrupt when the common input voltage leaves the window (i.e., falling-edge of ACWOUT)
1	1	0	Reserved
1	1	1	Reserved

37.9.12 ACx Configuration Register

Name: CONFx

Access Type: Read/Write

Offset: 0xD0,0xD4,0xD8,0xDC,0xE0,0xE4,0xE8,0xEC

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	ALWAYSON	FAST	HYS	

23	22	21	20	19	18	17	16
-	-	-	-	-	-	EVENP	EVENN

15	14	13	12	11	10	9	8
-	-	-	-	-	-	INSELN	

7	6	5	4	3	2	1	0
-	-	MODE		-	-	IS	

- **ALWAYSON: Always On**
  0: AC is disabled between measurements
  1: AC is always enabled
- **FAST: Fast Mode Enable**
  This bit controls whether the AC operates in fast or low-power mode, thus allowing a trade-off between its active power dissipation and speed (i.e., startup time and propagation delay):
  0: Low-power mode
  1: Fast mode
  Refer to Section 42. “Electrical Characteristics” on page 1121 for details on the current consumption, start-up time, and propagation delay of the AC when it is in either of these modes.
- **HYS: Hysteresis Voltage Value.**

HYS		Hysteresis value
0	0	Hysteresis voltage = 0 mV
0	1	Hysteresis voltage = 25 mV
1	0	Hysteresis voltage = 50 mV
1	1	Hysteresis voltage = 75 mV

- **EVENN: Peripheral Event Enable Negative**
  0: Do not output peripheral event when ACOUT is zero.
  1: Output peripheral event when ACOUT is zero.
- **EVENP: Peripheral Event Enable Positive**
  0: Do not output peripheral event when ACOUT is one.
  1: Output peripheral event when ACOUT is one.
- **INSELN: Negative Input Select**
  See *CONFx.INSELN: Negative Input Select* table in Module Configuration section.

- **MODE: Analog Comparator Mode**

MODE		Mode selected
0	0	Off
0	1	Continuous measurement mode
1	0	User triggered single measurement mode
1	1	Peripheral event triggered single measurement mode

- **IS: Interrupt Settings**

IS		Interrpt setting
0	0	When $V_{INP} > V_{INN}$
0	1	When $V_{INP} < V_{INN}$
1	0	On toggle of ACOUT
1	1	When comparison of V_{INP} and V_{INN} is done

37.10 Module configuration

The specific configuration for each ACIFC instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to the Power Manager chapter for details.

Table 37-5. ACIFC Configuration

Feature	ACIFC
AC_NO	8
ACW_NO	4

Table 37-6. ACIFC Clocks

Clock Name	Description
CLK_ACIFC	Clock for the ACIFC bus interface

Table 37-7. CONFx.INSELN: Negative Input Select

INSELN		Input selected
0	0	ACANx (ACBNx) pin selected
0	1	Reserved
1	0	Reserved
1	1	Reserved

Table 37-8. Register Reset Values

Register	Reset Value
VERSION	0x00000100
PARAMETER	0x0003000F

38. ADC Interface (ADCIFE)

Rev. 1.0.0.0

38.1 Features

- **Multi-channel Analog-to-Digital Converter with 12-bit resolution**
- **Selectable single ended or differential input voltage**
- **Programmable gain per conversion**
- **selectable ADC voltage input reference per conversion**
- **Numerous trigger sources**
  - **Software**
  - **Embedded 16-bit timer for periodic trigger**
  - **Continuous trigger**
  - **Peripheral Event trigger**
  - **External trigger, rising, falling or any-edge trigger**
- **Multiple sequencer modes:**
  - **Run multiple sequences using DMA transfer**
  - **Run a single conversion on a start-of-conversion**
- **Start of conversion or sequence without CPU intervention**
- **ADC Power Reduction Mode for low power ADC applications**
- **Window monitor, with selectable channel per conversion**
- **Programmable Sample & Hold time per conversion**

38.2 Overview

The ADC interface (ADCIFE) converts analog input voltages to digital values. The ADCIFE is based on a 12-bit Cyclic Pipeline topology to achieve sampling rates up to 300 KSPS (In differential mode without gain).

The ADC conversion results are reported in a common register for all channels. Conversions can be started for all enabled channels, either by a software trigger, by detection of a level change on the external trigger pin or event system, or by an integrated programmable 16-bit timer.

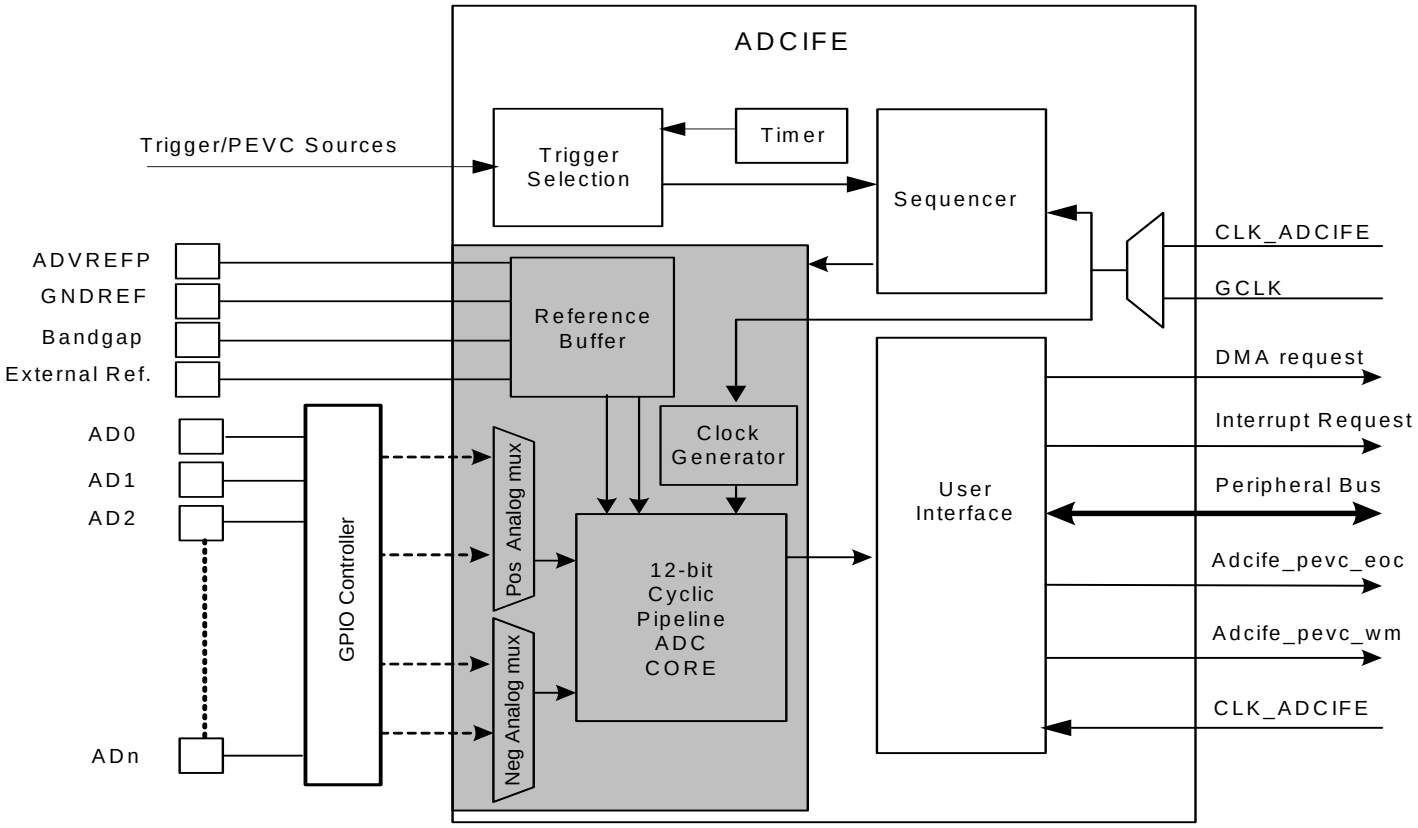
This ADC has selectable single-ended or fully differential inputs and takes benefits from a programmable gain from x0.5 to x64 included.

The conversions extend from -Vref to +Vref. 16 external channels on both positive and negative inputs can be managed.

The ADCIFE also integrates an Power Reduction mode, a Window Monitor mode, and connects with two Peripheral DMA Controller channels (One Rx for configuration and one Tx for transfering results). These features reduce both power consumption and processor intervention.

38.3 Block diagram

Figure 38-1. ADCIFE Block Diagram



38.4 I/O Lines Description

Table 38-1. I/O Lines decription table

Name	Description	Type
AD0-AD14	Analog input channels	Analog
External Ref.	2 External Voltage References	Analog
A33VDD	Analog power supply	Power
A33GND	Analog ground	Power
ADTRG	External trigger	Digital

38.5 Product dependencies

38.5.1 I/O Lines

The pins used for interfacing the ADCIFE may be multiplexed with I/O Controller lines. The programmer must first program the I/O controller to assign the desired ADCIFE pins to their

peripheral function. If I/O lines of the ADCIFE are not used by the application, they can be used for other purposes by the I/O controller.

38.5.2 Power Management

If the CPU enters a power reduction mode that disables clocks used by the ADCIFE, the ADCIFE will stop functioning and resume operation after the system wakes up from power reduction mode. Before entering a power reduction mode where the clock to the ADCIFE is stopped, make sure the Analog-to-Digital Converter cell is put in an inactive state.

If an event is sent to the ADCIFE and the clocks used by the ADCIFE are stopped, the ADCIFE clock will automatically be requested so that the conversion can be processed.

38.5.3 Clocks

The clock for the ADCIFE bus interface (CLK_ADCIFE) is generated by the Power Manager. This clock is disabled at reset, and can be enabled/disabled in the Power Manager. It is recommended to disable the ADCIFE before disabling the clock, to avoid freezing the ADCIFE in an undefined state. Additionally, the ADCIFE depends on a dedicated Generic Clock (GCLK). The GCLK can be set to a wide range of frequencies and clock sources, and must be enabled by the System Control Interface (SCIF) before the ADCIFE can be used.

This GCLK can also be muxed with the CLK_ADCIFE clock by writting the Clock Selection bit in the Configuration Register (CFG.CLKSEL). It is used during the sleep walking mode.

38.5.4 Interrupt Controller

The ADCIFE interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the ADCIFE requires the Interrupt Controller to be programmed first.

38.5.5 Event System

The event controller provides the ADCIFE a trigger source.

The ADCIFE also behaves as an event source:

- Event on an end of conversion (Figure 38-1 adcife_pevc_eoc signal)
- Event on a window monitor trigger (Figure 38-1 adcife_pevc_wm signal)

To operate correctly, the event source period should not exceed the ADC conversion time plus the startup time if needed.

Refer to the Peripheral Event System chapter for details.

38.6 Section 42. ”Electrical Characteristics” on page 1121Functional Description

38.6.1 Initializing the ADCIFE

To initialize the module the user first needs to configure the ADCIFE clocks (refer to Section 38.5.3). Then he needs to wait for the STARTUP time (Refer to Section 38.6.7) (If necessary). Then he can write a one to the Enable (EN) bit in the Control Register (CR). The user must check that ADCIFE has started correctly, firstly by checking that the Enable bit (EN) located in the Status Register (SR) is set. The Bandgap Buffer Request bit in the Control register (CR.BGREQEN) and the Reference Buffer Enable in the Control register (CR.REFBUFEN) must be set. If the reference buffer of the ADC cell is enabled and ready for use, SR.EN can tell if the ADCIFE is ready for operation since startup-time will be performed only when a sequencer trigger event occurs if necessary.

Note that all ADCIFE controls will be ignored until SR.EN goes to '1'.

Before the ADCIFE can be used, the Internal Reference Voltage signal must be selected by writing the CFG.REFSEL field and the I/O Controller must be configured correctly. Refer to I/O Controller section for details. The user must also configure the frequency range by writting the CFG.SPEED field.

Note that once configured, ADCIFE configuration registers should not be written during operation since they are permanently used by the ADCIFE. The user must ensure that ADCIFE is stopped during configuration unless he knows what he is doing.

38.6.2 Basic Operation

To convert analog values to digital values the user must first initialize the ADCIFE as described in Section 38.6.1. When the ADCIFE is initialized, the sequencer must be configured by writing into the Sequencer Configuration Register (SEQCFG) all the necessary bits or fields for the next conversion. This configuration can also be performed by Peripheral DMA Rx channel access.

Configuring channel N for a given conversion instructs the ADCIFE to convert the analog voltage applied to AD pin N. To start converting data the user can either manually start a conversion sequence by write a one to the sequencer trigger event (STRIG) bit in the Control Register (CR) or configure an automatic trigger to initiate the conversions. The automatic trigger can be configured to trig on many different conditions. Refer to Section 38.6.13 for details. The result of the conversions are stored in the Last Converted Value register (LCV) as they become available, overwriting the result from the previous conversion. To avoid data loss, the user must read the conversion results as they become available either by using an interrupt handler or by using a Peripheral DMA channel to copy the results to memory. Failing to do so will result in an Overrun Error condition, indicated by the LOVR bit in the Status Register (SR). To use an interrupt handler the user must enable the End Of Conversion (EOC) interrupt request by writing a one to the corresponding bit in the Interrupt Enable Register (IER). To clear the interrupt after the conversion result is read, the user must write a one to the corresponding bit in the Status Clear Register (SCR).

To use a Peripheral DMA Controller Tx channel, the user must configure the Peripheral DMA Controller appropriately. The DMA Controller will, when configured, automatically read converted data as they become available. There is no need to manually clear any bits in the Interrupt Status Register as this is performed by the hardware. If an Overrun Error condition happens during DMA operation, the LOVR bit in the SR will be set.

38.6.3 ADC Resolution

This ADC is a cyclic pipeline 12-bit or 8-bit resolution.

Resolution can be changed by writing the resolution field (RES) in the Sequencer Configuration Register (SEQCFG). By default, after a reset, the resolution is set to 12-bit.

38.6.4 Differential and Single-Ended Conversion Modes

The ADC has two conversion modes; differential and single-ended. If measuring signals where the positive input is always at a higher voltage than the negative input, the single-ended conversion mode should be used in order to have full 12-bit resolution in the conversion mode, which has only positive values. If however the positive input may go below the negative input creating some negative results, the differential mode should be used in order to get correct results. The configuration of the conversion mode is set in the field bipolar in SEQCFG (SEQCFG.BIPOLAR).

Note that the ADC works differentially in single-ended mode as well, as long as the positive input has a higher voltage than the negative.

38.6.5 ADC Clock Configuration

The ADCIFE generates an internal clock named CLK_ADC that is used by the Analog-to-Digital Converter cell to perform conversions. The CLK_ADC is selected by writing to the CLKSEL bit in the Configuration Register (CFG). The CLK_ADC frequency is the frequency of the clock selected by the CLKSEL bit divided by the prescaler field in the Configuration Register(CFG.PRESCAL).

The value of th frequency must be defined in order to provide an ADC clock frequency according to the maximum sampling rate parameter given in the Electrical Characteristics section. Failing to do so may result in incorrect Analog-to-Digital Converter operation.

The ADC cell converts an input voltage in 6 CLK_ADC periods and takes at least SHTIM+1 GCLK periods to sample for a 12-bit resolution.

Thus, the maximum achievable ADC sampling frequency is: $\frac{F(CLKADC)}{6}$

If a 8-bit result is generated, the maximum ADC sampling frequency is $\frac{F(CLKADC)}{4}$

38.6.6 Power Reduction Mode

The Power Reduction Mode maximizes power saving by automatically deactivating the Analog-to-Digital Converter cell when it is not being used for conversions. The Power Reduction Mode is enabled by writing a one to the Disable ADC(DIS) bit in the Control register (CR.DIS).

Before entering power reduction mode the user must make sure the ADCIFE is idle and that the Analog-to-Digital Converter cell is inactive. To make sure the ADCIFE is idle, write a zero to the Trigger Selection (TRGSEL) field in the Sequencer Configuration Register (SEQCFG) and wait for the sequencer busy (SBUSY) bit in the Status Register (SR) to be cleared. Note that by deactivating the Analog-to-Digital Converter cell, a startup time penalty as defined in the STARTUP field in the timing register (TIM) will apply on the next conversion.

The ADCIFE has the possibility to adjust the power consumption of the ADC cell according to the frequency range used. The SPEED field in the CFG register must be written to the right value.

38.6.7 Power-up and Startup Time

The Analog-to-Digital Converter cell has a startup time when the cell is activated for the first time. This startup time is at least 12 ADC_CLK. This timing should be manage by the user by setting the startup time field in the Timing Configuration register (TIM.STARTUP). The enable startup bit in the Timing Configuration register (TIM.ENSTUP) allows to enable or not the startup time.

For power-up and startup time values of the ADC cell, refer to the ADC cell chapter.

38.6.8 Operation Start/Stop

To reset ADCIFE to its initial state, user can enable the ADCIFE after it was previously disabled thanks to the Enable bit in the Control register (CR.EN) and the Disable bit in the Control register

(CR.DIS). Another way to reset ADCIFE is to write a one in the SWRST field of the Control Register (CR.SWRST). In both cases configuration registers won't be affected.

38.6.9 Analog Reference

Refer to Section 42. "Electrical Characteristics" on page 1121.

The ADC allows the possibility to select several voltage reference (Vref).

- An internal 1.0V voltage reference derived from the internal 1.1V
- 0.625*Vcc (to get 1.0V when Vcc=1.6V)
- Vcc/2
- 2 external reference inputs

38.6.10 GAIN

The ADC cell allows to affect different gain for each conversion by configuring appropriately the Gain Factor field GAIN in the SEQCFG register.

The programmable input amplification is from 1x to 64x. Moreover, this amplification can be divided by 2 by setting all bits in the field GAIN in the SEQCFG register to seven.

38.6.11 Conversion Results

If the Half Word Left Adjust (HWLA) bit in the SEQCFG register is set, then the result will be left adjusted on the 16 lower bits of the LCV register. Otherwise, results will be right-adjusted.

Positive and negative channels used in the last conversion are available both by reading the Sequencer Last Converted Value register (LCV).

38.6.12 Operating Modes Overview

Table 38-2. Operating modes description

Operating mode	Input range	Output code range	Conversion time	Output decimal code
Differential mode without gain	-Vref to + Vref	0 to 4095 (11 bits signed number)	6 clock_cycles	2047 + (Vin/Vref)*2047
Differential mode with gain=2n	-Vref/2^n to +Vref/2^n	0 to 4095 (11 bits signed number)	7 clock_cycles for n=1 (gain=2) 9 clock_cycles for n=6 (gain=64)	2047 + (2^n*Vin/Vref)*2047
Differential mode with division by 2	-2*Vref to +2*Vref	0 to 4095 (11 bits signed number)	7 clock_cycles	2047 + (Vin/(2*Vref))*2047
Differential mode with zoom and gain =2n (n>0)	-Vref/2n+Vshift, Vref/2n+Vshift Vshift=Vsup(1/4*zoomrange[1]+1/8*zoomrange[0])	0 to 4095 (11 bits signed number)	7 clock_cycles for n=1 (gain=2) 10 clock_cycles for n=6 (gain=64)	2047+(2^n*(Vin-Vshift)/Vref*2047

Operating mode	Input range	Output code range	Conversion time	Output decimal code
Unipolar mode without gain and without hysteresis	0 to Vref	0 to 4095 (12 bits unsigned number)	7 clock_cycles	4095*Vin/Vref
Unipolar mode without gain and with hysteresis (zoom-range[2]=1)	-0.05*vref to 0.95*vref	0 to 4095 (12 bits unsigned number)	7 clock_cycles	4095*(Vin+0.05*vref)/Vref
Unipolar mode without hysteresis and gain =2n	0 to Vref/2^n	0 to 4095 (12 bits unsigned number)	7 clock_cycles for n=1 (gain=2) 10 clock_cycles for n=6 (gain=64)	4095*(2^n*Vin)/Vref
Unipolar mode without hysteresis and with division by 2	0 to 2*Vref	0 to 4095 (12 bits unsigned number)	7 clock_cycles	4095*(Vin/2)/Vref)
Unipolar mode with hysteresis and gain =2n	(-0.05*Vref to 0.95*Vref) /2^n	0 to 4095 (12 bits unsigned number)	7 clock_cycles for n=1 (gain=2) 10 clock_cycles for n=6 (gain=64)	4095*(2^n*Vin+0.05*Vref)/Vref
Unipolar mode with hysteresis and with division by 2	-0.1*Vref to 1.9*Vref	0 to 4095 (12 bits unsigned number)	7 clock_cycles	4095*(0.5*Vin+0.05*Vref)/Vref

38.6.13 Sequencer Trigger Event (STRIG)

The sources must be configured through the TRGSEL field of the SEQCFG register (SEQCFG.TRGSEL). Selecting the event controller source allows any event controller source to generate a sequencer trigger event (STRIG). By configuring the continuous mode, STRIG will be generated continuously.

The ADC can serve a maximum of one STRIG every 6+1 CLK_ADC periods. Extra STRIG will be ignored. User will be informed thanks to the Sequencer Missed Trigger Event (SMTRG) field of the SR register (SR.SMTRG). If the STRIG frequency provided by the event controller exceeds the ADC capability, the event controller will generate an underrun status.

38.6.14 Internal Timer

The ADCIFE embeds an internal 16-bit timer used as a trigger source which can be configured by setting the ITMC field of the ITIMER register (ITIMER.ITMC).

*Internal Timer Trigger Period= (ITMC+1)*T(CLK_ADC)*

Once set as a STRIG source, the internal timer has to be started by writing a '1' in the TSTART bit of the CR register (CR.TSTART). It can be stopped in the same way by writing a '1' in the TSTOP bit of the CR register (CR.TSTOP). The current status of the internal timer can be read

from the Timer Busy field of the SR register (SR.TBUSY): 0 means stopped, 1 means running. In addition when the internal timer is running, if ITIMER.ITMC is written to change the internal timer timeout frequency, the internal counter is cleared to avoid rollover phenomena.

Note: It is possible to generate an internal timer event each GCLK period by writing 0 in ITIMER.ITMC and by selecting the internal timer as a STRIG source

38.6.15 Peripheral DMA Controller (PDCA) Capability

There are two PDCA channels. One Rx for transferring data result to memory, and one Tx for storing the updated configuration of the next conversion from memory.

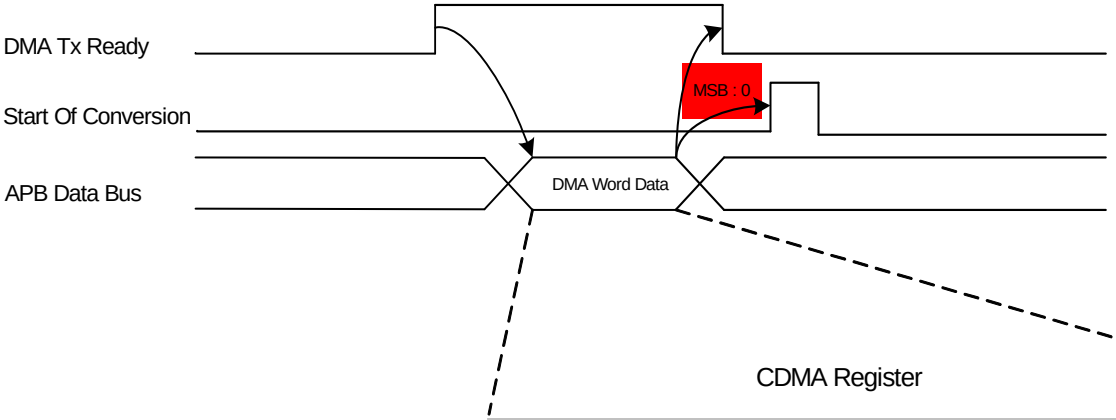
The LCV register contains the last converted value of the sequencer according to the conversion result format. The LCV register is updated each time the sequencer ends a conversion. If the last converted value has not been read, there's an overrun, the LOVR bit in the SR register indicates that at least one overrun error occurred. The LOVR bit of the SR register is cleared by writing a '1' in the LOVR field of the SCR register.

Tx transfer: If the configuration is performed by the PDCA, so the CDMA register contains configuration for the next conversion. If Window Mode is not used, only one word is usefull with the MSB bit to zero. If Window Mode is used, the first word has the MSB bit to one and the second word is dedicated for Window Mode with MSB bit set to zero.

Note: Rx PDCA transfers are 16 bits wide.

Note: In Tx, the first word received has always the same structure. The second word, if necessary, has always the same structure too.

Figure 38-2. One DMA Tx transfer: No Window Mode Configuration

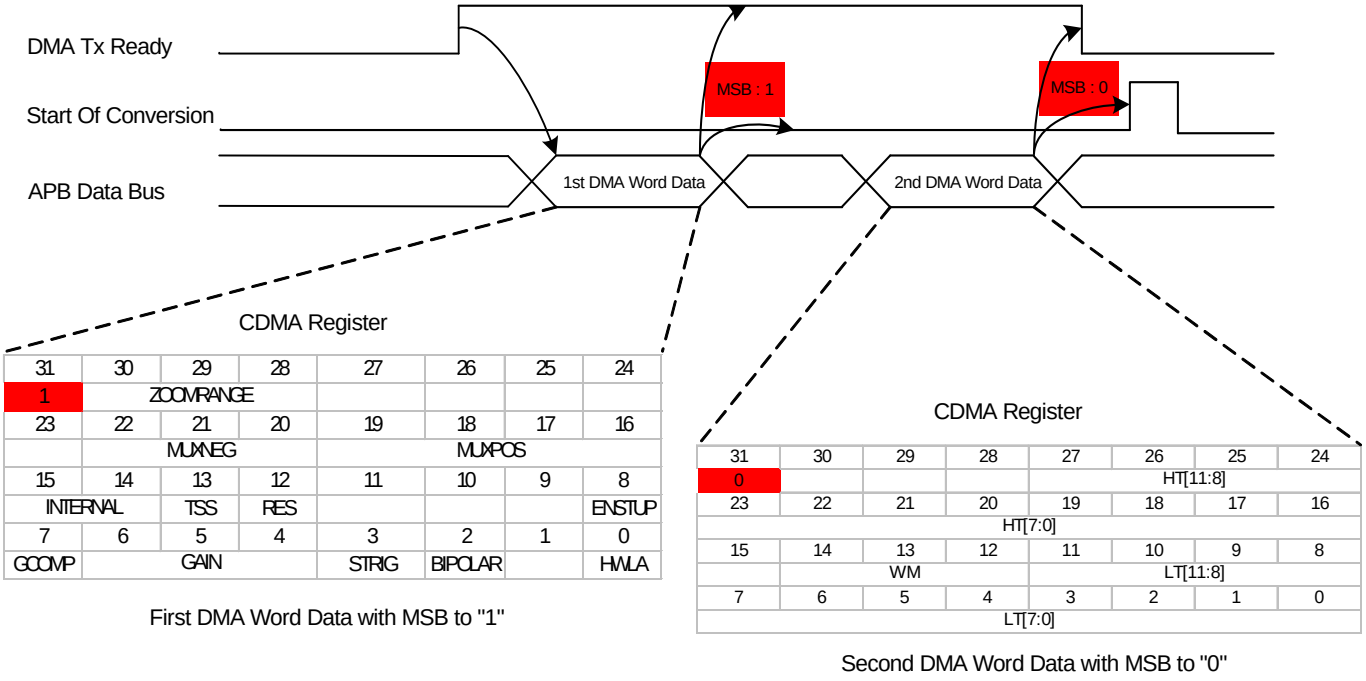


DMA Word Data with MSB to "0"

CDMA Register

31	30	29	28	27	26	25	24
0	ZOOMRANGE						
23	22	21	20	19	18	17	16
	MUXNEG			MUXPOS			
15	14	13	12	11	10	9	8
INTERNAL		TSS	RES				ENSTUP
7	6	5	4	3	2	1	0
GCOMP	GAIN			STRIG	BIPOLAR		HWLA

Figure 38-3. Two DMA Tx transfers: With Window Mode Configuration



38.6.16 Zoom Mode

The Zoom mode allows to choose a part of a range of the reference voltage (SEQCFG.ZOOMRANGE) and spread it from 0 to 4095 decimal. This particular mode of the ADC uses GAIN in which the input range is shifted with a programmable voltage.

The input voltage range is acording to a programmable voltage of Vdd and a range around this voltage (SEQCFG.GAIN).

38.6.17 Window Monitor

The window monitor monitors ADC results and make the ADCIFE behave as an analog comparator. Configuration is done by writing appropriately the Window Configuration Register (WCFG) and the Window Thresholds Register (WTH). When writing a one in the Monitor Filter Mode bit in the WCFG register (WCFG.MFM) , conversions are filtered using its index in the sequence. Otherwise, no filtering is applied, monitoring is performed on every conversion. Index is given by writing the field Source in the WCFG register (WCFG.SRC). Supported modes are selected by writing the Window Mode field in the WCFG register, refer to the Table 38-3 below.

Thresholds are given by writing the Low Threshold (LT) and High Threshold (HT) in WTH. Note that the significant WTH.LT and WT.HT bits are given by the precision selected in the SEQCFG.RES field. That means that if you are in 8-bit mode, only the 8 lower bits will be considered.

Table 38-3. Window Monitor Modes

WM field in WCFGy			Modes
0	0	0	No window mode (default)
0	0	1	Mode 1: active when result > LT
0	1	0	Mode 2: active when result < HT
0	1	1	Mode 3: active when LT < result < HT
1	0	0	Mode 4: active when (!(LT < result < HT))
1	0	1	reserved
1	1	0	reserved
1	1	1	reserved

Note: Comparisons are performed regardless with the SEQCFG.HWLA setting (half word left adjust).

38.6.18 Interrupts

Interrupt requests are enabled by writing a one to the corresponding bit in the Interrupt Enable Register (IER) and disabled by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). Enabled interrupts can be read from the Interrupt Mask Register (IMR). Active interrupt requests, but potentially masked, are visible in the Status Register (SR). To clear an active interrupt request, write a one to the corresponding bit in the Clear Register (CR).

The Status Register (SR) fields in common with IER/IDR/IMR show the status since the last write to the Interrupt Clear Register. Other SR fields show the status at the time being read.

Table 38-4. ADCIFE Interrupt Group

Line	Line Description	Related Status
0	Sequencer	Sequencer end of conversion (SEOC) Sequencer (last converted value) overrun (LOVR) Sequencer missed trigger event (SMTRG)
	Timing	Timer time-out
	Window	Window monitor

38.6.19 Conversion Performances

For performance and electrical characteristics of the ADCIFE, refer to Section 42. ”Electrical Characteristics” on page 1121.

38.7 User Interface

Table 38-5. ADCIFE Register Memory Map

Offset	Register	Register Name	Access	Reset
0x0000	Control Register	CR	Write-Only	0x00000000
0x0004	Configuration Register	CFG	Read/Write	0x00000000
0x0008	Status Register	SR	Read-Only	0x00000000
0x000C	Status Clear Register	SCR	Write-Only	0x00000000
0x0014	Sequencer Configuration Register	SEQCFG	Read/Write	0x00000000
0x0018	Configuration Direct Memory Access Register	CDMA	Write-Only	0x00000000
0x001C	Timing Configuration Register	TIM	Read/Write	0x00000000
0x0020	Internal Timer Register	ITIMER	Read/Write	0x00000000
0x0024	Window Monitor Configuration Register	WCFG	Read/Write	0x00000000
0x0028	Window Monitor Threshold Configuration Register	WTH	Read/Write	0x00000000
0x002C	Sequencer Last Converted Value Register	LCV	Read-Only	0x00000000
0x0030	Interrupt Enable Register	IER	Write-Only	0x00000000
0x0034	Interrupt Disable Register	IDR	Write-Only	0x00000000
0x0038	Interrupt Mask Register	IMR	Read-Only	0x00000000
0x003C	Calibration Register	CALIB	Read/Write	0x00000000
0x0040	Version Register	VERSION	Read-Only	-(1)
0x0044	Parameter Register	PARAMETER	Read-Only	-(1)

Note: 1. The reset value for this register is device specific. Refer to the Module Configuration section at the end of this chapter.

38.7.1 Control Register

Name: CR

Access Type: Write-Only

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	BGREQDIS	BGREQEN	DIS	EN

7	6	5	4	3	2	1	0
-	-	REFBUFDIS	REFBUFEN	STRIG	TSTART	TSTOP	SWRST

Writing a zero to any of those bits in this register has no effect.

- **BGREQDIS:Bandgap buffer request disable**
  Writing a one to this bit disables the bandgap buffer request
  Reading this bit always returns 0
- **BGREQEN:Bandgap buffer request enable**
  Writing a one to this bit enables the bandgap buffer request
  Reading this bit always returns 0
- **DIS:ADCIFE disable**
  Writing a one to this bit disables the ADCIFE
  Reading this bit always returns 0

Note: Changes do not apply immediately, ADCIFE status can be checked by reading the EN field of the SR register

- **EN:ADCIFE enable**
  Writing a one to this bit enables the ADCIFE
  Reading this bit always returns 0

Note: Changes do not apply immediately, ADCIFE status can be checked by reading the EN field of the SR register

- **REFBUFDIS: Reference buffer disable**
  Writing a one to this bit disables the Reference Buffer
  Reading this bit always returns 0
- **REFBUFEN: Reference buffer enable**
  Writing a one to this bit enables the Reference Buffer
  Reading this bit always returns 0
- **STRIG:Sequencer trigger**
  Writing a one to this bit generates a sequencer trigger event
  Reading this bit always returns 0

- **TSTART:Internal timer start bit**

  Writing a one to this bit starts the internal timer

  Reading this bit always returns 0

Note: The internal timer status can be read in the RUNT field of the SR register

- **TSTOP:Internal timer stop bit**

  Writing a one to this bit stops the internal timer

  Reading this bit always returns 0

Note: The internal timer status can be read in the RUNT field of the SR register

- **SWRST: Software reset**

  Writing a zero to this bit has no effect.

  Writing a one to this bit resets the ADCIFE, simulating a hardware reset. Using that control ensures that ADCIFE internal features will return to their initial states. Configuration registers won't be affected.

38.7.2 Configuration Register

Name: CFG

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	PRESCAL		

7	6	5	4	3	2	1	0
-	CLKSEL	SPEED		REFSEL			-

- **PRESCAL: Prescaler Rate Slection**

PRESCAL	Group Configuration	System clock division factor
000	DIV4	4
001	DIV8	8
010	DIV16	16
011	DIV32	32
100	DIV64	64
101	DIV128	128
110	DIV256	256
111	DIV512	512

- **CLKSEL: Clock Selection for sequencer/ADC cell**
  1: The APB clock is used
  0: The Generic clock is used

- **SPEED: ADC current reduction**

SPEED	Max speed
00	300 ksps
01	225 ksps
10	150 ksps
11	75 ksps

- **REFSEL: ADC Reference selection**

REFSEL	Description
000	Internal 1.0V (10/11*bandgap)
001	0.625*VCC
010	External reference 1
011	External reference 2 (DAC_VOUT)
1XX	VCC/2

38.7.3 Status Register

Name: SR

Access Type: Read-Only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	BGREQ	-	REFBUF	CBUSY	SBUSY	TBUSY	EN

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	TTO	-	SMTRG	WM	LOVR	SEOC

- **BGREQ: Bandgap buffer request Status**
  1: Bandgap buffer request is enabled
  0: Bandgap buffer request is disabled
- **BGREQREFBUF: Reference Buffer Status**
  1: Reference Buffer is enabled
  0: Reference Buffer is disabled
- **CBUSY: Conversion busy**
  1: ADCIFE is converting
  0: ADCIFE is not converting
- **SBUSY: Sequencer busy**
  1: ADCIFE sequencer is running
  0: ADCIFE sequencer is ready
- **TBUSY: Timer busy**
  1: ADCIFE internal timer is running
  0: ADCIFE internal timer is stopped
- **EN: Enable Status**
  1: ADCIFE is ready for operation
  0: ADCIFE is not ready
- **TTO: Timer time-out**
  This bit is set when the internal timer times out
  This bit is cleared when the corresponding bit in SCR is written to one

- **SMTRG:Sequencer missed trigger event**
  This bit is set when a sequencer trigger event is missed
  This bit is cleared when the corresponding bit in SCR is written to one
- **WM:Window monitor**
  This bit is set when the watched result value goes to the defined window
  This bit is cleared when the corresponding bit in SCR is written to one
- **LOVR:Sequencer last converted value overrun**
  This bit is set when an overrun error occurs on the LCV register
  This bit is cleared when the corresponding bit in SCR is written to one
- **SEOC:Sequencer end of conversion**
  This bit is set when an end of conversion occurs
  This bit is cleared when the corresponding bit in SCR is written to one

38.7.4 Status Clear Register

Name: SCR

Access Type: Write-Only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	TTO	-	SMTRG	WM	LOVR	SEOC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit clears the corresponding SR bit

38.7.5 Sequencer Configuration Register

Name: SEQCFG

Access Type: Read/Write

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	ZOOMRANGE			-	-	-	-

23	22	21	20	19	18	17	16
-	MUXNEG			MUXPOS			

15	14	13	12	11	10	9	8
INTERNAL		-	RES	-	TRGSEL		

7	6	5	4	3	2	1	0
GCOMP	GAIN			-	BIPOLAR	-	HWLA

- **ZOOMRANGE: Zoom shift/unipolar reference source selection**

ZOOMRANGE	Mode	Comment
000	Select Vref for shift cycle	All modes except zoom and unipolar with hysteresis
001	Select Vdd/4 for shift cycle	Zoom mode
010	Select Vdd/2 for shift cycle	Zoom mode
011	Select 3*Vdd/4 for shift cycle	Zoom mode
1XX	Select 0.9*Vref for shift cycle	Unipolar mode with hysteresis

- **MUXNEG: MUX selection on Negative ADC input channel**

BIPOLAR	INTERNAL	MUXNEG	Negative input to ADC
0	0X	XXX	primary_anahot_XXX tied to ground
0	1X	001 or 111	Pad Ground
1	0X	000	primary_anahot_0
1	0X	001	primary_anahot_1
1	0X	010	primary_anahot_2
1	0X	011	primary_anahot_3
1	0X	100	primary_anahot_4
1	0X	101	primary_anahot_5
1	0X	110	primary_anahot_6
1	0X	111	primary_anahot_7
1	1X	000	Vsingle = 0.9*Vref
1	1X	001	Pad Ground
1	1X	010	Vcalib3 = Vref/10
1	1X	011	Reference Ground
1	1v	100	Not used
1	1X	101	Vcalib3 = Vref/10
1	1X	110	Not used
1	1X	111	Pad Ground

- **MUXPOS: MUX selection on Positive ADC input channel**

BIPOLAR	INTERNAL	MUXPOS	Positive input to ADC
X	X0	0000	AD0
X	X0	0001	AD1
X	X0	0010	AD2
X	X0	0011	AD3

BIPOLAR	INTERNAL	MUXPOS	Positive input to ADC
X	X0	0100	AD4
X	X0	0101	AD5
X	X0	0110	AD6
X	X0	0111	AD7
X	X0	1000	AD8
X	X0	1001	AD9
X	X0	1010	AD10
X	X0	1011	AD11
X	X0	1100	AD12
X	X0	1101	AD13
X	X0	1110	AD14
X	X0	1111	Bandgap
X	X1	X000	Not used
X	X1	X001	Bandgap
X	X1	X010	Scaled Vcc, Vcc/10
X	X1	X011	DAC internal
X	X1	X100	Not used
X	X1	X101	Not used
X	X1	X110	Vsingle = 0.9*Vref
X	X1	X111	Reference Ground

- **INTERNAL: Internal Voltage Sources Selection**

INTERNAL	Positive Internal Selection	Negative Internal Selection
00	Enables the primary/secondary voltage sources	Enables the primary voltage sources
01	Enables the internal voltage sources	Enables the primary voltage sources
10	Enables the primary/secondary voltage sources	Enables the internal voltage sources
11	Enables the internal voltage sources	Enables the internal voltage sources

- **RES: Resolution**

RES	Resolution
0	12-bits
1	8-bits

- **TRGSEL: Trigger selection**

TRGSEL	Trigger
000	Software
001	internal ADC timer
010	internal trigger source (refer to module configuration section)
011	Continuous mode
100	External trigger pin rising edge
101	External trigger pin falling edge
110	External trigger pin both edges
111	Reserved

- **GCOMP: Gain Compensation**

  1: Enables the ADC gain error reduction

  0: Disables the ADC gain error reduction

- **GAIN: Gain factor**

GAIN	Gain factor
000	1 x
001	2 x
010	4 x
011	8 x
100	16 x
101	32 x
110	64 x
111	0,5 x

- **BIPOLAR: Bipolar Mode**
  1: Enables the differential mode
  0: Enables the single-ended mode
- **HWLA: Half Word Left Adjust**
  1: Enables the HWLA mode
  0: Disables the HWLA mode

38.7.6 Configuration Direct Memory Access

Name: CDMA

Access Type: Write-Only

Offset: 0x18

Reset Value: 0x00000000

First DMA Word

31	30	29	28	27	26	25	24
DW	ZOOMRANGE			-	-	-	-
23	22	21	20	19	18	17	16
-	MUXNEG			MUXPOS			
15	14	13	12	11	10	9	8
INTERNAL		TSS	RES	-	-	-	ENSTUP
7	6	5	4	3	2	1	0
GCOMP	GAIN			STRIG	BIPOLAR	-	HWLA

- **DW: Double Word transmitting**
  Number of DMA transfer remainning to end the configuration of the next conversion
- **TSS: Internal timer start or stop bit**
  Writing a one to this bit starts the internal timer
  Writing a zero to this bit stops the internal timer

Second DMA Word (Optional Window Mode)

31	30	29	28	27	26	25	24
DW	-	-	-	HT[11:8]			
23	22	21	20	19	18	17	16
HT[7:0]							
15	14	13	12	11	10	9	8
-	WM			LT[11:8]			
7	6	5	4	3	2	1	0
LT[7:0]							

- **DW: Double Word transmitting**
  This bit must be set to zero (configuration completed)

This register is used for DMA transfers to the ADCIFE module. The first word transmitted is the general configuration. If the MSB bit is set to one, a second word will be transfered to complete the configuration so that use the Window Mode. If the MSB bit is set to zero, the configuration is completed.

The second word is only used when the window mode is needed. In this case, its MSB bit is always set to zero (configuration completed).

38.7.7 Timing Configuration Register

Name: TIM

Access Type: Read/Write

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	ENSTUP

7	6	5	4	3	2	1	0
-	-	-	STARTUP				

- **ENSTUP: Enable Startup**
  1: Enables the Startup time
  0: Disables the Startup time
- **STARTUP: Startup time**
  Number of CLK_ADC clock cycles to wait for starting conversion: (STARTUP+1)

38.7.8 Internal Timer Register

Name: ITIMER

Access Type: Read/Write

Offset: 0x20

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
ITMC[15:8]							

7	6	5	4	3	2	1	0
ITMC[7:0]							

- **ITMC:Internal Timer Max Counter**
  f(itimer_timeout)=f(GCLK)/(ITMC+1)

38.7.9 Window Monitor Configuration

Name: WCFG

Access Type: Read/Write

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	WM			-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

- **WM: Window Monitor Mode**

WM	Window monitor mode
000	OFF
001	Mode 1: RES(SRC) > LT
010	Mode 2: RES(SRC) < HT
011	Mode 3: LT<RES(SRC)<HT
100	Mode 4: (LT>=RES(SRC)) || (RES(SRC)>=HT)
101	Reserved
110	Reserved
111	Reserved

38.7.10 Window Monitor Threshold Configuration

Name: WTH

Access Type: Read/Write

Offset: 0x28

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	HT[11:8]			

23	22	21	20	19	18	17	16
HT[7:0]							

15	14	13	12	11	10	9	8
-	-	-	-	LT[11:8]			

7	6	5	4	3	2	1	0
LT[7:0]							

- **HT:High Threshold**
  HighThreshold value
- **LT:Low Threshold**
  Low Threshold value

38.7.11 Sequencer Last Converted Value

Name: LCV

Access Type: Read-Only

Offset: 0x2C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	LCNC			LCPC			

15	14	13	12	11	10	9	8
LCV[15:0]							

7	6	5	4	3	2	1	0
LCV[7:0]							

- **LCNC: Last converted negative channel**
  This field is set by hardware to the last negative channel converted, i.e. what negative channel the LCV represents.
- **LCPC: Last converted positive channel**
  This field is set by hardware to the last positive channel converted, i.e. what positive channel the LCV represents.
- **LCV: Last converted value**
  This field is set by hardware to the last sequencer converted value depending on precision and on the choosen left adjustment mode (SEQCFG.HWLA).

38.7.12 Interrupt Enable Register

Name: IER

Access Type: Write-Only

Offset: 0x30

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	TTO	-	SMTRG	WM	LOVR	SEOC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

38.7.13 Interrupt Disable Register

Name: IDR

Access Type: Write-Only

Offset: 0x34

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	TTO	-	SMTRG	WM	LOVR	SEOC

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

38.7.14 Interrupt Mask Register

Name: IMR

Access Type: Read-Only

Offset: 0x38

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
	-	-	-		-	-	-

23	22	21	20	19	18	17	16
-	-	-	-		-	-	-

15	14	13	12	11	10	9	8
	-	-	-		-	-	-

7	6	5	4	3	2	1	0
-	-	TTO	-	SMTRG	WM	LOVR	SEOC

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

38.7.15 Calibration Register

Name: CALIB

Access Type: Read/Write

Offset: 0x3C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	FCD

15	14	13	12	11	10	9	8
BIASCAL				-	-	-	BIASSEL

7	6	5	4	3	2	1	0
CALIB							

- **FCD: Flash Calibration Done**

  Set to one when CALIB and BIASCAL have been updated by the flash fuses after a reset.

  1: The flash calibration will be redone after any reset

  0: The flash calibration will only be redone after a power-on reset

- **BIASCAL: Bias calibration**

BIASCAL		Ptat current	Bgap current	Total current
0000		3.0	1.1	4.1
0001		3.0+0.75	1.1	4.85
0010	Default setting	3.0+1.5	1.1	5.6
0011		3.0+1.5+0.75	1.1	6.35
0100		3.0	1.1+1.1	5.2
0101		3.0+0.75	1.1+1.1	5.95
0110		3.0+1.5	1.1+1.1	6.7
0111		3.0+1.5+0.7	1.1+1.1	7.45
10XX	Pure bandgap bias	0	5.5	5.5
11XX		0	5.5+1.1	6.6

- **BIASSEL: Select bias mode**

  1: Select bandgap bias

  0: Select mixed bias

- **CALIB: Calibration value**

CALIB Value	Description
7	S1 MSB
4	S1 LSB
3	S2 MSB
0	S2 LSB

38.7.16 Module Version

Name: VERSION

Access Type: Read-Only

Offset: 0x40

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

38.7.17 Parameter Register

Name: PARAMETER

Access Type: Read-Only

Offset: 0x44

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
N							

- **N: Number of channels**

38.8 Module Configuration

The specific configuration for each ADCIFE instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 38-6. ADCIFE Configuration

Feature	ADCIFE
External Ref 2 connection	DACC OUT output

Table 38-7. ADCIFE Clock Name

Clock Name	Description
CLK_ADCIFE	Clock for the ADCIFE bus interface
GCLK_ADCIFE	The generic clock used for the ADCIFE is GCLK10

Register	Reset Value
VERSION	0x00000100

39. LCD Controller (LCDCA)

Rev: 1.0.0.0

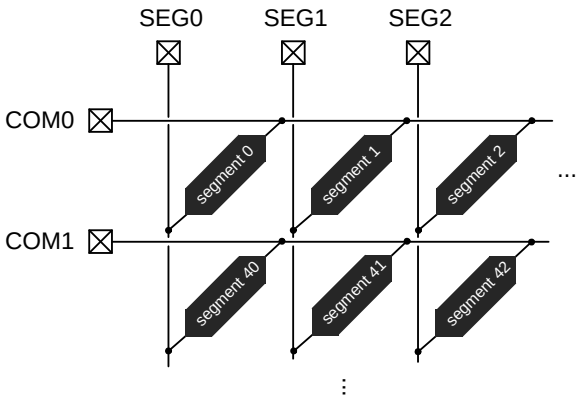
39.1 Features

- **Display capacity up to 40 segment and up to 4 common terminals**
- **Supports from static up to 1/4 duty**
- **Supports static and 1/3 bias**
- **Shadow display memory gives full freedom in segment update**
- **ASCII character mapping**
- **Automated characters string scrolling**
- **Automated characters string display**
- **Automated segments display**
- **Autonomous animation up to 8 segments**
- **LCD driver active in power save mode for low power operation**
- **Low power waveform**
- **Flexible selection of frame frequency**
- **Configurable blink mode and frequency**
- **Uses only 32 kHz clock**
- **On-chip LCD power supply**
- **Software contrast adjustment control**
- **Equal source and sink capability to increase LCD life time**
- **Interrupt mode for display update or wake-up from sleep mode**

39.2 Overview

A LCD display is made of several segments (or complete symbols) which can be visible or invisible. A segment has two electrodes with liquid crystal between them. These electrodes are the common terminal (COM) and the segment terminal (SEG). When a voltage above a threshold voltage is applied across the liquid crystal, the segment becomes visible. The voltage must alternate to avoid an electrophoresis effect in the liquid crystal, this effect degrades the display. Hence the voltage waveform across a segment must not have a DC-component.

Figure 39-1. LCD Segment/Common Terminals



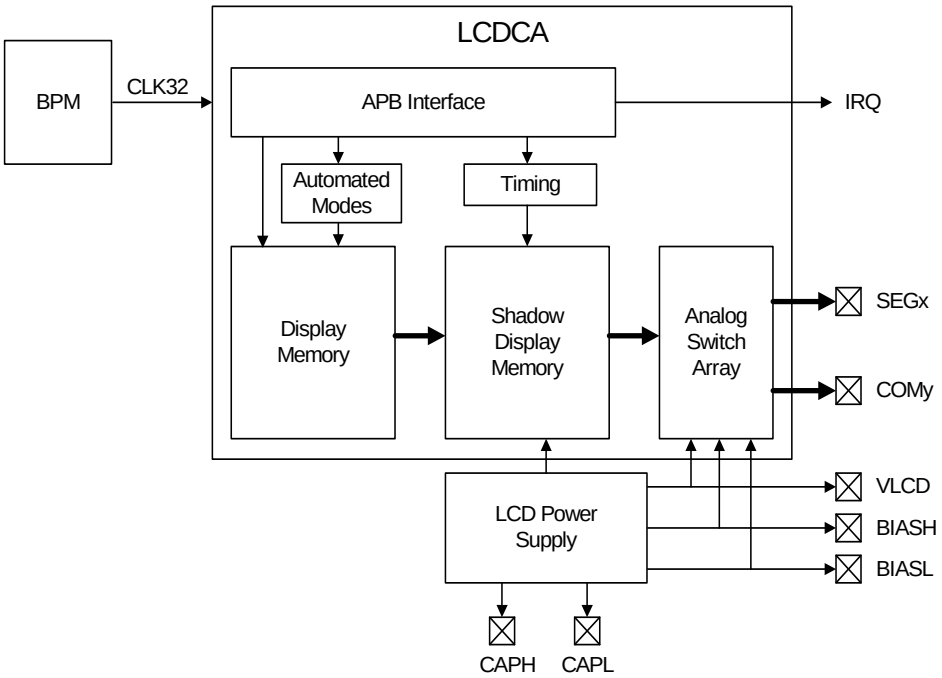
The LCD controller (LCDCA) is intended for monochrome passive liquid crystal display (LCD) with up to 4 common terminals and up to 40 segment terminals. Unused segment/common terminals (pins) are configured as general purpose I/O pins.The LCD controller uses a 32kHz clock (CLK_LCD) and can therefore be running in deep sleep modes.

Dedicated Low Power Waveform, Contrast Control, Extended Interrupt Mode, ASCII Character Mapping, automated modes... are defined to offload the CPU, reduce interrupts and reduce power consumption.

To reduce hardware design complexity, the module includes integrated LCD buffers, an integrated power supply voltage.

39.3 Block Diagram

Figure 39-2. LCDCA Block Diagram



39.4 I/O Lines Description

Table 39-1. I/O Lines Description

Pin Name	Pin Description	Type
SEGx	Segment terminal x	Analog
COMy	Common terminal y	Analog
VLCD	Bias voltage	Analog
BIAS1	Bias voltage (= 1/3 V_{LCD})	Analog
BIAS2	Bias voltage (= 2/3 V_{LCD})	Analog
CAPL	High voltage end of flying capacitor	Analog
CAPH	Low voltage end of flying capacitor	Analog

39.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

39.5.1 I/O Lines

The LCDCA pins (SEGx and COMy) are multiplexed with other peripherals. The user must first configure the I/O Controller to give control of the pins to the LCDCA.

VLCD, BIAS1, BIAS2, CAPL, CAPH are not multiplexed.

39.5.2 Power Management

This module can control the LCD display while CLK_LCDCA is disabled but stops functioning when CLK_LCD (32KHz) is disabled.

The power consumption of LCDCA itself can be minimized by:

- using the lowest acceptable frame rate (refer to the LCD glass technical characteristics),
- using the low power waveform (default mode),
- using automated modes,
- configuring the lowest possible contrast value.

39.5.3 Clocks

The clock for this module (CLK_LCDCA) is generated by the Power Manager. It can be enabled or disabled either manually through the user interface of the Power Manager or automatically when the system enters a sleep mode that disables the clocks to the peripheral bus modules.

The 32KHz clock (CLK_LCD) must be enabled before use. When system enters a sleep mode, 32KHz clock can be disabled, see Power Manager chapter for details.

39.5.4 Interrupts

The LCDCA interrupt request line is connected to the interrupt controller. Using the interrupt requires the interrupt controller to be configured first.

39.5.5 Wake Up

Wake up signal is connected to Power Manager (PM). Using wake up mechanism requires the PM to enable the corresponding asynchronous wake up source first. Also LCDCA interrupt must be enabled first.

39.5.6 Debug Operation

When an external debugger forces the CPU into debug mode, the LCDCA continues normal operation.

39.6 Functional Description

39.6.1 LCD Display

The display memory stores the values of all segments to display. Accessible through APB, it should be filled before next frame starts.

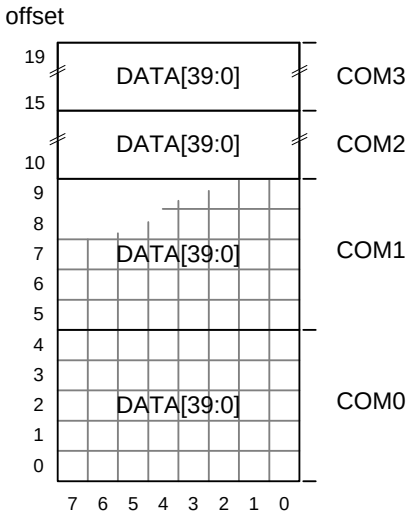
A start of a new frame triggers the update of the shadow display memory. The content of display memory is copied into the shadow display memory. A display memory refresh is possible without affecting data that is sent to the panel. Note that display memory is not initialized at power-up.

When a bit in the display memory is written to one, the corresponding segment will be energized (ON / opaque), and de-energized (OFF / transparent) when this bit is written to zero.

Addressing COM0 starts a frame by driving an opposite phase with large amplitude on COM0 as against non addressed COM terminals. Non-energized segments are in phase with the addressed COM0, and energized segments have opposite phase and large amplitude (refer to ”Operating modes” on page 1036).

Shadow display memory bits are multiplexed into the decoder. The decoder is configured by the LCD timing and controls the analog switches to produce an output waveform onto each selected COM terminals (duty value).

Figure 39-3. Display Memory Mapping



39.6.2 Operating modes

To energize a segment, an absolute voltage above the LCD threshold must be applied. This is done by setting the SEG terminal to opposite phase when the corresponding COM terminal is active. For a display with more than one common terminal, two additional voltage levels (1/3 bias) must be applied. Otherwise, non-energized segments on COM0 would be energized for all non-selected common lines.

Duty bits (DUTY) in the Configuration register (CFG) defines the duty cycle. Unused common terminals are driven to ground.

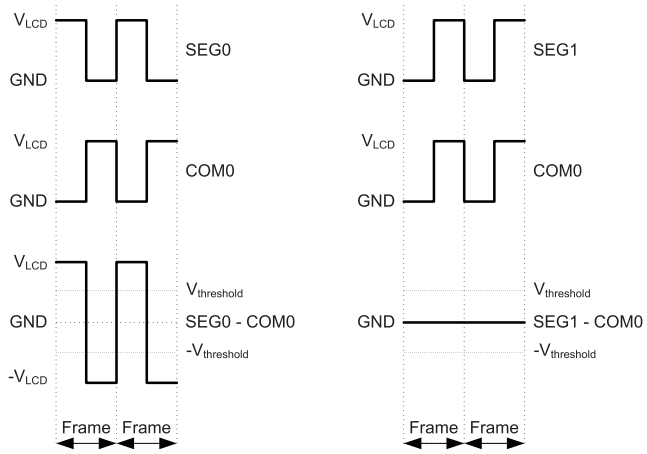
Table 39-2. Duty Selection

DUTY[1:0]	Duty	Bias	COM pins Used
0 0	1/4	1/3	COM[0:3]
0 1	Static	Static	COM0
1 0	1/2	1/3	COM[0:1]
1 1	1/3	1/3	COM[0:2]

39.6.2.1 Static Duty and Static Bias

If all segments on a LCD have one common electrode then each segment have a unique segment terminal. SEG0-COM0 is the voltage across a segment that is ON and SEG1-COM0 is the voltage across a segment that is OFF.

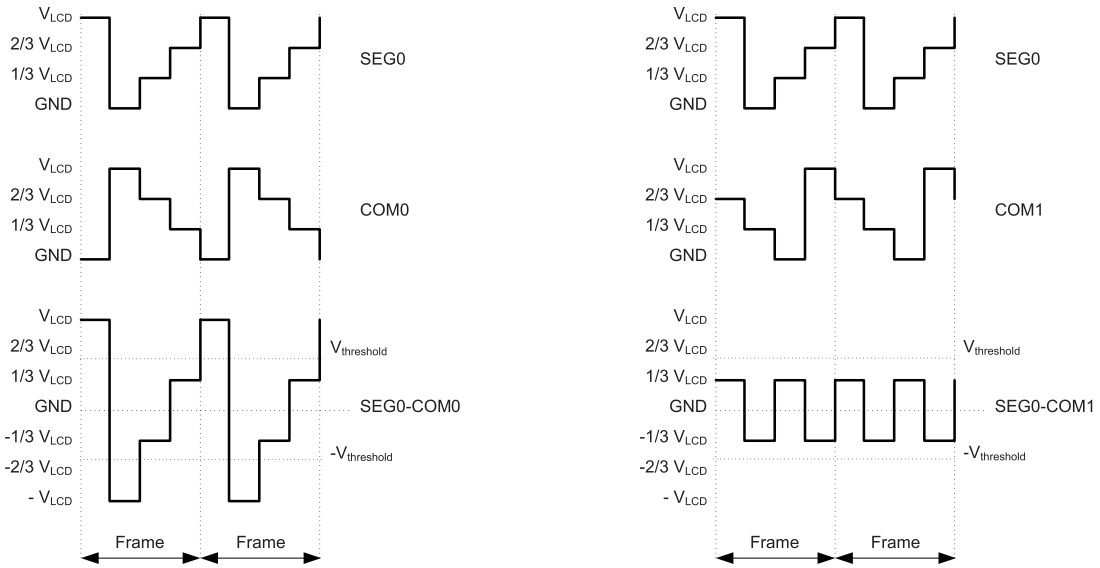
Figure 39-4. Driving a LCD With One Common Line



39.6.2.2 1/2 Duty and 1/3 Bias

For a LCD with two common terminals (1/2 duty) a more complex waveform must be used to individually control segments. SEG0-COM0 is the voltage across a segment that is ON and SEG0-COM1 is the voltage across a segment that is OFF.

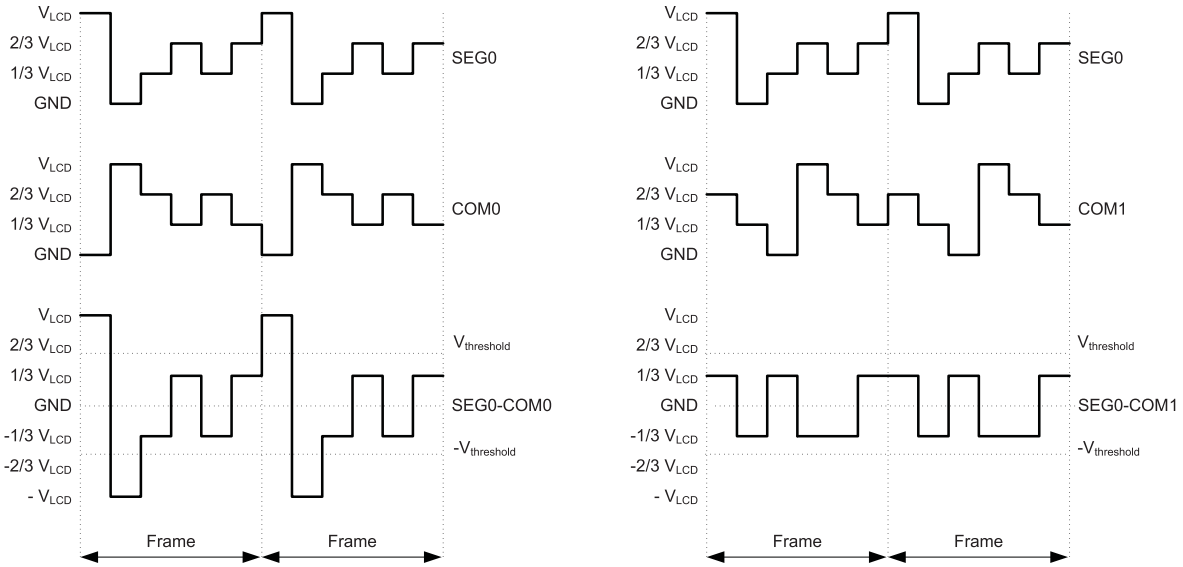
Figure 39-5. Driving an LCD With Two Common Lines



39.6.2.3 1/3 Duty and 1/3 Bias

1/3 bias is usually recommended for LCD with three common terminals (1/3 duty). SEG0-COM0 is the voltage across a segment that is ON and SEG0-COM1 is the voltage across a segment that is OFF.

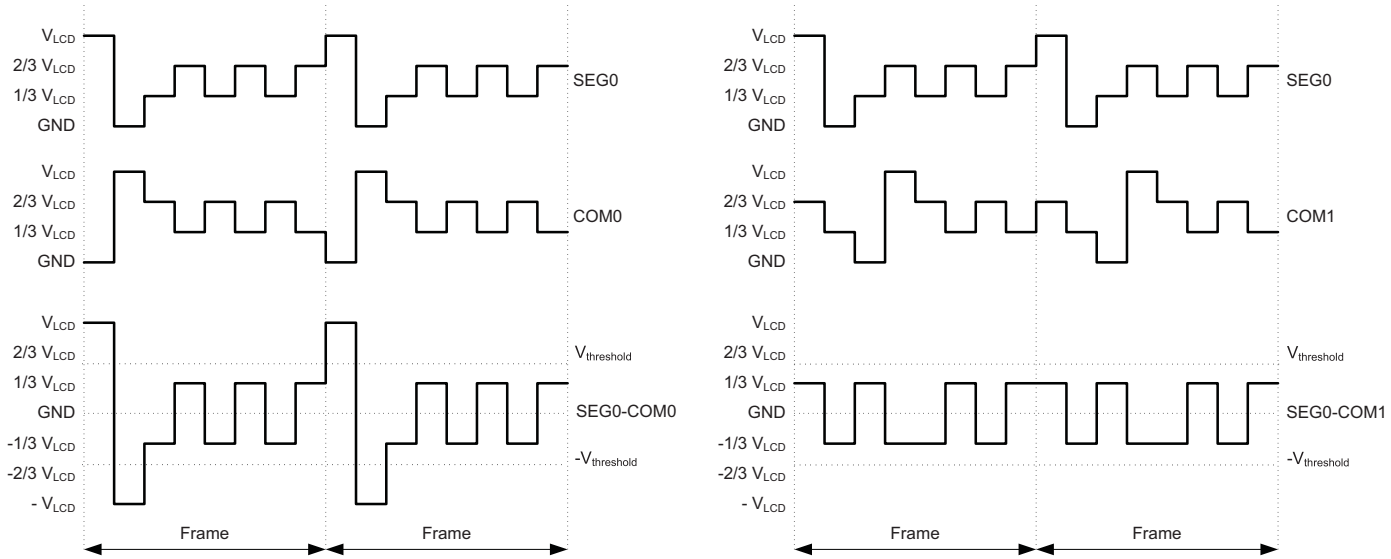
Figure 39-6. Driving a LCD With Three Common Lines



39.6.2.4 1/4 Duty and 1/3 Bias

1/3 bias is optimal for LCD with four common terminals (1/4 duty). SEG0-COM0 is the voltage across a segment that is ON and SEG0-COM1 is the voltage across a segment that is OFF.

Figure 39-7. Driving a LCD With Four Common Lines



39.6.3 Enabling/Disabling LCDCA

Write bit Enable (EN) to one in Control Register (CR) to enable the module. Write bit Disable (DIS) to one to disable it. LCD controller will be disabled after the completion of current frame. Bit Enable in Status Register (SR.EN) is set to one when LCD controller is ready to operate and set to zero when LCD power supply is off.

When LCD controller is disabled all segment and common terminals are driven to GND, discharging the LCD in order to avoid DC voltage across the segments and a slowly fading image. Data in the display memory is preserved.

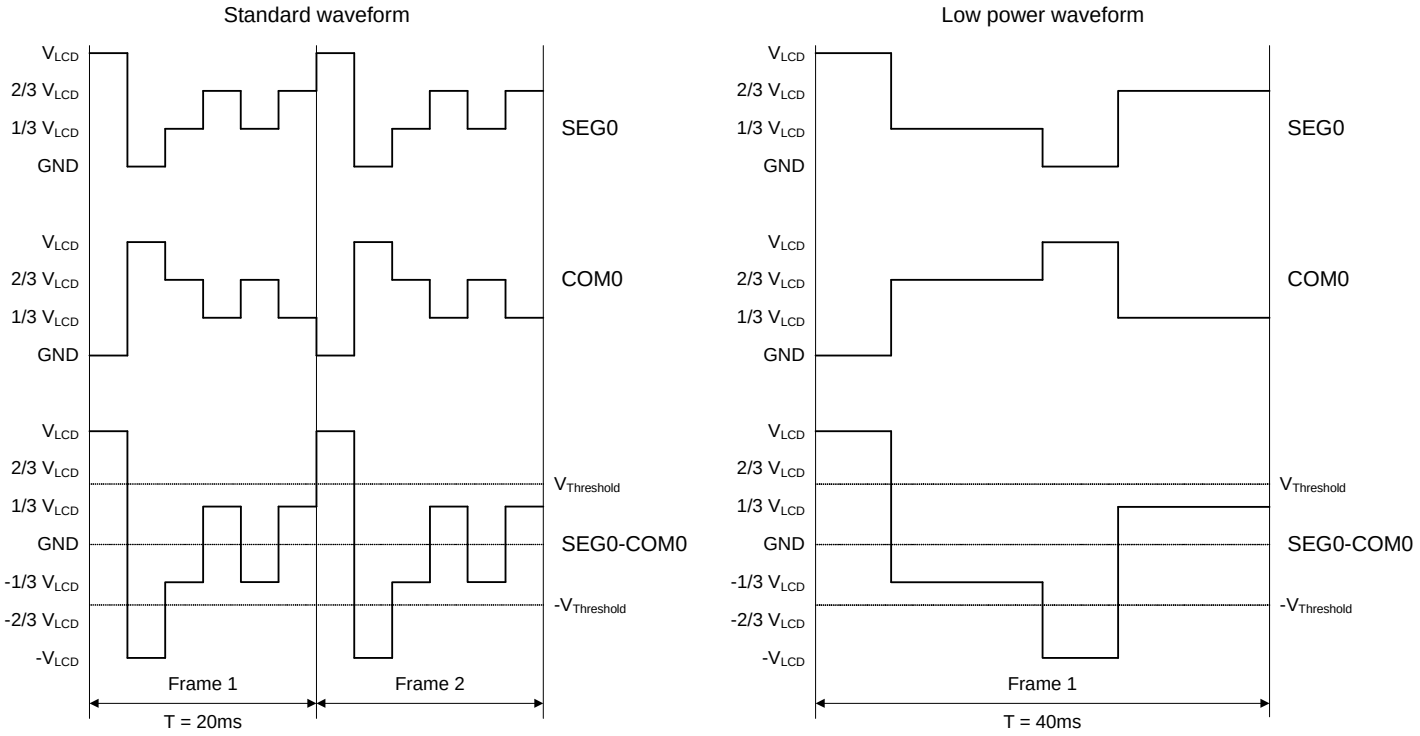
In order to restart correctly, it is preferrable to disable all functions (blinking, FCx,...) before disabling LCD controller.

39.6.4 Waveform Modes

To reduce toggle activity and hence power consumption, write a zero to the Waveform Mode (CFG.WMOD) to enable the low power waveform mode (the LCD glass must support this mode). The low power waveform period is then twice the standard waveform period, in both modes DC voltage is null.

To select the standard waveform, write a one to the CFG.WMOD bit. This bit shall not be modified when LCDCA is enabled.

Figure 39-8. Waveform Modes (Three Common Lines)



39.6.5 Timing Generation

39.6.5.1 Frame Rate

The Prescaler field (PRESC) in Timing register (TIM) selects a tap point from a ripple counter. The ripple counter output can be further divided by setting the Clock Divider (CLKDIV[2:0]).

Table 39-3. LCD Prescaler Selection

PRESC	Output From Prescaler
0	CLK_LCD / 8
1	CLK_LCD / 16

The Clock Division field (CLKDIV) in TIM register defines the division ratio in the clock divider. This gives extra flexibility in frame rate setting.

$$FrameRate = \frac{F(\text{CLK\_LCD})}{(K \times N \times (1 + CLKDIV) \times 2^{(1 - WMOD)})}$$

Where:

N = prescaler divider (8 or 16).
K = 8 for 1/4, 1/2 and static duty.
K = 6 for 1/3 duty.
WMOD = 0 in low power waveform mode, = 1 in standard waveform mode.

Table 39-4. LCD Clock Divider (1/4 Duty, WMOD=1)

CLKDIV[2:0]	Divided by	Frame rate (1/4 Duty)			
		F(CLK_LCD) = 32 kHz		F(CLK_LCD) = 32768 Hz	
		N=8	N=16	N=8	N=16
0 0 0	1	500 Hz	250 Hz	512 Hz	256 Hz
0 0 1	2	250 Hz	125 Hz	256 Hz	128 Hz
0 1 0	3	166.667 Hz	83.333 Hz	170.667 Hz	85.333 Hz
0 1 1	4	125 Hz	62.5 Hz	128 Hz	64 Hz
1 0 0	5	100 Hz	50 Hz	102.4 Hz	51.2 Hz
1 0 1	6	83.333 Hz	41.667 Hz	85.333 Hz	42.667 Hz
1 1 0	7	71.429 Hz	35.714 Hz	73.143 Hz	36.671 Hz
1 1 1	8	62.5 Hz	31.25 Hz	64 Hz	32 Hz

Note that when using 1/3 duty, the frame rate is increased by 33% compared to the values listed above and in low power waveform mode, the frame rate is divided by two.

Table 39-5. Frame Rate Examples (WMOD=1)

CLK_LCD	Duty	K	PRESC	N	CLKDIV[2:0]	Frame rate
32.768 kHz	Static	8	1	16	4	32768 / (8*16*(1+4)) = 51.2 Hz
32.768 kHz	1/2	8	1	16	4	32768 / (8*16*(1+4)) = 51.2 Hz
32.768 kHz	1/3	6	1	16	4	32768 / (6*16*(1+4)) = 68.267 Hz
32.768 kHz	1/4	8	1	16	4	32768 / (8*16*(1+4)) = 51.2 Hz

39.6.5.2 Frame Counters

For several functions (blinking, automated modes,...) a frame counter is used to create a time base. There are three independent frame counters (FC0, FC1 and FC2) which can be associated to any function (refer to corresponding section).

$$f_{FCx} = \frac{\text{FrameRate}}{(\text{TIM.FCx} \times 8) + 1}$$

For FC0 only, the prescaler of 8 can be bypassed by writting a one to TIM.FC0PB.

Note that frame counter frequency depends on frame duration therefore of waveform mode.

Each frame counter is enabled by writing a one to Frame Counter x Enable (CR.FCxEN) and disabled by writing a one to Frame Counter x Disable (CR.FCxDIS).Frame counter must be disabled (SR.FCxS=0) before the update of its associated TIM.FCx value.

39.6.6 CPU Display Memory Access

39.6.6.1 Direct Access

CPU can access display memory in direct access by writing to Data Register Low x (DRLx) and Data Register High x (DRHx). Read-modify-write operation is then required to update few bits.

To modify a segment defined by SEGx / COMy, select register R and bit B:

$$R = (y << 6 + x) >> 5$$

$$B = x \& 0x1F$$

Where R is the register index in the list {DRL0, DRH0, DRL1, DRH1, DRL2, DRH2, DRL3, DRH3} and B is the bit position in this register.

39.6.6.2 Indirect Access

CPU can also update up to 8 bits in display memory in indirect access by writing to Indirect Access Data Register (IADR). It allows to modify 1 up to 8 bits in a single operation without modifying masked bits in display memory (no read-modify-write operation). This register requires:

- DATA[7:0], each bit represents the state of a segment,
- DMASK[7:0], each bit is a mask for DATA field. When DMASK[x]=1, DATA[x] is not written to display memory,
- OFF[4:0], byte offset in display memory (see Figure 39-3 on page 1036).

To modify a segment defined by SEGx / COMy, write byte at offset OFF, bit B:

$$OFF = (5(y << 3) + x) >> 3$$

$$B = x \& 0x7$$

39.6.7 Locking Shadow Display Memory

Writing a one to LOCK bit, in Configuration register (CFG), freezes the shadow display memory update. Then if the display memory is modified, the display remains unchanged. When this bit is cleared, the shadow display memory is updated when a new frame starts.

39.6.8 Blinking Modes

39.6.8.1 Software Blinking

Writing bit BLANK in CFG register to one turns OFF all LCD segments at the next frame. If BLANK=0 the content of the display memory is output on the LCD. The blink frequency is then software dependant. To avoid unexpected intermediate display, blank command should be written after the end of the frame (SR.FC0R=1).

LCD controller must be running (frames are generated) to blink segments.

39.6.8.2 Hardware Blinking

To blink all segments on LCD panel, write a zero to Blink Mode (MODE) in Blink Configuration Register (BCFG). Write a one to MODE to blink selected segments.

Up to eight segments can be selected individually to blink. Each bit in Blink Segment Selection x field (BCFG.BSSx) selects a segment for blinking. If BSS0[y]=1, segment connected to SEG0/COMy is selected to blink. If BSS1[y]=1, segment connected to SEG1/COMy is selected to blink.

A segment will blink if its corresponding bit is one in the display memory, otherwise it remains OFF.

The blink frequency is defined by the number of frames (FCx in TIM register) between each state ON/OFF. So after FCx+1 frames, the segment will change state. Note that blinking frequency is also related to waveform mode, see "Waveform Modes" on page 1039.

The frame counter is selected by writing its number in BCFG.FCS field. BCFG.BSSx and BCFG.MODE should be modified after a frame has ended (SR.FC0R=1), BCFG.FCS shall not be modified when blinking is running.

The blinking is started by writing a one to CR.BSTART and stopped by writing a one to CR.BSTOP. The status of blinking can be read in SR.BLKS.

Table 39-6. Blinking Modes

BLANK	EN	BSS1[3:0] BSS0[3:0]	Comment
1	x	xxxx xxxx	All segments are OFF
0	0	xxxx xxxx	All segments are driven by the display memory
0	1	0000 0000	All segments are blinking at the blink frequency
		Not equal to zero	Selected segment(s) are blinking at the blink frequency

39.6.9 Autonomous Segment Animation

Up to eight pixels can be animated by an internal circular shift register. Up to eight states are then defined to make a running wheel for example. The number of segments used for animation must be written in SIZE field in Circular Shift Register Configuration (CSRCFG). If SIZE=N, circular shift register uses bits 0 to N-1.

The circular shift register bits are mapped on SEG2 and SEG3 of all common terminals.

Table 39-7. Circular Shift Register Bit Mapping

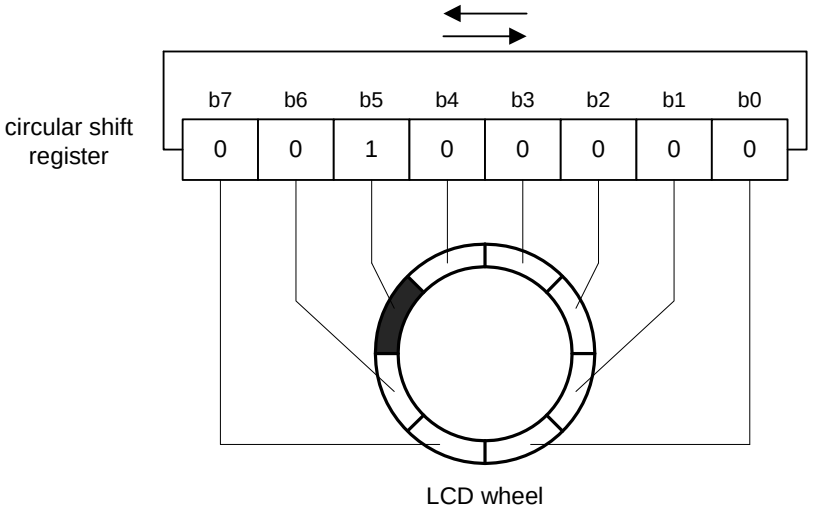
Shift Register Bit	COM	SEG
0	0	2
1	0	3
2	1	2
3	1	3
4	2	2
5	2	3
6	3	2
7	3	3

The shifting period is defined by the number of frame in TIM.FCx. The frame counter is selected by writing its number in CSRCFG.FCS field. If frame rate is 50Hz (20ms) in standard waveform mode, shifting period is 160ms up to 5.1s.

Initial value of circular shift register must be written in CSRCFG.DATA, the direction is defined by CSRCFG.DIR bit (0 for left, 1 for right) and circular shift register operation is started by writing a one to CR.CSTART.

Once enabled, data is shifted every TIM.FCx+1 frames, circular shift register is autonomous and system can enter any sleep mode (only 32KHz must be running).

Figure 39-9. Wheel Animation Example (8 Segments)



39.6.10 ASCII Character Mapping

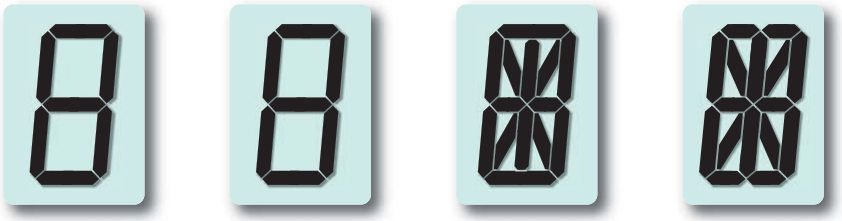
LCDCA handles up to four ASCII characters tables, configured in Character Mapping Configuration register (CMCFG). Instead of handling each segments in display memory for a selected digit, user writes ASCII code in Character Mapping Control Register (CMCR) to display the corresponding character.

User can then drive several digits with few operations:

1. select the Type of Digit (CMCFG.TDG), see Figure 39-10 on page 1044,
2. write the Start Segment value (CMCFG.STSEG) of the first digit,
3. select Digit Reverse Mode (CMCFG.DREV) if required. If DREV is one, segment index is decremented,
4. then write ASCII code in CMCR register.

If digit uses contiguous segments, repeat step 4 to update remaining digits. Segment index is updated automatically according to the number of segment used in the digit and DREV value.

Figure 39-10. Type of Digit Supported



7-Segment	7-Segment	14-Segment	16-Segment
- 3 COM term. - 3 SEG term.	- 4 COM term. - 2 SEG term.	- 4 COM term. - 4 SEG term.	- 3 COM term. - 6 SEG term.

Character mapping saves CPU execution time and allows a fast return to sleep mode after display update.

Table 39-8. 7-segments character table

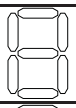
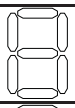
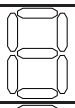
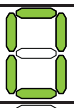
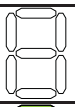
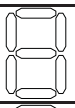
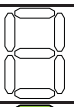
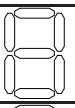
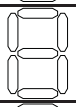
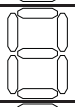
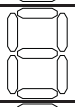
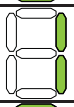
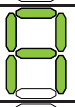
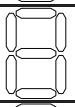
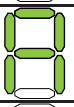
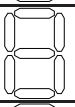
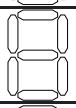
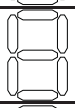
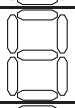
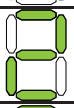
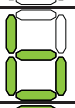
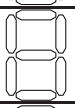
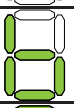
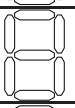
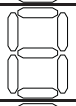
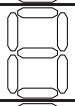
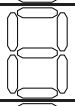
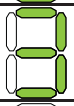
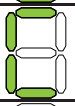
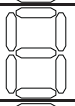
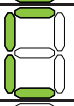
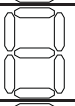
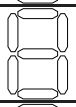
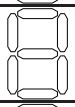
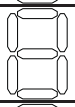
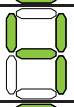
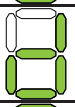
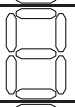
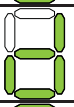
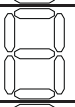
	x000	x001	x010	x011	x100	x101	x110	x111
0000								
0001								
0010								
0011								
0100								
0101								
0110								
0111								
1000								
1001								
1010								
1011								
1100								
1101								
1110								
1111								

Table 39-9. 14-segments character table

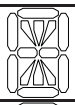
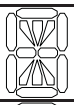
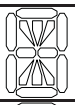
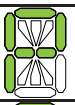
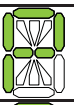
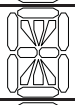
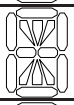
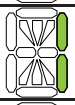
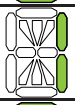
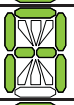
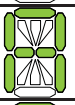
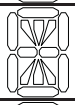
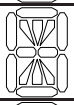
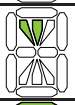
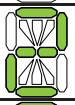
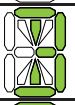
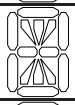
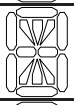
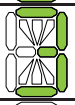
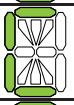
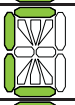
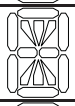
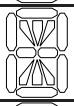
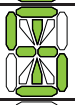
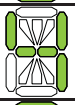
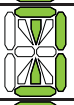
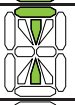
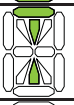
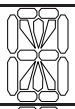
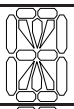
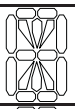
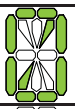
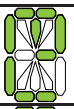
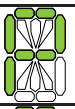
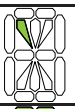
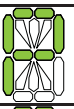
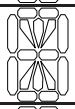
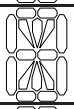
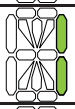
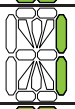
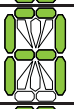
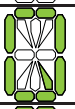
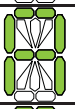
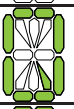
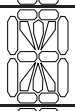
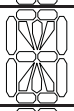
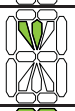
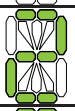
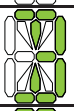
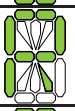
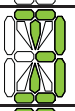
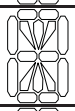
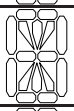
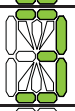
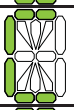
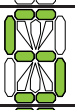
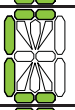
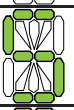
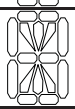
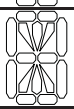
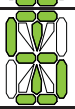
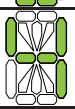
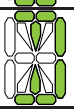
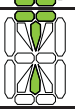
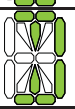
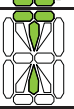
	x000	x001	x010	x011	x100	x101	x110	x111
0000								
0001								
0010								
0011								
0100								
0101								
0110								
0111								
1000								
1001								
1010								
1011								
1100								
1101								
1110								
1111								

Table 39-10. 16-segments character table

	x000	x001	x010	x011	x100	x101	x110	x111
0000								
0001								
0010								
0011								
0100								
0101								
0110								
0111								
1000								
1001								
1010								
1011								
1100								
1101								
1110								
1111								

39.6.11 Automated Character Mapping

Displaying predefined character strings can be automated using the Peripheral DMA Controller (PDCA). Two modes are available, defined by MODE bit in Automated Character Mapping Configuration register (ACMCFG):

- MODE=0, the sequential character string display mode is selected
- MODE=1, the scrolling of character string display mode is selected

39.6.11.1 Sequential Characters String Display

This mode displays characters of a string periodically (frame basis).
The configuration is:

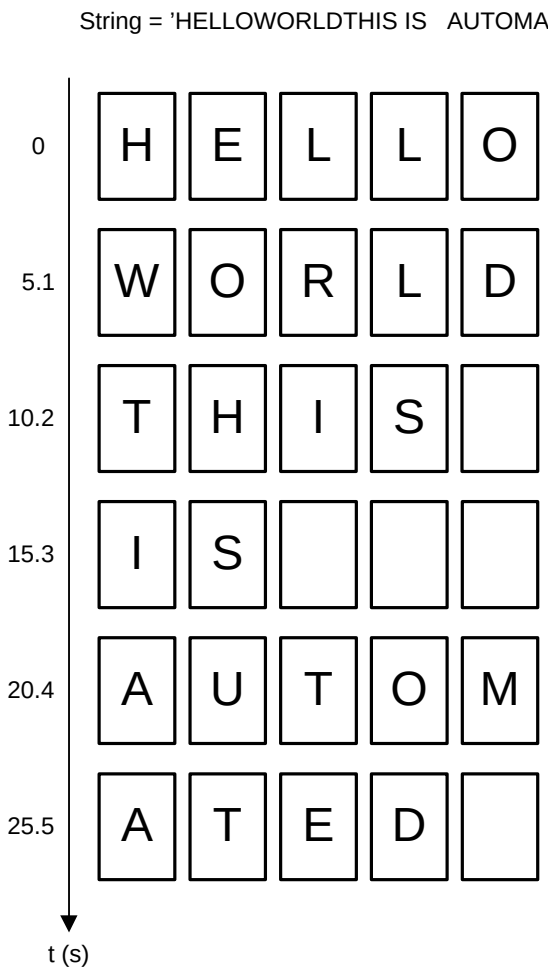
- specify the number of digits to display (ACMCFG.DIGN),
- specify the type of digit (ACMCFG.TDG), see ”ASCII Character Mapping” on page 1043,
- specify the start segment (ACMCFG.STSEG) of the first character of the string,
- select the digit reverse mode (ACMCFG.DREV) if required. This mode is useful if digits have been inverted on the glass (first digit on the right),
- set the display period by writing the number of frame between each display in Frame Counter x in TIM register (TIM.FCx),
- select a frame counter by writing its number in ACMCFG.FCS,
- enable this mode by writing a one to the ACMCFG.EN bit,
- configure PDCA to transfer an ASCII character string of any size (should a multiple of the number of digit). To repeat the character string display, repeat the PDCA transfer.

TIM.FCx defines the number of frames between each PDCA transfer (display update).

$$Nframes = 8 \times (FCx + 1)$$

If frame rate is 50Hz (20ms) in standard waveform mode, characters are displayed with a period from 160ms up to 5.1s.

Figure 39-11. Sequential Character String Example (DIGN = 5, frame rate = 50Hz, TIM.ACMFC = 31)



39.6.11.2 Scrolling of Characters String

This mode displays the same characters string periodically shifted by one character in left direction.

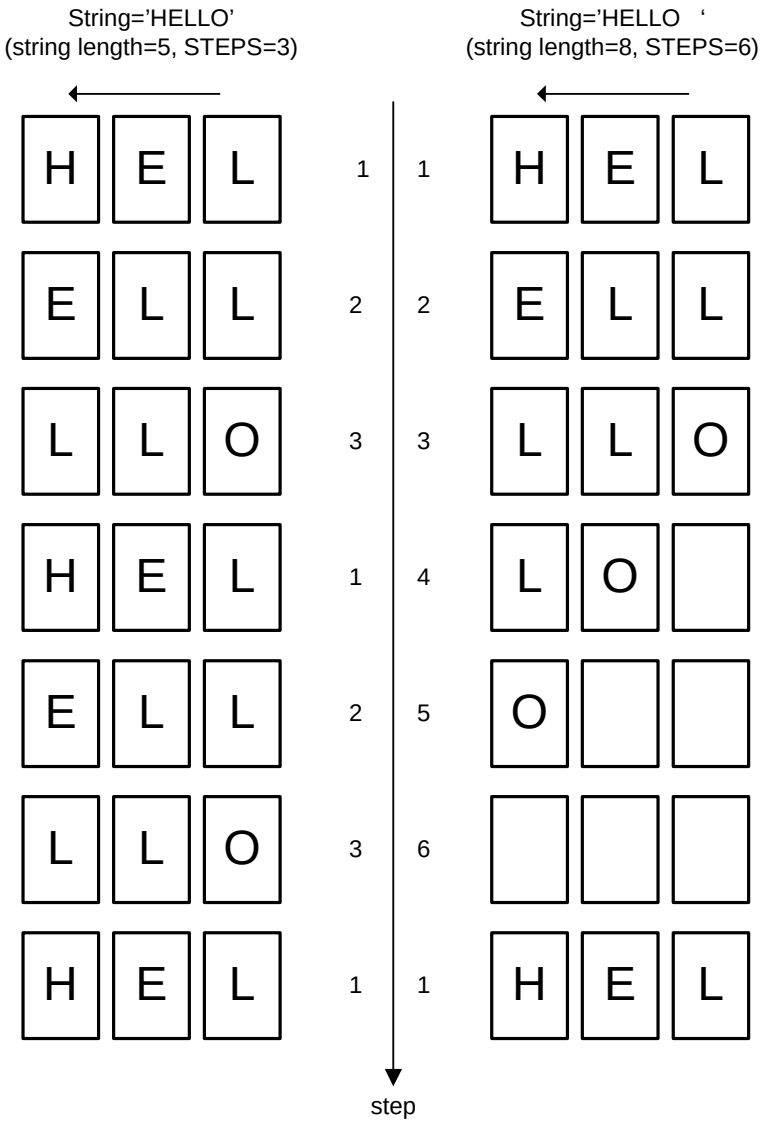
The configuration is:

- specify the number of digit (ACMCFG.DIGN),
- specify the number of scrolling steps ACMCFG.STEPS = string length - DIGN + 1,
- specify the type of digit (ACMCFG.TDG), see "ASCII Character Mapping" on page 1043,
- specify the start segment (ACMCFG.STSEG) of the first character to display
- select the digit reverse mode (ACMCFG.DREV). If enabled ACMCFG.STSEG must be initialized with the start segment corresponding to the last character on the LCD glass.
- set the display period by writing the number of frame in Frame Counter x (TIM.FCx),
- select a frame counter by writing its number in ACMCFG.FCS,
- enable this mode by writing a one to the ACMCFG.EN bit,
- configure PDCA to transfer the ASCII characters string to the Automated Character Mapping Data Register (ACMDR). PDCA must be configured to repeat transfer in order to scroll

character string. Number of transfer must be a multiple of STEPS to get a complete scrolling of the string. Blank characters can be added to string to create a complete scrolling.

The frame counter selected (TIM.FCx) defines the number of frames between each PDCA transfer, refer to ”Sequential Characters String Display” on page 1048

Figure 39-12. Characters String Scrolling Examples (DIGN = 3)



39.6.12 Automated Bit Mapping

Any segment can be modified using the Peripheral DMA Controller, making predefined animations possible. Data can be located in FLASH or RAM and are transferred periodically to the display memory.

When PDCA writes to the Automated Bit Mapping Data Register (ABMDR), up to 8 segments can be modified. PDCA must be configured to transfer a word made of:

- DATA[7:0], each bit represents the state of a segment,

- DMASK[7:0], each bit is a mask for DATA field. When DMASK[x]=1, DATA[x] is not written to display memory,
- OFF[4:0], byte offset in display memory (see Figure 39-3 on page 1036).

To update more than 8 segments, PDCA must transfer multiple words before shadow memory is updated. This number of words must also be written in SIZE field in Automated Bit Mapping Configuration register (ABMCFG), it indicates the number of writes in display memory to form a frame.

To make an automated animation of N states with M segments, PDCA must be configured to transfer N x SIZE (= M/8 or more) words. Note that if segments are at any position in display memory, DMASK is used then PDCA size can be up to N x M.

The display period (animation update) is defined by writing the number of frame in TIM.FCx. The frame counter is selected by writing its number in ABMCFG.FCS.

Animation can be repeated if PDCA is configured to repeat the whole transfer.

39.6.13 Contrast Adjustment

Contrast is defined by the maximum value of V_{LCD}. The higher value the higher contrast.

Fine Contrast value (FCST) in CFG register is a signed value (two's complement) which defines the maximum voltage V_{LCD} on segment and common terminals. New value takes effect at the beginning of next frame.

$$V_{LCD} = 3V + (FCST \times 0,016V)$$

39.6.14 Interrupts

LCDCA can generate an interrupt at the beginning of a frame. When Frame Counter 0 Rollover bit (SR.FC0R) is set to one and interrupt is not masked, LCDCA interrupt is pending.

Moreover Frame Counter 0 (TIM.FC0) can be used to select the interrupt period generation. This mode can provide a useful time base to update LCD.
If TIM.FC0PB=0:

$$\text{Interrupt Period} = ((\text{TIM.FC0} \times 8) + 1) \times \text{Frame Period}$$

If TIM.FC0PB=1:

$$\text{Interrupt Period} = (\text{TIM.FC0} + 1) \times \text{Frame Period}$$

Note that in low power waveform mode, frame period is twice the frame period in standard waveform mode.

An interrupt request will be generated if the corresponding bit in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER) and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in SR is cleared by writing a one to the corresponding bit in Status Clear Register (SCR).

39.6.15 LCD Wake Up

LCD controller can wake up CPU with the interrupt request line. But in sleep modes where APB clocks are off, LCD wake up mechanism must be enabled to wake up CPU.

Wake up mechanism is enabled by writing a one to the Wake up Enable (WEN) bit in configuration register. It is disabled by writing a one to the Wake up Disable bit (WDIS). Moreover LCDCA interrupt request must not be masked (see previous section) and LCDCA bit in Asynchronous Wake Up Enable register (AWEN.LCDCA) must be set to one (see Power Manager chapter).

Wake up signal is generated when frame counter 0 rolls over. When wake up is detected in Power Manager, system clocks are running therefore SR.FC0R is set to one and LCDCA irq is generated. CPU is then woken up.

Wake up signal is cleared by disabling wake up mechanism.

39.6.16 LCD Power Supply

To operate correctly, LCD controller requires a reference level. The External BIAS bit (XBIAS) in CFG register selects the source of V_{LCD}. If XBIAS is zero, V_{LCD} sources voltages from the internal bandgap reference. Otherwise, V_{LCD} must be powered externally.

Note that when using external V_{LCD}, the fine contrast controlled by CFG.FCST is inoperative.

Table 39-11. LCD Power Supply Pins

SR.EN	CFG.XBIAS	VLCD	BIAS2	BIAS1	CAPH / CAPL
0	x	H.Z.	H.Z.	H.Z.	H.Z.
1	0	V_{LCD}	2/3 V_{LCD} (also in static mode)	1/3 V_{LCD} (also in static mode)	Capacitor Pump Charge
	1	Input for VLCD	- Input for BIAS2 - H.Z. if static bias	- Input for BIAS1 - H.Z. if static bias	H.Z.

Figure 39-13. LCD Power Supply Block Diagram

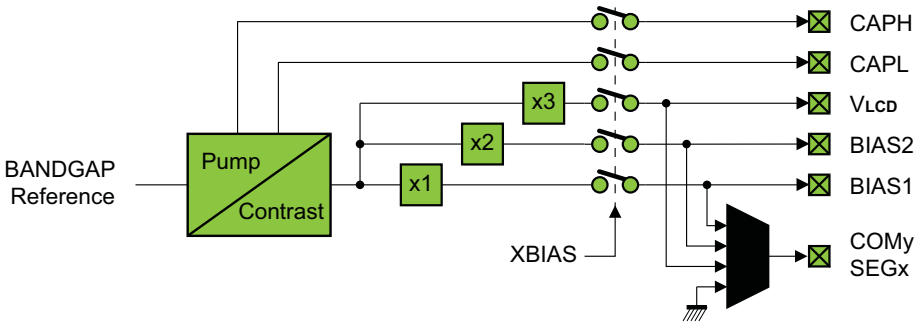
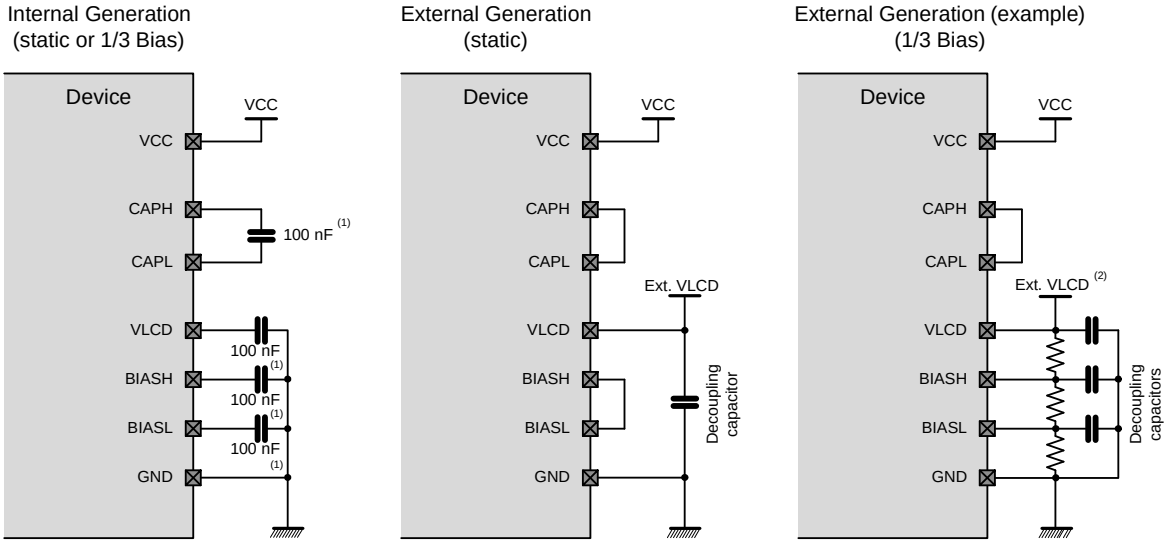


Figure 39-14. Internal and External Bias Generation



(1) Values are given for design guidance only.
(2) Bias generation can be provided by other voltage source than a divison resistor

39.7 User Interface

Table 39-12. LCDCA Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Control Register	CR	Write-Only	-
0x04	Configuration Register	CFG	Read/Write	0x00000000
0x08	Timing Register	TIM	Read/Write	0x00000000
0x0C	Status Register	SR	Read-Only	0x00000000
0x10	Status Clear Register	SCR	Write-Only	-
0x14	Data Register Low 0	DRL0	Read/Write	-
0x18	Data Register High 0	DRH0	Read/Write	-
0x1C	Data Register Low 1	DRL1	Read/Write	-
0x20	Data Register High 1	DRH1	Read/Write	-
0x24	Data Register Low 2	DRL2	Read/Write	-
0x28	Data Register High 2	DRH2	Read/Write	-
0x2C	Data Register Low 3	DRL3	Read/Write	-
0x30	Data Register High 3	DRH3	Read/Write	-
0x34	Indirect Access Data Register	IADR	Write-Only	-
0x38	Blink Configuration Register	BCFG	Read/Write	0x00000000
0x3C	Circular Shift Register Configuration	CSRCFG	Read/Write	0x00000000
0x40	Character Mapping Configuration Register	CMCFG	Read/Write	0x00000000
0x44	Character Mapping Data Register	CMDR	Write-Only	-
0x48	Automated Character Mapping Configuration Register	ACMCFG	Read/Write	0x00000000
0x4C	Automated Character Mapping Data Register	ACMDR	Write-Only	-
0x50	Automated Bit Mapping Configuration Register	ABMCFG	Read/Write	0x00000000
0x54	Automated Bit Mapping Data Register	ABMDR	Write-Only	-
0x58	Interrupt Enable Register	IER	Write-Only	-
0x5C	Interrupt Disable Register	IDR	Write-Only	-
0x60	Interrupt Mask Register	IMR	Read-Only	0x00000000
0x64	Version Register	VERSION	Read-Only	-(1)

Note: 1. The reset value for this register is device specific. Please refer to the Module Configuration section at the end of this chapter.

39.7.1 Control Register

Name: CR

Access Type: Write-Only

Offset: 0x00

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	CSTOP	CSTART	BSTOP	BSTART	WEN	WDIS	CDM

7	6	5	4	3	2	1	0
FC2EN	FC2DIS	FC1EN	FC1DIS	FC0EN	FC0DIS	EN	DIS

Writing a zero to a bit in this register has no effect.

- **CSTOP: Circular Shift Stop**
  Writing a one to this bit stops circular shift register.
- **CSTART: Circular Shift Start**
  Writing a one to this bit starts circular shift register.
- **BSTOP: Blinking Stop**
  Writing a one to this bit stops blinking.
- **BSTART: Blinking Start**
  Writing a one to this bit starts blinking.
- **WEN: Wake up Enable**
  Writing a one to this bit enables wake up mechanism.
- **WDIS: Wake up Disable**
  Writing a one to this bit disables wake up mechanism.
- **CDM: Clear Display Memory**
  Writing a one to this bit clears immediately the display memory.
- **FC2EN: Frame Counter 2 Enable**
  Writing a one to this bit enables the frame counter 2.
- **FC2DIS: Frame Counter 2 Disable**
  Writing a one to this bit disables the frame counter 2.
- **FC1EN: Frame Counter 1 Enable**
  Writing a one to this bit enables the frame counter 1.
- **FC1DIS: Frame Counter 1 Disable**
  Writing a one to this bit disables the frame counter 1.
- **FC0EN: Frame Counter 0 Enable**
  Writing a one to this bit enables the frame counter 0.
- **FC0DIS: Frame Counter 0 Disable**
  Writing a one to this bit disables the frame counter 0.
- **EN: Enable**
  Writing a one to this bit enables the LCD controller.

- **DIS: Disable**
  Writing a one to this bit disables the LCD controller.

39.7.2 Configuration Register

Name: CFG

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	NSU					

23	22	21	20	19	18	17	16
-	-	FCST					

15	14	13	12	11	10	9	8
-	-	-	-	-	-	DUTY	

7	6	5	4	3	2	1	0
-	-	-	-	LOCK	BLANK	WMOD	XBIAS

NSU, DUTY, WMOD and XBIAS shall not be modified when LCDCA is enabled (SR.EN=1).

- **NSU: Number of Segment Terminals in Use**
  This field indicates the number of segment terminals in use. It prevents any activity on muxes driving the unused segment terminals.
- **FCST: Fine Contrast**
  Defines the maximum voltage V_{LCD} on segment and common terminals. FCST is a signed number (two's complement). New value takes effect at the beginning of next frame.
  V_{LCD} = 3.0 V + (FCST[5:0] * 0.016 V)
- **DUTY: Duty Select**
  This field defines the duty cycle.

Table 39-13. Duty Selection

DUTY[1:0]	Duty	Bias	COM pins Used
0 0	1/4	1/3	COM[0:3]
0 1	Static	Static	COM0
1 0	1/2	1/3	COM[0:1]
1 1	1/3	1/3	COM[0:2]

- **LOCK: Lock**
  0: Shadow display memory is unlocked.
  1: Shadow display memory is locked and can't be updated.
- **BLANK: Blank LCD**
  0: LCD segments value is defined in shadow display memory.
  1: Turns OFF all LCD segments.
- **WMOD: Waveform Mode**
  0: low power waveform mode.
  1: standard waveform mode.

If this bit is modified during display operation the waveform mode is applied at the beginning of next frame.

- **XBIAS: External Bias Generation**
  0:Internal bias is used.
  1:External bias is used.

39.7.3 Timing Register

Name: TIM

Access Type: Read/Write

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	FC2				

23	22	21	20	19	18	17	16
-	-	-	FC1				

15	14	13	12	11	10	9	8
-	-	FC0PB	FC0				

7	6	5	4	3	2	1	0
-	-	-	-	CLKDIV			PRESC

CLKDIV and PRESC shall not be modified when LCD controller is enabled.

FCx shall not be modified when Frame Counter x is enabled.

- **FC2: Frame Counter 2**
  Number of frame before rollover = ((FC2 * 8) + 1).
- **FC1: Frame Counter 1**
  Number of frame before rollover = ((FC1 * 8) + 1).
- **FC0PB: Frame Counter 0 Prescaler Bypass**
  0: FC Prescaler is not bypassed
  1: FC Prescaler is bypassed
- **FC0: Frame Counter 0**
  Number of frame before rollover = ((FC0 * 8) + 1) if FC0PB=0 else (FC0 + 1).
- **CLKDIV: LCD Clock Division**
  Defines the LCD frame rate.

$$FrameRate = \frac{F(clk_{LCD})}{(K \times N \times (1 + CLKDIV) \times 2^{(1 - WMOD)})}$$

  K = 8 for 1/4, 1/2 and static duty.
  K = 6 for 1/3 duty.
- **PRESC: LCD Prescaler Select**
  0: N = 8
  1: N = 16

39.7.4 Status Register

Name: SR

Access Type: Read-Only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
				-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	CPS
7	6	5	4	3	2	1	0
CSRS	BLKS	WEN	EN	FC2S	FC1S	FC0S	FC0R

- **CPS: Charge Pump Status**
  0: Charge pump is inactive or not ready.
  1: Charge pump is ready.
- **CSRS: Circular Shift Register Status**
  0: CSR is not running.
  1: CSR is running.
- **BLKS: Blink Status**
  0: Blinking is not running.
  1: Blinking is running.
- **WEN: Wake up Status**
  0: Wake up mechanism is disabled.
  1: Wake up mechanism is enabled.
- **EN: LCDCA Status**
  0: LCD controller is disabled.
  1: LCD controller is enabled.
- **FC2S: Frame Counter 2 Status**
  0: Frame counter 2 is stopped.
  1: Frame counter 2 is running.
- **FC1S: Frame Counter 1 Status**
  0: Frame counter 1 is stopped.
  1: Frame counter 1 is running.
- **FC0S: Frame Counter 0 Status**
  0: Frame counter 0 is stopped.
  1: Frame counter 0 is running.
- **FC0R: Frame Counter 0 Rollover**
  This bit is set when frame counter 0 rollover.
  This bit is cleared when corresponding bit SCR is written to one.

39.7.5 Status Clear Register

Name: SCR

Access Type: Write-Only

Offset: 0x10

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	FC0R

Writing a zero to a bit in this register has no effect.
Writing a one to a bit clears the corresponding SR bit.

39.7.6 Data Register Low

Name: DRLx

Access Type: Read/Write

Offset: 0x14+8*x

Reset Value: -

31	30	29	28	27	26	25	24
DATA[31:24]							

23	22	21	20	19	18	17	16
DATA[23:16]							

15	14	13	12	11	10	9	8
DATA[15:8]							

7	6	5	4	3	2	1	0
DATA[7:0]							

Display memory is not initialized at startup.

- **DATA: Segments Value**
  Each bit defines the segment value in display memory.

39.7.7 Data Register High

Name: DRHx

Access Type: Read/Write

Offset: 0x18+8*x

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
DATA							

- **DATA: Segments Value**
  Each bit defines the segment value in display memory.

39.7.8 Indirect Access Data Register

Name: IADR

Access Type: Write-Only

Offset: 0x34

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	OFF				

15	14	13	12	11	10	9	8
DMASK							

7	6	5	4	3	2	1	0
DATA							

- **OFF: Byte Offset**
  Byte offset in display memory.
- **DMASK: Data Mask**
  Each bit is a mask for DATA field. When DMASK[x]=1, DATA[x] is not written to display memory.
- **DATA: Segments Value**
  Each bit defines the segment value.

39.7.9 Blink Configuration Register

Name: BCFG

Access Type: Read/Write

Offset: 0x38

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
BSS1				BSS0			

7	6	5	4	3	2	1	0
-	-	-	-	-	FCS		MODE

- **BSS1: Blink Segment Selection 1**
  If BSS1[x] is set, segment connected to SEG1/COMx is selected.
- **BSS0: Blink Segment Selection 0**
  If BSS0[x] is set, segment connected to SEG0/COMx is selected.
- **FCS: Frame Counter Selection**

Table 39-14. Frame Counter Selection

FCS	Frame Counter
00	FC0
01	FC1
10	FC2
11	reserved

- **MODE: Blinking Mode**
  0: All segments are allowed to blink.
  1: Selected segments are allowed to blink.

39.7.10 Circular Shift Register Configuration

Name: CSRCFG

Access Type: Read/Write

Offset: 0x3C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
DATA							

7	6	5	4	3	2	1	0
-	-	SIZE			FCS		DIR

- **DATA: Circular Shift Register Value**

- **SIZE: Size**
  Defines the size of the circular shift register, (SIZE + 1) bits.
- **FCS: Frame Counter Selection**

Table 39-15. Frame Counter Selection

FCS	Frame Counter
00	FC0
01	FC1
10	FC2
11	reserved

- **DIR: Direction**
  0: Left shifting.
  1: Right shifting.

39.7.11 Character Mapping Configuration Register

Name: CMCFG

Access Type: Read/Write

Offset: 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	STSEG					

7	6	5	4	3	2	1	0
-	-	-	-	-	TDG		DREV

- **STSEG: Start Segment**
  Defines the first segment terminal used to write the decoded display.
- **TDG: Type of Digit**

Table 39-16. Type of Digit

TDG	Digit
00	7-segment with 3 common terminals, COM[2:0]
01	7-segment with 4 common terminals, COM[3:0]
10	14-segment with 4 common terminals, COM[3:0]
11	16-segment with 3 common terminals, COM[2:0]

- **DREV: Digit Reverse Mode**
  0: Digit reverse mode is disabled.
  1: Digit reverse mode is enabled.

39.7.12 Character Mapping Data Register

Name: CMDR

Access Type: Write-Only

Offset: 0x44

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	ASCII						

- **ASCII: ASCII Code**

39.7.13 Automated Character Mapping Configuration Register

Name: ACMCFG

Access Type: Read/Write

Offset: 0x48

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	DIGN			

23	22	21	20	19	18	17	16
STEPS							

15	14	13	12	11	10	9	8
-	-	STSEG					

7	6	5	4	3	2	1	0
-	TDG		DREV	MODE	FCS		EN

- **DIGN: Digit Number**
  Defines the number of digit used (must be >1).
- **STEPS: Scrolling Steps**
  Defines the number of steps in scrolling mode. STEPS = string length - DIGN + 1.
- **STSEG: Start Segment**
  Defines the first segment terminal used to write the decoded display.
- **TDG: Type of Digit**

Table 39-17. Type of Digit

TDG	Digit
00	7-segment with 3 common terminals, COM[2:0]
01	7-segment with 4 common terminals, COM[3:0]
10	14-segment with 4 common terminals, COM[3:0]
11	16-segment with 3 common terminals, COM[2:0]

- **DREV: Digit Reverse**
  0: Digit reverse mode is disabled.
  1: Digit reverse mode is enabled.
- **MODE: Mode**
  0: Sequential.
  1: Scrolling.

- **FCS: Frame Counter Selection**

Table 39-18. Frame Counter Selection

FCS	Frame Counter
00	FC0
01	FC1
10	FC2
11	reserved

- **EN: Enable**
  0: Automated character mapping is disabled.
  1: Automated character mapping is enabled.

39.7.14 Automated Character Mapping Data Register

Name: ACMDR

Access Type: Write-Only

Offset: 0x4C

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	ASCII						

- **ASCII: ASCII Code**

39.7.15 Automated Bit Mapping Configuration Register

Name: ABMCFG

Access Type: Read/Write

Offset: 0x50

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-					

15	14	13	12	11	10	9	8
-	-	-	SIZE				

7	6	5	4	3	2	1	0
-	-	-	-	-	FCS		EN

- **SIZE: Size**
  Defines the number of PDCA writes to ABMDR to form a frame (must be >1).
- **FCS: Frame Counter Selection**

Table 39-19. Frame Counter Selection

FCS	Frame Counter
00	FC0
01	FC1
10	FC2
11	reserved

- **EN: Enable**
  0: Automated bit mapping is disabled.
  1: Automated bit mapping is enabled.

39.7.16 Automated Bit Mapping Data Register

Name: ABMDR

Access Type: Write-Only

Offset: 0x54

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	OFF				

15	14	13	12	11	10	9	8
DMASK							

7	6	5	4	3	2	1	0
DATA							

- **OFF: Byte Offset**
  Byte offset in display memory.
- **DMASK: Data Mask**
  Each bit is a mask for DATA field. When DMASK[x]=1, DATA[x] is not written to display memory.
- **DATA: Segments Value**
  Each bit defines the segment value.

39.7.17 Interrupt Enable Register

Name: IER

Access Type: Write-Only

Offset: 0x58

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	FC0R

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

39.7.18 Interrupt Disable Register

Name: IDR

Access Type: Write-Only

Offset: 0x5C

Reset Value: -

31	30	29	28	27	26	25	24
	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	FC0R

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

39.7.19 Interrupt Mask Register

Name: IMR

Access Type: Read-Only

Offset: 0x60

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-		-	-	-

23	22	21	20	19	18	17	16
-	-	-	-		-	-	-

15	14	13	12	11	10	9	8
-	-	-	-		-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	FC0R

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

39.7.20 Module Version

Name: VERSION

Access Type: Read-Only

Offset: 0x64

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

39.8 Module Configuration

The specific configuration for LCDCA is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. ”Power Manager (PM)” on page 109 for details.

Table 39-20. LCDCA Clocks

Clock Name	Description
CLK_LCDCA	LCDCA bus interface clock
CLK_LCD	LCD 32kHz clock

Table 39-21. Register Reset Values

Register	Reset Value
VERSION	0x00000100

Table 39-22. LCDCA Pinout vs Packages

Pin	Cluster	Package		
		TQFP100	TQFP64/QFN64	TQFP48/QFN48
PA12	A	COM0	COM0	COM0
PA11		COM1	COM1	COM1
PA10		COM2	COM2	COM2
PA09		COM3	COM3	COM3
PC15		SEG0		
PC16		SEG1		
PC17		SEG2		
PC18		SEG3		
PC19		SEG4		
PA13		SEG5	SEG0	SEG0
PA14		SEG6	SEG1	SEG1
PA15		SEG7	SEG2	SEG2
PA16		SEG8	SEG3	SEG3
PA17		SEG9	SEG4	SEG4
PC20		SEG10		
PC21		SEG11		
PC22		SEG12		
PC23		SEG13		
PB08		SEG14	SEG5	
PB09		SEG15	SEG6	
PB10		SEG16	SEG7	
PB11		SEG17	SEG8	
PA18		SEG18	SEG9	SEG5
PA19		SEG19	SEG10	SEG6
PA20		SEG20	SEG11	SEG7
PB07		SEG21	SEG12	
PB06		SEG22	SEG13	
PA08		SEG23	SEG14	SEG8

Table 39-22. LCDCA Pinout vs Packages

Pin	Cluster	Package		
		TQFP100	TQFP64/QFN64	TQFP48/QFN48
PC24	B	SEG24		
PC25		SEG25		
PC26		SEG26		
PC27		SEG27		
PC28		SEG28		
PC29		SEG29		
PC30		SEG30		
PC31		SEG31		
PB12	C	SEG32	SEG15	
PB13		SEG33	SEG16	
PA21		SEG34	SEG17	SEG9
PA22		SEG35	SEG18	SEG10
PB14		SEG36	SEG19	
PB15		SEG37	SEG20	
PA23		SEG38	SEG21	SEG11
PA24		SEG39	SEG22	SEG12

40. Parallel Capture (PARC)

Rev: 1.0.0.0

40.1 Features

- **Captures 8-bits data with external input clock**
- **External data enables supported**
- **Various enable conditions**
- **Peripheral DMA supported**
- **Peripheral events supported**

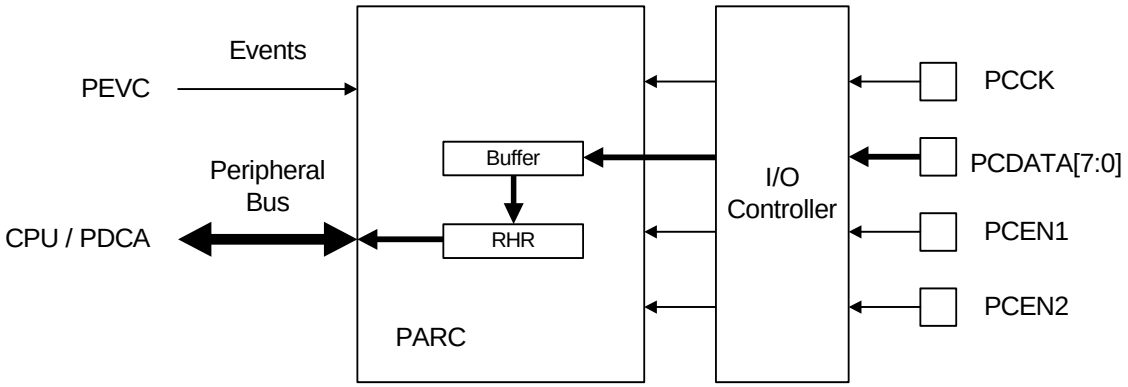
40.2 Overview

The Parallel Capture peripheral samples an external 8-bit bus with an external input clock. It can be connected to a CMOS digital image sensor, an ADC, a DSP synchronous port,...

The number of PARC modules implemented is device specific. Refer to the Module Configuration section for details.

40.3 Block Diagram

Figure 40-1. PARC Block Diagram



40.4 I/O Lines Description

Table 40-1. I/O Lines Description

Pin Name	Pin Description	Type
PCCK	Clock	Input
PCD[7:0]	Data	Input
PCEN1	Data Enable 1	Input
PCEN2	Data Enable 2	Input

40.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

40.5.1 I/O Lines

The PARC pins are multiplexed with other peripherals. The user must first configure the I/O Controller to give control of the pins to the PARC.

40.5.2 Power Management

PARC stops functioning when the system enters a sleep mode that disables its clock.

40.5.3 Clocks

The clock for PARC (CLK_PARC) is generated by the Power Manager. It can be disabled either manually through the user interface of the Power Manager or automatically when the system enters a sleep mode that disables the clocks to the peripheral bus modules. For correct behavior, CLK_PARC frequency must be at least twice the PCCK frequency.

40.5.4 DMA

The PARC DMA handshake interface is connected to the Peripheral DMA Controller (PDCA). Using the PARC DMA functionality requires the PDCA to be configured first.

40.5.5 Interrupt

The PARC interrupt request line is connected to the NVIC. Using the PARC interrupt requires the NVIC to be configured first.

40.5.6 Peripheral Events

The PARC peripheral events are connected via the Peripheral Event System. Refer to Section 31. ”Peripheral Event Controller (PEVC)” on page 845 for details.

40.6 Functional Description

40.6.1 Capture Operation

PARC is enabled by writing a one to the Enable bit in the Control Register (CR.EN). Data capture is enabled by writing a one to the Start bit in the Control Register (CR.START) and stopped by writing a one to the Stop bit in the Control Register (CR.STOP).

Data capture is made by sampling the data bus PCD[7:0] on the rising or falling edge of the PCCK input clock then re-synchronized to the PB clock domain. PCCK sampling edge is selected with EDGE bit in the Configuration Register (CFG.EDGE). User can select a sampling condition to capture the data. There are four modes defined by Sampling Mode field (CFG.SMODE): when PCEN1 is high, when PCEN1 and PCEN2 are high, when PCEN1 or PCEN2 is high and in last mode data is sampled without condition.

Data can be captured every two cycles if CFG.HALF bit is set to one. It can be used, for example, to capture the luminance Y of a CMOS digital image sensor. In addition, bit CFG.ODD specifies which of odd or even bytes are captured. Considering that first byte captured (byte 0), after reset, is an even byte.

Captured data are stored in the Receive Holding Register (RHR). Concatenated data can also be stored in RHR to make a 16-bit or a 32-bit data, with the first byte received in LSB position. Concatenated data size is configured by Data Size bit in CFG register (CFG.DSIZE). When the configured data bytes are captured, the Data Ready bit is set to one in the Status Register (SR.DRDY). DRDY is also set to one when internal buffer is not full but CR.STOP is set to one or stop event occurs (see “Peripheral Events”).

An overrun condition is detected if RHR is not read before internal buffer is full. Data in RHR is corrupted and Overrun bit is set to one in the Status Register (SR.OVR).

40.6.2 Peripheral DMA

PARC can be associated to a Peripheral DMA Controller (PDCA) channel. It will then perform data transfer from PARC to a memory buffer without any CPU intervention (see PDCA chapter for channel configuration). PDCA and PARC data size must be equal, if CFG.DSIZE is 32-bit PDCA must be configured to read 32-bit data size.

PARC requests data transfer when internal buffer is full. When data capture is disabled (with CR.STOP), PARC requests a last data transfer if internal buffer contains at least one byte.

40.6.3 Peripheral Events

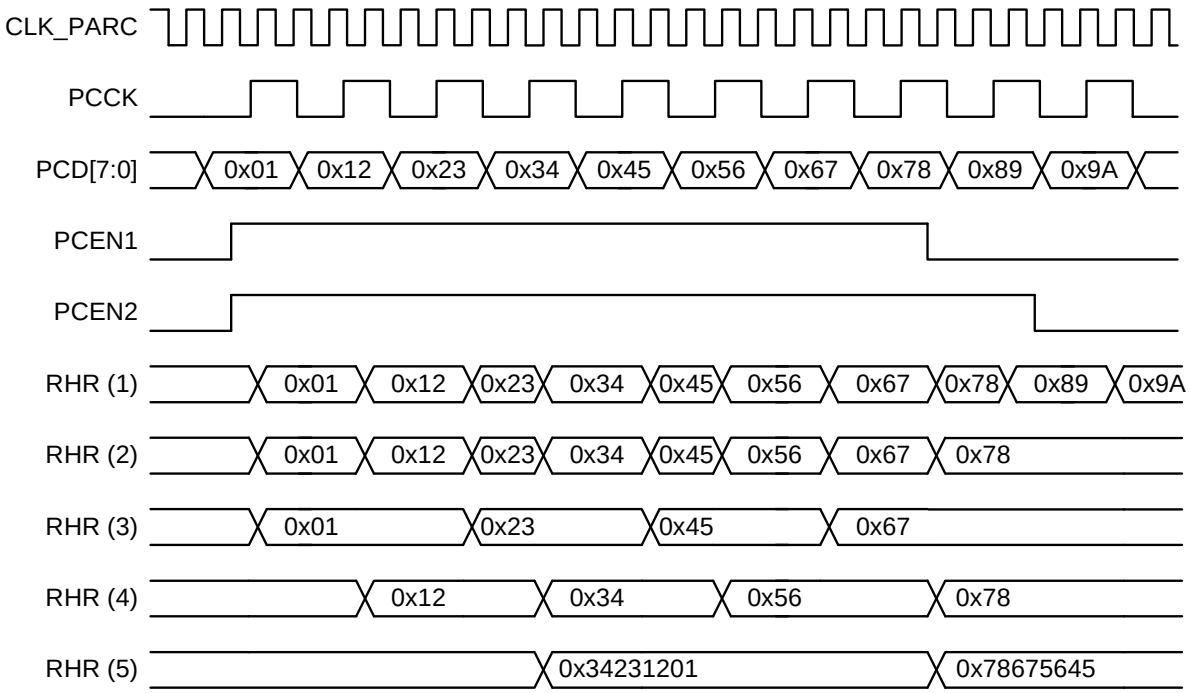
Data capture can be enabled or disabled by peripheral events if Event Mode is enabled (CFG.EMODE). Start event enables the data capture whereas stop event disables it.

40.6.4 Interrupt Generation

PARC has two interrupt sources, Data Ready (DRDY) and Overrun (OVR). The status of each interrupt source can be read from the Status Register. An interrupt request will be generated if a bit in SR and the corresponding bit in the Interrupt Mask Register (IMR) are set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in SR is cleared by writing a one to the corresponding bit in the Interrupt Status Clear Register (ICR).

The interrupt sources are ORed together to make one interrupt request which remains active until all interrupt bits in SR are cleared.

Figure 40-2. Parallel Capture Waveforms



(1) DSIZE=0, ALWYS=1, HALFS=0, ODD=0
(2) DSIZE=0, ALWYS=0, HALFS=0, ODD=0
(3) DSIZE=0, ALWYS=0, HALFS=1, ODD=0
(4) DSIZE=0, ALWYS=0, HALFS=1, ODD=1
(5) DSIZE=2, ALWYS=0, HALFS=0, ODD=0

40.7 User Interface

Table 40-2. PARC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Configuration Register	CFG	Read/Write	0x00000000
0x04	Control Register	CR	Read/Write	0x00000000
0x08	Interrupt Enable Register	IER	Write-only	0x00000000
0x0C	Interrupt Disable Register	IDR	Write-only	0x00000000
0x10	Interrupt Mask Register	IMR	Read-only	0x00000000
0x14	Status Register	SR	Read-only	0x00000000
0x18	Interrupt Status Clear Register	ICR	Write-only	0x00000000
0x1C	Receive Holding Register	RHR	Read	0x00000000
0x20	Version Register	VERSION	Read-only	-(1)

Note: 1. The reset values for these registers are device specific. Refer to the Module Configuration section at the end of this chapter.

40.7.1 Configuration Register

Name: CFG

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
ODD	HALF	EDGE	EMODE	SMODE		DSIZE	

To avoid unexpected behavior CFG must be written when PARC is disabled.

- **ODD: Odd Capture**
  0: only bytes with even index are captured.
  1: only bytes with odd index are captured.
- **HALF: Half Capture**
  0: all bytes are captured.
  1: one out of two bytes are captured.
- **EDGE: Sampling Edge Select**
  0: rising edge.
  1: falling edge.
- **EMODE: Events Mode**
  0: Events mode disabled.
  1: Events mode enabled. Peripheral events start/stop data capture.
- **SMODE: Sampling Mode**

SMODE		Mode
0	0	Capture data if PCEN1 is high
0	1	Capture data if PCEN1 and PCEN2 are high
1	0	Capture data if PCEN1 or PCEN2 is high
1	1	Always capture data

- **DSIZE: Data Size**

DSIZE		Size
0	0	data size in RHR is a byte (8-bit)
0	1	data size in RHR is a half-word (16-bit)
1	0	data size in RHR is a word (32-bit)
1	1	data size in RHR is a word (32-bit)

40.7.2 Control Register

Name: CR

Access Type: Read/Write

Offset: 0x04

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	-	-	STOP	START	DIS	EN

- **STOP: Stop Capture**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables data capture.
  This bit always reads as zero.
- **START: Start Capture**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables data capture.
  This bit always reads as zero.
- **DIS: Disable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables PARC.
  This bit always reads as zero.
- **EN: Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables PARC.
  This bit always reads as zero.

40.7.3 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x08

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	OVR	DRDY	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR.

40.7.4 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	OVR	DRDY	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

40.7.5 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

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

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
A bit in this register is cleared when the corresponding bit in IDR is written to one.
A bit in this register is set when the corresponding bit in IER is written to one.

40.7.6 Status Register

Name: SR

Access Type: Read-only

Offset: 0x14

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	-	-	OVR	DRDY	CS	EN

- **OVR: Overrun**
  - 0: No overrun error occurred since last read of RHR.
  - 1: At least one overrun error occurred since last read of RHR.
  - This bit is cleared when the corresponding bit in ICR is written to one.
  - This bit is set when a new byte has been captured and previous data in RHR has not been read.
- **DRDY: Data Ready**
  - 0: No data is ready in RHR.
  - 1: A new data is ready.
  - This bit is cleared when the corresponding bit in ICR is written to one.
  - This bit is set when a new data is captured (according to CR.DSIZE).
- **CS: Capture Status**
  - 0: PARC is not in capture mode.
  - 1: PARC is in capture mode.
- **EN: Enable Status**
  - 0: PARC is disabled.
  - 1: PARC is enabled.

40.7.7 Interrupt Status Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x18

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	OVR	DRDY	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

40.7.8 Receive Holding Register

Name: RHR

Access Type: Read/Write

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
CDATA[31:24]							

23	22	21	20	19	18	17	16
CDATA[23:16]							

15	14	13	12	11	10	9	8
CDATA[15:8]							

7	6	5	4	3	2	1	0
CDATA[7:0]							

- **CDATA: Captured Data**
  Captured data size is defined by CR.DSIZE.

40.7.9 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0x20

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

40.8 Module Configuration

The specific configuration for each PARC instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 40-3. PARC Clock Name

Module Name	Clock Name	Description
PARC	CLK_PARC	Peripheral clock for PARC

Table 40-4. Register Reset Values

Register	Reset Value
VERSION	0x00000100

41. Cyclic Redundancy Check Calculation Unit (CRCCU)

Rev: 2.0.2.0

41.1 Features

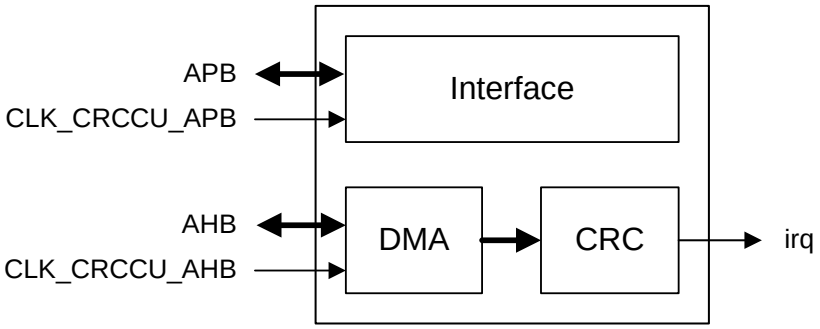
- **Performs Cyclic Redundancy Check Operation on Memory Area**
- **Single AHB Master Interface**
- **APB Configuration Interface**

41.2 Overview

The CRCCU performs CRC check on memory area.

41.3 Block Diagram

Figure 41-1. CRCCU Block Diagram



41.4 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

41.4.1 Power Management

CRCCU stops functioning when the system enters a sleep mode that disables its clock. CRCCU resumes operation after the system wakes up from sleep mode.

41.4.2 Clocks

CRCCU clocks (CLK_CRCCU_APB, CLK_CRCCU_AHB) are generated by the Power Manager. they can be disabled either manually through the user interface of the Power Manager or automatically when the system enters a sleep mode that disables the clocks to the peripheral bus modules.

41.4.3 Interrupts

The CRCCU interrupt request line is connected to the NVIC. Using the CRCCU interrupt requires the NVIC to be configured first.

41.4.4 Debug Operation

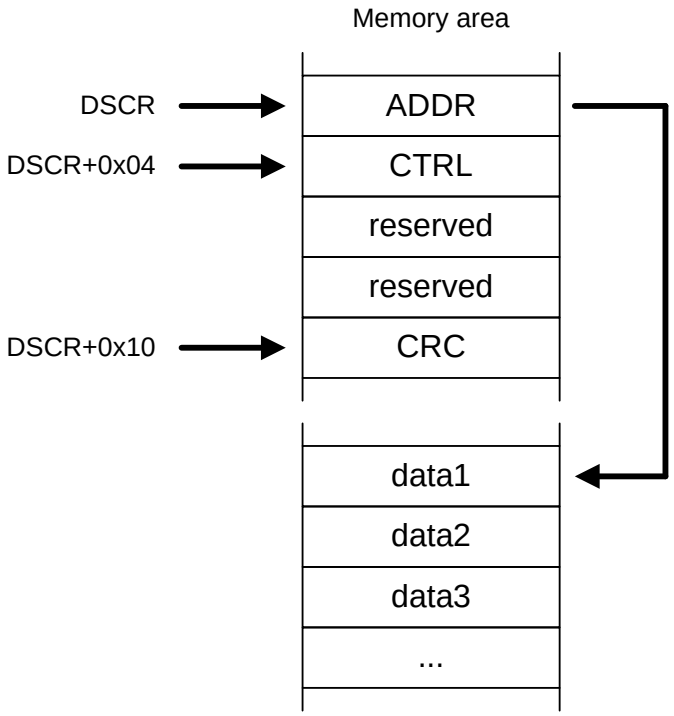
When an external debugger forces the CPU into debug mode, CRCCU continues normal operation. If the CRCCU is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

41.5 Functional Description

Once configured (Mode Register, MR) and enabled by writing a one to MR.ENABLE, the CRC engine performs a checksum computation from memory data. CRC computation is performed from LSB to MSB bit. Three different polynomials (CCIT802.3, CASTAGNOLI, CCIT16) can be configured in MR.PTYPE.

CRCCU uses its own DMA mechanism to read memory area (Flash or RAM area). DMA uses a descriptor located in memory area. Descriptor location is defined by the Descriptor Base Address Register (DSCR).

Figure 41-2. CRCCU Descriptor



Once enabled (DMAEN register), DMA reads descriptor to get instructions:

- ADDR returns the address of memory area to compute,
- CTRL.TRWIDTH indicates the transfer size (byte, halfword or word),
- CTRL.BTSIZE indicates the buffer size,
- CTRL.IEN enables the transfer-complete interrupt.

Then reads data located at ADDR and CRC engine computes the checksum. The CRC result is available in Status Register (SR). BTSIZE is automatically decremented after each read. When BTSIZE is zero, DMA is stopped and the status bit DMASR in DMASR register is set to zero.

If MR.COMPARE is set to one, CRC register in descriptor is compared with the last CRC computed. If a mismatch occurs, the error bit ERRISR in ISR register is set to one and interrupt is generated (if not masked, see IER/IDR/IMR register).

CRCCU makes single access (TRWIDTH size) to memory in order to limit the bandwidth usage. The field DIVIDER in MR can be used to lower the bandwidth by dividing the frequency of single accesses. The transfer request frequency is then divided by 2^(DIVIDER+1).

To compute CRC for a memory size larger than 256Kbytes of for non-contiguous memory area, CRCCU can be enabled again for a new memory area. CRC will be updated accordingly. Conversely, set CR.RESET to one to reset the intermediate CRC to its default value (0xFFFFFFFF).

41.6 User Interface

Table 41-1. CRCCU Register Memory Map

Offset	Register	Register Name	Access	Reset
0x00	Descriptor Base Register	DSCR	Read/Write	0x00000000
0x08	DMA Enable Register	DMAEN	Write-only	-
0x0C	DMA Disable Register	DMADIS	Write-only	-
0x10	DMA Status Register	DMASR	Read-only	0x00000000
0x14	DMA Interrupt Enable Register	DMAIER	Write-only	-
0x18	DMA Interrupt Disable Register	DMAIDR	Write-only	-
0x1C	DMA Interrupt Mask Register	DMAIMR	Read-only	0x00000000
0x20	DMA Interrupt Status Register	DMAISR	Read-only	0x00000000
0x34	Control Register	CR	Write-only	-
0x38	Mode Register	MR	Read/Write	0x00000000
0x3C	Status Register	SR	Read-only	0xFFFFFFFF
0x40	Interrupt Enable Register	IER	Write-only	-
0x44	Interrupt Disable Register	IDR	Write-only	-
0x48	Interrupt Mask Register	IMR	Read-only	0x00000000
0x4C	Interrupt Status Register	ISR	Read-only	0x00000000
0xFC	Version Register	VERSION	Read-only	-(1)

Notes: 1. The reset value is device specific. Refer to the Module Configuration section at the en of this chapter.

Table 41-2. CRCCU Register Memory Map (RAM)

Offset	Register	Register Name	Access	Reset
DSCR	Address Register	ADDR	Read/Write	-
DSCR + 0x04	Control Register	CTRL	Read/Write	-
DSCR + 0x10	CRC Reference Register	CRC	Read/Write	-

41.6.1 Descriptor Base Address Register

Name: DSCR

Access Type: Read/Write

Offset: 0x00

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DSCR[22:15]							
23	22	21	20	19	18	17	16
DSCR[14:7]							
15	14	13	12	11	10	9	8
DSCR[6:0]							-
7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

- **DSCR: Description Base Address**
  Address of CRC descriptor (512-byte aligned).

41.6.2 DMA Enable Register

Name: DMAEN

Access Type: Write-only

Offset: 0x08

Reset Value: -

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

- **DMAEN: DMA Enable**
  Write a one to enable DMA channel.
  Writing a zero has no effect.

41.6.3 DMA Disable Register

Name: DMADIS

Access Type: Write-only

Offset: 0x0C

Reset Value: -

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

- **DMADIS: DMA Disable**
  Write a one to disable DMA channel.
  Writing a zero has no effect.

41.6.4 DMA Status Register

Name: DMASR

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	DMASR

- **DMASR: DMA Channel Status**
  0: DMA channel is disabled.
  1: DMA channel is enabled.

41.6.5 DMA Interrupt Enable Register

Name: DMAIER

Access Type: Write-only

Offset: 0x14

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	DMAIER

- **DMAIER: DMA Interrupt Enable**
  Write a one to enable DMA interrupt.
  Writing a zero has no effect.

41.6.6 DMA Interrupt Disable Register

Name: DMAIDR

Access Type: Write-only

Offset: 0x18

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	DMAIDR

- **DMAIDR: DMA Interrupt Disable**
  Write a one to disable DMA interrupt.
  Writing a zero has no effect.

41.6.7 DMA Interrupt Mask Register

Name: DMAIMR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000000

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

- **DMAIMR: DMA Interrupt Mask Status**
  0: DMA interrupt is disabled.
  1: DMA interrupt is enabled.

41.6.8 DMA Interrupt Status Register

Name: DMAISR

Access Type: Read-only

Offset: 0x20

Reset Value: 0x00000000

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

- **DMAISR: DMA Interrupt Status**
  0: No DMA transfer or in-progress.
  1: DMA transfer is completed.
  This bit is set to zero when DMAISR is read.

41.6.9 Control Register

Name: CR

Access Type: Write-only

Offset: 0x34

Reset Value: -

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

- **RESET: Reset CRCComputation**
  Write a one to reset SR.
  Writing a zero has no effect.

41.6.10 Mode Register

Name: MR

Access Type: Read/Write

Offset: 0x38

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
DIVIDER				PTYPE		COMPARE	ENABLE

- **DIVIDER: Bandwidth Divider**
  DMA bandwidth, required for CRC computation, is divided by 2^(DIVIDER+1).
- **PTYPE: Polynomial Type**

Value	Name	Description
0	CCIT8023	Polynom 0x04C11DB7
1	CASTAGNOLI	Polynom 0x1EDC6F41
2	CCIT16	Polynom 0x1021

- **COMPARE: CRC Compare**
  0: No comparison.
  1: CRC computed is compared with stored value.
- **ENABLE: CRC Computation Enable**
  0: No computation.
  1: CRC computation enabled.

41.6.11 Status Register

Name: SR

Access Type: Read-only

Offset: 0x3C

Reset Value: 0xFFFFFFFF

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

- **CRC: Cyclic Redundancy Check Value**
  CRC computation result.
  If MR.COMPARE=1, SR is not readable.

41.6.12 Interrupt Enable Register

Name: IER

Access Type: Write-only

Offset: 0x40

Reset Value: –

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ERRIER

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will set the corresponding bit in IMR

41.6.13 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x44

Reset Value: –

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

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR

41.6.14 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x48

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ERRIMR

0: The corresponding interrupt is disabled.
1: The corresponding interrupt is enabled.
This bit is cleared when the corresponding bit in IDR is written to one.
This bit is set when the corresponding bit in IER is written to one.

41.6.15 Interrupt Status Register

Name: ISR

Access Type: Read-only

Offset: 0x4C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ERRISR

- **ERRISR: CRC Error Interrupt Status**
  0: No CRC error
  1: CRC error, CRC computed and CRC stored are different.
  ERRISR is cleared when this register is read.

41.6.16 Version Register

Name: VERSION

Access Type: Read-only

Offset: 0xFC

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	VARIANT		

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

41.6.17 Transfer Address Register

Name: ADDR

Access Type: Read/Write

Offset: DSCR

Reset Value: -

31	30	29	28	27	26	25	24
ADDR[31:24]							

23	22	21	20	19	18	17	16
ADDR[23:16]							

15	14	13	12	11	10	9	8
ADDR[15:8]							

7	6	5	4	3	2	1	0
ADDR[7:0]							

- **ADDR: Transfer Address**
  Address of memory block to compute.

41.6.18 Transfer Control Register

Name: CTRL

Access Type: Read/Write

Offset: DSCR + 0x04

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	IEN	-	TRWIDTH	

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
BTSIZE[15:8]							

7	6	5	4	3	2	1	0
BTSIZE[7:0]							

- **IEN: Interrupt Mask Enable**
  0: Interrupt is enabled
  1: Interrupt is masked
- **TRWIDTH: Transfer Width**

TRWIDTH	Size
00	Byte
01	Halfword
10	Word

- **BTSIZE: Buffer Transfer Size**
  Buffer Transfert Size

41.6.19 Transfer Reference Register

Name: CRC

Access Type: Read/Write

Offset: DSCR + 0x10

Reset Value: -

31	30	29	28	27	26	25	24
REFCRC[31:24]							

23	22	21	20	19	18	17	16
REFCRC[23:16]							

15	14	13	12	11	10	9	8
REFCRC[15:8]							

7	6	5	4	3	2	1	0
REFCRC[7:0]							

- **REFCRC: Reference CRC**
  When compare mode is enabled, checksum is compared with this register.

41.7 Module Configuration

The specific configuration for each CRCCU instance is listed in the following tables.The module bus clocks listed here are connected to the system bus clocks. Refer to Section 10. “Power Manager (PM)” on page 109 for details.

Table 41-3. CRCCU Clock Name

Clock Name	Description
CLK_CRCCU_APB	Clock for the CRCCU bus interface
CLK_CRCCU_AHB	Clock for the CRCCU AHB interface

Register	Reset Value
VERSION	0x00000202

42. Electrical Characteristics

42.1 Absolute Maximum Ratings*

Table 42-1. Absolute Maximum Ratings

Operating temperature ..... -40°C to +85°C
Storage temperature..... -60°C to +150°C
Voltage on input pins with respect to ground .....-0.3V to V_{VDD} (1)+0.3V
Total DC output current on all I/O pins VDDIO ..... 120 mA
Total DC output current on all I/O pins VDDIN ..... 100 mA
Total DC output current on all I/O pins VDDANA..... 50 mA
Maximum operating voltage VDDIO, VDDIN..... 3.6V

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details

42.2 Operating Conditions

All the electrical characteristics are applicable to the following conditions unless otherwise specified :

- operating voltage range 1,68V to 3,6V for VDDIN, VDDIO & VDDANA
- Power Scaling 0 and 2 modes
- operating temperature range: TA = -40°C to 85°C and for a junction temperature up to TJ = 100°C.

Typical values are base on TA = 25°c and VDDIN,VDDIO,VDDANA = 3,3V unless otherwise specified

42.3 Supply Characteristics

Table 42-2. Supply Characteristics

Symbol	Conditions	Voltage		
		Min	Max	Unit
V_{VDDIO}, V_{VDDIN}, V_{VDDANA}	PS1 (FCPU<=12MHz) Linear mode	1.68	3.6	V
	PS0 & PS2 (FCPU>12MHz) Linear mode	1.8		
	Switching mode	2.0 (1)		

1. Below 2.3V, linear mode is more power efficient than switching mode.

Refer to Section 5. ”Power and Startup Considerations” on page 43 for details about Power Supply

Table 42-3. Supply Rise Rates and Order [1]

VDDIO, VDDIN and VDDANA must be connected together and as a consequence, rise synchronously

Symbol	Parameter	Rise Rate			
		Min	Max	Unit	Comment
V_{VDDIO}	DC supply peripheral I/Os	0.0001	2.5	V/µs	
V_{VDDIN}	DC supply peripheral I/Os and internal regulator	0.0001	2.5	V/µs	
V_{VDDANA}	Analog supply voltage	0.0001	2.5	V/µs	

1. These values are based on characterization. These values are not covered by test limits in production.

42.4 Maximum Clock Frequencies

Table 42-4. Maximum Clock Frequencies in Power Scaling Mode 0/2 and RUN Mode

Symbol	Parameter	Description	Max	Units
f_{CPU}	CPU clock frequency		48	MHz
f_{PBA}	PBA clock frequency		48	
f_{PBB}	PBB clock frequency		48	
f_{PBC}	PBC clock frequency		48	
f_{PBD}	PBD clock frequency		48	
f_{GCLK0}	GCLK0 clock frequency	DFLLIF main reference, GCLK0 pin	50	
f_{GCLK1}	GCLK1 clock frequency	DFLLIF dithering and SSG reference, GCLK1 pin	50	
f_{GCLK2}	GCLK2 clock frequency	AST, GCLK2 pin	20	
f_{GCLK3}	GCLK3 clock frequency	CATB, GCLK3 pin	50	
f_{GCLK4}	GCLK4 clock frequency	FLO and AESA	50	
f_{GCLK5}	GCLK5 clock frequency	GLOC, TC0 and RC32KIFB_REF	80	
f_{GCLK6}	GCLK6 clock frequency	ABDACB and IISC	50	
f_{GCLK7}	GCLK7 clock frequency	USBC	50	
f_{GCLK8}	GCLK8 clock frequency	TC1 and PEVC[0]	50	
f_{GCLK9}	GCLK9 clock frequency	PLL0 and PEVC[1]	50	
f_{GCLK10}	GCLK10 clock frequency	ADCIFE	50	
f_{GCLK11}	GCLK11 clock frequency	Master generic clock. Can be used as source for other generic clocks	150	
f_{OSC0}	OSC0 output frequency	Oscillator 0 in crystal mode	30	
		Oscillator 0 in digital clock mode	50	
f_{PLL}	PLL output frequency	Phase Locked Loop	240	
f_{DFLL}	DFLL output frequency	Digital Frequency Locked Loop	220	
f_{RC80M}	RC80M output frequency	Internal 80MHz RC Oscillator	80	

Table 42-5. Maximum Clock Frequencies in Power Scaling Mode 1 and RUN Mode

Symbol	Parameter	Description	Max	Units
f_{CPU}	CPU clock frequency		12	MHz
f_{PBA}	PBA clock frequency		12	
f_{PBB}	PBB clock frequency		12	
f_{PBC}	PBC clock frequency		12	
f_{PBD}	PBD clock frequency		12	
f_{GCLK0}	GCLK0 clock frequency	DFLLIF main reference, GCLK0 pin	16.6	
f_{GCLK1}	GCLK1 clock frequency	DFLLIF dithering and SSGreference, GCLK1 pin	16.6	
f_{GCLK2}	GCLK2 clock frequency	AST, GCLK2 pin	6.6	
f_{GCLK3}	GCLK3 clock frequency	CATB, GCLK3 pin	17.3	
f_{GCLK4}	GCLK4 clock frequency	FLO and AESA	16.6	
f_{GCLK5}	GCLK5 clock frequency	GLOC, TC0 and RC32KIFB_REF	26.6	
f_{GCLK6}	GCLK6 clock frequency	ABDACB and IISC	16.6	
f_{GCLK7}	GCLK7 clock frequency	USBC	16.6	
f_{GCLK8}	GCLK8 clock frequency	TC1 and PEVC[0]	16.6	
f_{GCLK9}	GCLK9 clock frequency	PLL0 and PEVC[1]	16.6	
f_{GCLK10}	GCLK10 clock frequency	ADCIFE	16.6	
f_{GCLK11}	GCLK11 clock frequency	Master generic clock. Can be used as source for other generic clocks	51.2	
f_{OSC0}	OSC0 output frequency	Oscillator 0 in crystal mode	16	
		Oscillator 0 in digital clock mode	16	
f_{PLL}	PLL output frequency	Phase Locked Loop	N/A	
f_{DFLL}	DFLL output frequency	Digital Frequency Locked Loop	N/A	
f_{RC80M}	RC80M output frequency	Internal 80MHz RC Oscillator	N/A	

42.5 Power Consumption

42.5.1 Power Scaling 0 and 2

The values in Table 42-6 are measured values of power consumption under the following conditions, except where noted:

- Operating conditions for power scaling mode 0 and 2
  - V_{VDDIN} = 3.3V
  - Power Scaling mode 0 is used for CPU frequencies under 36MHz
  - Power Scaling mode 2 is used for CPU frequencies above 36MHz
- Wake up time from low power modes is measured from the edge of the wakeup signal to the first instruction fetched in flash.
- Oscillators
  - OSC0 (crystal oscillator) stopped
  - OSC32K (32kHz crystal oscillator) running with external 32kHz crystal
  - DFLL using OSC32K as reference and running at 48MHz
- Clocks
  - DFLL used as main clock source
  - CPU, AHB clocks undivided
  - APBC and APBD clocks divided by 4
  - APBA and APBB bridges off
  - The following peripheral clocks running
    - PM, SCIF, AST, FLASHCALW, APBC and APBD bridges
  - All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU is running on flash with 1 wait state
- Low power cache enabled
- BOD18 and BOD33 disabled

Table 42-6. ATSAM4L4/2 Current consumption and Wakeup time for power scaling mode 0 and 2

Mode	Conditions	T_A	Typical Wakeup Time	Typ	Max (1)	Unit
RUN	CPU running a Fibonacci algorithm Linear mode	25°C	N/A	296	326	µA/MHz
		85°C		300	332	
	CPU running a CoreMark algorithm Linear mode	25°C	N/A	320	377	
		85°C		326	380	
	CPU running a Fibonacci algorithm Switching mode	25°C	N/A	177	198	
		85°C		179	200	
	CPU running a CoreMark algorithm Switching mode	25°C	N/A	186	232	
		85°C		195	239	

Table 42-6. ATSAM4L4/2 Current consumption and Wakeup time for power scaling mode 0 and 2

Mode	Conditions	T_A	Typical Wakeup Time	Typ	Max (1)	Unit
SLEEP0	Switching mode	25°C	9 * Main clock cycles	3817	4033	µA
		85°C		3934	4174	
SLEEP1	Switching mode	25°C	9 * Main clock cycles + 500ns	2341	2477	
		85°C		2437	2585	
SLEEP2	Switching mode	25°C	9 * Main clock cycles + 500ns	1758	1862	
		85°C		1847	1971	
SLEEP3	Linear mode	25°C		51	60	
WAIT	OSC32K and AST running Fast wake-up enable		1.5µs	5.9	8.7	
	OSC32K and AST stopped Fast wake-up enable			4.7	7.6	
RETENTION	OSC32K running AST running at 1kHz		1.5µs	3.1	5.1	
	AST and OSC32K stopped			2.2	4.2	
BACKUP	OSC32K running AST running at 1kHz			1.5	3.1	
	AST and OSC32K stopped			0.9	1.7	

1. These values are based on characterization. These values are not covered by test limits in production.

Table 42-7. ATSAM4L8 Current consumption and Wakeup time for power scaling mode 0 and 2

Mode	Conditions	T_A	Typical Wakeup Time	Typ	Max (1)	Unit
RUN	CPU running a Fibonacci algorithm Linear mode	25°C	N/A	319	343	µA/MHz
		85°C		326	350	
	CPU running a CoreMark algorithm Linear mode	25°C	N/A	343	387	
		85°C		351	416	
	CPU running a Fibonacci algorithm Switching mode	25°C	N/A	181	198	
		85°C		186	203	
	CPU running a CoreMark algorithm Switching mode	25°C	N/A	192	232	
		85°C		202	239	

Table 42-7. ATSAM4L8 Current consumption and Wakeup time for power scaling mode 0 and 2

Mode	Conditions	T_A	Typical Wakeup Time	Typ	Max (1)	Unit
SLEEP0	Switching mode	25°C	9 * Main clock cycles	3817	4033	µA
		85°C		4050	4507	
SLEEP1	Switching mode	25°C	9 * Main clock cycles + 500ns	2341	2477	
		85°C		2525	2832	
SLEEP2	Switching mode	25°C	9 * Main clock cycles + 500ns	1758	1862	
		85°C		1925	1971	
SLEEP3	Linear mode	25°C		51	60	
WAIT	OSC32K and AST running Fast wake-up enable		1.5µs	6.7		
	OSC32K and AST stopped Fast wake-up enable			5.5		
RETENTION	OSC32K running AST running at 1kHz		1.5µs	3.9		
	AST and OSC32K stopped			3.0		
BACKUP	OSC32K running AST running at 1kHz			1.5	3.1	
	AST and OSC32K stopped			0.9	1.7	

1. These values are based on characterization. These values are not covered by test limits in production.

42.5.2 Power Scaling 1

The values in Table 34-7 are measured values of power consumption under the following conditions, except where noted:

- Operating conditions for power scaling mode 1
  - V_{VDDIN} = 3.3V
- Wake up time from low power modes is measured from the edge of the wakeup signal to the first instruction fetched in flash.
- Oscillators
  - OSC0 (crystal oscillator) and OSC32K (32kHz crystal oscillator) stopped
  - RCFAST Running at 12MHz
- Clocks
  - RCFAST used as main clock source
  - CPU, AHB clocks undivided
  - APBC and APBD clocks divided by 4
  - APBA and APBB bridges off
  - The following peripheral clocks running
    - PM, SCIF, AST, FLASHCALW, APBC and APBD bridges

– All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU is running on flash with 1 wait state
- Low power cache enabled
- BOD18 and BOD33 disabled

Table 42-8. ATSAM4L4/2 Current consumption and Wakeup time for power scaling mode 1

Mode	Conditions	T_A	Typical Wakeup Time	Typ	Max (1)	Unit
RUN	CPU running a Fibonacci algorithm Linear mode	25°C	N/A	205	224	µA/MHz
		85°C		212	231	
	CPU running a CoreMark algorithm Linear mode	25°C	N/A	213	244	
		85°C		230	270	
	CPU running a Fibonacci algorithm Switching mode	25°C	N/A	95	112	
		85°C		100	119	
	CPU running a CoreMark algorithm Switching mode	25°C	N/A	100	128	
		85°C		107	138	
SLEEP0	Switching mode	25°C	9 * Main clock cycles	527	627	µA
		85°C		579	739	
SLEEP1	Switching mode	25°C	9 * Main clock cycles + 500ns	369	445	
		85°C		404	564	
SLEEP2	Switching mode	25°C	9 * Main clock cycles + 500ns	305	381	
		85°C		334	442	
SLEEP3	Linear mode	25°C		46	55	
WAIT	OSC32K and AST running Fast wake-up enable		1.5µs	4.7	7.5	
	OSC32K and AST stopped Fast wake-up enable			3.5	6.3	
RETENTION	OSC32K running AST running at 1kHz		1.5µs	2.6	4.8	
	AST and OSC32K stopped			1.5	4	
BACKUP	OSC32K running AST running at 1kHz			1.5	3.1	
	AST and OSC32K stopped			0.9	1.7	

1. These values are based on characterization. These values are not covered by test limits in production.

Table 42-9. ATSAM4L8 Current consumption and Wakeup time for power scaling mode 1

Mode	Conditions	T_A	Typical Wakeup Time	Typ	Max (1)	Unit
RUN	CPU running a Fibonacci algorithm Linear mode	25°C	N/A	222	240	µA/MHz
		85°C		233	276	
	CPU running a CoreMark algorithm Linear mode	25°C	N/A	233	276	
		85°C		230	270	
	CPU running a Fibonacci algorithm Switching mode	25°C	N/A	100	112	
		85°C		100	119	
	CPU running a CoreMark algorithm Switching mode	25°C	N/A	104	128	
		85°C		107	138	
SLEEP0	Switching mode	25°C	9 * Main clock cycles	527	627	µA
		85°C		579	739	
SLEEP1	Switching mode	25°C	9 * Main clock cycles + 500ns	369	445	
		85°C		404	564	
SLEEP2	Switching mode	25°C	9 * Main clock cycles + 500ns	305	381	
		85°C		334	442	
SLEEP3	Linear mode	25°C		46	55	
WAIT	OSC32K and AST running Fast wake-up enable		1.5µs	5.5		
	OSC32K and AST stopped Fast wake-up enable			4.3		
RETENTION	OSC32K running AST running at 1kHz		1.5µs	3.4		
	AST and OSC32K stopped			2.3		
BACKUP	OSC32K running AST running at 1kHz			1.5	3.1	
	AST and OSC32K stopped			0.9	1.7	

1. These values are based on characterization. These values are not covered by test limits in production.

Table 42-10. Typical Power Consumption running CoreMark on CPU clock sources (1)

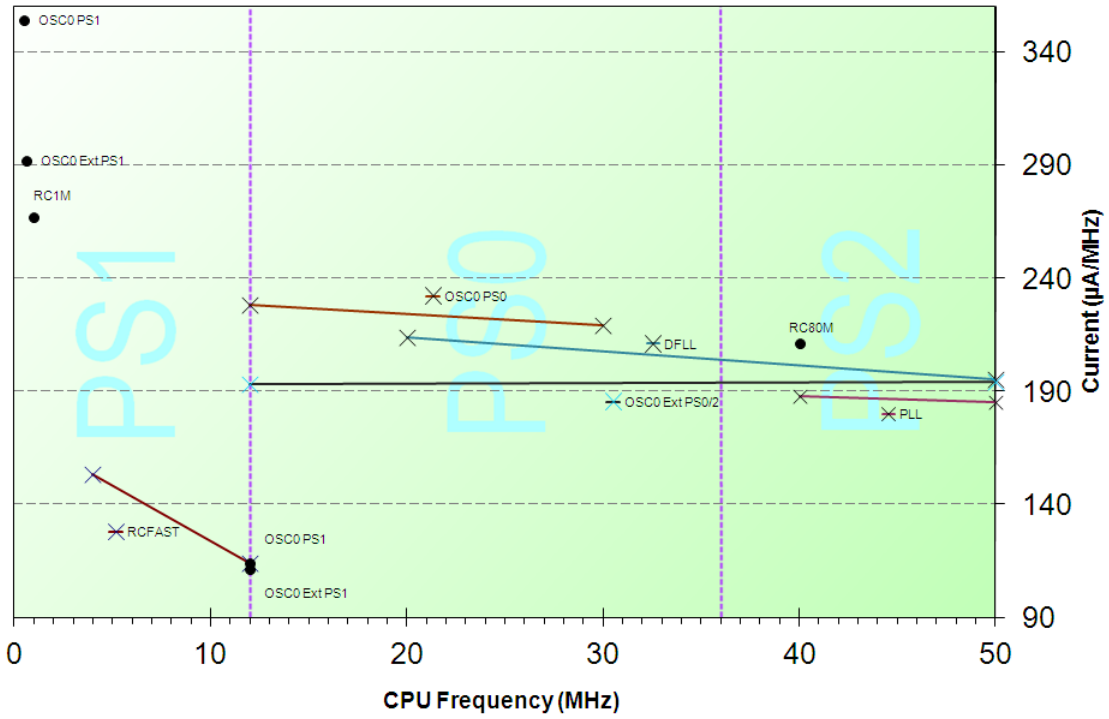
Clock Source	Conditions	Regulator	Frequency (MHz)	Typ	Unit

Table 42-10. Typical Power Consumption running CoreMark on CPU clock sources [1]

RCSYS (MCSEL = 0)	Power scaling mode 1	Switching Mode	0.115	978	µA/MHz
OSC0 (MCSEL = 1)	Power scaling mode 1		0.5	354	
			12	114	
	Power scaling mode 0		12	228	
			30	219	
OSC0 (MCSEL = 1) External Clock (MODE=0)	Power scaling mode 1		0.6	292	
			12	111	
	Power scaling mode 0		12	193	
	Power scaling mode 2		50	194	
PLL (MCSEL = 2)	Power scaling mode 2 Input Freq = 4MHz from OSC0		40	188	
			50	185	
DFLL (MCSEL = 3)	Power scaling mode 0 Input Freq = 32kHz from OSC32K		20	214	
	Power scaling mode 2 Input Freq = 32kHz from OSC32K		50	195	
RC1M (MCSEL = 4)	Power scaling mode 1		1	267	
RCFAST (MCSEL = 5)	Power scaling mode 1 RCFAST frequency is configurable from 4 to 12MHz		4	153	
			12	114	
RC80M (MCSEL = 6)	Power scaling mode 2 f_{CPU} = RC80M / 2 = 40MHz		40	211	

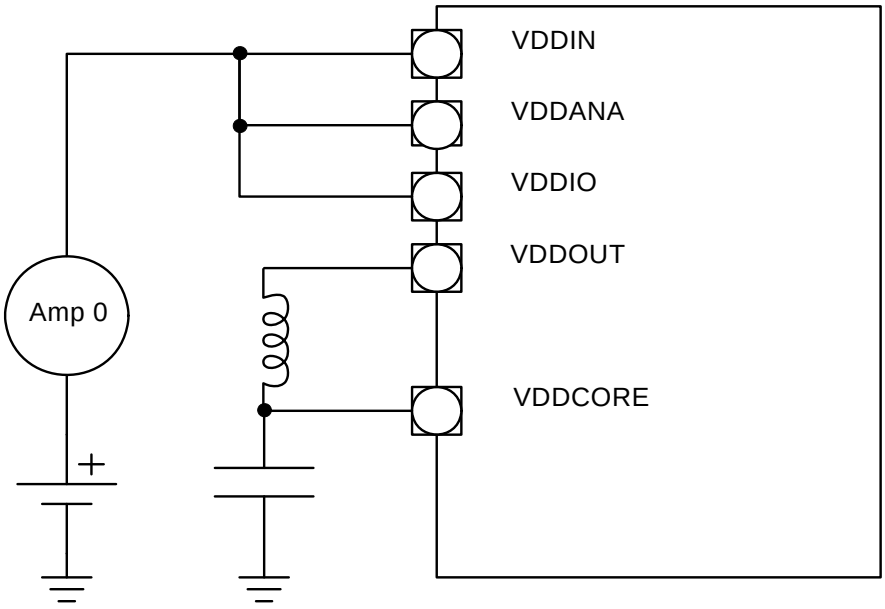
1. These values are based on characterization. These values are not covered by test limits in production.

Figure 42-1. Typical Power Consumption running Coremark (from above table)



Note: For variable frequency oscillators, linear interpolation between high and low settings

Figure 42-2. Measurement Schematic, Switching Mode



42.5.3 Peripheral Power Consumption in Power Scaling mode 0 and 2

The values in Table 42-11 are measured values of power consumption under the following conditions:

- Operating conditions, internal core supply (Figure 42-2)
  - V_{VDDIN} = 3.3V
  - $V_{VDDCORE}$ supplied by the internal regulator in switching mode
- T_A = 25°C
- Oscillators
  - OSC0 (crystal oscillator) stopped
  - OSC32K (32KHz crystal oscillator) running with external 32KHz crystal
  - DFLL running at 48MHz with OSC32K as reference clock
- Clocks
  - DFLL used as main clock source
  - CPU, AHB, and PB clocks undivided
- I/Os are inactive with internal pull-up
- Flash enabled in high speed mode
- CPU in SLEEP0 mode
- BOD18 and BOD33 disabled

Consumption active is the added current consumption when the module clock is turned on.

Table 42-11. Typical Current Consumption by Peripheral in Power Scaling Mode 0 and 2 (1)

Peripheral	Typ Consumption Active	Unit
IISC	1.0	µA/MHz
SPI	1.9	
TC	6.3	
TWIM	1.5	
TWIS	1.2	
USART	8.5	
ADCIFE(2)	3.1	
DACC	1.3	
ACIFC (2)	3.1	
GLOC	0.4	
ABDACB	0.7	
TRNG	0.9	
PARC	0.7	
CATB	3.0	
LCDCA	4.4	
PDCA	1.0	
CRCCU	0.3	
USBC	1.5	
PEVC	5.6	
CHIPID	0.1	
SCIF	6.4	
FREQM	0.5	
GPIO	7.1	
BPM	0.9	
BSCIF	4.6	
AST	1.5	
WDT	1.4	
EIC	0.6	
PICOUART	0.3	

1. These numbers are valid for the measured condition only and must not be extrapolated to other frequencies
2. Includes the current consumption on VDDANA and ADVREFP.

42.5.4 .Peripheral Power Consumption in Power Scaling mode 1

The values in Table 42-13 are measured values of power consumption under the following conditions:

- Operating conditions, internal core supply (Figure 42-2)
  - V_{VDDIN} = 3.3V
  - $V_{VDDCORE}$ = 1.2 V, supplied by the internal regulator in switching mode
- T_A = 25°C
- Oscillators
  - OSC0 (crystal oscillator) stopped
  - OSC32K (32KHz crystal oscillator) running with external 32KHz crystal
  - RCFAST running @ 12MHz
- Clocks
  - RCFAST used as main clock source
  - CPU, AHB, and PB clocks undivided
- I/Os are inactive with internal pull-up
- Flash enabled in normal mode
- CPU in SLEEP0 mode
- BOD18 and BOD33 disabled

Consumption active is the added current consumption when the module clock is turned on

Table 42-12. Typical Current Consumption by Peripheral in Power Scaling Mode 1 (1)

Peripheral	Typ Consumption Active	Unit
IISC	0.5	µA/MHz
SPI	1.1	
TC	3.1	
TWIM	0.8	
TWIS	0.7	
USART	4.4	
ADCIFE(2)	1.6	
DACC	0.6	
ACIFC (2)	1.6	
GLOC	0.1	
ABDACB	0.3	
TRNG	0.3	
PARC	0.3	
CATB	1.5	
LCDCA	2.2	
PDCA	0.4	
CRCCU	0.3	
USBC	0.9	
PEVC	2.8	
CHIPID	0.1	
SCIF	3.1	
FREQM	0.2	
GPIO	3.4	
BPM	0.4	
BSCIF	2.3	
AST	0.8	
WDT	0.8	
EIC	0.3	
PICOUART	0.2	

1. These numbers are valid for the measured condition only and must not be extrapolated to other frequencies
2. Includes the current consumption on VDDANA and ADVREFP.

42.6 I/O Pin Characteristics

42.6.1 Normal I/O Pin

Table 42-13. Normal I/O Pin Characteristics (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)				40		kΩ
$R_{PULLDOWN}$	Pull-down resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.2 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.8 * V_{VDD}		V_{VDD} + 0.3	V
V_{OL}	Output low-level voltage					0.4	V
V_{OH}	Output high-level voltage			V_{VDD} - 0.4			V
I_{OL}	Output low-level current (3)	ODCR0=0	1.68V<V_{VDD}<2.7V			0.8	mA
			2.7V<V_{VDD}<3.6V			1.6	mA
		ODCR0=1	1.68V<V_{VDD}<2.7V			1.6	mA
			2.7V<V_{VDD}<3.6V			3.2	mA
I_{OH}	Output high-level current(3)	ODCR0=0	1.68V<V_{VDD}<2.7V			0.8	mA
			2.7V<V_{VDD}<3.6V			1.6	mA
		ODCR0=1	1.68V<V_{VDD}<2.7V			1.6	mA
			2.7V<V_{VDD}<3.6V			3.2	mA
t_{RISE}	Rise time(2)	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, load = 25pF			35	ns
		OSRR0=1				45	ns
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, load = 25pF			19	ns
		OSRR0=1				23	ns
t_{FALL}	Fall time(2)	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, load = 25pF			36	ns
		OSRR0=1				47	ns
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, load = 25pF			20	ns
		OSRR0=1				24	ns
F_{PINMAX}	Output frequency(2)	OSRR0=0	ODCR0=0, V_{VDD}>2.7V load = 25pF			17	MHz
		OSRR0=1				15	MHz
		OSRR0=0	ODCR0=1, V_{VDD}>2.7V load = 25pF			27	MHz
		OSRR0=1				23	MHz
I_{LEAK}	Input leakage current(3)		Pull-up resistors disabled		0.01	1	µA
C_{IN}	Input capacitance(2)				5		pF

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization

3. These values are based on characterization. These values are not covered by test limits in production

42.6.2 High-drive I/O Pin : PA02, PC04, PC05, PC06

Table 42-14. High-drive I/O Pin Characteristics (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)				40		kΩ
$R_{PULLDOWN}$	Pull-down resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.2 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.8 * V_{VDD}		V_{VDD} + 0.3	
V_{OL}	Output low-level voltage					0.4	
V_{OH}	Output high-level voltage			V_{VDD} - 0.4			
I_{OL}	Output low-level current (3)	ODCR0=0	1.68V<V_{VDD}<2.7V			1.8	mA
			2.7V<V_{VDD}<3.6V			3.2	
		ODCR0=1	1.68V<V_{VDD}<2.7V			3.2	mA
			2.7V<V_{VDD}<3.6V			6	
I_{OH}	Output high-level current(3)	ODCR0=0	1.68V<V_{VDD}<2.7V			1.6	mA
			2.7V<V_{VDD}<3.6V			3.2	
		ODCR0=1	1.68V<V_{VDD}<2.7V			3.2	mA
			2.7V<V_{VDD}<3.6V			6	
t_{RISE}	Rise time(2)	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, Cload = 25pF			20	ns
		OSRR0=1				40	
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, Cload = 25pF			11	ns
		OSRR0=1				18	
t_{FALL}	Fall time(2)	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, Cload = 25pF			20	ns
		OSRR0=1				40	
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, Cload = 25pF			11	ns
		OSRR0=1				18	
F_{PINMAX}	Output frequency(2)	OSRR0=0	ODCR0=0, V_{VDD}>2.7V load = 25pF			22	MHz
		OSRR0=1				17	MHz
		OSRR0=0	ODCR0=1, V_{VDD}>2.7V load = 25pF			35	MHz
		OSRR0=1				26	MHz
I_{LEAK}	Input leakage current(3)	Pull-up resistors disabled			0.01	2	µA
C_{IN}	Input capacitance(2)				10		pF

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization
3. These values are based on characterization. These values are not covered by test limits in production

42.6.3 USB I/O Pin : PA25, PA26

Table 42-15. USB I/O Pin Characteristics in GPIO configuration (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)				40		kΩ
$R_{PULLDOWN}$	Pull-down resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.2 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.8 * V_{VDD}		V_{VDD} + 0.3	
V_{OL}	Output low-level voltage					0.4	
V_{OH}	Output high-level voltage			V_{VDD} - 0.4			
I_{OL}	Output low-level current (3)	ODCR0=0	1.68V<V_{VDD}<2.7V		20		mA
			2.7V<V_{VDD}<3.6V		30		
I_{OH}	Output high-level current(3)	ODCR0=0	1.68V<V_{VDD}<2.7V		20		mA
			2.7V<V_{VDD}<3.6V		30		
F_{PINMAX}	Maximum frequency(2)	ODCR0=0 OSRR0=0	load = 25pF			20	MHz
I_{LEAK}	Input leakage current(3)	Pull-up resistors disabled			0.01	1	µA
C_{IN}	Input capacitance(2)				5		pF

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization
3. These values are based on characterization. These values are not covered by test limits in production

42.6.4 TWI Pin : PA21, PA22, PA23, PA24, PB14, PB15

Table 42-16. TWI Pin Characteristics in TWI configuration (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)				40		kΩ
$R_{PULLDOWN}$	Pull-down resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.3 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.7 * V_{VDD}		V_{VDD} + 0.3	V
V_{OL}	Output low-level voltage					0.4	V
I_{OL}	Output low-level current (3)	DRIVEL=0				0.5	mA
		DRIVEL=1				1.0	
		DRIVEL=2				1.6	
		DRIVEL=3				3.1	
		DRIVEL=4				6.2	
		DRIVEL=5				9.3	
		DRIVEL=6				15.5	
		DRIVEL=7				21.8	

Table 42-16. TWI Pin Characteristics in TWI configuration (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
I_{CS}	Current Source(3)	DRIVEH=0			0.5		mA
		DRIVEH=1			1		
		DRIVEH=2			1.5		
		DRIVEH=3			3		
f_{MAX}	Max frequency(2)	HsMode with Current source; DRIVEx=3, SLEW=0 Cbus = 400pF, V_{VDD} = 1.68V		3.5	6.4		MHz
t_{RISE}	Rise time(2)	HsMode Mode, DRIVEx=3, SLEW=0 Cbus = 400pF, Rp = 440Ohm, V_{VDD} = 1.68V			28	38	ns
t_{FALL}	Fall time(2)	Standard Mode, DRIVEx=3, SLEW=0 Cbus = 400pF, Rp = 440Ohm, V_{VDD} = 1.68V			50	95	ns
		HsMode Mode, DRIVEx=3, SLEW=0 Cbus = 400pF, Rp = 440Ohm, V_{VDD} = 1.68V			50	95	

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization
3. These values are based on characterization. These values are not covered by test limits in production

Table 42-17. TWI Pin Characteristics in GPIO configuration (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)				40		kΩ
$R_{PULLDOWN}$	Pull-up resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.2 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.8 * V_{VDD}		V_{VDD} + 0.3	V
V_{OL}	Output low-level voltage					0.4	V
V_{OH}	Output high-level voltage			V_{VDD} - 0.4			
I_{OL}	Output low-level current (3)	ODCR0=0	1.68V<V_{VDD}<2.7V			1.8	mA
			2.7V<V_{VDD}<3.6V			3.5	
		ODCR0=1	1.68V<V_{VDD}<2.7V			3.6	
			2.7V<V_{VDD}<3.6V			6.8	
I_{OH}	Output high-level current(3)	ODCR0=0	1.68V<V_{VDD}<2.7V			1.8	mA
			2.7V<V_{VDD}<3.6V			3.5	
		ODCR0=1	1.68V<V_{VDD}<2.7V			3.6	
			2.7V<V_{VDD}<3.6V			6.8	

Table 42-17. TWI Pin Characteristics in GPIO configuration [1]

Symbol	Parameter	Conditions		Min	Typ	Max	Units
t_{RISE}	Rise time[2]	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, Cload = 25pF		18		ns
		OSRR0=1			110		
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, Cload = 25pF		10		ns
		OSRR0=1			50		
t_{FALL}	Fall time[2]	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, Cload = 25pF		19		ns
		OSRR0=1			140		
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, Cload = 25pF		12		ns
		OSRR0=1			63		

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization
3. These values are based on characterization. These values are not covered by test limits in production

Table 42-18. Common TWI Pin Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_{LEAK}	Input leakage current [1]	Pull-up resistors disabled		0.01	1	µA
C_{IN}	Input capacitance[2]			5		pF

1. These values are based on simulation. These values are not covered by test limits in production or characterization

42.6.5 High Drive TWI Pin : PB00, PB01

Table 42-19. High Drive TWI Pin Characteristics in TWI configuration (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)	PB00, PB01			40		kΩ
$R_{PULLDOWN}$	Pull-down resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.3 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.7 * V_{VDD}		V_{VDD} + 0.3	
V_{OL}	Output low-level voltage					0.4	
V_{OH}	Output high-level voltage			V_{VDD} - 0.4			
I_{OL}	Output low-level current (3)	DRIVEL=0				0.5	mA
		DRIVEL=1				1.0	
		DRIVEL=2				1.6	
		DRIVEL=3				3.1	
		DRIVEL=4				6.2	
		DRIVEL=5				9.3	
		DRIVEL=6				15.5	
		DRIVEL=7				21.8	
I_{CS}	Current Source(2)	DRIVEH=0			0.5		mA
		DRIVEH=1			1		
		DRIVEH=2			1.5		
		DRIVEH=3			3		
f_{MAX}	Max frequency(2)	HsMode with Current source; DRIVEx=3, SLEW=0 Cbus = 400pF, V_{VDD} = 1.68V		3.5	6.4		MHz
t_{RISE}	Rise time(2)	HsMode Mode, DRIVEx=3, SLEW=0 Cbus = 400pF, Rp = 440Ohm, V_{VDD} = 1.68V			28	38	ns
t_{FALL}	Fall time(2)	Standard Mode, DRIVEx=3, SLEW=0 Cbus = 400pF, Rp = 440Ohm, V_{VDD} = 1.68V			50	95	ns
		HsMode Mode, DRIVEx=3, SLEW=0 Cbus = 400pF, Rp = 440Ohm, V_{VDD} = 1.68V			50	95	

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization
3. These values are based on characterization. These values are not covered by test limits in production

Table 42-20. High Drive TWI Pin Characteristics in GPIO configuration (1)

Symbol	Parameter	Conditions		Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance (2)				40		kΩ
$R_{PULLDOWN}$	Pull-up resistance(2)				40		kΩ
V_{IL}	Input low-level voltage			-0.3		0.2 * V_{VDD}	V
V_{IH}	Input high-level voltage			0.8 * V_{VDD}		V_{VDD} + 0.3	
V_{OL}	Output low-level voltage					0.4	
V_{OH}	Output high-level voltage			V_{VDD} - 0.4			
I_{OL}	Output low-level current (3)	ODCR0=0	1.68V<V_{VDD}<2.7V			3.4	mA
			2.7V<V_{VDD}<3.6V			6	
		ODCR0=1	1.68V<V_{VDD}<2.7V			5.2	mA
			2.7V<V_{VDD}<3.6V			8	
I_{OH}	Output high-level current(3)	ODCR0=0	1.68V<V_{VDD}<2.7V			3.4	mA
			2.7V<V_{VDD}<3.6V			6	
		ODCR0=1	1.68V<V_{VDD}<2.7V			5.2	mA
			2.7V<V_{VDD}<3.6V			8	
t_{RISE}	Rise time(2)	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, Cload = 25pF		18		ns
		OSRR0=1			110		
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, Cload = 25pF		10		ns
		OSRR0=1			50		
t_{FALL}	Fall time(2)	OSRR0=0	ODCR0=0 1.68V<V_{VDD}<2.7V, Cload = 25pF		19		ns
		OSRR0=1			140		
		OSRR0=0	ODCR0=0 2.7V<V_{VDD}<3.6V, Cload = 25pF		12		ns
		OSRR0=1			63		

1. V_{VDD} corresponds to either V_{VDDIN} or V_{VDDIO}, depending on the supply for the pin. Refer to Section 3-5 on page 13 for details
2. These values are based on simulation. These values are not covered by test limits in production or characterization
3. These values are based on characterization. These values are not covered by test limits in production

Table 42-21. Common High Drive TWI Pin Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_{LEAK}	Input leakage current (1)	Pull-up resistors disabled		0.01	2	µA
C_{IN}	Input capacitance(1)			10		pF

1. These values are based on simulation. These values are not covered by test limits in production or characterization

42.7 Oscillator Characteristics

42.7.1 Oscillator 0 (OSC0) Characteristics

42.7.1.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN.

Table 42-22. Digital Clock Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
f_{CPXIN}	XIN clock frequency (1)				50	MHz
t_{CPXIN}	XIN clock duty cycle(1)		40		60	%
$t_{STARTUP}$	Startup time			N/A		cycles

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.7.1.2 Crystal Oscillator Characteristics

The following table describes the characteristics for the oscillator when a crystal is connected between XIN and XOUT as shown in Figure 42-3. The user must choose a crystal oscillator where the crystal load capacitance C_L is within the range given in the table. The exact value of C_L can be found in the crystal datasheet. The capacitance of the external capacitors (C_{LEXT}) can then be computed as follows:

$$C_{LEXT} = 2(C_L - C_{STRAY} - C_{SHUNT})$$

where C_{STRAY} is the capacitance of the pins and PCB, C_{SHUNT} is the shunt capacitance of the crystal.

Table 42-23. Crystal Oscillator Characteristics

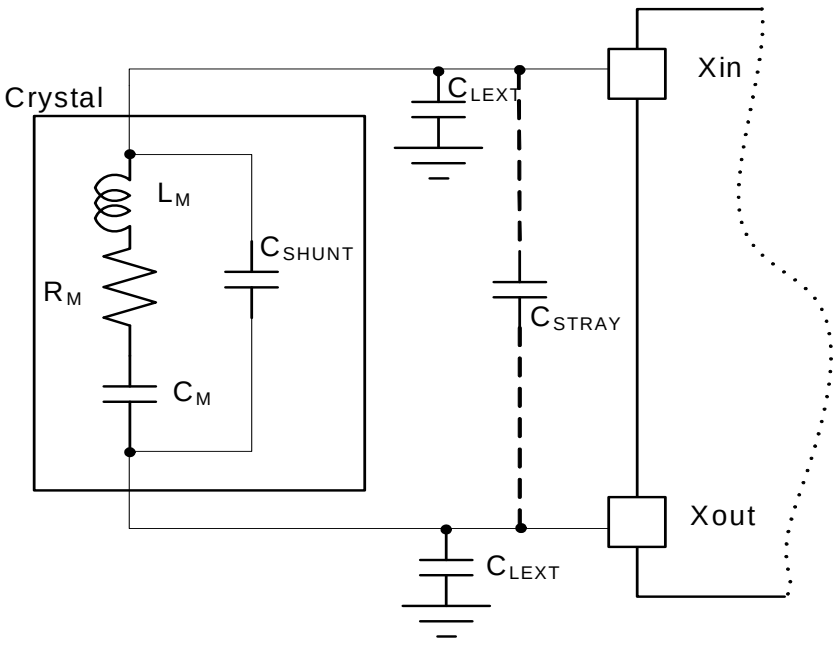
Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Crystal oscillator frequency (1)		0.6		30	MHz
ESR	Crystal Equivalent Series Resistance (2)	f = 0.455 MHz, C_{LEXT} = 100pF SCIF.OSCCTRL.GAIN = 0			17000	Ω
		f = 2MHz, C_{LEXT} = 20pF SCIF.OSCCTRL.GAIN = 0			2000	
		f = 4MHz, C_{LEXT} = 20pF SCIF.OSCCTRL.GAIN = 1			1500	
		f = 8 MHz, C_{LEXT} = 20pF SCIF.OSCCTRL.GAIN = 2			300	
		f = 16 MHz, C_{LEXT} = 20pF SCIF.OSCCTRL.GAIN = 3			350	
		f = 30MHz, C_{LEXT} = 18pF SCIF.OSCCTRL.GAIN = 4			45	

Table 42-23. Crystal Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
C_L	Crystal load capacitance(1)		6		18	pF
C_{SHUNT}	Crystal shunt capacitance(1)				7	
C_{XIN}	Parasitic capacitor load(2)	TQFP100 package		4.91		
C_{XOUT}	Parasitic capacitor load(2)			3.22		
$t_{STARTUP}$	Startup time(1)	SCIF.OSCCTRL.GAIN = 2		30000(3)		cycles
I_{OSC}	Current consumption(1)	Active mode, f = 0.6MHz, SCIF.OSCCTRL.GAIN = 0		30		µA
		Active mode, f = 4MHz, SCIF.OSCCTRL.GAIN = 1		130		
		Active mode, f = 8MHz, SCIF.OSCCTRL.GAIN = 2		260		
		Active mode, f = 16MHz, SCIF.OSCCTRL.GAIN = 3		590		
		Active mode, f = 30MHz, SCIF.OSCCTRL.GAIN = 4		960		

1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. These values are based on characterization. These values are not covered by test limits in production.
3. Nominal crystal cycles.

Figure 42-3. Oscillator Connection



42.7.2 32kHz Crystal Oscillator (OSC32K) Characteristics

Figure 42-3 and the equation above also applies to the 32kHz oscillator connection. The user must choose a crystal oscillator where the crystal load capacitance C_L is within the range given in the table. The exact value of C_L can then be found in the crystal datasheet.

Table 42-24. Digital Clock Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
$f_{CPXIN32}$	XIN32 clock frequency (1)				6	MHz
	XIN32 clock duty cycle(1)		40		60	%
$t_{STARTUP}$	Startup time			N/A		cycles

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

Table 42-25. 32 kHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
f_{OUT}	Crystal oscillator frequency				32 768		Hz
$t_{STARTUP}$	Startup time (1)	R_m = 100kΩ, C_L = 12.5pF			30000(2)		cycles
C_L	Crystal load capacitance(1)			6		12.5	pF
C_{SHUNT}	Crystal shunt capacitance(1)			0.8		1.7	
C_{XIN}	Parasitic capacitor load (3)	TQFP100 package			3.4		
C_{XOUT}	Parasitic capacitor load(3)				2.72		
I_{OSC32K}	Current consumption(1)				350		nA
ESR_{XTAL}	Crystal equivalent series resistance(1) f=32.768kHz OSCCTRL32.MODE=1 Safety Factor = 3	OSCCTRL32.SELCURR=0	C_L=6pF			28	kΩ
		OSCCTRL32.SELCURR=4				72	
		OSCCTRL32.SELCURR=8				114	
		OSCCTRL32.SELCURR=15				313	
		OSCCTRL32.SELCURR=0	C_L=9pF			14	kΩ
		OSCCTRL32.SELCURR=4				36	
		OSCCTRL32.SELCURR=8				100	
		OSCCTRL32.SELCURR=15				170	
	Crystal equivalent series resistance(3) f=32.768kHz OSCCTRL32.MODE=1 Safety Factor = 3	OSCCTRL32.SELCURR=4	C_L=12.5pF			15.2	kΩ
		OSCCTRL32.SELCURR=6				61.8	
		OSCCTRL32.SELCURR=8				101.8	
		OSCCTRL32.SELCURR=10				138.5	
		OSCCTRL32.SELCURR=15				228.5	

1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. Nominal crystal cycles.
3. These values are based on characterization. These values are not covered by test limits in production.

42.7.3 Phase Locked Loop (PLL) Characteristics

Table 42-26. Phase Locked Loop Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)	PLL is not availabe in PS1	48		240	MHz
f_{IN}	Input frequency(1)		4		16	
I_{PLL}	Current consumption(1)	fout=80MHz			200	µA
		fout=240MHz			500	
$t_{STARTUP}$	Startup time, from enabling the PLL until the PLL is locked(1)	Wide Bandwidth mode disabled			8	µs
		Wide Bandwidth mode enabled			30	

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.7.4 Digital Frequency Locked Loop (DFLL) Characteristics

Table 42-27. Digital Frequency Locked Loop Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)	DFLL is not availabe in PS1	20		150	MHz
f_{REF}	Reference frequency(1)		8		150	kHz
	Accuracy(1)	FINE lock, f_{REF} = 32kHz, SSG disabled (2)		0.1	0.5	%
		ACCURATE lock, f_{REF} = 32kHz, dither clk RCSYS/2, SSG disabled(2)		0.06	0.5	
		FINE lock, f_{REF} = 8-150kHz, SSG disabled(2)		0.2	1	
		ACCURATE lock, f_{REF} = 8-150kHz, dither clk RCSYS/2, SSG disabled(2)		0.1	1	
I_{DFLL}	Power consumption(1)	RANGE 0 96 to 220MHz COARSE=0, FINE=0, DIV=0	430	509	545	µA
		RANGE 0 96 to 220MHz COARSE=31, FINE=255, DIV=0	1545	1858	1919	
		RANGE 1 50 to 110MHz COARSE=0, FINE=0, DIV=0	218	271	308	
		RANGE 1 50 to 110MHz COARSE=31, FINE=255, DIV=0	704	827	862	
		RANGE 2 25 to 55MHz COARSE=0, FINE=0, DIV=1	140	187	226	
		RANGE 2 25 to 55MHz COARSE=31, FINE=255, DIV=1	365	441	477	
		RANGE 3 20 to 30MHz COARSE=0, FINE=0, DIV=1	122	174	219	
		RANGE 3 20 to 30MHz COARSE=31, FINE=255, DIV=1	288	354	391	

Table 42-27. Digital Frequency Locked Loop Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{STARTUP}$	Startup time(1)	Within 90% of final values			100	µs
t_{LOCK}	Lock time(1)	f_{REF} = 32kHz, FINE lock, SSG disabled(2)		600		
		f_{REF} = 32kHz, ACCURATE lock, dithering clock = RCSYS/2, SSG disabled(2)		1100		

1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. Spread Spectrum Generator (SSG) is disabled by writing a zero to the EN bit in the SCIF.DFLL0SSG register.

42.7.5 32kHz RC Oscillator (RC32K) Characteristics

Table 42-28. 32kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)	Calibrated against a 32.768kHz reference Temperature compensation disabled	20	32.768	44	kHz
I_{RC32K}	Current consumption (2)	Without temperature compensation		0.5		µA
		Temperature compensation enabled		2		µA
$t_{STARTUP}$	Startup time(1)			1		cycle

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.7.6 System RC Oscillator (RCSYS) Characteristics

Table 42-29. System RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)	Calibrated at 85°C	110	113.6	116	kHz
I_{RCSYS}	Current consumption (2)				12	µA
$t_{STARTUP}$	Startup time(1)		25	38	63	µs
Duty	Duty cycle(1)		49.6	50	50.3	%

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.7.7 1MHz RC Oscillator (RC1M) Characteristics

Table 42-30. RC1M Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)		0.91	1	1.12	MHz
I_{RC1M}	Current consumption (2)			35		µA
Duty	Duty cycle(1)		48.6	49.9	54.4	%

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.7.8 4/8/12MHz RC Oscillator (RCFAST) Characteristics

Table 42-31. RCFAST Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)	Calibrated, FRANGE=0	4	4.3	4.6	MHz
		Calibrated, FRANGE=1	7.8	8.2	8.5	
		Calibrated, FRANGE=2	11.3	12	12.3	
I_{RCFAST}	Current consumption (2)	Calibrated, FRANGE=0		90	110	µA
		Calibrated, FRANGE=1		130	150	
		Calibrated, FRANGE=2		180	205	
Duty	Duty cycle(1)	Calibrated, FRANGE=0	48.8	49.6	50.1	%
		Calibrated, FRANGE=1	47.8	49.2	50.1	
		Calibrated, FRANGE=2	46.7	48.8	50.0	
$t_{STARTUP}$	Startup time(1)	Calibrated, FRANGE=2	0.1	0.31	0.71	µs

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.7.9 80MHz RC Oscillator (RC80M) Characteristics

Table 42-32. Internal 80 MHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency (1)	After calibration Note that RC80M is not available in PS1	60	80	100	MHz
I_{RC80M}	Current consumption (2)			330		µA
$t_{STARTUP}$	Startup time(1)		0.57	1.72	3.2	µs
Duty	Duty cycle(2)		45	50	55	%

1. These values are based on characterization. These values are not covered by test limits in production.
2. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.8 Flash Characteristics

Table 42-33 gives the device maximum operating frequency depending on the number of flash wait states and the flash read mode. The FWS bit in the FLASHCALW FCR register controls the number of wait states used when accessing the flash memory.

Table 42-33. Maximum Operating Frequency (1)

PowerScaling Mode	Flash Read Mode	Flash Wait States	Maximum Operating Frequency	Unit
0	Low power (HSDIS) + Flash internal reference: BPM.PMCON.FASTWKUP=1	1	12	MHz
	Low power(HSDIS)	0	18	
		1	36	
1	Low power (HSDIS) + Flash internal reference: BPM.PMCON.FASTWKUP=1	1	12	
	Low power (HSDIS)	0	8	
		1	12	
2	High speed (HSEN)	0	24	
		1	48	

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

Table 42-34. Flash Characteristics (1)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
t_{FPP}	Page programming time	f_{CLK_AHB} = 48MHz		4.38		ms
t_{FPE}	Page erase time			4.38		
t_{FFP}	Fuse programming time			0.63		
t_{FEA}	Full chip erase time (EA)			5.66		
t_{FCE}	JTAG chip erase time (CHIP_ERASE)	f_{CLK_AHB} = 115kHz		304		

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

Table 42-35. Flash Endurance and Data Retention (1)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
N_{FARRAY}	Array endurance (write/page)	f_{CLK_AHB} > 10MHz	100k			cycles
N_{FFUSE}	General Purpose fuses endurance (write/bit)	f_{CLK_AHB} > 10MHz	10k			
t_{RET}	Data retention		15			years

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.9 Analog Characteristics

42.9.1 Voltage Regulator Characteristics

Table 42-36. VREG Electrical Characteristics in Linear and Switching Modes

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_{OUT}	DC output current (1) Power scaling mode 0 & 2	Low power mode (WAIT)	2000	3600	5600	µA
		Ultra Low power mode (RETENTION)	100	180	300	
	DC output current(1) Power scaling mode 1	Low power mode (WAIT)	4000	7000	10000	
		Ultra Low power mode (RETENTION)	200	350	600	
$V_{VDDCORE}$	DC output voltage	All modes			1.9	V

1. These values are based on simulation. These values are not covered by test limits in production.

Table 42-37. VREG Electrical Characteristics in Linear mode

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{VDDIN}	Input voltage range	I_{OUT}=10mA	1.68		3.6	V
		I_{OUT}=50mA	1.8		3.6	
$V_{VDDCORE}$	DC output voltage (1) Power scaling mode 0 & 2	I_{OUT} = 0 mA	1.777	1.814	1.854	
		I_{OUT} = 50 mA	1.75	1.79	1.83	
I_{OUT}	DC output current(1)	$V_{VDDCORE}$ > 1.65V			100	mA
	Output DC load regulation(1) Transient load regulation	I_{OUT} = 0 to 80mA, V_{VDDIN} = 3V	-34	-27	-19	mV
	Output DC regulation(1)	I_{OUT} = 80 mA, V_{VDDIN} = 2V to 3.6V	10	28	48	mV
I_Q	Quescient current(1)	I_{OUT} = 0 mA RUN and SLEEPx modes	88	107	128	µA

1. These values are based on characterization. These values are not covered by test limits in production.

Table 42-38. External components requirements in Linear Mode

Symbol	Parameter	Technology	Typ	Units
C_{IN1}	Input regulator capacitor 1		33	nF
C_{IN2}	Input regulator capacitor 2		100	
C_{IN3}	Input regulator capacitor 3		10	µF
C_{OUT1}	Output regulator capacitor 1		100	nF
C_{OUT2}	Output regulator capacitor 2	Tantalum or MLCC 0.5<ESR<10Ω	4.7	µF

Table 42-39. VREG Electrical Characteristics in Switching mode

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{VDDIN}	Input voltage range	$V_{VDDCORE}$ = 1.65V, I_{OUT}=50mA	2.0		3.6	V
$V_{VDDCORE}$	DC output voltage (1) Power scaling mode 0 & 2	I_{OUT} = 0 mA	1.75	1.82	1.87	
		I_{OUT} = 50 mA	1.66	1.71	1.79	

Table 42-39. VREG Electrical Characteristics in Switching mode

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_{OUT}	DC output current(1)	$V_{VDDCORE}$ > 1.65V			55	mA
	Output DC load regulation(1) Transient load regulation	I_{OUT} = 0 to 50mA, V_{VDDIN} = 3V	-136	-101	-82	mV
	Output DC regulation(1)	I_{OUT} = 50 mA, V_{VDDIN} = 2V to 3.6V	-20	38	99	mV
I_Q	Quescient current(1)	V_{VDDIN} = 2V, I_{OUT} = 0 mA	97	186	546	µA
		V_{VDDIN} > 2.2V, I_{OUT} = 0 mA	97	111	147	
P_{EFF}	Power efficiency(1)	I_{OUT} = 5mA, 50mA Reference power not included	82.7	88.3	95	%

1. These values are based on characterization. These values are not covered by test limits in production.

Table 42-40. Decoupling Requirements in Switching Mode

Symbol	Parameter	Technology	Typ	Units
C_{IN1}	Input regulator capacitor 1		33	nF
C_{IN2}	Input regulator capacitor 2		100	
C_{IN3}	Input regulator capacitor 3		10	µF
C_{OUT1}	Output regulator capacitor 1	X7R MLCC	100	nF
C_{OUT2}	Output regulator capacitor 2	X7R MLCC (ex : GRM31CR71A475)	4.7	µF
L_{EXT}	External inductance	(ex: Murata LQH3NPN220MJ0)	22	µH
R_{DCLEXT}	Serial resistance of L_{EXT}		0.7	Ω
$ISAT_{LEXT}$	Saturation current of L_{EXT}		300	mA

Note: 1. Refer to Section 5. on page 43.

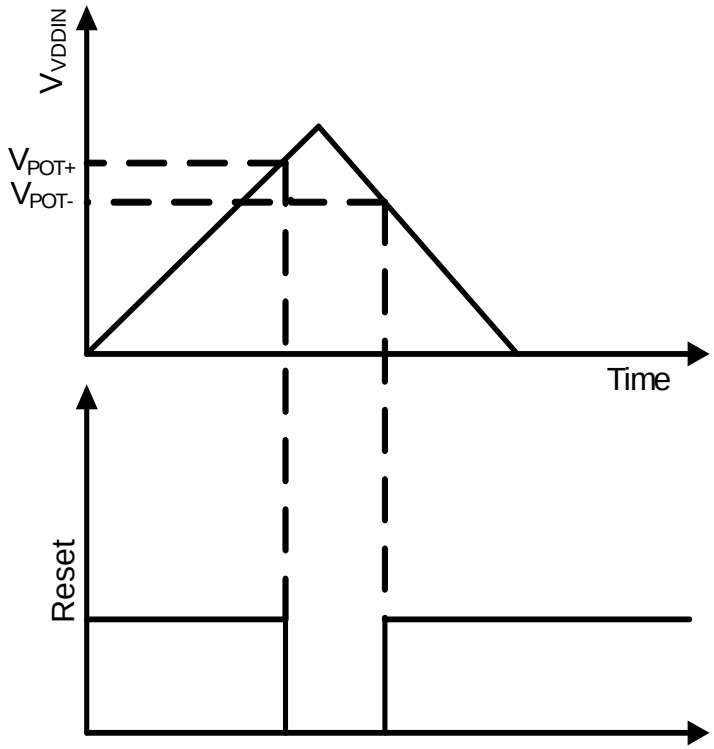
42.9.2 Power-on Reset 33 Characteristics

Table 42-41. POR33 Characteristics (1)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{POT+}	Voltage threshold on V_{VDDIN} rising		1.25		1.55	V
V_{POT-}	Voltage threshold on V_{VDDIN} falling		0.95		1.30	

1. These values are based on characterization. These values are not covered by test limits in production.

Figure 42-4. POR33 Operating Principle



42.9.3 Brown Out Detectors Characteristics

Table 42-42. BOD18 Characteristics (1)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Step size, between adjacent values in BSCIF.BOD18LEVEL(1)			10.1		mV
V_{HYST}	BOD hysteresis(1)	T = 25°C	3		40	
t_{DET}	Detection time(1)	Time with $V_{VDDCORE}$ < BOD18.LEVEL necessary to generate a reset signal	1.2			µs
I_{BOD}	Current consumption(1)	on VDDIN		7.4	14	µA
		on VDDCORE			7	
$t_{STARTUP}$	Startup time(1)				4.5	µs

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

The values in Table 42-43 describe the values of the BOD33.LEVEL in the flash User Page fuses.

Table 42-43. BOD33.LEVEL Values

BOD33.LEVEL Value	Min	Typ	Max	Units
16		2.08		V
20		2.18		
24		2.33		
28		2.48		
32		2.62		
36		2.77		
40		2.92		
44		3.06		
48		3.21		

Table 42-44. BOD33 Characteristics (1)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Step size, between adjacent values in BSCIF.BOD33LEVEL(1)			34.4		mV
V_{HYST}	Hysteresis(1)		45		170	
t_{DET}	Detection time(1)	Time with VDDIN < V_{TH} necessary to generate a reset signal				µs
I_{BOD33}	Current consumption(1)	Normal mode			36	µA
$t_{STARTUP}$	Startup time(1)	Normal mode			6	µs

1. These values are based on simulation. These values are not covered by test limits in production or characterization.

42.9.4 Analog- to Digital Converter Characteristics

Table 42-45. Operating conditions

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Temperature range		-40		+85	°C
	Resolution (1)	Max		12	12 (2)	Bit
	Sampling clock (3)	Differential modes, Gain=1X	5		300	kHz
		Unipolar modes, Gain=1X	5		250	
f_{ADC}	ADC clock frequency(3)	Differential modes	0.03		1.8	MHz
		Unipolar modes	0.03		1.5	
$T_{SAMPLEHOLD}$	Sampling time(3)	Differential modes	16.5		277	µs
		Unipolar modes	16.5		333	
	Conversion rate(1)	1X gain, differential			300	kSps
	Internal channel conversion rate(3)	V_{VDD}/10, Bandgap and Temperature channels			125	kSps
	Conversion time (latency) Differential mode (no windowing)	1X gain, (resolution/2)+gain (4)			6	Cycles
		2X and 4X gain			7	
		8X and 16X gain			8	
		32X and 64X gain			9	
		64X gain and unipolar			10	

1. These values are based on characterization. These values are not covered by test limits in production
2. Single ended or using divide by two max resolution: 11 bits
3. These values are based on simulation. These values are not covered by test limits in production
4. See Figure 42-5

Figure 42-5. Maximum input common mode voltage

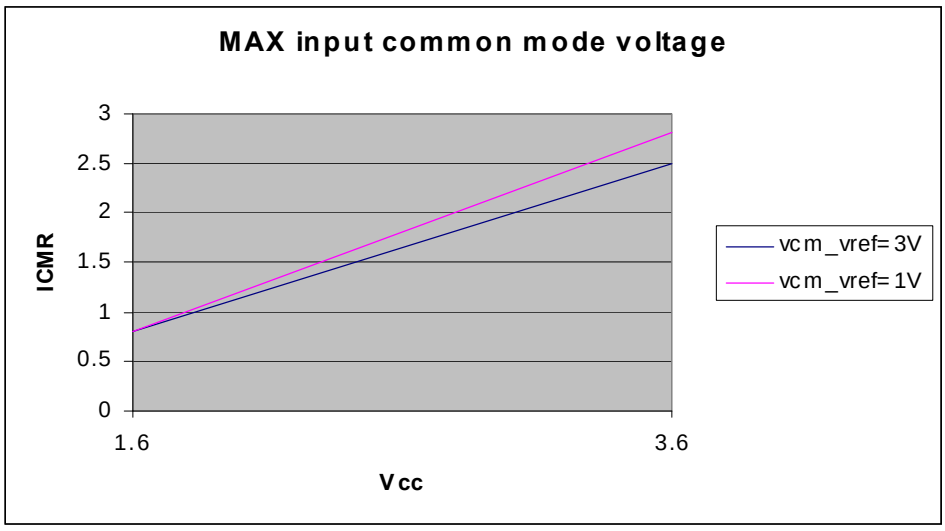


Table 42-46. DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
VDDANA	Supply voltage (1)		1.6		3.6	V
	Reference range (2)	Differential mode	1.0		VDDANA -0.6	V
		Unipolar and Window modes	1.0		1.0	
		Using divide by two function (differential)	2.0		VDDANA	
	Absolute min, max input voltage(2)		-0,1		VDDANA +0.1	V
	Start up time(2)	ADC with reference already enabled		12	24	Cycles
		No gain compensation Reference buffer			5	µs
		Gain compensation Reference buffer			60	Cycles
R_{SAMPLE}	Input channel source resistance(2)				0.5	kΩ
C_{SAMPLE}	Sampling capacitance(2)		2.9	3.6	4.3	pF
	Reference input source resistance(2)	Gain compensation			2	kΩ
		No gain compensation			1	MΩ
	ADC reference settling time(2)	After changing reference/mode (3)		5	60	Cycles

1. These values are based on characterization. These values are not covered by test limits in production
2. These values are based on simulation. These values are not covered by test limits in production
3. Requires refresh/flush otherwise conversion time (latency) + 1

Table 42-47. Differential mode, gain=1

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Accuracy without compensation (1)			7		ENOB
	Accuracy after compensation(1)	(INL, gain and offset)			11	ENOB
INL	Integral Non Linearity (2)	After calibration, Gain compensation		1.2	1.7	LSBs
DNL	Differential Non Linearity(2)	After calibration		0.7	1.0	LSBs
	Gain error (2)	External reference	-5.0	-1.0	5.0	mV
		VDDANA/1.6	-40		40	
		VDDANA/2.0	-40		40	
		Bandgap After calibration	-30		30	
	Gain error drift vs voltage(1)	External reference	-2		2	mV/V
	Gain error drift vs temperature(1)	After calibration + bandgap drift If using onchip bandgap			0.08	mV/°K
	Offset error (2)	External reference	-5.0		5.0	mV
		VDDANA/1.6	-10		10	
		VDDANA/2.0	-10		10	
		Bandgap After calibration	-10		10	
	Offset error drift vs voltage(1)		-4		4	mV/V

Table 42-47. Differential mode, gain=1

	Offset error drift vs temperature[1]				0.04	mV/°K
	Conversion range [2]	Vin-Vip	-Vref		Vref	V
	ICMR[1]			see Figure 42-5		
	PSRR[1]	fvdd=1Hz, ext ADVREFP=3.0V V_{VDD}=3.6V		100		dB
		fvdd=2MHz, ext ADVREFP=3.0V V_{VDD}=3.6		50		
	DC supply current [2]	VDDANA=3.6V, ADVREFP=3.0V		1.2		mA
		VDDANA=1.6V, ADVREFP=1.0V		0.6		

1. These values are based on simulation only. These values are not covered by test limits in production or characterization
2. These values are based on characterization and not tested in production, and valid for an input voltage between 10% to 90% of reference voltage.

Table 42-48. Unipolar mode, gain=1

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Accuracy without compensation [1]			7		ENOB
	Accuracy after compensation[1]				11	ENOB
INL	Integral Non Linearity [2]	After calibration Dynamic tests No gain compensation			±3	LSBs
		After calibration Dynamic tests Gain compensation			±3	
DNL	Differential Non Linearity[2]	After calibration			±2.8	LSBs
	Gain error[2]	External reference	-15		15	mV
		VDDANA/1.6	-50		50	
		VDDANA/2.0	-30		30	
		Bandgap After calibration	-10		10	
	Gain error drift vs voltage[1]	External reference	-8		8	mV/V
	Gain error drift temperature[1]	+ bandgap drift If using bandgap			0.08	mV/°K
	Offset error[2]	External reference	-15		15	mV
		VDDANA/1.6	-15		15	
		VDDANA/2.0	-15		15	
		Bandgap After calibration	-10		10	
	Offset error drift[1]		-4		4	mV/V
	Offset error drift temperature[1]			0	0.04	mV/°K
	Conversion range[1]	Vin-Vip	-Vref		Vref	V
	ICMR[1]			see Figure 42-5		

Table 42-48. Unipolar mode, gain=1

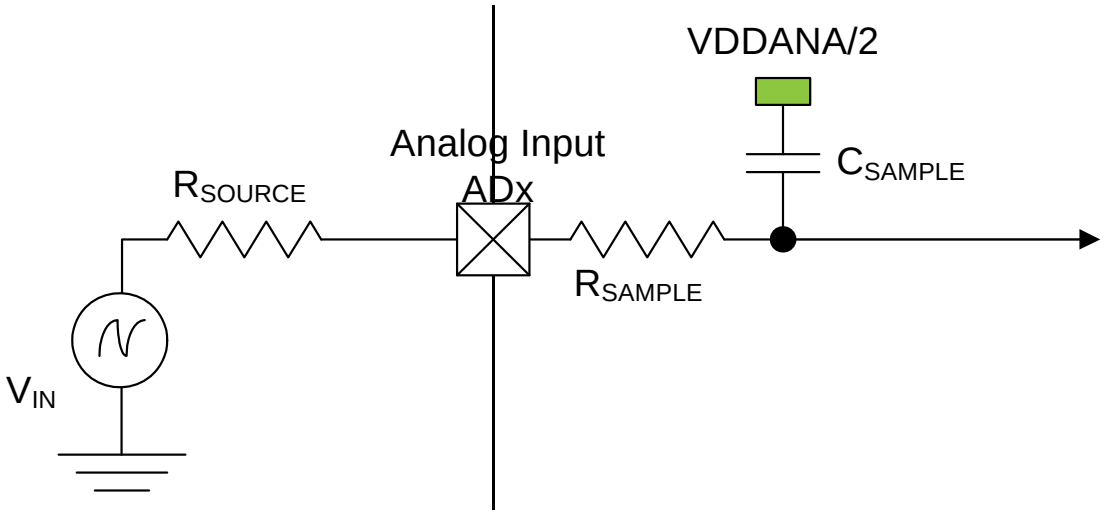
	PSRR[1]	fVdd=100kHz, VDDIO=3.6V		62		dB
		fVdd=1MHz, VDDIO=3.6V		49		
	DC supply current[1]	VDDANA=3.6V		1	2	mA
		VDDANA=1.6V, ADVREFP=1.0V		1	1.3	

1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. These values are based on characterization and not tested in production, and valid for an input voltage between 10% to 90% of reference voltage.

42.9.4.1 Inputs and Sample and Hold Acquisition Times

The analog voltage source must be able to charge the sample and hold (S/H) capacitor in the ADC in order to achieve maximum accuracy. Seen externally the ADC input consists of a resistor (R_{SAMPLE}) and a capacitor (C_{SAMPLE}). In addition, the source resistance (R_{SOURCE}) must be taken into account when calculating the required sample and hold time. Figure 42-6 shows the ADC input channel equivalent circuit.

Figure 42-6. ADC Input



To achieve *n* bits of accuracy, the C_{SAMPLE} capacitor must be charged at least to a voltage of

$$V_{CSAMPLE} \geq V_{IN} \times (1 - 2^{-(n+1)})$$

The minimum sampling time $t_{SAMPLEHOLD}$ for a given R_{SOURCE} can be found using this formula:

$$t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times (n+1) \times \ln(2)$$

for a 12 bits accuracy : $t_{SAMPLEHOLD} \geq (R_{SAMPLE} + R_{SOURCE}) \times (C_{SAMPLE}) \times 9,02$

where

$$t_{SAMPLEHOLD} = \frac{1}{2 \times fADC}$$

42.9.5 Digital to Analog Converter Characteristics

Table 42-49. Operating conditions

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Analog Supply Voltage (1)	on VDDANA	2.4	3	3.6	V
	Digital Supply Voltage(1)	on VDDCORE	1.62	1.8	1.98	V
	Resolution (2)			10		bits
	Clock frequency(1)	Cload = 50pF ; Rload = 5kΩ			500	kHz
	Load(1)	CLoad			50	pF
		RLoad	5			kΩ
INL	Integral Non Linearity (1)	Best fit-line method			±2	LSBs
DNL	Differential Non Linearity (1)	Best fit-line method	-0.9		+1	LSBs
	Zero Error (offset) (1)	CDR[9:0] = 0		1	5	mV
	Gain Error (1)	CDR[9:0] = 1023		5	10	mV
	Total Harmonic Distortion(1)	80% of VDDANA @ fin = 70kHz	-56		7	dB
	Delay to vout (1)	CDR[9:0] = 512/ Cload = 50 pF / Rload = 5 kΩ	2			µs
	Startup time(1)	CDR[9:0] = 512	5		9	µs
	Output Voltage Range	(ADVREFP < VDDANA – 100mV) is mandatory	0		ADVREFP	V
	ADVREFP Voltage Range(1)	(ADVREFP < VDDANA – 100mV) is mandatory	2.3		3.5	V
	ADVREFN Voltage Range(1)	ADVREFP = GND		0		V
	Standby Current(1)	On VDDANA			500	nA
		On VDDCORE			100	
	DC Current consumption(1)	On VDDANA (no Rload)		485	660	µA
		On ADVREFP (CDR[9:0] = 512)		250	295	

1. These values are based on simulation. These values are not covered by test limits in production or characterization
2. These values are based on characterization. These values are not covered by test limits in production

42.9.6 Analog Comparator Characteristics

Table 42-50. Analog Comparator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Positive input voltage range		0.1		VDDIO-0.1	V
	Negative input voltage range		0.1		VDDIO-0.1	
	Offset (1)	V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 0 (2) Fast mode	-12		13	mV
		V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 0(2) Low power mode	-11		12	mV

Table 42-50. Analog Comparator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
	Hysteresis[1]	V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 1[2] Fast mode	10		55	mV
		V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 1[2] Low power mode	10		68	mV
		V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 2[2] Fast mode	26		83	mV
		V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 2[2] Low power mode	19		91	mV
		V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 3[2] Fast mode	43		106	mV
		V_{ACREFN} =0.1V to VDDIO-0.1V, hysteresis = 3[2] Low power mode	32		136	mV
	Propagation delay[1]	Changes for V_{ACM}=VDDIO/2 100mV Overdrive Fast mode			67	ns
		Changes for V_{ACM}=VDDIO/2 100mV Overdrive Low power mode			315	ns
$t_{STARTUP}$	Startup time[1]	Enable to ready delay Fast mode			1.19	µs
		Enable to ready delay Low power mode			3.61	µs
I_{AC}	Channel current consumption [3]	Low power mode, no hysteresis		4.9	8.7	µA
		Fast mode, no hysteresis		63	127	

1. These values are based on characterization. These values are not covered by test limits in production
2. HYSTAC.CONFn.HYS field, refer to the Analog Comparator Interface chapter
3. These values are based on simulation. These values are not covered by test limits in production or characterization

42.9.7 Liquid Crystal Display Controler characteristics

Table 42-51. Liquid Crystal Display Controler characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
SEG	Segment Terminal Pins				40	
COM	Common Terminal Pins				4	
f_{Frame}	LCD Frame Frequency	F_{CLKLCD}	31.25		512	Hz
C_{Flying}	Flying Capacitor			100		nF
V_{LCD}	LCD Regulated Voltages (1) CFG.FCST=0	C_{Flying} = 100nF 100nF on V_{LCD}, BIAS2 and BIAS1 pins		3		V
BIAS2				$2*V_{LCD}/3$		
BIAS1				$V_{LCD}/3$		

1. These values are based on simulation. These values are not covered by test limits in production or characterization

42.9.7.1 Liquid Crystal Controler supply current

The values in Table 42-52 are measured values of power consumption under the following conditions, except where noted:

- T=25°C, WAIT mode, Low power waveform, Frame Rate = 32Hz from OSC32K
- Configuration: 4COMx40SEG, 1/4 Duty, 1/3 Bias, No animation
- All segments on, Load = 160 x 22pF between each COM and each SEG.
- LCDCA current based on $I_{LCD} = I_{WAIT}$(Lcd On) - I_{WAIT}(Lcd Off)

Table 42-52. Liquid Crystal Display Controler supply current

Symbol	Conditions		Min	Typ	Max	Units
I_{LCD}	Internal voltage generation CFG.FCST=0	V_{VDDIN} = 3.6V		8.85		µA
		V_{VDDIN} = 1.8V		6.16		
	External bias V_{LCD}=3.0V	V_{VDDIN} = 3.3V		0.98		
		V_{VDDIN} = 1.8V		1.17		

42.10 Timing Characteristics

42.10.1 RESET_N Timing

Table 42-53. RESET_N Waveform Parameters (1)

Symbol	Parameter	Conditions	Min	Max	Units
t_{RESET}	RESET_N minimum pulse length		10		ns

1. These values are based on simulation. These values are not covered by test limits in production.

42.10.2 USART in SPI Mode Timing

42.10.2.1 Master mode

Figure 42-7. USART in SPI Master Mode with (CPOL= CPHA= 0) or (CPOL= CPHA= 1)

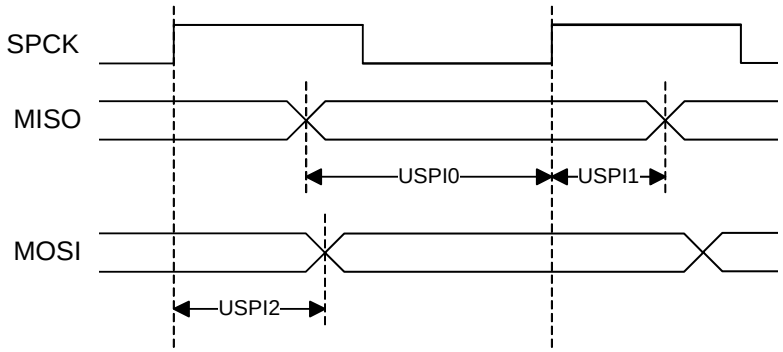


Figure 42-8. USART in SPI Master Mode with (CPOL= 0 and CPHA= 1) or (CPOL= 1 and CPHA= 0)

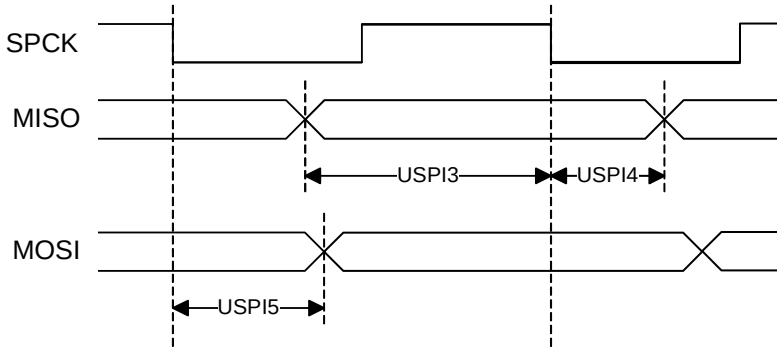


Table 42-54. USART0 in SPI Mode Timing, Master Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI0	MISO setup time before SPCK rises	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	123.2 + t_{SAMPLE}[2]		ns
USPI1	MISO hold time after SPCK rises		24.74 -t_{SAMPLE}[2]		
USPI2	SPCK rising to MOSI delay			513.56	
USPI3	MISO setup time before SPCK falls		125.99 + t_{SAMPLE}[2]		
USPI4	MISO hold time after SPCK falls		24.74 -t_{SAMPLE}[2]		
USPI5	SPCK falling to MOSI delay			516.55	

Table 42-55. USART1 in SPI Mode Timing, Master Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI0	MISO setup time before SPCK rises	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	69.28 + t_{SAMPLE}[2]		ns
USPI1	MISO hold time after SPCK rises		25.75 -t_{SAMPLE}[2]		
USPI2	SPCK rising to MOSI delay			99.66	
USPI3	MISO setup time before SPCK falls		73.12 + t_{SAMPLE}[2]		
USPI4	MISO hold time after SPCK falls		28.10 -t_{SAMPLE}[2]		
USPI5	SPCK falling to MOSI delay			102.01	

Table 42-56. USART2 in SPI Mode Timing, Master Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI0	MISO setup time before SPCK rises	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	69.09 + t_{SAMPLE}[2]		ns
USPI1	MISO hold time after SPCK rises		26.52 -t_{SAMPLE}[2]		
USPI2	SPCK rising to MOSI delay			542.96	
USPI3	MISO setup time before SPCK falls		72.55 + t_{SAMPLE}[2]		
USPI4	MISO hold time after SPCK falls		28.37 -t_{SAMPLE}[2]		
USPI5	SPCK falling to MOSI delay			544.80	

Table 42-57. USART3 in SPI Mode Timing, Master Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI0	MISO setup time before SPCK rises	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	147.24 + t_{SAMPLE}[2]		ns
USPI1	MISO hold time after SPCK rises		25.80 -t_{SAMPLE}[2]		
USPI2	SPCK rising to MOSI delay			88.23	
USPI3	MISO setup time before SPCK falls		154.9 + t_{SAMPLE}[2]		
USPI4	MISO hold time after SPCK falls		26.89 -t_{SAMPLE}[2]		
USPI5	SPCK falling to MOSI delay			89.32	

Notes: 1. These values are based on simulation. These values are not covered by test limits in production.

2. Where: $t_{SAMPLE} = t_{SPCK} - \left(\left\lfloor \frac{t_{SPCK}}{2 \times t_{CLKUSART}} \right\rfloor \frac{1}{2} \right) \times t_{CLKUSART}$

Maximum SPI Frequency, Master Output

The maximum SPI master output frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(f_{PINMAX}, \frac{1}{SPIn}, \frac{f_{CLKSPI} \times 2}{9})$$

Where $SPIn$ is the MOSI delay, USPI2 or USPI5 depending on CPOL and NCPHA. f_{PINMAX} is the maximum frequency of the SPI pins. refer to the I/O Pin Characteristics section for the maximum frequency of the pins. f_{CLKSPI} is the maximum frequency of the CLK_SPI. Refer to the SPI chapter for a description of this clock.

Maximum SPI Frequency, Master Input

The maximum SPI master input frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(\frac{1}{SPIn + t_{VALID}}, \frac{f_{CLKSPI} \times 2}{9})$$

Where $SPIn$ is the MISO setup and hold time, USPI0 + USPI1 or USPI3 + USPI4 depending on CPOL and NCPHA. T_{VALID} is the SPI slave response time. refer to the SPI slave datasheet for $T_{VALID} \cdot f_{CLKSPI}$ is the maximum frequency of the CLK_SPI. Refer to the SPI chapter for a description of this clock.

42.10.2.2 Slave mode

Figure 42-9. USART in SPI Slave Mode with (CPOL= 0 and CPHA= 1) or (CPOL= 1 and CPHA= 0)

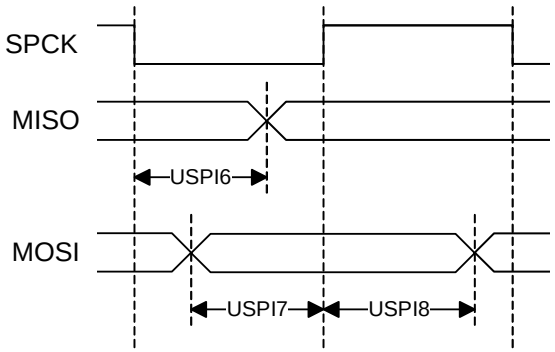


Figure 42-10. USART in SPI Slave Mode with (CPOL= CPHA= 0) or (CPOL= CPHA= 1)

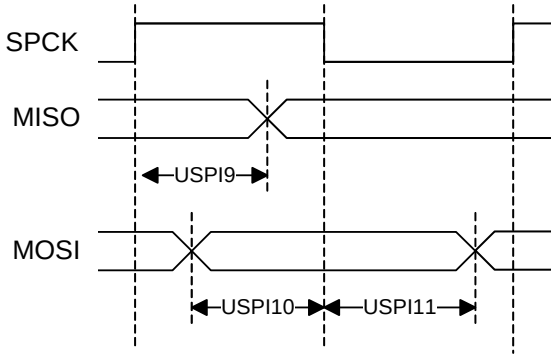


Figure 42-11. USART in SPI Slave Mode, NPCS Timing

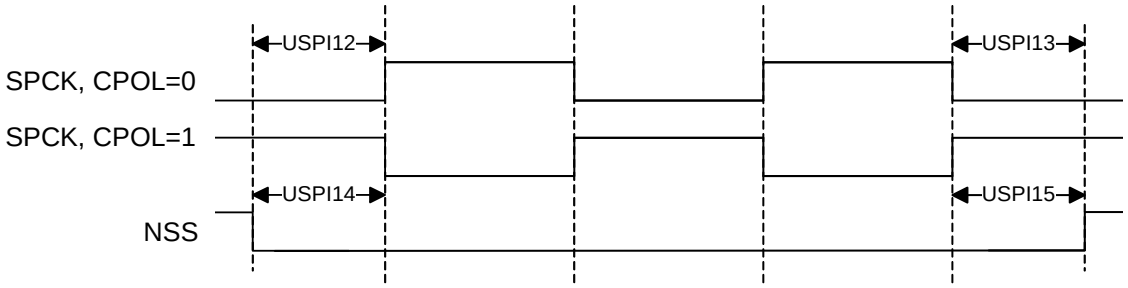


Table 42-58. USART0 in SPI mode Timing, Slave Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI6	SPCK falling to MISO delay	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF		740.67	ns
USPI7	MOSI setup time before SPCK rises		56.73 + t_{SAMPLE}[2] + t_{CLK_USART}		
USPI8	MOSI hold time after SPCK rises		45.18 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI9	SPCK rising to MISO delay			670.18	
USPI10	MOSI setup time before SPCK falls		56.73 +(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI11	MOSI hold time after SPCK falls		45.18 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI12	NSS setup time before SPCK rises		688.71		
USPI13	NSS hold time after SPCK falls		-2.25		
USPI14	NSS setup time before SPCK falls		688.71		
USPI15	NSS hold time after SPCK rises		-2.25		

Table 42-59. USART1 in SPI mode Timing, Slave Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI6	SPCK falling to MISO delay	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF		373.58	ns
USPI7	MOSI setup time before SPCK rises		4.16 + t_{SAMPLE}[2] + t_{CLK_USART}		
USPI8	MOSI hold time after SPCK rises		46.69 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI9	SPCK rising to MISO delay			373.54	
USPI10	MOSI setup time before SPCK falls		4.16 +(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI11	MOSI hold time after SPCK falls		46.69 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI12	NSS setup time before SPCK rises		200.43		
USPI13	NSS hold time after SPCK falls		-16.5		
USPI14	NSS setup time before SPCK falls		200.43		
USPI15	NSS hold time after SPCK rises		-16.5		

Table 42-60. USART2 in SPI mode Timing, Slave Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI6	SPCK falling to MISO delay	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF		770.02	ns
USPI7	MOSI setup time before SPCK rises		136.56 + t_{SAMPLE}[2] + t_{CLK_USART}		
USPI8	MOSI hold time after SPCK rises		47.9 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI9	SPCK rising to MISO delay			570.19	
USPI10	MOSI setup time before SPCK falls		136.73 +(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI11	MOSI hold time after SPCK falls		47.9 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI12	NSS setup time before SPCK rises		519.87		
USPI13	NSS hold time after SPCK falls		-1.83		
USPI14	NSS setup time before SPCK falls		519.87		
USPI15	NSS hold time after SPCK rises		-1.83		

Table 42-61. USART3 in SPI mode Timing, Slave Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
USPI6	SPCK falling to MISO delay	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF		593.9	ns
USPI7	MOSI setup time before SPCK rises		45.93 + t_{SAMPLE}[2] + t_{CLK_USART}		
USPI8	MOSI hold time after SPCK rises		47.03 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI9	SPCK rising to MISO delay			593.38	
USPI10	MOSI setup time before SPCK falls		45.93 +(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI11	MOSI hold time after SPCK falls		47.03 -(t_{SAMPLE}[2] + t_{CLK_USART})		
USPI12	NSS setup time before SPCK rises		237.5		
USPI13	NSS hold time after SPCK falls		-1.81		
USPI14	NSS setup time before SPCK falls		237.5		
USPI15	NSS hold time after SPCK rises		-1.81		

Notes: 1. These values are based on simulation. These values are not covered by test limits in production.

2. Where: $t_{SAMPLE} = t_{SPCK} - \left(\left\lfloor \frac{t_{SPCK}}{2 \times t_{CLKUSART}} \right\rfloor + \frac{1}{2}\right) \times t_{CLKUSART}$

Maximum SPI Frequency, Slave Input Mode

The maximum SPI slave input frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(\frac{f_{CLKSPI} \times 2}{9}, \frac{1}{SPIn})$$

Where $SPIn$ is the MOSI setup and hold time, USPI7 + USPI8 or USPI10 + USPI11 depending on CPOL and NCPHA. f_{CLKSPI} is the maximum frequency of the CLK_SPI. Refer to the SPI chapter for a description of this clock.

Maximum SPI Frequency, Slave Output Mode

The maximum SPI slave output frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(\frac{f_{CLKSPI} \times 2}{9}, f_{PINMAX}, \frac{1}{SPIn + t_{SETUP}})$$

Where $SPIn$ is the MISO delay, USPI6 or USPI9 depending on CPOL and NCPHA. T_{SETUP} is the SPI master setup time. refer to the SPI master datasheet for T_{SETUP} . f_{CLKSPI} is the maximum frequency of the CLK_SPI. Refer to the SPI chapter for a description of this clock. f_{PINMAX} is the maximum frequency of the SPI pins. refer to the I/O Pin Characteristics section for the maximum frequency of the pins.

42.10.3 SPI Timing

42.10.3.1 Master mode

Figure 42-12. SPI Master Mode with (CPOL= NCPHA= 0) or (CPOL= NCPHA= 1)

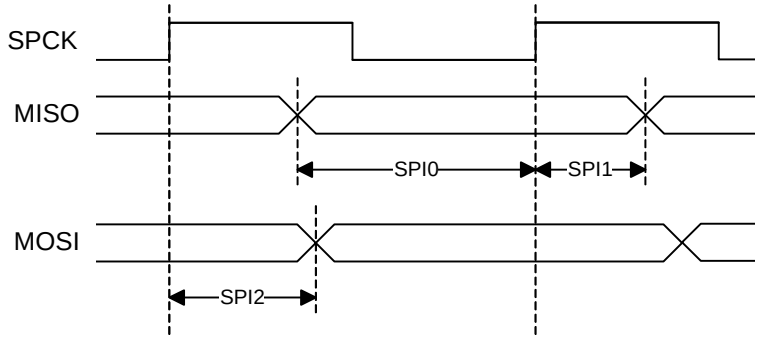


Figure 42-13. SPI Master Mode with (CPOL= 0 and NCPHA= 1) or (CPOL= 1 and NCPHA= 0)

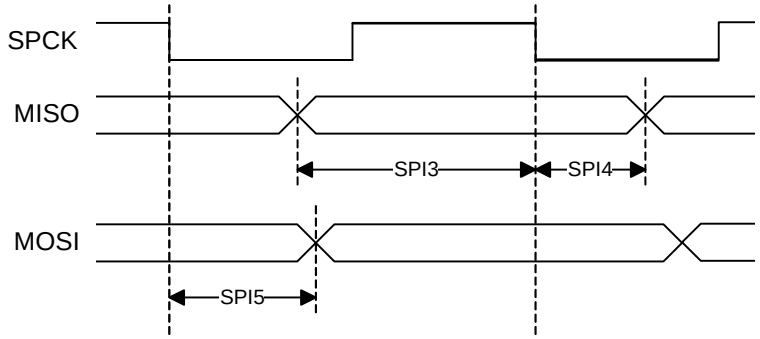


Table 42-62. SPI Timing, Master Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
SPI0	MISO setup time before SPCK rises	V_{VDDIO} from 2.85V to 3.6V, maximum external capacitor = 40pF	9		ns
SPI1	MISO hold time after SPCK rises		0		
SPI2	SPCK rising to MOSI delay		9	21	
SPI3	MISO setup time before SPCK falls		7.3		
SPI4	MISO hold time after SPCK falls		0		
SPI5	SPCK falling to MOSI delay		9	22	

Note: 1. These values are based on simulation. These values are not covered by test limits in production.

Maximum SPI Frequency, Master Output

The maximum SPI master output frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(f_{PINMAX}, \frac{1}{SPIn})$$

Where $SPIn$ is the MOSI delay, SPI2 or SPI5 depending on CPOL and NCPHA. f_{PINMAX} is the maximum frequency of the SPI pins. refer to the I/O Pin Characteristics section for the maximum frequency of the pins.

Maximum SPI Frequency, Master Input

The maximum SPI master input frequency is given by the following formula:

$$f_{SPCKMAX} = \frac{1}{SPIn + t_{VALID}}$$

Where $SPIn$ is the MISO setup and hold time, SPI0 + SPI1 or SPI3 + SPI4 depending on CPOL and NCPHA. t_{VALID} is the SPI slave response time. refer to the SPI slave datasheet for t_{VALID} .

42.10.3.2 Slave mode

Figure 42-14. SPI Slave Mode with (CPOL= 0 and NCPHA= 1) or (CPOL= 1 and NCPHA= 0)

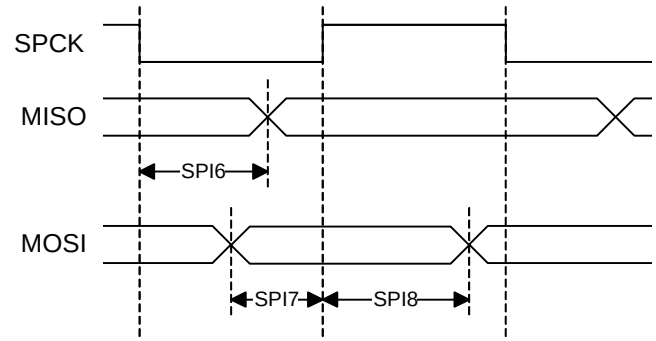


Figure 42-15. SPI Slave Mode with (CPOL= NCPHA= 0) or (CPOL= NCPHA= 1)

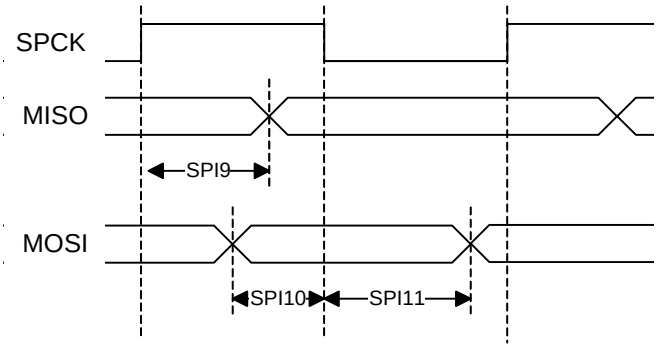


Figure 42-16. SPI Slave Mode, NPCS Timing

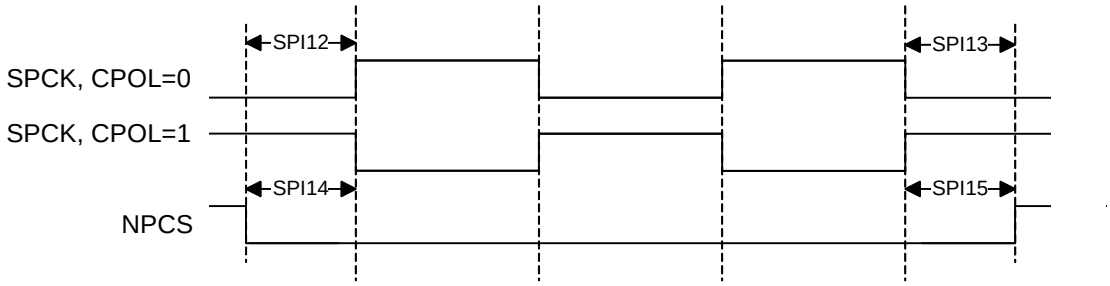


Table 42-63. SPI Timing, Slave Mode[1]

Symbol	Parameter	Conditions	Min	Max	Units
SPI6	SPCK falling to MISO delay	V_{VDDIO} from 2.85V to 3.6V, maximum external capacitor = 40pF	19	47	ns
SPI7	MOSI setup time before SPCK rises		0		
SPI8	MOSI hold time after SPCK rises		5.4		
SPI9	SPCK rising to MISO delay		19	46	
SPI10	MOSI setup time before SPCK falls		0		
SPI11	MOSI hold time after SPCK falls		5.3		
SPI12	NPCS setup time before SPCK rises		4		
SPI13	NPCS hold time after SPCK falls		2.5		
SPI14	NPCS setup time before SPCK falls		6		
SPI15	NPCS hold time after SPCK rises		1.1		

Note: 1. These values are based on simulation. These values are not covered by test limits in production.

Maximum SPI Frequency, Slave Input Mode

The maximum SPI slave input frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(f_{CLKSPI}, \frac{1}{SPIn})$$

Where $SPIn$ is the MOSI setup and hold time, SPI7 + SPI8 or SPI10 + SPI11 depending on CPOL and NCPHA. f_{CLKSPI} is the maximum frequency of the CLK_SPI. Refer to the SPI chapter for a description of this clock.

Maximum SPI Frequency, Slave Output Mode

The maximum SPI slave output frequency is given by the following formula:

$$f_{SPCKMAX} = MIN(f_{PINMAX}, \frac{1}{SPIn + t_{SETUP}})$$

Where $SPIn$ is the MISO delay, SPI6 or SPI9 depending on CPOL and NCPHA. t_{SETUP} is the SPI master setup time. refer to the SPI master datasheet for t_{SETUP} . f_{PINMAX} is the maximum frequency of the SPI pins. refer to the I/O Pin Characteristics section for the maximum frequency of the pins.

42.10.4 TWIM/TWIS Timing

Figure 42-64 shows the TWI-bus timing requirements and the compliance of the device with them. Some of these requirements (t_r and t_f) are met by the device without requiring user intervention. Compliance with the other requirements (t_{HD-STA}, t_{SU-STA}, t_{SU-STO}, t_{HD-DAT}, $t_{SU-DAT-TWI}$, $t_{LOW-TWI}$, t_{HIGH}, and f_{TWCK}) requires user intervention through appropriate programming of the relevant TWIM and TWIS user interface registers. refer to the TWIM and TWIS sections for more information.

Table 42-64. TWI-Bus Timing Requirements

Symbol	Parameter	Mode	Minimum		Maximum		Unit
			Requirement	Device	Requirement	Device	
t_r	TWCK and TWD rise time	Standard(1)	-		1000		ns
		Fast(1)	20 + 0.1C_b		300		
t_f	TWCK and TWD fall time	Standard	-		300		ns
		Fast	20 + 0.1C_b		300		
t_{HD-STA}	(Repeated) START hold time	Standard	4	t_{clkpb}	-		µs
		Fast	0.6				
t_{SU-STA}	(Repeated) START set-up time	Standard	4.7	t_{clkpb}	-		µs
		Fast	0.6				
t_{SU-STO}	STOP set-up time	Standard	4.0	$4t_{clkpb}$	-		µs
		Fast	0.6				
t_{HD-DAT}	Data hold time	Standard	0.3(2)	$2t_{clkpb}$	3.45()	$15t_{prescaled} + t_{clkpb}$	µs
		Fast			0.9()		
$t_{SU-DAT-TWI}$	Data set-up time	Standard	250	$2t_{clkpb}$	-		ns
		Fast	100				
t_{SU-DAT}		-	-	t_{clkpb}	-		-
$t_{LOW-TWI}$	TWCK LOW period	Standard	4.7	$4t_{clkpb}$	-		µs
		Fast	1.3				
t_{LOW}		-	-	t_{clkpb}	-		-
t_{HIGH}	TWCK HIGH period	Standard	4.0	$8t_{clkpb}$	-		µs
		Fast	0.6				
f_{TWCK}	TWCK frequency	Standard	-		100	$\frac{1}{12t_{clkpb}}$	kHz
		Fast			400		

Notes: 1. Standard mode: $f_{TWCK} \leq 100$ kHz ; fast mode: $f_{TWCK} > 100$ kHz .

2. A device must internally provide a hold time of at least 300 ns for TWD with reference to the falling edge of TWCK.

Notations:

C_b = total capacitance of one bus line in pF

t_{clkpb} = period of TWI peripheral bus clock

$t_{prescaled}$ = period of TWI internal prescaled clock (see chapters on TWIM and TWIS)

The maximum $t_{HD;DAT}$ has only to be met if the device does not stretch the LOW period ($t_{LOW-TWI}$) of TWCK.

42.10.5 JTAG Timing

Figure 42-17. JTAG Interface Signals

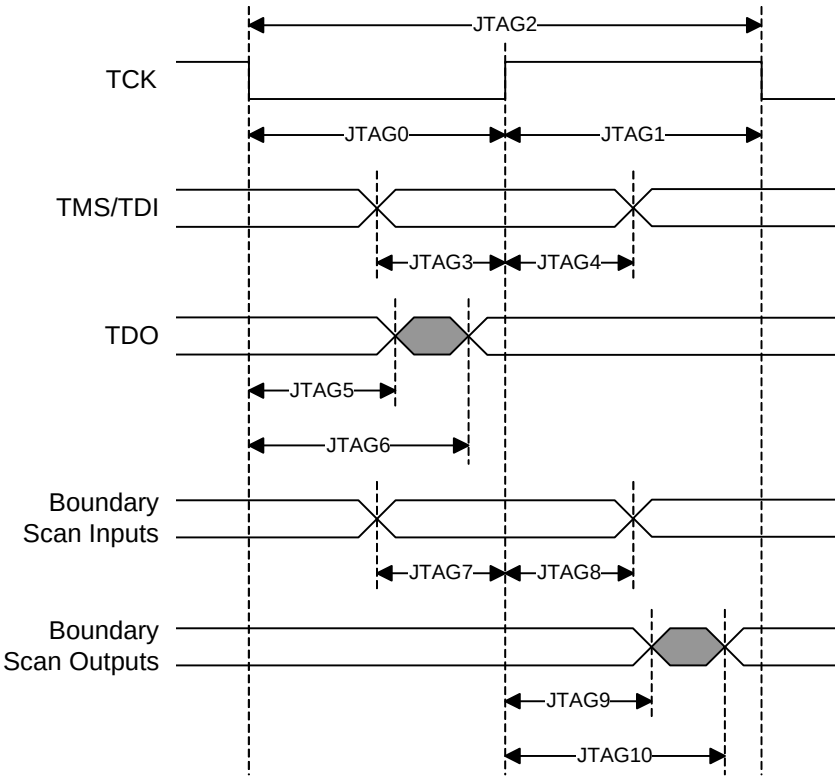


Table 42-65. JTAG Timings[1]

Symbol	Parameter	Conditions	Min	Max	Units
JTAG0	TCK Low Half-period	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	21.8		ns
JTAG1	TCK High Half-period		8.6		
JTAG2	TCK Period		30.3		
JTAG3	TDI, TMS Setup before TCK High		2.0		
JTAG4	TDI, TMS Hold after TCK High		2.3		
JTAG5	TDO Hold Time		9.5		
JTAG6	TCK Low to TDO Valid			21.8	
JTAG7	Boundary Scan Inputs Setup Time		0.6		
JTAG8	Boundary Scan Inputs Hold Time		6.9		
JTAG9	Boundary Scan Outputs Hold Time		9.3		
JTAG10	TCK to Boundary Scan Outputs Valid			32.2	

Note: 1. These values are based on simulation. These values are not covered by test limits in production.

42.10.6 SWD Timing

Figure 42-18. SWD Interface Signals

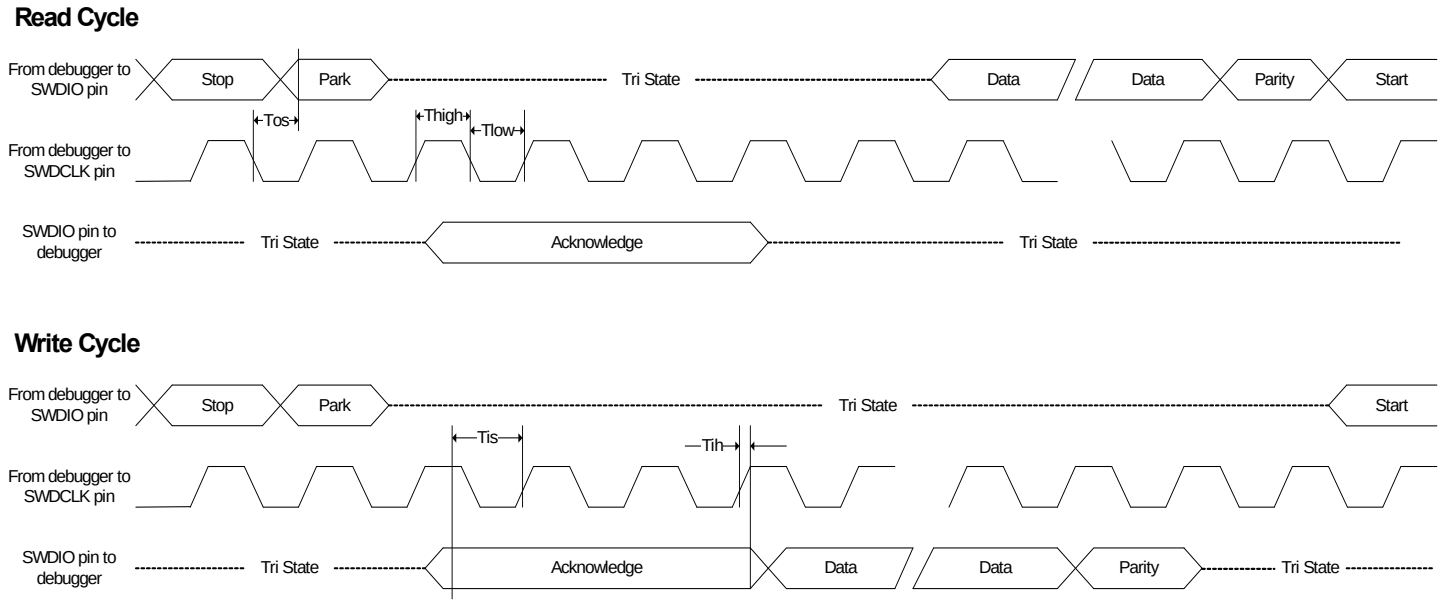


Table 42-66. SWD Timings[1]

Symbol	Parameter	Conditions	Min	Max	Units
Thigh	SWDCLK High period	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF	10	500 000	ns
Tlow	SWDCLK Low period		10	500 000	
Tos	SWDIO output skew to falling edge SWDCLK		-5	5	
Tis	Input Setup time required between SWDIO		4	-	
Tih	Input Hold time required between SWDIO and rising edge SWDCLK		1	-	

Note: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

43. Mechanical Characteristics

43.1 Thermal Considerations

43.1.1 Thermal Data

Table 43-1 summarizes the thermal resistance data depending on the package.

Table 43-1. Thermal Resistance Data

Symbol	Parameter	Condition	Package	Typ	Unit
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	TQFP100	48.1	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP100	13.3	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	VFBGA100	31.1	°C/W
θ_{JC}	Junction-to-case thermal resistance		VFBGA100	6.9	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	WLCSP64	26.9	°C/W
θ_{JC}	Junction-to-case thermal resistance		WLCSP64	0.2	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	TQFP64	49.6	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP64	13.5	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	QFN64	22.0	°C/W
θ_{JC}	Junction-to-case thermal resistance		QFN64	1.3	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	TQFP48	51.1	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP48	13.7	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	QFN48	24.9	°C/W
θ_{JC}	Junction-to-case thermal resistance		QFN48	1.3	

43.1.2 Junction Temperature

The average chip-junction temperature, T_J, in °C can be obtained from the following:

1. $T_J = T_A + (P_D \times \theta_{JA})$
2. $T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$

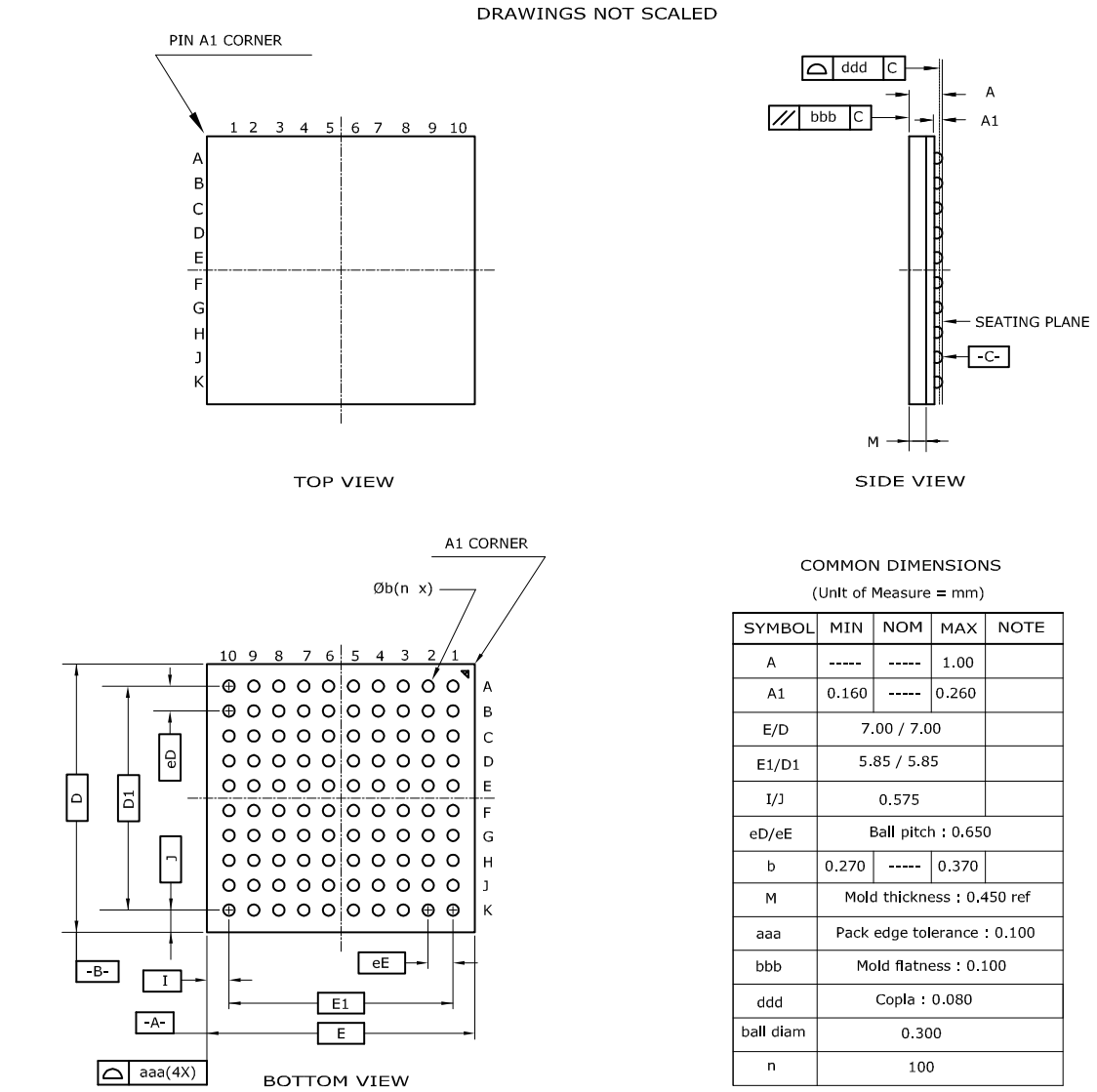
where:

- θ_{JA} = package thermal resistance, Junction-to-ambient (°C/W), provided in Table 43-1.
- θ_{JC} = package thermal resistance, Junction-to-case thermal resistance (°C/W), provided in Table 43-1.
- $\theta_{HEAT\ SINK}$ = cooling device thermal resistance (°C/W), provided in the device datasheet.
- P_D = device power consumption (W) estimated from data provided in Section 42.5 on page 1125.
- T_A = ambient temperature (°C).

From the first equation, the user can derive the estimated lifetime of the chip and decide if a cooling device is necessary or not. If a cooling device is to be fitted on the chip, the second equation should be used to compute the resulting average chip-junction temperature T_J in °C.

43.2 Package Drawings

Figure 43-1. VFBGA-100 package drawing



COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	-----	-----	1.00	
A1	0.160	-----	0.260	
E/D	7.00 / 7.00			
E1/D1	5.85 / 5.85			
I/J	0.575			
eD/eE	Ball pitch : 0.650			
b	0.270	-----	0.370	
M	Mold thickness : 0.450 ref			
aaa	Pack edge tolerance : 0.100			
bbb	Mold flatness : 0.100			
ddd	Copla : 0.080			
ball diam	0.300			
n	100			

Notes : 1. No JEDEC Drawing Reference.
2. Array as seen from the bottom of the package.
3. Dimension A includes stand-off height A1, package body thickness, and lid height, but does not include attached features.
4. Dimension b is measured at the maximum ball diameter, parallel to primary datum C.

Table 43-2. Device and Package Maximum Weight

120	mg

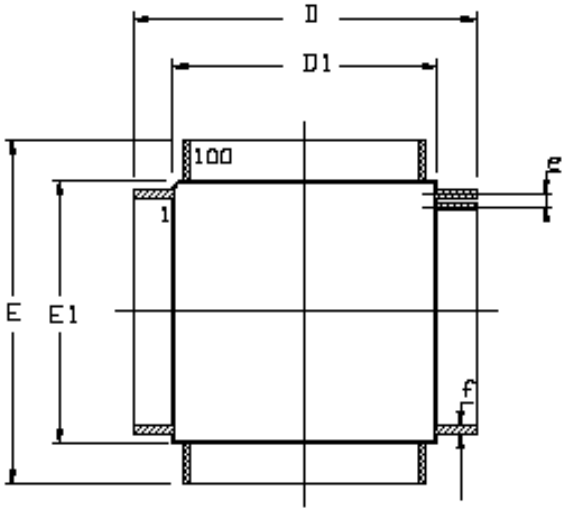
Table 43-3. Package Characteristics

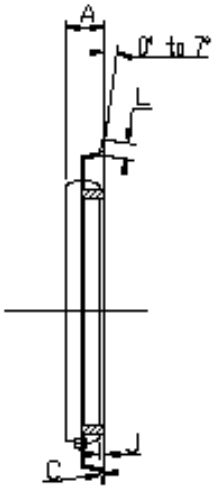
Moisture Sensitivity Level	MSL3

Table 43-4. Package Reference

JEDEC Drawing Reference	N/A
JESD97 Classification	E1

Figure 43-2. TQFP-100 Package Drawing





COMMON DIMENSIONS IN MM

SYMBOL	Min	Max	NOTES
A	----	1.20	
A1	0.95	1.05	
C	0.09	0.20	
D	16.00 BSC		
D1	14.00 BSC		
E	16.00 BSC		
E1	14.00 BSC		
J	0.05	0.15	
L	0.45	0.75	
e	0.50 BSC		
f	0.17	0.27	

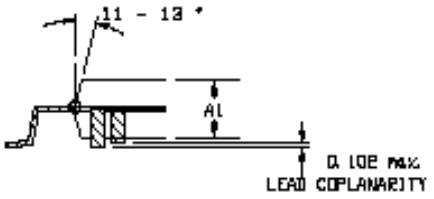


Table 43-5. Device and Package Maximum Weight

500	mg

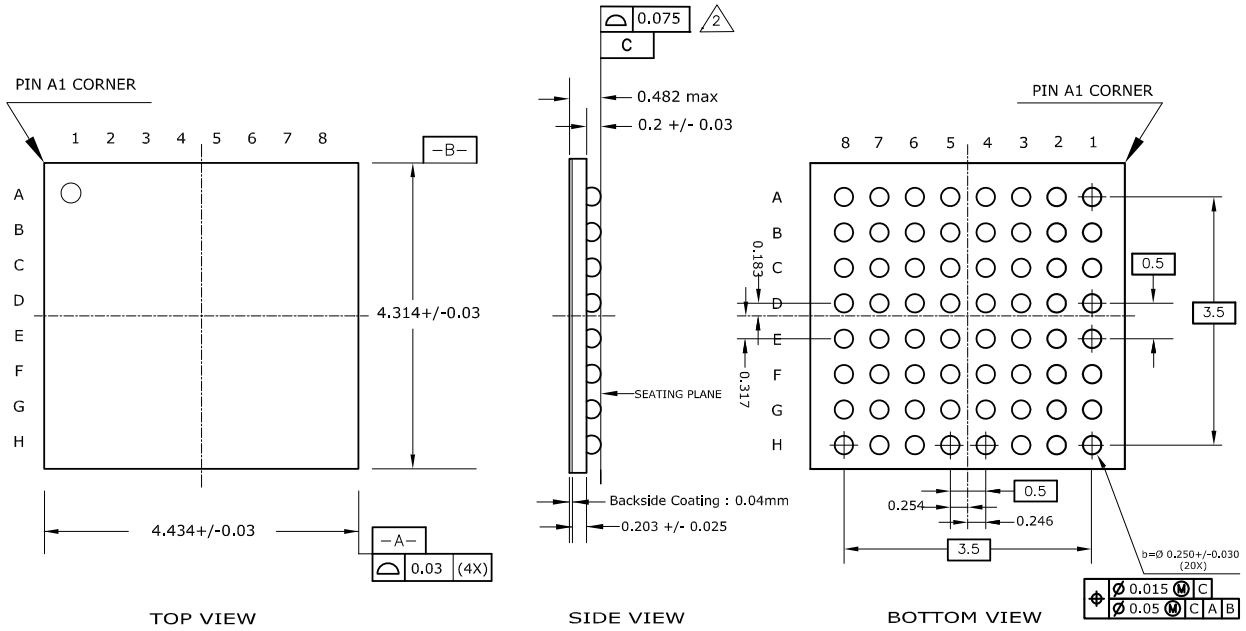
Table 43-6. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-7. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

Figure 43-3. WLCSP64 SAM4LC4/2 Package Drawing



COMMON DIMENSIONS
(Unit of Measure = mm)

BALL	SIGNAL	X COORD	Y COORD
A1	PB04	1.746	1.683
A2	GNDANA	1.246	1.683
A3	ADVREFP	0.746	1.683
A4	VDDANA	0.246	1.683
A5	PA09	-0.254	1.683
A6	CAPL	-0.754	1.683
A7	CAPH	-1.254	1.683
A8	PA12	-1.754	1.683
B1	PB03	1.746	1.183
B2	XIN32	1.246	1.183
B3	XOUT32	0.746	1.183
B4	PA08	0.246	1.183
B5	PB06	-0.254	1.183
B6	PA10	-0.754	1.183
B7	PA11	-1.254	1.183
B8	VLCD	-1.754	1.183
C1	VDDIN	1.746	0.683
C2	PB01	1.246	0.683
C3	PA05	0.746	0.683
C4	PA06	0.246	0.683
C5	PA07	-0.254	0.683
C6	PB07	-0.754	0.683

BALL	SIGNAL	X COORD	Y COORD
C7	PA13	-1.254	0.683
C8	BIAS1	-1.754	0.683
D1	VDDOUT	1.746	0.183
D2	PB00	1.246	0.183
D3	PA04	0.746	0.183
D4	PB05	0.246	0.183
D5	PB12	-0.254	0.183
D6	PB08	-0.754	0.183
D7	PA14	-1.254	0.183
D8	BIAS2	-1.754	0.183
E1	GNDIN	1.746	-0.317
E2	PA03	1.246	-0.317
E3	PB02	0.746	-0.317
E4	RESET_N	0.246	-0.317
E5	PB13	-0.254	-0.317
E6	PB09	-0.754	-0.317
E7	PA15	-1.254	-0.317
E8	GNDIO0	-1.754	-0.317
F1	VDDCORE	1.746	-0.817
F2	TCK	1.246	-0.817
F3	PA02	0.746	-0.817
F4	PB14	0.246	-0.817

BALL	SIGNAL	X COORD	Y COORD
F5	PA22	-0.254	-0.817
F6	PB10	-0.754	-0.817
F7	PA16	-1.254	-0.817
F8	VLCDIN	-1.754	-0.817
G1	GNDIO1	1.746	-1.317
G2	PA26	1.246	-1.317
G3	PA24	0.746	-1.317
G4	PA00	0.246	-1.317
G5	PA01	-0.254	-1.317
G6	PA19	-0.754	-1.317
G7	PA18	-1.254	-1.317
G8	PA17	-1.754	-1.317
H1	VDDIO1	1.746	-1.817
H2	PA25	1.246	-1.817
H3	PA23	0.746	-1.817
H4	PB15	0.246	-1.817
H5	PA21	-0.254	-1.817
H6	VDDIO0	-0.754	-1.817
H7	PA20	-1.254	-1.817
H8	PB11	-1.754	-1.817

Notes : 1. Dimension "b" is measured at the maximum ball diameter in a plane to the seating plane.

2. Applied to whole wafer.

Table 43-8. Device and Package Maximum Weight

14.8	mg

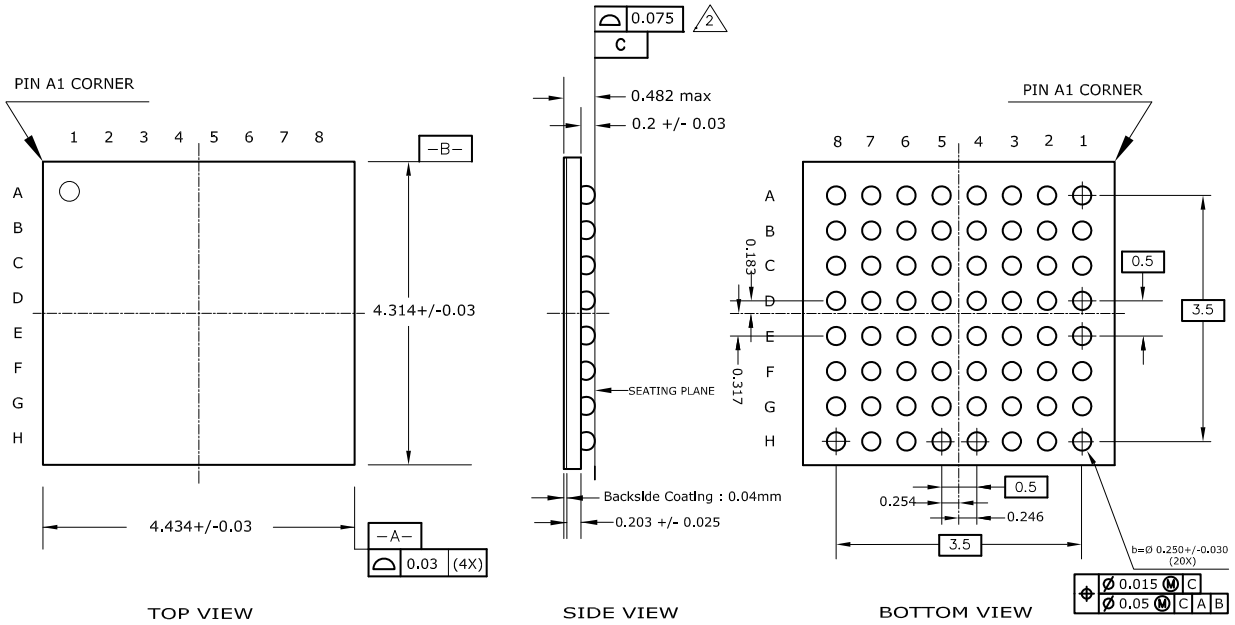
Table 43-9. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-10. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E1

Figure 43-4. WLCSP64 SAM4LS4/2 Package Drawing



COMMON DIMENSIONS
(Unit of Measure = mm)

BALL	SIGNAL	X COORD	Y COORD
A1	PB04	1.746	1.683
A2	GNDANA	1.246	1.683
A3	ADVREFP	0.746	1.683
A4	VDDANA	0.246	1.683
A5	PA09	-0.254	1.683
A6	PA28	-0.754	1.683
A7	PA27	-1.254	1.683
A8	PA12	-1.754	1.683
B1	PB03	1.746	1.183
B2	XIN32	1.246	1.183
B3	XOUT32	0.746	1.183
B4	PA08	0.246	1.183
B5	PB06	-0.254	1.183
B6	PA10	-0.754	1.183
B7	PA11	-1.254	1.183
B8	PA29	-1.754	1.183
C1	VDDIN	1.746	0.683
C2	PB01	1.246	0.683
C3	PA05	0.746	0.683
C4	PA06	0.246	0.683
C5	PA07	-0.254	0.683
C6	PB07	-0.754	0.683

BALL	SIGNAL	X COORD	Y COORD
C7	PA13	-1.254	0.683
C8	GNDIO0	-1.754	0.683
D1	VDDOUT	1.746	0.183
D2	PB00	1.246	0.183
D3	PA04	0.746	0.183
D4	PB05	0.246	0.183
D5	PB12	-0.254	0.183
D6	PB08	-0.754	0.183
D7	PA14	-1.254	0.183
D8	VLCDIN	-1.754	0.183
E1	GNDIN	1.746	-0.317
E2	PA03	1.246	-0.317
E3	PB02	0.746	-0.317
E4	RESET_N	0.246	-0.317
E5	PB13	-0.254	-0.317
E6	PB09	-0.754	-0.317
E7	PA15	-1.254	-0.317
E8	PA30	-1.754	-0.317
F1	VDDCORE	1.746	-0.817
F2	TCK	1.246	-0.817
F3	PA02	0.746	-0.817
F4	PB14	0.246	-0.817

BALL	SIGNAL	X COORD	Y COORD
F5	PA22	-0.254	-0.817
F6	PB10	-0.754	-0.817
F7	PA16	-1.254	-0.817
F8	PA31	-1.754	-0.817
G1	GNDIO1	1.746	-1.317
G2	PA26	1.246	-1.317
G3	PA24	0.746	-1.317
G4	PA00	0.246	-1.317
G5	PA01	-0.254	-1.317
G6	PA19	-0.754	-1.317
G7	PA18	-1.254	-1.317
G8	PA17	-1.754	-1.317
H1	VDDIO1	1.746	-1.817
H2	PA25	1.246	-1.817
H3	PA23	0.746	-1.817
H4	PB15	0.246	-1.817
H5	PA21	-0.254	-1.817
H6	VDDIO0	-0.754	-1.817
H7	PA20	-1.254	-1.817
H8	PB11	-1.754	-1.817

Notes : 1. Dimension "b" is measured at the maximum ball diameter in a plane to the seating plane.

2. Applied to whole wafer.

Table 43-11. Device and Package Maximum Weight

14.8	mg

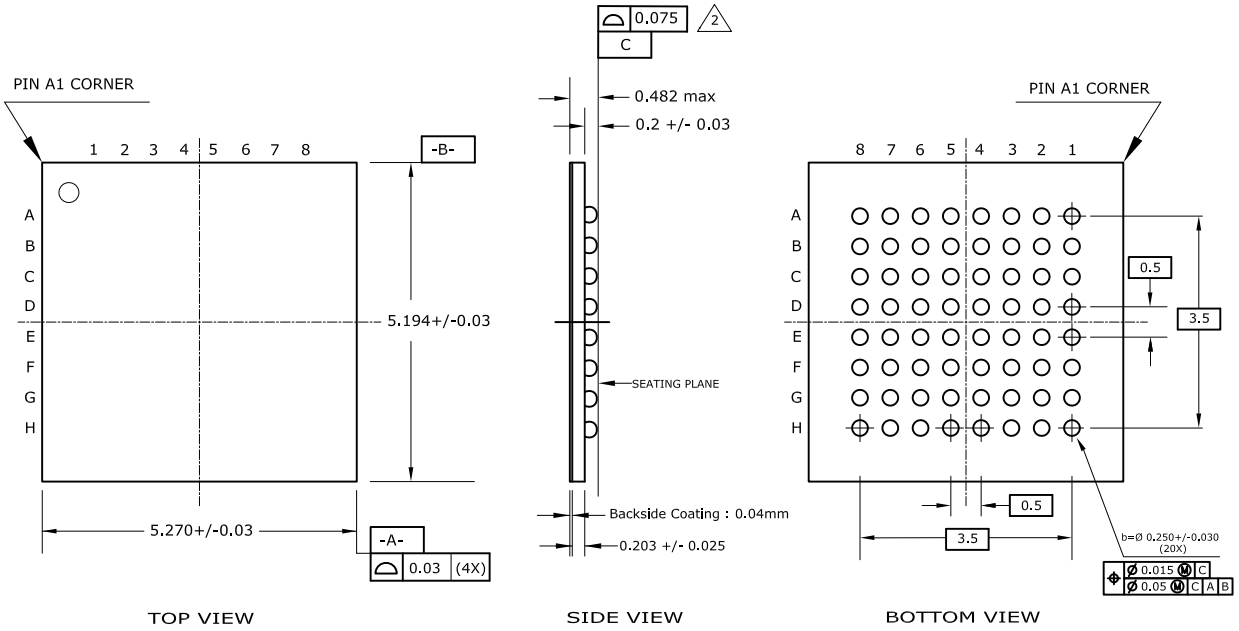
Table 43-12. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-13. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E1

Figure 43-5. WLCSP64 SAM4LC8 Package Drawing



COMMON DIMENSIONS
(Unit of Measure = mm)

BALL	SIGNAL	X COORD	Y COORD
A1	PB04	1.75	1.75
A2	GNDANA	1.75	1.25
A3	ADVREFP	1.75	0.75
A4	VDDANA	1.75	0.25
A5	PA09	1.75	-0.25
A6	CAPL	1.75	-0.75
A7	CAPH	1.75	-1.25
A8	PA12	1.75	-1.75
B1	PB03	1.25	1.75
B2	XIN32	1.25	1.25
B3	XOUT32	1.25	0.75
B4	PA08	1.25	0.25
B5	PB06	1.25	-0.25
B6	PA10	1.25	-0.75
B7	PA11	1.25	-1.25
B8	VLCD	1.25	-1.75
C1	VDDIN	0.75	1.75
C2	PB01	0.75	1.25
C3	PA05	0.75	0.75
C4	PA06	0.75	0.25
C5	PA07	0.75	-0.25
C6	PB07	0.75	-0.75

BALL	SIGNAL	X COORD	Y COORD
C7	PA13	0.75	-1.25
C8	BIAS1	0.75	-1.75
D1	VDDOUT	0.25	1.75
D2	PB00	0.25	1.25
D3	PA04	0.25	0.75
D4	PB05	0.25	0.25
D5	PB12	0.25	-0.25
D6	PB08	0.25	-0.75
D7	PA14	0.25	-1.25
D8	BIAS2	0.25	-1.75
E1	GNDIN	-0.25	1.75
E2	PA03	-0.25	1.25
E3	PB02	-0.25	0.75
E4	RESET_N	-0.25	0.25
E5	PB13	-0.25	-0.25
E6	PB09	-0.25	-0.75
E7	PA15	-0.25	-1.25
E8	GNDIO0	-0.25	-1.75
F1	VDDCORE	-0.75	1.75
F2	TCK	-0.75	1.25
F3	PA02	-0.75	0.75
F4	PB14	-0.75	0.25

BALL	SIGNAL	X COORD	Y COORD
F5	PA22	-0.75	-0.25
F6	PB10	-0.75	-0.75
F7	PA16	-0.75	-1.25
F8	VLCDIN	-0.75	-1.75
G1	GNDIO1	-1.25	1.75
G2	PA26	-1.25	1.25
G3	PA24	-1.25	0.75
G4	PA00	-1.25	0.25
G5	PA01	-1.25	-0.25
G6	PA19	-1.25	-0.75
G7	PA18	-1.25	-1.25
G8	PA17	-1.25	-1.75
H1	VDDIO1	-1.75	1.75
H2	PA25	-1.75	1.25
H3	PA23	-1.75	0.75
H4	PB15	-1.75	0.25
H5	PA21	-1.75	-0.25
H6	VDDIO0	-1.75	-0.75
H7	PA20	-1.75	-1.25
H8	PB11	-1.75	-1.75

Notes : 1. Dimension "b" is measured at the maximum ball diameter in a plane to the seating plane.

2. Applied to whole wafer.

Table 43-14. Device and Package Maximum Weight

14.8	mg

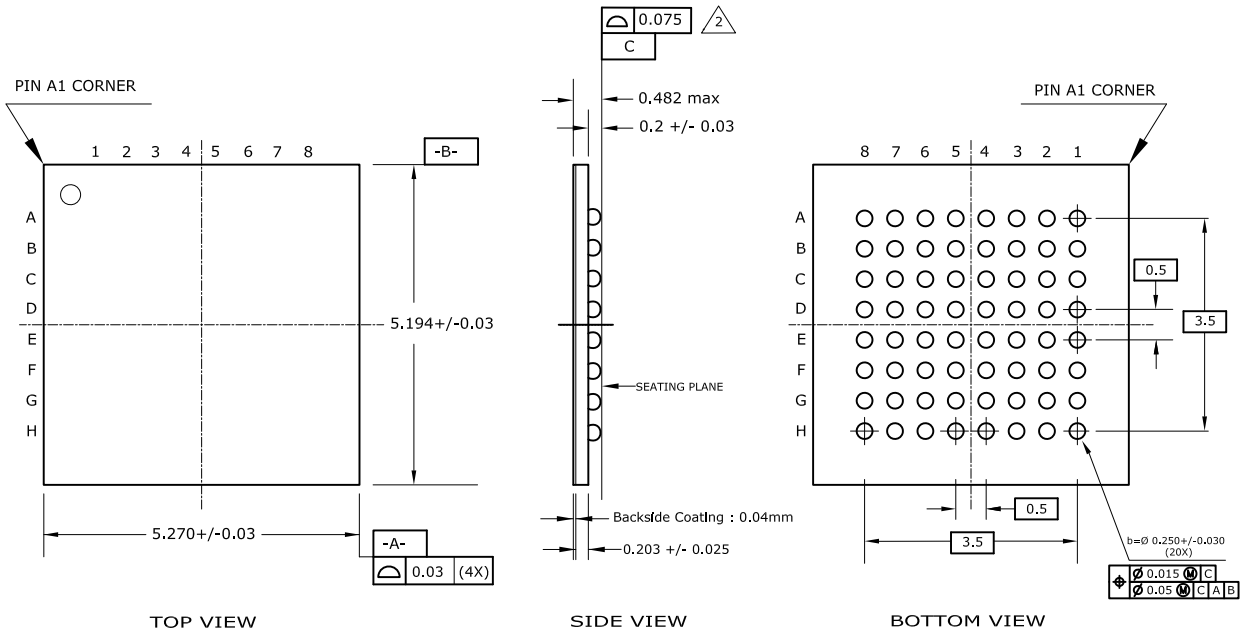
Table 43-15. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-16. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E1

Figure 43-6. WLCSP64 SAM4LS8 Package Drawing



COMMON DIMENSIONS
(Unit of Measure = mm)

BALL	SIGNAL	X COORD	Y COORD
A1	PB04	1.75	1.75
A2	GNDANA	1.75	1.25
A3	ADVREFP	1.75	0.75
A4	VDDANA	1.75	0.25
A5	PA09	1.75	-0.25
A6	PA28	1.75	-0.75
A7	PA27	1.75	-1.25
A8	PA12	1.75	-1.75
B1	PB03	1.25	1.75
B2	XIN32	1.25	1.25
B3	XOUT32	1.25	0.75
B4	PA08	1.25	0.25
B5	PB06	1.25	-0.25
B6	PA10	1.25	-0.75
B7	PA11	1.25	-1.25
B8	PA29	1.25	-1.75
C1	VDDIN	0.75	1.75
C2	PB01	0.75	1.25
C3	PA05	0.75	0.75
C4	PA06	0.75	0.25
C5	PA07	0.75	-0.25
C6	PB07	0.75	-0.75

BALL	SIGNAL	X COORD	Y COORD
C7	PA13	0.75	-1.25
C8	GNDIO0	0.75	-1.75
D1	VDDOUT	0.25	1.75
D2	PB00	0.25	1.25
D3	PA04	0.25	0.75
D4	PB05	0.25	0.25
D5	PB12	0.25	-0.25
D6	PB08	0.25	-0.75
D7	PA14	0.25	-1.25
D8	VLCDIN	0.25	-1.75
E1	GNDIN	-0.25	1.75
E2	PA03	-0.25	1.25
E3	PB02	-0.25	0.75
E4	RESET_N	-0.25	0.25
E5	PB13	-0.25	-0.25
E6	PB09	-0.25	-0.75
E7	PA15	-0.25	-1.25
E8	PA30	-0.25	-1.75
F1	VDDCORE	-0.75	1.75
F2	TCK	-0.75	1.25
F3	PA02	-0.75	0.75
F4	PB14	-0.75	0.25

BALL	SIGNAL	X COORD	Y COORD
F5	PA22	-0.75	-0.25
F6	PB10	-0.75	-0.75
F7	PA16	-0.75	-1.25
F8	PA31	-0.75	-1.75
G1	GNDIO1	-1.25	1.75
G2	PA26	-1.25	1.25
G3	PA24	-1.25	0.75
G4	PA00	-1.25	0.25
G5	PA01	-1.25	-0.25
G6	PA19	-1.25	-0.75
G7	PA18	-1.25	-1.25
G8	PA17	-1.25	-1.75
H1	VDDIO1	-1.75	1.75
H2	PA25	-1.75	1.25
H3	PA23	-1.75	0.75
H4	PB15	-1.75	0.25
H5	PA21	-1.75	-0.25
H6	VDDIO0	-1.75	-0.75
H7	PA20	-1.75	-1.25
H8	PB11	-1.75	-1.75

Notes : 1. Dimension "b" is measured at the maximum ball diameter in a plane to the seating plane.

2. Applied to whole wafer.

Table 43-17. Device and Package Maximum Weight

14.8	mg

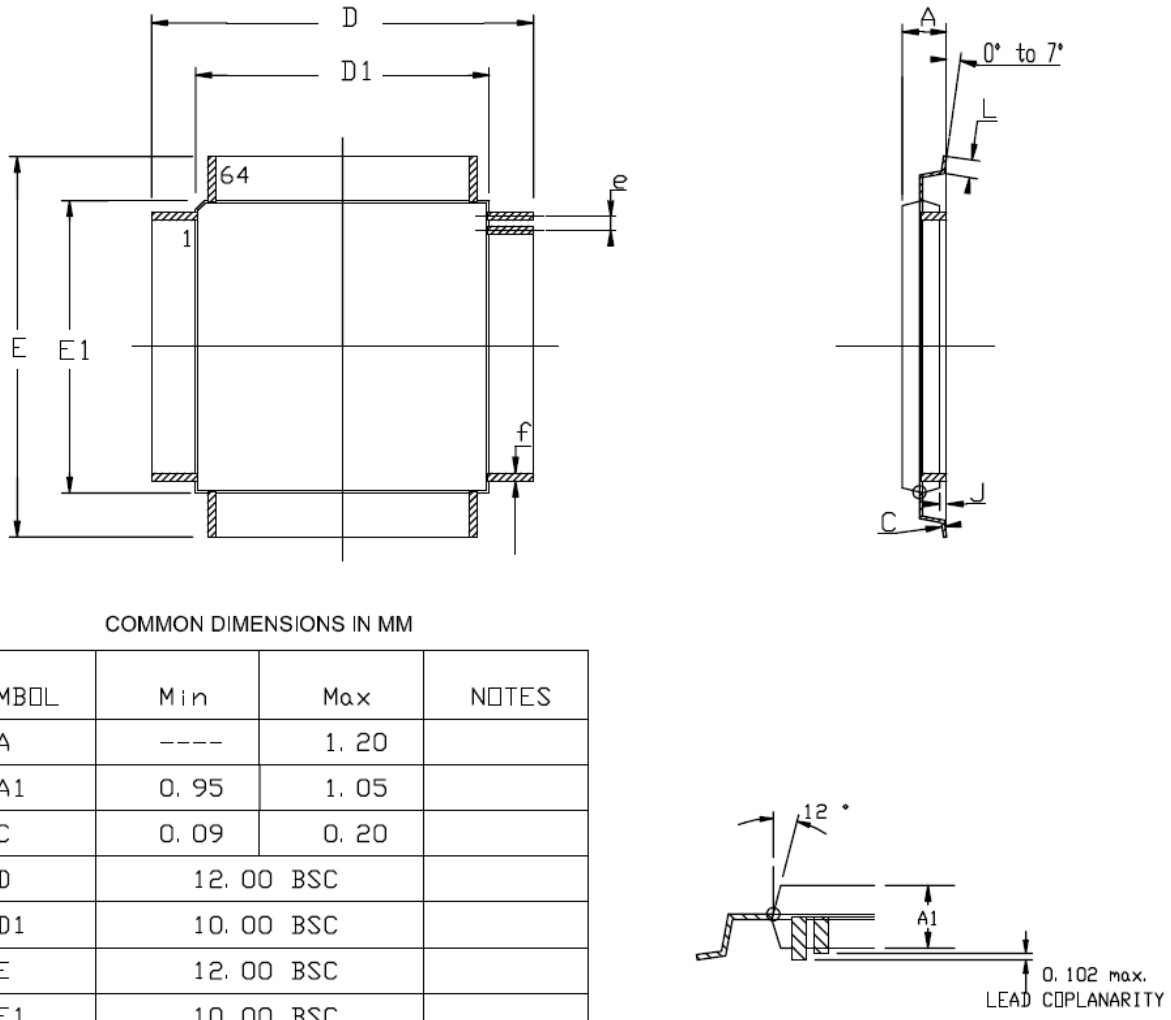
Table 43-18. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-19. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E1

Figure 43-7. TQFP-64 Package Drawing



COMMON DIMENSIONS IN MM

SYMBOL	Min	Max	NOTES
A	----	1.20	
A1	0.95	1.05	
C	0.09	0.20	
D	12.00 BSC		
D1	10.00 BSC		
E	12.00 BSC		
E1	10.00 BSC		
J	0.05	0.15	
L	0.45	0.75	
e	0.50 BSC		
f	0.17	0.27	

Table 43-20. Device and Package Maximum Weight

300	mg

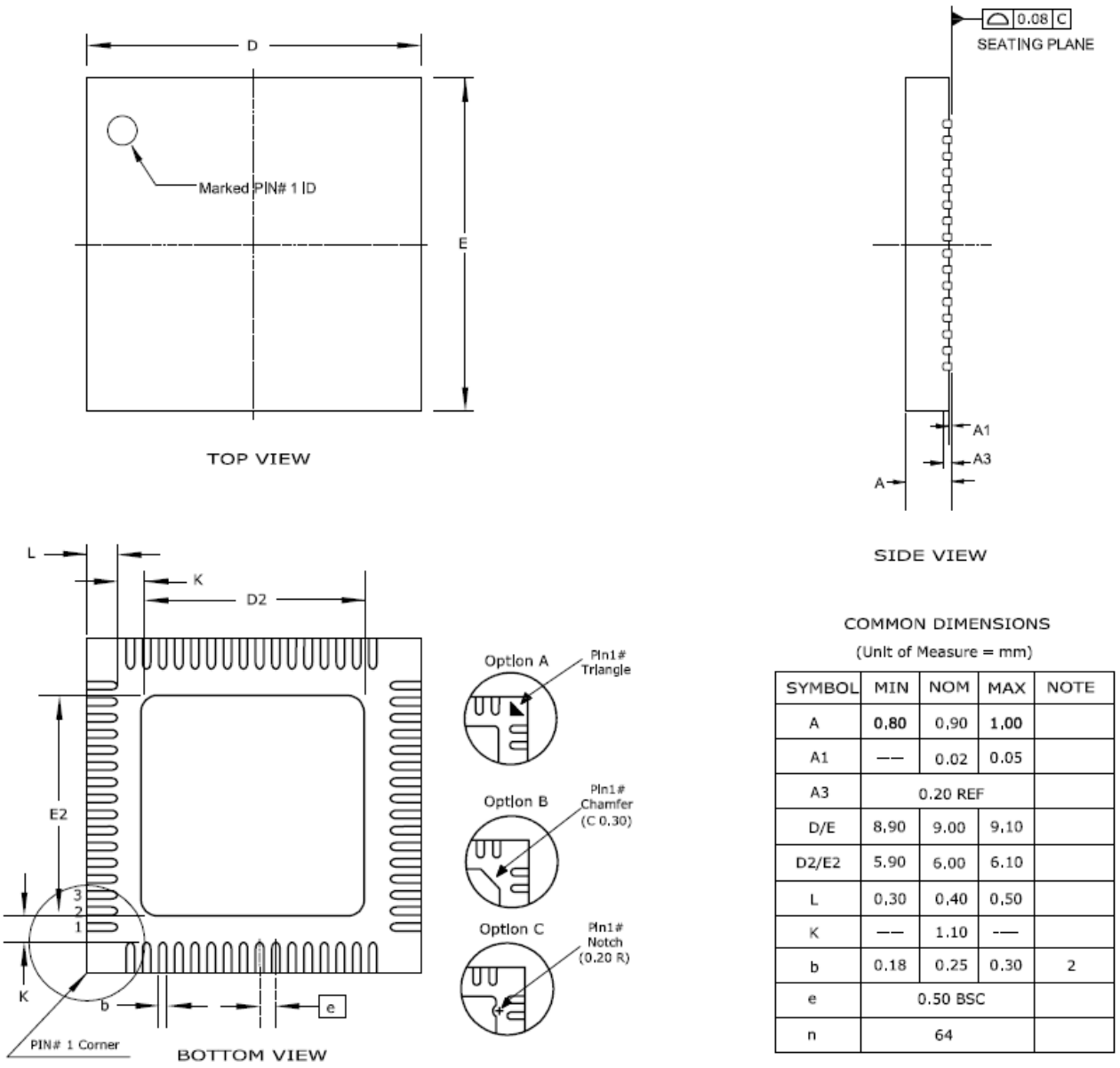
Table 43-21. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-22. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

Figure 43-8. QFN-64 Package Drawing



COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	—	0.02	0.05	
A3		0.20 REF		
D/E	8.90	9.00	9.10	
D2/E2	5.90	6.00	6.10	
L	0.30	0.40	0.50	
K	—	1.10	—	
b	0.18	0.25	0.30	2
e		0.50 BSC		
n		64		

Note: The exposed pad is not connected to anything internally, but should be soldered to ground to increase board level reliability.

Table 43-23. Device and Package Maximum Weight

200	mg

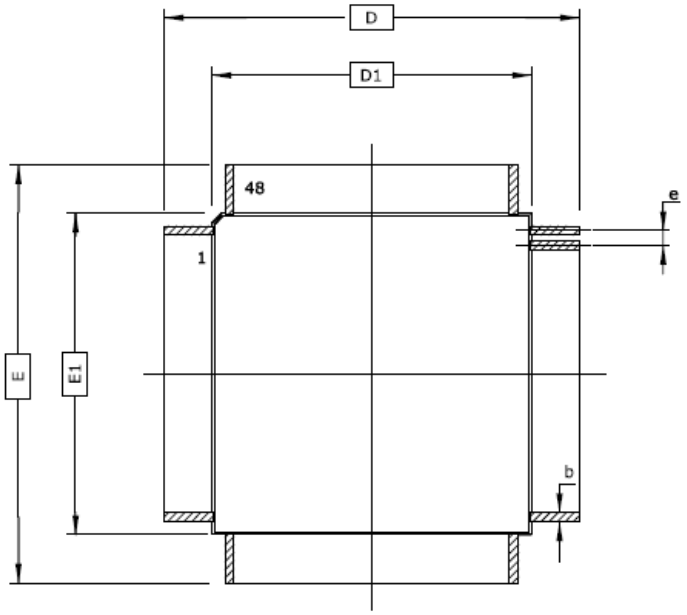
Table 43-24. Package Characteristics

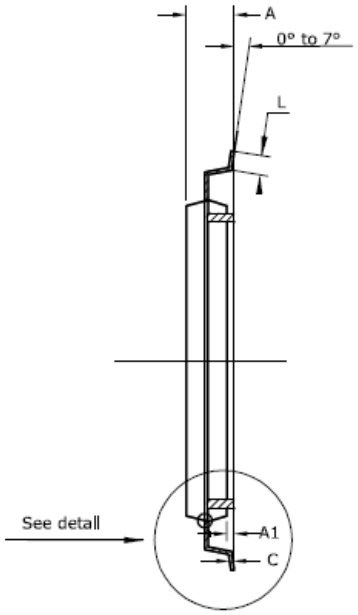
Moisture Sensitivity Level	MSL3

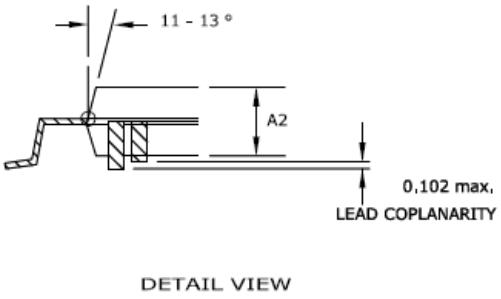
Table 43-25. Package Reference

JEDEC Drawing Reference	MO-220
JESD97 Classification	E3

Figure 43-9. TQFP-48 (ATSAM4LC4/2 and ATSAM4LS4/2 Only) Package Drawing







COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	---	---	1,20	
A1	0,05	---	0,15	
A2	0,95	---	1,05	
C	0,09	---	0,20	
D/E	9.00 BSC			
D1/E1	7.00 BSC			
L	0,45	---	0,75	
b	0,17	---	0,27	
e	0,50 BSC			

Table 43-26. Device and Package Maximum Weight

140	mg

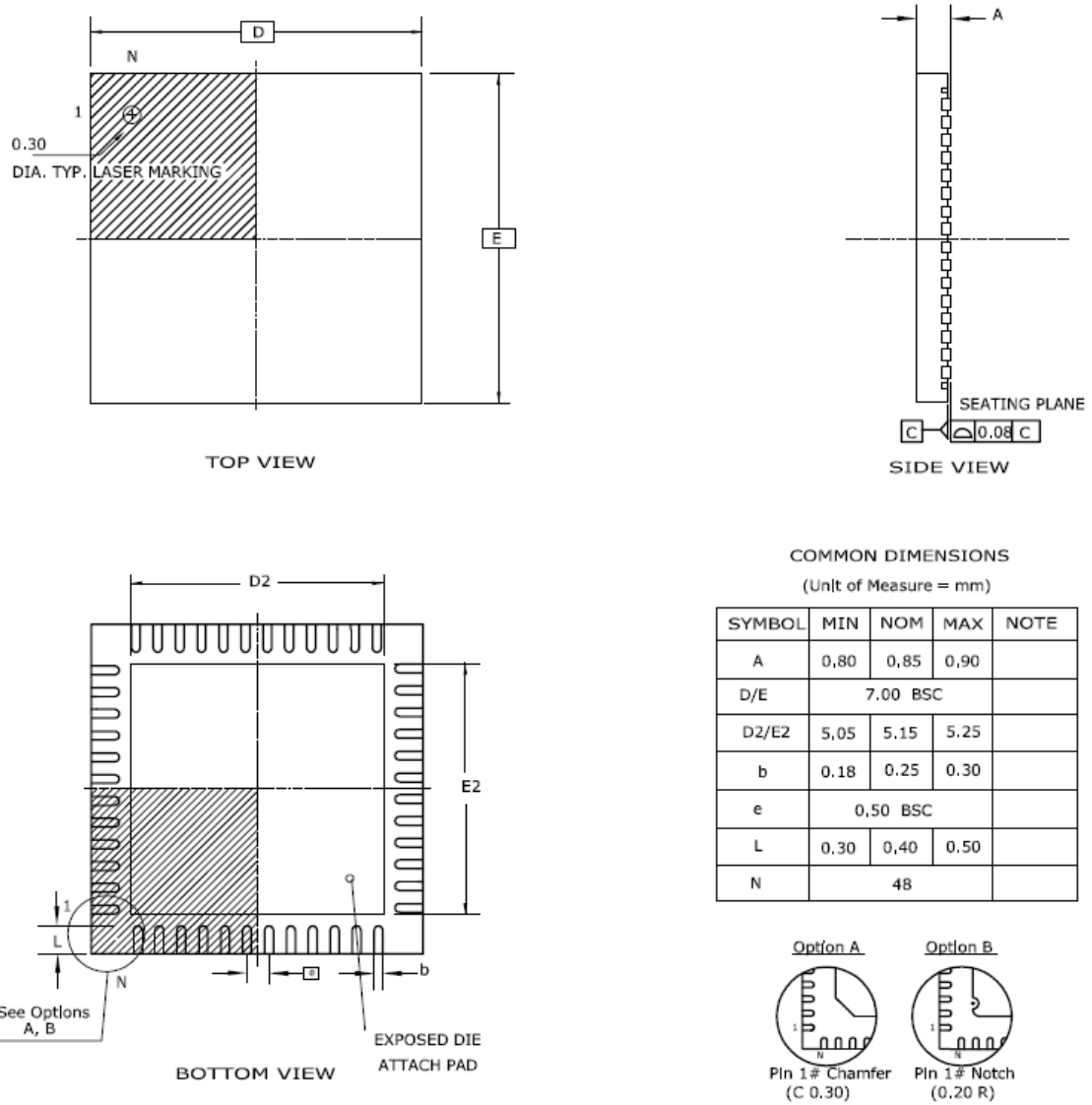
Table 43-27. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-28. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

Figure 43-10. QFN-48 Package Drawing for ATSAM4LC4/2 and ATSAM4LS4/2



SYMBOL	MIN	NOM	MAX	NOTE
A	0,80	0,85	0,90	
D/E		7.00 BSC		
D2/E2	5,05	5.15	5.25	
b	0.18	0.25	0.30	
e		0,50 BSC		
L	0.30	0,40	0.50	
N		48		

Note: The exposed pad is not connected to anything internally, but should be soldered to ground to increase board level reliability.

Table 43-29. Device and Package Maximum Weight

140	mg

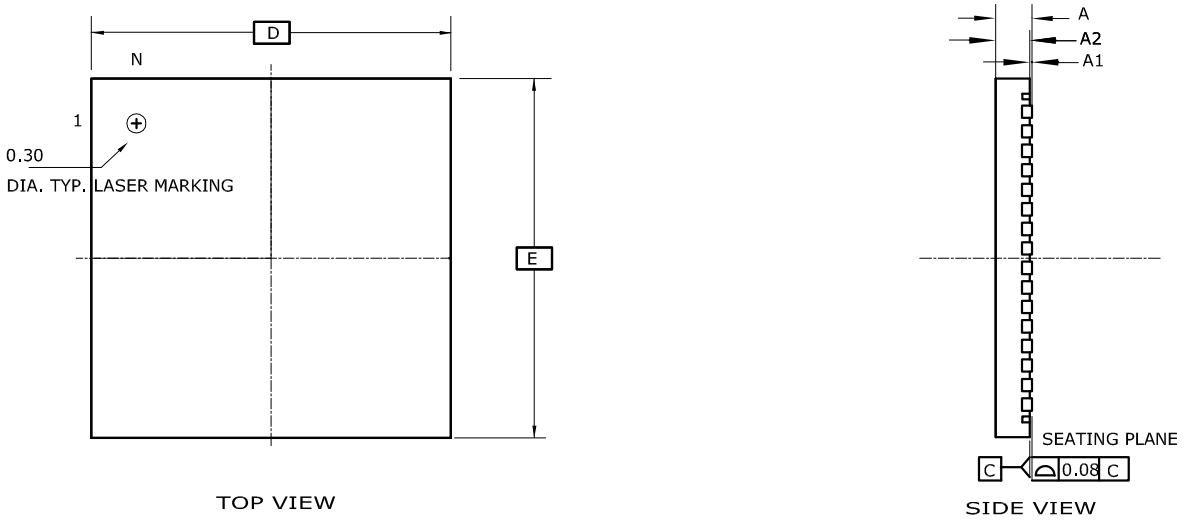
Table 43-30. Package Characteristics

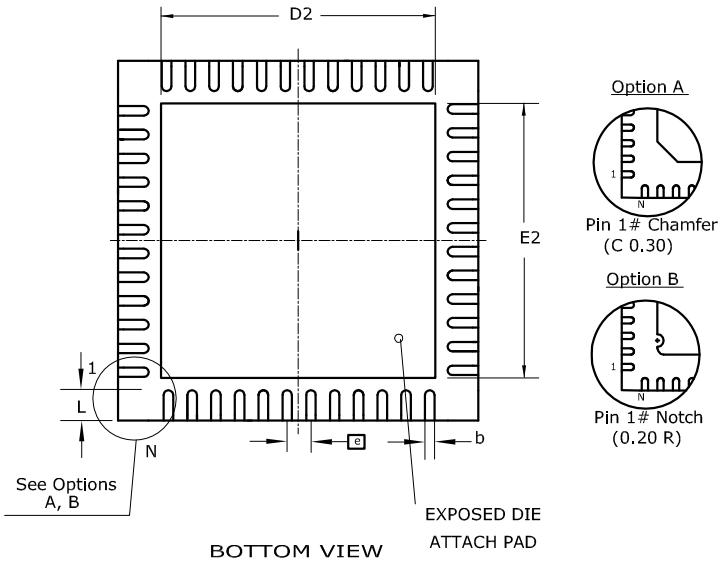
Moisture Sensitivity Level	MSL3

Table 43-31. Package Reference

JEDEC Drawing Reference	MO-220
JESD97 Classification	E3

Figure 43-11. QFN-48 Package Drawing for ATSAM4LC8 and ATSAM4LS8





COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.85	0.90	
A1	0.00	0.02	0.05	
A2	0.75	-----	1.00	
D/E	7 BSC			
D2/E2	5.50	5.60	5.70	
L	0.30	0.40	0.50	
b	0.17	-----	0.30	2
e	0.50 BSC			
n	48			

Note: The exposed pad is not connected to anything internally, but should be soldered to ground to increase board level reliability.

Table 43-32. Device and Package Maximum Weight

140	mg

Table 43-33. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 43-34. Package Reference

JEDEC Drawing Reference	MO-220
JESD97 Classification	E3

43.3 Soldering Profile

Table 43-35 gives the recommended soldering profile from J-STD-20.

Table 43-35. Soldering Profile

Profile Feature	Green Package
Average Ramp-up Rate (217°C to Peak)	3°C/s max
Preheat Temperature 175°C ±25°C	150-200°C
Time Maintained Above 217°C	60-150 s
Time within 5·C of Actual Peak Temperature	30 s
Peak Temperature Range	260°C
Ramp-down Rate	6°C/s max
Time 25·C to Peak Temperature	8 minutes max

A maximum of three reflow passes is allowed per component.

44. Ordering Information

Table 44-1. ATSAM4LC8 Sub Serie Ordering Information

Ordering Code	Flash (Kbytes)	RAM (Kbytes)	Package	Conditioning	Package Type	Temperature Operating Range
ATSAM4LC8CA-AU	512	64	TQFP100	Tray	Green	Industrial -40°C to 85°C
ATSAM4LC8CA-AUR				Reel		
ATSAM4LC8CA-CFU			VFBGA100	Tray		
ATSAM4LC8CA-CFUR				Reel		
ATSAM4LC8BA-AU			TQFP64	Tray		
ATSAM4LC8BA-AUR				Reel		
ATSAM4LC8BA-MU			QFN64	Tray		
ATSAM4LC8BA-MUR				Reel		
ATSAM4LC8BA-UUR			WLCSP64	Reel		
ATSAM4LC8AA-MU			QFN48	Tray		
ATSAM4LC8AA-MUR				Reel		

Table 44-2. ATSAM4LC4 Sub Serie Ordering Information

Ordering Code	Flash (Kbytes)	RAM (Kbytes)	Package	Conditioning	Package Type	Temperature Operating Range
ATSAM4LC4CA-AU-ES	256	32	TQFP100	ES	Green	N/A
ATSAM4LC4CA-AU				Tray		Industrial -40°C to 85°C
ATSAM4LC4CA-AUR				Reel		
ATSAM4LC4CA-CFU			VFBGA100	Tray		Industrial -40°C to 85°C
ATSAM4LC4CA-CFUR				Reel		
ATSAM4LC4BA-AU-ES			TQFP64	ES		N/A
ATSAM4LC4BA-AU				Tray		Industrial -40°C to 85°C
ATSAM4LC4BA-AUR				Reel		
ATSAM4LC4BA-MU-ES			QFN64	ES		N/A
ATSAM4LC4BA-MU				Tray		Industrial -40°C to 85°C
ATSAM4LC4BA-MUR				Reel		
ATSAM4LC4BA-UUR			WLCSP64	Reel		Industrial -40°C to 85°C
ATSAM4LC4AA-AU-ES			TQFP48	ES		N/A
ATSAM4LC4AA-AU				Tray		Industrial -40°C to 85°C
ATSAM4LC4AA-AUR				Reel		
ATSAM4LC4AA-MU-ES			QFN48	ES		N/A
ATSAM4LC4AA-MU				Tray		Industrial -40°C to 85°C
ATSAM4LC4AA-MUR				Reel		

Table 44-3. ATSAM4LC2 Sub Serie Ordering Information

Ordering Code	Flash (Kbytes)	RAM (Kbytes)	Package	Conditioning	Package Type	Temperature Operating Range
ATSAM4LC2CA-AU	128	32	TQFP100	Tray	Green	Industrial -40°C to 85°C
ATSAM4LC2CA-AUR				Reel		
ATSAM4LC2CA-CFU			VFBGA100	Tray		
ATSAM4LC2CA-CFUR				Reel		
ATSAM4LC2BA-AU			TQFP64	Tray		
ATSAM4LC2BA-AUR				Reel		
ATSAM4LC2BA-MU			QFN64	Tray		
ATSAM4LC2BA-MUR				Reel		
ATSAM4LC2BA-UUR			WLCSP64	Reel		
ATSAM4LC2AA-AU			TQFP48	Tray		
ATSAM4LC2AA-AUR				Reel		
ATSAM4LC2AA-MU			QFN48	Tray		
ATSAM4LC2AA-MUR				Reel		

Table 44-4. ATSAM4LS8 Sub Serie Ordering Information

Ordering Code	Flash (Kbytes)	RAM (Kbytes)	Package	Conditioning	Package Type	Temperature Operating Range
ATSAM4LS8CA-AU	512	64	TQFP100	Tray	Green	Industrial -40°C to 85°C
ATSAM4LS8CA-AUR				Reel		
ATSAM4LS8CA-CFU			VFBGA100	Tray		
ATSAM4LS8CA-CFUR				Reel		
ATSAM4LS8BA-AU			TQFP64	Tray		
ATSAM4LS8BA-AUR				Reel		
ATSAM4LS8BA-MU			QFN64	Tray		
ATSAM4LS8BA-MUR				Reel		
ATSAM4LS8BA-UUR			WLCSP64	Reel		
ATSAM4LS8AA-MU			QFN48	Tray		
ATSAM4LS8AA-MUR				Reel		

Table 44-5. ATSAM4LS4 Sub Serie Ordering Information

Ordering Code	Flash (Kbytes)	RAM (Kbytes)	Package	Conditioning	Package Type	Temperature Operating Range
ATSAM4LS4CA-AU-ES	256	32	TQFP100	ES	Green	N/A
ATSAM4LS4CA-AU				Tray		Industrial -40°C to 85°C
ATSAM4LS4CA-AUR				Reel		
ATSAM4LS4CA-CFU			VFBGA100	Tray		Industrial -40°C to 85°C
ATSAM4LS4CA-CFUR				Reel		
ATSAM4LS4BA-AU-ES			TQFP64	ES		N/A
ATSAM4LS4BA-AU				Tray		Industrial -40°C to 85°C
ATSAM4LS4BA-AUR				Reel		
ATSAM4LS4BA-MU-ES			QFN64	ES		N/A
ATSAM4LS4BA-MU				Tray		Industrial -40°C to 85°C
ATSAM4LS4BA-MUR				Reel		
ATSAM4LS4BA-UUR			WLCSP64	Reel		Industrial -40°C to 85°C
ATSAM4LS4AA-AU-ES			TQFP48	ES		N/A
ATSAM4LS4AA-AU				Tray		Industrial -40°C to 85°C
ATSAM4LS4AA-AUR				Reel		
ATSAM4LS4AA-MU-ES			QFN48	ES		N/A
ATSAM4LS4AA-MU				Tray		Industrial -40°C to 85°C
ATSAM4LS4AA-MUR				Reel		

Table 44-6. ATSAM4LS2 Sub Serie Ordering Information

Ordering Code	Flash (Kbytes)	RAM (Kbytes)	Package	Conditioning	Package Type	Temperature Operating Range
ATSAM4LS2CA-AU	128	32	TQFP100	Tray	Green	Industrial -40°C to 85°C
ATSAM4LS2CA-AUR				Reel		
ATSAM4LS2CA-CFU			VFBGA100	Tray		
ATSAM4LS2CA-CFUR				Reel		
ATSAM4LS2BA-AU			TQFP64	Tray		
ATSAM4LS2BA-AUR				Reel		
ATSAM4LS2BA-MU			QFN64	Tray		
ATSAM4LS2BA-MUR				Reel		
ATSAM4LS2BA-UUR			WLCSP64	Reel		
ATSAM4LS2AA-AU			TQFP48	Tray		
ATSAM4LS2AA-AUR				Reel		
ATSAM4LS2AA-MU			QFN48	Tray		
ATSAM4LS2AA-MUR				Reel		

45. Errata

45.1 ATSAM4L4 /2 Rev. B & ATSAM4L8 Rev. A

45.1.1 General

PS2 mode is not supported by Engineering Samples
PS2 mode support is supported only by parts with calibration version higher than 0.
Fix/Workaround
The calibration version can be checked by reading a 32-bit word at address 0x0080020C. The calibration version bitfield is 4-bit wide and located from bit 4 to bit 7 in this word. Any value higher than 0 ensures that the part supports the PS2 mode

45.1.2 SCIF

PLLCOUNT value larger than zero can cause PLLEN glitch
Initializing the PLLCOUNT with a value greater than zero creates a glitch on the PLLEN signal during asynchronous wake up.
Fix/Workaround
The lock-masking mechanism for the PLL should not be used.
The PLLCOUNT field of the PLL Control Register should always be written to zero.

45.1.3 WDT

WDT Control Register does not have synchronization feedback
When writing to the Timeout Prescale Select (PSEL), Time Ban Prescale Select (TBAN), Enable (EN), or WDT Mode (MODE) fieldss of the WDT Control Register (CTRL), a synchronizer is started to propagate the values to the WDT clcok domain. This synchronization takes a finite amount of time, but only the status of the synchronization of the EN bit is reflected back to the user. Writing to the synchronized fields during synchronization can lead to undefined behavior.
Fix/Workaround
-When writing to the affected fields, the user must ensure a wait corresponding to 2 clock cycles of both the WDT peripheral bus clock and the selected WDT clock source.
-When doing writes that changes the EN bit, the EN bit can be read back until it reflects the written value.

45.1.4 SPI

SPI data transfer hangs with CSR0.CSAAT==1 and MR.MODFDIS==0
When CSR0.CSAAT==1 and mode fault detection is enabled (MR.MODFDIS==0), the SPI module will not start a data transfer.
Fix/Workaround
Disable mode fault detection by writing a one to MR.MODFDIS.

SPI disable does not work in SLAVE mode
SPI disable does not work in SLAVE mode.

Fix/Workaround
Read the last received data, then perform a software reset by writing a one to the Software Reset bit in the Control Register (CR.SWRST).

Disabling SPI has no effect on the SR.TDRE bit
Disabling SPI has no effect on the SR.TDRE bit whereas the write data command is filtered when SPI is disabled. Writing to TDR when SPI is disabled will not clear SR.TDRE. If SPI is disabled during a PDCA transfer, the PDCA will continue to write data to TDR until its buffer is empty, and this data will be lost.
Fix/Workaround
Disable the PDCA, add two NOPs, and disable the SPI. To continue the transfer, enable the SPI and PDCA.

SPI bad serial clock generation on 2nd chip_select when SCBR=1, CPOL=1, and NCPHA=0
When multiple chip selects (CS) are in use, if one of the baudrates equal 1 while one (CSRn.SCBR=1) of the others do not equal 1, and CSRn.CPOL=1 and CSRn.NCPHA=0, then an additional pulse will be generated on SCK.
Fix/Workaround
When multiple CS are in use, if one of the baudrates equals 1, the others must also equal 1 if CSRn.CPOL=1 and CSRn.NCPHA=0.

45.1.5 TC

Channel chaining skips first pulse for upper channel
When chaining two channels using the Block Mode Register, the first pulse of the clock between the channels is skipped.
Fix/Workaround
Configure the lower channel with RA = 0x1 and RC = 0x2 to produce a dummy clock cycle for the upper channel. After the dummy cycle has been generated, indicated by the SR.CPCS bit, reconfigure the RA and RC registers for the lower channel with the real values.

45.1.6 USBC

In USB host mode, entering suspend mode for low speed device can fail when the USB freeze (USBCON.FRZCLK=1) is done just after UHCON.SOFE=0.
Fix/Workaround
When entering suspend mode (UHCON.SOFE is cleared), check that USBFSM.DRDSTATE is not equal to three before freezing the clock (USBCON.FRZCLK=1).

In USB host mode, the asynchronous attach detection (UDINT.HWUPI) can fail when the USB clock freeze (USBCON.FRZCLK=1) is done just after setting the USB-STA.VBUSRQ bit.
Fix/Workaround
After setting USBSTA.VBUSRQ bit, wait until the USBFSM register value is ‘A_WAIT_BCON’ before setting the USBCON.FRZCLK bit.

45.1.7 FLASHCALW

Corrupted data in flash may happen after flash page write operations.

After a flash page write operation, reading (data read or code fetch) in flash may fail. This may lead to an expecption or to others errors derived from this corrupted read access.

Fix/Workaround

Before any flash page write operation, each 64-bit doublewords write in the page buffer must preceded by a 64-bit doublewords write in the page buffer with 0xFFFFFFFF_FFFFFFFF content at any address in the page. Note that special care is required when loading page buffer, refer to Section 14.5.8 ”Page Buffer Operations” on page 271.

46. Datasheet Revision History

Note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

46.1 Rev. A – 09/12

1. Initial revision.

46.2 Rev. B – 10/12

1. Fixed ordering code
2. Changed BOD18CTRL and BOD33CTRL ACTION field from “Reserved” to ‘No action”

46.3 Rev. C – 02/13

1. Fixed ball pitch for VFBGA100 package
2. Added VFBGA100 and WLCSP64 pinouts
3. Added Power Scaling Mode 2 for high frequency support
4. Minor update on several modules chapters
5. Major update on Electrical characteristics
6. Updated errata
7. Fixed GPIO multiplexing pin numbers

46.4 Rev. D – 03/13

1. Removed WLCSP package information
2. Added errata text for detecting whether a part supports PS2 mode or not
3. Removed temperature sensor feature (not supported by production flow)
4. Fixed MUX selection on Positive ADC input channel table
5. Added information about TWI instances capabilities
6. Added some details on errata Corrupted data in flash may happen after flash page write operations.1193

46.5 Rev. E – 07/13

1. Added ATSAM4L8 derivatives and WLCSP packages for ATSAM4L4/2
2. Added operating conditions details in Electrical Characteristics Chapter
3. Fixed “Supply Rise Rates and Order”
4. Added number of USART available in sub-series
5. Fixed IO line considerations for USB pins
6. Removed useless information about CPU local bus which is not implemented
7. Removed useless information about Modem support which is not implemented
8. Added information about unsupported features in Power Scaling mode 1
9. Fixed SPI timings

46.6 Rev. F– 12/13

1. Fixed table 3-6 - TDI is connected to pin G3 in WLCSP package
2. Changed table 42-48 -ADCIFE Electricals in unipolar mode : PSRR & DC supply current typical values
3. Fixed SPI timing characteristics
4. Fixed BOD33 typical step size value

46.7 Rev. G– 03/14

1. Added WLCSP64 packages for SAM4LC8 and SAM4LS8 sub-series
2. Removed unsuppported SWAP feature in LCD module
3. Added mnimal value for ADC Reference range

46.8 Rev. H– 11/16

1. Fixed AESA configuration in Overview chapter for SAM4LS sub-series

Table of Contents

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
16F, Shin Osaki Kangyo Bldg.
1-6-4 Osaka Shinagawa-ku
Tokyo 104-0032
JAPAN
Tel: (+81) 3-6417-0300
Fax: (+81) 3-6417-0370

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Atmel:

SAM4LC4AA-MU SAM4LC4BA-MU SAM4LC4CA-CFU SAM4LS4AA-MU SAM4LS4BA-MU SAM4LS4CA-CFU ATSAM4LC8BA-UUR ATSAM4LC8CA-CFU ATSAM4LS4AA-AUR ATSAM4LC2CA-AU ATSAM4LS4CA-AU ATSAM4LC4BA-UUR ATSAM4LC8BA-AU ATSAM4LS8BA-AU ATSAM4LS4CA-CFUR ATSAM4LC4BA-AU ATSAM4LC8BA-MUR ATSAM4LS4AA-AU ATSAM4LC2AA-AU ATSAM4LC2CA-CFU ATSAM4LS2AA-AUR ATSAM4LS4BA-AUR ATSAM4LS8BA-UUR ATSAM4LS4CA-CFU ATSAM4LC4AA-AUR ATSAM4LC8BA-MU ATSAM4LC8CA-CFUR ATSAM4LS4CA-AUR ATSAM4LC8CA-AU ATSAM4LC4BA-AUR ATSAM4LC8BA-AUR ATSAM4LS4BA-AU ATSAM4LS2AA-AU ATSAM4LC8AA-MU ATSAM4LS2BA-AUR ATSAM4LS8BA-MU ATSAM4LC2AA-AUR ATSAM4LS4BA-UUR ATSAM4LC2BA-AU ATSAM4LS2BA-AU ATSAM4LC4CA-AUR ATSAM4LC4CA-AU ATSAM4LC2CA-AUR ATSAM4LS2CA-CFUR ATSAM4LC4AA-AU