Atmel SAM D20J / SAM D20G / SAM D20E

ARM-Based Microcontroller

PRELIMINARY DATASHEET

Description

The Atmel® SAM D20 is a series of low-power microcontrollers using the 32-bit ARM® Cortex®-M0+ processor, and ranging from 32- to 64-pins with up to 256KB Flash and 32KB of SRAM. The SAM D20 devices operate at a maximum frequency of 48MHz and reach 2.14 Coremark/MHz. They are designed for simple and intuitive migration with identical peripheral modules, hex compatible code, identical linear address map and pin compatible migration paths between all devices in the product series. All devices include intelligent and flexible peripherals, Atmel Event System for inter-peripheral signaling, and support for capacitive touch button, slider and wheel user interfaces.

The Atmel SAM D20 devices provide the following features: In-system programmable Flash, eight-channel Event System, programmable interrupt controller, up to 52 programmable I/O pins, 32-bit real-time clock and calendar, up to eight 16-bit Timer/Counters (TC), where each TC can be configured to perform frequency and waveform generation, accurate program execution timing or input capture with time and frequency measurement of digital signals. The TCs can operate in 8- or 16-bit mode, and two TCs can be cascaded to form a 32-bit TC. The series provide up to six Serial Communication Modules (SERCOM) that each can be configured to act as an USART, UART, SPI and I^2C up to 400kHz; up to twenty-channel 350ksps 12-bit ADC with programmable gain and optional oversampling and decimation supporting up to 16-bit resolution, one 10-bit 350ksps DAC, two analog comparators with window mode, Peripheral Touch Controller supporting up to 256 buttons, sliders, wheels, and proximity sensing; programmable Watchdog Timer, brown-out detector and power-on reset, and two-pin Serial Wire Debug (SWD) program and debug interface.

All devices have accurate and low-power external and internal oscillators. All oscillators can be used as a source for the system clock. Different clock domains can be independently configured to run at different frequencies, enabling power saving by running each peripheral at its optimal clock frequency, and thus maintaining a high CPU frequency while reducing power consumption.

The SAM D20 devices have two software-selectable sleep modes, idle and standby. In idle mode the CPU is stopped while all other functions can be kept running. In standby all clocks and functions are stopped expect those selected to continue running. The device supports SleepWalking, which is the module's ability to wake itself up and wake up its own clock, and hence perform predefined tasks without waking up the CPU. The CPU can then be only woken on a need basis, e.g. a threshold is crossed or a result is ready. The Event System supports synchronous and asynchronous events, allowing peripherals to receive, react to and send events even in standby mode.

The Flash program memory can be reprogrammed in-system through the SWD interface. The same interface can be used for non-intrusive on-chip debug of application code. A boot loader running in the device can use any communication interface to download and upgrade the application program in the Flash memory.

The Atmel SAM D20 devices are supported with a full suite of program and system development tools, including C compilers, macro assemblers, program debugger/simulators, programmers and evaluation kits.

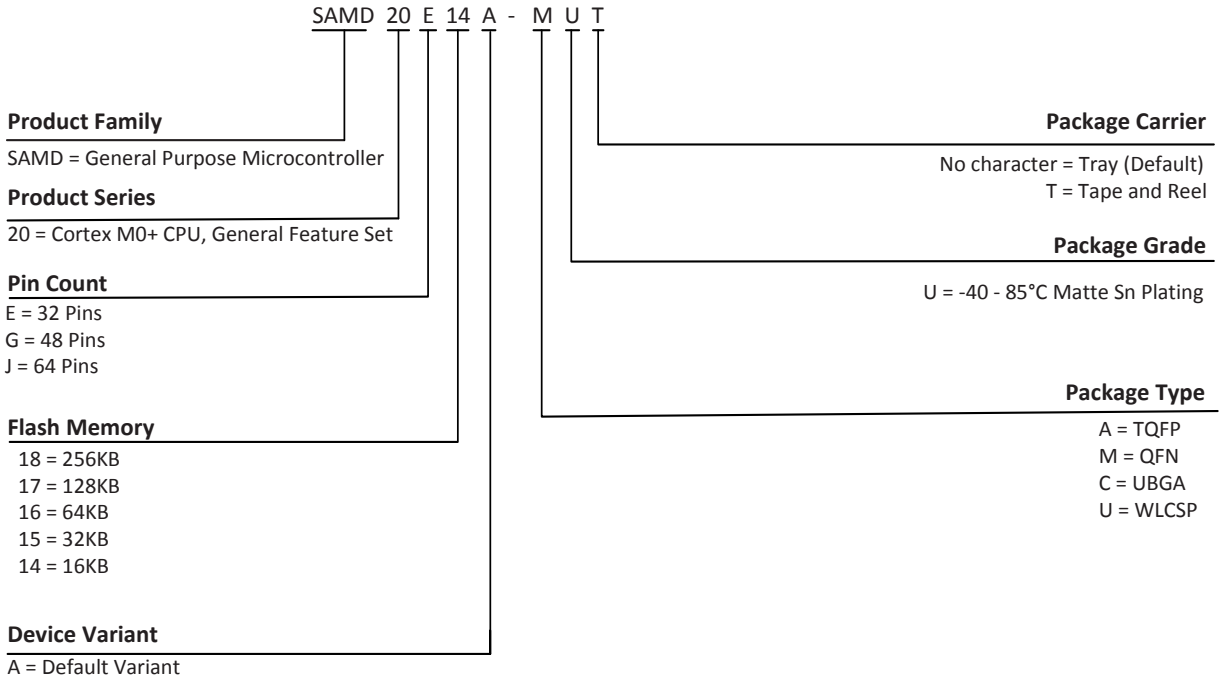
Features

- Processor
  - ARM Cortex-M0+ CPU running at up to 48MHz
    - Single-cycle hardware multiplier
- Memories
  - 16/32/64/128/256KB in-system self-programmable flash
  - 2/4/8/16/32KB SRAM
- System
  - Power-on reset (POR) and brown-out detection (BOD)
  - Internal and external clock options with 48MHz Digital Frequency Locked Loop (DFLL48M)
  - External Interrupt Controller (EIC)
  - 16 external interrupts
  - One non-maskable interrupt
  - Two-pin Serial Wire Debug (SWD) programming, test and debugging interface
- Low Power
  - Idle and standby sleep modes
  - SleepWalking peripherals
- Peripherals
  - 8-channel Event System
  - Up to eight 16-bit Timer/Counters (TC), configurable as either:
    - One 16-bit TC with compare/capture channels
    - One 8-bit TC with compare/capture channels
    - One 32-bit TC with compare/capture channels, by using two TCs
  - 32-bit Real Time Counter (RTC) with clock/calendar function
  - Watchdog Timer (WDT)
  - CRC-32 generator
  - Up to six Serial Communication Interfaces (SERCOM), each configurable to operate as either:
    - USART with full-duplex and single-wire half-duplex configuration
    - I^2C up to 400kHz
    - SPI
  - One 12-bit, 350ksps Analog-to-Digital Converter (ADC) with up to 20 channels
    - Differential and single-ended channels
    - 1/2x to 16x gain stage
    - Automatic offset and gain error compensation
    - Oversampling and decimation in hardware to support 13-, 14-, 15- or 16-bit resolution
  - 10-bit, 350ksps Digital-to-Analog Converter (DAC)
  - Two Analog Comparators (AC) with window compare function
  - Peripheral Touch Controller (PTC)
    - 256-Channel capacitive touch and proximity sensing
- I/O
  - Up to 52 programmable I/O pins
- Packages
  - 64-pin TQFP, QFN
  - 48-pin TQFP, QFN
  - 32-pin TQFP, QFN
- Operating Voltage
  - 1.62V – 3.63V
- Power Consumption
  - Down to 70µA/MHz in active mode
  - Down to 6µA running the Peripheral Touch Controller

1. Configuration Summary

	SAM D20J	SAM D20G	SAM D20E
Number of pins	64	48	32
General Purpose I/O-pins (GPIOs)	52	38	26
Flash	256/128/64/32/16KB	256/128/64/32/16KB	256/128/64/32/16KB
SRAM	32/16/8/4/2KB	32/16/8/4/2KB	32/16/8/4/2KB
Timer Counter (TC) instances	8	6	6
Waveform output channels per Timer Counter instance	2	2	2
Serial Communication Interface (SERCOM) instances	6	6	4
Analog-to-Digital Converter (ADC) channels	20	14	10
Analog Comparators (AC)	2	2	2
Digital-to-Analog Converter (DAC) instances	1	1	1
Real-Time Counter (RTC)	Yes	Yes	Yes
RTC alarms	1	1	1
RTC compare values	1 32-bit value or 2 16-bit values	1 32-bit value or 2 16-bit values	1 32-bit value or 2 16-bit values
External Interrupt lines	16	16	16
Peripheral Touch Controller (PTC) X and Y lines	16x16	12x10	10x6
Maximum CPU frequency	48MHz		
Packages	QFN TQFP	QFN TQFP	QFN TQFP
Oscillators	32.768kHz crystal oscillator (XOSC32K) 0.4-32MHz crystal oscillator (XOSC) 32.768kHzinternal oscillator (OSC32K) 32kHz ultra-low-power internal oscillator (OSCULP32K) 8MHz high-accuracy internal oscillator (OSC8M) 48MHz Digital Frequency Locked Loop (DFLL48M)		
Event System channels	8	8	8
SW Debug Interface	Yes	Yes	Yes
Watchdog Timer (WDT)	Yes	Yes	Yes

2. Ordering Information



2.1 SAM D20E

Ordering Code	FLASH (bytes)	SRAM (bytes)	Package	Carrier Type
SAMD20E14A-AU	16K	2K	TQFP32	Tray
SAMD20E14A-AUT				Tape & Reel
SAMD20E14A-MU			QFN32	Tray
SAMD20E14A-MUT				Tape & Reel
SAMD20E15A-AU	32K	4K	TQFP32	Tray
SAMD20E15A-AUT				Tape & Reel
SAMD20E15A-MU			QFN32	Tray
SAMD20E15A-MUT				Tape & Reel
SAMD20E16A-AU	64K	8K	TQFP32	Tray
SAMD20E16A-AUT				Tape & Reel
SAMD20E16A-MU			QFN32	Tray
SAMD20E16A-MUT				Tape & Reel
SAMD20E17A-AU	128K	16K	TQFP32	Tray
SAMD20E17A-AUT				Tape & Reel
SAMD20E17A-MU			QFN32	Tray
SAMD20E17A-MUT				Tape & Reel

Ordering Code	FLASH (bytes)	SRAM (bytes)	Package	Carrier Type
SAMD20E18A-AU	256K	32K	TQFP32	Tray
SAMD20E18A-AUT				Tape & Reel
SAMD20E18A-MU			QFN32	Tray
SAMD20E18A-MUT				Tape & Reel

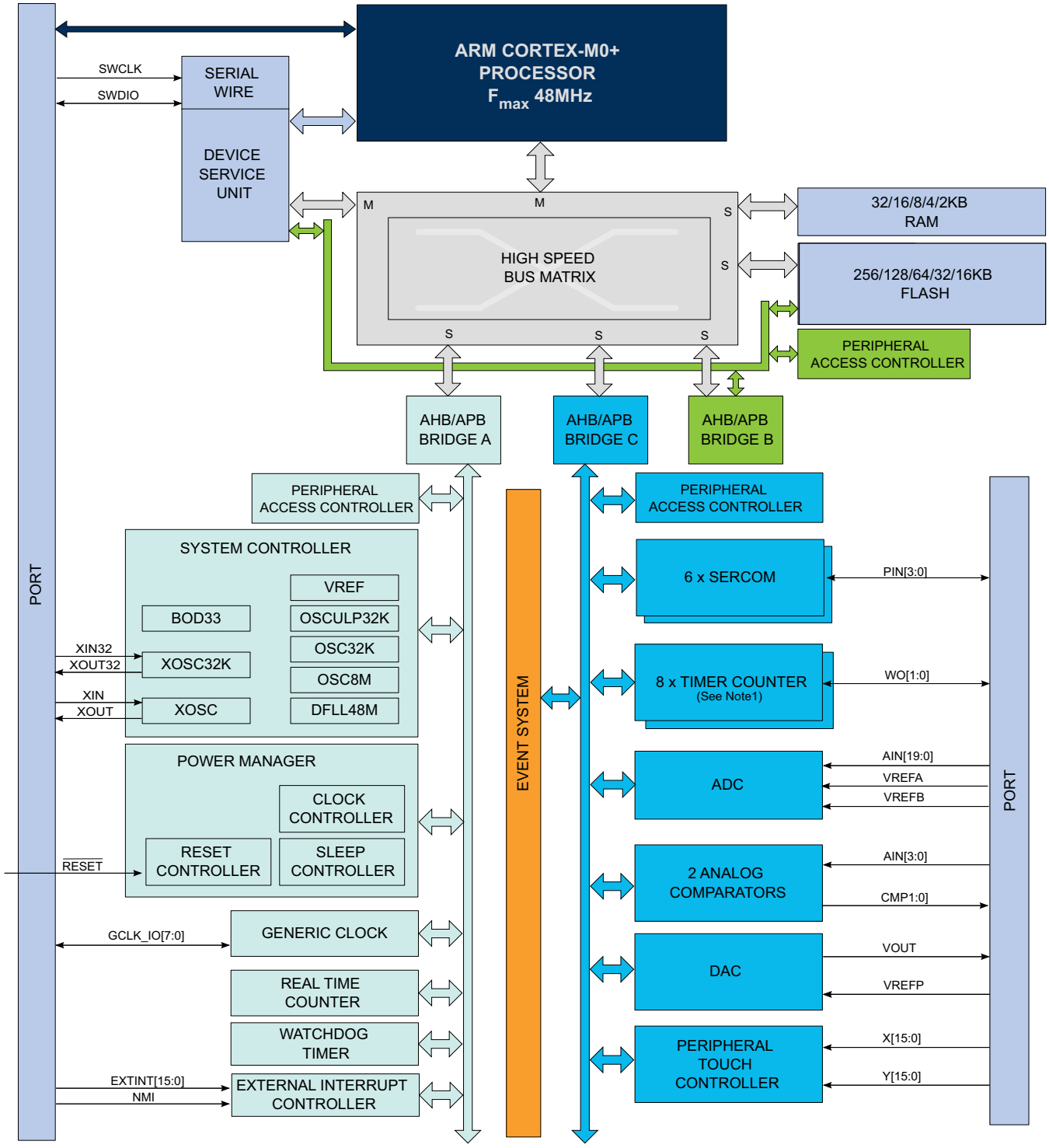
2.2 SAM D20G

Ordering Code	FLASH (bytes)	SRAM (bytes)	Package	Carrier Type
SAMD20G14A-AU	16K	2K	TQFP48	Tray
SAMD20G14A-AUT				Tape & Reel
SAMD20G14A-MU			QFN48	Tray
SAMD20G14A-MUT				Tape & Reel
SAMD20G15A-AU	32K	4K	TQFP48	Tray
SAMD20G15A-AUT				Tape & Reel
SAMD20G15A-MU			QFN48	Tray
SAMD20G15A-MUT				Tape & Reel
SAMD20G16A-AU	64K	8K	TQFP48	Tray
SAMD20G16A-AUT				Tape & Reel
SAMD20G16A-MU			QFN48	Tray
SAMD20G16A-MUT				Tape & Reel
SAMD20G17A-AU	128K	16K	TQFP48	Tray
SAMD20G17A-AUT				Tape & Reel
SAMD20G17A-MU			QFN48	Tray
SAMD20G17A-MUT				Tape & Reel
SAMD20G18A-AU	256K	32K	TQFP48	Tray
SAMD20G18A-AUT				Tape & Reel
SAMD20G18A-MU			QFN48	Tray
SAMD20G18A-MUT				Tape & Reel

2.3 SAM D20J

Ordering Code	FLASH (bytes)	SRAM (bytes)	Package	Carrier Type
SAMD20J14A-AU	16K	2K	TQFP64	Tray
SAMD20J14A-AUT				Tape & Reel
SAMD20J14A-MU			QFN64	Tray
SAMD20J14A-MUT				Tape & Reel
SAMD20J15A-AU	32K	4K	TQFP64	Tray
SAMD20J15A-AUT				Tape & Reel
SAMD20J15A-MU			QFN64	Tray
SAMD20J15A-MUT				Tape & Reel
SAMD20J16A-AU	64K	8K	TQFP64	Tray
SAMD20J16A-AUT				Tape & Reel
SAMD20J16A-MU			QFN64	Tray
SAMD20J16A-MUT				Tape & Reel
SAMD20J17A-AU	128K	16K	TQFP64	Tray
SAMD20J17A-AUT				Tape & Reel
SAMD20J17A-MU			QFN64	Tray
SAMD20J17A-MUT				Tape & Reel
SAMD20J18A-AU	256K	32K	TQFP64	Tray
SAMD20J18A-AUT				Tape & Reel
SAMD20J18A-MU			QFN64	Tray
SAMD20J18A-MUT				Tape & Reel

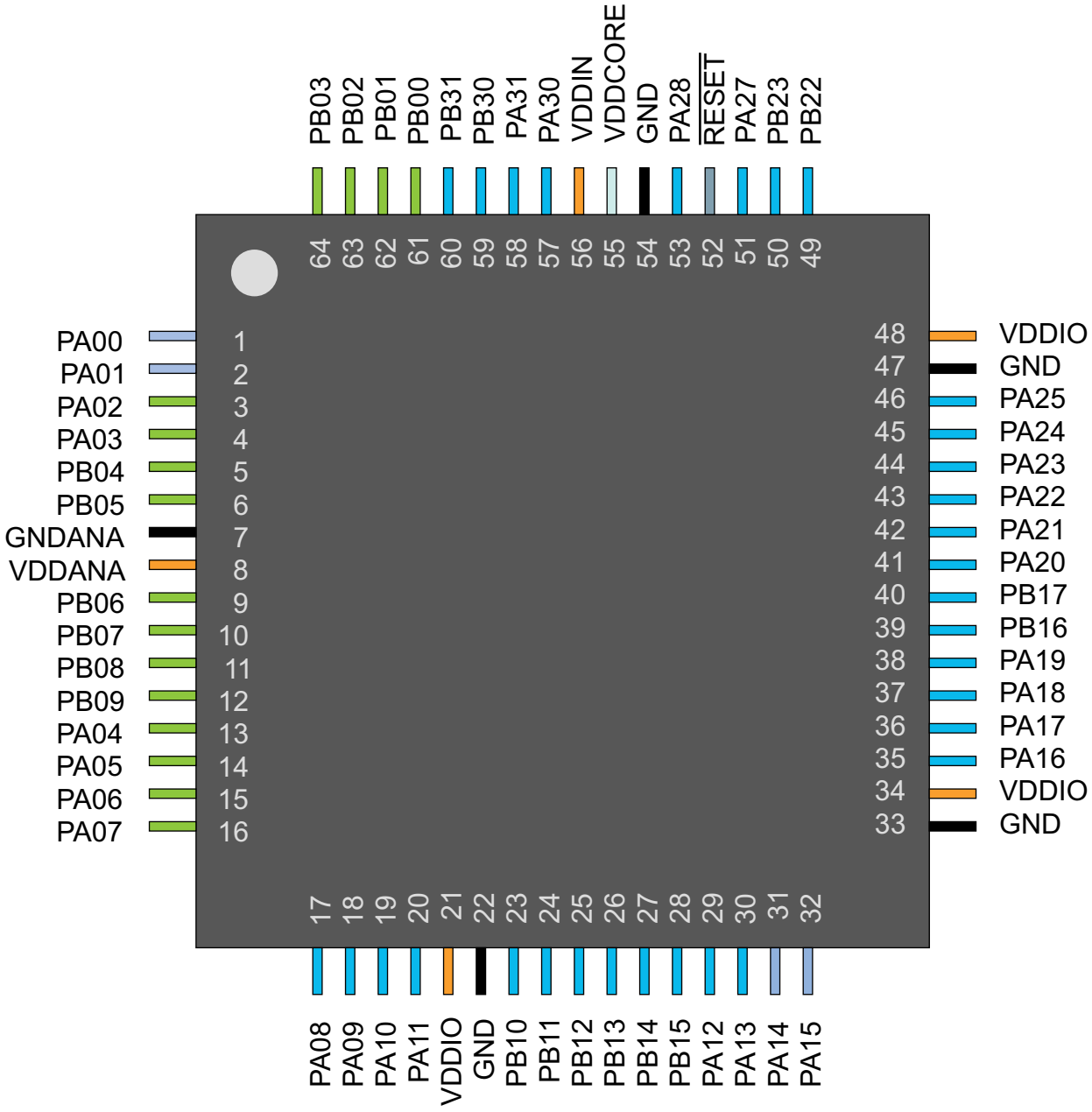
3. Block Diagram



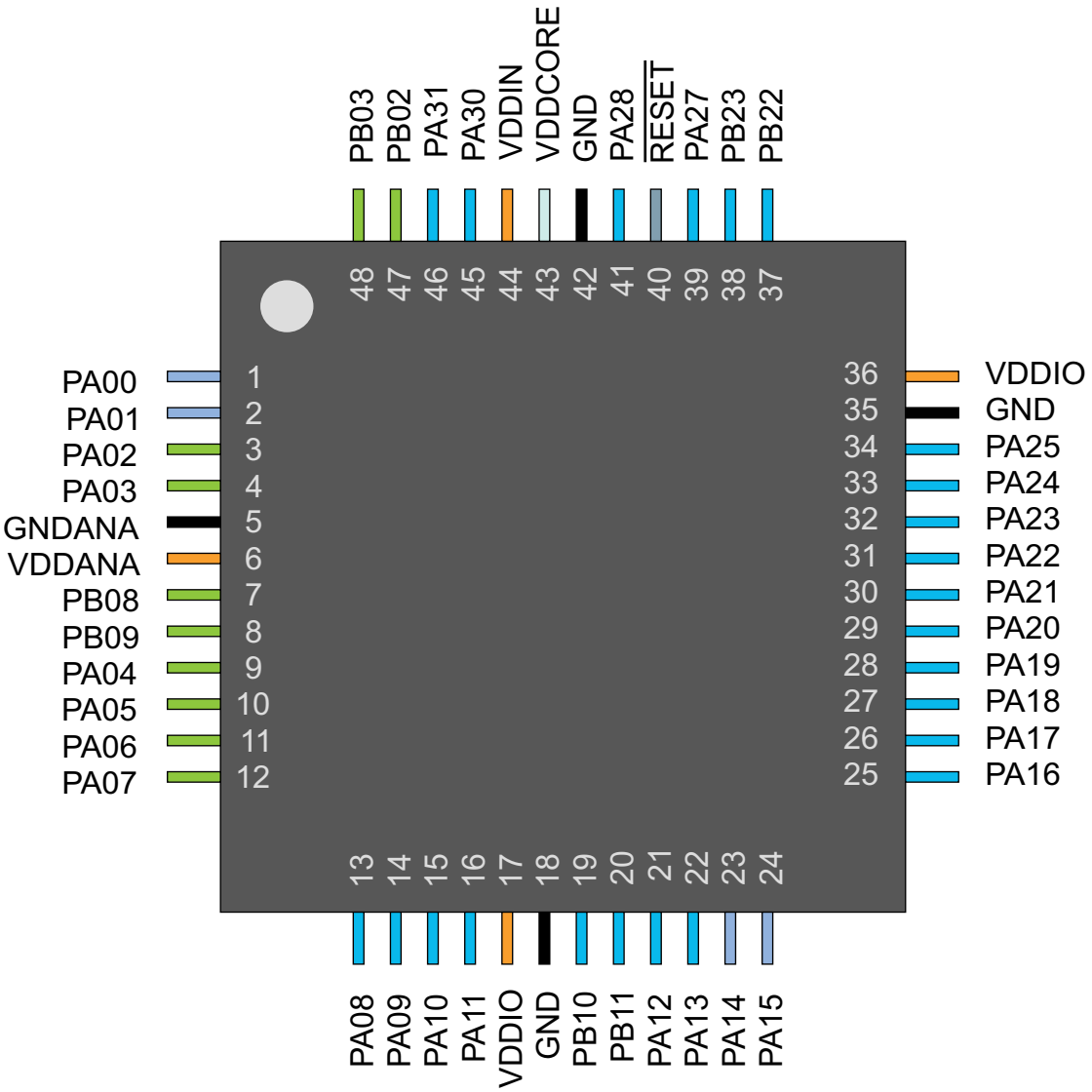
Notes: 1. Some products have different number of SERCOM instances, Timer/Counter instances, PTC signals and ADC signals. Refer to "Configuration Summary" on page 3 for details.

4. Pinout

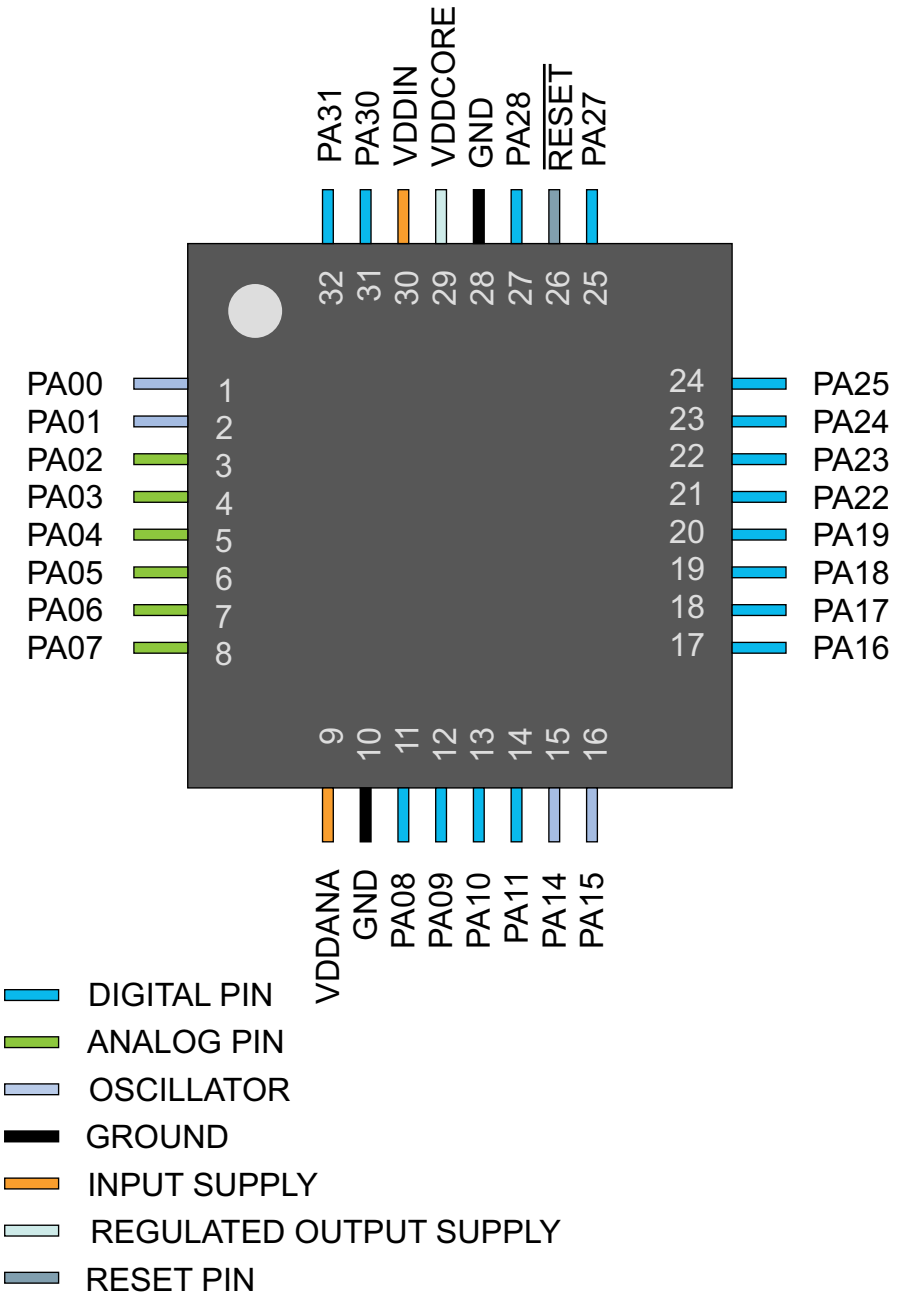
4.1 SAM D20J



4.2 SAM D20G


PB03
PB02
PA31
PA30
VDDIN
VDDCORE
GND
PA28
RESET
PA27
PB23
PB22
48
47
46
45
44
43
42
41
40
39
38
37
PA00
PA01
PA02
PA03
GNDANA
VDDANA
PB08
PB09
PA04
PA05
PA06
PA07
1
2
3
4
5
6
7
8
9
10
11
12
36
35
34
33
32
31
30
29
28
27
26
25
VDDIO
GND
PA25
PA24
PA23
PA22
PA21
PA20
PA19
PA18
PA17
PA16
13
14
15
16
17
18
19
20
21
22
23
24
PA08
PA09
PA10
PA11
VDDIO
GND
PB10
PB11
PA12
PA13
PA14
PA15
DIGITAL PIN
ANALOG PIN
OSCILLATOR
GROUND
INPUT SUPPLY
REGULATED OUTPUT SUPPLY
RESET PIN

4.3 SAM D20E


PA31
PA30
VDDIN
VDDCORE
GND
PA28
RESET
PA27
32
31
30
29
28
27
26
25
PA00
PA01
PA02
PA03
PA04
PA05
PA06
PA07
1
2
3
4
5
6
7
8
24
23
22
21
20
19
18
17
PA25
PA24
PA23
PA22
PA19
PA18
PA17
PA16
9
10
11
12
13
14
15
16
VDDANA
GND
PA08
PA09
PA10
PA11
PA14
PA15
DIGITAL PIN
ANALOG PIN
OSCILLATOR
GROUND
INPUT SUPPLY
REGULATED OUTPUT SUPPLY
RESET PIN

5. I/O Multiplexing and Considerations

5.1 Multiplexed Signals

Each pin is by default controlled by the PORT as a general purpose I/O and alternatively it can be assigned to one of the peripheral functions A, B, C, D, E, F, G or H. To enable a peripheral function on a pin, the Peripheral Multiplexer Enable bit in the Pin Configuration register corresponding to that pin (PINCFGn.PMUXEN, n = 0-31) in the PORT must be written to one. The selection of peripheral function A to H is done by writing to the Peripheral Multiplexing Odd and Even bits in the Peripheral Multiplexing register (PMUXn.PMUXE/O) in the PORT. Refer to "PORT" on page 276 for details on how to configure the I/O multiplexing.

Table 5-1 describes the peripheral signals multiplexed to the PORT I/O pins.

Table 5-1. PORT Function Multiplexing

Pin			I/O Pin	Supply	Type	A	B[1]					C	D	E	F	G	H
SAM D20E	SAM D20G	SAM D20J				EIC	REF	ADC	AC	PTC	DAC	SERCOM[2]		TC[3]			AC/GCLK
1	1	1	PA00	VDDANA		EXTINT[0]							SERCOM1/ PAD[0]		TC2/WO[0]		
2	2	2	PA01	VDDANA		EXTINT[1]							SERCOM1/ PAD[1]		TC2/WO[1]		
3	3	3	PA02	VDDANA		EXTINT[2]		AIN[0]		Y[0]	VOUT						
4	4	4	PA03	VDDANA		EXTINT[3]	ADC/VREFA DAC/VREFP	AIN[1]		Y[1]							
		5	PB04	VDDANA		EXTINT[4]		AIN[12]		Y[10]							
		6	PB05	VDDANA		EXTINT[5]		AIN[13]		Y[11]							
		9	PB06	VDDANA		EXTINT[6]		AIN[14]		Y[12]							
		10	PB07	VDDANA		EXTINT[7]		AIN[15]		Y[13]							
	7	11	PB08	VDDANA		EXTINT[8]		AIN[2]		Y[14]			SERCOM4/ PAD[0]		TC4/WO[0]		
	8	12	PB09	VDDANA		EXTINT[9]		AIN[3]		Y[15]			SERCOM4/ PAD[1]		TC4/WO[1]		
5	9	13	PA04	VDDANA		EXTINT[4]	ADC VREFB	AIN[4]	AIN[0]	Y[2]			SERCOM0/ PAD[0]		TC0/WO[0]		
6	10	14	PA05	VDDANA		EXTINT[5]		AIN[5]	AIN[1]	Y[3]			SERCOM0/ PAD[1]		TC0/WO[1]		
7	11	15	PA06	VDDANA		EXTINT[6]		AIN[6]	AIN[2]	Y[4]			SERCOM0/ PAD[2]		TC1/WO[0]		
8	12	16	PA07	VDDANA		EXTINT[7]		AIN[7]	AIN[3]	Y[5]			SERCOM0/ PAD[3]		TC1/WO[1]		
11	13	17	PA08	VDDIO	I^2C	NMI		AIN[16]		X[0]		SERCOM0/ PAD[0]	SERCOM2/ PAD[0]	TC0/WO[0]			
12	14	18	PA09	VDDIO	I^2C	EXTINT[9]		AIN[17]		X[1]		SERCOM0/ PAD[1]	SERCOM2/ PAD[1]	TC0/WO[1]			
13	15	19	PA10	VDDIO		EXTINT[10]		AIN[18]		X[2]		SERCOM0/ PAD[2]	SERCOM2/ PAD[2]	TC1/WO[0]			GCLK/IO[4]
14	16	20	PA11	VDDIO		EXTINT[11]		AIN[19]		X[3]		SERCOM0/ PAD[3]	SERCOM2/ PAD[3]	TC1/WO[1]			GCLK/IO[5]
	19	23	PB10	VDDIO		EXTINT[10]							SERCOM4/ PAD[2]		TC5/WO[0]		GCLK/IO[4]
	20	24	PB11	VDDIO		EXTINT[11]							SERCOM4/ PAD[3]		TC5/WO[1]		GCLK/IO[5]
		25	PB12	VDDIO	I^2C	EXTINT[12]				X[12]		SERCOM4/ PAD[0]		TC4/WO[0]			GCLK/IO[6]
		26	PB13	VDDIO	I^2C	EXTINT[13]				X[13]		SERCOM4/ PAD[1]		TC4/WO[1]			GCLK/IO[7]
		27	PB14	VDDIO		EXTINT[14]				X[14]		SERCOM4/ PAD[2]		TC5/WO[0]			GCLK/IO[0]

Table 5-1. PORT Function Multiplexing (Continued)

Pin			I/O Pin	Supply	Type	A	B[1]					C	D	E	F	G	H
SAM D20E	SAM D20G	SAM D20J				EIC	REF	ADC	AC	PTC	DAC	SERCOM[2]		TC[3]			AC/GCLK
		28	PB15	VDDIO		EXTINT[15]				X[15]		SERCOM4/ PAD[3]		TC5/WO[1]			GCLK/IO[1]
	21	29	PA12	VDDIO	I^2C	EXTINT[12]						SERCOM2/ PAD[0]	SERCOM4/ PAD[0]	TC2/WO[0]			AC/CMP[0]
	22	30	PA13	VDDIO	I^2C	EXTINT[13]						SERCOM2/ PAD[1]	SERCOM4/ PAD[1]	TC2/WO[1]			AC/CMP[1]
15	23	31	PA14	VDDIO		EXTINT[14]						SERCOM2/ PAD[2]	SERCOM4/ PAD[2]	TC3/WO[0]			GCLK/IO[0]
16	24	32	PA15	VDDIO		EXTINT[15]						SERCOM2/ PAD[3]	SERCOM4/ PAD[3]	TC3/WO[1]			GCLK/IO[1]
17	25	35	PA16	VDDIO	I^2C	EXTINT[0]				X[4]		SERCOM1/ PAD[0]	SERCOM3/ PAD[0]		TC2/WO[0]		GCLK/IO[2]
18	26	36	PA17	VDDIO	I^2C	EXTINT[1]				X[5]		SERCOM1/ PAD[1]	SERCOM3/ PAD[1]		TC2/WO[1]		GCLK/IO[3]
19	27	37	PA18	VDDIO		EXTINT[2]				X[6]		SERCOM1/ PAD[2]	SERCOM3/ PAD[2]		TC3/WO[0]		AC/CMP[0]
20	28	38	PA19	VDDIO		EXTINT[3]				X[7]		SERCOM1/ PAD[3]	SERCOM3/ PAD[3]		TC3/WO[1]		AC/CMP[1]
		39	PB16	VDDIO	I^2C	EXTINT[0]						SERCOM5/ PAD[0]		TC6/WO[0]			GCLK/IO[2]
		40	PB17	VDDIO	I^2C	EXTINT[1]						SERCOM5/ PAD[1]		TC6/WO[1]			GCLK/IO[3]
	29	41	PA20	VDDIO		EXTINT[4]				X[8]		SERCOM5/ PAD[2]	SERCOM3/ PAD[2]	TC7/WO[0]			GCLK/IO[4]
	30	42	PA21	VDDIO		EXTINT[5]				X[9]		SERCOM5/ PAD[3]	SERCOM3/ PAD[3]	TC7/WO[1]			GCLK/IO[5]
21	31	43	PA22	VDDIO	I^2C	EXTINT[6]				X[10]		SERCOM3/ PAD[0]	SERCOM5/ PAD[0]		TC4/WO[0]		GCLK/IO[6]
22	32	44	PA23	VDDIO	I^2C	EXTINT[7]				X[11]		SERCOM3/ PAD[1]	SERCOM5/ PAD[1]		TC4/WO[1]		GCLK/IO[7]
23	33	45	PA24	VDDIO		EXTINT[12]						SERCOM3/ PAD[2]	SERCOM5/ PAD[2]		TC5/WO[0]		
24	34	46	PA25	VDDIO		EXTINT[13]						SERCOM3/ PAD[3]	SERCOM5/ PAD[3]		TC5/WO[1]		
	37	49	PB22	VDDIO		EXTINT[6]							SERCOM5/ PAD[2]		TC7/WO[0]		GCLK/IO[0]
	38	50	PB23	VDDIO		EXTINT[7]							SERCOM5/ PAD[3]		TC7/WO[1]		GCLK/IO[1]
25	39	51	PA27	VDDIO		EXTINT[15]											GCLK/IO[0]
27	41	53	PA28	VDDIO		EXTINT[8]											GCLK/IO[0]
31	45	57	PA30	VDDIO		EXTINT[10]							SERCOM1/ PAD[2]		TC1/WO[0]	SWCLK	GCLK/IO[0]
32	46	58	PA31	VDDIO		EXTINT[11]							SERCOM1/ PAD[3]		TC1/WO[1]		
		59	PB30	VDDIO	I^2C	EXTINT[14]							SERCOM5/ PAD[0]		TC0/WO[0]		
		60	PB31	VDDIO	I^2C	EXTINT[15]							SERCOM5/ PAD[1]		TC0/WO[1]		
		61	PB00	VDDANA		EXTINT[0]		AIN[8]		Y[6]			SERCOM5/ PAD[2]		TC7/WO[0]		
		62	PB01	VDDANA		EXTINT[1]		AIN[9]		Y[7]			SERCOM5/ PAD[3]		TC7/WO[1]		
	47	63	PB02	VDDANA		EXTINT[2]		AIN[10]		Y[8]			SERCOM5/ PAD[0]		TC6/WO[0]		
	48	64	PB03	VDDANA		EXTINT[3]		AIN[11]		Y[9]			SERCOM5/ PAD[1]		TC6/WO[1]		

Note: 1. All analog pin functions are on peripheral function B. Peripheral function B must be selected to disable the digital control of the pin.

2. Only some pins can be used in SERCOM I^2C mode. See the Type column for using a SERCOM pin in I^2C mode. Refer to the "I2C Pins" on page 565 for details on the I^2C pin characteristics
3. Note that TC6 and TC7 are not supported on the SAM D20G. Refer to "Configuration Summary" on page 3 for details.

5.2 Other Functions

5.2.1 Oscillator Pinout

The oscillators are not mapped to the normal PORT functions and their multiplexing are controlled by registers in the System Controller (SYSCTRL). Refer to "SYSCTRL – System Controller" on page 127 for more information.

Oscillator	Supply	Signal	I/O Pin
XOSC	VDDIO	XIN	PA14
		XOUT	PA15
XOSC32K	VDDANA	XIN32	PA00
		XOUT32	PA01

5.2.2 Serial Wire Debug Interface Pinout

Only the SWCLK pin is mapped to the normal PORT functions. A debugger cold-plugging or hot-plugging detection will automatically switch the SWDIO port to the SWDIO function. Refer to "DSU – Device Service Unit" on page 36 for more information.

Signal	Supply	I/O Pin
SWCLK	VDDIO	PA30
SWDIO	VDDIO	PA31

6. Signal Descriptions List

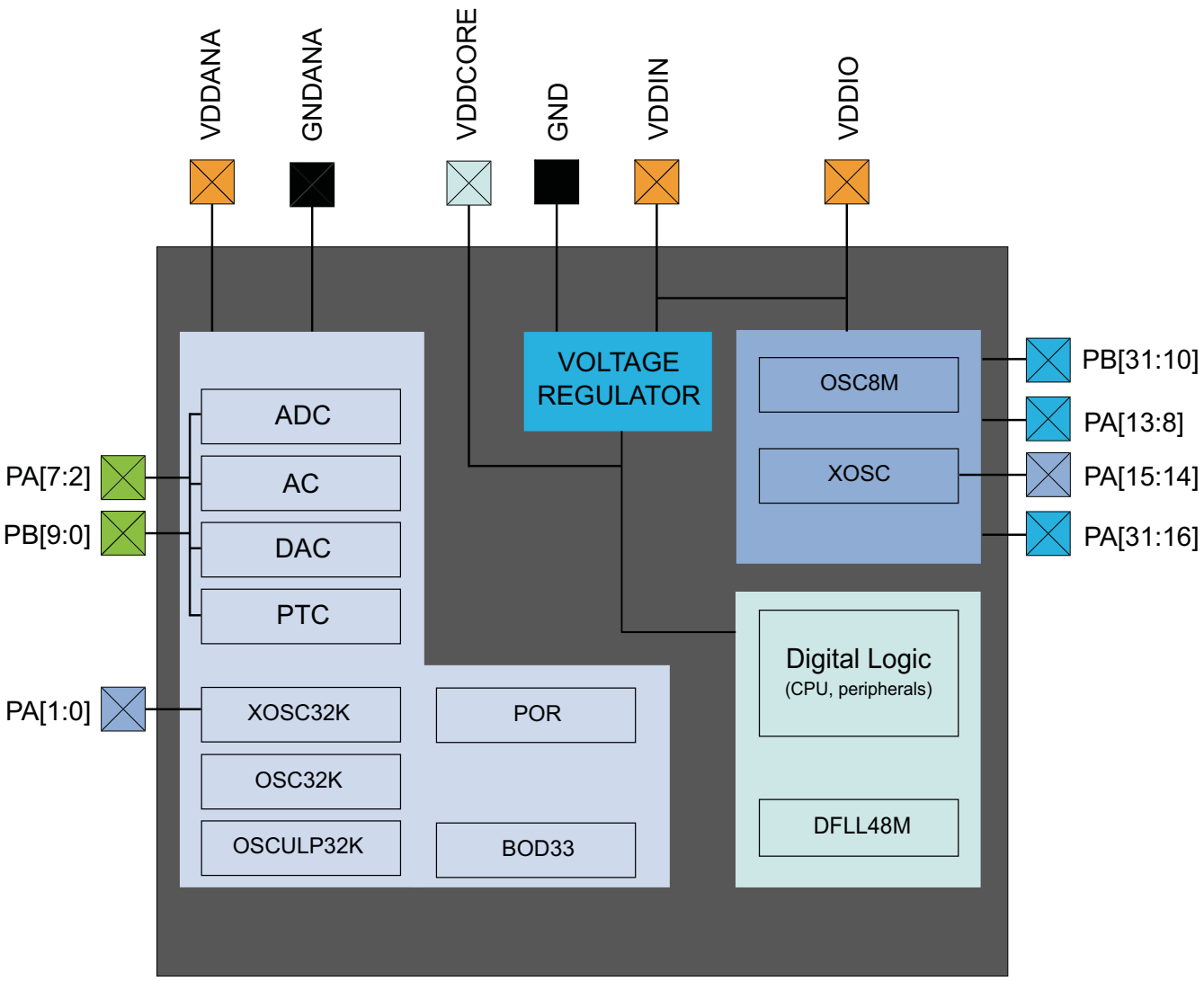
The following table gives details on signal names classified by peripheral.

Signal Name	Function	Type	Active Level
Analog Comparators - AC			
AIN[3:0]	AC Analog Inputs	Analog	
CMP[1:0]	AC Comparator Outputs	Digital	
Analog Digital Converter - ADC			
AIN[19:0]	ADC Analog Inputs	Analog	
VREFP	ADC Voltage External Reference	Analog	
Digital Analog Converter - DAC			
VOUT	DAC Voltage output	Analog	
VREFP	DAC Voltage External Reference	Analog	
External Interrupt Controller - EIC			
EXTINT[15:0]	External Interrupts	Input	
NMI	External Non-Maskable Interrupt	Input	
Generic Clock Generator - GCLK			
IO[7:0]	Generic Clock (source clock or generic clock generator output)	I/O	
Power Manager - PM			
$\overline{\text{RESET}}$	Reset	Input	Low
Serial Communication Interface - SERCOMx			
PAD[3:0]	SERCOM I/O Pads	I/O	
System Control - SYSCTRL			
XIN	Crystal Input	Analog/ Digital	
XIN32	32kHz Crystal Input	Analog/ Digital	
XOUT	Crystal Output	Analog	
XOUT32	32kHz Crystal Output	Analog	
Timer Counter - TCx			
WO[1:0]	Waveform Outputs	Output	Low
Peripheral Touch Controller - PTC			
X[15:0]	PTC Input	Analog	
Y[15:0]	PTC Input	Analog	
General Purpose I/O - PORT			
PA25 - PA00	Parallel I/O Controller I/O Port A	I/O	

Signal Name	Function	Type	Active Level
PA28 - PA27	Parallel I/O Controller I/O Port A	I/O	
PA31 - PA30	Parallel I/O Controller I/O Port A	I/O	
PB17 - PB00	Parallel I/O Controller I/O Port B	I/O	
PB23 - PB22	Parallel I/O Controller I/O Port B	I/O	
PB31 - PB30	Parallel I/O Controller I/O Port B	I/O	

7. Power Supply and Start-Up Considerations

7.1 Power Domain Overview



7.2 Power Supply Considerations

7.2.1 Power Supplies

The Atmel® SAM D20 has several different power supply pins:

- VDDIO: Powers I/O lines, OSC8M and XOSC. Voltage is 1.62V to 3.63V.
- VDDIN: Powers I/O lines and the internal regulator. Voltage is 1.62V to 3.63V.
- VDDANA: Powers I/O lines and the ADC, AC, DAC, PTC, OSCULP32K, OSC32K, XOSC32K. Voltage is 1.62V to 3.63V.
- VDDCORE: Internal regulated voltage output. Powers the core, memories and peripherals. Voltage is 1.2V.

The same voltage must be applied to both VDDIN, VDDIO and VDDANA. This common voltage is referred to as V_{DD} in the datasheet.

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 checklist.

Refer to “Schematic Checklist” on page 598 for details.

7.2.2 Voltage Regulator

The voltage regulator has two different modes:

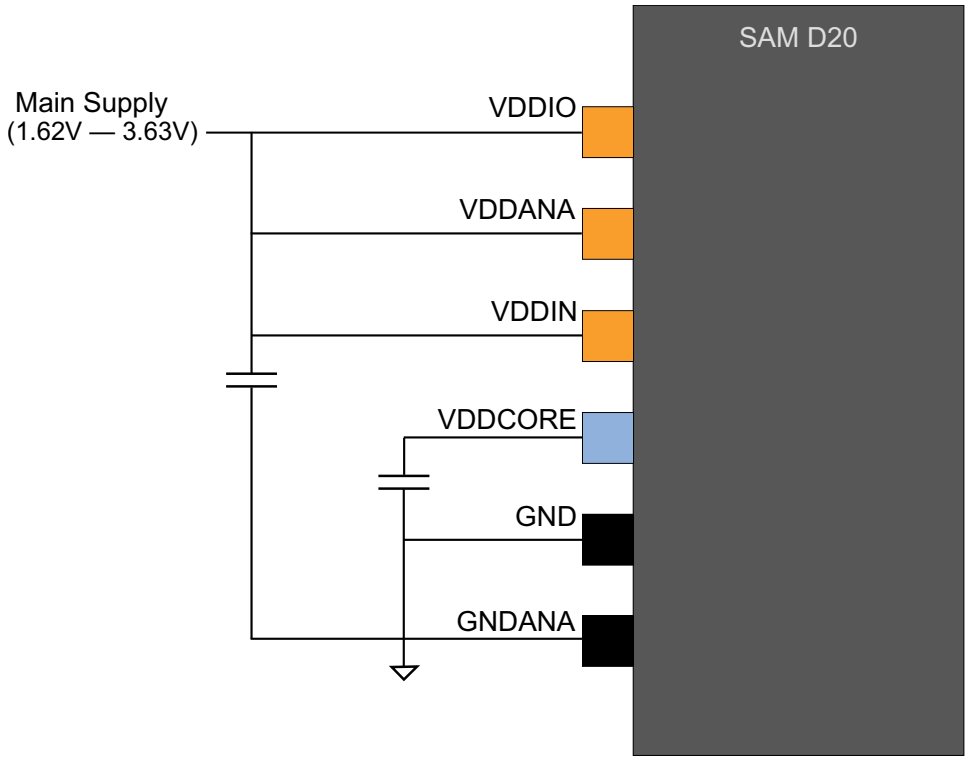
- Normal mode: To be used when the CPU and peripherals are running
- Low Power (LP) mode: To be used when the regulator draws small static current. It can be used in standby mode

7.2.3 Typical Powering Schematics

The SAM D20 uses a single supply from 1.62V to 3.63V.

The following figure shows the recommended power supply connection.

Figure 7-1. Power Supply Connection



7.2.4 Power-Up Sequence

7.2.4.1 Minimum Rise Rate

The integrated power-on reset (POR) circuitry monitoring the VDDANA power supply requires a minimum rise rate. Refer to the “Electrical Characteristics” on page 558 for details.

7.2.4.2 Maximum Rise Rate

The rise rate of the power supply must not exceed the values described in Electrical Characteristics. Refer to the “Electrical Characteristics” on page 558 for details.

7.3 Power-Up

This section summarizes the power-up sequence of the SAM D20. The behavior after power-up is controlled by the Power Manager. Refer to “PM – Power Manager” on page 100 for details.

7.3.1 Starting of Clocks

After power-up, the device is set to its initial state and kept in reset, until the power has stabilized throughout the device. Once the power has stabilized, the device will use a 1MHz clock. This clock is derived from the 8MHz Internal Oscillator (OSC8M), which is divided by eight and used as a clock source for generic clock generator 0. Generic clock generator 0 is the main clock for the Power Manager (PM).

Some synchronous system clocks are active, allowing software execution.

Refer to the “Clock Mask Register” section in “PM – Power Manager” on page 100 for the list of default peripheral clocks running. Synchronous system clocks that are running are by default not divided and receive a 1MHz clock through generic clock generator 0. Other generic clocks are disabled except GCLK_WDT, which is used by the Watchdog Timer (WDT).

7.3.2 I/O Pins

After power-up, the I/O pins are tri-stated.

7.3.3 Fetching of Initial Instructions

After reset has been released, the CPU starts fetching PC and SP values from the reset address, which is 0x00000000. This address points to the first executable address in the internal flash. The code read from the internal flash is free to configure the clock system and clock sources. Refer to “PM – Power Manager” on page 100, “GCLK – Generic Clock Controller” on page 78 and “SYSCTRL – System Controller” on page 127 for details. Refer to the ARM Architecture Reference Manual for more information on CPU startup (http://www.arm.com).

7.4 Power-On Reset and Brown-Out Detector

The SAM D20 embeds three features to monitor, warn and/or reset the device:

- POR: Power-on reset on VDDANA
- BOD33: Brown-out detector on VDDANA

7.4.1 Power-On Reset on VDDANA

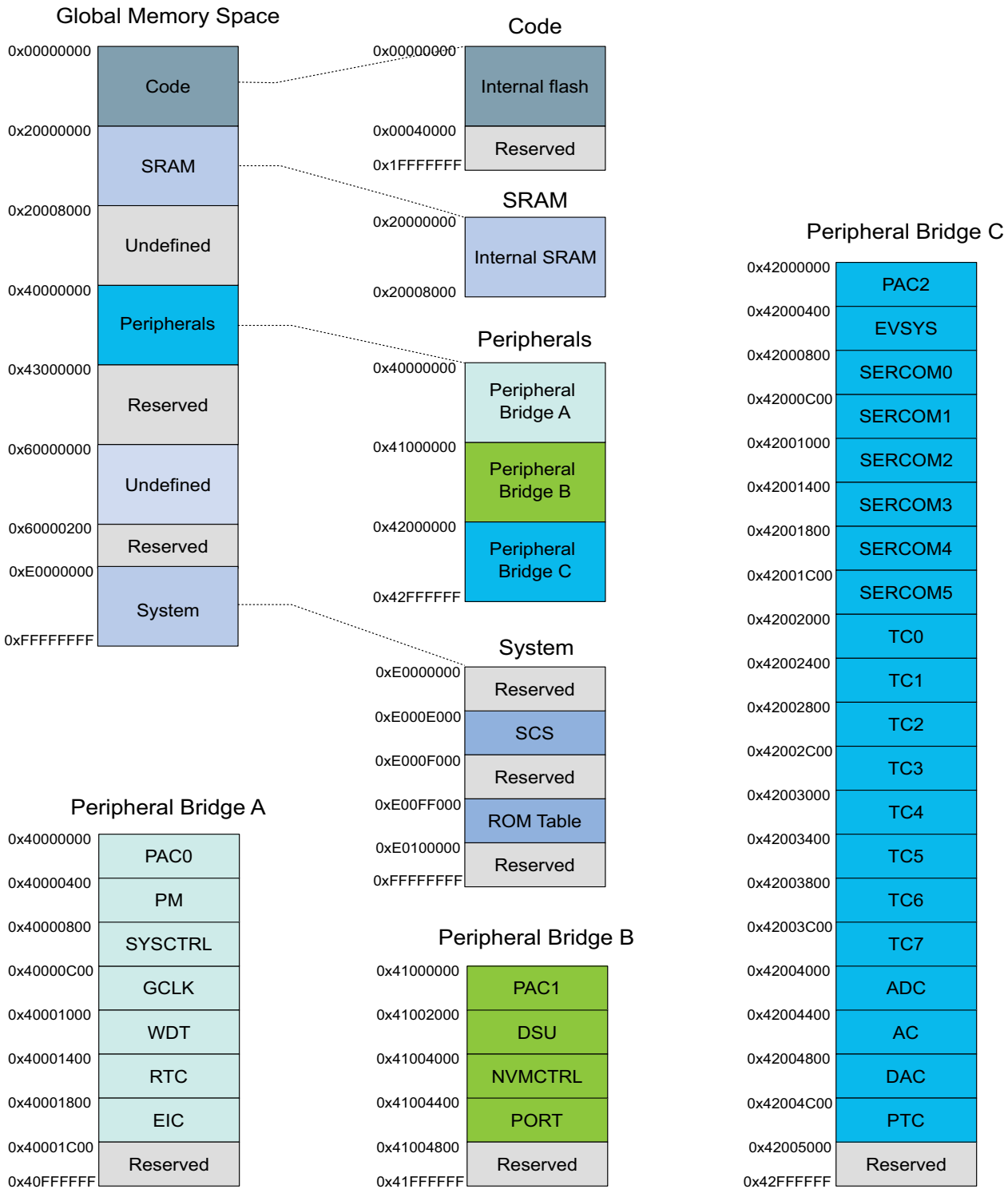
POR monitors VDDANA. It is always activated and monitors voltage at startup and also during all the sleep modes. If VDDANA goes below the threshold voltage, the entire chip is reset.

7.4.2 Brown-Out Detector on VDDANA

BOD33 monitors VDDANA. Refer to “SYSCTRL – System Controller” on page 127 for details.

8. Product Mapping

Figure 8-1. SAM D20 Product Mapping



This figure represents the full configuration of the Atmel® SAM D20 with maximum flash and SRAM capabilities and a full set of peripherals. Refer to the "Configuration Summary" on page 3 for details.

9. Memories

9.1 Embedded Memories

- Internal high-speed flash
- Internal high-speed RAM, single-cycle access at full speed
- Dedicated flash area for EEPROM emulation

9.2 Physical Memory Map

The High-Speed bus is implemented as a Bus Matrix. Refer to "High-Speed Bus Matrix" on page 25 for details. All High-Speed bus addresses are fixed, and they are never remapped. The 32-bit physical address space is mapped as follows:

Table 9-1. SAM D20 Physical Memory Map(1)

Memory	Start address	Size				
		SAMD20x18	SAMD20x17	SAMD20x16	SAMD20x15	SAMD20x14
Embedded Flash	0x00000000	256KB	128KB	64KB	32KB	16KB
Embedded SRAM	0x20000000	32KB	16KB	8KB	4KB	2KB
Peripheral Bridge A	0x40000000	64KB	64KB	64KB	64KB	64KB
Peripheral Bridge B	0x41000000	64KB	64KB	64KB	64KB	64KB
Peripheral Bridge C	0x42000000	64KB	64KB	64KB	64KB	64KB

Note: 1. x = G, J or E. SAMD20E18 is not available. Refer to "Ordering Information" on page 4 for details.

Table 9-2. Flash Memory Parameters(1)

Device	Flash Size	Number of Pages (NVMP)	Page Size (PSZ)	Row Size
SAMD20x18	256KB	4096	64 bytes	4 pages = 256 bytes
SAMD20x17	128KB	2048	64 bytes	4 pages = 256 bytes
SAMD20x16	64KB	1024	64 bytes	4 pages = 256 bytes
SAMD20x15	32KB	512	64 bytes	4 pages = 256 bytes
SAMD20x14	16KB	256	64 bytes	4 pages = 256 bytes

Notes: 1. x = G, J or E. SAMD20E18 is not available. Refer to "Ordering Information" on page 4 for details.

2. The number of pages (NVMP) and page size (PSZ) can be read from the NVM Pages and Page Size bits in the NVM Parameter register in the NVMCTRL (PARAM.NVMP and PARAM.PSZ, respectively). Refer to PARAM for details.

9.3 Non-Volatile Memory (NVM) User Row Mapping

The NVM User Row contains calibration data that are automatically read at device power on.

The NVM User Row can be read at address 0x804000.

To write the NVM User Row refer to “NVMCTRL – Non-Volatile Memory Controller” on page 254.

Note that when writing to the User Row the values will only be loaded at device reset.

Table 9-3. NVM User Row Mapping

Bit Position	Name	Description
2:0	BOOTPROT	Used to select one of eight different bootloader sizes. Refer to “NVMCTRL – Non-Volatile Memory Controller” on page 254.
3	Reserved	
6:4	EEPROM	Used to select one of eight different EEPROM area sizes. Refer to “NVMCTRL – Non-Volatile Memory Controller” on page 254.
7	Reserved	
13:8	BOD33 Level	BOD33 Threshold Level (BOD33.LEVEL) at power on. Refer to BOD33 register.
14	BOD33 Enable	BOD33 Enable at power on. Refer to BOD33 register.
16:15	BOD33 Action	BOD33 Action at power on. Refer to BOD33 register.
21:17	Reserved	
22	Reserved	
24:23	Reserved	
25	WDT Enable	WDT Enable at power on. Refer to WDT CTRL register.
26	WDT Always-On	WDT Always-On at power on. Refer to WDT CTRL register.
30:27	WDT Period	WDT Period at power on. Refer to WDT CONFIG register.
34:31	WDT Window	WDT Window mode time-out at power on. Refer to WDT CONFIG register.
38:35	WDT EWOFFSET	WDT Early Warning Interrupt Time Offset at power on. Refer to WDT EWCTRL register.
39	WDT WEN	WDT Timer Window Mode Enable at power on. Refer to WDT CTRL register.
40	BOD33 Hysteresis	BOD33 Hysteresis configuration at power on. Refer to BOD33 register.
41	Reserved	
47:42	Reserved	
63:48	LOCK	NVM Region Lock Bits. Refer to “NVMCTRL – Non-Volatile Memory Controller” on page 254.

9.4 NVM Software Calibration Row Mapping

The NVM Software Calibration Row contains calibration data that are measured and written during production test. These calibration values should be read by the application software and written back to the corresponding register.

The NVM Software Calibration Row can be read at address 0x806020.

The NVM Software Calibration Row can not be written.

Table 9-4. NVM Software Calibration Row Mapping

Bit Position	Name	Description
2:0	Reserved	
14:3	Reserved	
26:15	Reserved	
34:27	ADC LINEARITY	ADC Linearity Calibration. Should be written to CALIB register.
37:35	ADC BIASCAL	ADC Bias Calibration. Should be written to CALIB register.
44:38	OSC32K CAL	OSC32KCalibration. Should be written to OSC32K register.
63:38	Reserved	

10. Processor and Architecture

10.1 Cortex-M0+ Processor

The Atmel® SAM D20 implements the ARM® Cortex®-M0+ processor, which is based on the ARMv6 architecture and Thumb®-2 ISA. The Cortex M0+ is 100% instruction set compatible with its predecessor, the Cortex-M0 processor, and upward compatible with the Cortex-M3 and Cortex-M4 processors. The ARM Cortex-M0+ implemented is revision r0p1. For more information, refer to www.arm.com.

10.1.1 Cortex-M0+ Configuration

Feature	Configurable Option	SAM D20 Configuration
Interrupts	External interrupts 0-32	32
Data endianness	Little-endian or big-endian	Little-endian
SysTick timer	Present or absent	Present
Number of watchpoint comparators	0, 1, 2	2
Number of breakpoint comparators	0, 1, 2, 3, 4	4
Halting debug support	Present or absent	Present
Multiplier	Fast or small	Fast (single cycle)
Single-cycle I/O port	Present or absent	Present
Wake-up interrupt controller	Supported or not supported	Not supported
Vector Table Offset Register	Present or absent	Present
Unprivileged/Privileged support	Present or absent	Absent
Memory Protection Unit	Not present or 8-region	Not present
Reset all registers	Present or absent	Absent(1)
Instruction fetch width	16-bit only or mostly 32-bit	32-bit

Note: 1. All software run in privileged mode only

The ARM Cortex-M0+ processor has two bus interfaces:

- Single 32-bit AMBA® 3 AHB-Lite™ system interface that provides connections to peripherals and all system memory, including flash and RAM
- Single 32-bit I/O port bus interfacing to the PORT with one-cycle loads and stores

10.1.2 Cortex-M0+ Peripherals

- System Control Space (SCS)
  - The processor provides debug through registers in the SCS. Refer to the Cortex-M0+ Technical Reference Manual for details (www.arm.com).
- System Timer (SysTick)
  - The System Timer is a 24-bit timer that extends the functionality of both the processor and the NVIC. Refer to the Cortex-M0+ Technical Reference Manual for details (www.arm.com).

- Nested Vectored Interrupt Controller (NVIC)
  - External interrupt signals connect to the NVIC, and the NVIC prioritizes the interrupts. Software can set the priority of each interrupt. The NVIC and the Cortex-M0+ processor core are closely coupled, providing low latency interrupt processing and efficient processing of late arriving interrupts. Refer to "Nested Vector Interrupt Controller" on page 24 and the Cortex-M0+ Technical Reference Manual for details (www.arm.com).
- System Control Block (SCB)
  - The System Control Block provides system implementation information, and system control. This includes configuration, control, and reporting of the system exceptions. Refer to the Cortex-M0+ Devices Generic User Guide for details (www.arm.com).

10.1.3 Cortex-M0+ Address Map

Table 10-1. Cortex-M0+ Address Map

Address	Peripheral
0xE000E000	System Control Space (SCS)
0xE000E010	System Timer (SysTick)
0xE000E100	Nested Vectored Interrupt Controller (NVIC)
0xE000ED00	System Control Block (SCB)

10.1.4 I/O Interface

10.1.4.1 Overview

Because accesses to the AMBA AHB-Lite and the single-cycle I/O interface can be made concurrently, the Cortex-M0+ processor can fetch the next instructions while accessing the I/Os. This enables single-cycle I/O accesses to be sustained for as long as needed.

10.1.4.2 Description

Direct access to PORT registers.

10.2 Nested Vector Interrupt Controller

10.2.1 Overview

The Nested Vectored Interrupt Controller (NVIC) in the SAM D20 supports 32 interrupt lines with four different priority levels. For more details, refer to the Cortex-M0+ Technical Reference Manual (www.arm.com).

10.2.2 Interrupt Line Mapping

Each of the 32 interrupt lines is connected to one peripheral instance, as shown in the table below. Each peripheral can have one or more interrupt flags, located in the peripheral's Interrupt Flag Status and Clear (INTFLAG) register. The interrupt flag is set when the interrupt condition occurs. Each interrupt in the peripheral can be individually enabled by writing a one to the corresponding bit in the peripheral's Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the peripheral's Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated from the peripheral when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt requests for one peripheral are ORed together on system level, generating one interrupt request for each peripheral. An interrupt request will set the corresponding interrupt pending bit in the NVIC interrupt pending registers (SETPEND/CLRPEND bits in ISPR/ICPR). For the NVIC to activate the interrupt, it must be enabled in the NVIC interrupt enable register (SETENA/CLRENA bits in ISER/ICER). The NVIC interrupt priority registers IPR0-IPR7 provide a priority field for each interrupt.

Peripheral Source	NVIC Line
EIC NMI – External Interrupt Controller Non Maskable Interrupt	NMI
PM – Power Manager	0
SYSCTRL – System Controller	1
WDT – Watchdog Timer	2
RTC – Real Time Counter	3
EIC – External Interrupt Controller	4
NVMCTRL – Non-Volatile Memory Controller	5
EVSYS – Event System	6
SERCOM0 – Serial Communication Interface 0	7
SERCOM1 – Serial Communication Interface 1	8
SERCOM2 – Serial Communication Interface 2	9
SERCOM3 – Serial Communication Interface 3	10
SERCOM4 – Serial Communication Interface 4	11
SERCOM5 – Serial Communication Interface 5	12
TC0 – Timer/Counter 0	13
TC1 – Timer/Counter 1	14
TC2 – Timer/Counter 2	15
TC3 – Timer/Counter 3	16
TC4 – Timer/Counter 4	17
TC5 – Timer/Counter 5	18
TC6 – Timer/Counter 6	19
TC7 – Timer/Counter 7	20
ADC – Analog-to-Digital Converter	21
AC – Analog Comparator	22
DAC – Digital-to-Analog Converter	23
PTC – Peripheral Touch Controller	24

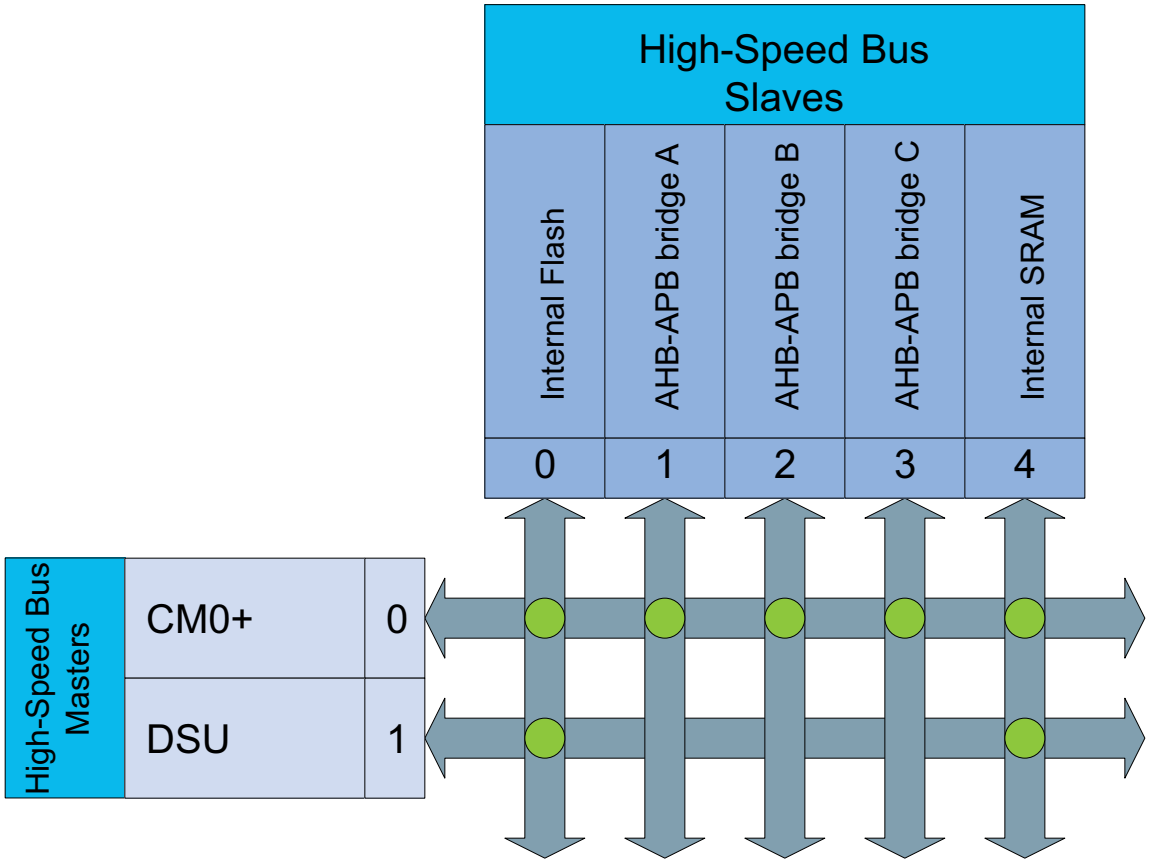
10.3 High-Speed Bus Matrix

10.3.1 Features

The High-Speed Bus Matrix includes these features:

- Symmetric crossbar bus switch implementation
- Allows concurrent accesses from different masters to different slaves
- 32-bit data bus
- Operation at a one-to-one clock frequency with the bus masters

10.3.2 Configuration



10.4 AHB-APB Bridge

The AHB-APB bridge is an AHB slave, providing an interface between the high-speed AHB domain and the low-power APB domain. It is used to provide access to the programmable control registers of peripherals (see “Product Mapping” on page 19).

to operate the AHB-APB bridge, the clock (CLK_HPBx_AHB) must be enabled. See “PM – Power Manager” on page 100 for details.

10.5 PAC – Peripheral Access Controller

10.5.1 Overview

There is one PAC associated with each AHB-APB bridge. The PAC can provide write protection for registers of each peripheral connected on the same bridge.

The PAC peripheral bus clock (CLK_PACx_APB) is enabled by default, and can be enabled and disabled in the Power Manager. Refer to “PM – Power Manager” on page 100 for details. The PAC will continue to operate in any sleep mode where the selected clock source is running.

Write-protection does not apply for debugger access. When the debugger makes an access to a peripheral, write-protection is ignored so that the debugger can update the register.

Write-protect registers allow the user to disable a selected peripheral’s write-protection without doing a read-modify-write operation. These registers are mapped into two I/O memory locations, one for clearing and one for setting the register bits. Writing a one to a bit in the Write Protect Clear register (WPCLR) will clear the corresponding bit in both registers (WPCLR and WPSET) and disable the write-protection for the corresponding peripheral, while writing a one to a bit in the Write Protect Set (WPSET) register will set the corresponding bit in both registers (WPCLR and WPSET) and enable the write-protection for the corresponding peripheral. Both registers (WPCLR and WPSET) will return the same value when read.

If a peripheral is write-protected, and if a write access is performed, data will not be written, and the peripheral will return an access error (CPU exception).

The PAC also offers a safety feature for correct program execution, with a CPU exception generated on double write-protection or double unprotection of a peripheral. If a peripheral *n* is write-protected and a write to one in WPSET[n] is detected, the PAC returns an error. This can be used to ensure that the application follows the intended program flow by always following a write-protect with an unprotect, and vice versa. However, in applications where a write-protected peripheral is used in several contexts, e.g., interrupts, care should be taken so that either the interrupt can not happen while the main application or other interrupt levels manipulate the write-protection status, or when the interrupt handler needs to unprotect the peripheral, based on the current protection status, by reading WPSET.

10.6 Register Description

Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Refer to “Product Mapping” on page 19 for PAC locations.

10.6.1 Write Protect Clear

Name: WPCLR

Offset: 0x00

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		EIC	**RTC**	**WDT**	**GCLK**	**SYSCTRL**	**PM**	
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:7 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 6:1 – EIC, RTC, WDT, GCLK, SYSCTRL, PM: Write Protect Disable**
  0: Write-protection is disabled.
  1: Write-protection is enabled.
  Writing a zero to these bits has no effect.
  Writing a one to these bits will clear the Write Protect bits for the corresponding peripherals.
- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

10.6.2 Write Protect Set

Name: WPSET

Offset: 0x04

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		EIC	RTC	WDT	GCLK	SYSCTRL	PM	
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:7 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 6:1 – EIC, RTC, WDT, GCLK, SYSCTRL, PM: Write Protect Enable**
  0: Write-protection is disabled.
  1: Write-protection is enabled.
  Writing a zero to these bits has no effect.
  Writing a one to these bits will set the Write Protect bit for the corresponding peripherals.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

10.6.3 PAC1 Register Description

Write Protect Clear

Name: WPCLR

Offset: 0x00

Reset: 0x00000002

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					PORT	NVMCTRL	DSU	
Access	R	R	R	R	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	1	0

- **Bits 31:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:1 – PORT, NVMCTRL, DSU: Write Protect**
  0: Write-protection is disabled.
  1: Write-protection is enabled.
  Writing a zero to these bits has no effect.
  Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

Write Protect Set

Name: WPSET

Offset: 0x04

Reset: 0x00000002

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					PORT	NVMCTRL	DSU	
Access	R	R	R	R	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	1	0

- **Bits 31:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 3:1 – PORT, NVMCTRL, DSU: Write Protect**
  0: Write-protection is disabled.
  1: Write-protection is enabled.
  Writing a zero to these bits has no effect.
  Writing a one to these bits will set the Write Protect bit for the corresponding peripherals.
- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

10.6.4 PAC2 Register Description

Write Protect Clear

Name: WPCLR

Offset: 0x00

Reset: 0x00100000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
					PTC	DAC	AC	ADC
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SERCOM5	SERCOM4	SERCOM3	SERCOM2	SERCOM1	SERCOM0	EVSYS	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:20 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to reset value when this register is written. These bits will always return reset value when read.
- **Bits 19:1 – PTC, DAC, AC, ADC, TC7, TC6, TC5, TC4, TC3, TC2, TC1, TC0, SERCOM5, SERCOM4, SERCOM3, SERCOM2, SERCOM1, SERCOM0, EVSYS: Write Protect**
  0: Write-protection is disabled.
  1: Write-protection is enabled.
  Writing a zero to these bits has no effect.
  Writing a one to these bits will clear the Write Protect bit for the corresponding peripherals.
- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

Write Protect Set

Name: WPSET

Offset: 0x04

Reset: 0x00100000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
					PTC	DAC	AC	ADC
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SERCOM5	SERCOM4	SERCOM3	SERCOM2	SERCOM1	SERCOM0	EVSYS	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:20 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to reset value when this register is written. These bits will always return reset value when read.
- **Bits 19:1 – PTC, DAC, AC, ADC, TC7, TC6, TC5, TC4, TC3, TC2, TC1, TC0, SERCOM5, SERCOM4, SERCOM3, SERCOM2, SERCOM1, SERCOM0, EVSYS: Write Protect Enable**
  0: Write-protection is disabled.
  1: Write-protection is enabled.
  Writing a zero to these bits has no effect.
  Writing a one to these bits will set the Write Protect bit for the corresponding peripherals.
- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

11. Peripherals Configuration Overview

The following table shows an overview of all the peripherals in the device. The IRQ Line column shows the interrupt mapping, as described in “Nested Vector Interrupt Controller” on page 24.

The AHB and APB clock indexes correspond to the bit in the AHBMASK and APBMASK (x = A, B or C) registers in the Power Manager, while the Enabled at Reset column shows whether the peripheral clock is enabled at reset (Y) or not (N). Refer to the Power Manager AHBMASK, APBAMASK, APBBMASK and APBCMASK registers for details.

The Generic Clock Index column corresponds to the value of the Generic Clock Selection ID bits in the Generic Clock Control register (CLKCTRL.ID) in the Generic Clock Controller. Refer to the GCLK CLKCTRL register description for details.

The PAC Index column corresponds to the bit in the PACi (i = 0, 1 or 2) registers, while the Prot at Reset column shows whether the peripheral is protected at reset (Y) or not (N). Refer to “PAC – Peripheral Access Controller” on page 27 for details.

The numbers in the Events User column correspond to the value of the User Multiplexer Selection bits in the User Multiplexer register (USER.USER) in the Event System. See the USER register description and Table 22-6 for details.

The numbers in the Events Generator column correspond to the value of the Event Generator bits in the Channel register (CHANNEL.EVGEN) in the Event System. See the CHANNEL register description and Table 22-3 for details.

Table 11-1. Peripherals Configuration Overview

Peripheral Name	Base Address	IRQ Line	AHB Clock		APB Clock		Generic Clock	PAC		Events	
			Index	Enabled at Reset	Index	Enabled at Reset	Index	Index	Prot at Reset	User	Generator
HPB0 Peripheral Bridge A	0x40000000		0	Y							
PAC0	0x40000000				0	Y					
PM	0x40000400	0			1	Y		1	N		
SYSCTRL	0x40000800	1			2	Y	0: DFLL48M reference	2	N		
GCLK	0x40000C00				3	Y		3	N		
WDT	0x40001000	2			4	Y	1	4	N		
RTC	0x40001400	3			5	Y	2	5	N		1: CMP0/ALARM0 2: CMP1 3: OVF 4-11: PER0-7
EIC	0x40001800	NMI, 4			6	Y	3	6	N		12-27: EXTINT0-15
HPB1 Peripheral Bridge B	0x41000000		1	Y							
PAC1	0x41000000				0	Y					
DSU	0x41002000		3	Y	1	Y		1	Y		
NVMCTRL	0x41004000	5	4	Y	2	Y		2	N		
PORT	0x41004400				3	Y		3	N		
HPB2 Peripheral Bridge C	0x42000000		2	Y							
PAC2	0x42000000				0	N					
EVSYS	0x42000400	6			1	N	4-11: one per CHANNEL	1	N		
SERCOM0	0x42000800	7			2	N	13: CORE 12: SLOW	2	N		
SERCOM1	0x42000C00	8			3	N	14: CORE 12: SLOW	3	N		

Table 11-1. Peripherals Configuration Overview

Peripheral Name	Base Address	IRQ Line	AHB Clock		APB Clock		Generic Clock	PAC		Events	
			Index	Enabled at Reset	Index	Enabled at Reset	Index	Index	Prot at Reset	User	Generator
SERCOM2	0x42001000	9			4	N	15: CORE 12: SLOW	4	N		
SERCOM3	0x42001400	10			5	N	16: CORE 12: SLOW	5	N		
SERCOM4	0x42001800	11			6	N	17: CORE 12: SLOW	6	N		
SERCOM5	0x42001C00	12			7	N	18: CORE 12: SLOW	7	N		
TC0	0x42002000	13			8	N	19	8	N	0: TC	28: OVF 29-30: MC0-1
TC1	0x42002400	14			9	N	19	9	N	1: TC	31: OVF 32-33: MC0-1
TC2	0x42002800	15			10	N	20	10	N	2: TC	34: OVF 35-36: MC0-1
TC3	0x42002C00	16			11	N	20	11	N	3: TC	37: OVF 38-39: MC0-1
TC4	0x42003000	17			12	N	21	12	N	4: TC	40: OVF 41-42: MC0-1
TC5	0x42003400	18			13	N	21	13	N	5: TC	43: OVF 44-45: MC0-1
TC6	0x42003800	19			14	N	22	14	N	6: TC	46: OVF 47-48: MC0-1
TC7	0x42003C00	20			15	N	22	15	N	7: TC	49: OVF 50-51: MC0-1
ADC	0x42004000	21			16	Y	23	16	N	8: START 9: SYNC	52: RESRDY 53: WINMON
AC	0x42004400	22			17	N	24: DIG 25: ANA	17	N	10-11: COMP0-1	54-55: COMP0-1 56: WIN0
DAC	0x42004800	23			18	N	26	18	N	12: START	57: EMPTY
PTC	0x42004C00	24			19	N	27	19	N	13: STCONV	58: EOC 59: WCOMP

12. DSU – Device Service Unit

12.1 Overview

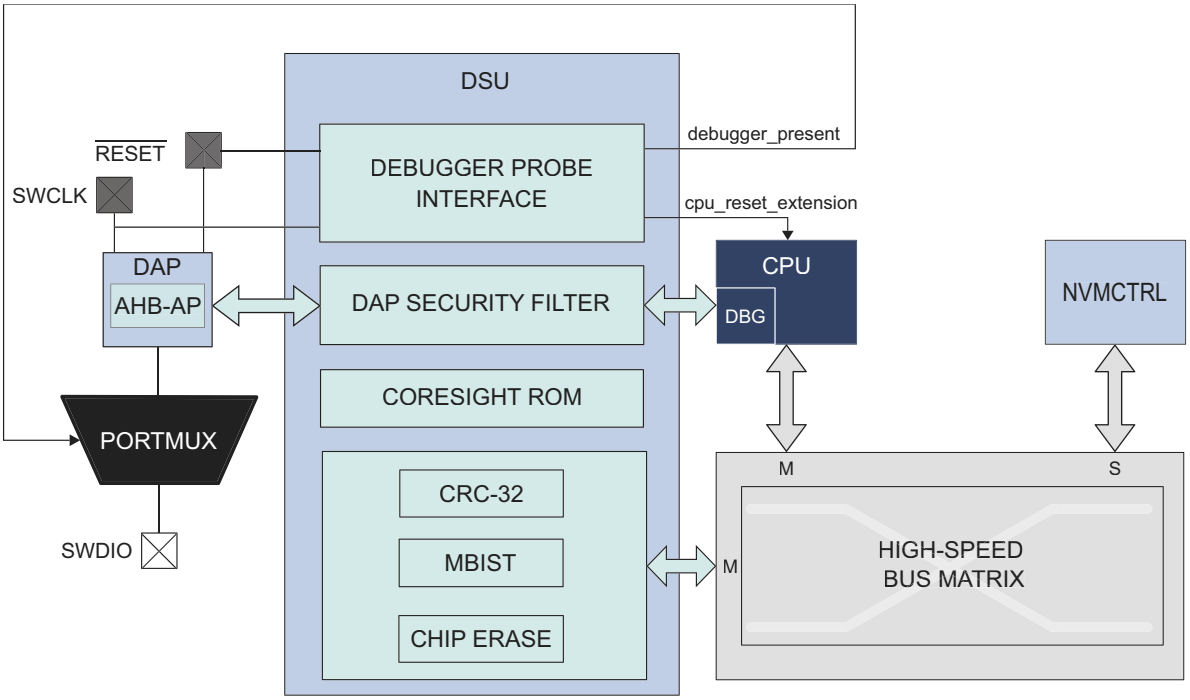
The Device Service Unit (DSU) provides a means to detect debugger probes. This enables the ARM Debug Access Port (DAP) to have control over multiplexed debug pads and CPU reset. The DSU also provides system-level services to debug adapters in an ARM debug system. It implements a CoreSight Debug ROM that provides device identification as well as identification of other debug components in the system. Hence, it complies with the ARM Peripheral Identification specification. The DSU also provides system services to applications that need memory testing, as required for IEC60730 Class B compliance, for example. The DSU can be accessed simultaneously by a debugger and the CPU, as it is connected on the High-Speed Bus Matrix. For security reasons, some of the DSU features will be limited or unavailable when the device is protected by the NVMCTRL security bit (refer to “Security Bit” on page 260).

12.2 Features

- CPU reset extension
- Debugger probe detection (Cold- and Hot-Plugging)
- Chip-Erase command and status
- 32-bit cyclic redundancy check (CRC32) of any memory accessible through the bus matrix
- ARM® CoreSight™ compliant device identification
- Two debug communications channels
- Debug access port security filter
- Onboard memory built-in self-test (MBIST)

12.3 Block Diagram

Figure 12-1. DSU Bock Diagram



12.4 Signal Description

Signal Name	Type	Description
$\overline{\text{RESET}}$	Digital Input	External reset
SWCLK	Digital Input	SW clock
SWDIO	Digital I/O	SW bidirectional data pin

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral.

12.5 Product Dependencies

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

12.5.1 I/O Lines

The SWCLK pin is by default assigned to the DSU module to allow debugger probe detection and the condition to stretch the CPU reset phase. For more information, refer to "Debugger Probe Detection" on page 38. The Hot-Plugging feature depends on the PORT configuration. If the SWCLK pin function is changed in the PORT or if the PORT_MUX is disabled, the Hot-Plugging feature is disabled until a power-reset or an external reset.

12.5.2 Power Management

The DSU will continue to operate in any sleep mode where the selected source clock is running.

Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

12.5.3 Clocks

The DSU bus clocks (CLK_DSU_APB and CLK_DSU_AHB) can be enabled and disabled in the Power Manager. For more information on the CLK_DSU_APB and CLK_DSU_AHB clock masks, refer to "PM – Power Manager" on page 100.

12.5.4 Interrupts

Not applicable.

12.5.5 Events

Not applicable.

12.5.6 Register Access Protection

All registers with write access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Debug Communication Channel 0 register (DCC0)
- Debug Communication Channel 1 register (DCC1)

Write-protection is denoted by the Write-Protection property in the register description.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

12.5.7 Analog Connections

Not applicable.

12.6 Debug Operation

12.6.1 Principle of Operation

The DSU provides basic services to allow on-chip debug using the ARM Debug Access Port and the ARM processor debug resources:

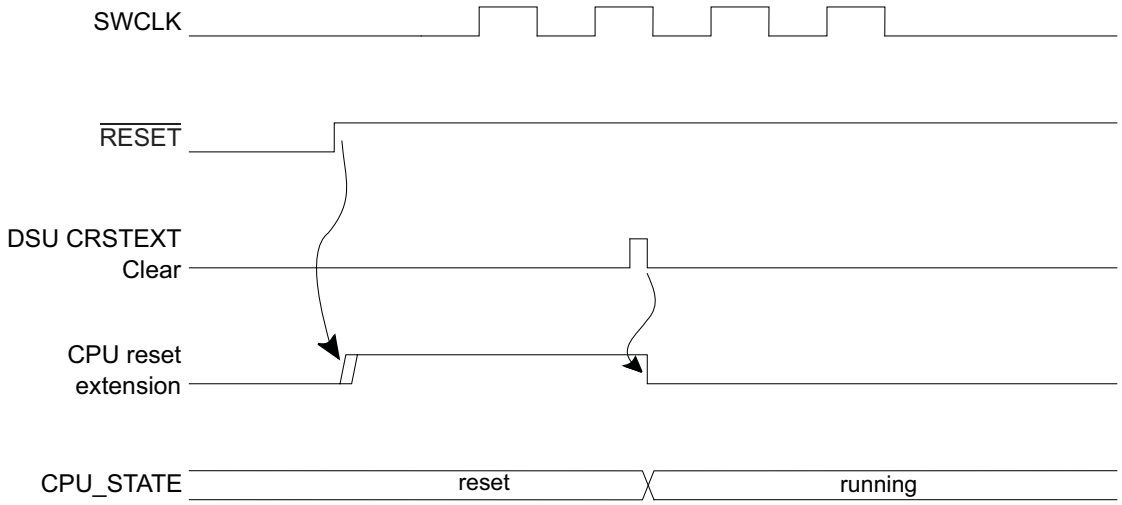
- CPU reset extension
- Debugger probe detection

For more details on the ARM debug components, refer to the ARM Debug Interface v5Architecture Specification.

12.6.2 CPU Reset Extension

"CPU reset extension" refers to the extension of the reset phase of the CPU core after the external reset is released. This ensures that the CPU is not executing code at startup while a debugger connects to the system. It is detected on a $\overline{\text{RESET}}$ release event when SWCLK is low. At startup, SWCLK is internally pulled up to avoid false detection of a debugger if SWCLK is left unconnected. When the CPU is held in the reset extension phase, the CPU Reset Extension bit (CRSTEXT) of the Status A register (STATUSA.CRSTEXT) is set. To release the CPU, write a one to STATUSA.CRSTEXT. STATUSA.CRSTEXT will then be set to zero. Writing a zero to STATUSA.CRSTEXT has no effect. For security reasons, it is not possible to release the CPU reset extension when the device is protected by the NVMCTRL security bit (refer to "Security Bit" on page 260). Trying to do so sets the Protection Error bit (PERR) of the Status A register (STATUSA.PERR).

Figure 12-2. Typical CPU Reset Extension Set and Clear Timing Diagram



12.6.3 Debugger Probe Detection

12.6.3.1 Cold-Plugging

Cold-Plugging is the detection of a debugger when the system is in reset. Cold-Plugging is detected when the CPU reset extension is requested, as described above.

12.6.3.2 Hot-Plugging

Hot-Plugging is the detection of a debugger probe when the system is not in reset. Hot-Plugging is not possible under reset because the detector is reset when POR or $\overline{\text{RESET}}$ are asserted. Hot-Plugging is active when a SWCLK falling edge is detected. The SWCLK pad is multiplexed with other functions and the user must ensure that its default function is assigned to the debug system. If the SWCLK function is changed, the Hot-Plugging feature is disabled until a power-reset or external reset occurs. Availability of the Hot-Plugging feature can be read from the Hot-Plugging Enable bit of the Status B register (STATUSB.HPE).

Figure 12-3. Hot-Plugging Detection Timing Diagram

The presence of a debugger probe is detected when either Hot-Plugging or Cold-Plugging is detected. Once detected, the Debugger Present bit of the Status B register (STATUSB.DBGPRES) is set. For security reasons, Hot-Plugging is not available when the device is protected by the NVMCTRL security bit (refer to "Security Bit" on page 260).

This detection requires that pads are correctly powered. Thus, at cold startup, this detection cannot be done until POR is released. If the device is protected, Cold-Plugging is the only way to detect a debugger probe, and so the external reset timing must be longer than the POR timing. If external reset is deasserted before POR release, the user must retry the procedure above until it gets connected to the device.

12.7 Chip-Erase

Chip-Erase consists of removing all sensitive information stored in the chip and clearing the NVMCTRL security bit (refer to "Security Bit" on page 260). Hence, all volatile memories and the flash array (including the EEPROM emulation area) will be erased. The flash auxiliary rows, including the user row, will not be erased. When the device is protected, the debugger must reset the device in order to be detected. This ensures that internal registers are reset after the protected state is removed. The Chip-Erase operation is triggered by writing a one to the Chip-Erase bit in the Control register (CTRL.CE). This command will be discarded if the DSU is protected by the Peripheral Access Controller (PAC). Once issued, the module clears volatile memories prior to erasing the flash array. To ensure that the Chip-Erase operation is completed, check the Done bit of the Status A register (STATUSA.DONE). The Chip-Erase operation depends on clocks and power management features that can be altered by the CPU. For that reason, it is recommended to issue a Chip-Erase after a Cold-Plugging procedure to ensure that the device is in a known and safe state.

The recommended sequence is as follows:

1. Issue the Cold-Plugging procedure (refer to "Cold-Plugging" on page 38). The device then:
    1. Detects the debugger probe
    2. Holds the CPU in reset
2. Issue the Chip-Erase command by writing a one to CTRL.CE. The device then:
    1. Clears the system volatile memories
    2. Erases the whole flash array (including the EEPROM emulation area, not including auxiliary rows)
    3. Erases the lock row, removing the NVMCTRL security bit protection
3. Check for completion by polling STATUSA.DONE (read as one when completed).
4. Reset the device to let the NVMCTRL update fuses.

12.8 Programming

Programming of the flash or RAM memories is available when the device is not protected by the NVMCTRL security bit (refer to "Security Bit" on page 260).

1. At power up, $\overline{\text{RESET}}$ 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 (refer to "Power-On Reset (POR) Characteristics" on page 567). The sys-

tem continues to be held in this static state until the internally regulated supplies have reached a safe operating state.

2. The 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.
3. The debugger maintains a low level on SWCLK. Releasing $\overline{\text{RESET}}$ results in a debugger Cold-Plugging procedure.
4. The debugger generates a clock signal on the SWCLK pin, the Debug Access Port (DAP) receives a clock.
5. The CPU remains in reset due to the Cold-Plugging procedure; meanwhile, the rest of the system is released.
6. A Chip-Erase is issued to ensure that the flash is fully erased prior to programming.
7. Programming is available through the AHB-AP.
8. After operation is completed, the chip can be restarted either by asserting $\overline{\text{RESET}}$, toggling power or writing a one to the Status A register CPU Reset Phase Extension bit (STATUSA.CRSTEXT). Make sure that the SWCLK pin is high when releasing $\overline{\text{RESET}}$ to prevent extending the CPU reset.

12.9 Intellectual Property Protection

Intellectual property protection consists of restricting access to internal memories from external tools when the device is protected, and is accomplished by setting the NVMCTRL security bit (refer to “Security Bit” on page 260). This protected state can be removed by issuing a Chip-Erase (refer to “Chip-Erase” on page 39). When the device is protected, read/write accesses using the AHB-AP are limited to the DSU address range and DSU commands are restricted.

The DSU implements a security filter that monitors the AHB transactions generated by the ARM AHB-AP inside the DAP. If the device is protected, then AHB-AP read/write accesses outside the DSU external address range are discarded, causing an error response that sets the ARM AHB-AP sticky error bits (refer to the ARM Debug Interface v5 Architecture Specification on http://www.arm.com).

The DSU is intended to be accessed either:

- Internally from the CPU, without any limitation, even when the device is protected
- Externally from a debug adapter, with some restrictions when the device is protected

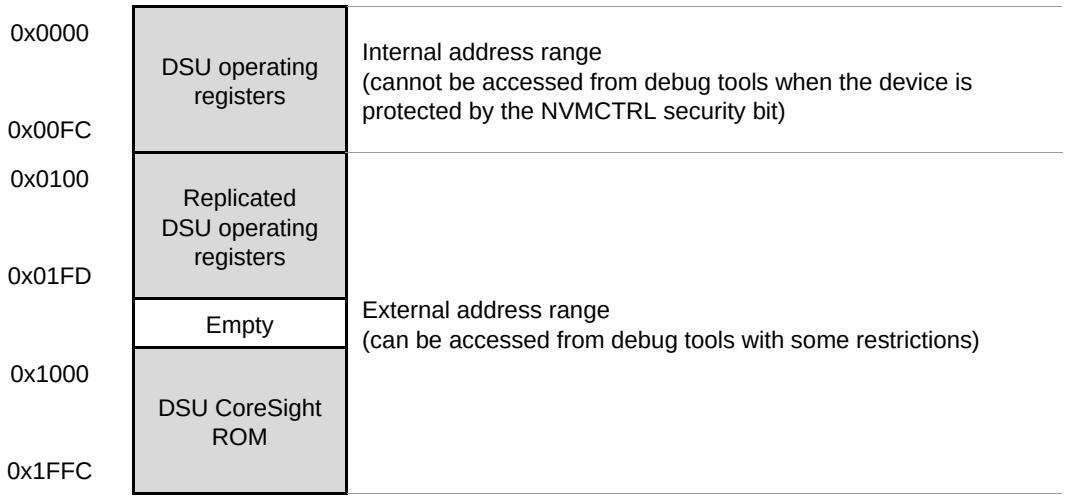
For security reasons, DSU features have limitations when used from a debug adapter. To differentiate external accesses from internal ones, the first 0x100 bytes of the DSU register map have been replicated at offset 0x100:

- The first 0x100 bytes form the internal address range
- The next 0x100 bytes form the external address range

When the device is protected, the DAP can only issue MEM-AP accesses in the DSU address range limited to the 0x100-0x2000 offset range.

The DSU operating registers are located in the 0x00-0xFF area and remapped in 0x100-0x1FF to differentiate accesses coming from a debugger and the CPU. If the device is protected and an access is issued in the region 0x100-0x1FF, it is subject to security restrictions. For more information, refer to Table 12-1.

Figure 12-4. APB Memory Mapping



Some features not activated by APB transactions are not available when the device is protected:

Table 12-1. Feature Availability Under Protection

Features	Availability When the Device is Protected
CPU reset extension	Yes
Debugger Cold-Plugging	Yes
Debugger Hot-Plugging	No

12.10 Device Identification

Device identification relies on the ARM CoreSight component identification scheme, which allows the chip to be identified as an ATMEL device implementing a DSU. The DSU contains identification registers to differentiate the device.

12.10.1 CoreSight Identification

A system-level ARM CoreSight ROM table is present in the device to identify the vendor and the chip identification method. Its address is provided in the MEM-AP BASE register inside the ARM Debug Access Port. The CoreSight ROM implements a 64-bit conceptual ID composed as follows from the PID0 to PID7 CoreSight ROM Table registers:

Figure 12-5. Conceptual 64-Bit Peripheral ID

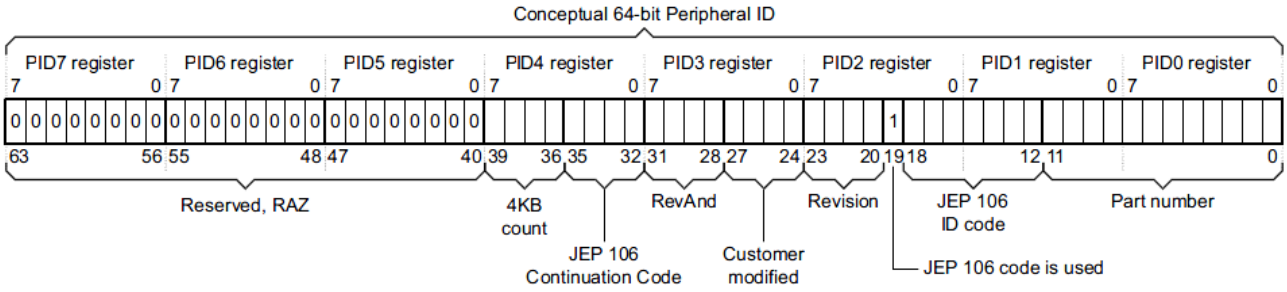


Table 12-2. Conceptual 64-Bit Peripheral ID Bit Descriptions

Field	Size	Description	Location
JEP-106 CC code	4	Atmel continuation code: 0x0	PID4
JEP-106 ID code	7	Atmel device ID: 0x1F	PID1+PID2
4KB count	4	Indicates that the CoreSight component is a ROM: 0x0	PID4
RevAnd	4	Not used; read as 0	PID3
CUSMOD	4	Not used; read as 0	PID3
PARTNUM	12	Contains 0xCD0 to indicate that DSU is present	PID0+PID1
REVISION	4	DSU revision (starts at 0x0 and increments by 1 at both major and minor revisions). Identifies DSU identification method variants. If 0x0, this indicates that device identification can be completed by reading the Device Identification register (DID)	PID3

For more information, refer to the ARM Debug Interface Version 5 Architecture Specification.

12.10.2 DSU Chip Identification Method:

The DSU DID register identifies the device by implementing the following information:

- Processor identification
- Family identification
- Subfamily identification
- Device select

12.11 Functional Description

12.11.1 Principle of Operation

The DSU provides memory services such as CRC32 or MBIST that require almost the same interface. Hence, the Address, Length and Data registers are shared. They must be configured first; then a command can be issued by writing the Control register. When a command is ongoing, other commands are discarded until the current operation is completed. Hence, the user must wait for the STATUSA.DONE bit to be set prior to issuing another one.

12.11.2 Basic Operation

12.11.2.1 Initialization

The module is enabled by enabling its clocks. For more details, refer to “Clocks” on page 37. The DSU registers can be write-protected. Refer to “PAC – Peripheral Access Controller” on page 27.

12.11.2.2 Operation from a debug adapter

Debug adapters should access the DSU registers in the external address range 0x100 – 0x2000. If the device is protected by the NVMCTRL security bit (refer to “Security Bit” on page 260), accessing the first 0x100 bytes causes the system to return an error (refer to “Intellectual Property Protection” on page 40).

12.11.2.3 Operation from the CPU

There are no restrictions when accessing DSU registers from the CPU. However, the user should access DSU registers in the internal address range (0x0 – 0x100) to avoid external security restrictions (refer to “Intellectual Property Protection” on page 40).

12.11.3 32-bit Cyclic Redundancy Check (CRC32)

The DSU unit provides support for calculating a cyclic redundancy check (CRC32) value for a memory area (including flash and AHB RAM).

When the CRC32 command is issued from:

- The internal range, the CRC32 can be operated at any memory location
- The external range, the CRC32 operation is restricted; DATA, ADDR and LENGTH values are forced (see below)

Table 12-3. AMOD Bit Descriptions when Operating CRC32

AMOD[1:0]	Short Name	External Range Restrictions
0	ARRAY	CRC32 is restricted to the full flash array area (EEPROM emulation area not included) DATA forced to 0xFFFFFFFF before calculation (no seed)
1	EEPROM	CRC32 of the whole EEPROM emulation area DATA forced to 0xFFFFFFFF before calculation (no seed)
2-3	Reserved	

The algorithm employed is the industry standard CRC32 algorithm using the generator polynomial 0xEDB88320 (reversed representation).

12.11.3.1 Starting CRC32 Calculation

CRC32 calculation for a memory range is started after writing the start address into the Address register (ADDR) and the size of the memory range into the Length register (LENGTH). Both must be word-aligned.

The initial value used for the CRC32 calculation must be written to the Data register. This value will usually be 0xFFFFFFFF, but can be, for example, the result of a previous CRC32 calculation if generating a common CRC32 of separate memory blocks.

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

If the device is in protected state by the NVMCTRL security bit (refer to “Security Bit” on page 260), it is only possible to calculate the CRC32 of the whole flash array. In most cases, this area will be the entire onboard non-volatile memory. The Address, Length and Data registers will be forced to predefined values once the CRC32 operation is started, and values written by the user are ignored. This allows the user to verify the contents of a protected device.

The actual test is started by writing a one in the 32-bit Cyclic Redundancy Check bit of the Control register (CTRL.CRC). A running CRC32 operation can be canceled by resetting the module (writing a one to CTRL.SWRST).

12.11.3.2 Interpreting the Results

The user should monitor the Status A register. When the operation is completed, STATUSA.DONE is set. Then the Bus Error bit of the Status A register (STATUSA.BERR) must be read to ensure that no bus error occurred.

12.11.4 Debug Communication Channels

The Debug Communication Channels (DCCO and DCC1) consist of a pair of registers with associated handshake logic, accessible by both CPU and debugger even if the device is protected by the NVMCTRL security bit (refer to “Security Bit” on page 260). The registers can be used to exchange data between the CPU and the debugger, during run time as well as in debug mode. This enables the user to build a custom debug protocol using only these registers. The DCC0 and DCC1 registers are accessible when the protected state is active. When the device is protected, however, it is not possible to connect a debugger while the CPU is running (STATUSA.CRSTEXT is not writable and the CPU is held under reset). Dirty bits in the status registers indicate whether a new value has been written in DCC0 or DCC1. These bits,DCC0D and DCC1D, are located in the STATUSB registers. They are automatically set on write and cleared on

read. The DCC0 and DCC1 registers are shared with the onboard memory testing logic (MBIST). Accordingly, DCC0 and DCC1 must not be used while performing MBIST operations.

12.11.5 Testing of Onboard Memories (MBIST)

The DSU implements a feature for automatic testing of memory also known as MBIST. This is primarily intended for production test of onboard memories. MBIST cannot be operated from the external address range when the device is protected by the NVMCTRL security bit (refer to “Security Bit” on page 260). If a MBIST command is issued when the device is protected, a protection error is reported in the Protection Error bit in the Status A register (STATUSA.PERR).

1. Algorithm

   The algorithm used for testing is a type of March algorithm called "March LR". This algorithm is able to detect a wide range of memory defects, while still keeping a linear run time. The algorithm is:

   1. Write entire memory to 0, in any order.
   2. Bit for bit read 0, write 1, in descending order.
   3. Bit for bit read 1, write 0, read 0, write 1, in ascending order.
   4. Bit for bit read 1, write 0, in ascending order.
   5. Bit for bit read 0, write 1, read 1, write 0, in ascending order.
   6. Read 0 from entire memory, in ascending order.

   The specific implementation used has a run time of O(14n) where n is the number of bits in the RAM. The detected faults are:

   - Address decoder faults
   - Stuck-at faults
   - Transition faults
   - Coupling faults
   - Linked Coupling faults
   - Stuck-open faults

2. Starting MBIST

   To test a memory, you need to write the start address of the memory to the ADDR.ADDR bit group, and the size of the memory into the Length register. See “Physical Memory Map” on page 20 to know which memories are available, and which address they are at.

   For best test coverage, an entire physical memory block should be tested at once. It is possible to test only a subset of a memory, but the test coverage will then be somewhat lower.

   The actual test is started by writing a one to CTRL.MBIST. A running MBIST operation can be canceled by writing a one to CTRL.SWRST.

3. Interpreting the Results

   The tester should monitor the STATUSA register. When the operation is completed, STATUSA.DONE is set. There are three different modes:

   - ADDR.AMOD=0: exit-on-error (default)

   In this mode, the algorithm terminates either when a fault is detected or on successful completion. In both cases, STATUSA.DONE is set. If an error was detected, STATUSA.FAIL will be set. User then can read the DATA and ADDR registers to locate the fault. Refer to “Locating Errors” on page 44.

   - ADDR.AMOD=1: pause-on-error

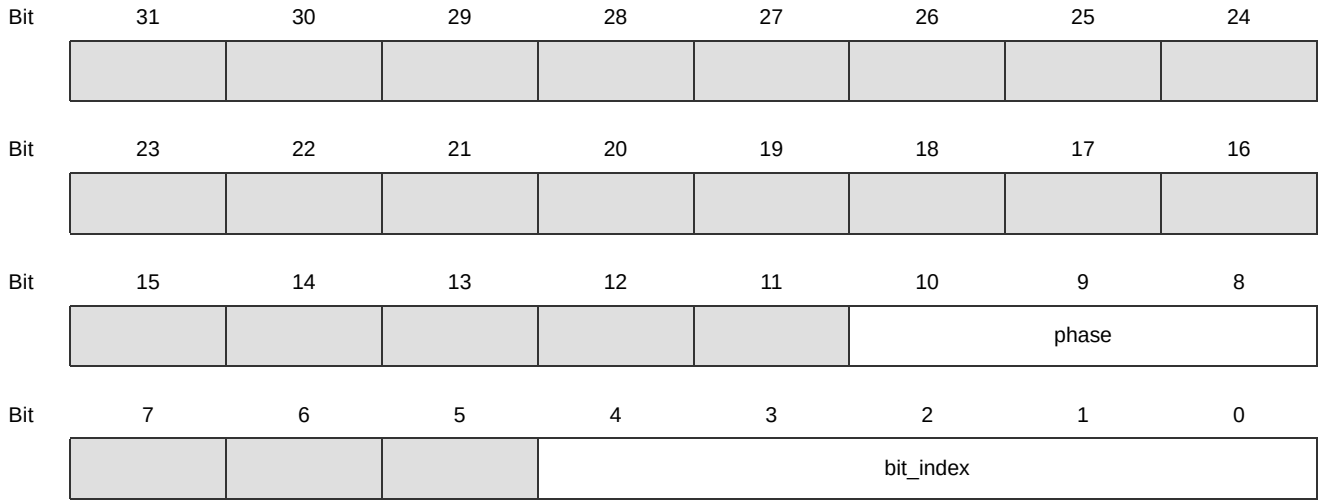
   In this mode, the MBIST algorithm is paused when an error is detected. In such a situation, only STATUSA.FAIL is asserted. The state machine waits for user to clear STATUSA.FAIL by writing a one in STATUSA.FAIL to resume. Prior to resuming, user can read the DATA and ADDR registers to locate the fault. Refer to “Locating Errors” on page 44.

4. Locating Errors

   If the test stops with STATUSA.FAIL set, one or more bits failed the test. The test stops at the first detected error. The position of the failing bit can be found by reading the following registers:

- ADDR: Address of the word containing the failing bit.
- DATA: contains data to identify which bit failed, and during which phase of the test it failed. The DATA register will in this case contains the following bit groups:

Table 12-4. DATA bits Description When MBIST Operation Returns An Error



- bit_index: contains the bit number of the failing bit
- phase: indicates which phase of the test failed and the cause of the error. See Table 12-5 on page 45.

Table 12-5. MBIST Operation Phases

Phase	Test Actions
0	Write all bits to zero. This phase cannot fail.
1	Read 0, write 1, increment address
2	Read 1, write 0
3	Read 0, write 1, decrement address
4	Read 1, write 0, decrement address
5	Read 0, write 1
6	Read 1, write 0, decrement address
7	Read all zeros. bit_index is not used

12.11.6 System Services Availability When Accessed Externally

External access: Access performed in the DSU address offset 0x200-0x1FFF range.

Internal access: Access performed in the DSU address offset 0x0-0x100 range.

Table 12-6. Available Features When Operated From The External Address Range

Features	Availability From The External Address Range
Chip-Erase command and status	Yes
CRC32	Yes, only full array or full EEPROM
CoreSight Compliant Device identification	Yes
Debug communication channels	Yes
Testing of onboard memories (MBIST)	Yes
STATUSA.CRSTEXT clearing	No (STATUSA.PERR is set when attempting to do so)

12.12 Register Summary

Table 12-7. Register Summary

Offset	Name	Bit Pos								
0x0000	CTRL	7:0				CE	MBIST	CRC		SWRST
0x0001	STATUSA	7:0				PERR	FAIL	BERR	CRSTEXT	DONE
0x0002	STATUSB	7:0				HPE	DCCD1	DCCD0	DBGPRES	PROT
0x0003	Reserved									
0x0004	ADDR	7:0	ADDR[5:0]							
0x0005		15:8	ADDR[13:6]							
0x0006		23:16	ADDR[21:14]							
0x0007		31:24	ADDR[29:22]							
0x0008	LENGTH	7:0	LENGTH[5:0]							
0x0009		15:8	LENGTH[13:6]							
0x000A		23:16	LENGTH[21:14]							
0x000B		31:24	LENGTH[29:22]							
0x000C	DATA	7:0	DATA[7:0]							
0x000D		15:8	DATA[15:8]							
0x000E		23:16	DATA[23:16]							
0x000F		31:24	DATA[31:24]							
0x0010	DCC0	7:0	DATA[7:0]							
0x0011		15:8	DATA[15:8]							
0x0012		23:16	DATA[23:16]							
0x0013		31:24	DATA[31:24]							
0x0014	DCC1	7:0	DATA[7:0]							
0x0015		15:8	DATA[15:8]							
0x0016		23:16	DATA[23:16]							
0x0017		31:24	DATA[31:24]							
0x0018	DID	7:0	DEVSEL[7:0]							
0x0019		15:8	DIE[3:0]				REVISION[3:0]			
0x001A		23:16	SERIES[7:0]							
0x001C		31:24	PROCESSOR[3:0]				FAMILY[3:0]			
0x001D	Reserved									
...	...									
0x00FF	Reserved									
0x0100-0x01FF	External address range: Replicates the 0x00:0x1C address range, Gives access to the same resources but with security restrictions when the device is protected. This address range is the only one accessible externally (using the ARM DAP) when the device is protected.									

0x1000	ENTRY0	7:0								FMT	EPRES	
0x1001		15:8	ADDOFF[3:0]									
0x1002		23:16	ADDOFF[11:4]									
0x1003		31:24	ADDOFF[19:12]									
0x1004	ENTRY1	7:0								FMT	EPRES	
0x1005		15:8	ADDOFF[3:0]									
0x1006		23:16	ADDOFF[11:4]									
0x1007		31:24	ADDOFF[19:12]									
0x1008	END	7:0	END[7:0]									
0x1009		15:8	END[15:8]									
0x100A		23:16	END[23:16]									
0x100B		31:24	END[31:24]									
0x1FCC	MEMTYPE	7:0									SMEMP	
0x1FCD		15:8										
0x1FCE		23:16										
0x1FCF		31:24										
0x1FD0	PID4	7:0	FKBC[3:0]				JEPCC[3:0]					
0x1FD1		15:8										
0x1FD2		23:16										
0x1FD3		31:24										
0x1FD4	Reserved											
...	...											
0x1FDF	Reserved											
0x1FE0	PID0	7:0	PARTNBL[7:0]									
0x1FE1		15:8										
0x1FE2		23:16										
0x1FE3		31:24										
0x1FE4	PID1	7:0	JEPIDCL[3:0]				PARTNBH[3:0]					
0x1FE5		15:8										
0x1FE6		23:16										
0x1FE7		31:24										
0x1FE8	PID2	7:0	REVISION[3:0]				JEPU	JEPIDCH[2:0]				
0x1FE9		15:8										
0x1FEA		23:16										
0x1FEB		31:24										
0x1FEC	PID3	7:0	REVAND[3:0]				CUSMOD[3:0]					
0x1FED		15:8										
0x1FEE		23:16										
0x1FEF		31:24										

0x1FF0	CID0	7:0	PREAMBLEB0[7:0]							
0x1FF1		15:8								
0x1FF2		23:16								
0x1FF3		31:24								
0x1FF4	CID1	7:0	CCLASS[3:0]				PREAMBLE[3:0]			
0x1FF5		15:8								
0x1FF6		23:16								
0x1FF7		31:24								
0x1FF8	CID2	7:0	PREAMBLEB2[7:0]							
0x1FF9		15:8								
0x1FFA		23:16								
0x1FFB		31:24								
0x1FFC	CID3	7:0	PREAMBLEB3[7:0]							
0x1FFD		15:8								
0x1FFE		23:16								
0x1FFF		31:24								

12.13 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 37 for details.

12.13.1 Control

Name: CTRL

Offset: 0x0000

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				CE	MBIST	CRC		SWRST
Access	R	R	R	W	W	W	R	W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 4 – CE: Chip Erase**
  Writing a zero to this bit has no effect.
  Writing a one to this bit starts the Chip-Erase operation.
- **Bit 3 – MBIST: Memory Built-In Self-Test**
  Writing a zero to this bit has no effect.
  Writing a one to this bit starts the memory BIST algorithm.
- **Bit 2 – CRC: 32-bit Cyclic Redundancy Check**
  Writing a zero to this bit has no effect.
  Writing a one to this bit starts the cyclic redundancy check algorithm.
- **Bit 1 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written.
- **Bit 0 – SWRST: Software Reset**
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets the module.

12.13.2 Status A

Name: STATUSA

Offset: 0x0001

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				PERR	FAIL	BERR	CRSTEXT	DONE
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 4 – PERR: Protection Error**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Protection Error bit.
  This bit is set when a command that is not allowed in protected state is issued.
- **Bit 3 – FAIL: Failure**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Failure bit.
  This bit is set when a DSU operation failure is detected.
- **Bit 2 – BERR: Bus Error**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Bus Error bit.
  This bit is set when a bus error is detected.
- **Bit 1 – CRSTEXT: CPU Reset Phase Extension**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the CPU Reset Phase Extension bit.
  This bit is set when a debug adapter Cold-Plugging is detected, which extends the CPU reset phase.
- **Bit 0 – DONE: Done**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Done bit.
  This bit is set when a DSU operation is completed.

12.13.3 Status B

Name: STATUSB

Offset: 0x0002

Reset: 0x1X

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				HPE	**DCCD1**	**DCCD0**	**DBGPRES**	**PROT**
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	X	X

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – HPE: Hot-Plugging Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.
  This bit is set when Hot-Plugging is enabled.
  This bit is cleared when Hot-Plugging is disabled. This is the case when the SWCLK function is changed. Only a power-reset or a external reset can set it again.

- **Bits 3:2 – DCCDx [x=1..0]: Debug Communication Channel x Dirty**
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.
  This bit is set when DCCx is written.
  This bit is cleared when DCCx is read.

- **Bit 1 – DBGPRES: Debugger Present**
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.
  This bit is set when a debugger probe is detected.
  This bit is never cleared.

- **Bit 0 – PROT: Protected**
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.
  This bit is set at powerup when the device is protected.
  This bit is never cleared.

12.13.4 Address

Name: ADDR

Offset: 0x0004

Reset: 0x00000000

Property: Write-Protected

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

- **Bits 31:2 – ADDR[29:0]: Address**
  Initial word start address needed for memory operations.

- **Bits 1:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

12.13.5 Length

Name: LENGTH

Offset: 0x0008

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	LENGTH[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	LENGTH[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LENGTH[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LENGTH[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:2 – LENGTH[29:0]: Length**
  Length in words needed for memory operations.

- **Bits 1:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

12.13.6 Data

Name: DATA

Offset: 0x000C

Reset: 0x00000000

Property: Write-Protected

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

- **Bits 31:0 – DATA[31:0]: Data**
  Memory operation initial value or result value.

12.13.7 Debug Communication Channel n

Name: DCCn

Offset: 0x0010+n*0x4 [n=0..1]

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
	DATA[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DATA[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DATA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – DATA[31:0]: Data**
  Data register.

12.13.8 Device Identification

Name: DID

Offset: 0x0018

Reset: 0x1000XXXX

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	PROCESSOR[3:0]				FAMILY[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SERIES[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DIE[3:0]				REVISION[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

Bit	7	6	5	4	3	2	1	0
	DEVSEL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

The information in this register is related to the ordering code. Refer to the “Ordering Information” on page 4 for details.

- **Bits 31:28 – PROCESSOR[3:0]: Processor**
  The value of this field defines the processor used on the device. For this device, the value of this field is 0x1, corresponding to the ARM Cortex-M0+ processor.
- **Bits 27:24 – FAMILY[3:0]: Product Family**
  The value of this field corresponds to the Product Family part of the ordering code. For this device, the value of this field is 0x0, corresponding to the SAM D family of base line microcontrollers.
- **Bits 23:16 – SERIES[7:0]: Product Series**
  The value of this field corresponds to the Product Series part of the ordering code. For this device, the value of this field is 0x00, corresponding to a product with the Cortex-M0+ processor and a basic feature set.
- **Bits 15:12 – DIE[3:0]: Die Identification**
  Identifies the die in the family.
- **Bits 11:8 – REVISION[3:0]: Revision**

Identifies the die revision number.

- **Bits 7:0 – DEVSEL[7:0]: Device Selection**
  DEVSEL is used to identify a device within a product family and product series. The value corresponds to the Flash memory density, pin count and device variant parts of the ordering code. Refer to Table 12-8. for details.

Table 12-8. Device Selection

DEVSEL	Device	Flash	RAM	Pincount
0x0	SAMD20J18A	256KB	32KB	64
0x1	SAMD20J17A	128KB	16KB	64
0x2	SAMD20J16A	64KB	8KB	64
0x3	SAMD20J15A	32KB	4KB	64
0x4	SAMD20J14A	16KB	2KB	64
0x5	SAMD20G18A	256KB	32KB	48
0x6	SAMD20G17A	128KB	16KB	48
0x7	SAMD20G16A	64KB	8KB	48
0x8	SAMD20G15A	32KB	4KB	48
0x9	SAMD20G14A	16KB	2KB	48
0xA	Reserved			
0xB	SAMD20E17A	128KB	16KB	32
0xC	SAMD20E16A	64KB	8KB	32
0xD	SAMD20E15A	32KB	4KB	32
0xE	SAMD20E14A	16KB	2KB	32
0xF-0xFF	Reserved			

12.13.9 CoreSight ROM Table Entry n

Name: ENTRYn

Offset: 0x1000+n*0x4 [n=0..1]

Reset: 0xXXXXX00X

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	ADDOFF[19:12]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

Bit	23	22	21	20	19	18	17	16
	ADDOFF[11:4]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

Bit	15	14	13	12	11	10	9	8
	ADDOFF[3:0]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	0	0	0	0

Bit	7	6	5	4	3	2	1	0
							FMT	EPRES
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	1	X

- **Bits 31:12 – ADDOFF[19:0]: Address Offset**
  The base address of the component, relative to the base address of this ROM table.
- **Bits 11:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 1 – FMT: Format**
  Always read as one, indicates a 32-bit ROM table.
- **Bit 0 – EPRES: Entry Present**
  This bit indicates whether an entry is present at this location in the ROM table.
  This bit is set at powerup if the device is not protected indicating that the entry is not present.
  This bit is cleared at powerup if the device is not protected indicating that the entry is present.

12.13.10 CoreSight ROM Table End

Name: END

Offset: 0x1008

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
	END[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	END[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	END[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	END[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – END[31:0]: End Marker**
  Indicates the end of the CoreSight ROM table entries.

12.13.11 Coresight ROM Table Memory Type

Name: MEMTYPE

Offset: 0x1FCC

Reset: 0x0000000X

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								SMEMP
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	X

- **Bits 31:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – SMEMP: System Memory Present**
  This bit indicates whether system memory is present on the bus that connects to the ROM table.
  This bit is set at powerup if the device is not protected indicating that the system memory is accessible from a debug adapter.
  This bit is cleared at powerup if the device is protected indicating that the system memory is not accessible from a debug adapter.

12.13.12 Peripheral Identification 4

Name: PID4

Offset: 0x1FD0

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FKBC[3:0]				JEPCC[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:4 – FKBC[3:0]: 4KB Count**
  These bits will always return zero when read, indicating that this debug component occupies one 4KB block.
- **Bits 3:0 – JEPCC[3:0]: JEP-106 Continuation Code**
  These bits will always return zero when read, indicating a Atmel device.

12.13.13 Peripheral Identification 0

Name: PID0

Offset: 0x1FE0

Reset: 0x000000D0

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PARTNBL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	0	1	0	0	0	0

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:0 – PARTNBL[7:0]: Part Number Low**
  These bits will always return 0xD0 when read, indicating that this device implements a DSU module instance.

12.13.14 Peripheral Identification 1

Name: PID1

Offset: 0x1FE4

Reset: 0x000000FC

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	JEPIDCL[3:0]				PARTNBH[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	0	0

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:4 – JEPIDCL[3:0]: Low part of the JEP-106 Identity Code**
  These bits will always return 0xF when read, indicating a Atmel device (Atmel JEP-106 identity code is 0x1F).
- **Bits 3:0 – PARTNBH[3:0]: Part Number High**
  These bits will always return 0xC when read, indicating that this device implements a DSU module instance.

12.13.15 Peripheral Identification 2

Name: PID2

Offset: 0x1FE8

Reset: 0x00000009

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	REVISION[3:0]				JEPU	JEPIDCH[2:0]		
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	0	0	1

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:4 – REVISION[3:0]: Revision Number**
  Revision of the peripheral. Starts at 0x0 and increments by one at both major and minor revisions.
- **Bit 3 – JEPU: JEP-106 Identity Code is used**
  This bit will always return one when read, indicating that JEP-106 code is used.
- **Bits 2:0 – JEPIDCH[2:0]: JEP-106 Identity Code High**
  These bits will always return 0x1 when read, indicating an Atmel device (Atmel JEP-106 identity code is 0x1F).

12.13.16 Peripheral Identification 3

Name: PID3

Offset: 0x1FEC

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	REVAND[3:0]				CUSMOD[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:4 – REVAND[3:0]: Revision Number**
  These bits will always return 0x0 when read.
- **Bits 3:0 – CUSMOD[3:0]: ARM CUSMOD**
  These bits will always return 0x0 when read.

12.13.17 Component Identification 0

Name: CID0

Offset: 0x1FF0

Reset: 0x0000000D

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PREAMBLEB0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	1	0	1

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:0 – PREAMBLEB0[7:0]: Preamble Byte 0**
  These bits will always return 0xD when read.

12.13.18 Component Identification 1

Name: CID1

Offset: 0x1FF4

Reset: 0x00000010

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CCLASS[3:0]				PREAMBLE[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	0	0

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:4 – CCLASS[3:0]: Component Class**
  These bits will always return 0x1 when read indicating that this ARM CoreSight component is ROM table (refer to the ARM Debug Interface v5 Architecture Specification at http://www.arm.com).
- **Bits 3:0 – PREAMBLE[3:0]: Preamble**
  These bits will always return 0x0 when read.

12.13.19 Component Identification 2

Name: CID2

Offset: 0x1FF8

Reset: 0x00000005

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PREAMBLEB2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	1

- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:0 – PREAMBLEB2[7:0]: Preamble Byte 2**
  These bits will always return 0x05 when read.

12.13.20 Component Identification 3

Name: CID3

Offset: 0x1FFC

Reset: 0x000000B1

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PREAMBLEB3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	1	0	0	0	1

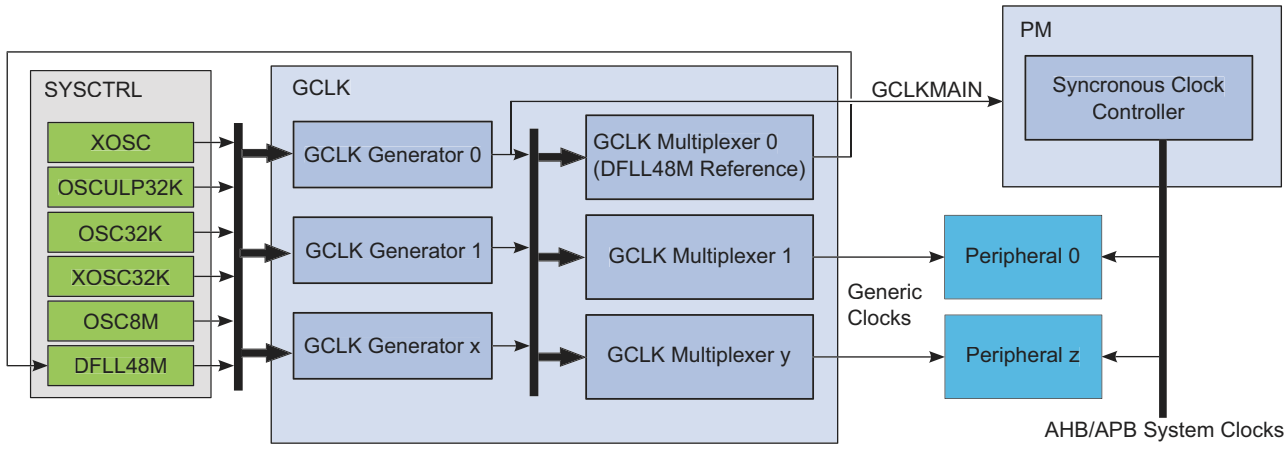
- **Bits 31:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:0 – PREAMBLEB3[7:0]: Preamble Byte 3**
  These bits will always return 0xB1 when read.

13. Clock System

This chapter only aims to summarize the clock distribution and terminology in the SAM D20 device. It will not explain every detail of its configuration. For in-depth documentation, see the referenced module chapters.

13.1 Clock Distribution

Figure 13-1. Clock distribution

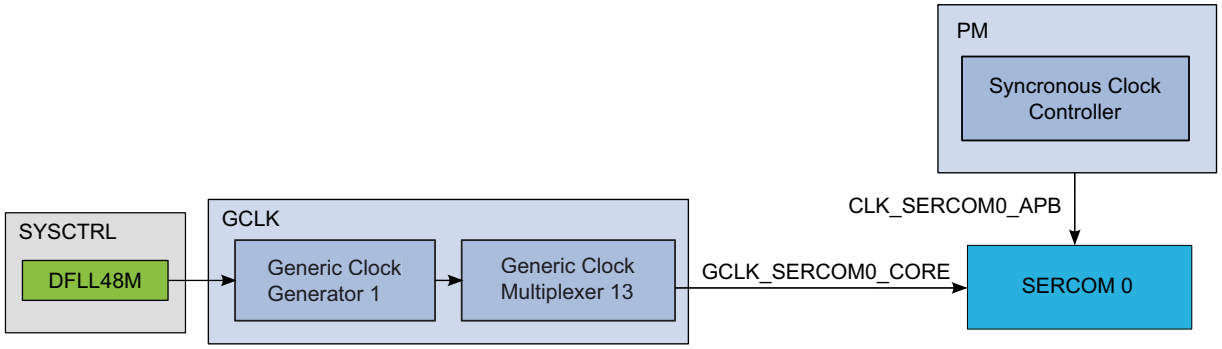


The clock system on the SAM D20 consists of:

- ***Clock sources***, controlled by SYSCTRL
  - A Clock source is the base clock signal used in the system. Example clock sources are the internal 8MHz oscillator (OSC8M), External crystal oscillator (XOSC) and the Digital frequency locked loop (DFLL48M).
- ***Generic Clock Controller (GCLK)*** which controls the clock distribution system, made up of:
  - ***Generic Clock generators****:* A programmable prescaler, that can use any of the system clock sources as its source clock. The Generic Clock Generator 0, also called GCLKMAIN, is the clock feeding the Power Manager used to generate synchronous clocks.
  - ***Generic Clocks***: Typically the clock input of a peripheral on the system. The generic clocks, through the Generic Clock Multiplexer, can use any of the Generic Clock generators as its clock source. Multiple instances of a peripheral will typically have a separate generic clock for each instance. The DFLL48M clock input (when multiplying another clock source) is generic clock 0.
- **Power Manager (PM)**
  - The PM controls synchronous clocks on the system. This includes the CPU, bus clocks (APB, AHB) as well as the synchronous (to the CPU) user interfaces of the peripherals. It contains clock masks that can turn on/off the user interface of a peripheral as well as prescalers for the CPU and bus clocks.

Figure 13-2 shows an example where SERCOM0 is clocked by the DFLL48M in open loop mode. The DFLL48M is enabled, the Generic Clock Generator 1 uses the DFLL48M as its clock source, and the generic clock 13, also called GCLK_SERCOM0_CORE, that is connected to SERCOM0 uses generator 1 as its source. The SERCOM0 interface, clocked by CLK_SERCOM0_APB, has been unmasked in the APBC Mask register in the PM.

Figure 13-2. Example of SERCOM clock



13.2 Synchronous and Asynchronous Clocks

As the CPU and the peripherals can be clocked from different clock sources, possibly with widely different clock speeds, some peripheral accesses by the CPU needs to be synchronized between the different clock domains. In these cases the peripheral includes a SYNCBUSY status flag that can be used to check if a sync operation is in progress. As the nature of the synchronization might vary between different peripherals, detailed description for each peripheral can be found in the sub-chapter “synchronization” for each peripheral where this is necessary.

In the datasheet references to synchronous clocks are referring to the CPU and bus clocks, while asynchronous clocks are clock generated by generic clocks.

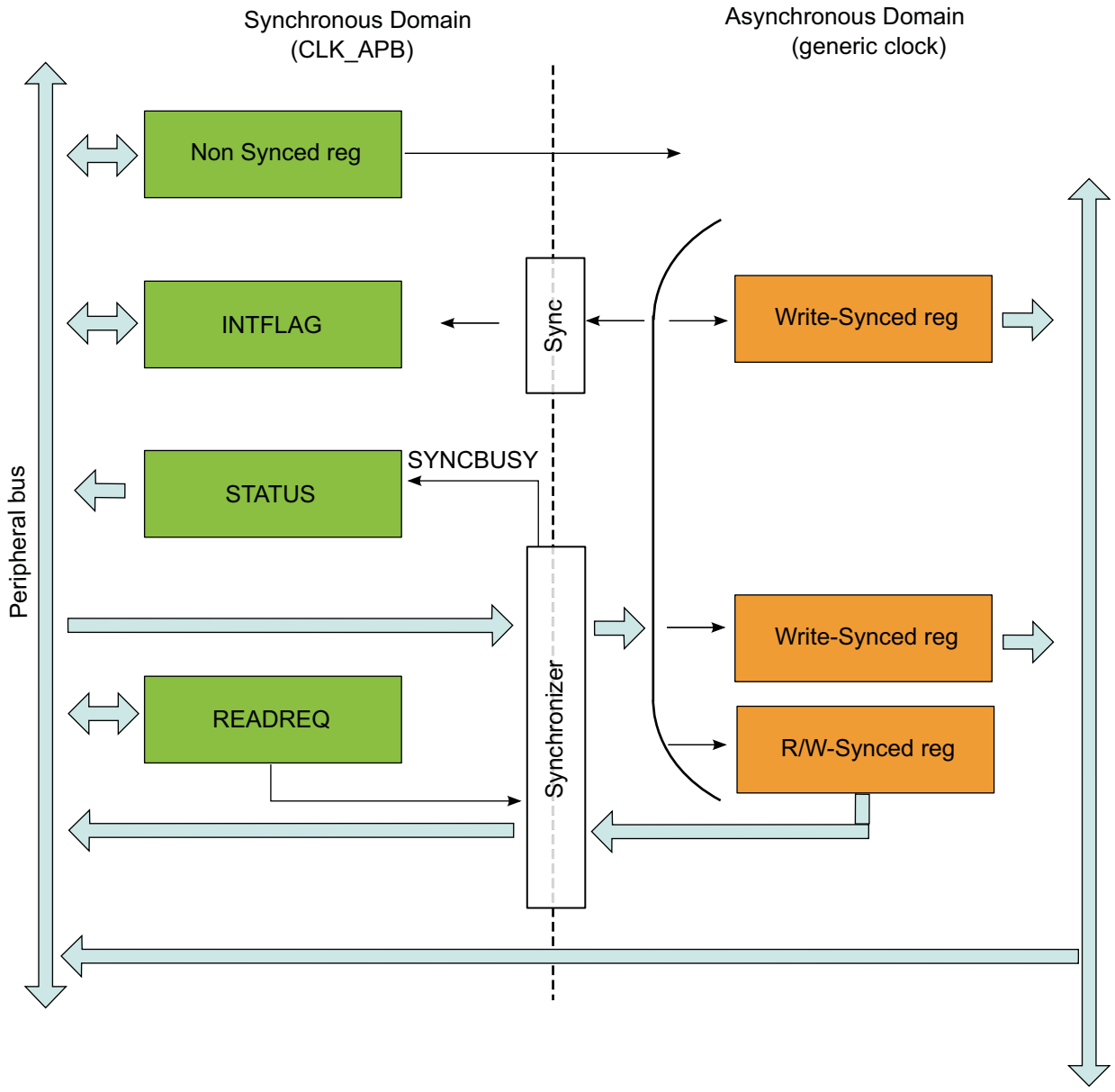
13.3 Register Synchronization

13.3.1 Overview

All peripherals are composed of one digital bus interface, which is connected to the APB or AHB bus and clocked using a corresponding synchronous clock, and one core clock, which is clocked using a generic clock. Access between these clock domains must be synchronized. As this mechanism is implemented in hardware the synchronization process takes place even if the different clocks domains are clocked from the same source and on the same frequency. All registers in the bus interface are accessible without synchronization. All core registers in the generic clock domain must be synchronized when written. Some core registers must be synchronized when read. Registers that need synchronization has this denoted in each individual register description. Two properties are used: write-synchronization and read-synchronization.

A common synchronizer is used for all registers in one peripheral, as shown in Figure 13-3. Therefore, only one register per peripheral can be synchronized at a time.

Figure 13-3. Synchronization



13.3.2 Write-Synchronization

The write-synchronization is triggered by a write to any generic clock core register. The Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set when the write-synchronization starts and cleared when the write-synchronization is complete. Refer to “Synchronization Delay” on page 76 for details on the synchronization delay.

When the write-synchronization is ongoing (STATUS.SYNCBUSY is one), any of the following actions will cause the peripheral bus to stall until the synchronization is complete:

- Writing a generic clock core register
- Reading a read-synchronized core register
- Reading the register that is being written (and thus triggered the synchronization)

Core registers without read-synchronization will remain static once they have been written and synchronized, and can be read while the synchronization is ongoing without causing the peripheral bus to stall. APB registers can also be read while the synchronization is ongoing without causing the peripheral bus to stall.

13.3.3 Read-Synchronization

Reading a read-synchronized core register will cause the peripheral bus to stall immediately until the read-synchronization is complete. STATUS.SYNCBUSY will not be set. Refer to “Synchronization Delay” on page 76 for details on the synchronization delay. Note that reading a read-synchronized core register while STATUS.SYNCBUSY is one will cause the peripheral bus to stall twice; first because of the ongoing synchronization, and then again because reading a read-synchronized core register will cause the peripheral bus to stall immediately.

13.3.4 Completion of synchronization

The user can either poll STATUS.SYNCBUSY or use the Synchronisation Ready interrupt (if available) to check when the synchronization is complete. It is also possible to perform the next read/write operation and wait, as this next operation will be started once the previous write/read operation is synchronized and/or complete.

13.3.5 Read Request

The read request functionality is only available to peripherals that have the Read Request register (READREQ) implemented. Refer to the register description of individual peripheral chapters for details.

To avoid forcing the peripheral bus to stall when reading read-synchronized core registers, the read request mechanism can be used.

13.3.5.1 Basic Read Request

Writing a one to the Read Request bit in the Read Request register (READREQ.RREQ) will request read-synchronization of the register specified in the Address bits in READREQ (READREQ.ADDR) and set STATUS.SYNCBUSY. When read-synchronization is complete, STATUS.SYNCBUSY is cleared. The read-synchronized value is then available for reading without delay until READREQ.RREQ is written to one again.

The address to use is the offset to the peripheral's base address of the register that should be synchronized.

13.3.5.2 Continuous Read Request

Writing a one to the Read Continuously bit in READREQ (READREQ.RCONT) will force continuous read-synchronization of the register specified in READREQ.ADDR. The latest value is always available for reading without stalling the bus, as the synchronization mechanism is continuously synchronizing the given value.

SYNCBUSY is set for the first synchronization, but not for the subsequent synchronizations. If another synchronization is attempted, i.e. by executing a write-operation of a write-synchronized register, the read request will be stopped, and will have to be manually restarted.

Note that continuous read-synchronization is paused in sleep modes where the generic clock is not running. This means that a new read request is required if the value is needed immediately after exiting sleep.

13.3.6 Enable Write-Synchronization

Writing to the Enable bit in the Control register (CTRL.ENABLE) will also trigger write-synchronization and set STATUS.SYNCBUSY. CTRL.ENABLE will read its new value immediately after being written. The Synchronisation Ready interrupt (if available) cannot be used for Enable write-synchronization.

When the enable write-synchronization is ongoing (STATUS.SYNCBUSY is one), attempt to do any of the following will cause the peripheral bus to stall until the enable synchronization is complete:

- Writing a core register
- Writing an APB register
- Reading a read-synchronized core register

APB registers can be read while the enable write-synchronization is ongoing without causing the peripheral bus to stall.

13.3.7 Software Reset Write-Synchronization

Writing a one to the Software Reset bit in CTRL (CTRL.SWRST) will also trigger write-synchronization and set STATUS.SYNCBUSY. When writing a one to the CTRL.SWRST bit it will immediately read as one. CTRL.SWRST and

STATUS.SYNCBUSY will be cleared by hardware when the peripheral has been reset. Writing a zero to the CTRL.SWRST bit has no effect. The Synchronisation Ready interrupt (if available) cannot be used for Software Reset write-synchronization.

When the software reset is in progress (STATUS.SYNCBUSY and CTRL.SWRST are one), attempt to do any of the following will cause the peripheral bus to stall until the Software Reset synchronization and the reset is complete:

- Writing a core register
- Writing an APB register
- Reading a read-synchronized register

APB registers can be read while the software reset is being write-synchronized without causing the peripheral bus to stall.

13.3.8 Synchronization Delay

The synchronization will delay the write or read access duration by a delay D, given by the equation:

$$5 \cdot P_{GCLK} + 2 \cdot P_{APB} < D < 6 \cdot P_{GCLK} + 3 \cdot P_{APB}$$

Where P_{GCLK} is the period of the generic clock and P_{APB} is the period of the peripheral bus clock. A normal peripheral bus register access duration is $2 \cdot P_{APB}$.

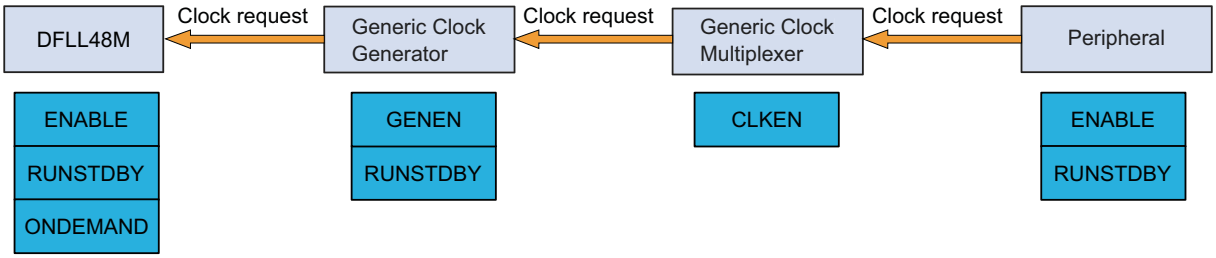
13.4 Enabling a Peripheral

To enable a peripheral clocked by a generic clock, the following parts of the system needs to be configured:

- A running clock source.
- A clock from the Generic Clock Generator must be configured to use one of the running clock sources, and the generator must be enabled.
- The generic clock, through the Generic Clock Multiplexer, that connects to the peripheral needs to be configured with a running clock from the Generic Clock Generator, and the generic clock must be enabled.
- The user interface of the peripheral needs to be unmasked in the PM. If this is not done the peripheral registers will read as all 0's and any writes to the peripheral will be discarded.

13.5 On-demand, Clock Requests

Figure 13-4. Clock request routing



All the clock sources in the system can be run in an on-demand mode, where the clock source is in a stopped state when no peripherals are requesting the clock source. Clock requests propagate from the peripheral, via the GCLK, to the clock source. If one or more peripheral is using a clock source, the clock source will be started/kept running. As soon as the clock source is no longer needed and no peripheral have an active request the clock source will be stopped until requested again. For the clock request to reach the clock source, the peripheral, the generic clock and the clock from the Generic Clock Generator in-between must be enabled. The time taken from a clock request being asserted to the clock source being ready is dependent on the clock source startup time, clock source frequency as well as the divider used in the Generic Clock Generator. The total startup time from a clock request to the clock is available for the peripheral is:

Delay_start_max = Clock source startup time + 2 * clock source periods + 2 * divided clock source periods

Delay_start_min = Clock source startup time + 1 * clock source period + 1 * divided clock source period

The delay for shutting down the clock source when there is no longer an active request is:

Delay_stop_min = 1 * divided clock source period + 1 * clock source period

Delay_stop_max = 2 * divided clock source periods + 2 * clock source periods

The On-Demand principle can be disabled individually for each clock source by clearing the ONDEMAND bit located in each clock source controller. The clock is always running whatever is the clock request. This has the effect to remove the clock source startup time at the cost of the power consumption.

In standby mode, the clock request mechanism is still working if the modules are configured to run in standby mode (RUNSTDBY bit).

13.6 Power Consumption vs Speed

Due to the nature of the asynchronous clocking of the peripherals there are some considerations that needs to be taken if either targeting a low-power or a fast-acting system. If clocking a peripheral with a very low clock, the active power consumption of the peripheral will be lower. At the same time the synchronization to the synchronous (CPU) clock domain is dependent on the peripheral clock speed, and will be longer with a slower peripheral clock; giving lower response time and more time waiting for the synchronization to complete.

13.7 Clocks after Reset

On any reset the synchronous clocks start to their initial state:

- OSC8M is enabled and divided by 8
- GCLKMAIN uses OSC8M as source
- CPU and BUS clocks are undivided

On a power reset the GCLK starts to their initial state:

- All generic clock generators disabled except:
  - the generator 0 (GCLKMAIN) using OSC8M as source, with no division
  - the generator 2 using OSCULP32K as source, with no division
- All generic clocks disabled except:
  - the WDT generic clock using the generator 2 as source

On a user reset the GCLK starts to their initial state, except for:

- generic clocks that are write-locked (WRTLOCK is written to one prior to reset or the WDT generic clock if the WDT Always-On at power on bit set in the NVM User Row)
- The generic clock dedicated to the RTC if the RTC generic clock is enabled

On any reset the clock sources are reset to their initial state except the 32KHz clock sources which are reset only by a power reset.

14. GCLK – Generic Clock Controller

14.1 Overview

Several peripherals may require specific clock frequencies to operate correctly. The GCLK provides a number of generic clock generators that can provide a wide range of clock frequencies. The generic clock generators can be set to use different external and internal clock sources. The clock in each generic clock generator can be divided down. The outputs from the generic clock generators are used as clock sources for the generic clock multiplexers, which select one of the sources to generate a generic clock (GCLK_PERIPH), as shown in Figure 14-2. The number of generic clocks, m, depends on how many peripherals the device has.

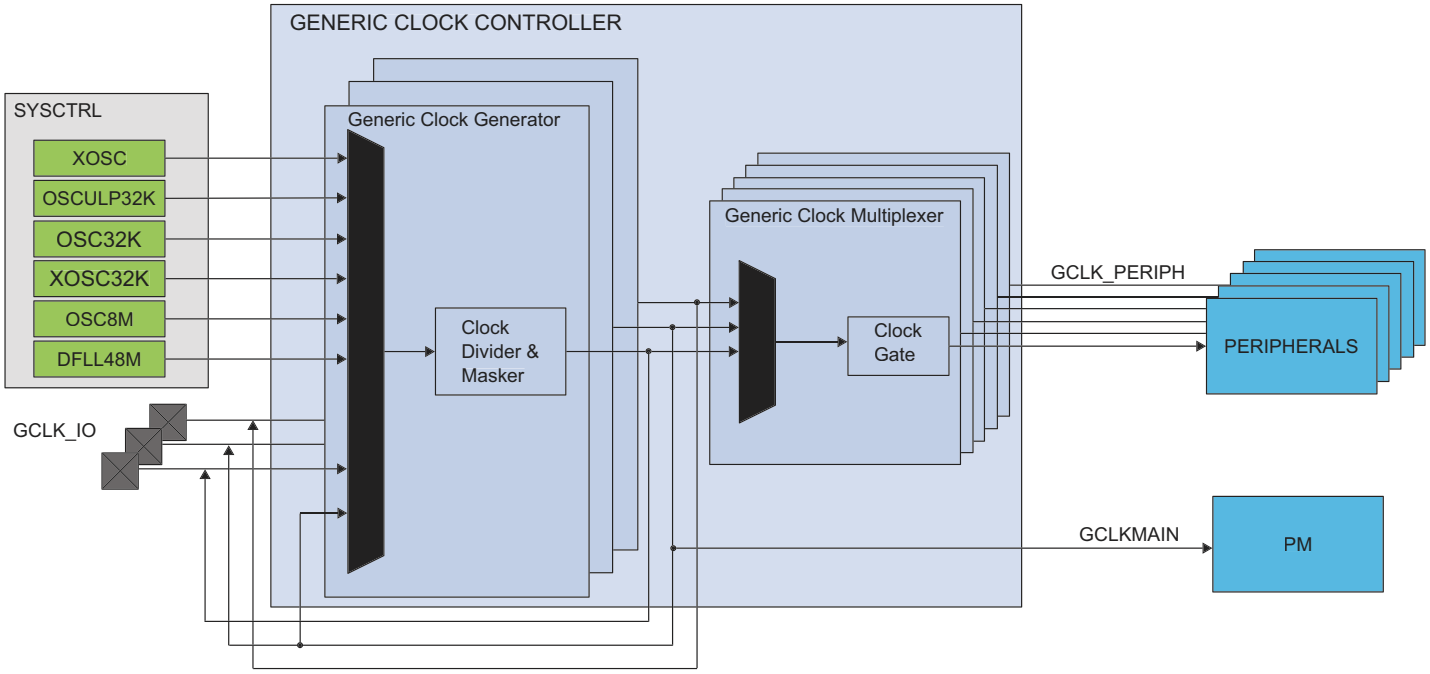
14.2 Features

- Provides a user-defined number (max 64) of generic clocks
- Wide frequency range

14.3 Block Diagram

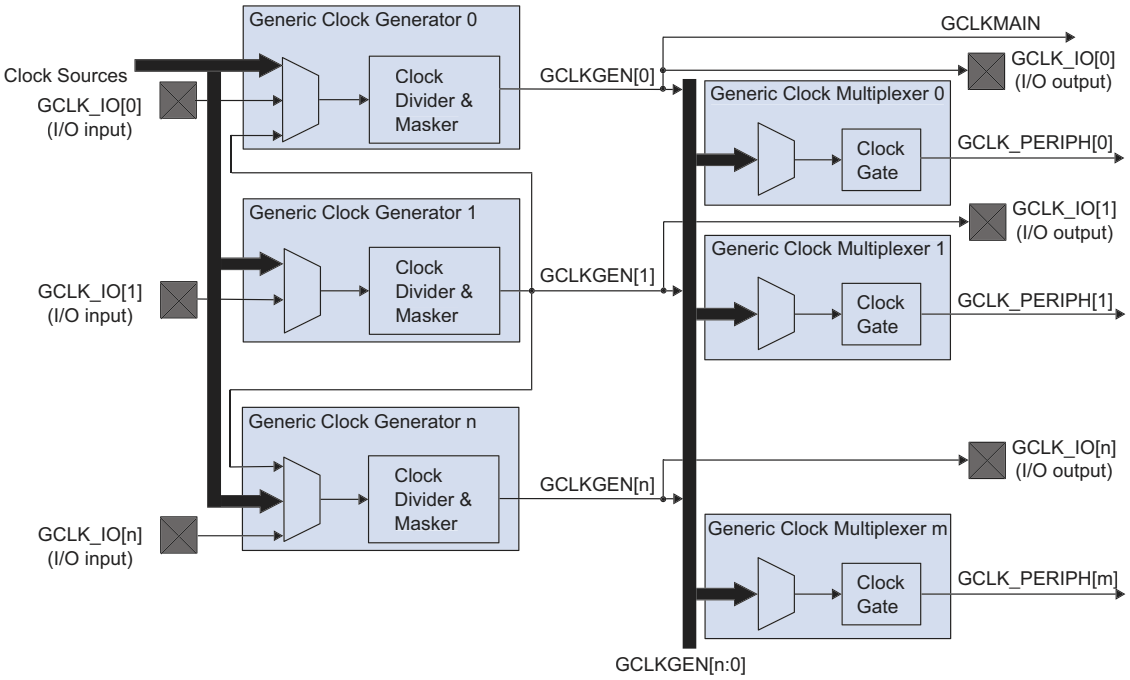
The GCLK can be seen in the clocking diagram, which is shown in Figure 14-1 .

Figure 14-1. Device Clocking Diagram



The GCLK block diagram is shown in Figure 14-2.

Figure 14-2. Generic Clock Controller Block Diagram



14.4 Signal Description

Signal Name	Type	Description
GCLK_IO[n..0]	Digital I/O	Source clock when input Generic clock when output

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

14.5 Product Dependencies

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

14.5.1 I/O Lines

Using the GCLK's I/O lines requires the I/O pins to be configured. Refer to "PORT" on page 276 for details.

14.5.2 Power Management

The GCLK can operate in all sleep modes, if required. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

14.5.3 Clocks

The GCLK bus clock (CLK_GCLK_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_GCLK_APB can be found in the Peripheral Clock Masking section in "PM – Power Manager" on page 100.

14.5.4 Interrupts

Not applicable.

14.5.5 Events

Not applicable.

14.5.6 Debug Operation

Not applicable.

14.5.7 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC).

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode or the CPU reset is extended, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

14.5.8 Analog Connections

Not applicable.

14.6 Functional Description

14.6.1 Principle of Operation

The GCLK module is comprised of eight generic clock generators sourcing m generic clock multiplexers.

A clock source selected as input to one of the generic clock generators can be used directly, or it can be prescaled in the generic clock generator before the generator output is used as input to one or more of the generic clock multiplexers. A generic clock multiplexer provides a generic clock to the peripherals (GCLK_PERIPHERAL). A generic clock can act as the clock to one or several of the peripherals.

14.6.2 Basic Operation

14.6.2.1 Initialization

Before a generic clock is enabled, the clock source of its generic clock generator should be enabled. The generic clock must be configured as outlined by the following steps:

1. The generic clock generator division factor must be set by performing a single 32-bit write to the Generic Clock Generator Division register (GENDIV):
   - The generic clock generator that will be selected as the source of the generic clock must be written to the ID bit group (GENDIV.ID)
   - The division factor must be written to the DIV bit group (GENDIV.DIV)
2. The generic clock generator must be enabled by performing a single 32-bit write to the Generic Clock Generator Control register (GENCTRL):
   - The generic clock generator that will be selected as the source of the generic clock must be written to the ID bit group (GENCTRL.ID)
   - The generic clock generator must be enabled by writing a one to the GENEN bit (GENCTRL.GENEN)
3. The generic clock must be configured by performing a single 16-bit write to the Generic Clock Control register (CLKCTRL):
   - The generic clock that will be configured must be written to the ID bit group (CLKCTRL.ID)
   - The generic clock generator used as the source of the the generic clock must be written to the GEN bit group (CLKCTRL.GEN).

14.6.2.2 Enabling, Disabling and Resetting

The GCLK module has no enable/disable bit to enable or disable the whole module.

The GCLK is reset by writing a one to the Software Reset bit in the Control register (CTRL.SWRST). All registers in the GCLK will be reset to their initial state except for generic clocks and associated generators that have their Write Lock bit written to one. Refer to “Configuration Lock” on page 83 for details.

14.6.2.3 Generic Clock Generator

Each generic clock generator (GCLKGEN) can be set to run from one of eight different clock sources except GCLKGEN[1] which can be set to run from one of seven sources. GCLKGEN[1] can act as source to the other generic clock generators but can not act as source to itself.

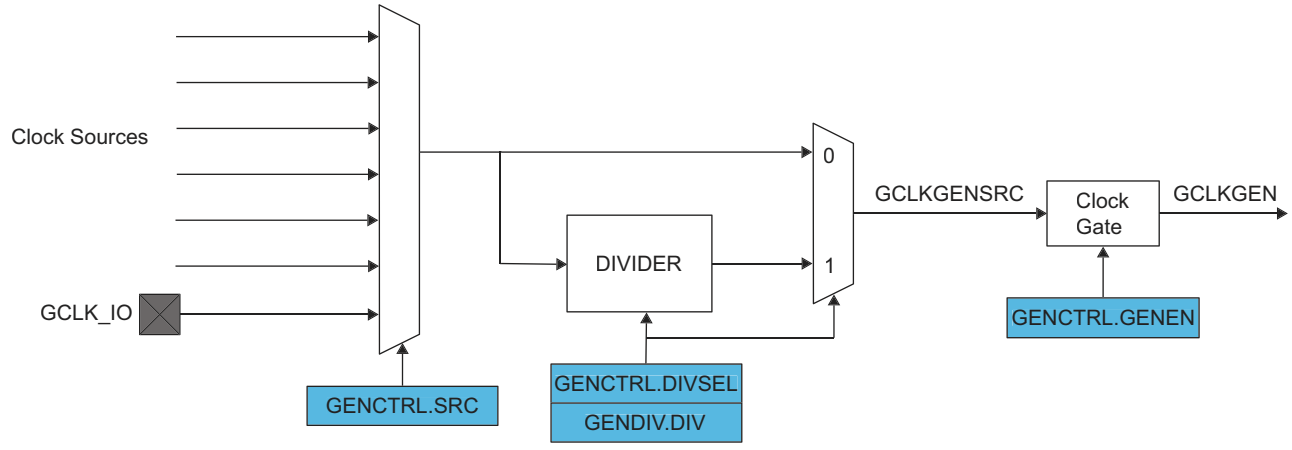
Each generic clock generator GCLKGEN[x] can be connected to one specific GCLK_IO[x] pin. The GCLK_IO[x] can be set to act as source to GCLKGEN[x] or GCLK_IO[x] can be set up to output the clock generated by GCLKGEN[x].

The selected source (GCLKGENSRC see Figure 14-3) can optionally be divided. Each generic clock generator can be independently enabled and disabled.

Each GCLKGEN clock can then be used as a clock source for the generic clock multiplexers. Each generic clock is allocated to one or several peripherals.

GCLKGEN[0], is used as GCLKMAIN for the synchronous clock controller inside the Power Manager. Refer to “PM – Power Manager” on page 100 for details on the synchronous clock generation.

Figure 14-3. Generic Clock Generator



14.6.2.4 Enabling a Generic Clock Generator

A generic clock generator is enabled by writing a one to the Generic Clock Generator Enable bit in the Generic Clock Generator Control register (GENCTRL.GENEN).

14.6.2.5 Disabling a Generic Clock Generator

A generic clock generator is disabled by writing a zero to GENCTRL.GENEN. When GENCTRL.GENEN is read as zero, the GCLKGEN clock is disabled and clock gated.

14.6.2.6 Selecting a Clock Source for the Generic Clock Generator

Each generic clock generator can individually select a clock source by writing to the Source Select bit group in GENCTRL (GENCTRL.SRC). Changing from one clock source, A, to another clock source, B, can be done on the fly. If clock source B is not ready, the generic clock generator will continue running with clock source A. As soon as clock source B is ready, however, the generic clock generator will switch to it. During the switching, the generic clock generator holds clock requests to clock sources A and B and then releases the clock source A request when the switch is done.

The available clock sources are device dependent (usually the oscillators, RC oscillators, PLL and DFLL clocks). GCLKGEN[1] can be used as a common source for all the generic clock generators except generic clock generator 1.

14.6.2.7 Changing Clock Frequency

The selected generic clock generator source, GENCLKSRC can optionally be divided by writing a division factor

in the Division Factor bit group in the Generic Clock Generator Division register (GENDIV.DIV). Depending on the value of the Divide Selection bit in GENCTRL (GENCTRL.DIVSEL), it can be interpreted in two ways by the integer divider, as shown in Table 14-1.

Note that the number of DIV bits for each generic clock generator is device dependent. Refer to Table 14-10 for details.

Table 14-1. Division Factor

GENCTRL.DIVSEL	Division Factor
0	GENDIV.DIV
1	2^(GENDIV.DIV+1)

If GENCTRL.DIVSEL is zero and GENDIV.DIV is zero or one, the output clock will be undivided.

14.6.2.8 Duty Cycle

When dividing a clock with an odd division factor, the duty-cycle will not be 50/50. Writing a one to the Improve Duty Cycle bit in GENCTRL (GENCTRL.IDC) will result in a 50/50 duty cycle.

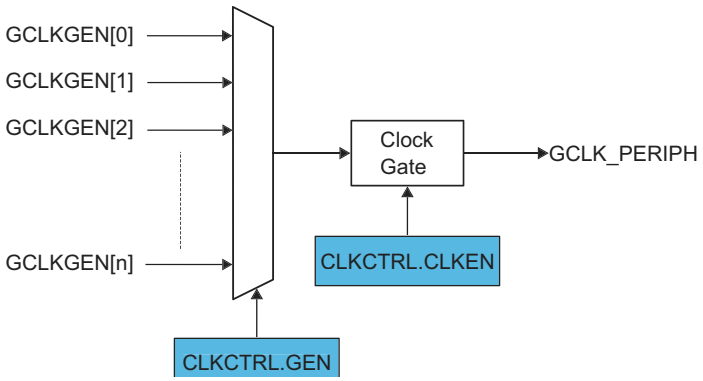
14.6.2.9 External Clock

Each generic clock generator output clock (GCLKGEN) can be output. If the Output Enable bit in GENCTRL (GENCTRL.OE) is one and the generic clock generator is enabled (GENCTRL.GENEN is one), the generic clock generator requests its clock source and the GCLKGEN clock is output to a GCLK_IO pin. If GENCTRL.OE is zero, GCLK_IO is set according to the Output Off Value bit. If the Output Off Value bit in GENCTRL (GENCTRL.OOV) is zero, the output clock will be low when turned off. If GENCTRL.OOV is one, the output clock will be high when turned off.

In standby mode, if the clock is output (GENCTRL.OE is one), the clock on the GCLK_IO pin is frozen to the OOV value if the Run In Standby bit in GENCTRL (GENCTRL.RUNSTDBY) is zero. If GENCTRL.RUNSTDBY is one, the GCLKGEN clock is kept running and output to GCLK_IO.

14.6.3 Generic Clock

Figure 14-4. Generic Clock



14.6.3.1 Enabling a Generic Clock

Before a generic clock is enabled, one of the generic clock generators must be selected as the source for the generic clock by writing to CLKCTRL.GEN. The clock source selection is individually set for each generic clock.

When a generic clock generator has been selected, the generic clock is enabled by writing a one to the Clock Enable bit in CLKCTRL (CLKCTRL.CLKEN). The CLKCTRL.CLKEN bit must be synchronized to the generic clock domain. CLKCTRL.CLKEN will continue to read as its previous state until the synchronization is complete.

14.6.3.2 Disabling a Generic Clock

A generic clock is disabled by writing a zero to CLKCTRL.CLKEN. The CLKCTRL.CLKEN bit must be synchronized to the generic clock domain and the clock actually switched off. CLKCTRL.CLKEN will continue to read as its previous state until the synchronization is complete. When the generic clock is disabled, the generic clock is clock gated.

14.6.3.3 Selecting a Clock Source for the Generic Clock

When changing a generic clock source by writing to CLKCTRL.GEN, the generic clock must be disabled before being re-enabled with the new clock source setting. This prevents glitches during the transition:

a. Write a zero to CLKCTRL.CLKEN
b. Wait until CLKCTRL.CLKEN reads as zero
c. Change the source of the generic clock by writing CLKCTRL.GEN
d. Re-enable the generic clock by writing a one to CLKCTRL.CLKEN

14.6.3.4 Configuration Lock

The generic clock configuration is locked for further write accesses by writing the Write Lock bit (WRTLOCK) in the CLKCTRL register. All writes to the CLKCTRL register will be ignored. It can only be unlocked by a power reset.

The generic clock generator sources of a “locked” generic clock are also locked. The corresponding GENCTRL and GENDIV are locked, and can be unlocked only by a power reset.

There is one exception concerning the GCLKGEN[0]. As it is used as GCLKMAIN, it can not be locked. It is reset by any reset to startup with a known configuration.

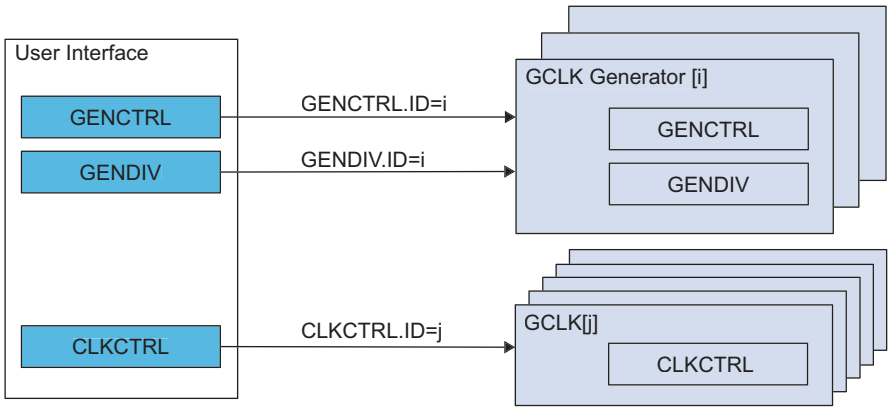
The SWRST can not unlock the registers.

14.6.4 Additional Features

14.6.4.1 Indirect Access

The Generic Clock Generator Control and Division registers (GENCTRL and GENDIV) and the Generic Clock Control register (CLKCTRL) are indirectly addressed as shown in Figure 14-5.

Figure 14-5. GCLK Indirect Access



Writing these registers is done by setting the corresponding ID bit group.

To read a register, the user must write (byte access) the ID of the channel, i, in the corresponding register. The value of the register for the corresponding ID is available in the user interface by a read access.

For example, the sequence to read the GENCTRL register of generic clock generator i is:

a. Do an 8-bit write of the i value to GENCTRL.ID
b. Read GENCTRL

14.6.4.2 Generic Clock Enable after Reset

The Generic Clock Controller must be able to provide a generic clock to some specific peripherals after a reset. That means that the configuration of the generic clock generators and generic clocks after reset is device-dependent.

Refer to Table 14-8 and Table 14-9 for details on GENCTRL reset.

Refer to Table 14-12 and Table 14-13 for details on GENDIV reset.

Refer to Table 14-4 and Table 14-5 for details on CLKCTRL reset.

14.6.5 Sleep Mode Operation

14.6.5.1 SleepWalking

The GCLK module supports the SleepWalking feature. During a sleep mode where the generic clocks are stopped, a peripheral that needs its generic clock to execute a process must request it from the Generic Clock Controller.

The Generic Clock Controller will receive this request and then determine which generic clock generator is involved and which clock source needs to be awakened. It then wakes up the clock source, enables the generic clock generator and generic clock stages successively and delivers the generic clock to the peripheral.

14.6.5.2 Run in Standby Mode

In standby mode, the GCLK can continuously output the generic clock generator output to GCLK_IO. Refer to “External Clock” on page 82 for details.

14.6.6 Synchronization

Due to the asynchronicity between CLK_GCLK_APB and GCLKGENSRC some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following registers need synchronization when written:

- Generic Clock Generator Control register (GENCTRL)
- Generic Clock Generator Division register (GENDIV)
- Control register (CTRL)

Write-synchronization is denoted by the Write-Synchronization property in the register description.

Refer to the Synchronization chapter for further details.

14.7 Register Summary

Offset	Name	Bit Pos.								
0x0	CTRL	7:0								SWRST
0x1	STATUS	7:0	SYNCBUSY							
0x2	CLKCTRL	7:0			ID[5:0]					
0x3		15:8	WRTLOCK	CLKEN			GEN[3:0]			
0x4	GENCTRL	7:0					ID[3:0]			
0x5		15:8				SRC[4:0]				
0x6		23:16			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
0x7		31:24								
0x8	GENDIV	7:0					ID[3:0]			
0x9		15:8	DIV[7:0]							
0xA		23:16	DIV[15:8]							
0xB		31:24								

14.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-protected property in each individual register description. Refer to “Register Access Protection” on page 80 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 84 for details.

14.8.1 Control

Name: CTRL

Offset: 0x0

Reset: 0x00

Property: Write-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
								SWRST
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: There is a reset operation ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the GCLK to their initial state after a power reset, except for generic clocks and associated generators that have their WRTLOCK bit in CLKCTRL read as one.
  Refer to Table 14-8 for details on GENCTRL reset.
  Refer to Table 14-12 for details on GENDIV reset.
  Refer to Table 14-4 for details on CLKCTRL reset.
  Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

14.8.2 Status

Name: STATUS

Offset: 0x1

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy Status**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

14.8.3 Generic Clock Control

This register allows the user to configure one of the generic clocks, as specified in the CLKCTRL.ID bit group. To write to the CLKCTRL register, do a 16-bit write with all configurations and the ID.

To read the CLKCTRL register, first do an 8-bit write to the CLKCTRL.ID bit group with the ID of the generic clock whose configuration is to be read, and then read the CLKCTRL register.

Name: CLKCTRL

Offset: 0x2

Reset: 0x0000

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
	WRTLOCK	CLKEN			GEN[3:0]			
Access	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			ID[5:0]					
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – WRTLOCK: Write Lock**
  When this bit is written, it will lock from further writes the generic clock pointed to by CLKCTRL.ID, the generic clock generator pointed to in CLKCTRL.GEN and the division factor used in the generic clock generator. It can only be unlocked by a power reset.
  One exception to this is generic clock generator 0, which cannot be locked.
  0: The generic clock and the associated generic clock generator and division factor are not locked.
  1: The generic clock and the associated generic clock generator and division factor are locked.

- **Bit 14 – CLKEN: Clock Enable**
  This bit is used to enable and disable a generic clock.
  0: The generic clock is disabled.
  1: The generic clock is enabled.

- **Bits 13:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 11:8 – GEN[3:0]: Generic Clock Generator**
  These bits select the generic clock generator to be used as the source of a generic clock. The value of the GEN bit group versus generic clock generator is shown in Table 14-2.

Table 14-2. Generic Clock Generator

Value	Description
0x0	Generic clock generator 0
0x1	Generic clock generator 1
0x2	Generic clock generator 2
0x3	Generic clock generator 3
0x4	Generic clock generator 4
0x5	Generic clock generator 5
0x6	Generic clock generator 6
0x7	Generic clock generator 7
0x8-0xF	Reserved

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:0 – ID[5:0]: Generic Clock Selection ID**

These bits select the generic clock that will be configured. The value of the ID bit group versus module instance is shown in Table 14-3.

Table 14-3. Generic Clock Selection ID

Value	Description
0x00	DFLL48M Reference
0x01	WDT
0x02	RTC
0x03	EIC
0x04	EVSYS_CHANNEL_0
0x05	EVSYS_CHANNEL_1
0x06	EVSYS_CHANNEL_2
0x07	EVSYS_CHANNEL_3
0x08	EVSYS_CHANNEL_4
0x09	EVSYS_CHANNEL_5
0x0A	EVSYS_CHANNEL_6
0x0B	EVSYS_CHANNEL_7
0x0C	SERCOMx_SLOW
0x0D	SERCOM0_CORE
0x0E	SERCOM1_CORE

Table 14-3. Generic Clock Selection ID (Continued)

0x0F	SERCOM2_CORE
0x10	SERCOM3_CORE
0x11	SERCOM4_CORE
0x12	SERCOM5_CORE
0x13	TC0,TC1
0x14	TC2,TC3
0x15	TC4,TC5
0x16	TC6,TC7
0x17	ADC
0x18	AC_DIG
0x19	AC_ANA
0x1A	DAC
0x1B	PTC
0x1C-0x3F	Reserved

A power reset will reset the CLKCTRL register for all IDs, including the RTC. If the WRTLOCK bit of the corresponding ID is zero and the ID is not the RTC, a user reset will reset the CLKCTRL register for this ID.

After a power reset, the reset value of the CLKCTRL register versus module instance is as shown in Table 14-4.

Table 14-4. CLKCTRL Reset Value after a Power Reset

Module Instance	Reset Value after a Power Reset		
	CLKCTRL.GEN	CLKCTRL.CLKEN	CLKCTRL.WRTLOCK
RTC	0x00	0x00	0x00
WDT	0x02	0x01 if WDT Enable bit in NVM User Row written to one 0x00 if WDT Enable bit in NVM User Row written to zero	0x01 if WDT Always-On bit in NVM User Row written to one 0x00 if WDT Always-On bit in NVM User Row written to zero
Others	0x00	0x00	0x00

After a user reset, the reset value of the CLKCTRL register versus module instance is as shown in Table 14-5.

Table 14-5. CLKCTRL Reset Value after a User Reset

Module Instance	Reset Value after a User Reset		
	CLKCTRL.GEN	CLKCTRL.CLKEN	CLKCTRL.WRTLOCK
RTC	0x00 if WRTLOCK=0 and CLKEN=0 No change if WRTLOCK=1 or CLKEN=1	0x00 if WRTLOCK=0 and CLKEN=0 No change if WRTLOCK=1 or CLKEN=1	No change
WDT	0x02 if WRTLOCK=0 No change if WRTLOCK=1	If WRTLOCK=0 0x01 if WDT Enable bit in NVM User Row written to one 0x00 if WDT Enable bit in NVM User Row written to zero If WRTLOCK=1 no change	No change
Others	0x00 if WRTLOCK=0 No change if WRTLOCK=1	0x00 if WRTLOCK=0 No change if WRTLOCK=1	No change

14.8.4 Generic Clock Generator Control

This register allows the user to configure one of the generic clock generators, as specified in the GENCTRL.ID bit group. To write to the GENCTRL register, do a 32-bit write with all configurations and the ID.

To read the GENCTRL register, first do an 8-bit write to the GENCTRL.ID bit group with the ID of the generic clock generator twhose configuration is to be read, and then read the GENCTRL register.

Name: GENCTRL

Offset: 0x4

Reset: 0x00010600

Property: Write-protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				SRC[4:0]				
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					ID[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 21 – RUNSTDBY: Run in Standby**
  This bit is used to keep the generic clock generator running when it is configured to be output to its dedicated GCLK_IO pin. If GENCTRL.OE is zero, this bit has no effect and the generic clock generator will only be running if a peripheral requires the clock.

  0: The generic clock generator is stopped in standby and the GCLK_IO pin state (one or zero) will be dependent on the setting in GENCTRL.OOV.

  1: The generic clock generator is kept running and output to its dedicated GCLK_IO pin during standby mode.

- **Bit 20 – DIVSEL: Divide Selection**
  This bit is used to decide how the clock source used by the generic clock generator will be divided. If the clock source should not be divided, the DIVSEL bit must be zero and the GENDIV.DIV value for the corresponding generic clock generator must be zero or one.
  0: The generic clock generator equals the clock source divided by GENDIV.DIV.
  1: The generic clock generator equals the clock source divided by 2^(GENDIV.DIV+1).

- **Bit 19 – OE: Output Enable**
  This bit is used to enable output of the generated clock to GCLK_IO when GCLK_IO is not selected as a source in the GENCLK.SRC bit group.
  0: The generic clock generator is not output.
  1: The generic clock generator is output to the corresponding GCLK_IO, unless the corresponding GCLK_IO is selected as a source in the GENCLK.SRC bit group.

- **Bit 18 – OOV: Output Off Value**
  This bit is used to control the value of GCLK_IO when GCLK_IO is not selected as a source in the GENCLK.SRC bit group.
  0: The GCLK_IO will be zero when the generic clock generator is turned off or when the OE bit is zero.
  1: The GCLK_IO will be one when the generic clock generator is turned off or when the OE bit is zero.

- **Bit 17 – IDC: Improve Duty Cycle**
  This bit is used to improve the duty cycle of the generic clock generator when odd division factors are used.
  0: The generic clock generator duty cycle is not 50/50 for odd division factors.
  1: The generic clock generator duty cycle is 50/50.

- **Bit 16 – GENEN: Generic Clock Generator Enable**
  This bit is used to enable and disable the generic clock generator.
  0: The generic clock generator is disabled.
  1: The generic clock generator is enabled.

- **Bits 15:13 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 12:8 – SRC[4:0]: Source Select**
  These bits define the clock source to be used as the source for the generic clock generator, as shown in Table 14-6.

Table 14-6. Source Select

Value	Name	Description
0x00	XOSC	XOSC oscillator output
0x01	GCLKIN	Generator input pad
0x02	GCLKGEN1	Generic clock generator 1 output
0x03	OSCULP32K	OSCULP32K oscillator output
0x04	OSC32K	OSC32K oscillator output
0x05	XOSC32K	XOSC32K oscillator output
0x06	OSC8M	OSC8M oscillator output
0x07	DFLL48M	DFLL48M output
0x08-0x1F	Reserved	Reserved for future use

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 3:0 – ID[3:0]: Generic Clock Generator Selection**
  These bits select the generic clock generator that will be configured or read. The value of the ID bit group versus which generic clock generator is configured is shown in Table 14-7.

Table 14-7. Generic Clock Generator Selection

Value	Description
0x0	Generic clock generator 0
0x1	Generic clock generator 1
0x2	Generic clock generator 2
0x3	Generic clock generator 3
0x4	Generic clock generator 4
0x5	Generic clock generator 5
0x6	Generic clock generator 6
0x7	Generic clock generator 7
0x8-0xF	Reserved

A power reset will reset the GENCTRL register for all IDs, including the generic clock generator used by the RTC. If a generic clock generator ID other than generic clock generator 0 is not a source of a “locked” generic clock or a source of the RTC generic clock, a user reset will reset the GENCTRL for this ID.

After a power reset, the reset value of the GENCTRL register is as shown in Table 14-8.

Table 14-8. GENCTRL Reset Value after a Power Reset

GCLK Generator ID	Reset Value after a Power Reset
0x00	0x00010600
0x01	0x00000001
0x02	0x00010302
0x03	0x00000003
0x04	0x00000004
0x05	0x00000005
0x06	0x00000006
0x07	0x00000007

After a user reset, the reset value of the GENCTRL register is as shown in Table 14-9.

.

Table 14-9. GENCTRL Reset Value after a User Reset

GCLK Generator ID	Reset Value after a User Reset
0x00	0x00010600
0x01	0x00000001 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x02	0x00010302 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x03	0x00000003 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x04	0x00000004 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x05	0x00000005 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x06	0x00000006 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x07	0x00000007 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one

14.8.5 Generic Clock Generator Division

This register allows the user to configure one of the generic clock generators, as specified in the GENDIV.ID bit group. To write to the GENDIV register, do a 32-bit write with all configurations and the ID.

To read the GENDIV register, first do an 8-bit write to the GENDIV.ID bit group with the ID of the generic clock generator whose configuration is to be read, and then read the GENDIV register.

Name: GENDIV

Offset: 0x8

Reset: 0x00000000

Property: Write-protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIV[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DIV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					ID[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:24 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 23:8 – DIV[15:0]: Division Factor**
  These bits apply a division on each selected generic clock generator. The number of DIV bits each generator has can be seen in Table 14-10. Writes to bits above the specified number will be ignored.

Table 14-10. Division Factor

Generator	Division Factor Bits
Generic clock generator 0	8 division factor bits - DIV[7:0]
Generic clock generator 1	16 division factor bits - DIV[15:0]
Generic clock generators 2	5 division factor bits - DIV[4:0]
Generic clock generators 3 - 7	8 division factor bits - DIV[7:0]

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 3:0 – ID[3:0]: Generic Clock Generator Selection**
  These bits select the generic clock generator on which the division factor will be applied, as shown in Table 14-11.

Table 14-11. Generic Clock Generator Selection

Value	Description
0x0	Generic clock generator 0
0x1	Generic clock generator 1
0x2	Generic clock generator 2
0x3	Generic clock generator 3
0x4	Generic clock generator 4
0x5	Generic clock generator 5
0x6	Generic clock generator 6
0x7	Generic clock generator 7
0x8-0xF	Reserved

A power reset will reset the GENDIV register for all IDs, including the generic clock generator used by the RTC. If a generic clock generator ID other than generic clock generator 0 is not a source of a “locked” genericclock or a source of the RTC generic clock, a user reset will reset the GENDIV for this ID.

After a power reset, the reset value of the GENDIV register is as shown in Table 14-12.

Table 14-12. GENDIV Reset value after a Power Reset

GCLK Generator ID	Reset Value after a Power Reset
0x00	0x00000000
0x01	0x00000001
0x02	0x00000002
0x03	0x00000003
0x04	0x00000004
0x05	0x00000005
0x06	0x00000006
0x07	0x00000007

After a user reset, the reset value of the GENDIV register is as shown in Table 14-13.

Table 14-13. GENDIV Reset Value after a User Reset

GCLK Generator ID	Reset Value after a User Reset
0x00	0x00000000
0x01	0x00000001 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x02	0x00000002 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x03	0x00000003 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x04	0x00000004 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x05	0x00000005 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x06	0x00000006 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one
0x07	0x00000007 if the generator is not used by the RTC No change if the generator is used by the RTC or used by a GCLK with a WRTLOCK as one

15. PM – Power Manager

15.1 Overview

The Power Manager (PM) controls the reset, clock generation and power save modes of the microcontroller.

Utilizing a main clock chosen from a large number of clock sources from the GCLK, the clock controller provides synchronous system clocks to the CPU and the modules connected to the AHB and the APBx bus. The synchronous system clocks are divided into a number of clock domains; one for the CPU and AHB and one for each APBx. Any synchronous system clock can be changed at run-time during normal operation. The clock domains can run at different speeds, enabling the user to save power by running peripherals at a relatively low clock frequency, while maintaining high CPU performance. In addition, the clock can be masked for individual modules, enabling the user to minimize power consumption. If for some reason the main clock stops oscillating, the clock failure detector allows switching the main clock to the safe OSC8M clock.

Before entering the STANDBY sleep mode the user must make sure that a significant amount of clocks and peripherals are disabled, so that the voltage regulator is not overloaded.

Various sleep modes and clock gating are provided in order to fit power consumption requirements. This enables the microcontroller to stop unused modules to save power. In ACTIVE mode, the CPU is executing application code. When the device enters a sleep mode, program execution is stopped and some modules and clock domains are automatically switched off by the PM according to the sleep mode. The application code decides which sleep mode to enter and when. Interrupts from enabled peripherals and all enabled reset sources can restore the microcontroller from a sleep mode to ACTIVE mode.

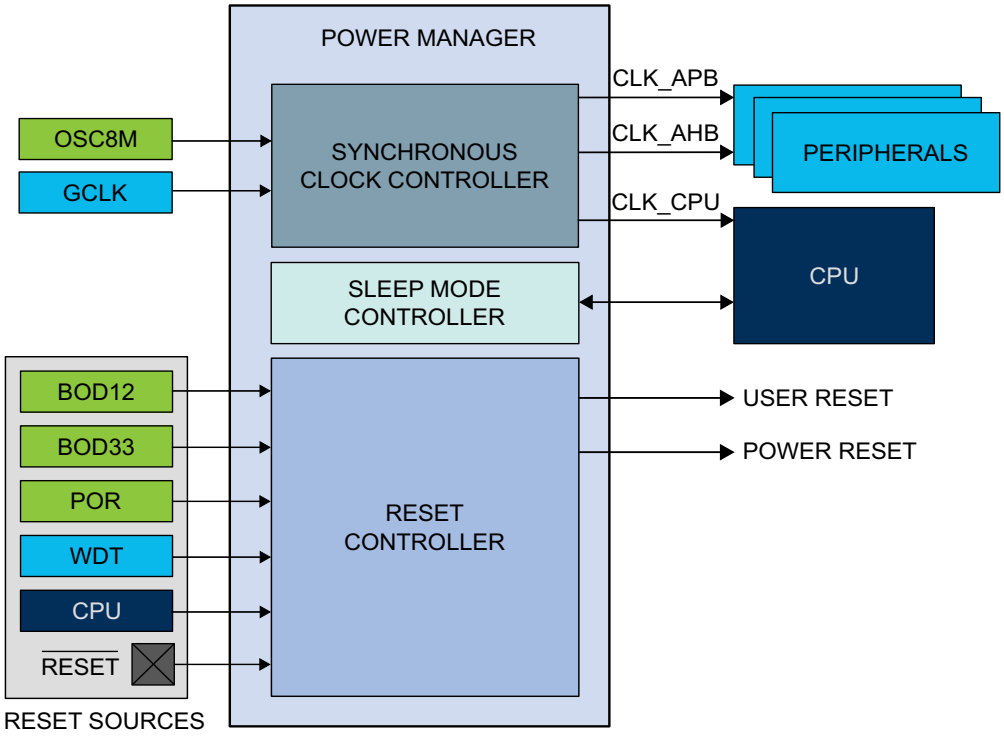
The PM also contains a reset controller, which collects all possible reset sources. It issues a microcontroller reset and sets the device to its initial state, and allows the reset source to be identified by software.

15.2 Features

- Reset control
  - Reset the microcontroller and set it to an initial state according to the reset source
  - Multiple reset sources
    - Power reset sources: POR, BOD33
    - User reset sources: External reset ($\overline{\text{RESET}}$), Watchdog Timer reset, software reset
  - Reset status register for reading the reset source from the application code
- Clock control
  - Generates CPU, AHB and APB system clocks
    - Multiple clock sources and division factor from GCLK
    - Clock prescaler with 1x to 128x division
  - Safe run-time clock switching from GCLK
  - Module-level clock gating through maskable peripheral clocks
  - Clock failure detector
- Power management control
  - Sleep modes: IDLE, STANDBY
  - SleepWalking support on APB and GCLK clocks

15.3 Block Diagram

Figure 15-1. PM Block Diagram



15.4 Signal Description

Signal Name	Type	Description
$\overline{\text{RESET}}$	Digital input	External reset

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

15.5 Product Dependencies

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

15.5.1 I/O Lines

Not applicable.

15.5.2 Power Management

Not applicable.

15.5.3 Clocks

The PM bus clock (CLK_PM_APB) can be enabled and disabled in the power manager, and the default state of CLK_PM_APB can be found in Table 15-1. If this clock is disabled in the Power Manager, it can only be re-enabled by a reset.

A generic clock (GCLK_MAIN) is required to generate the main clock. This clock is configured by default in the Generic Clock Controller, and can be re-configured by the user if needed. Refer to "GCLK – Generic Clock Controller" on page 78 for details.

15.5.3.1 Main Clock

The main clock (CLK_MAIN) is the common source for the synchronous clocks. This is fed into the common 8-bit prescaler that is used to generate synchronous clocks to the CPU, AHB and APBx modules.

15.5.3.2 CPU Clock

The CPU clock (CLK_CPU) is routed to the CPU. Halting the CPU clock inhibits the CPU from executing instructions.

15.5.3.3 AHB Clock

The AHB clock (CLK_AHB) is the root clock source used by peripherals requiring an AHB clock. The AHB clock is always synchronous to the CPU clock and has the same frequency, but may run even when the CPU clock is turned off. A clock gate is inserted from the common AHB clock to any AHB clock of a peripheral.

15.5.3.4 APBx Clocks

The APBx clock (CLK_APBX) is the root clock source used by modules requiring a clock on the APBx bus. The APBx clock is always synchronous to the CPU clock, but can be divided by a prescaler, and will run even when the CPU clock is turned off. A clock gater is inserted from the common APB clock to any APBx clock of a module on APBx bus.

15.5.4 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the PM interrupt requires the Interrupt Controller to be configured first.

15.5.5 Events

Not applicable.

15.5.6 Debug Operation

When the CPU is halted in debug mode, the PM continues normal operation. In sleep mode, the clocks generated from the PM are kept running to allow the debugger accessing any modules. As a consequence, power measurements are not possible in debug mode.

15.5.7 Register Access Protection

All registers with write access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag register (INTFLAG)
- Reset Cause register (RCAUSE)

Write-protection is denoted by the Write-Protection property in the register description.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

15.5.8 Analog Connections

Not applicable.

15.6 Functional Description

15.6.1 Principle of Operation

15.6.1.1 Synchronous Clocks

The GCLK_MAIN clock from GCLK module provides the source for the main clock, which is the common root for the synchronous clocks for the CPU and APBx modules. The main clock is divided by an 8-bit prescaler, and each of the derived clocks can run from any tapping off 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 to respond 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 sleep mode, some clock domains can be turned off.

15.6.1.2 Reset Controller

The Reset Controller collects the various reset sources and generates resets for the device. The device contains a power-on-reset (POR) 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.

15.6.1.3 Sleep Mode Controller

In ACTIVE mode, all clock domains are active, allowing software execution and peripheral operation. The PM Sleep Mode Controller allows the user to choose between different sleep modes depending on application requirements, to save power.

15.6.2 Basic Operation

15.6.2.1 Initialization

After a power-on reset, the PM is enabled and the Reset Cause (RCAUSE) register indicates the POR source. The default clock source of the GCLK_MAIN clock is started and calibrated before the CPU starts running. The GCLK_MAIN clock is selected as the main clock without any division on the prescaler. The device is in the ACTIVE mode.

By default, only the necessary clocks are enabled (see Table 15-1).

15.6.2.2 Enabling, Disabling and Resetting

The PM module is always enabled and can not be reset.

15.6.2.3 Selecting the Main Clock Source

Refer to “GCLK – Generic Clock Controller” on page 78 for details on how to configure the main clock source.

15.6.2.4 Selecting the 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 the CPU Prescaler Selection bits in the CPU Select register (CPUSEL.CPUDIV), resulting in a CPU clock frequency determined by this equation:

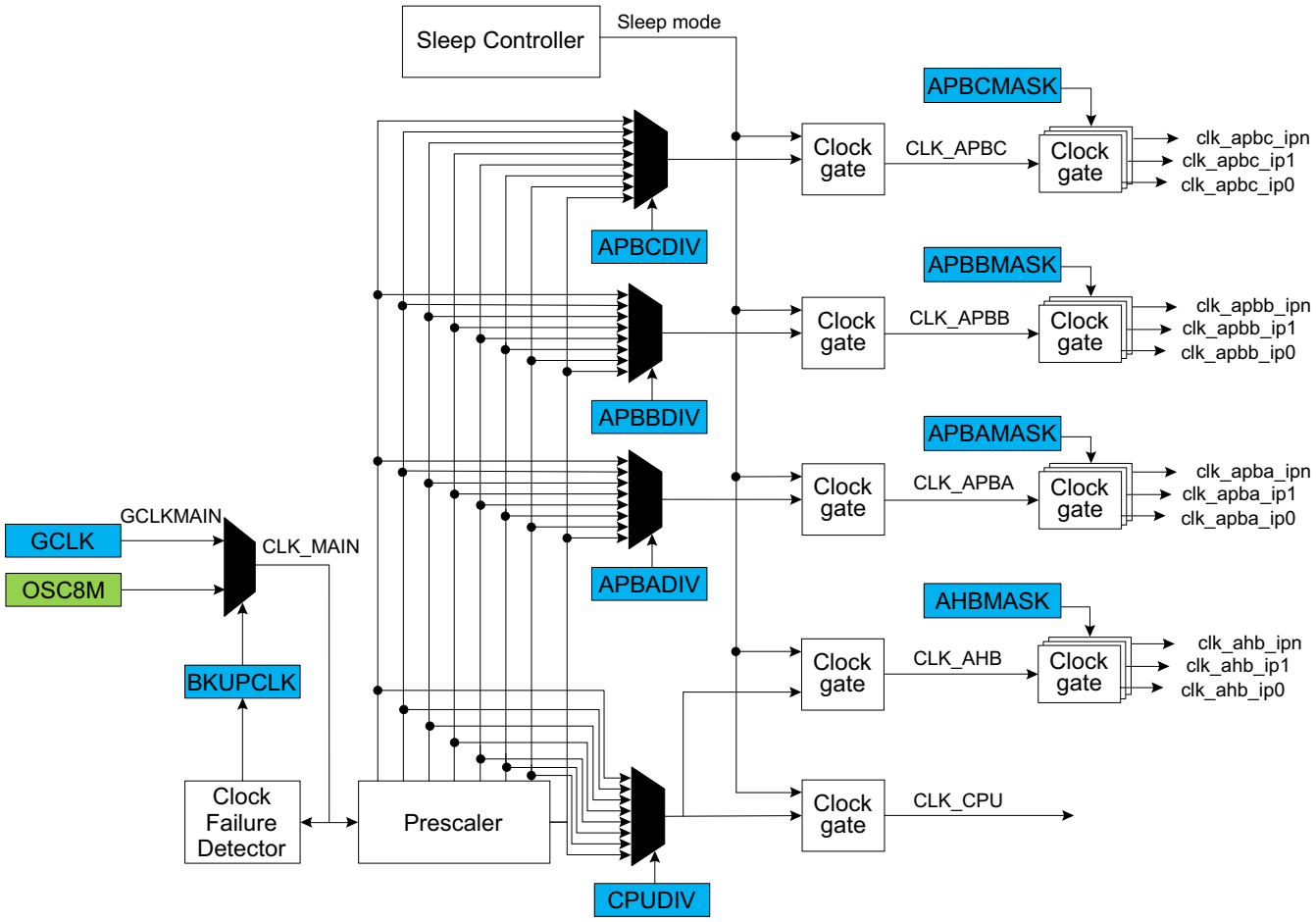
$$f_{CPU} = \frac{f_{main}}{2^{CPUDIV}}$$

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.

Note that the AHB clock is always equal to the CPU clock.

CPUSEL and APBxSEL can be written without halting or disabling peripheral modules. Writing CPUSEL and APBxSEL 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. This way, it is possible to, for example, scale the CPU speed according to the required performance, while keeping the APBx frequency constant.

Figure 15-2. Synchronous Clock Selection and Prescaler



15.6.2.5 Clock Ready Flag

There is a slight delay from when CPUSEL and APBxSEL are written until the new clock setting becomes effective. During this interval, the Clock Ready flag in the Interrupt Flag Status and Clear register (INTFLAG.CKRDY) will read as zero. If CKRDY in the INTENSET register is written to one, the Power Manager interrupt can be triggered when the new clock setting is effective. CKSEL must not be re-written while CKRDY is zero, or the system may become unstable or hang.

15.6.2.6 Peripheral Clock Masking

It is possible to disable or enable the clock for a peripheral in the AHB or APBx clock domain by writing the corresponding bit in the Clock Mask register (APBxMASK) to zero or one. Refer to Table 15-1 for the default state of each of the peripheral clocks.

Table 15-1. Peripheral Clock Default State

Peripheral Clock	Default State
CLK_PAC0_APB	Enabled
CLK_PM_APB	Enabled
CLK_SYSCTRL_APB	Enabled
CLK_GCLK_APB	Enabled
CLK_WDT_APB	Enabled
CLK_RTC_APB	Enabled
CLK_EIC_APB	Enabled
CLK_PAC1_APB	Enabled
CLK_DSU_APB	Enabled
CLK_NVMCTRL_APB	Enabled
CLK_PORT_APB	Enabled
CLK_PAC2_APB	Disabled
CLK_SERCOMx_APB	Disabled
CLK_TCx_APB	Disabled
CLK_ADC_APB	Enabled
CLK_AC_APB	Disabled
CLK_DAC_APB	Disabled
CLK_PTC_APB	Disabled

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 (for instance, AHB and APB), in which case it will have several mask bits.

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 NVM Controller (NVMCTRL) will cause a problem if the CPU needs to read from the flash memory. 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.

15.6.2.7 Clock Failure Detector

This mechanism allows the main clock to be switched automatically to the safe OSC8M clock when the main clock source is considered off. This may happen for instance when an external crystal oscillator is selected as the clock source for the main clock and the crystal fails. The mechanism is to designed to detect, during a OSCULP32K clock period, at least one rising edge of the main clock. If no rising edge is seen, the clock is considered failed.

The clock failure detector is enabled by writing a one to the Clock Failure Detector Enable bit in CTRL (CFDEN.CTRL).

As soon as the Clock Failure Detector Enable bit (CTRL.CFDEN) is one, the clock failure detector (CFD) will monitor the divided main clock. When a clock failure is detected, the main clock automatically switches to the OSC8M clock and the Clock Failure Detector flag in the interrupt Flag Status and Clear register (INTFLAG.CFD) is generated, if enabled. The BKUPCLK bit in the CTRL register is set by hardware to indicate that the main clock comes from OSC8M. The

GCLKMAIN clock source can be selected again by writing a zero to the CTRL.BKUPCLK bit. Writing the bit does not fix the failure, however.

Note 1: The detector does not monitor while the main clock is temporarily unavailable (startup time after a wake-up, etc.) or in sleep mode.

Note 2: The clock failure detector must not be enabled if the source of the main clock is not significantly faster than the OSCULP32K clock. For instance, if GCLKMAIN is the internal 32kHz RC, then the clock failure detector must be disabled.

15.6.2.8 Reset Controller

The latest reset cause is available in RCAUSE, and can be read during the application boot sequence in order to determine proper action.

There are two groups of reset sources:

- Power Reset: Resets caused by an electrical issue.
- User Reset: Resets caused by the application.

The table below lists the parts of the device that are reset, depending on the reset type.

Table 15-2. Effects of the Different Reset Events

	Power Reset	User Reset	
	POR, BOD12, BOD33	External Reset	WDT Reset, SysResetReq
RTC All the 32kHz sources WDT with ALWAYSON feature GCLK with WRTLOCK feature	Y	N	N
Debug logic	Y	Y	N
Others	Y	Y	Y

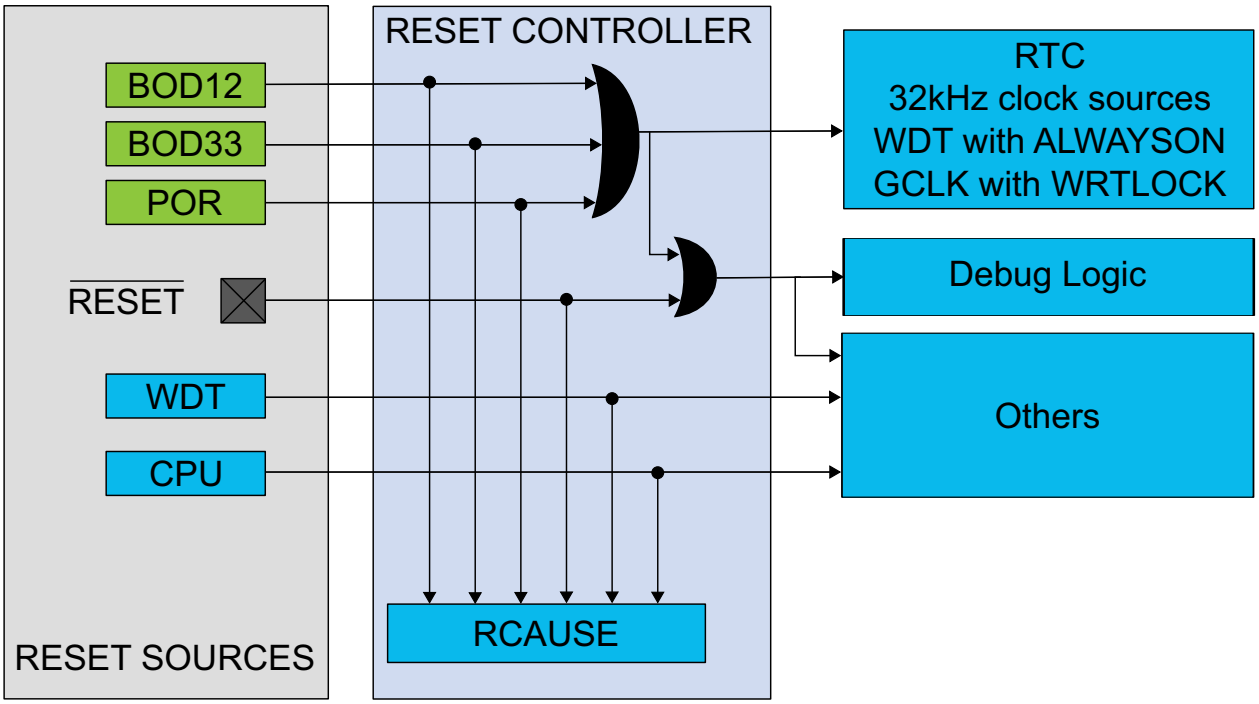
The external reset is generated when pulling the $\overline{\text{RESET}}$ pin low. This pin has an internal pull-up, and does not need to be driven externally during normal operation.

The POR, BOD12 and BOD33 reset sources are generated by their corresponding module in the System Controller Interface (SYSCTRL).

The WDT reset is generated by the Watchdog Timer.

The System Reset Request (SysResetReq) is a software reset generated by the CPU when asserting the SYSRESETREQ bit located in the Reset Control register of the CPU (See the ARM® Cortex® Technical Reference Manual on http://www.arm.com).

Figure 15-3. Reset Controller



15.6.2.9 Sleep Mode Controller

Sleep mode is activated by the Wait For Interrupt instruction (WFI). The Idle bits in the Sleep Mode register (SLEEP.IDLE) and the SLEEPDEEP bit of the System Control register of the CPU should be used as argument to select the level of the sleep mode.

There are two main types of sleep mode:

- IDLE mode: The CPU is stopped. Optionally, some synchronous clock domains are stopped, depending on the IDLE argument. Regulator operates in normal mode
- STANDBY mode: All clock sources are stopped, except those where the RUNSTDBY bit is set. Regulator operates in low-power mode

Table 15-3. Sleep Mode Entry and Exit Table

Mode	Level	Mode Entry	Wake-Up Sources
IDLE	0	SCR.SLEEPDEEP = 0 SLEEP.IDLE=Level WFI	Synchronous (APB, AHB), asynchronous
	1		Synchronous (APB), asynchronous
	2		Asynchronous
STANDBY		SCR.SLEEPDEEP = 1 WFI	Asynchronous

Table 15-4. Sleep Mode Overview

SLEEP Mode	SLEEP. IDLE	CPU Clock	AHB Clock	APB Clock	Clock Sources[1][2]	Main Clock	Regulator Mode	RAM Mode
IDLE	0	Stop	Run	Run	Run if (ONDEMAND == 0) | ((ONDEMAND == 1) & (Module request))	Run	Normal	Normal
	1	Stop	Stop	Run		Run		
	2	Stop	Stop	Stop		Run		
STANDBY		Stop	Stop	Stop	Run if RUNSTDBY & ((ONDEMAND == 0) | ((ONDEMAND == 1) & (Module request)))	Stop	Low power	Low power

Notes: 1. Refer to "SYSCTRL – System Controller" on page 127 to determine the clock sources.
2. Refer to "On-demand, Clock Requests" on page 76.

IDLE Mode

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

The CPU is stopped. To further reduce power consumption, the user can disable the clocking of modules and clock sources by configuring the SLEEP.IDLE bit group. The module will be halted regardless of the bit settings of the mask registers in the Power Manager (PM.AHBMASK, PM.APBxMASK).

Regulator operates in normal mode.

- Entering IDLE mode: The IDLE mode is entered by executing the WFI instruction. Additionally, if the SLEEPONEXIT bit in the ARM Cortex System Control register (SCR) is set, the IDLE mode will also be entered when the CPU 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 IDLE mode, the user must configure the IDLE mode configuration bit group and must write a zero to the SCR.SLEEPDEEP bit.
- Exiting IDLE mode: The processor wakes the system up when it detects any non-masked interrupt with sufficient priority to cause exception entry. The system goes back to the ACTIVE mode. The CPU and affected modules are restarted.

STANDBY Mode

The STANDBY modes allow achieving very low power consumption.

In this mode, all clocks are stopped except those which are kept running if requested by a running module or have the ONDEMAND bit set to zero. For example, the RTC can operate in STANDBY mode. In this case, its GCLK clock source will also be enabled.

The regulator and the RAM operate in low-power mode.

A SLEEPONEXIT feature is also available.

- Entering STANDBY mode: This mode is entered by executing the WFI instruction with the SCR.SLEEPDEEP bit of the CPU is written to 1.
- Exiting STANDBY mode: Any peripheral able to generate an asynchronous interrupt can wake up the system. For example, a module running on a GCLK clock can trigger an interrupt. When the enabled asynchronous wake-up event occurs and the system is woken up, the device will either execute the interrupt service routine or continue the normal program execution according to the Priority Mask Register (PRIMASK) configuration of the CPU.

15.6.3 Additional Features

15.6.4 Interrupts

The peripheral has the following interrupt sources:

- Clock Ready flag
- Clock failure detector

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the peripheral is reset. An interrupt flag is cleared by writing a one to the corresponding bit in the INTFLAG register. Each peripheral can have one interrupt request line per interrupt source or one common interrupt request line for all the interrupt sources. Refer to “Nested Vector Interrupt Controller” on page 24 for details. If the peripheral has one common interrupt request line for all the interrupt sources, the user must read the INTFLAG register to determine which interrupt condition is present.

15.6.5 Events

Not applicable.

15.6.6 Sleep Mode Operation

In all IDLE sleep modes, the power manager is still running on the selected main clock.

In STANDDBY sleep mode, the power manager is frozen and is able to go back to ACTIVE mode upon any asynchronous interrupt.

15.7 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRL	7:0				BKUPCLK		CFDEN		
0x01	SLEEP	7:0							IDLE[1:0]	
0x02	Reserved									
0x03	Reserved									
…	…	…	…	…	…	…	…	…	…	…
0x06	Reserved									
0x07	Reserved									
0x08	CPUSEL	7:0						CPUDIV[2:0]		
0x09	APBASEL	7:0						APBADIV[2:0]		
0x0A	APBBSEL	7:0						APBBDIV[2:0]		
0x0B	APBCSEL	7:0						APBCDIV[2:0]		
0x0C	Reserved									
0x0D	Reserved									
…										
0x12	Reserved									
0x13	Reserved									
0x14	AHBMASK	7:0				NVMCTRL	DSU	HPB2	HPB1	HPB0
0x15		15:8								
0x16		23:16								
0x17		31:24								
0x18	APBAMASK	7:0		EIC	RTC	WDT	GCLK	SYSCTRL	PM	PAC0
0x19		15:8								
0x1A		23:16								
0x1B		31:24								
0x1C	APBBMASK	7:0					PORT	NVMCTRL	DSU	PAC1
0x1D		15:8								
0x1E		23:16								
0x1F		31:24								
0x20	APBCMASK	7:0	SERCOM5	SERCOM4	SERCOM3	SERCOM2	SERCOM1	SERCOM0	EVSYS	PAC2
0x21		15:8	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
0x22		23:16					PTC	DAC	AC	ADC
0x23		31:24								
0x24	Reserved									
0x25	Reserved									
…	…	…	…	…	…	…	…	…	…	…
0x32	Reserved									

Offset	Name	Bit Pos.								
0x33	Reserved									
0x34	INTENCLR	7:0							CFD	CKRDY
0x35	INTENSET	7:0							CFD	CKRDY
0x36	INTFLAG	7:0							CFD	CKRDY
0x37	Reserved									
0x38	RCAUSE	7:0		SYST	WDT	EXT		BOD33		POR

15.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Exception for APBASEL, APBBSEL and APBCSEL: These registers must only be accessed with 8-bit access.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 102 for details.

15.8.1 Control

Name: CTRL

Offset: 0x00

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				BKUPCLK		CFDEN		
Access	R	R	R	R/W	R	R/W	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – BKUPCLK: Backup Clock Select**
  This bit is set by hardware when a clock failure is detected.
  0: The GCLKMAIN clock is selected for the main clock.
  1: The OSC8M backup clock is selected for the main clock.

- **Bit 3 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 2 – CFDEN: Clock Failure Detector Enable**
  0: The clock failure detector is disabled.
  1: The clock failure detector is enabled.

- **Bits 1:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

15.8.2 Sleep Mode

Name: SLEEP

Offset: 0x01

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							IDLE[1:0]	
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – IDLE[1:0]: Idle Mode Configuration**
  These bits select the Idle mode configuration after a WFI instruction.

Table 15-5. Idle Mode Configuration

IDLE[1:0]	Description
0x0	The CPU clock domain is stopped
0x1	The CPU and AHB clock domains are stopped
0x2	The CPU, AHB and APB clock domains are stopped
0x3	Reserved

15.8.3 CPU Clock Select

Name: CPUSEL

Offset: 0x08

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						CPUDIV[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – CPUDIV[2:0]: CPU Prescaler Selection**
  These bits define the division ratio of the main clock prescaler (2^n).

Table 15-6. CPU Clock Frequency Ratio

CPUDIV[1:0]	Description
0x0	Divide by 1
0x1	Divide by 2
0x2	Divide by 4
0x3	Divide by 8
0x4	Divide by 16
0x5	Divide by 32
0x6	Divide by 64
0x7	Divide by 128

15.8.4 APBA Clock Select

Name: APBASEL

Offset: 0x09

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						APBADIV[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – APBADIV[2:0]: APBA Prescaler Selection**
  These bits define the division ratio of the APBA clock prescaler (2^n).

Table 15-7. APBA Prescaler Selection

APBADIV[1:0]	Description
0x0	Divide by 1
0x1	Divide by 2
0x2	Divide by 4
0x3	Divide by 8
0x4	Divide by 16
0x5	Divide by 32
0x6	Divide by 64
0x7	Divide by 128

15.8.5 APBB Clock Select

Name: APBBSEL

Offset: 0x0A

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						APBBDIV[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – APBBDIV[2:0]: APBB Prescaler Selection**
  These bits define the division ratio of the APBB clock prescaler (2^n).

Table 15-8. APBB Prescaler Selection

APBBDIV[1:0]	Description
0x0	Divide by 1
0x1	Divide by 2
0x2	Divide by 4
0x3	Divide by 8
0x4	Divide by 16
0x5	Divide by 32
0x6	Divide by 64
0x7	Divide by 128

15.8.6 APBC Clock Select

Name: APBCSEL

Offset: 0x0B

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						APBCDIV[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – APBCDIV[2:0]: APBC Prescaler Selection**
  These bits define the division ratio of the APBC clock prescaler (2^n).

Table 15-9. APBC Prescaler Selection

APBCDIV[1:0]	Description
0x0	Divide by 1
0x1	Divide by 2
0x2	Divide by 4
0x3	Divide by 8
0x4	Divide by 16
0x5	Divide by 32
0x6	Divide by 64
0x7	Divide by 128

15.8.7 AHB Mask

Name: AHBMASK

Offset: 0x14

Reset: 0x0000001F

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				NVMCTRL	DSU	HPB2	HPB1	HPB0
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

- **Bits 31:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 4:0 – NVMCTRL, DSU, HPB2, HPB1, HPB0: AHB Clock Enable**
  For any bit:
  0: The AHB clock for the corresponding module is stopped.
  1: The AHB clock for the corresponding module is enabled.

15.8.8 APBA Mask

Name: APBAMASK

Offset: 0x18

Reset: 0x0000007F

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		EIC	**RTC**	**WDT**	**GCLK**	**SYSCTRL**	**PM**	**PAC0**
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	1	1	1	1

- **Bits 31:7 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 6:0 – EIC, RTC, WDT, GCLK, SYSCTRL, PM, PAC0: APB Clock Enable**
  For any bit:
  0: The APBA clock for the corresponding module is stopped.
  1: The APBA clock for the corresponding module is enabled.

15.8.9 APBB Mask

Name: APBBMASK

Offset: 0x1C

Reset: 0x0000001F

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					PORT	NVMCTRL	DSU	PAC1
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	1

- **Bits 31:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:0 – PORT, NVMCTRL, DSU, PAC1: APB Clock Enable**
  For any bit:
  0: The APBB clock for the corresponding module is stopped.
  1: The APBB clock for the corresponding module is enabled.

15.8.10 APBC Mask

Name: APBCMASK

Offset: 0x20

Reset: 0x00010000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
					PTC	DAC	AC	ADC
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SERCOM5	SERCOM4	SERCOM3	SERCOM2	SERCOM1	SERCOM0	EVSYS	PAC2
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:20 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 19:0 – PTC, DAC, AC, ADC, TC7, TC6, TC5, TC4, TC3, TC2, TC1, TC0, SERCOM5, SERCOM4, SERCOM3, SERCOM2, SERCOM1, SERCOM0, EVSYS, PAC2: APB Clock Enable**
  For any bit:
  0: The APBC clock for the corresponding module is stopped.
  1: The APBC clock for the corresponding module is enabled.

15.8.11 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set (INTENSET) register.

Name: INTENCLR

Offset: 0x34

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							CFD	CKRDY
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – CFD: Clock Failure Detector Interrupt Enable**
  0: The Clock Failure Detector interrupt is disabled.
  1: The Clock Failure Detector interrupt is enabled and an interrupt request will be generated when the Clock Failure Detector Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Clock Failure Detector Interrupt Enable bit and the corresponding interrupt request.

- **Bit 0 – CKRDY: Clock Ready Interrupt Enable**
  0: The Clock Ready interrupt is disabled.
  1: The Clock Ready interrupt is enabled and will generate an interrupt request when the Clock Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Clock Ready Interrupt Enable bit and the corresponding interrupt request.

15.8.12 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear (INTENCLR) register.

Name: INTENSET

Offset: 0x35

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							CFD	**CKRDY**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – CFD: Clock Failure Detector Interrupt Enable**
  0: The Clock Failure Detector interrupt is disabled.
  1: The Clock Failure Detector interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Clock Failure Detector Interrupt Enable bit and enable the Clock Failure Detector interrupt.

- **Bit 0 – CKRDY: Clock Ready Interrupt Enable**
  0: The Clock Ready interrupt is disabled.
  1: The Clock Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Clock Ready Interrupt Enable bit and enable the Clock Ready interrupt.

15.8.13 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x36

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
							CFD	**CKRDY**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – CFD: Clock Failure Detector**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next cycle after a clock failure detector occurs and will generate an interrupt request if INTENCLR/SET.CFD is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Clock Failure Detector Interrupt flag.

- **Bit 0 – CKRDY: Clock Ready**
  This flag is cleared by writing a one to the flag.
  This flag is set when the synchronous CPU and APBx clocks have frequencies as indicated in the CPUSEL and APBxSEL registers, and will generate an interrupt if INTENCLR/SET.CKRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Clock Ready Interrupt flag.

15.8.14 Reset Cause

Name: RCAUSE

Offset: 0x38

Reset: Latest Reset Source

Property: –

Bit	7	6	5	4	3	2	1	0
		SYST	WDT	EXT		BOD33		POR
Access	R	R	R	R	R	R	R	R
Reset	0	X	X	X	0	X	X	X

- **Bit 7 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.
- **Bit 6 – SYST: System Reset Request**
  This bit is set if a system reset request has been performed. Refer to the Cortex processor documentation for more details.
- **Bit 5 – WDT: Watchdog Reset**
  This flag is set if a Watchdog Timer reset occurs.
- **Bit 4 – EXT: External Reset**
  This flag is set if an external reset occurs.
- **Bit 3 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.
- **Bit 2 – BOD33: Brown Out 33 Detector Reset**
  This flag is set if a BOD33 reset occurs.
- **Bit 1 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.
- **Bit 0 – POR: Power-On Reset**
  This flag is set if a POR occurs.

16. SYSCTRL – System Controller

16.1 Overview

The System Controller (SYSCTRL) provides a user interface to the XOSC, XOSC32K, OSC32K, OSCULP32K, OSC8M, DFLL48M, BOD33, and VREF.

Through the interface registers, it is possible to enable, disable, calibrate and monitor the SYSCTRL sub-peripherals.

All sub-peripheral statuses are collected in the Power and Clocks Status register (PCLKSR). They can additionally trigger interrupts upon status changes via the INTENSET, INTENCLR and INTFLAG registers.

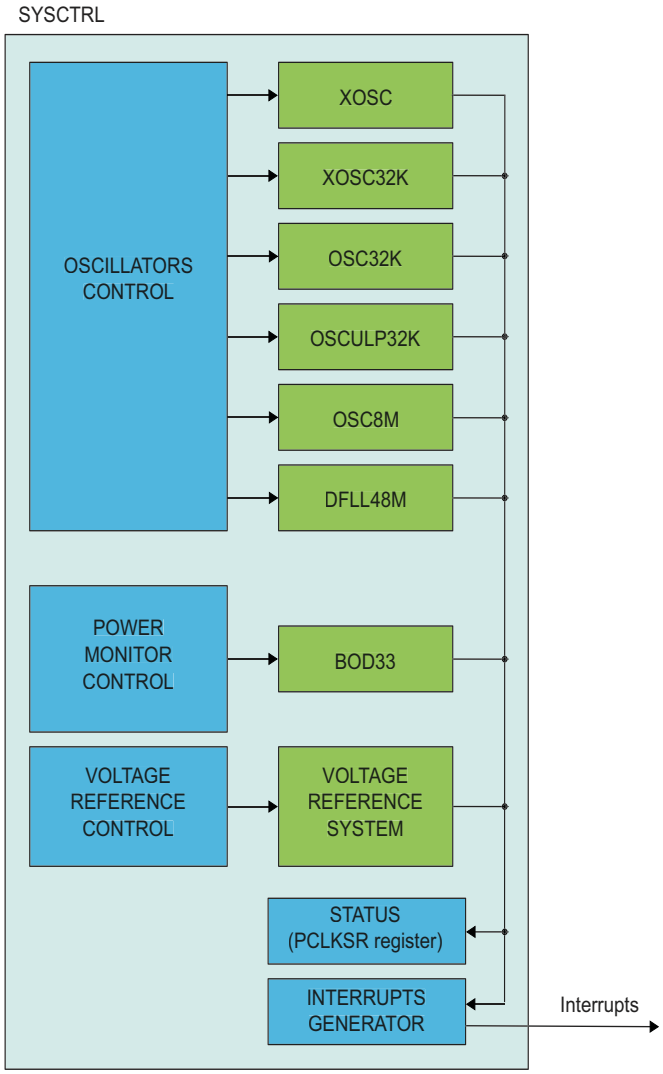
Additionally, BOD33 interrupts can be used to wake up the device from standby mode upon a programmed brown-out detection.

16.2 Features

- 0.4-32MHz Crystal Oscillator (XOSC)
  - Tunable gain control
  - Programmable start-up time
  - Crystal or external input clock on XIN I/O
- 32.768kHz Crystal Oscillator (XOSC32K)
  - Automatic or manual gain control
  - Programmable start-up time
  - Crystal or external input clock on XIN32 I/O
- 32.768kHz High Accuracy Internal Oscillator (OSC32K)
  - Frequency fine tuning
  - Programmable start-up time
- 32.768kHz Ultra Low Power Internal Oscillator (OSCULP32K)
  - Ultra low power, always-on oscillator
  - Frequency fine tuning
  - Calibration value loaded from Flash Factory Calibration at reset
- 8MHz Internal Oscillator (OSC8M)
  - Fast startup
  - Output frequency fine tuning
  - 4/2/1MHz divided output frequencies available
  - Calibration value loaded from Flash Factory Calibration at reset
- Digital Frequency Locked Loop (DFLL48M)
  - Internal oscillator with no external components
  - 48MHz output frequency
  - Operates standalone as a high-frequency programmable oscillator in open loop mode
  - Operates as an accurate frequency multiplier against a known frequency in closed loop mode
- 3.3V Brown-Out Detector (BOD33)
  - Programmable threshold
  - Threshold value loaded from Flash User Calibration at startup
  - Triggers resets or interrupts
  - Operating modes:
    - Continuous mode
    - Sampled mode for low power applications (programmable refresh frequency)
  - Hysteresis
- Voltage Reference System (VREF)
  - Bandgap voltage generator with programmable calibration value
  - Temperature sensor
  - Bandgap calibration value loaded from Flash Factory Calibration at startup

16.3 Block Diagram

Figure 16-1. SYSCTRL Block Diagram



16.4 Signal Description

Signal Name	Type	Description
XIN	Analog Input	Multipurpose Crystal Oscillator or external clock generator input
XOUT	Analog Output	External Multipurpose Crystal Oscillator output
XIN32	Analog Input	32kHz Crystal Oscillator or external clock generator input
XOUT32	Analog Output	32kHz Crystal Oscillator output

The I/O lines are automatically selected when XOSC or XOSC32K are enabled.

16.5 Product Dependencies

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

16.5.1 I/O Lines

I/O lines are configured by SYSCTRL when either XOSC or XOSC32K are enabled, and need no user configuration.

16.5.2 Power Management

The SYSCTRL can continue to operate in any sleep mode where the selected source clock is running. The SYSCTRL interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to “PM – Power Manager” on page 100 for details on the different sleep modes.

16.5.3 Clocks

The SYSCTRL gathers controls for all device oscillators and provides clock sources to the Generic Clock Controller (GCLK). The available clock sources are: XOSC, XOSC32K, OSC32K, OSCULP32K, OSC8M and DFLL48M.

The SYSCTRL bus clock (CLK_SYSCTRL_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_SYSCTRL_APB can be found in the Peripheral Clock Masking section in the “PM – Power Manager” on page 100.

The clock used by BOD33in sampled mode is asynchronous to the user interface clock (CLK_SYSCTRL_APB). Likewise, the DFLL48M control logic uses the DFLL oscillator output, which is also asynchronous to the user interface clock (CLK_SYSCTRL_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to “Synchronization” on page 137 for further details.

16.5.4 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the SYSCTRL interrupts requires the interrupt controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

16.5.5 Debug Operation

When the CPU is halted in debug mode, the SYSCTRL continues normal operation. If the SYSCTRL 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.

If a debugger connection is detected by the system, BOD33 reset will be blocked.

16.5.6 Register Access Protection

All registers with write-access are optionally write-protected by the peripheral access controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

16.5.7 Analog Connections

The 32.768kHz crystal must be connected between the XIN32 and XOUT32 pins, and the 0.4-32MHz crystal must be connected between the XIN and XOUT pins, along with any required load capacitors. For details on recommended oscillator characteristics and capacitor load. Refer to the “Electrical Characteristics” on page 558 for details.

16.6 Functional Description

16.6.1 Principle of Operation

XOSC, XOSC32K, OSC32K, OSCULP32K, OSC8M, DFLL48M, BOD33, and VREF are configured via SYSCTRL control registers. Through this interface, the sub-peripherals are enabled, disabled or have their calibration values updated.

The Power and Clocks Status register gathers different status signals coming from the sub-peripherals controlled by the SYSCTRL. The status signals can be used to generate system interrupts, and in some cases wake up the system from standby mode, provided the corresponding interrupt is enabled.

16.6.2 External Multipurpose Crystal Oscillator (XOSC) Operation

The XOSC can operate in two different modes:

- External clock, with an external clock signal connected to the XIN pin
- Crystal oscillator, with an external 0.4-32MHz crystal

The XOSC can be used as a clock source for generic clock generators, as described in the “GCLK – Generic Clock Controller” on page 78.

At reset, the XOSC is disabled, and the XIN/XOUT pins can be used as General Purpose I/O (GPIO) pins or by other peripherals in the system. When XOSC is enabled, the operating mode determines the GPIO usage. When in crystal oscillator mode, the XIN and XOUT pins are controlled by the SYSCTRL, and GPIO functions are overridden on both pins. When in external clock mode, the only XIN pin will be overridden and controlled by the SYSCTRL, while the XOUT pin can still be used as a GPIO pin.

The XOSC is enabled by writing a one to the Enable bit in the External Multipurpose Crystal Oscillator Control register (XOSC.ENABLE). To enable the XOSC as a crystal oscillator, the XTAL Enable bit (XOSC.XTALEN) must written to one. If XOSC.XTALEN is zero, external clock input will be enabled.

When in crystal oscillator mode (XOSC.XTALEN is one), the External Multipurpose Crystal Oscillator Gain (XOSC.GAIN) must be set to match the external crystal oscillator frequency. If the External Multipurpose Crystal Oscillator Automatic Amplitude Gain Control (XOSC.AMPGC) is one, the oscillator amplitude will be automatically adjusted, and in most cases result in a lower power consumption.

The XOSC will behave differently in different sleep modes based on the settings of XOSC.RUNSTDBY, XOSC.ONDEMAND and XOSC.ENABLE:

XOSC.RUNSTDBY	XOSC.ONDEMAND	XOSC.ENABLE	Sleep Behavior
-	-	0	Disabled
0	0	1	Always run in IDLE sleep modes. Disabled in STANDBY sleep mode.
0	1	1	Only run in IDLE sleep modes if requested by a peripheral. Disabled in STANDBY sleep mode.
1	0	1	Always run in IDLE and STANDBY sleep modes.
1	1	1	Only run in IDLE or STANDBY sleep modes if requested by a peripheral.

After a hard reset, or when waking up from a sleep mode where the XOSC was disabled, the XOSC will need a certain amount of time to stabilize on the correct frequency. This start-up time can be configured by changing the Oscillator

Start-Up Time bit group (XOSC.STARTUP) in the External Multipurpose Crystal Oscillator Control register. During the start-up time, the oscillator output is masked to ensure that no unstable clock propagates to the digital logic. The External Multipurpose Crystal Oscillator Ready bit in the Power and Clock Status register (PCLKSR.XOSCRDY) is set when the external clock or crystal oscillator is stable and ready to be used as a clock source. An interrupt is generated on a zero-to-one transition on PCLKSR.XOSCRDY if the External Multipurpose Crystal Oscillator Ready bit in the Interrupt Enable Set register (INTENSET.XOSCRDY) is set.

16.6.3 32kHz External Crystal Oscillator (XOSC32K) Operation

The XOSC32K can operate in two different modes:

- External clock, with an external clock signal connected to XIN32
- Crystal oscillator, with an external 32.768kHz crystal connected between XIN32 and XOUT32

The XOSC32K can be used as a source for generic clock generators, as described in the “GCLK – Generic Clock Controller” on page 78.

At power-on, reset the XOSC32K is disabled, and the XIN32/XOUT32 pins can be used as General Purpose I/O (GPIO) pins or by other peripherals in the system. When XOSC32K is enabled, the operating mode determines the GPIO usage. When in crystal oscillator mode, XIN32 and XOUT32 are controlled by the SYSCTRL, and GPIO functions are overridden on both pins. When in external clock mode, only the XIN32 pin will be overridden and controlled by the SYSCTRL, while the XOUT32 pin can still be used as a GPIO pin.

The external clock or crystal oscillator is enabled by writing a one to the Enable bit (XOSC32K.ENABLE) in the 32kHz External Crystal Oscillator Control register. To enable the XOSC32K as a crystal oscillator, a one must be written to the XTAL Enable bit (XOSC32K.XTALEN). If XOSC32K.XTALEN is zero, external clock input will be enabled.

The oscillator is disabled by writing a zero to the Enable bit (XOSC32K.ENABLE) in the 32kHz External Crystal Oscillator Control register while keeping the other bits unchanged. Writing to the XOSC32K.ENABLE bit while writing to other bits may result in unpredictable behavior. The oscillator remains enabled in all sleep modes if it has been enabled beforehand. The start-up time of the 32kHz External Crystal Oscillator is selected by writing to the Oscillator Start-Up Time bit group (XOSC32K.STARTUP) in the in the 32kHz External Crystal Oscillator Control register. The SYSCTRL masks the oscillator output during the start-up time to ensure that no unstable clock propagates to the digital logic. The 32kHz External Crystal Oscillator Ready bit (PCLKSR.XOSC32KRDY) in the Power and Clock Status register is set when the oscillator is stable and ready to be used as a clock source. An interrupt is generated on a zero-to-one transition of PCLKSR.XOSC32KRDY if the 32kHz External Crystal Oscillator Ready bit (INTENSET.XOSC32KRDY) in the Interrupt Enable Set Register is set.

As a crystal oscillator usually requires a very long start-up time (up to one second), the 32kHz External Crystal Oscillator will keep running across resets, except for power-on reset (POR).

The XOSC32K has a 32.768kHz output enabled by writing a one to the 32kHz External Crystal Oscillator 32kHz Output Enable bit (XOSC32K.EN32K) in the 32kHz External Crystal Oscillator Control register. The XOSC32K also has a 1.024kHz clock output enabled by writing a one to the 32kHz External Crystal Oscillator 1kHz Output Enable bit (XOSC32K.EN1K) in the External 32kHz Crystal Oscillator Control register. XOSC32K.EN32K and XOSC32K.EN1K are only usable when XIN32 is connected to a crystal, and not when an external digital clock is applied on XIN32.

16.6.4 32kHz Internal Oscillator (OSC32K) Operation

The OSC32K provides a tunable, low-speed and low-power clock source.

The OSC32K can be used as a source for the generic clock generators, as described in the “GCLK – Generic Clock Controller” on page 78.

The OSC32K is disabled by default. The OSC32K is enabled by writing a one to the 32kHz Internal Oscillator Enable bit (OSC32K.ENABLE) in the 32kHz Internal Oscillator Control register. It is disabled by writing a zero to OSC32K.ENABLE. The OSC32K has a 32.768kHz output enabled by writing a one to the 32kHz Internal Oscillator 32kHz Output Enable bit (OSC32K.EN32K). The OSC32K also has a 1.024kHz clock output enabled by writing a one to the 32kHz Internal Oscillator 1kHz Output Enable bit (OSC32K.EN1K).

The frequency of the OSC32K oscillator is controlled by the value in the 32kHz Internal Oscillator Calibration bits (OSC32K.CALIB) in the 32kHz Internal Oscillator Control register. The CALIB value must be written by the user. Flash Factory Calibration values are stored in the non-volatile memory. When writing to the Calibration bits, the user must wait for the PCLKSR.OSC32KRDY bit to go high before the value is committed to the oscillator.

16.6.5 32kHz Ultra Low Power Internal Oscillator (OSCULP32K) Operation

The OSCULP32K provides a tunable, low-speed and ultra-low-power clock source. The OSCULP32K is factory-calibrated under typical voltage and temperature conditions. The OSCULP32K should be preferred to the OSC32K whenever the power requirements are prevalent over frequency stability and accuracy.

The OSCULP32K can be used as a source for the generic clock generators, as described in the “GCLK – Generic Clock Controller” on page 78.

The OSCULP32K is enabled by default after a power-on reset (POR) and will always run except during POR. The OSCULP32K has a 32.768kHz output and a 1.024kHz output that are always running.

The frequency of the OSCULP32K oscillator is controlled by the value in the 32kHz Ultra Low Power Internal Oscillator Calibration bits (OSCULP32K.CALIB) in the 32kHz Ultra Low Power Internal Oscillator Control register. OSCULP32K.CALIB is automatically loaded from Flash Factory Calibration during startup, and is used to compensate for process variation, as described in the “Electrical Characteristics” on page 558. The calibration value can be overridden by the user by writing to OSCULP32K.CALIB.

16.6.6 8MHz Internal Oscillator (OSC8M) Operation

OSC8M is an internal oscillator operating in open-loop mode and generating an 8MHz frequency. The OSC8M is factory-calibrated under typical voltage and temperature conditions.

OSC8M is the default clock source that is used after a power-on reset (POR). The OSC8M can be used as a source for the generic clock generators, as described in the “GCLK – Generic Clock Controller” on page 78, as well as function as the backup clock if a main clock failure is detected.

OSC8M is enabled by writing a one to the Oscillator Enable bit (OSC8M.ENABLE) in the OSC8M Control register, and disabled by writing a zero to this bit. When enabling OSC8M, OSC8M.ENABLE must be read back until it reads one. The user must ensure that the OSC8M is fully disabled before enabling it, and that the OSC8M is fully enabled before disabling it by reading OSC8M.ENABLE.

The frequency of the OSC8M oscillator is controlled by the value in the calibration bits (OSC8M.CALIB) in the OSC8M Control register. CALIB is automatically loaded from Flash Factory Calibration during startup, and is used to compensate for process variation, as described in the “Electrical Characteristics” on page 558.

The user can control the oscillation frequency by writing to the Frequency Range (FRANGE) and Calibration (CALIB) bit groups in the 8MHz RC Oscillator Control register (OSC8M). The FRANGE and CALIB bit groups should only be updated when the OSC8M is disabled. As this is in open-loop mode, the frequency will be voltage, temperature and process dependent. Refer to the “Electrical Characteristics” on page 558 for details.

OSC8M is automatically switched off in certain sleep modes to reduce power consumption, as described in the “PM – Power Manager” on page 100.

16.6.7 Digital Frequency Locked Loop (DFLL48M) Operation

The DFLL48M 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 the reference clock to get high accuracy on the output clock (CLK_DFLL48M).

The DFLL48M can be used as a source for the generic clock generators, as described in the “GCLK – Generic Clock Controller” on page 78.

16.6.7.1 Basic Operation

Open-Loop Operation

After any reset, the open-loop mode is selected. When operating in open-loop mode, the output frequency of the DFLL48M will be determined by the values written to the DFLL Coarse Value bit group and the DFLL Fine Value bit group (DFLLVAL.COARSE and DFLLVAL.FINE) in the DFLL Value register.

It is possible to change the values of DFLLVAL.COARSE and DFLLVAL.FINE and thereby the output frequency of the DFLL48M output clock, CLK_DFLL48M, while the DFLL48M is enabled and in use. CLK_DFLL48M is ready to be used when PCLKSR.DFLLRDY is set after enabling the DFLL48M.

Closed-Loop Operation

In closed-loop operation, the output frequency is continuously regulated against a reference clock. Once the multiplication factor is set, the oscillator fine tuning is automatically adjusted. The DFLL48M must be correctly configured before closed-loop operation can be enabled. After enabling the DFLL48M, it must be configured in the following way:

1. Enable and select a reference clock (CLK_DFLL48M_REF). CLK_DFLL48M_REF is Generic Clock Channel 0 (DFLL48M_Reference). Refer to "GCLK – Generic Clock Controller" on page 78 for details.
2. Select the maximum step size allowed in finding the Coarse and Fine values by writing the appropriate values to the DFLL Coarse Maximum Step and DFLL Fine Maximum Step bit groups (DFLLMUL.CSTEP and DFLLMUL.FSTEP) in the DFLL Multiplier register. A small step size will ensure low overshoot on the output frequency, but will typically result in longer lock times. A high value might give a large overshoot, but will typically provide faster locking. DFLLMUL.CSTEP and DFLLMUL.FSTEP should not be higher than 50% of the maximum value of DFLLVAL.COARSE and DFLLVAL.FINE, respectively.
3. Select the multiplication factor in the DFLL Multiply Factor bit group (DFLLMUL.MUL) in the DFLL Multiplier register. Care must be taken when choosing DFLLMUL.MUL so that the output frequency does not exceed the maximum frequency of the device. If the target frequency is below the minimum frequency of the DFLL48M, the output frequency will be equal to the DFLL minimum frequency.
4. Start the closed loop mode by writing a one to the DFLL Mode Selection bit (DFLLCTRL.MODE) in the DFLL Control register.

The frequency of CLK_DFLL48M ($F_{clkdfll48m}$) is given by:

$$F_{clkdfll48m} = DFLLMUL \cdot MUL \times F_{clkdfll48mref}$$

where $F_{clkdfll48mref}$ is the frequency of the reference clock (CLK_DFLL48M_REF). DFLLVAL.COARSE and DFLLVAL.FINE are read-only in closed-loop mode, and are controlled by the frequency tuner shown in Figure 16-1 to meet user specified frequency. In closed-loop mode, the value in DFLLVAL.COARSE is used by the frequency tuner as a starting point for Coarse. Writing DFLLVAL.COARSE to a value close to the final value before entering closed-oop mode will reduce the time needed to get a lock on Coarse.

Frequency Locking

The locking of the frequency in closed-loop mode is divided into two stages. In the first, coarse stage, the control logic quickly finds the correct value for DFLLVAL.COARSE and sets the output frequency to a value close to the correct frequency. On coarse lock, the DFLL Locked on Coarse Value bit (PCLKSR.DFLLLOCKC) in the Power and Clocks Status register will be set.

In the second, fine stage, the control logic tunes the value in DFLLVAL.FINE so that the output frequency is very close to the desired frequency. On fine lock, the DFLL Locked on Fine Value bit (PCLKSR.DFLLLOCKF) in the Power and Clocks Status register will be set.

Interrupts are generated by both PCLKSR.DFLLLOCKC and PCLKSR.DFLLLOCKF if INTENSET.DFLLOCKC or INTENSET.DFLLOCKF are written to one.

CLK_DFLL48M is ready to be used when the DFLL Ready bit (PCLKSR.DFLLRDY) in the Power and Clocks Status register is set, but the accuracy of the output frequency depends on which locks are set. For lock times, refer to the "Electrical Characteristics" on page 558.

Frequency Error Measurement

The ratio between CLK_DFLL48M_REF and CLK48M_DFLL is measured automatically when the DFLL48M is in closed-loop mode. The difference between this ratio and the value in DFLLMUL.MUL is stored in the DFLL Multiplication Ratio Difference bit group(DFLLVAL.DIFF) in the DFLL Value register. The relative error on CLK_DFLL48M compared to the target frequency is calculated as follows:

$$ERROR = \frac{DIFF}{MUL}$$

Drift Compensation

If the Stable DFLL Frequency bit (DFLLCTRL.STABLE) in the DFLL Control register is zero, the frequency tuner will automatically compensate for drift in the CLK_DFLL48M without losing either of the locks. This means that DFLLVAL.FINE can change after every measurement of CLK_DFLL48M. If the DFLLVAL.FINE value overflows or underflows due to large drift in temperature and/or voltage, the DFLL Out Of Bounds bit (PCLKSR.DFLLOOB) in the Power and Clocks Status register will be set. After an Out of Bounds error condition, the user must rewrite DFLLMUL.MUL to ensure correct CLK_DFLL48M frequency. An interrupt is generated on a zero-to-one transition on PCLKSR.DFLLOOB if the DFLL Out Of Bounds bit (INTENSET.DFLLOOB) in the Interrupt Enable Set register is set. This interrupt will also be set if the tuner is not able to lock on the correct Coarse value.

Reference Clock Stop Detection

If CLK_DFLL48M_REF stops or is running at a very low frequency (slower than CLK_DFLL48M/(2 * MUL_{MAX})), the DFLL Reference Clock Stopped bit (PCLKSR.DFLLRCS) in the Power and Clocks Status register will be set. Detecting a stopped reference clock can take a long time, on the order of 2^{17} CLK_DFLL48M cycles. When the reference clock is stopped, the DFLL48M will operate as if in open-loop mode. Closed-loop mode operation will automatically resume if the CLK_DFLL48M_REF is restarted. An interrupt is generated on a zero-to-one transition on PCLKSR.DFLLRCS if the DFLL Reference Clock Stopped bit (INTENSET.DFLLRCS) in the Interrupt Enable Set register is set.

16.6.7.2 Additional Features

Dealing with Delay in the DFLL in Closed-Loop Mode

The time from selecting a new CLK_DFLL48M frequency until this frequency is output by the DFLL48M can be up to several microseconds. If the value in DFLLMUL.MUL is small, this can lead to instability in the DFLL48M locking mechanism, which can prevent the DFLL48M from achieving locks. To avoid this, a chill cycle, during which the CLK_DFLL48M frequency is not measured, can be enabled. The chill cycle is enabled by default, but can be disabled by writing a one to the DFLL Chill Cycle Disable bit (DFLLCTRL.CCDIS) in the DFLL Control register. Enabling chill cycles might double the lock time.

Another solution to this problem consists of using less strict lock requirements. This is called Quick Lock (QL), which is also enabled by default, but it can be disabled by writing a one to the Quick Lock Disable bit (DFLLCTRL.QLDIS) in the DFLL Control register. The Quick Lock might lead to a larger spread in the output frequency than chill cycles, but the average output frequency is the same.

Wake from Sleep Modes

DFLL48M can optionally reset its lock bits when it is disabled. This is configured by the Lose Lock After Wake bit (DFLLCTRL.LLAW) in the DFLL Control register. If DFLLCTRL.LLAW is zero, the DFLL48M will be re-enabled and start running with the same configuration as before being disabled, 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 DFLLCTRL.STABLE is zero. If DFLLCTRL.LLAW is one when disabling the DFLL48M, the DFLL48M will lose all its locks, and needs to regain these through the full lock sequence.

Accuracy

There are three main factors that determine the accuracy of $F_{clkdfll48m}$. 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 $F_{clkdfll48m}$ is low, i.e., the ratio between the CLK_DFLL48M frequency and the CLK_DFLL48M_REF frequency is small, then the DFLL48M might lock at a frequency that is lower than the targeted frequency. It is recommended to use a reference clock frequency of 32kHz or lower to avoid this issue for low target frequencies.
- The accuracy of the reference clock.

16.6.8 3.3V Brown-Out Detector Operation

The 3.3V BOD monitors the 3.3V VDDANA supply (BOD33). It supports continuous or sampling modes.

The threshold value action (reset the device or generate an interrupt), the Hysteresis configuration, as well as the enable/disable settings are loaded from Flash User Calibration at startup, and can be overridden by writing to the corresponding BOD33 register bit groups.

16.6.8.1 3.3V Brown-Out Detector (BOD33)

The 3.3V Brown-Out Detector (BOD33) monitors the VDDANA supply and compares the voltage with the brown-out threshold level set in the BOD33 Level bit group (BOD33.LEVEL) in the BOD33 register. The Brown-Out Detector can generate either an interrupt or a reset when VDDANA crosses below the brown-out threshold level. The BOD33 detection status can be read from the BOD33 Detection bit (PCLKSR.BOD33DET) in the Power and Clocks Status register.

At startup or at power-on reset (POR), the BOD33 register values are loaded from the Flash User Row. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

16.6.8.2 Continuous Mode

When the BOD33 Mode bit (BOD33.MODE) in the BOD33 register is written to zero and the BOD33 is enabled, the BOD33 operates in continuous mode. In this mode, the BOD33 is continuously monitoring the VDDANA supply voltage.

Continuous mode is the default mode for BOD33.

16.6.8.3 Sampling Mode

The sampling mode is a low-power mode where the BOD33 is being repeatedly enabled on a sampling clock’s ticks. The BOD33 will monitor the supply voltage for a short period of time and then go to a low-power disabled state until the next sampling clock tick.

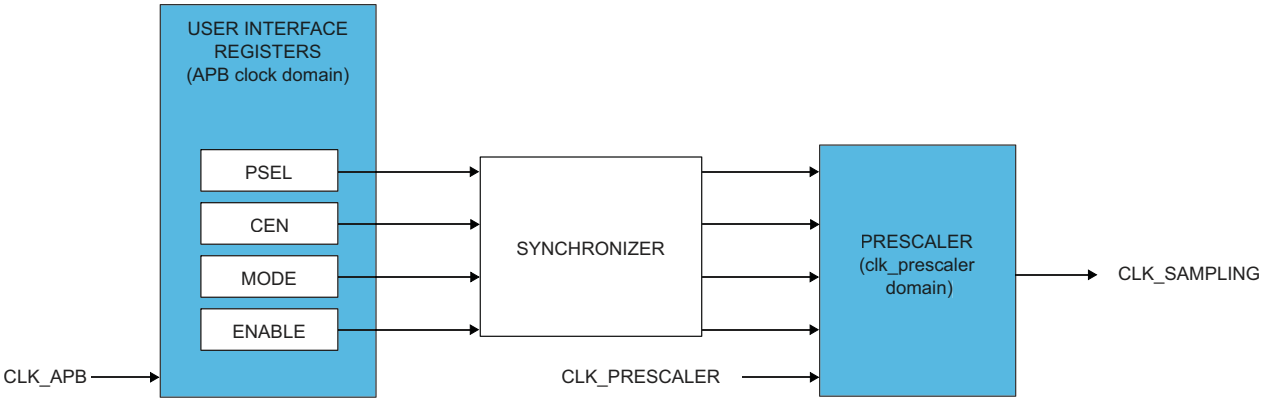
Sampling mode is enabled by writing one to BOD33.MODE. The frequency of the clock ticks ($F_{clksampling}$) is controlled by the BOD33 Prescaler Select bit group (BOD33.PSEL) in the BOD33 register.

$$F_{clksampling} = \frac{F_{clkprescaler}}{2^{(PSEL+1)}}$$

The prescaler signal ($F_{clkprescaler}$) is a 1kHz clock, output from the32kHz Ultra Low Power Oscillator, OSCULP32K.

As the sampling mode clock is different from the APB clock domain, synchronization among the clocks is necessary. Figure 16-2 shows a block diagram of the sampling mode. The BOD33Synchronization Ready bits (PCLKSR.B33SRDY) in the Power and Clocks Status register show the synchronization ready status of the synchronizer. Writing attempts to the BOD33 register are ignored while PCLKSR.B33SRDY is zero.

Figure 16-2. Sampling Mode Block diagram



The BOD33 Clock Enable bit (BOD33.CEN) in the BOD33 registershould always be disabled before changing the prescaler value. To change the prescaler value for the BOD33during sampling mode, the following steps need to be taken:

1. Wait until the PCLKSR.B33SRDY bit is set.
2. Write the selected value to the BOD33.PSEL bit group.

16.6.8.4 Hysteresis

The hysteresis functionality can be used in both continuous and sampling mode. Writing a one to the BOD33 Hysteresis bit (BOD33.HYST) in the BOD33 register will add hysteresis to the BOD33 threshold level.

16.6.9 Voltage Reference System Operation

The Voltage Reference System (VREF) consists of a Bandgap Reference Voltage Generator and a temperature sensor.

The Bandgap Reference Voltage Generator is factory-calibrated under typical voltage and temperature conditions.

At reset, the VREF.CAL register value is loaded from Flash Factory Calibration.

The temperature sensor can be used to get an absolute temperature in the temperature range of CMIN to CMAX degrees Celsius. The sensor will output a linear voltage proportional to the temperature. The output voltage and temperature range are located in the “Electrical Characteristics” on page 558. To calculate the temperature from a measured voltage, the following formula can be used:

$$C_{MIN} + (Vmes - Vout_{MAX})\frac{\Delta temperature}{\Delta voltage}$$

16.6.9.1 User Control of the Voltage Reference System

To enable the temperature sensor, write a one the Temperature Sensor Enable bit (VREF.TSEN) in the VREF register.

The temperature sensor can be redirected to the ADC for conversion. The Bandgap Reference Voltage Generator output can also be routed to the ADC if the Bandgap Output Enable bit (VREF.BGOUTEN) in the VREF register is set.

The Bandgap Reference Voltage Generator output level is determined by the CALIB bit group (VREF.CALIB) value in the VREF register.The default calibration value can be overridden by the user by writing to the CALIB bit group.

16.6.10 Interrupts

The SYSCTRL has the following interrupt sources:

- XOSCRDY - Multipurpose Crystal Oscillator Ready: A “0-to-1” transition on the PCLKSR.XOSCRDY bit is detected
- XOSC32KRDY - 32kHz Crystal Oscillator Ready: A “0-to-1” transition on the PCLKSR.XOSC32KRDY bit is detected
- OSC32KRDY - 32kHz Internal Oscillator Ready: A “0-to-1” transition on the PCLKSR.OSC32KRDY bit is detected
- OSC8MRDY - 8MHz Internal Oscillator Ready: A “0-to-1” transition on the PCLKSR.OSC8MRDY bit is detected
- DFLLRDY - DFLL48M Ready: A “0-to-1” transition on the PCLKSR.DFLLRDY bit is detected

- DFLLOOB - DFLL48M Out Of Boundaries: A “0-to-1” transition on the PCLKSR.DFLLOOB bit is detected
- DFLLLOCKF - DFLL48M Fine Lock: A “0-to-1” transition on the PCLKSR.DFLLLOCKF bit is detected
- DFLLLOCKC - DFLL48M Coarse Lock: A “0-to-1” transition on the PCLKSR.DFLLLOCKC bit is detected
- DFLLRCS - DFLL48M Reference Clock has Stopped: A “0-to-1” transition on the PCLKSR.DFLLRCS bit is detected
- BOD33RDY - BOD33 Ready: A “0-to-1” transition on the PCLKSR.BOD33RDY bit is detected
- BOD33DET - BOD33 Detection: A “0-to-1” transition on the PCLKSR.BOD33DET bit is detected
- B33SRDY - BOD33 Synchronization Ready: A “0-to-1” transition on the PCLKSR.B33SRDY bit is detected

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the SYSCTRL is reset. See the INTFLAG register for details on how to clear interrupt flags.

The SYSCTRL has one common interrupt request line for all the interrupt sources.The user must read the INTFLAG register to determine which interrupt condition is present. Refer to the INTFLAG register for details.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to the “Nested Vector Interrupt Controller” on page 24 for details.

16.6.11 Synchronization

Due to the multiple clock domains, values in the DFLL48M control 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 DFLL48M control registers, the user must check that the DFLL Ready bit (PCLKSR.DFLLRDY) in PCLKSR is set to one. When this bit is set, the DFLL48M can be configured and CLK_DFLL48M is ready to be used. Any write to any of the DFLL48M control registers while DFLLRDY is zero will be ignored. An interrupt is generated on a zero-to-one transition of DFLLRDY if the DFLLRDY bit (INTENSET.DFLLDY) in the Interrupt Enable Set register is set.

In order to read from any of the DFLL48M configuration registers, the user must request a read synchronization by writing a one to DFLLSYNC.READREQ. The registers can be read only when PCLKSR.DFLLRDY is set. If DFLLSYNC.READREQ is not written before a read, a synchronization will be started, and the bus will be halted until the synchronization is complete. Reading the DFLL48M registers when the DFLL48M is disabled will not halt the bus.

The prescaler counter used to trigger one-shot brown-out detections also operates asynchronously from the peripheral bus. As a consequence, the prescaler registers require synchronization when written or read. The synchronization results in a delay from when the initialization of the write or read operation begins until the operation is complete.

The write-synchronization is triggered by a write to the BOD33 control register. The Synchronization Ready bit (PCLKSR.B33SRDY) in the PCLKSR register will be cleared when the write-synchronization starts and set when the write-synchronization is complete. When the write-synchronization is ongoing (PCLKSR.B33SRDYis zero), an attempt to do any of the following will cause the peripheral bus to stall until the synchronization is complete:

- Writing to the BOD33control register
- Reading the BOD33 control register that was written

The user can either poll PCLKSR.B33SRDY or use the INTENSET.B33SRDY interrupts to check when the synchronization is complete. It is also possible to perform the next read/write operation and wait, as this next operation will be completed after the ongoing read/write operation is synchronized.

16.7 Register Summary

Table 16-1. SYSCTRL Register Summary

Offset	Name	Bit Pos.								
0x00	INTENCLR	7:0	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
0x01		15:8					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
0x02		23:16								
0x03		31:24								
0x04	INTENSET	7:0	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
0x05		15:8					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
0x06		23:16								
0x07		31:24								
0x08	INTFLAG	7:0	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
0x09		15:8					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
0x0A		23:16								
0x0B		31:24								
0x0C	PCLKSR	7:0	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
0x0D		15:8					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
0x0E		23:16								
0x0F		31:24								
0x10	XOSC	7:0	ONDEMAND	RUNSTDBY				XTALEN	ENABLE	
0x11		15:8	STARTUP[3:0]				AMPGC	GAIN[2:0]		
0x12	Reserved									
0x13	Reserved									
0x14	XOSC32K	7:0	ONDEMAND	RUNSTDBY	AAMPEN	EN1K	EN32K	XTALEN	ENABLE	
0x15		15:8				WRTLOCK		STARTUP[2:0]		
0x16	Reserved									
0x17	Reserved									
0x18	OSC32K	7:0	ONDEMAND	RUNSTDBY			EN1K	EN32K	ENABLE	
0x19		15:8				WRTLOCK		STARTUP[2:0]		
0x1A		23:16		CALIB[6:0]						
0x1B		31:24								
0x1C	OSCULP32K	7:0	WRTLOCK			CALIB[4:0]				
0x1D	Reserved									
0x1E	Reserved									
0x1F	Reserved									
0x20	OSC8M	7:0	ONDEMAND	RUNSTDBY					ENABLE	
0x21		15:8							PRESC[1:0]	
0x22		23:16	CALIB[7:0]							
0x23		31:24	FRANGE[1:0]				CALIB[11:8]			
0x24	DFLLCTRL	7:0	ONDEMAND	RUNSTDBY		LLAW	STABLE	MODE	ENABLE	
0x25		15:8							QLDIS	CCDIS
0x26	Reserved									
0x27	Reserved									

Table 16-1. SYSCTRL Register Summary (Continued)

Offset	Name	Bit Pos.								
0x28	DFLLVAL	7:0	FINE[7:0]							
0x29		15:8	COARSE[5:0]						FINE[9:8]	
0x2A		23:16	DIFF[7:0]							
0x2B		31:24	DIFF[15:8]							
0x2C	DFLLMUL	7:0	MUL[7:0]							
0x2D		15:8	MUL[15:8]							
0x2E		23:16	FSTEP[7:0]							
0x2F		31:24	CSTEP[5:0]						FSTEP[9:8]	
0x30	DFLLSYNC	7:0	READREQ							
0x31	Reserved									
0x32	Reserved									
0x33	Reserved									
0x34	BOD33	7:0		RUNSTDBY		ACTION[1:0]		HYST	ENABLE	
0x35		15:8	PSEL[3:0]						CEN	MODE
0x36		23:16			LEVEL[5:0]					
0x37		31:24								
0x38	Reserved									
...										
0x3F	Reserved									
0x40	VREF	7:0						BGOUTEN	TSEN	
0x41		15:8								
0x42		23:16	CALIB[7:0]							
0x43		31:24						CALIB[10:8]		
0x45	Reserved									
0x46	Reserved									
0x47	Reserved									
0x51	Reserved									
0x52	Reserved									
0x53	Reserved									

16.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 129 and the “PAC – Peripheral Access Controller” on page 27 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Synchronized property in each individual register description. Refer to “Synchronization” on page 137 for details.

16.8.1 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x00

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 11 – B33SRDY: BOD33 Synchronization Ready Interrupt Enable**
  0: The BOD33 Synchronization Ready interrupt is disabled.
  1: The BOD33 Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Synchronization Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the BOD33 Synchronization Ready Interrupt Enable bit, which disables the BOD33 Synchronization Ready interrupt.

- **Bit 10 – BOD33DET: BOD33 Detection Interrupt Enable**
  0: The BOD33 Detection interrupt is disabled.

1: The BOD33 Detection interrupt is enabled, and an interrupt request will be generated when the BOD33 Detection Interrupt flag is set.

Writing a zero to this bit has no effect.

Writing a one to this bit will clear the BOD33 Detection Interrupt Enable bit, which disables the BOD33 Detection interrupt.

- **Bit 9 – BOD33RDY: BOD33 Ready Interrupt Enable**

  0: The BOD33 Ready interrupt is disabled.

  1: The BOD33 Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Ready Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will clear the BOD33 Ready Interrupt Enable bit, which disables the BOD33 Ready interrupt.

- **Bit 8 – DFLLRCS: DFLL Reference Clock Stopped Interrupt Enable**

  0: The DFLL Reference Clock Stopped interrupt is disabled.

  1: The DFLL Reference Clock Stopped interrupt is enabled, and an interrupt request will be generated when the DFLL Reference Clock Stopped Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will clear the DFLL Reference Clock Stopped Interrupt Enable bit, which disables the DFLL Reference Clock Stopped interrupt.

- **Bit 7 – DFLLLCKC: DFLL Lock Coarse Interrupt Enable**

  0: The DFLL Lock Coarse interrupt is disabled.

  1: The DFLL Lock Coarse interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Coarse Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will clear the DFLL Lock Coarse Interrupt Enable bit, which disables the DFLL Lock Coarse interrupt.

- **Bit 6 – DFLLLCKF: DFLL Lock Fine Interrupt Enable**

  0: The DFLL Lock Fine interrupt is disabled.

  1: The DFLL Lock Fine interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Fine Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will clear the DFLL Lock Fine Interrupt Enable bit, which disables the DFLL Lock Fine interrupt.

- **Bit 5 – DFLLOOB: DFLL Out Of Bounds Interrupt Enable**

  0: The DFLL Out Of Bounds interrupt is disabled.

  1: The DFLL Out Of Bounds interrupt is enabled, and an interrupt request will be generated when the DFLL Out Of Bounds Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will clear the DFLL Out Of Bounds Interrupt Enable bit, which disables the DFLL Out Of Bounds interrupt.

- **Bit 4 – DFLLRDY: DFLL Ready Interrupt Enable**

  0: The DFLL Ready interrupt is disabled.

  1: The DFLL Ready interrupt is enabled, and an interrupt request will be generated when the DFLL Ready Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will clear the DFLL Ready Interrupt Enable bit, which disables the DFLL Ready interrupt.

- **Bit 3 – OSC8MRDY: OSC8M Ready Interrupt Enable**
  0: The OSC8M Ready interrupt is disabled.
  1: The OSC8M Ready interrupt is enabled, and an interrupt request will be generated when the OSC8M Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the OSC8M Ready Interrupt Enable bit, which disables the OSC8M Ready interrupt.
- **Bit 2 – OSC32KRDY: OSC32K Ready Interrupt Enable**
  0: The OSC32K Ready interrupt is disabled.
  1: The OSC32K Ready interrupt is enabled, and an interrupt request will be generated when the OSC32K Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the OSC32K Ready Interrupt Enable bit, which disables the OSC32K Ready interrupt.
- **Bit 1 – XOSC32KRDY: XOSC32K Ready Interrupt Enable**
  0: The XOSC32K Ready interrupt is disabled.
  1: The XOSC32K Ready interrupt is enabled, and an interrupt request will be generated when the XOSC32K Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the XOSC32K Ready Interrupt Enable bit, which disables the XOSC32K Ready interrupt.
- **Bit 0 – XOSCRDY: XOSC Ready Interrupt Enable**
  0: The XOSC Ready interrupt is disabled.
  1: The XOSC Ready interrupt is enabled, and an interrupt request will be generated when the XOSC Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the XOSC Ready Interrupt Enable bit, which disables the XOSC Ready interrupt.

16.8.2 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 11 – B33SRDY: BOD33 Synchronization Ready Interrupt Enable**
  0: The BOD33 Synchronization Ready interrupt is disabled.
  1: The BOD33 Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Synchronization Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the BOD33 Synchronization Ready Interrupt Enable bit, which enables the BOD33 Synchronization Ready interrupt.

- **Bit 10 – BOD33DET: BOD33 Detection Interrupt Enable**
  0: The BOD33 Detection interrupt is disabled.

1: The BOD33 Detection interrupt is enabled, and an interrupt request will be generated when the BOD33 Detection Interrupt flag is set.

Writing a zero to this bit has no effect.

Writing a one to this bit will set the BOD33 Detection Interrupt Enable bit, which enables the BOD33 Detection interrupt.

- **Bit 9 – BOD33RDY: BOD33 Ready Interrupt Enable**

  0: The BOD33 Ready interrupt is disabled.

  1: The BOD33 Ready interrupt is enabled, and an interrupt request will be generated when the BOD33 Ready Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will set the BOD33 Ready Interrupt Enable bit, which enables the BOD33 Ready interrupt.

- **Bit 8 – DFLLRCS: DFLL Reference Clock Stopped Interrupt Enable**

  0: The DFLL Reference Clock Stopped interrupt is disabled.

  1: The DFLL Reference Clock Stopped interrupt is enabled, and an interrupt request will be generated when the DFLL Reference Clock Stopped Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will set the DFLL Reference Clock Stopped Interrupt Enable bit, which enables the DFLL Reference Clock Stopped interrupt.

- **Bit 7 – DFLLLCKC: DFLL Lock Coarse Interrupt Enable**

  0: The DFLL Lock Coarse interrupt is disabled.

  1: The DFLL Lock Coarse interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Coarse Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will set the DFLL Lock Coarse Interrupt Enable bit, which enables the DFLL Lock Coarse interrupt.

- **Bit 6 – DFLLLCKF: DFLL Lock Fine Interrupt Enable**

  0: The DFLL Lock Fine interrupt is disabled.

  1: The DFLL Lock Fine interrupt is enabled, and an interrupt request will be generated when the DFLL Lock Fine Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will set the DFLL Lock Fine Interrupt Disable/Enable bit, disable the DFLL Lock Fine interrupt and set the corresponding interrupt request.

- **Bit 5 – DFLLOOB: DFLL Out Of Bounds Interrupt Enable**

  0: The DFLL Out Of Bounds interrupt is disabled.

  1: The DFLL Out Of Bounds interrupt is enabled, and an interrupt request will be generated when the DFLL Out Of Bounds Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will set the DFLL Out Of Bounds Interrupt Enable bit, which enables the DFLL Out Of Bounds interrupt.

- **Bit 4 – DFLLRDY: DFLL Ready Interrupt Enable**

  0: The DFLL Ready interrupt is disabled.

  1: The DFLL Ready interrupt is enabled, and an interrupt request will be generated when the DFLL Ready Interrupt flag is set.

  Writing a zero to this bit has no effect.

  Writing a one to this bit will set the DFLL Ready Interrupt Enable bit, which enables the DFLL Ready interrupt and set the corresponding interrupt request.

- **Bit 3 – OSC8MRDY: OSC8M Ready Interrupt Enable**
  0: The OSC8M Ready interrupt is disabled.
  1: The OSC8M Ready interrupt is enabled, and an interrupt request will be generated when the OSC8M Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the OSC8M Ready Interrupt Enable bit, which enables the OSC8M Ready interrupt.
- **Bit 2 – OSC32KRDY: OSC32K Ready Interrupt Enable**
  0: The OSC32K Ready interrupt is disabled.
  1: The OSC32K Ready interrupt is enabled, and an interrupt request will be generated when the OSC32K Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the OSC32K Ready Interrupt Enable bit, which enables the OSC32K Ready interrupt.
- **Bit 1 – XOSC32KRDY: XOSC32K Ready Interrupt Enable**
  0: The XOSC32K Ready interrupt is disabled.
  1: The XOSC32K Ready interrupt is enabled, and an interrupt request will be generated when the XOSC32K Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the XOSC32K Ready Interrupt Enable bit, which enables the XOSC32K Ready interrupt.
- **Bit 0 – XOSCRDY: XOSC Ready Interrupt Enable**
  0: The XOSC Ready interrupt is disabled.
  1: The XOSC Ready interrupt is enabled, and an interrupt request will be generated when the XOSC Ready Interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the XOSC Ready Interrupt Enable bit, which enables the XOSC Ready interrupt.

16.8.3 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x08

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Note: Depending on the fuse settings, various bits of the INTFLAG register can be set to one at startup. Therefore the user should clear those bits before using the corresponding interrupts.

- **Bits 31:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 11 – B33SRDY: BOD33 Synchronization Ready**
  This flag is cleared by writing a one to it.
  This flag is set on a zero-to-one transition of the BOD33 Synchronization Ready bit in the Status register (PCLKSR.B33SRDY) and will generate an interrupt request if INTENSET.B33SRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the BOD33 Synchronization Ready interrupt flag

- **Bit 10 – BOD33DET: BOD33 Detection**
  This flag is cleared by writing a one to it.
  This flag is set on a zero-to-one transition of the BOD33 Detection bit in the Status register (PCLKSR.BOD33DET) and will generate an interrupt request if INTENSET.BOD33DET is one.

Writing a zero to this bit has no effect.

Writing a one to this bit clears the BOD33 Detection interrupt flag.

- **Bit 9 – BOD33RDY: BOD33 Ready**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the BOD33 Ready bit in the Status register (PCLKSR.BOD33RDY) and will generate an interrupt request if INTENSET.BOD33RDY is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the BOD33 Ready interrupt flag.

- **Bit 8 – DFLLRCS: DFLL Reference Clock Stopped**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the DFLL Reference Clock Stopped bit in the Status register (PCLKSR.DFLLRCS) and will generate an interrupt request if INTENSET.DFLLRCS is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the DFLL Reference Clock Stopped interrupt flag.

- **Bit 7 – DFLLLCKC: DFLL Lock Coarse**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the DFLL Lock Coarse bit in the Status register (PCLKSR.DFLLLCKC) and will generate an interrupt request if INTENSET.DFLLLCKC is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the DFLL Lock Coarse interrupt flag.

- **Bit 6 – DFLLLCKF: DFLL Lock Fine**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the DFLL Lock Fine bit in the Status register (PCLKSR.DFLLLCKF) and will generate an interrupt request if INTENSET.DFLLLCKF is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the DFLL Lock Fine interrupt flag.

- **Bit 5 – DFLLOOB: DFLL Out Of Bounds**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the DFLL Out Of Bounds bit in the Status register (PCLKSR.DFLLOOB) and will generate an interrupt request if INTENSET.DFLLOOB is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the DFLL Out Of Bounds interrupt flag.

- **Bit 4 – DFLLRDY: DFLL Ready**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the DFLL Ready bit in the Status register (PCLKSR.DFLLRDY) and will generate an interrupt request if INTENSET.DFLLRDY is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the DFLL Ready interrupt flag.

- **Bit 3 – OSC8MRDY: OSC8M Ready**
  This flag is cleared by writing a one to it.

  This flag is set on a zero-to-one transition of the OSC8M Ready bit in the Status register (PCLKSR.OSC8MRDY) and will generate an interrupt request if INTENSET.OSC8MRDY is one.

  Writing a zero to this bit has no effect.

  Writing a one to this bit clears the OSC8M Ready interrupt flag.

- **Bit 2 – OSC32KRDY: OSC32K Ready**
  This flag is cleared by writing a one to it.
  This flag is set on a zero-to-one transition of the OSC32K Ready bit in the Status register (PCLKSR.OSC32KRDY) and will generate an interrupt request if INTENSET.OSC32KRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the OSC32K Ready interrupt flag.

- **Bit 1 – XOSC32KRDY: XOSC32K Ready**
  This flag is cleared by writing a one to it.
  This flag is set on a zero-to-one transition of the XOSC32K Ready bit in the Status register (PCLKSR.XOSC32KRDY) and will generate an interrupt request if INTENSET.XOSC32KRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the XOSC32K Ready interrupt flag.

- **Bit 0 – XOSCRDY: XOSC Ready**
  This flag is cleared by writing a one to it.
  This flag is set on a zero-to-one transition of the XOSC Ready bit in the Status register (PCLKSR.XOSCRDY) and will generate an interrupt request if INTENSET.XOSCRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the XOSC Ready interrupt flag.

16.8.4 Power and Clocks Status

Name: PCLKSR

Offset: 0x0C

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					B33SRDY	BOD33DET	BOD33RDY	DFLLRCS
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DFLLLCKC	DFLLLCKF	DFLLOOB	DFLLRDY	OSC8MRDY	OSC32KRDY	XOSC32KRDY	XOSCRDY
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 11 – B33SRDY: BOD33 Synchronization Ready**
  0: BOD33 synchronization is ongoing.
  1: BOD33 synchronization is complete.
- **Bit 10 – BOD33DET: BOD33 Detection**
  0: No BOD33 detection.
  1: BOD33 has detected that the I/O power supply is going below the BOD33 reference value.
- **Bit 9 – BOD33RDY: BOD33 Ready**
  0: BOD33 is not ready.
  1: BOD33 is ready.
- **Bit 8 – DFLLRCS: DFLL Reference Clock Stopped**
  0: DFLL reference clock is running.
  1: DFLL reference clock has stopped.

- **Bit 7 – DFLLLCKC: DFLL Lock Coarse**
  0: No DFLL coarse lock detected.
  1: DFLL coarse lock detected.
- **Bit 6 – DFLLLCKF: DFLL Lock Fine**
  0: No DFLL fine lock detected.
  1: DFLL fine lock detected.
- **Bit 5 – DFLLOOB: DFLL Out Of Bounds**
  0: No DFLL Out Of Bounds detected.
  1: DFLL Out Of Bounds detected.
- **Bit 4 – DFLLRDY: DFLL Ready**
  0: DFLL is not ready.
  1: DFLL is stable and ready to be used as a clock source.
- **Bit 3 – OSC8MRDY: OSC8M Ready**
  0: OSC8M is not ready.
  1: OSC8M is stable and ready to be used as a clock source.
- **Bit 2 – OSC32KRDY: OSC32K Ready**
  0: OSC32K is not ready.
  1: OSC32K is stable and ready to be used as a clock source.
- **Bit 1 – XOSC32KRDY: XOSC32K Ready**
  0: XOSC32K is not ready.
  1: XOSC32K is stable and ready to be used as a clock source.
- **Bit 0 – XOSCRDY: XOSC Ready**
  0: XOSC is not ready.
  1: XOSC is stable and ready to be used as a clock source.

16.8.5 External Multipurpose Crystal Oscillator (XOSC) Control

Name: XOSC

Offset: 0x10

Reset: 0x0080

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
	STARTUP[3:0]				AMPGC	GAIN[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY				XTALEN	ENABLE	
Access	R/W	R/W	R	R	R	R/W	R/W	R
Reset	1	0	0	0	0	0	0	0

- **Bits 15:12 – STARTUP[3:0]: Start-Up Time**
  These bits select start-up time for the oscillator according to Table 16-2.
  The OSCULP32K oscillator is used to clock the start-up counter.

Table 16-2. Start-UpTime for External Multipurpose Crystal Oscillator

STARTUP[3:0]	Number of OSCULP32K Clock Cycles	Number of XOSC Clock Cycles	Approximate Equivalent Time[1][2]
0x0	1	3	31µs
0x1	2	3	61µs
0x2	4	3	122µs
0x3	8	3	244µs
0x4	16	3	488µs
0x5	32	3	977µs

Table 16-2. Start-UpTime for External Multipurpose Crystal Oscillator

0x6	64	3	1953µs
0x7	128	3	3906µs
0x8	256	3	7813µs
0x9	512	3	15625µs
0xA	1024	3	31250µs
0xB	2048	3	62500µs
0xC	4096	3	125000µs
0xD	8192	3	250000µs
0xE	16384	3	500000µs
0xF	32768	3	1000000µs

Notes: 1. Actual startup time is 1 OSCULP32K cycle + 3 XOSC cycles.
2. The given time neglects the 3 XOSC cycles before OSCULP32K cycle.

- **Bit 11 – AMPGC: Automatic Amplitude Gain Control**
  0: The automatic amplitude gain control is disabled.
  1: The automatic amplitude gain control is enabled. Amplitude gain will be automatically adjusted during Crystal Oscillator operation.

- **Bits 10:8 – GAIN[2:0]: Oscillator Gain**
  These bits select the gain for the oscillator, given in table Table 16-3. The listed maximum frequencies are recommendations, and might vary based on capacitive load and crystal characteristics. Setting this bit group has no effect when the Automatic Amplitude Gain Control is active.

Table 16-3. External Multipurpose Crystal Oscillator Gain Settings

GAIN[2:0]	Recommended Max Frequency
0x0	2MHz
0x1	4MHz
0x2	8MHz
0x3	16MHz
0x4	30MHz
0x5-0x7	Reserved

- **Bit 7 – ONDEMAND: On Demand Control**
  The On Demand operation mode allows an oscillator to be enabled or disabled, depending on peripheral clock requests.
  In On Demand operation mode, i.e., if the XOSC.ONDEMAND bit has been previously written to one, the oscillator will be running only when requested by a peripheral. If there is no peripheral requesting the oscillator's clock source, the oscillator will be in a disabled state.
  If On Demand is disabled, the oscillator will always be running when enabled.
  In standby sleep mode, the On Demand operation is still active if the XOSC.RUNSTDBY bit is one. If XOSC.RUNSTDBY is zero, the oscillator is disabled.
  0: The oscillator is always on, if enabled.

1: The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

- **Bit 6 – RUNSTDBY: Run in Standby**
  This bit controls how the XOSC behaves during standby sleep mode:
  0: The oscillator is disabled in standby sleep mode.
  1: The oscillator is not stopped in standby sleep mode. If XOSC.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If XOSC.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

- **Bits 5:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – XTALEN: Crystal Oscillator Enable**
  This bit controls the connections between the I/O pads and the external clock or crystal oscillator:
  0: External clock connected on XIN. XOUT can be used as general-purpose I/O.
  1: Crystal connected to XIN/XOUT.

- **Bit 1 – ENABLE: Oscillator Enable**
  0: The oscillator is disabled.
  1: The oscillator is enabled.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

16.8.6 32kHz External Crystal Oscillator (XOSC32K) Control

Name: XOSC32K

Offset: 0x14

Reset: 0x0080

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
				WRTLOCK		STARTUP[2:0]		
Access	R	R	R	R/W	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY	AAMPEN	EN1K	EN32K	XTALEN	ENABLE	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	1	0	0	0	0	0	0	0

- **Bits 15:13 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 12 – WRTLOCK: Write Lock**
  This bit locks the XOSC32K register for futur writes to fix the XOSC32K configuration.
  0: The XOSC32K configuration is not locked.
  1: The XOSC32K configuration is locked.

- **Bit 11 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 10:8 – STARTUP[2:0]: Oscillator Start-Up Time**
  These bits select the start-up time for the oscillator according to Table 16-4
  The OSCULP32K oscillator is used to clock the start-up counter.

Table 16-4. Start-Up Time for 32kHz External Crystal Oscillator

STARTUP[2:0]	Number of OSCULP32K Clock Cycles	Number of XOSC32K Clock Cycles	Approximate Equivalent Time (OSCULP = 32kHz)[1][2]
0x0	1	3	122µs
0x1	32	3	1068µs
0x2	2048	3	62592µs
0x3	4096	3	125092µs
0x4	16384	3	500092µs
0x5	32768	3	1000092µs
0x6	65536	3	2000092µs
0x7	131072	3	4000092µs

Notes: 1. Actual startup time is 1 OSCULP32K cycle + 3 XOSC32K cycles.
2. The given time assumes an XTAL frequency of 32768Hz.

- **Bit 7 – ONDEMAND: On Demand Control**
  The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.
  In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator's clock source, the oscillator will be in a disabled state.
  If On Demand is disabled the oscillator will always be running when enabled.
  In standby sleep mode, the On Demand operation is still active if the XOSC32K.RUNSTDBY bit is one. If XOSC32K.RUNSTDBY is zero, the oscillator is disabled.
  0: The oscillator is always on, if enabled.
  1: The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

- **Bit 6 – RUNSTDBY: Run in Standby**
  This bit controls how the XOSC32K behaves during standby sleep mode:
  0: The oscillator is disabled in standby sleep mode.
  1: The oscillator is not stopped in standby sleep mode. If XOSC32K.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If XOSC32K.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

- **Bit 5 – AAMPEN: Automatic Amplitude Control Enable**
  0: The automatic amplitude control for the crystal oscillator is disabled.
  1: The automatic amplitude control for the crystal oscillator is enabled.

- **Bit 4 – EN1K: 1kHz Output Enable**
  0: The 1kHz output is disabled.
  1: The 1kHz output is enabled.

- **Bit 3 – EN32K: 32kHz Output Enable**
  0: The 32kHz output is disabled.
  1: The 32kHz output is enabled.

- **Bit 2 – XTALEN: Crystal Oscillator Enable**
  This bit controls the connections between the I/O pads and the external clock or crystal oscillator:
  0: External clock connected on XIN32. XOUT32 can be used as general-purpose I/O.
  1: Crystal connected to XIN32/XOUT32.

- **Bit 1 – ENABLE: Oscillator Enable**
  0: The oscillator is disabled.
  1: The oscillator is enabled.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

16.8.7 32kHz Internal Oscillator (OSC32K) Control

Name: OSC32K

Offset: 0x18

Reset: 0x00000080

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
		CALIB[6:0]						
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				WRTLOCK		STARTUP[2:0]		
Access	R	R	R	R/W	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY			EN1K	EN32K	ENABLE	
Access	R/W	R/W	R	R	R/W	R/W	R/W	R
Reset	1	0	0	0	0	0	0	0

- **Bits 31:23 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 22:16 – CALIB[6:0]: Oscillator Calibration**
  These bits control the oscillator calibration.
  This value must be written by the user.
  Factory calibration values can be loaded from the non-volatile memory. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21.

- **Bits 15:13 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 12 – WRTLOCK: Write Lock**
  This bit locks the OSC32K register for futur writes to fix the OSC32K configuration.
  0: The OSC32K configuration is not locked.
  1: The OSC32K configuration is locked.

- **Bit 11 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 10:8 – STARTUP[2:0]: Oscillator Start-Up Time**
  These bits select start-up time for the oscillator according to Table 16-5.
  The OSCULP32K oscillator is used as input clock to the startup counter.

Table 16-5. Start-Up Time for 32kHz Internal Oscillator

STARTUP[2:0]	Number of OSC32K clock cycles	Approximate Equivalent Time (OSCULP= 32 kHz)(1)(2)
0x0	3	92µs
0x1	4	122µs
0x2	6	183µs
0x3	10	305µs
0x4	18	549µs
0x5	34	1038µs
0x6	66	2014µs
0x7	130	3967µs

Notes:
1. Start-up time is given by STARTUP + 3 OSC32K cycles.
2. The given time assumes an XTAL frequency of 32.768kHz.

- **Bit 7 – ONDEMAND: On Demand Control**
  The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.
  In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator's clock source, the oscillator will be in a disabled state.
  If On Demand is disabled the oscillator will always be running when enabled.
  In standby sleep mode, the On Demand operation is still active if the OSC32K.RUNSTDBY bit is one. If OSC32K.RUNSTDBY is zero, the oscillator is disabled.
  0: The oscillator is always on, if enabled.
  1: The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

- **Bit 6 – RUNSTDBY: Run in Standby**
  This bit controls how the OSC32K behaves during standby sleep mode:
  0: The oscillator is disabled in standby sleep mode.
  1: The oscillator is not stopped in standby sleep mode. If OSC32K.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If OSC32K.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

- **Bits 5:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 3 – EN1K: 1kHz Output Enable**
  0: The 1kHz output is disabled.

1: The 1kHz output is enabled.

- **Bit 2 – EN32K: 32kHz Output Enable**
  0: The 32kHz output is disabled.
  1: The 32kHz output is enabled.
- **Bit 1 – ENABLE: Oscillator Enable**
  0: The oscillator is disabled.
  1: The oscillator is enabled.
- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

16.8.8 32kHz Ultra Low Power Internal Oscillator (OSCULP32K) Control

Name: OSCULP32K

Offset: 0x1C

Reset: 0xXX

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	WRTLOCK			CALIB[4:0]				
Access	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	X	X	X	X	X

- **Bit 7 – WRTLOCK: Write Lock**
  This bit locks the OSCULP32K register for futur writes to fix the OSCULP32K configuration.
  0: The OSCULP32K configuration is not locked.
  1: The OSCULP32K configuration is locked.

- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 4:0 – CALIB[4:0]: Oscillator Calibration**
  These bits control the oscillator calibration.
  These bits are loaded from Flash Calibration at startup.

16.8.9 8MHz Internal Oscillator (OSC8M) Control

Name: OSC8M

Offset: 0x20

Reset: 0x0XXX0082

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	FRANGE[1:0]				CALIB[11:8]			
Access	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	X	X	X	X

Bit	23	22	21	20	19	18	17	16
	CALIB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	X	X	X	X	X	X	X	x

Bit	15	14	13	12	11	10	9	8
							PRESC[1:0]	
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY					ENABLE	
Access	R/W	R/W	R	R	R	R	R/W	R
Reset	1	0	0	0	0	0	1	0

- **Bits 31:30 – FRANGE[1:0]: Oscillator Frequency Range**
  These bits control the oscillator frequency range according to Table 16-6.

Table 16-6. Oscillator Frequency Range

FRANGE[1:0]	Description
0x0	4 to 6MHz
0x1	6 to 8MHz
0x2	8 to 11MHz
0x3	11 to 15MHz

- **Bits 29:28 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 27:16 – CALIB[11:0]: Oscillator Calibration**
  These bits control the oscillator calibration. The calibration field is split in two:

CALIB[11:7] is for temperature calibration

CALIB[6:0] is for overall process calibration

These bits are loaded from Flash Calibration at startup.

- **Bits 15:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 9:8 – PRESC[1:0]: Oscillator Prescaler**
  These bits select the oscillator prescaler factor setting according to the Table 16-7.

Table 16-7. Oscillator Prescaler

PRESC[1:0]	Description
0x0	1
0x1	2
0x2	4
0x3	8

- **Bit 7 – ONDEMAND: On Demand Control**
  The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.

  In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator's clock source, the oscillator will be in a disabled state.

  If On Demand is disabled the oscillator will always be running when enabled.

  In standby sleep mode, the On Demand operation is still active if the OSC8M.RUNSTDBY bit is one. If OSC8M.RUNSTDBY is zero, the oscillator is disabled.

  0: The oscillator is always on, if enabled.

  1: The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

- **Bit 6 – RUNSTDBY: Run in Standby**
  This bit controls how the OSC8M behaves during standby sleep mode:

  0: The oscillator is disabled in standby sleep mode.

  1: The oscillator is not stopped in standby sleep mode. If OSC8M.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If OSC8M.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

- **Bits 5:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – ENABLE: Oscillator Enable**
  0: The oscillator is disabled.

  1: The oscillator is enabled.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

16.8.10 DFLL48M Control

Name: DFLLCTRL

Offset: 0x24

Reset: 0x0080

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
							QLDIS	CCDIS
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY		LLAW	STABLE	MODE	ENABLE	
Access	R/W	R/W	R	R/W	R/W	R/W	R/W	R
Reset	1	0	0	0	0	0	0	0

- **Bits 15:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 9 – QLDIS: Quick Lock Disable**
  0: Quick Lock is enabled.
  1: Quick Lock is disabled.

- **Bit 8 – CCDIS: Chill Cycle Disable**
  0: Chill Cycle is enabled.
  1: Chill Cycle is disabled.

- **Bit 7 – ONDEMAND: On Demand Control**
  The On Demand operation mode allows an oscillator to be enabled or disabled depending on peripheral clock requests.
  In On Demand operation mode, i.e., if the ONDEMAND bit has been previously written to one, the oscillator will only be running when requested by a peripheral. If there is no peripheral requesting the oscillator's clock source, the oscillator will be in a disabled state.
  If On Demand is disabled the oscillator will always be running when enabled.
  In standby sleep mode, the On Demand operation is still active if the DFLLCTRL.RUNSTDBY bit is one. If DFLLCTRL.RUNSTDBY is zero, the oscillator is disabled.
  0: The oscillator is always on, if enabled.
  1: The oscillator is enabled when a peripheral is requesting the oscillator to be used as a clock source. The oscillator is disabled if no peripheral is requesting the clock source.

- **Bit 6 – RUNSTDBY: Run in Standby**
  This bit controls how the DFLL behaves during standby sleep mode:
  0: The oscillator is disabled in standby sleep mode.
  1: The oscillator is not stopped in standby sleep mode. If DFLLCTRL.ONDEMAND is one, the clock source will be running when a peripheral is requesting the clock. If DFLLCTRL.ONDEMAND is zero, the clock source will always be running in standby sleep mode.

- **Bit 5 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 4 – LLAW: Lose Lock After Wake**
  0: Locks will not be lost after waking up from sleep modes if the DFLL clock has been stopped.
  1: Locks will be lost after waking up from sleep modes if the DFLL clock has been stopped.

- **Bit 3 – STABLE: Stable DFLL Frequency**
  0: FINE calibration tracks changes in output frequency.
  1: FINE calibration register value will be fixed after a fine lock.

- **Bit 2 – MODE: Operating Mode Selection**
  0: The DFLL operates in open-loop operation.
  1: The DFLL operates in closed-loop operation.

- **Bit 1 – ENABLE: DFLL Enable**
  0: The DFLL oscillator is disabled.
  1: The DFLL oscillator is enabled.
  Due to synchronization, there is delay from updating the register until the peripheral is enabled/disabled. The value written to DFLLCTRL.ENABLE will read back immediately after written.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

16.8.11 DFLL48M Value

Name: DFLLVAL

Offset: 0x28

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	DIFF[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIFF[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COARSE[5:0]						FINE[9:8]	
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FINE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – DIFF: Multiplication Ratio Difference**
  In closed-loop mode (DFLLCTRL.MODE is written to one), this bit group indicates the difference between the ideal number of DFLL cycles and the counted number of cycles. This value is not updated in open-loop mode, and should be considered invalid in that case.

- **Bits 15:10 – COARSE: Coarse Value**
  Set the value of the Coarse Calibration register. In closed-loop mode, this field is read-only.

- **Bits 9:0 – FINE: Fine Value**
  Set the value of the Fine Calibration register. In closed-loop mode, this field is read-only.

16.8.12 DFLL48M Multiplier

Name: DFLLMUL

Offset: 0x2C

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	CSTEP[5:0]						FSTEP[9:8]	
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FSTEP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MUL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MUL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:26 – CSTEP: Coarse Maximum Step**
  This bit group indicates the maximum step size allowed during coarse adjustment in closed-loop mode. When adjusting to a new frequency, the expected output frequency overshoot depends on this step size.
- **Bits 25:16 – FSTEP: Fine Maximum Step**
  This bit group indicates the maximum step size allowed during fine adjustment in closed-loop mode. When adjusting to a new frequency, the expected output frequency overshoot depends on this step size.
- **Bits 15:0 – MUL: DFLL Multiply Factor**
  This field determines the ratio of the CLK_DFLL output frequency to the CLK_DFLL_REF input frequency. Writing to the MUL bits will cause locks to be lost and the fine calibration value to be reset to its midpoint.

16.8.13 DFLL48M Synchronization

Name: DFLLSYNC

Offset: 0x30

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	READREQ							
Access	W	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – READREQ: Read Request**
  To be able to read the current value of DFLLVAL in closed-loop mode, this bit should be written to one. The updated value is available in DFLLVAL when PCLKSR.DFLLRDY is set.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

16.8.14 3.3V Brown-Out Detector (BOD33) Control

Name: BOD33

Offset: 0x34

Reset: 0x00XX00XX

Property: Synchronized, Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			LEVEL[5:0]					
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	X	X	X	X	X	X

Bit	15	14	13	12	11	10	9	8
	PSEL[3:0]						CEN	MODE
Access	R/W	R/W	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		RUNSTDBY		ACTION[1:0]		HYST	ENABLE	
Access	R	R/W	R	R/W	R/W	R/W	R/W	R
Reset	0	0	0	X	X	X	X	0

- **Bits 31:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 21:16 – LEVEL[5:0]: BOD33 Threshold Level**
  This field sets the triggering voltage threshold for the BOD33. See the “Electrical Characteristics” on page 558 for actual voltage levels. Note that any change to the LEVEL field of the BOD33 register should be done when the BOD33 is disabled in order to avoid spurious resets or interrupts.
  These bits are loaded from Flash User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

- **Bits 15:12 – PSEL[3:0]: Prescaler Select**
  Selects the prescaler divide-by output for the BOD33 sampling mode, as given in Table 16-8. The input clock comes from the OSCULP32K 1kHz output.

Table 16-8. BOD33 Prescaler Select

PSEL[3:0]	Name	Description
0x0	DIV2	Divide clock by 2
0x1	DIV4	Divide clock by 4
0x2	DIV8	Divide clock by 8
0x3	DIV16	Divide clock by 16
0x4	DIV32	Divide clock by 32
0x5	DIV64	Divide clock by 64
0x6	DIV128	Divide clock by 128
0x7	DIV256	Divide clock by 256
0x8	DIV512	Divide clock by 512
0x9	DIV1K	Divide clock by 1024
0xA	DIV2K	Divide clock by 2048
0xB	DIV4K	Divide clock by 4096
0xC	DIV8K	Divide clock by 8192
0xD	DIV16K	Divide clock by 16384
0xE	DIV32K	Divide clock by 32768
0xF	DIV64K	Divide clock by 65536

- **Bits 11:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 9 – CEN: Clock Enable**
  0: The BOD33 sampling clock is either disabled and stopped, or enabled but not yet stable.
  1: The BOD33 sampling clock is either enabled and stable, or disabled but not yet stopped.
  Writing a zero to this bit will stop the BOD33 sampling clock.
  Writing a one to this bit will start the BOD33 sampling clock.
- **Bit 8 – MODE: Operation Mode**
  0: The BOD33 operates in continuous mode.
  1: The BOD33 operates in sampling mode.
- **Bit 7 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.
- **Bit 6 – RUNSTDBY: Run in Standby**
  0: The BOD33 is disabled in standby sleep mode.
  1: The BOD33 is enabled in standby sleep mode.
- **Bit 5 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 4:3 – ACTION: BOD33 Action**
  These bits are used to select the BOD33 action when the supply voltage crosses below the BOD33 threshold, as shown in Table 16-9.

  These bits are loaded from Flash User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

Table 16-9. BOD33 Action

ACTION[1:0]	Name	Description
0x0	NONE	No action
0x1	RESET	The BOD33 generates a reset
0x2	INTERRUPT	The BOD33 generates an interrupt
0x3	-	Reserved

- **Bit 2 – HYST: Hysteresis**
  This bit indicates whether hysteresis is enabled for the BOD33 threshold voltage:

  0: No hysteresis.

  1: Hysteresis enabled.

  This bit is loaded from Flash User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

- **Bit 1 – ENABLE: Enable**
  0: BOD33 is disabled.

  1: BOD33 is enabled.

  This bit is loaded from Flash User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

16.8.15 Voltage References System (VREF) Control

Name: VREF

Offset: 0x40

Reset: 0x0XXX0000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
						CALIB[10:8]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	X	X	X

Bit	23	22	21	20	19	18	17	16
	CALIB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	X	X	X	X	X	X	X	X

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						BGOUTEN	TSEN	
Access	R	R	R	R	R	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:27 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 26:16 – CALIB[10:0]: Bandgap Voltage Generator Calibration**
  These bits are used to calibrate the output level of the bandgap voltage reference. These bits are loaded from Flash Calibration Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.
- **Bits 15:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 2 – BGOUTEN: Bandgap Output Enable**
  0: The bandgap output is not available as an ADC input channel.
  1: The bandgap output is routed to an ADC input channel.
- **Bit 1 – TSEN: Temperature Sensor Enable**
  0: Temperature sensor is disabled.

1: Temperature sensor is enabled and routed to an ADC input channel.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

17. WDT – Watchdog Timer

17.1 Overview

The Watchdog Timer (WDT) is a system function for monitoring correct program operation. It makes it possible to recover from error situations such as runaway or deadlocked code. The WDT is configured to a predefined time-out period, and is constantly running when enabled. If the WDT is not cleared within the time-out period, it will issue a system reset. An early-warning interrupt is available to indicate an upcoming watchdog time-out condition.

The window mode makes it possible to define a time slot (or window) inside the total time-out period during which the WDT must be cleared. If the WDT is cleared outside this window, either too early or too late, a system reset will be issued. Compared to the normal mode, this can also catch situations where a code error causes the WDT to be cleared frequently.

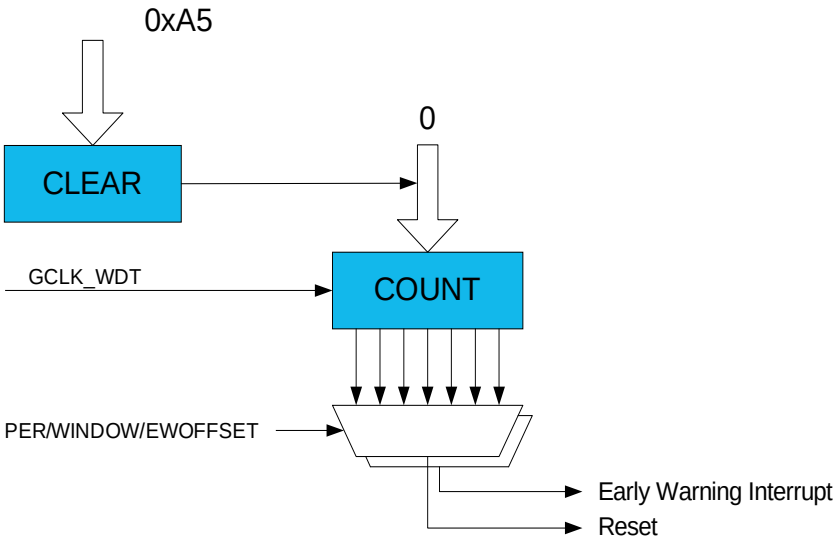
When enabled, the WDT will run in active mode and all sleep modes. It is asynchronous and runs from a CPU-independent clock source.The WDT will continue operation and issue a system reset or interrupt even if the main clocks fail.

17.2 Features

- Issues a system reset if the Watchdog Timer is not cleared before its time-out period
- Early Warning interrupt generation
- Asynchronous operation from dedicated oscillator
- Two types of operation:
  - Normal mode
  - Window mode
- Selectable time-out periods, from 8 cycles to 16,000 cycles in normal mode or 16 cycles to 32,000 cycles in window mode
- Always-on capability

17.3 Block Diagram

Figure 17-1. WDT Block Diagram



17.4 Signal Description

Not applicable.

17.5 Product Dependencies

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

17.5.1 I/O Lines

Not applicable.

17.5.2 Power Management

The WDT can continue to operate in any sleep mode where the selected source clock is running. The WDT interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to “PM – Power Manager” on page 100 for details on the different sleep modes.

17.5.3 Clocks

The WDT bus clock (CLK_WDT_APB) is enabled by default, and can be enabled and disabled in the Power Manager. Refer to “PM – Power Manager” on page 100 for details.

A generic clock (GCLK_WDT) is required to clock the WDT. This clock must be configured and enabled in the Generic Clock Controller before using the WDT. Refer to “GCLK – Generic Clock Controller” on page 78 for details.

This generic clock is asynchronous to the user interface clock (CLK_WDT_APB). Due to this asynchronicity, accessing certain registers will require synchronization between the clock domains. Refer to “Synchronization” on page 178 for further details.

GCLK_WDT is intended to be sourced from the clock of the internal ultra-low-power (ULP) oscillator. Due to the ultra-low-power design, the oscillator is not very accurate, and so the exact time-out period may vary from device to device. This variation must be kept in mind when designing software that uses the WDT to ensure that the time-out periods used are valid for all devices. For more information on ULP oscillator accuracy, consult the “Ultra Low Power Internal 32kHz RC Oscillator (OSCULP32K) Characteristics” on page 579.

GCLK_WDT can also be clocked from other sources if a more accurate clock is needed, but at the cost of higher power consumption.

17.5.4 DMA

Not applicable.

17.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the WDT interrupts requires the interrupt controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

17.5.6 Events

Not applicable.

17.5.7 Debug Operation

When the CPU is halted in debug mode, the WDT will halt normal operation. If the WDT 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. The WDT can be forced to halt operation during debugging.

17.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the peripheral access controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

17.5.9 Analog Connections

Not applicable.

17.6 Functional Description

17.6.1 Principle of Operation

The Watchdog Timer (WDT) is a system for monitoring correct program operation, making it possible to recover from error situations such as runaway code by issuing a reset. When enabled, the WDT is a constantly running timer that is configured to a predefined time-out period. Before the end of the time-out period, the WDT should be reconfigured.

The WDT has two modes of operation, normal and window. Additionally, the user can enable Early Warning interrupt generation in each of the modes. The description for each of the basic modes is given below. The settings in the Control register (CTRL) and the Interrupt Enable register (INTENCLR/SET) determine the mode of operation, as illustrated in Table 17-1.

Table 17-1. WDT Operating Modes

ENABLE	WEN	Interrupt Enable	Mode
0	x	x	Stopped
1	0	0	Normal
1	0	1	Normal with Early Warning interrupt
1	1	0	Window
1	1	1	Window with Early Warning interrupt

17.6.2 Basic Operation

17.6.2.1 Initialization

The following registers are enable-protected:

- Control register (CTRL), except the Enable bit (CTRL.ENABLE)
- Configuration register (CONFIG)
- Early Warning Interrupt Control register (EWCTRL)

Any writes to these bits or registers when the WDT is enabled or is being enabled (CTRL.ENABLE is one) will be discarded. Writes to these registers while the WDT is being disabled will be completed after the disabling is complete.

Enable-protection is denoted by the Enable-Protected property in the register description.

Initialization of the WDT can be done only while the WDT is disabled. The WDT is configured by defining the required Time-Out Period bits in the Configuration register (CONFIG.PER). If window-mode operation is required, the Window

Enable bit in the Control register (CTRL.WEN) must be written to one and the Window Period bits in the Configuration register (CONFIG.WINDOW) must be defined.

17.6.2.2 Configurable Reset Values

On a power-on reset, some registers will be loaded with initial values from the NVM User Row. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

This encompasses the following bits and bit groups:

- Enable bit in the Control register (CTRL.ENABLE)
- Always-On bit in the Control register (CTRL.ALWAYSON)
- Watchdog Timer Windows Mode Enable bit in the Control register (CTRL.WEN)
- Watchdog Timer Windows Mode Time-Out Period bits in the Configuration register (CONFIG.WINDOW)
- Time-Out Period in the Configuration register (CONFIG.PER)
- Early Warning Interrupt Time Offset bits in the Early Warning Interrupt Control register (EWCTRL.EWOFFSET)

For more information about fuse locations, see “Non-Volatile Memory (NVM) User Row Mapping” on page 21.

17.6.2.3 Enabling and Disabling

The WDT is enabled by writing a one to the Enable bit in the Control register (CTRL.ENABLE). The WDT is disabled by writing a zero to CTRL.ENABLE.

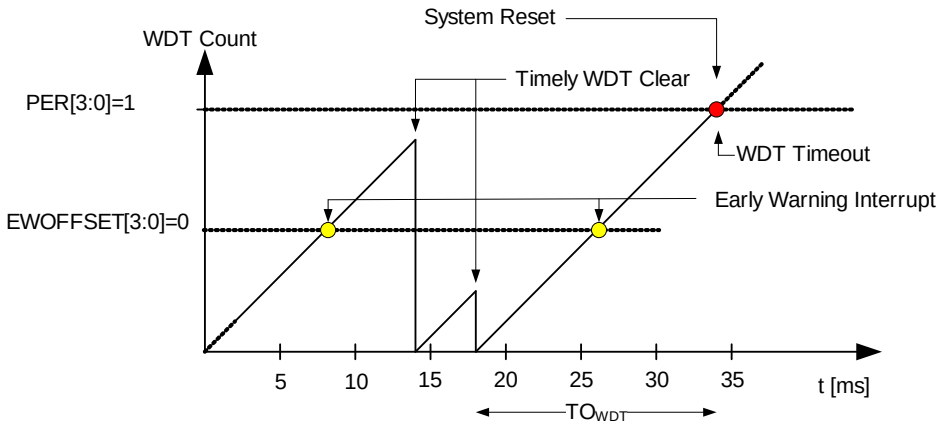
The WDT can be disabled only while the Always-On bit in the Control register (CTRL.ALWAYSON) is zero.

17.6.2.4 Normal Mode

In normal-mode operation, the length of a time-out period is configured in CONFIG.PER. The WDT is enabled by writing a one to the Enable bit in the Control register (CTRL.ENABLE). Once enabled, if the WDT is not cleared from the application code before the time-out occurs, the WDT will issue a system reset. There are 12 possible WDT time-out (TO_{WDT}) periods, selectable from 8ms to 16s, and the WDT can be cleared at any time during the time-out period. A new WDT time-out period will be started each time the WDT is cleared by writing 0xA5 to the Clear register (CLEAR). Writing any value other than 0xA5 to CLEAR will issue an immediate system reset.

By default, WDT issues a system reset upon a time-out, and the early warning interrupt is disabled. If an early warning interrupt is required, the Early Warning Interrupt Enable bit in the Interrupt Enable register (INTENSET.EW) must be enabled. Writing a one to the Early Warning Interrupt bit in the Interrupt Enable Set register (INTENSET.EW) enables the interrupt, and writing a one to the Early Warning Interrupt bit in the Interrupt Enable Clear register (INTENCLR.EW) disables the interrupt. If the Early Warning Interrupt is enabled, an interrupt is generated prior to a watchdog time-out condition. In normal mode, the Early Warning Offset bits in the Early Warning Interrupt Control register (EWCTRL.EWOFFSET) define the time where the early warning interrupt occurs. The normal-mode operation is illustrated in Figure 17-2.

Figure 17-2. Normal-Mode Operation



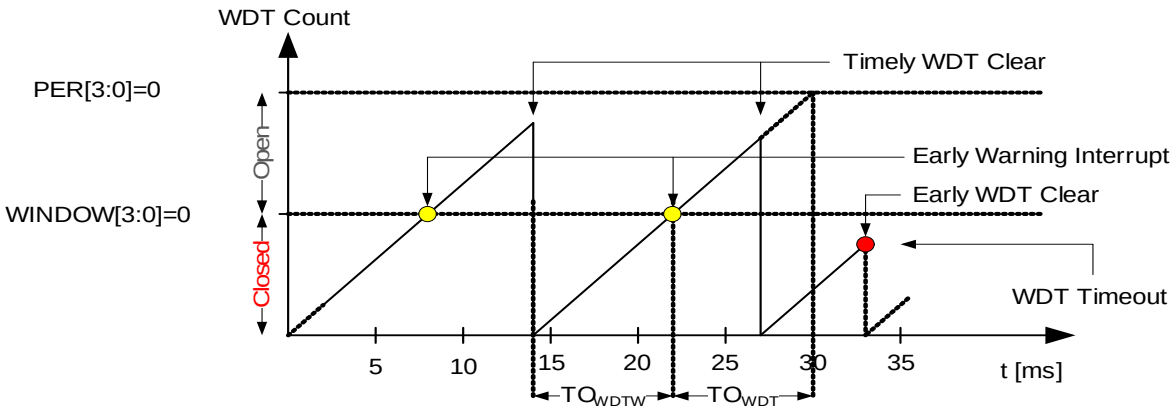
17.6.2.5 Window Mode

In window-mode operation, the WDT uses two different time-out periods, a closed window time-out period (TO_{WDTW}) and the normal, or open, time-out period (TO_{WDT}). The closed window time-out period defines a duration from 8ms to 16s where the WDT cannot be reset. If the WDT is cleared during this period, the WDT will issue a system reset. The normal WDT time-out period, which is also from 8ms to 16s, defines the duration of the open period during which the WDT can be cleared. The open period will always follow the closed period, and so the total duration of the time-out period is the sum of the closed window and the open window time-out periods. The closed window is defined by the Window Period bits in the Configuration register (CONFIG.WINDOW), and the open window is defined by the Period bits in the Configuration register (CONFIG.PER).

By default, the WDT issues a system reset upon a time-out and the Early Warning interrupt is disabled. If an Early Warning interrupt is required, INTENCLR/SET.EW must be set. Writing a one to INTENSET.EW enables the interrupt, and writing a one to INTENCLR.EW disables the interrupt. If the Early Warning interrupt is enabled in window mode, the interrupt is generated at the start of the open window period.

The window mode operation is illustrated in Figure 17-3.

Figure 17-3. Window-Mode Operation



17.6.3 Additional Features

17.6.3.1 Always-On Mode

The always-on mode is enabled by writing a one to the Always-On bit in the Control register (CTRL.ALWAYSON). When the always-on mode is enabled, the WDT runs continuously, regardless of the state of CTRL.ENABLE. Once written, the Always-On bit can only be cleared by a power-on reset. The Configuration (CONFIG) and Early Warning Control (EWCTRL) registers are read-only registers while the CTRL.ALWAYSON bit is set. Thus, the time period configuration bits (CONFIG.PER, CONFIG.WINDOW, EWCTRL.EWOFFSET) of the WDT cannot be changed.

Enabling or disabling window-mode operation by writing the Window Enable bit (CTRL.WEN) is allowed while in the always-on mode, but note that CONFIG.PER cannot be changed.

The Interrupt Clear and Interrupt Set registers are accessible in the always-on mode. The Early Warning interrupt can still be enabled or disabled while in the always-on mode, but note that EWCTRL.EWOFFSET cannot be changed.

Table 17-2 shows the operation of the WDT when CTRL.ALWAYSON is set.

Table 17-2. WDT Operating Modes With Always-On

WEN	Interrupt enable	Mode
0	0	Always-on and normal mode
0	1	Always-on and normal mode with Early Warning interrupt
1	0	Always-on and window mode
1	1	Always-on and window mode with Early Warning interrupt

17.6.4 Interrupts

The WDT has the following interrupt sources:

- Early Warning

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the WDT is reset. See INTFLAG for details on how to clear interrupt flags.

The WDT has one common interrupt request line for all the interrupt sources. The user must read INTFLAG to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

The Early Warning interrupt behaves differently in normal mode and in window mode. In normal mode, the Early Warning interrupt generation is defined by the Early Warning Offset in the Early Warning Control register (EWCTRL.EWOFFSET). The Early Warning Offset bits define the number of GCLK_WDT clocks before the interrupt is generated, relative to the start of the watchdog time-out period. For example, if the WDT is operating in normal mode with CONFIG.PER = 0x2 and EWCTRL.EWOFFSET = 0x1, the Early Warning interrupt is generated 16 GCLK_WDT clock cycles from the start of the watchdog time-out period, and the watchdog time-out system reset is generated 32 GCLK_WDT clock cycles from the start of the watchdog time-out period. The user must take caution when programming the Early Warning Offset bits. If these bits define an Early Warning interrupt generation time greater than the watchdog time-out period, the watchdog time-out system reset is generated prior to the Early Warning interrupt. Thus, the Early Warning interrupt will never be generated.

In window mode, the Early Warning interrupt is generated at the start of the open window period. In a typical application where the system is in sleep mode, it can use this interrupt to wake up and clear the Watchdog Timer, after which the system can perform other tasks or return to sleep mode.

17.6.5 Synchronization

Due to the asynchronicity between CLK_WDT_APB and GCLK_WDT some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register(STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The synchronization Ready interrupt can be used to signal when sync is complete. This can be accessed via the Synchronization Ready Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.SYNCRDY).

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following registers need synchronization when written:

- Control register (CTRL)
- Clear register (CLEAR)

Write-synchronization is denoted by the Write-Synchronized property in the register description.

17.7 Register Summary

Register summary

Offset	Name	Bit Pos.								
0x0	CTRL	7:0	ALWAYSON					WEN	ENABLE	
0x1	CONFIG	7:0	WINDOW[3:0]				PER[3:0]			
0x2	EWCTRL	7:0					EWOFFSET[3:0]			
0x3	Reserved									
0x4	INTENCLR	7:0								EW
0x5	INTENSET	7:0								EW
0x6	INTFLAG	7:0								EW
0x7	STATUS	7:0	SYNCBUSY							
0x8	CLEAR	7:0	CLEAR[7:0]							

17.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 174 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 178 for details.

Some registers are enable-protected, meaning they can be written only when the WDT is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

17.8.1 Control

Name: CTRL

Offset: 0x0

Reset: N/A - Loaded from NVM User Row at startup

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
	ALWAYSON					WEN	ENABLE	
Access	R/W1	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	X	0	0	0	0	X	X	0

- **Bit 7 – ALWAYSON: Always-On**
  This bit allows the WDT to run continuously. After being written to one, this bit cannot be written to zero, and the WDT will remain enabled until a power-on reset is received. When this bit is one, the Control register (CTRL), the Configuration register (CONFIG) and the Early Warning Control register (EWCTRL) will be read-only, and any writes to these registers are not allowed. Writing a zero to this bit has no effect.
  0: The WDT is enabled and disabled through the ENABLE bit.
  1: The WDT is enabled and can only be disabled by a power-on reset (POR).
  This bit is not enable-protected.
  These bits are loaded from NVM User Row at startup. Refer to "Non-Volatile Memory (NVM) User Row Mapping" on page 21 for more details.

- **Bits 6:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – WEN: Watchdog Timer Window Mode Enable**
  This bit enables window mode. Can be written only while CTRL.ALWAYSON is zero. The initial value of this bit is loaded from Flash Calibration.
  0: Window mode is disabled (normal operation).
  1: Window mode is enabled.
  This bit is loaded from NVM User Row at startup. Refer to "Non-Volatile Memory (NVM) User Row Mapping" on page 21 for more details.

- **Bit 1 – ENABLE: Enable**
  This bit enables or disables the WDT. Can only be written while CTRL.ALWAYSON is zero.
  0: The WDT is disabled.
  1: The WDT is enabled.
  Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.
  This bit is not enable-protected.
  This bit is loaded from NVM User Row at startup. Refer to "Non-Volatile Memory (NVM) User Row Mapping" on page 21 for more details.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

17.8.2 Configuration

Name: CONFIG

Offset: 0x1

Reset: N/A - Loaded from NVM User Row at startup

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
	WINDOW[3:0]				PER[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	X	X	X	X	X	X	X	X

- **Bits 7:4 – WINDOW[3:0]: Window Mode Time-Out Period**
  In window mode, these bits determine the watchdog closed window period as a number of oscillator cycles. The closed window periods are defined in Table 17-3.
  These bits are loaded from NVM User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

Table 17-3. Window Mode Time-Out Period

Value	Description
0x0	8 clock cycles
0x1	16 clock cycles
0x2	32 clock cycles
0x3	64 clock cycles
0x4	128 clock cycles
0x5	256 clocks cycles
0x6	512 clocks cycles
0x7	1024 clock cycles
0x8	2048 clock cycles
0x9	4096 clock cycles
0xA	8192 clock cycles
0xB	16384 clock cycles
0xC-0xF	Reserved

- **Bits 3:0 – PER[3:0]: Time-Out Period**
  These bits determine the watchdog time-out period as a number of GCLK_WDT clock cycles. In window mode operation, these bits define the open window period. The different typical time-out periods are found in Table 17-4.
  These bits are loaded from NVM User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

Table 17-4. Time-Out Period

Value	Description
0x0	8 clock cycles
0x1	16 clock cycles
0x2	32 clock cycles
0x3	64 clock cycles
0x4	128 clock cycles
0x5	256 clocks cycles
0x6	512 clocks cycles
0x7	1024 clock cycles
0x8	2048 clock cycles
0x9	4096 clock cycles
0xA	8192 clock cycles
0xB	16384 clock cycles
0xC-0xF	Reserved

17.8.3 Early Warning Interrupt Control

Name: EWCTRL

Offset: 0x2

Reset: N/A - Loaded from NVM User Row at startup

Property: Write-Protected, Enable-Protected

Bit	7	6	5	4	3	2	1	0
					EWOFFSET[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	X	X	X	X

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:0 – EWOFFSET[3:0]: Early Warning Interrupt Time Offset**
  These bits determine the number of GCLK_WDT clocks in the offset from the start of the watchdog time-out period to when the Early Warning interrupt is generated. The Early Warning Offset is defined in Table 17-5. These bits are loaded from NVM User Row at startup. Refer to “Non-Volatile Memory (NVM) User Row Mapping” on page 21 for more details.

Table 17-5. Early Warning Interrupt Time Offset

Value	Description
0x0	8 clock cycles
0x1	16 clock cycles
0x2	32 clock cycles
0x3	64 clock cycles
0x4	128 clock cycles
0x5	256 clocks cycles
0x6	512 clocks cycles
0x7	1024 clock cycles
0x8	2048 clock cycles
0x9	4096 clock cycles
0xA	8192 clock cycles
0xB	16384 clock cycles
0xC-0xF	Reserved

17.8.4 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x4

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								EW
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – EW: Early Warning Interrupt Enable**
  0: The Early Warning interrupt is disabled.
  1: The Early Warning interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the Early Warning interrupt.

17.8.5 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x5

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								EW
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – EW: Early Warning Interrupt Enable**
  0: The Early Warning interrupt is disabled.
  1: The Early Warning interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the Early Warning interrupt.

17.8.6 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x6

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
								EW
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**

  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – EW: Early Warning**

  This flag is set when an Early Warning interrupt occurs, as defined by the EWOFFSET bit group in EWCTRL.

  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Early Warning interrupt flag.

17.8.7 Status

Name: STATUS

Offset: 0x7

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

17.8.8 Clear

Name: CLEAR

Offset: **Offset:** 0x8

Reset: 0x00

Property: Write-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
	CLEAR[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:0 – CLEAR: Watchdog Clear**
  Writing 0xA5 to this register will clear the Watchdog Timer and the watchdog time-out period is restarted. Writing any other value will issue an immediate system reset.

17.9 Asynchronous Watchdog Clock Characterization

The source intended for the asynchronous watchdog clock (OSCULP32K) has a variance of +/- 15%. In a typical application with GCLK_WDT = 1kHz or GCLK_WDT = 32kHz, the time period is illustrated in Table 17-6.

Table 17-6. Typical Time-Out Period

	Typical Time-Out Periods					
	GCLK_WDT = 1kHz			GCLK_WDT = 32kHz		
	Min	Typ	Max	Min	Typ	Max
8 clock cycles	6.64ms	7.81ms	8.98ms	0.20ms	0.24ms	0.28ms
16 clock cycles	13.28ms	15.62ms	17.97ms	0.41ms	0.48ms	0.56ms
32 clock cycles	26.56ms	31.25ms	39.94ms	0.83ms	0.97ms	1.12ms
64 clock cycles	53.12ms	62.50ms	71.87ms	1.66ms	1.95ms	2.24ms
128 clock cycles	0.10s	0.12s	0.14s	3.32ms	3.90ms	4.49ms
256 clocks cycles	0.21s	0.25s	0.28s	6.64ms	7.81ms	8.98ms
512 clocks cycles	0.42s	0.50s	0.57s	13.28ms	15.62ms	17.96ms
1024 clock cycles	0.85s	1.00s	1.15s	26.56ms	31.25ms	35.93ms
2048 clock cycles	1.70s	2.00s	2.30s	53.12ms	62.50ms	71.87ms
4096 clock cycles	3.40s	4.00s	4.60s	0.10s	0.12s	0.14s
8192 clock cycles	6.80s	8.00s	9.20s	0.21s	0.25s	0.28s
16384 clock cycles	13.60s	16.00s	18.40s	0.42s	0.50s	0.57s

18. RTC – Real-Time Counter

18.1 Overview

The Real-Time Counter (RTC) is a 32-bit counter with a 10-bit programmable prescaler that typically runs continuously to keep track of time. The RTC can wake up the device from sleep modes using the alarm/compare wake up, periodic wake up or overflow wake up mechanisms.

The RTC is typically clocked by the 1.024kHz output from the 32.768kHz High-Accuracy Internal Crystal Oscillator(OSC32K) and this is the configuration optimized for the lowest power consumption. The faster 32.768kHz output can be selected if the RTC needs a resolution higher than 1ms. The RTC can also be clocked from other sources, selectable through the Generic Clock module (GCLK).

The RTC can generate periodic peripheral events from outputs of the prescaler, as well as alarm/compare 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 an alarm/compare match. This allows periodic interrupts and peripheral events at very long and accurate intervals.

The 10-bit programmable prescaler can scale down the clock source, and so a wide range of resolutions and time-out periods can be configured. With a 32.768kHz clock source, the minimum counter tick interval is 30.5µs, and time-out periods can range up to 36 hours. With the counter tick interval configured to 1s, the maximum time-out period is more than 136 years.

18.2 Features

- 32-bit counter with 10-bit prescaler
- Multiple clock sources
- 32-bit or 16-bit Counter mode
  - One 32-bit or two 16-bit compare values
- Clock/Calendar mode
  - Time in seconds, minutes and hours (12/24)
  - Date in day of month, month and year
  - Leap year correction
- Digital prescaler correction/tuning for increased accuracy
- Overflow, alarm/compare match and prescaler interrupts and events
  - Optional clear on alarm/compare match

18.3 Block Diagram

Figure 18-1. RTC Block Diagram (Mode 0 — 32-Bit Counter)

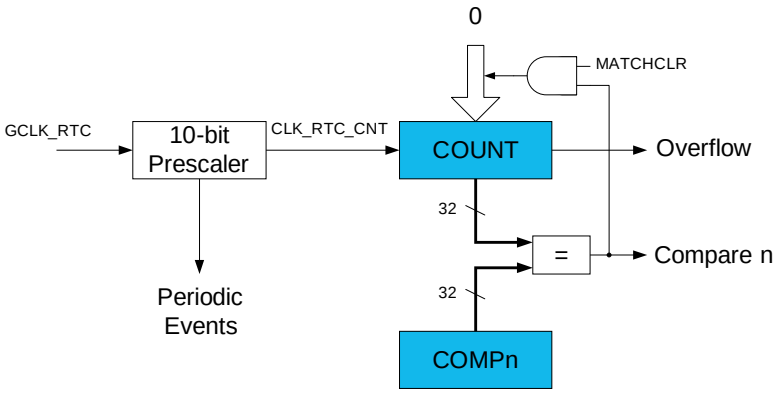


Figure 18-2. RTC Block Diagram (Mode 1 — 16-Bit Counter)

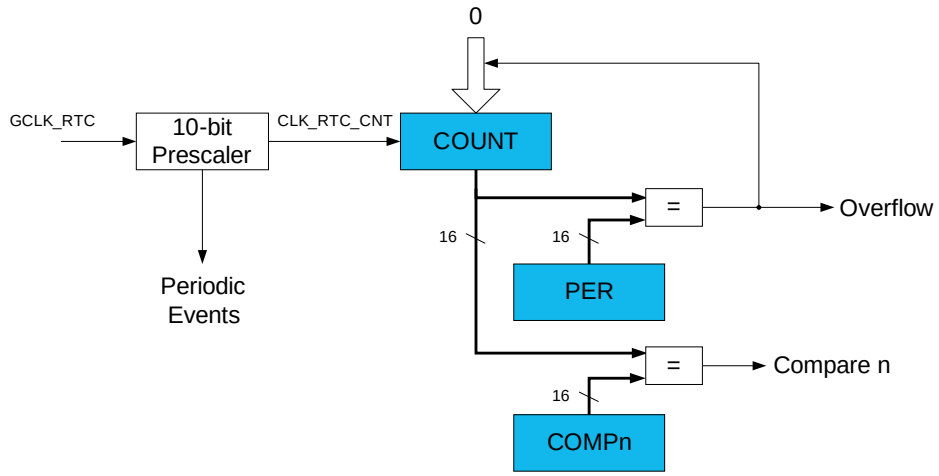
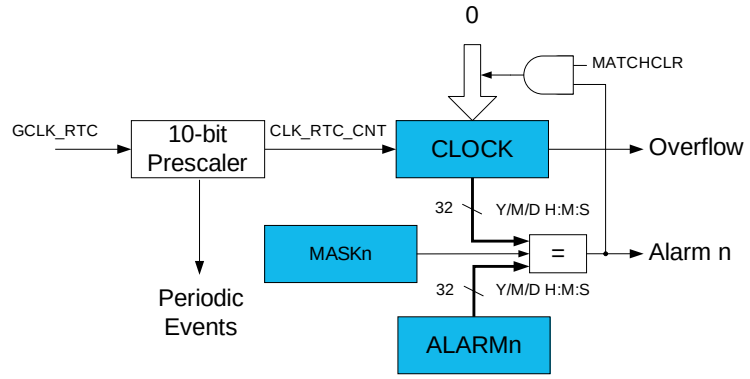


Figure 18-3. RTC Block Diagram (Mode 2 — Clock/Calendar)



18.4 Signal Description

Not applicable.

18.5 Product Dependencies

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

18.5.1 I/O Lines

Not applicable.

18.5.2 Power Management

The RTC can continue to operate in any sleep mode. The RTC interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to “PM – Power Manager” on page 100 for details on the different sleep modes.

The RTC will be reset only at power-on (POR) or by writing a one to the Software Reset bit in the Control register (CTRL.SWRST).

18.5.3 Clocks

The RTC bus clock (CLK_RTC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_RTC_APB can be found in the Peripheral Clock Masking section in the “PM – Power Manager” on page 100.

A generic clock (GCLK_RTC) is required to clock the RTC. This clock must be configured and enabled in the Generic Clock Controller before using the RTC. Refer to “GCLK – Generic Clock Controller” on page 78 for details.

This generic clock is asynchronous to the user interface clock (CLK_RTC_APB). Due to this asynchronicity, accessing certain registers will require synchronization between the clock domains. Refer to “Synchronization” on page 197 for further details.

18.5.4 DMA

Not applicable.

18.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the RTC interrupts requires the interrupt controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

18.5.6 Events

To use the RTC event functionality, the corresponding events need to be configured in the event system. Refer to “EVSYS – Event System” on page 301 for details.

18.5.7 Debug Operation

When the CPU is halted in debug mode the RTC will halt normal operation. The RTC can be forced to continue operation during debugging. Refer to the DBGCTRL register for details.

18.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the peripheral access controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Read Request register (READREQ)
- Status register (STATUS)
- Debug register (DBGCTRL)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

18.5.9 Analog Connections

A 32.768kHz crystal can be connected to the TOSC1 and TOSC2 pins, along with any required load capacitors. For details on recommended crystal characteristics and load capacitors, refer to “Electrical Characteristics” on page 558 for details.

18.6 Functional Description

18.6.1 Principle of Operation

The RTC keeps track of time in the system and enables periodic events, as well as interrupts and events at a specified time. The RTC consists of a 10-bit prescaler that feeds a 32-bit counter. The actual format of the 32-bit counter depends on the RTC operating mode.

18.6.2 Basic Operation

18.6.2.1 Initialization

The following bits are enable-protected, meaning that they can only be written when the RTC is disabled (CTRL.ENABLE is zero):

- Operating Mode bits in the Control register (CTRL.MODE)
- Prescaler bits in the Control register (CTRL.PRESCALER)
- Clear on Match bit in the Control register (CTRL.MATCHCLR)
- Clock Representation bit in the Control register (CTRL.CLKREP)

The following register is enable-protected:

- Event Control register (EVCTRL)

Any writes to these bits or registers when the RTC is enabled or being disabled (CTRL.ENABLE is one) will be discarded. Writes to these bits or registers while the RTC is being disabled will be completed after the disabling is complete.

Enable-protection is denoted by the Enable-Protection property in the register description.

Before the RTC is enabled, it must be configured, as outlined by the following steps:

- RTC operation mode must be selected by writing the Operating Mode bit group in the Control register (CTRL.MODE)
- Clock representation must be selected by writing the Clock Representation bit in the Control register (CTRL.CLKREP)
- Prescaler value must be selected by writing the Prescaler bit group in the Control register (CTRL.PRESCALER)

The RTC prescaler divides down the source clock for the RTC counter. The frequency of the RTC clock (CLK_RTC_CNT) is given by the following formula:

$$f_{CLK\_RTC\_CNT} = \frac{f_{GCLK\_RTC}}{2^{PRESCALER}}$$

The frequency of the generic clock, GCLK_RTC, is given by f_{GCLK_RTC}, and $f_{CLK\_RTC\_CNT}$ is the frequency of the internal prescaled RTC clock, CLK_RTC_CNT.

Note that in the Clock/Calendar mode, the prescaler must be configured to provide a 1Hz clock to the counter for correct operation.

18.6.2.2 Enabling, Disabling and Resetting

The RTC is enabled by writing a one to the Enable bit in the Control register (CTRL.ENABLE). The RTC is disabled by writing a zero to CTRL.ENABLE.

The RTC should be disabled before resetting it.

The RTC is reset by writing a one to the Software Reset bit in the Control register (CTRL.SWRST). All registers in the RTC, except DBGCTRL, will be reset to their initial state, and the RTC will be disabled.

Refer to the CTRL register for details.

18.6.3 Operating Modes

The RTC counter supports three RTC operating modes: 32-bit Counter, 16-bit Counter and Clock/Calendar. The operating mode is selected by the Operating Mode bit group in the Control register (CTRL.MODE).

18.6.3.1 32-Bit Counter (Mode 0)

When the RTC Operating Mode bits in the Control register (CTRL.MODE) are zero, the counter operates in 32-bit Counter mode. The block diagram of this mode is shown in Figure 18-1. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. The counter will increment until it reaches the top value of

0xFFFFFFFF, and then wrap to 0x00000000. This sets the Overflow Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF).

The RTC counter value can be read from or written to the Counter Value register (COUNT) in 32-bit format.

The counter value is continuously compared with the 32-bit Compare register (COMP0). When a compare match occurs, the Compare 0Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMP0) is set on the next 0-to-1 transition of CLK_RTC_CNT.

If the Clear on Match bit in the Control register (CTRL.MATCHCLR) is one, the counter is cleared on the next counter cycle when a compare match with COMP0 occurs. This allows the RTC to generate periodic interrupts or events with longer periods than are possible with the prescaler events. Note that when CTRL.MATCHCLR is one, INTFLAG.CMP0 and INTFLAG.OVF will both be set simultaneously on a compare match with COMP0.

18.6.3.2 16-Bit Counter (Mode 1)

When CTRL.MODE is one, the counter operates in 16-bit Counter mode as shown in Figure 18-2. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. In 16-bit Counter mode, the 16-bit Period register (PER) holds the maximum value of the counter. The counter will increment until it reaches the PER value, and then wrap to 0x0000. This sets the Overflow Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF).

The RTC counter value can be read from or written to the Counter Value register (COUNT) in 16-bit format.

The counter value is continuously compared with the 16-bit Compare registers (COMPn, n=0–1). When a compare match occurs, the Compare n Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn, n=0–1) is set on the next 0-to-1 transition of CLK_RTC_CNT.

18.6.3.3 Clock/Calendar (Mode 2)

When CTRL.MODE is two, the counter operates in Clock/Calendar mode, as shown in Figure 18-3. When the RTC is enabled, the counter will increment on every 0-to-1 transition of CLK_RTC_CNT. The selected clock source and RTC prescaler must be configured to provide a 1Hz clock to the counter for correct operation in this mode.

The time and date can be read from or written to the Clock Value register (CLOCK) in a 32-bit time/date format. Time is represented as:

- Seconds
- Minutes
- Hours

Hours can be represented in either 12- or 24-hour format, selected by the Clock Representation bit in the Control register (CTRL.CLKREP). This bit can be changed only while the RTC is disabled.

Date is represented as:

- Day as the numeric day of the month (starting at 1)
- Month as the numeric month of the year (1 = January, 2 = February, etc.)
- Year as a value counting the offset from a reference value that must be defined in software

The date is automatically adjusted for leap years, assuming every year divisible by 4 is a leap year. Therefore, the reference value must be a leap year, e.g. 2000. The RTC will increment until it reaches the top value of 23:59:59 December 31 of year 63, and then wrap to 00:00:00 January 1 of year 0. This will set the Overflow Interrupt flag in the Interrupt Flag Status and Clear registers (INTFLAG.OVF).

The clock value is continuously compared with the 32-bit Alarm register (ALARM0). When an alarm match occurs, the Alarm 0 Interrupt flag in the Interrupt Flag Status and Clear registers (INTFLAG.ALARMn0) is set on the next 0-to-1 transition of CLK_RTC_CNT.

A valid alarm match depends on the setting of the Alarm Mask Selection bits in the Alarm 0 Mask register (MASK0.SEL). These bits determine which time/date fields of the clock and alarm values are valid for comparison and which are ignored.

If the Clear on Match bit in the Control register (CTRL.MATCHCLR) is one, the counter is cleared on the next counter cycle when an alarm match with ALARM0 occurs. This allows the RTC to generate periodic interrupts or events with longer periods than are possible with the prescaler events (see “Periodic Events” on page 196). Note that when CTRL.MATCHCLR is one, INTFLAG.ALARM0 and INTFLAG.OVF will both be set simultaneously on an alarm match with ALARM0.

18.6.4 Additional Features

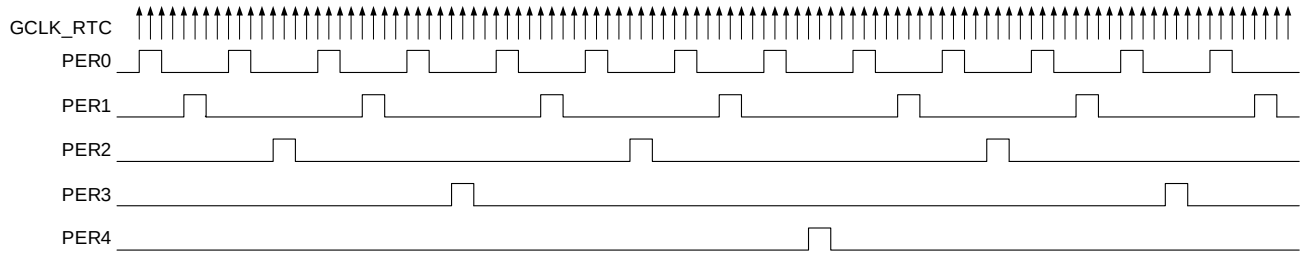
18.6.4.1 Periodic Events

The RTC prescaler can generate events at periodic intervals, allowing flexible system tick creation. Any of the upper eight bits of the prescaler (bits 2 to 9) can be the source of an event. When one of the Periodic Event Output bits in the Event Control register (EVCTRL.PEREOm) is one, an event is generated on the 0-to1 transition of the related bit in the prescaler, resulting in a periodic event frequency of:

$$f_{PERIODIC} = \frac{f_{GCLK\_RTC}}{2^{n+3}}$$

f_{GCLK_RTC} is the frequency of the internal prescaler clock, GCLK_RTC, and *n* is the position of the EVCTRL.PERnEO bit. For example, PER0 will generate an event every 8 GCLK_RTC cycles, PER1 every 16 cycles, etc. This is shown in Figure 18-4. Periodic events are independent of the prescaler setting used by the RTC counter, except if CTRL.PRESCALER is zero. Then, no periodic events will be generated.

Figure 18-4. Example Periodic Events



18.6.4.2 Frequency Correction

The RTC Frequency Correction module employs periodic counter corrections to compensate for a too-slow or too-fast oscillator. Frequency correction requires that CTRL.PRESCALER is greater than 1.

The digital correction circuit adds or subtracts cycles from the RTC prescaler to adjust the frequency in approximately 1PPM steps. Digital correction is achieved by adding or skipping a single count in the prescaler once every 1024 GCLK_RTC cycles. The Value bit group in the Frequency Correction register (FREQCORR.VALUE) determines the number of times the adjustment is applied over 976 of these periods. The resulting correction is as follows:

$$\text{Correction in PPM} = \frac{\text{FREQCORR.VALUE}}{1024 \cdot 976} \cdot 10^6 \text{PPM}$$

This results in a resolution of 1.0006PPM.

The Sign bit in the Frequency Correction register (FREQCORR.SIGN) determines the direction of the correction. A positive value will speed up the frequency, and a negative value will slow down the frequency.

Digital correction also affects the generation of the periodic events from the prescaler. When the correction is applied at the end of the correction cycle period, the interval between the previous periodic event and the next occurrence may also be shortened or lengthened depending on the correction value.

18.6.5 DMA Operation

Not applicable.

18.6.6 Interrupts

The RTC has the following interrupt sources:

- Overflow
- Compare m
- Alarm m
- Synchronization Ready

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the RTC is reset. See INTFLAG for details on how to clear interrupt flags. The RTC has one common interrupt request line for all the interrupt sources. The user must read INTFLAG to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

18.6.7 Events

The RTC can generate the following output events, which are generated in the same way as the corresponding interrupts:

- Overflow (OVF)
- Period n (PERn)
- Compare n (CMPn)
- Alarm n (ALARMn)

Output events must be enabled to be generated. Writing a one to an Event Output bit in the Event Control register (EVCTRL.xxEO) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. Refer to “EVSYS – Event System” on page 301 for details.

18.6.8 Sleep Mode Operation

The RTC will continue to operate in any sleep mode where the source clock is active. The RTC interrupts can be used to wake up the device from a sleep mode, or the RTC events can trigger other operations in the system without exiting the sleep mode.

An interrupt request will be generated after the wake-up if the Interrupt Controller is configured accordingly. Otherwise the CPU will wake up directly, without triggering an interrupt. In this case, the CPU will continue executing from the instruction following the entry into sleep.

The periodic events can also wake up the CPU through the interrupt function of the Event System. In this case, the event must be enabled and connected to an event channel with its interrupt enabled. See “EVSYS – Event System” on page 301 for more information.

18.6.9 Synchronization

Due to the asynchronicity between CLK_RTC_APB and GCLK_RTC some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read

- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register(STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The synchronization Ready interrupt can be used to signal when sync is complete. This can be accessed via the Synchronization Ready Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.SYNCRDY).

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control register (CTRL.SWRST)
- Enable bit in the Control register (CTRL.ENABLE)

The following registers need synchronization when written:

- The Counter Value register (COUNT)
- The Clock Value register (CLOCK)
- The Counter Period register (PER)
- The Compare n Value registers (COMPn)
- The Alarm n Value registers (ALARMn)
- The Frequency Correction register (FREQCORR)
- The Alarm n Mask register (MASKn)

Write-synchronization is denoted by the Write-Synchronization property in the register description.

The following registers need synchronization when read:

- The Counter Value register (COUNT)
- The Clock Value register (CLOCK)

Read-synchronization is denoted by the Read-Synchronization property in the register description.

18.7 Register Summary

The register mapping depends on the Operating Mode bits in the Control register (CTRL.MODE). The register summary is presented for each of the three modes.

Table 18-1. Register Summary - Mode 0 Registers

Offset	Name	Bit Pos.								
0x00	CTRL	7:0	MATCHCLR	CLKREP			MODE[1:0]		ENABLE	SWRST
0x01		15:8					PRESCALER[3:0]			
0x02	READREQ	7:0			ADDR[5:0]					
0x03		15:8	RREQ	RCONT						
0x04	EVCTRL	7:0	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
0x05		15:8	OVFEO							CMPEO0
0x06	INTENCLR	7:0	OVF	SYNCRDY						CMP0
0x07	INTENSET	7:0	OVF	SYNCRDY						CMP0
0x08	INTFLAG	7:0	OVF	SYNCRDY						CMP0
0x09	Reserved									
0x0A	STATUS	7:0	SYNCBUSY							
0x0B	DBGCTRL	7:0								DBGRUN
0x0C	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x0D	Reserved									
0x0E	Reserved									
0x0F	Reserved									
0x10	COUNT	7:0	COUNT[7:0]							
0x11		15:8	COUNT[15:8]							
0x12		23:16	COUNT[23:16]							
0x13		31:24	COUNT[31:24]							
0x14	Reserved									
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	COMP0	7:0	COMP[7:0]							
0x19		15:8	COMP[15:8]							
0x1A		23:16	COMP[23:16]							
0x1B		31:24	COMP[31:24]							

Table 18-2. Register Summary - Mode 1 Registers

Offset	Name	Bit Pos.								
0x00	CTRL	7:0	MATCHCLR	CLKREP			MODE[1:0]		ENABLE	SWRST
0x01		15:8					PRESCALER[3:0]			
0x02	READREQ	7:0			ADDR[5:0]					
0x03		15:8	RREQ	RCONT						
0x04	EVCTRL	7:0	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
0x05		15:8	OVFEO						CMPEO1	CMPEO0
0x06	INTENCLR	7:0	OVF	SYNCRDY					CMP1	CMP0
0x07	INTENSET	7:0	OVF	SYNCRDY					CMP1	CMP0
0x08	INTFLAG	7:0	OVF	SYNCRDY					CMP1	CMP0
0x09	Reserved									
0x0A	STATUS	7:0	SYNCBUSY							
0x0B	DBGCTRL	7:0								DBGRUN
0x0C	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x0D	Reserved									
0x0E	Reserved									
0x0F	Reserved									
0x10	COUNT	7:0	COUNT[7:0]							
0x11		15:8	COUNT[15:8]							
0x12	Reserved									
0x13	Reserved									
0x14	PER	7:0	PER[7:0]							
0x15		15:8	PER[15:8]							
0x16	Reserved									
0x17	Reserved									
0x18	COMP0	7:0	COMP[7:0]							
0x19		15:8	COMP[15:8]							
0x1A	COMP1	7:0	COMP[7:0]							
0x1B		15:8	COMP[15:8]							

Table 18-3. Register Summary - Mode 2 Registers

Offset	Name	Bit Pos.								
0x00	CTRL	7:0	MATCHCLR	CLKREP			MODE[1:0]		ENABLE	SWRST
0x01		15:8					PRESCALER[3:0]			
0x02	READREQ	7:0			ADDR[5:0]					
0x03		15:8	RREQ	RCONT						
0x04	EVCTRL	7:0	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
0x05		15:8	OVFEO							ALARMEO0
0x06	INTENCLR	7:0	OVF	SYNCRDY						ALARM0
0x07	INTENSET	7:0	OVF	SYNCRDY						ALARM0
0x08	INTFLAG	7:0	OVF	SYNCRDY						ALARM0
0x09	Reserved									
0x0A	STATUS	7:0	SYNCBUSY							
0x0B	DBGCTRL	7:0								DBGRUN
0x0C	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x0D	Reserved									
0x0E	Reserved									
0x0F	Reserved									
0x10	CLOCK	7:0	MINUTE[1:0]		SECOND[5:0]					
0x11		15:8	HOUR[3:0]				MINUTE[5:2]			
0x12		23:16	MONTH[1:0]		DAY[4:0]					HOUR[4]
0x13		31:24	YEAR[5:0]						MONTH[3:2]	
0x14	Reserved									
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	ALARM0	7:0	MINUTE[1:0]		SECOND[5:0]					
0x19		15:8	HOUR[3:0]				MINUTE[5:2]			
0x1A		23:16	MONTH[1:0]		DAY[4:0]					HOUR[4]
0x1B		31:24	YEAR[5:0]						MONTH[3:2]	
0x1C	MASK0	7:0						SEL[2:0]		

18.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 193 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 197 for details.

Some registers are enable-protected, meaning they can only be written when the RTC is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

18.8.1 Control

18.8.1.1 Mode 0

Name: CTRL

Offset: 0x00

Reset: 0x0000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
					PRESCALER[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MATCHCLR				MODE[1:0]		ENABLE	SWRST
Access	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 11:8 – PRESCALER[3:0]: Prescaler**
  These bits define the prescaling factor for the RTC clock source (GCLK_RTC) to generate the counter clock (CLK_RTC_CNT).
  These bits are not synchronized.

Table 18-4. Prescaler

PRESCALER[3:0]	Prescaler	Description
0x0	DIV1	CLK_RTC_CNT = GCLK_RTC/1
0x1	DIV2	CLK_RTC_CNT = GCLK_RTC/2
0x2	DIV4	CLK_RTC_CNT = GCLK_RTC/4
0x3	DIV8	CLK_RTC_CNT = GCLK_RTC/8
0x4	DIV16	CLK_RTC_CNT = GCLK_RTC/16
0x5	DIV32	CLK_RTC_CNT = GCLK_RTC/32
0x6	DIV64	CLK_RTC_CNT = GCLK_RTC/64
0x7	DIV128	CLK_RTC_CNT = GCLK_RTC/128
0x8	DIV256	CLK_RTC_CNT = GCLK_RTC/256
0x9	DIV512	CLK_RTC_CNT = GCLK_RTC/512
0xA	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
0xB-0xF	-	Reserved

- **Bit 7 – MATCHCLR: Clear on Match**
  This bit is valid only in Mode 0 and Mode 2. This bit can be written only when the peripheral is disabled.

  0: The counter is not cleared on a Compare/Alarm 0 match.
  1: The counter is cleared on a Compare/Alarm 0 match.

  This bit is not synchronized.

- **Bits 6:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:2 – MODE[1:0]: Operating Mode**
  These bits define the operating mode of the RTC.

  These bits are not synchronized.

Table 18-5. Peripheral Operating Mode

MODE[1:0]	Operating Mode	Description
0x0	COUNT32	Mode 0: 32-bit Counter
0x1	COUNT16	Mode 1: 16-bit Counter
0x2	CLOCK	Mode 2: Clock/Calendar
0x3	-	Reserved

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled or being disabled.

  1: The peripheral is enabled or being enabled.

  Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

  This bit is not enable-protected.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.

  1: The reset operation is ongoing.

  Writing a zero to this bit has no effect.

  Writing a one to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.

  Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

  Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

  This bit is not enable-protected.

18.8.1.2 Mode 1

Name: CTRL

Offset: 0x00

Reset: 0x0000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
					PRESCALER[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					MODE[1:0]		ENABLE	SWRST
Access	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 11:8 – PRESCALER[3:0]: Prescaler**
  These bits define the prescaling factor for the RTC clock source (GCLK_RTC) to generate the counter clock (CLK_RTC_CNT).
  These bits are not synchronized.

Table 18-6. Prescaler

PRESCALER[3:0]	Prescaler	Description
0x0	DIV1	CLK_RTC_CNT = GCLK_RTC/1
0x1	DIV2	CLK_RTC_CNT = GCLK_RTC/2
0x2	DIV4	CLK_RTC_CNT = GCLK_RTC/4
0x3	DIV8	CLK_RTC_CNT = GCLK_RTC/8
0x4	DIV16	CLK_RTC_CNT = GCLK_RTC/16
0x5	DIV32	CLK_RTC_CNT = GCLK_RTC/32
0x6	DIV64	CLK_RTC_CNT = GCLK_RTC/64
0x7	DIV128	CLK_RTC_CNT = GCLK_RTC/128
0x8	DIV256	CLK_RTC_CNT = GCLK_RTC/256
0x9	DIV512	CLK_RTC_CNT = GCLK_RTC/512
0xA	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
0xB-0xF	-	Reserved

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:2 – MODE[1:0]: Operating Mode**
  These bits define the operating mode of the RTC.
  These bits are not synchronized.

Table 18-7. Peripheral Operating Mode

MODE[1:0]	Operating Mode	Description
0x0	COUNT32	Mode 0: 32-bit Counter
0x1	COUNT16	Mode 1: 16-bit Counter
0x2	CLOCK	Mode 2: Clock/Calendar
0x3	-	Reserved

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled or being disabled.
  1: The peripheral is enabled or being enabled.
  Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.
  This bit is not enable-protected.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.
  Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.
  Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.
  This bit is not enable-protected.

18.8.1.3 Mode 2

Name: CTRL

Offset: 0x00

Reset: 0x0000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
					PRESCALER[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MATCHCLR	CLKREP			MODE[1:0]		ENABLE	SWRST
Access	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 11:8 – PRESCALER[3:0]: Prescaler**
  These bits define the prescaling factor for the RTC clock source (GCLK_RTC) to generate the counter clock (CLK_RTC_CNT).
  These bits are not synchronized.

Table 18-8. Prescaler

PRESCALER[3:0]	Prescaler	Description
0x0	DIV1	CLK_RTC_CNT = GCLK_RTC/1
0x1	DIV2	CLK_RTC_CNT = GCLK_RTC/2
0x2	DIV4	CLK_RTC_CNT = GCLK_RTC/4
0x3	DIV8	CLK_RTC_CNT = GCLK_RTC/8
0x4	DIV16	CLK_RTC_CNT = GCLK_RTC/16
0x5	DIV32	CLK_RTC_CNT = GCLK_RTC/32
0x6	DIV64	CLK_RTC_CNT = GCLK_RTC/64
0x7	DIV128	CLK_RTC_CNT = GCLK_RTC/128
0x8	DIV256	CLK_RTC_CNT = GCLK_RTC/256
0x9	DIV512	CLK_RTC_CNT = GCLK_RTC/512
0xA	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
0xB-0xF	-	Reserved

- **Bit 7 – MATCHCLR: Clear on Match**
  This bit is valid only in Mode 0 and Mode 2. This bit can be written only when the peripheral is disabled.

  0: The counter is not cleared on a Compare/Alarm 0 match.
  1: The counter is cleared on a Compare/Alarm 0 match.

  This bit is not synchronized.

- **Bit 6 – CLKREP: Clock Representation**
  This bit is valid only in Mode 2 and determines how the hours are represented in the Clock Value (CLOCK) register. This bit can be written only when the peripheral is disabled.

  0: 24 Hour
  1: 12 Hour (AM/PM)

  This bit is not synchronized.

- **Bits 5:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:2 – MODE[1:0]: Operating Mode**
  These bits define the operating mode of the RTC.

  These bits are not synchronized.

Table 18-9. Peripheral Operating Mode

MODE[1:0]	Operating Mode	Description
0x0	COUNT32	Mode 0: 32-bit Counter
0x1	COUNT16	Mode 1: 16-bit Counter
0x2	CLOCK	Mode 2: Clock/Calendar
0x3	-	Reserved

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled or being disabled.

  1: The peripheral is enabled or being enabled.

  Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

  This bit is not enable-protected.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.

  1: The reset operation is ongoing.

  Writing a zero to this bit has no effect.

  Writing a one to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.

  Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

  Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

  This bit is not enable-protected.

18.8.2 Read Request

Name: READREQ

Offset: 0x02

Reset: 0x0010

Property: –

-

Bit	15	14	13	12	11	10	9	8
	RREQ	RCONT						
Access	W	R/W	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			ADDR[5:0]					
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	0	0

- **Bit 15 – RREQ: Read Request**
  Writing a zero to this bit has no effect.
  Writing a one to this bit requests synchronization of the register pointed to by the Address bit group (READREQ.ADDR) and sets the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY).

- **Bit 14 – RCONT: Read Continuously**
  Writing a zero to this bit disables continuous synchronization.
  Writing a one to this bit enables continuous synchronization of the register pointed to by READREQ.ADDR. The register value will be synchronized automatically every time the register is updated. READREQ.RCONT prevents READREQ.RREQ from clearing automatically.
  This bit is cleared when the register pointed to by READREQ.ADDR is written.

- **Bits 13:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:0 – ADDR: Address**
  These bits select the offset of the register that needs read synchronization. In the RTC only the COUNT and CLOCK registers, which share the same address, are available for read synchronization. Therefore, the ADDR bit group is a read-only constant of 0x10.

18.8.3 Event Control

18.8.3.1 Mode 0

Name: EVCTRL

Offset: 0x04

Reset: 0x0000

Property: Write-Protected, Enable-Protected

Bit	15	14	13	12	11	10	9	8
	OVFEO							CMPEO0
Access	R/W	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – OVFEO: Overflow Event Output Enable**
  0: Overflow event is disabled and will not be generated.
  1: Overflow event is enabled and will be generated for every overflow.
- **Bits 14:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 8 – CMPEO0: Compare 0 Event Output Enable**
  0: Compare 0 event is disabled and will not be generated.
  1: Compare 0 event is enabled and will be generated for every compare match.
- **Bits 7:0 – PEREOx: Periodic Interval x Event Output Enable**
  0: Periodic Interval m event is disabled and will not be generated.
  1: Periodic Interval m event is enabled and will be generated.

18.8.3.2 Mode 1

Name: EVCTRL

Offset: 0x04

Reset: 0x0000

Property: Write-Protected, Enable-Protected

Bit	15	14	13	12	11	10	9	8
	OVFEO						CMPEO1	CMPEO0
Access	R/W	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – OVFEO: Overflow Event Output Enable**
  0: Overflow event is disabled and will not be generated.
  1: Overflow event is enabled and will be generated for every overflow.
- **Bits 14:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 9 – CMPEO1: Compare Event Output Enable 1**
  0: Compare 1 event is disabled and will not be generated.
  1: Compare 1 event is enabled and will be generated for every compare match.
- **Bit 8 – CMPEO0: Compare Event Output Enable 0**
  0: Compare 0 event is disabled and will not be generated.
  1: Compare 0 event is enabled and will be generated for every compare match.
- **Bits 7:0 – PEREOx: Periodic Interval x Event Output Enable**
  0: Periodic Interval m event is disabled and will not be generated.
  1: Periodic Interval m event is enabled and will be generated.

18.8.3.3 Mode 2

Name: EVCTRL

Offset: 0x04

Reset: 0x0000

Property: Write-Protected, Enabled-Protected

Bit	15	14	13	12	11	10	9	8
	OVFEO							ALARMEO0
Access	R/W	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – OVFEO: Overflow Event Output Enable**
  0: Overflow event is disabled and will not be generated.
  1: Overflow event is enabled and will be generated for every overflow.
- **Bits 14:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 8 – ALARMEO0: Alarm 0 Event Output Enable**
  0: Alarm 0 event is disabled and will not be generated.
  1: Alarm 0 event is enabled and will be generated for every alarm.
- **Bits 7:0 – PEREOx: Periodic Interval *x* Event Output Enable**
  0: Periodic Interval n event is disabled and will not be generated.
  1: Periodic Interval n event is enabled and will be generated.

18.8.4 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

18.8.4.1 Mode 0

Name: INTENCLR

Offset: 0x06

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY						CMP0
Access	R/W	R/W	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow Interrupt Enable**
  0: The Overflow interrupt is disabled.
  1: The Overflow interrupt is enabled, and an interrupt request will be generated when the Overflow interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Overflow Interrupt Enable bit and disable the corresponding interrupt.

- **Bit 6 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and disable the corresponding interrupt.

- **Bits 5:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – CMP0: Compare 0 Interrupt Enable**
  0: The Compare 0 interrupt is disabled.
  1: The Compare 0 interrupt is enabled, and an interrupt request will be generated when the Compare 0 interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Compare 0 Interrupt Enable bit and disable the corresponding interrupt.

18.8.4.2 Mode 1

Name: INTENCLR

Offset: 0x06

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY					CMP1	CMP0
Access	R/W	R/W	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow Interrupt Enable**
  0: The Overflow interrupt is disabled.
  1: The Overflow interrupt is enabled, and an interrupt request will be generated when the Overflow interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Overflow Interrupt Enable bit and disable the corresponding interrupt.

- **Bit 6 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and disable the corresponding interrupt.

- **Bits 5:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – CMP1: Compare 1 Interrupt Enable**
  0: The Compare 1 interrupt is disabled.
  1: The Compare 1 interrupt is enabled, and an interrupt request will be generated when the Compare 1 interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Compare 1 Interrupt Enable bit and disable the corresponding interrupt.

- **Bit 0 – CMP0: Compare 0 Interrupt Enable**
  0: The Compare 0 interrupt is disabled.
  1: The Compare 0 interrupt is enabled, and an interrupt request will be generated when the Compare 0 interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Compare 0 Interrupt Enable bit and disable the corresponding interrupt.

18.8.4.3 Mode 2

Name: INTENCLR

Offset: 0x06

Reset: 0x00

Property: Write-protected

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY						ALARM0
Access	R/W	R/W	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow Interrupt Enable**
  0: The Overflow interrupt is disabled.
  1: The Overflow interrupt is enabled, and an interrupt request will be generated when the Overflow interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Overflow Interrupt Enable bit and disable the corresponding interrupt.

- **Bit 6 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The synchronization ready interrupt is disabled.
  1: The synchronization ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and disable the corresponding interrupt.

- **Bits 5:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – ALARM0: Alarm 0 Interrupt Enable**
  0: The Alarm 0 interrupt is disabled.
  1: The Alarm 0 interrupt is enabled, and an interrupt request will be generated when the Alarm 0 interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the Alarm 0 interrupt.

18.8.5 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear (INTENCLR) register.

18.8.5.1 Mode 0

Name: INTENSET

Offset: 0x07

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY						CMP0
Access	R/W	R/W	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow Interrupt Enable**
  0: The overflow interrupt is disabled.
  1: The overflow interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Overflow Interrupt Enable bit and enable the Overflow interrupt.

- **Bit 6 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The synchronization ready interrupt is disabled.
  1: The synchronization ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit and enable the Synchronization Ready interrupt.

- **Bits 5:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – CMP0: Compare 0 Interrupt Enable**
  0: The compare 0 interrupt is disabled.
  1: The compare 0 interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Compare 0 Interrupt Enable bit and enable the Compare 0 interrupt.

18.8.5.2 Mode 1

Name: INTENSET

Offset: 0x07

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY					CMP1	CMP0
Access	R/W	R/W	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow Interrupt Enable**
  0: The overflow interrupt is disabled.
  1: The overflow interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Overflow interrupt bit and enable the Overflow interrupt.

- **Bit 6 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The synchronization ready interrupt is disabled.
  1: The synchronization ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit and enable the Synchronization Ready interrupt.

- **Bits 5:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – CMP1: Compare 1 Interrupt Enable**
  0: The compare 1 interrupt is disabled.
  1: The compare 1 interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Compare 1 Interrupt Enable bit and enable the Compare 1 interrupt.

- **Bit 0 – CMP0: Compare 0 Interrupt Enable**
  0: The compare 0 interrupt is disabled.
  1: The compare 0 interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Compare 0 Interrupt Enable bit and enable the Compare 0 interrupt.

18.8.5.3 Mode 2

Name: INTENSET

Offset: 0x07

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY						ALARM0
Access	R/W	R/W	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow Interrupt Enable**
  0: The overflow interrupt is disabled.
  1: The overflow interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Overflow Interrupt Enable bit and enable the Overflow interrupt.

- **Bit 6 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The synchronization ready interrupt is disabled.
  1: The synchronization ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Synchronization Ready Interrupt bit and enable the Synchronization Ready interrupt.
  Reading this bit returns the state of the synchronization ready interrupt enable.

- **Bits 5:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – ALARM0: Alarm0 Interrupt Enable**
  0: The alarm 0 interrupt is disabled.
  1: The alarm 0 interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Alarm 0 Interrupt Enable bit and enable the Alarm 0 interrupt.

18.8.6 Interrupt Flag Status and Clear

18.8.6.1 Mode 0

Name: INTFLAG

Offset: 0x08

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY						CMP0
Access	R/W	R/W	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs, and an interrupt request will be generated ifINTENCLR/SET.OVF is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Overflow interrupt flag.

- **Bit 6 – SYNCRDY: Synchronization Ready**
  This flag is cleared by writing a one to the flag.
  This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by Enable or software Reset, and an interrupt request will be generated if INTENCLR/SET.SYNCRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Synchronization Ready interrupt flag.

- , and an interrupt request will be generated **Bits 5:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – CMP0: Compare 0**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition, and an interrupt request will be generated if INTENCLR/SET.COMP0 is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Compare 0 interrupt flag.

18.8.6.2 Mode 1

Name: INTFLAG

Offset: 0x08

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY					CMP1	CMP0
Access	R/W	R/W	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs, and an interrupt request will be generated ifINTENCLR/SET.OVF is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Overflow interrupt flag.

- **Bit 6 – SYNCRDY: Synchronization Ready**
  This flag is cleared by writing a one to the flag.
  This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by Enable or software Reset, and an interrupt request will be generated if INTENCLR/SET.SYNCRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Synchronization Ready interrupt flag.

- , and an interrupt request will be generated**Bits 5:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – CMP1: Compare 1**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition, and an interrupt request will be generated if INTENCLR/SET.COMP1 is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Compare 1 interrupt flag.

- **Bit 0 – CMP0: Compare 0**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition, and an interrupt request will be generated if INTENCLR/SET.COMP0 is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Compare 0 interrupt flag.

18.8.6.3 Mode 2

Name: INTFLAG

Offset: 0x08

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
	OVF	SYNCRDY						ALARM0
Access	R/W	R/W	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – OVF: Overflow**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs, and an interrupt request will be generated ifINTENCLR/SET.OVF is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Overflow interrupt flag.

- **Bit 6 – SYNCRDY: Synchronization Ready**
  This flag is cleared by writing a one to the flag.
  This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by Enable or software Reset, and an interrupt request will be generated if INTEN-CLR/SET.SYNCRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Synchronization Ready interrupt flag.

- , and an interrupt request will be generated**Bits 5:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – ALARM0: Alarm 0**
  This flag is cleared by writing a one to the flag.
  This flag is set on the next CLK_RTC_CNT cycle after a match with ALARM0 condition occurs, and an interrupt request will be generated if INTENCLR/SET.ALARM0 is also one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Alarm 0 interrupt flag.

18.8.7 Status

Name: STATUS

Offset: 0x0A

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

18.8.8 Debug Control

Name: DBGCTRL

Offset: 0x0B

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
								DBGRUN
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – DBGRUN: Run During Debug**
  This bit is not reset by a software reset.

  Writing a zero to this bit causes the RTC to halt during debug mode.
  Writing a one to this bit allows the RTC to continue normal operation during debug mode.

18.8.9 Frequency Correction

Name: FREQCORR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
	SIGN	VALUE[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SIGN: Correction Sign**
  0: The correction value is positive, i.e., frequency will be increased.
  1: The correction value is negative, i.e., frequency will be decreased.

- **Bits 6:0 – VALUE[6:0]: Correction Value**
  These bits define the amount of correction applied to the RTC prescaler.
  0: Correction is disabled and the RTC frequency is unchanged.
  1–127: The RTC frequency is adjusted according to the value.

18.8.10 Counter Value

18.8.10.1 Mode 0

Name: COUNT

Offset: 0x10

Reset: 0x00000000

Property: Write-Protected, Write-Synchronized, Read-Synchronized

Bit	31	30	29	28	27	26	25	24
	COUNT[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COUNT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – COUNT[31:0]: Counter Value**
  These bits define the value of the 32-bit RTC counter.

18.8.10.2 Mode 1

Name: COUNT

Offset: 0x10

Reset: 0x0000

Property: Write-Protected, Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – COUNT[15:0]: Counter Value**
  These bits define the value of the 16-bit RTC counter.

18.8.11 Clock Value

18.8.11.1 Mode 2

Name: CLOCK

Offset: 0x10

Reset: 0x00000000

Property: Write-Protected, Write-Synchronized, Read-Synchronized

Bit	31	30	29	28	27	26	25	24
	YEAR[5:0]						MONTH[3:2]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	MONTH[1:0]		DAY[4:0]					HOUR[4]
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	HOUR[3:0]				MINUTE[5:2]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MINUTE[1:0]		SECOND[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:26 – YEAR[5:0]: Year**
  The year offset with respect to the reference year (defined in software).
  The year is considered a leap year if YEAR[1:0] is zero.
- **Bits 25:22 – MONTH[3:0]: Month**
  1 – January
  2 – February
  …
  12 – December
- **Bits 21:17 – DAY[4:0]: Day**
  Day starts at 1 and ends at 28, 29, 30 or 31, depending on the month and year.
- **Bits 16:12 – HOUR[4:0]: Hour**
  When CTRL.CLKREP is zero, the Hour bit group is in 24-hour format, with values 0-23. When CTRL.CLKREP is one, HOUR[3:0] has values 1-12 and HOUR[4] represents AM (0) or PM (1).
- **Bits 11:6 – MINUTE[5:0]: Minute**
  0 – 59.

- **Bits 5:0 – SECOND[5:0]: Second**
  0– 59.

18.8.12 Counter Period

18.8.12.1 Mode 1

Name: PER

Offset: 0x14

Reset: 0x0000

Property: Write-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	PER[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – PER[15:0]: Counter Period**
  These bits define the value of the 16-bit RTC period.

18.8.13 Compare n Value

18.8.13.1 Mode 0

Name: COMPn

Offset: 0x18 + n*0x4 [n=0..3]

Reset: 0x00000000

Property: Write-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
	COMP[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COMP[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COMP[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COMP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – COMP[31:0]: Compare Value**
  The 32-bit value of COMPn is continuously compared with the 32-bit COUNT value. When a match occurs, the Compare n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn) is set on the next counter cycle, and the counter value is cleared if CTRL.MATCHCLR is one.

18.8.13.2 Mode 1

Name: COMPn

Offset: 0x18 + n*0x2 [n=0..5]

Reset: 0x0000

Property: Write-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	COMP[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COMP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – COMP[15:0]: Compare Value**
  The 16-bit value of COMPn is continuously compared with the 16-bit COUNT value. When a match occurs, the Compare n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn) is set on the next counter cycle.

18.8.14 Alarm n Value

18.8.14.1 Mode 2

Name: ALARMn

Offset: 0x18 + n*0x8 [n=0..3]

Reset: 0x00000000

Property: Write-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
	YEAR[5:0]						MONTH[3:2]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	MONTH[1:0]		DAY[4:0]					HOUR[4]
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	HOUR[3:0]				MINUTE[5:2]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MINUTE[1:0]		SECOND[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

The 32-bit value of ALARMn is continuously compared with the 32-bit CLOCK value, based on the masking set by MASKn.SEL. When a match occurs, the Alarm n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.ALARMn) is set on the next counter cycle, and the counter is cleared if CTRL.MATCHCLR is one.

- **Bits 31:26 – YEAR[5:0]: Year**
  The alarm year. Years are only matched if MASKn.SEL is 6.
- **Bits 25:22 – MONTH[3:0]: Month**
  The alarm month. Months are matched only if MASKn.SEL is greater than 4.
- **Bits 21:17 – DAY[4:0]: Day**
  The alarm day. Days are matched only if MASKn.SEL is greater than 3.
- **Bits 16:12 – HOUR[4:0]: Hour**
  The alarm hour. Hours are matched only if MASKn.SEL is greater than 2.
- **Bits 11:6 – MINUTE[5:0]: Minute**
  The alarm minute. Minutes are matched only if MASKn.SEL is greater than 1.

- **Bits 5:0 – SECOND[5:0]: Second**
  The alarm second. Seconds are matched only if MASKn.SEL is greater than 0.

18.8.15 Alarm n Mask

18.8.15.1 Mode 2

Name: MASKn

Offset: 0x1C + n*0x8 [n=0..3]

Reset: 0x00

Property: Write-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
						SEL[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – SEL[2:0]: Alarm Mask Selection**
  These bits define which bit groups of Alarm n are valid.

Table 18-10. Alarm Mask Selection

SEL[2:0]	Alarm Mask Selection	Description
0x0	OFF	Alarm Disabled
0x1	SS	Match seconds only
0x2	MMSS	Match seconds and minutes only
0x3	HHMMSS	Match seconds, minutes and hours only
0x4	DDHHMMSS	Match seconds, minutes, hours and days only
0x5	MMDDHHMMSS	Match seconds, minutes, hours, days and months only
0x6	YYMMDDHHMMSS	Match seconds, minutes, hours, days, months and years
0x7	-	Reserved

19. EIC – External Interrupt Controller

19.1 Overview

The External Interrupt Controller (EIC) allows external pins to be configured as interrupt lines. Each interrupt line can be individually masked and can generate an interrupt on rising, falling or both edges, or on high or low levels. Each external pin has a configurable filter to remove spikes. Each external pin can also be configured to be asynchronous in order to wake up the device from sleep modes where all clocks have been disabled. External pins can also generate an event.

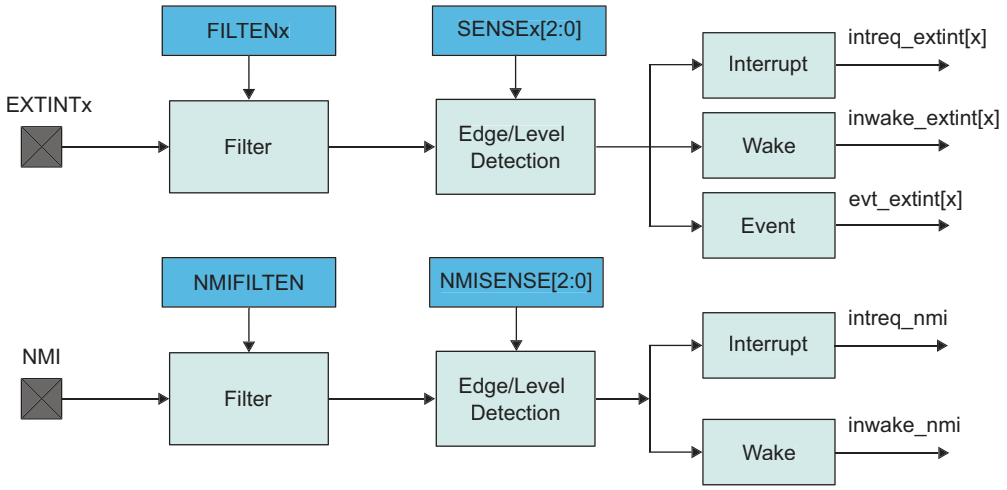
A separate non-maskable interrupt (NMI) is also supported. It has properties similar to the other external interrupts, but is connected to the NMI request of the CPU, enabling it to interrupt any other interrupt mode.

19.2 Features

- 16 external pins, plus 1 non-maskable pin
- Dedicated interrupt line for each pin
- Individually maskable interrupt lines
- Interrupt on rising, falling or both edges
- Interrupt on high or low levels
- Asynchronous interrupts for sleep modes without clock
- Filtering of external pins
- Event generation
- Configurable wake-up for sleep modes

19.3 Block Diagram

Figure 19-1. EIC Block Diagram



19.4 Signal Description

Signal Name	Type	Description
EXTINT[15..0]	Digital Input	External interrupt pin
NMI	Digital Input	Non-maskable interrupt pin

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

19.5 Product Dependencies

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

19.5.1 I/O Lines

Using the EIC's I/O lines requires the I/O pins to be configured. Refer to "PORT" on page 276 for details.

19.5.2 Power Management

All interrupts are available in all sleep modes, but the EIC can be configured to automatically mask some interrupts in order to prevent device wake-up.

The EIC will continue to operate in any sleep mode where the selected source clock is running. The EIC's interrupts can be used to wake up the device from sleep modes. Events connected to the Event System can trigger other operations in the system without exiting sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

19.5.3 Clocks

The EIC bus clock (CLK_EIC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_EIC_APB can be found in the Peripheral Clock Masking section in "PM – Power Manager" on page 100.

A generic clock (GCLK_EIC) is required to clock the peripheral. This clock must be configured and enabled in the Generic Clock Controller before using the peripheral. Refer to "GCLK – Generic Clock Controller" on page 78 for details.

This generic clock is asynchronous to the user interface clock (CLK_EIC_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to "Synchronization" on page 240 for further details.

19.5.4 Interrupts

There are two interrupt request lines, one for the external interrupts (EXTINT) and one for non-maskable interrupt (NMI).

The EXTINT interrupt request line is connected to the interrupt controller. Using the EIC interrupt requires the interrupt controller to be configured first. Refer to "Nested Vector Interrupt Controller" on page 24 for details.

The NMI interrupt request line is also connected to the interrupt controller, but does not require the interrupt to be configured.

19.5.5 Events

The events are connected to the Event System. Using the events requires the Event System to be configured first. Refer to "EVSYS – Event System" on page 301 for details.

19.5.6 Debug Operation

When the CPU is halted in 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.

19.5.7 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Non-Maskable Interrupt Flag Status and Clear register (NMIFLAG)

Write-protection is denoted by the Write-Protected property in the register description.

Write-protection does not apply to accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

19.5.8 Analog Connections

Not applicable.

19.6 Functional Description

19.6.1 Principle of Operation

The EIC detects edge or level condition to generate interrupts to the CPU Interrupt Controller or events to the Event System. Each external interrupt pin (EXTINT) can be filtered using majority vote filtering, clocked by generic clock GCLK_EIC.

19.6.2 Basic Operation

19.6.2.1 Initialization

The EIC must be initialized in the following order:

1. Enable CLK_EIC_APB
2. If edge detection or filtering is required, GCLK_EIC must be enabled
3. Write the EIC configuration registers (NMICTRL, EVCTRL, WAKEUP, CONFIGy)
4. Enable the EIC

When NMI is used, GCLK_EIC must be enabled after EIC configuration

19.6.2.2 Enabling, Disabling and Resetting

The EIC is enabled by writing a one to the Enable bit in the Control register (CTRL.ENABLE). The EIC is disabled by writing a zero to CTRL.ENABLE.

The EIC is reset by writing a one to the Software Reset bit in the Control register (CTRL.SWRST). All registers in the EIC will be reset to their initial state, and the EIC will be disabled.

Refer to CTRL register for details.

19.6.3 External Pin Processing

Each external pin can be configured to generate an interrupt/event on edge detection (rising, falling or both edges) or level detection (high or low). The sense of external pins is configured by writing the Interrupt Sense x bits in the Config y register (CONFIGy.SENSEx). The corresponding interrupt flag (INTFLAG.EXTINT[x]) in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition is met (CONFIGy.SENSEx must be different from zero).

When the interrupt has been cleared in edge-sensitive mode, INTFLAG.EXTINT[x] will only be set if a new interrupt condition is met. In level-sensitive mode, when interrupt has been cleared, INTFLAG.EXTINT[x] will be set immediately if the EXTINTx pin still matches the interrupt condition.

Each external pin can be filtered by a majority vote filtering, clocked by GCLK_EIC. Filtering is enabled if bit Filter Enable x in the Configuration y register (CONFIGy.FILTENx) is written to one. The majority vote filter samples the external pin three times with GCLK_EIC and outputs the value when two or more samples are equal.

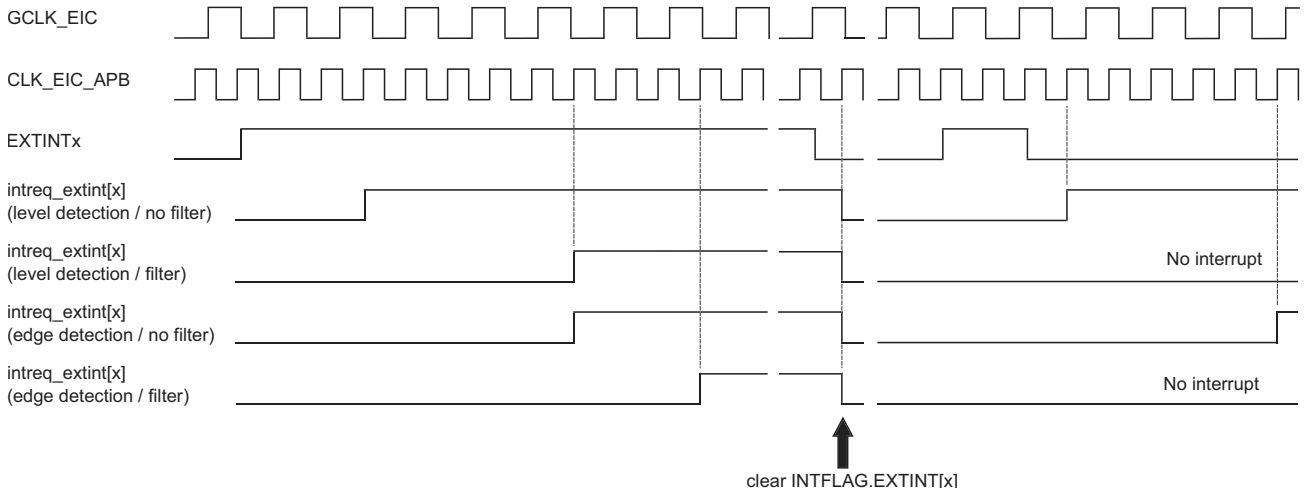
Table 19-1. Majority Vote Filter

Samples [0, 1, 2]	Filter Output
[0,0,0]	0
[0,0,1]	0
[0,1,0]	0
[0,1,1]	1
[1,0,0]	0
[1,0,1]	1
[1,1,0]	1
[1,1,1]	1

When an external interrupt is configured for level detection, or if filtering is disabled, detection is made asynchronously, and GCLK_EIC is not required.

If filtering or edge detection is enabled, the EIC automatically requests the GCLK_EIC to operate (GCLK_EIC must be enabled in the GCLK module, see "GCLK – Generic Clock Controller" on page 78 for details). If level detection is enabled, GCLK_EIC is not required, but interrupt and events can still be generated.

Figure 19-2. Interrupt detections



The detection delay depends on the detection mode.

Table 19-2. Interrupt Latency

Detection Mode	Latency (Worst Case)
Level without filter	3 CLK_EIC_APB periods
Level with filter	4 GCLK_EIC periods + 3 CLK_EIC_APB periods
Edge without filter	4 GCLK_EIC periods + 3 CLK_EIC_APB periods
Edge with filter	6 GCLK_EIC periods + 3 CLK_EIC_APB periods

19.6.4 Additional Features

The non-maskable interrupt pin can also generate an interrupt on edge or level detection, but it is configured with the dedicated NMI Control register (NMICTRL). To select the sense for NMI, write to the NMISENSE bit group in the NMI Control register (NMICTRL.NMISENSE). NMI filtering is enabled by writing a one to the NMI Filter Enable bit (NMICTRL.NMIFILTEN).

NMI detection is enabled only by the NMICTRL.NMISENSE value, and the EIC is not required to be enabled.

After reset, NMI is configured to no detection mode.

When an NMI is detected, the non-maskable interrupt flag in the NMI Flag Status and Clear register is set (NMIFLAG.NMI). NMI interrupt generation is always enabled, and NMIFLAG.NMI generates an interrupt request when set.

19.6.5 Interrupts

The EIC has the following interrupt sources:

- External interrupt pins (EXTINTx). See “Basic Operation” on page 237
- Non-maskable interrupt pin (NMI). See “Additional Features” on page 239

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when an interrupt condition occurs (NMIFLAG for NMI). Each interrupt, except NMI, can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the EIC is reset. See the INTFLAG register for details on how to clear interrupt flags. The EIC has one common interrupt request line for all the interrupt sources (except the NMI interrupt request line). Refer to “Processor and Architecture” on page 23 for details. The user must read the INTFLAG (or NMIFLAG) register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Processor and Architecture” on page 23 for details.

19.6.6 Events

The EIC can generate the following output events:

- External event from pin (EXTINTx).

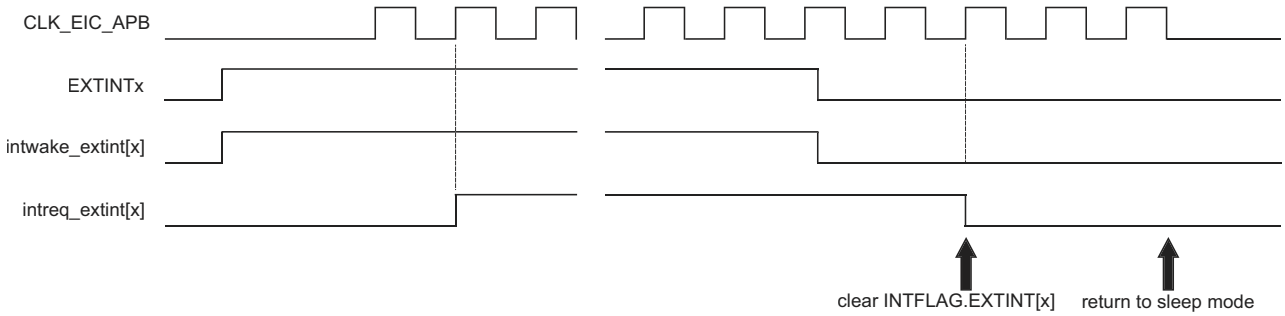
Writing a one to an Event Output Control register (EVCTRLEXTINTEO) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. Refer to “EVSYS – Event System” on page 301 for details on configuring the Event System.

When the condition on pin EXTINTx matches the configuration in the CONFIGy register, the corresponding event is generated, if enabled.

19.6.7 Sleep Mode Operation

In sleep modes, an EXTINTx pin can wake up the device if the corresponding condition matches the configuration in CONFIGy register. Writing a one to a Wake-Up Enable bit (WAKEUP.WAKEUPEN[x]) enables the wake-up from pin EXTINTx. Writing a zero to a Wake-Up Enable bit (WAKEUP.WAKEUPEN[x]) disables the wake-up from pin EXTINTx.

Figure 19-3. Wake-Up Operation Example (High-Level Detection, No Filter, WAKEUPEN[x]=1)



19.6.8 Synchronization

Due to the asynchronicity between CLK_EIC_APB and GCLK_EIC, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled, and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control register (CTRL.SWRST)
- Enable bit in the Control register (CTRL.ENABLE)

No register needs synchronization when written.

No register needs synchronization when read.

19.7 Register Summary

Table 19-3. Register Summary

Offset	Name	Bit Pos.								
0x00	CTRL	7:0							ENABLE	SWRST
0x01	STATUS	7:0	SYNCBUSY							
0x02	NMICTRL	7:0					NMIFILTEN	NMISENSE[2:0]		
0x03	NMIFLAG	7:0								NMI
0x04	EVCTRL	7:0	EXTINTEO[7:0]							
0x05		15:8	EXTINTEO[15:8]							
0x06		23:16								
0x07		31:24								
0x08	INTENCLR	7:0	EXTINT[7:0]							
0x09		15:8	EXTINT[15:8]							
0x0A		23:16								
0x0B		31:24								
0x0C	INTENSET	7:0	EXTINT[7:0]							
0x0D		15:8	EXTINT[15:8]							
0x0E		23:16								
0x0F		31:24								
0x10	INTFLAG	7:0	EXTINT[7:0]							
0x11		15:8	EXTINT[15:8]							
0x12		23:16								
0x13		31:24								
0x14	WAKEUP	7:0	WAKEUPEN[7:0]							
0x15		15:8	WAKEUPEN[15:8]							
0x16		23:16								
0x17		31:24								
0x18	CONFIG0	7:0	FILTEN1	SENSE1[2:0]			FILTEN0	SENSE0[2:0]		
0x19		15:8	FILTEN3	SENSE3[2:0]			FILTEN2	SENSE2[2:0]		
0x1A		23:16	FILTEN5	SENSE5[2:0]			FILTEN4	SENSE4[2:0]		
0x1B		31:24	FILTEN7	SENSE7[2:0]			FILTEN6	SENSE6[2:0]		
0x1C	CONFIG1	7:0	FILTEN9	SENSE9[2:0]			FILTEN8	SENSE8[2:0]		
0x1D		15:8	FILTEN11	SENSE11[2:0]			FILTEN10	SENSE10[2:0]		
0x1E		23:16	FILTEN13	SENSE13[2:0]			FILTEN12	SENSE12[2:0]		
0x1F		31:24	FILTEN15	SENSE15[2:0]			FILTEN14	SENSE14[2:0]		

19.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-protected property in each individual register description. Refer to “Register Access Protection” on page 237 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Synchronized property in each individual register description. Refer to “Synchronization” on page 240 for details.

Some registers are enable-protected, meaning they can be written only when the EIC is disabled. Enable-protection is denoted by the Enabled-Protected property in each individual register description.

19.8.1 Control

Name: CTRL

Offset: 0x00

Reset: 0x00

Property: Write-Protected,Write-Synchronized

Bit	7	6	5	4	3	2	1	0
							ENABLE	SWRST
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – ENABLE: Enable**
  0: The EIC is disabled.
  1: The EIC is enabled.
  Due to synchronization, there is delay from writing CTRL.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

- **Bit 0 – SWRST: Software Reset**
  0: There is no ongoing reset operation.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the EIC to their initial state, and the EIC will be disabled.
  Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write operation will be discarded.
  Due to synchronization, there is a delay from writing CTRL.SWRST until the reset is complete. CTRL.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

19.8.2 Status

Name: STATUS

Offset: 0x01

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

19.8.3 Non-Maskable Interrupt Control

Name: NMICTRL

Offset: 0x02

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
					NMIFILTEN	NMISENSE[2:0]		
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 3 – NMIFILTEN: Non-Maskable Interrupt Filter Enable**
  0: NMI filter is disabled.
  1: NMI filter is enabled.
- **Bits 2:0 – NMISENSE: Non-Maskable Interrupt Sense**
  These bits define on which edge or level the NMI triggers.

Table 19-4. NMI Sense Configuration

NMISENSE	Name	Description
0x0	NONE	No detection
0x1	RISE	Rising-edge detection
0x2	FALL	Falling-edge detection
0x3	BOTH	Both-edges detection
0x4	HIGH	High-level detection
0x5	LOW	Low-level detection
0x6-0x7	-	Reserved

19.8.4 Non-Maskable Interrupt Flag Status and Clear

Name: NMIFLAG

Offset: 0x03

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
								NMI
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – NMI: Non-Maskable Interrupt**
  This flag is cleared by writing a one to it.
  This flag is set when the NMI pin matches the NMI sense configuration, and will generate an interrupt request.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the non-maskable interrupt flag.

19.8.5 Event Control

Name: EVCTRL

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EXTINTEO[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EXTINTEO[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:0 – EXTINTEO: External Interrupt x Event Output Enable**
  These bits indicate whether the event associated with the EXTINTx pin is enabled or not to generated for every detection.

  0: Event from pin EXTINTx is disabled.

  1: Event from pin EXTINTx is enabled.

19.8.6 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x08

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EXTINT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EXTINT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:0 – EXTINT: External Interrupt x Enable**
  0: The external interrupt x is disabled.
  1: The external interrupt x is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the External Interrupt x Enable bit, which enables the external interrupt.

19.8.7 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear (INTENCLR) register.

Name: INTENSET

Offset: 0x0C

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EXTINT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EXTINT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:0 – EXTINT: External Interrupt x Enable**
  0: The external interrupt x is disabled.
  1: The external interrupt x is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the External Interrupt x Enable bit, which enables the external interrupt.

19.8.8 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x10

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EXTINT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EXTINT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:0 – EXTINT: External Interrupt x**
  This flag is cleared by writing a one to it.
  This flag is set when EXTINTx pin matches the external interrupt sense configuration and will generate an interrupt request if INTENCLR/SET.EXTINT[x] is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the External Interrupt x flag.

19.8.9 Wake-Up Enable

Name: WAKEUP

Offset: 0x14

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WAKEUPEN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WAKEUPEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:0 – WAKEUPEN: External Interrupt x Wake-up Enable**
  This bit enables or disables wake-up from sleep modes when the EXTINTx pin matches the external interrupt sense configuration.

  0: Wake-up from the EXTINTx pin is disabled.

  1: Wake-up from the EXTINTx pin is enabled.

19.8.10 Configuration n

Name: CONFIGn

Offset: 0x18+n*0x4 [n=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	FILTEN7	SENSE7[2:0]			FILTEN6	SENSE6[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FILTEN5	SENSE5[2:0]			FILTEN4	SENSE4[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FILTEN3	SENSE3[2:0]			FILTEN2	SENSE2[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FILTEN1	SENSE1[2:0]			FILTEN0	SENSE0[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31, 27, 23, 19, 15, 11, 7, 3 – FILTENx [x=7..0]: Filter x Enable**
  0: Filter is disabled for EXTINT[n*8+x] input.
  1: Filter is enabled for EXTINT[n*8+x] input.
- **Bits 30:28, 26:24, 22:20, 18:16, 14:12, 10:8, 6:4, 2:0 – SENSEx[2:0] [x=7..0]: Input Sense x Configuration**
  These bits define on which edge or level the interrupt or event for EXTINT[n*8+x] will be generated.

Table 19-5. Sense Configuration

SENSE	Name	Description
0x0	NONE	No detection
0x1	RISE	Rising-edge detection
0x2	FALL	Falling-edge detection
0x3	BOTH	Both-edges detection
0x4	HIGH	High-level detection
0x5	LOW	Low-level detection
0x6-0x7	-	Reserved

20. NVMCTRL – Non-Volatile Memory Controller

20.1 Overview

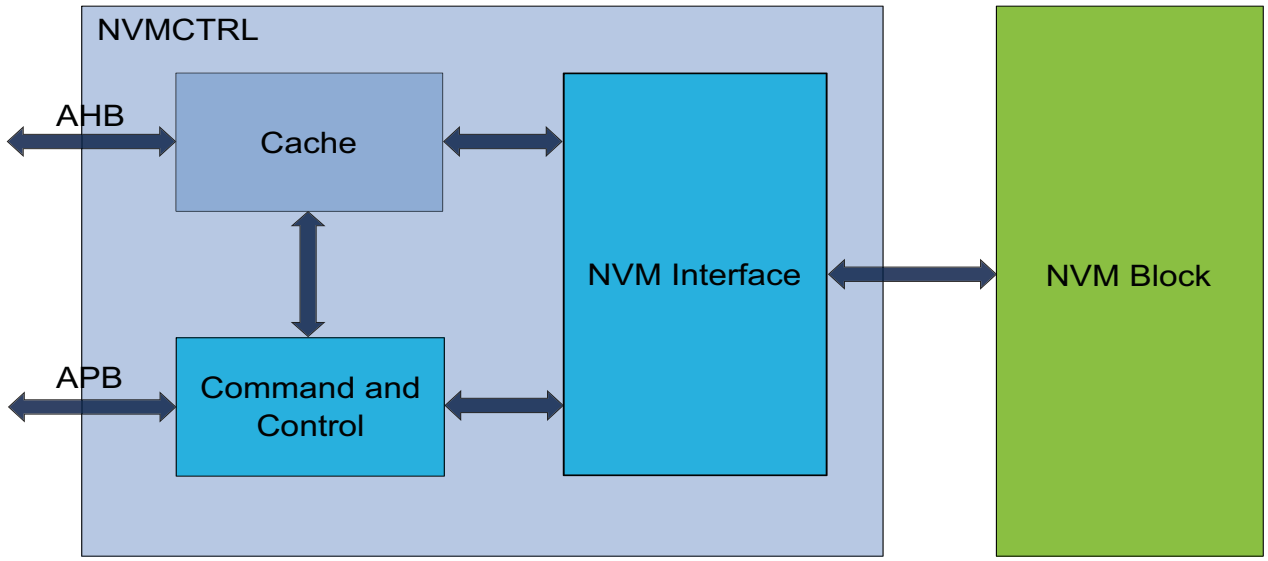
Non-volatile memory (NVM) is a reprogrammable flash memory that retains program and data storage even with power off. The NVM Controller (NVMCTRL) connects to the AHB and APB bus interfaces for system access to the NVM block. The AHB interface is used for reads and writes to the NVM block, while the APB interface is used for commands and configuration.

20.2 Features

- 32-bit AHB interface for reads and writes
- All NVM sections are memory mapped to the AHB, including calibration and system configuration
- 32-bit APB interface for commands and control
- Programmable wait states for read optimization
- 16 regions can be individually protected or unprotected
- Additional protection for boot loader
- Supports device protection through a security bit
- Interface to Power Manager for power-down of flash blocks in sleep modes
- Can optionally wake up on exit from sleep or on first access
- Direct-mapped cache

20.3 Block Diagram

Figure 20-1. Block Diagram



20.4 Signal Description

Not applicable

20.5 Product Dependencies

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

20.5.1 Power Management

The NVMCTRL will continue to operate in any sleep mode where the selected source clock is running. The NVMCTRL's interrupts can be used to wake up the device from sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

The Power Manager will automatically put the NVM block into a low-power state when entering sleep mode. This is based on the Control B register (CTRLB) SLEEPPRM bit setting. Read the CTRLB register description for more details.

20.5.2 Clocks

Two synchronous clocks are used by the NVMCTRL. One is provided by the AHB bus (CLK_NVMCTRL_AHB) and the other is provided by the APB bus (CLK_NVMCTRL_APB). For higher system frequencies, a programmable number of wait states can be used to optimize performance. When changing the AHB bus frequency, the user must ensure that the NVM Controller is configured with the proper number of wait states. Refer to the "Electrical Characteristics" on page 558 for the exact number of wait states to be used for a particular frequency range.

20.5.3 Interrupts

The NVM Controller interrupt request line is connected to the interrupt controller. Using the NVMCTRL interrupt requires the interrupt controller to be programmed first.

20.5.4 Debug Operation

When an external debugger forces the CPU into debug mode, the peripheral continues normal operation.

Access to the NVM block can be protected by the security bit. In this case, the NVM block will not be accessible. See "Security Bit" on page 260 for details.

20.5.5 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Status register (STATUS)

Write-protection is denoted by the Write-Protected property in the register description. Write-protection does not apply for accesses through an external debugger.

When the CPU is halted in debug mode, all write-protection is automatically disabled. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

20.5.6 Analog Connections

Not applicable.

20.6 Functional Description

20.6.1 Principle of Operation

The NVM Controller is a slave on the AHB and APB buses. It responds to commands, read requests and write requests, based on user configuration.

20.6.1.1 Initialization

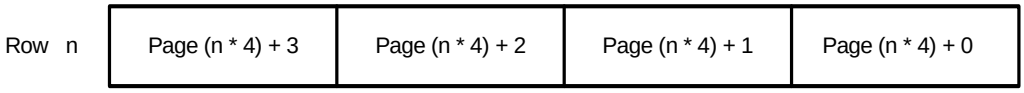
After power up, the NVM Controller goes through a power-up sequence. During this time, access to the NVM Controller from the AHB bus is halted. Upon power-up completion, the NVM Controller is operational without any need for user configuration.

20.6.2 Memory Organization

Refer to “Physical Memory Map” on page 20 for memory sizes and addresses for each device.

The NVM is organized into rows, where each row contains four pages, as shown in Figure 20-2. The NVM has a row-erase granularity, while the write granularity is by page. In other words, a single row erase will erase all four pages in the row, while four write operations are used to write the complete row.

Figure 20-2. Row Organization

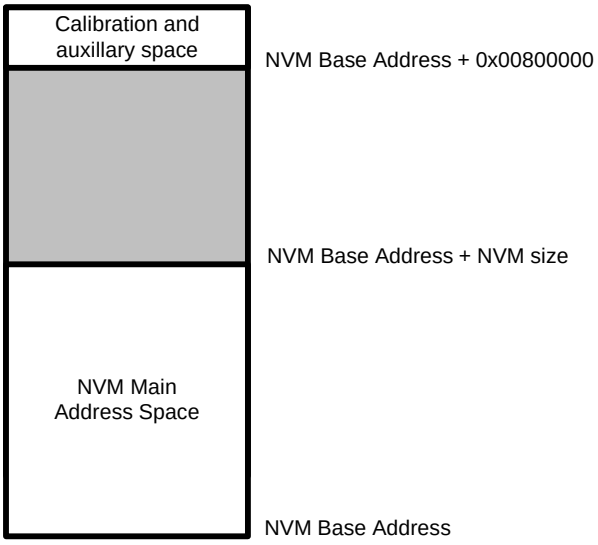


The NVM block contains a calibration and auxiliary space that is memory mapped. Refer to Figure 20-3 for details.

The calibration and auxiliary space contains factory calibration and system configuration information. This space can be read from the AHB bus in the same way as the main NVM main address space.

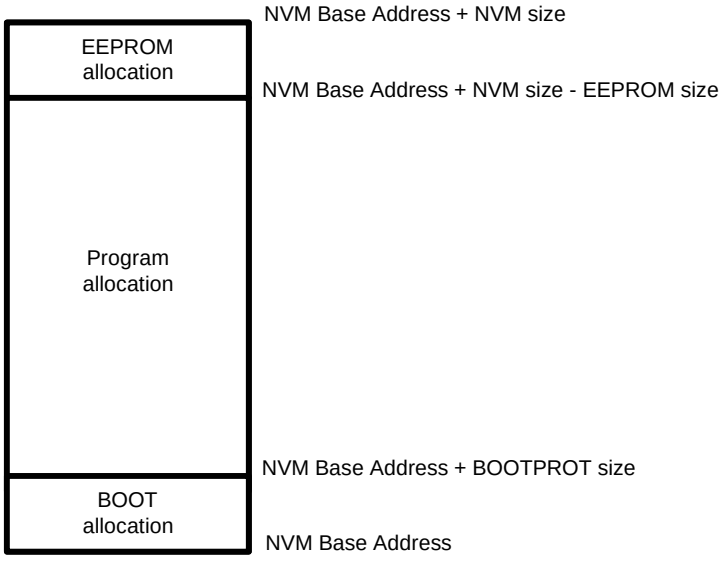
In addition, a boot loader section can be allocated at the beginning of the main array, and an EEPROM section can be allocated at the end of the NVM main address space.

Figure 20-3. NVM Memory Organization



The lower rows in the NVM main address space can be allocated as a boot loader section by using the BOOTPROT fuses, and the upper rows can be allocated to EEPROM, as shown in Figure 20-4. The boot loader section is protected by the lock bit(s) corresponding to this address space and by the BOOTPROT[2:0] fuse. The EEPROM rows can be written regardless of the region lock status. The number of rows protected by BOOTPROT and the number of rows allocated to the EEPROM are given in Table 20-2 and Table 20-3, respectively.

Figure 20-4. EEPROM and Boot LoaderAllocation



20.6.3 Region Lock Bits

The NVM block is grouped into 16 equally sized regions. The region size is dependent on the flash memory size, and is given in the table below. Each region has a dedicated lock bit preventing writing and erasing pages in the region. After production, all regions will be unlocked.

Table 20-1. Region Size

Memory Size [KB]	Region Size [KB]
256	16
128	8
64	4
32	2

To lock or unlock a region, the Lock Region and Unlock Region commands are provided. Writing one of these commands will temporarily lock/unlock the region containing the address loaded in the ADDR register. ADDR can be written by software, or the automatically loaded value from a write operation can be used. The new setting will stay in effect until the next reset, or the setting can be changed again using the lock and unlock commands. The current status of the lock can be determined by reading the LOCK register.

To change the default lock/unlock setting for a region, the user configuration section of the auxiliary space must be written using the Write Auxiliary Page command. Writing to the auxiliary space will take effect after the next reset. Therefore, a boot of the device is needed for changes in the lock/unlock setting to take effect. See “Physical Memory Map” on page 20 for calibration and auxiliary space address mapping.

20.6.4 Command and Data Interface

The NVM Controller is addressable from the APB bus, while the NVM main address space is addressable from the AHB bus. Read and automatic page write operations are performed by addressing the NVM main address space directly, while other operations such as manual page writes and row erase must be performed by issuing commands through the NVM Controller.

To issue a command, the CTRLA.CMD bits must be written along with the CTRLA.CMDEX value. When a command is issued, INTFLAG.READY will be cleared until the command has completed. Any commands written while INTFLAG.READY is low will be ignored. Read the CTRLA register description for more details.

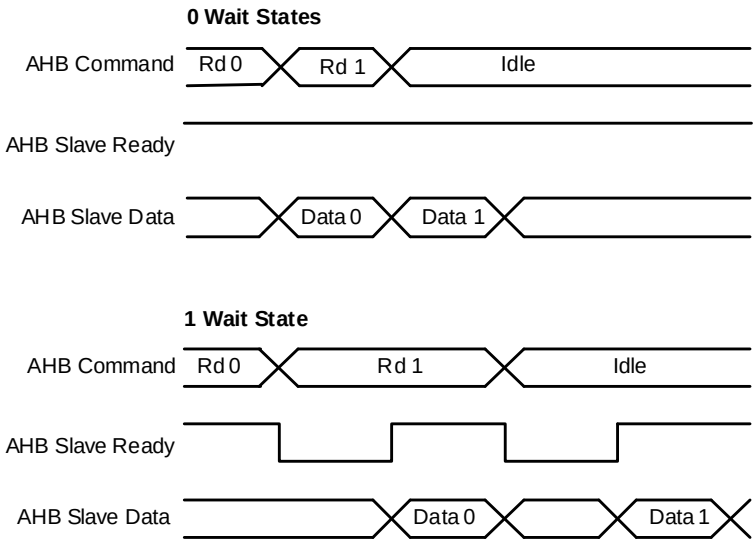
The CTRLB register must be used to control the power reduction mode, read wait states and the write mode.

20.6.4.1 NVM Read

Reading from the NVM main address space is performed via the AHB bus by addressing the NVM main address space or auxiliary address space directly. Read data is available after the configured number of read wait states (CTRLB.RWS) set in the NVM Controller.

The number of cycles data are delayed to the AHB bus is determined by the read wait states. Examples of using zero and one wait states are shown in Figure 20-5.

Figure 20-5. Read Wait State Examples



20.6.4.2 NVM Write

The NVM Controller requires that an erase must be done before programming. The entire NVM main address space can be erased by a debugger Chip Erase command. Alternatively, rows can be individually erased by the Erase Row command.

After programming, the region that the page resides in can be locked to prevent spurious write or erase sequences. Locking is performed on a per-region basis, and so locking a region locks all pages inside the region.

Data to be written to the NVM block are first written and stored in an internal buffer called the page buffer. The page buffer contains the same number of bytes as an NVM page. Writes to the page buffer must be 16 or 32 bits. 8-bit writes to the page buffer is not allowed, and will cause a system exception.

Writing to the NVM block via the AHB bus is performed by a load operation to the page buffer. For each AHB bus write, the address is stored in the ADDR register. After the page buffer has been loaded with the required number of bytes, the page can be written to the addressed location by setting CMD to Write Page and setting the key value to CMDEX. The LOAD bit in the STATUS register indicates whether the page buffer has been loaded or not. Before writing the page to memory, the accessed row must be erased.

By default, automatic page writes are enabled (MANW=0). This will trigger a write operation to the page addressed by ADDR when the last location of the page is written.

Because the address is automatically stored in ADDR during the I/O bus write operation, the last given address will be present in the ADDR register. There is no need to load the ADDR register manually, unless a different page in memory is to be written.

Procedure for Manual Page Writes (MANW=1)

The row to be written must be erased before the write command is given.

- Write to the page buffer by addressing the NVM main address space directly
- Write the page buffer to memory: CMD=Write Page and CMDEX
- The READY bit in the INTFLAG register will be low while programming is in progress, and access through the AHB will be stalled

Procedure for Automatic Page Writes (MANW=0)

The row to be written must be erased before the last write to the page buffer is performed.

Note that partially written pages must be written with a manual write.

- Write to the page buffer by addressing the NVM main address space directly.
  - When the last location in the page buffer is written, the page is automatically written to NVM main address space.
- INTFLAG.READY will be zero while programming is in progress and access through the AHB will be stalled.

20.6.4.3 Page Buffer Clear

The page buffer is automatically cleared to all ones after a page write is performed. If a partial page has been written and it is desired to clear the contents of the page buffer, the Page Buffer Clear command can be used.

20.6.4.4 Erase Row

Before a page can be written, the row that contains the page must be erased. The Erase Row command can be used to erase the desired row. Erasing the row sets all bits to one. If the row resides in a region that is locked, the erase will not be performed and the Lock Error bit in the Status register (STATUS.LOCKE) will be set.

Procedure for Erase Row

- Write the address of the row to erase ADDR. Any address within the row can be used.
- Issue an Erase Row command.

20.6.4.5 Lock and Unlock Region

These commands are used to lock and unlock regions as detailed in section “Region Lock Bits” on page 257.

20.6.4.6 Set and Clear Power Reduction Mode

The NVM Controller and block can be taken in and out of power reduction mode through the set and clear power reduction mode commands. When the NVM Controller and block are in power reduction mode, the Power Reduction Mode bit in the Status register (STATUS.PRM) is set.

20.6.5 NVM User Configuration

The NVM user configuration resides in the auxiliary space. See “Physical Memory Map” on page 20 for calibration and auxiliary space address mapping.

The bootloader resides in the main array starting at offset zero. The allocated boot loader section is protected against write.

Table 20-2. Boot Loader Size

BOOTPROT [2:0]	Rows Protected by BOOTPROT	Boot Loader Size in Bytes
7	None	0
6	2	512
5	4	1024
4	8	2048

BOOTPROT [2:0]	Rows Protected by BOOTPROT	Boot Loader Size in Bytes
3	16	4096
2	32	8192
1	64	16384
0	128	32768

The EEPROM bits indicates the Flash size reserved for EEPROM emulation according to the Table 20-3. EEPROM resides in the upper rows of the NVM main address space and are writable, regardless of the region lock status.

Table 20-3. Flash size for EEPROM emulation

EEPROM[2:0]	Rows Allocated to EEPROM	EEPROM Size in Bytes for EEPROM emulation(1)
7	None	0
6	1	256
5	2	512
4	4	1024
3	8	2048
2	16	4096
1	32	8192
0	64	16384

Note: 1. the actual size of the EEPROM depends on the emulation software. For more information see Application Note AT03265

20.6.6 Security Bit

The security bit allows the entire chip to be locked from external access for code security. The security bit can be written by a dedicated command, Set Security Bit (SSB). Once set, the only way to clear the security bit is through a debugger Chip Erase command. After issuing the SSB command, the PROGE error bit can be checked. Refer to “DSU – Device Service Unit” on page 36 for details.

20.6.7 Cache

The NVM Controller cache reduces the device power consumption and improves system performance when wait states are required. It is a direct-mapped cache that implements 8 lines of 64 bits (i.e., 64 bytes). NVM Controller cache can be enabled by writing a zero in the CACHEDIS bit in the CTRLB register (CTRLB.CACHEDIS). Cache can be configured to three different modes using the READMODE bit group in the CTRLB register. Refer to CTRLB register description for more details. The INVALL command can be issued through the CTRLA register to invalidate all cache lines. Commands affecting NVM content automatically invalidate cache lines.

20.7 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0		CMD[6:0]						
0x01		15:8	CMDEX[7:0]							
0x2	Reserved									
0x3	Reserved									
0x04	CTRLB	7:0	MANW			RWS[3:0]				
0x05		15:8							SLEEPPRM[1:0]	
0x06		23:16						CACHEDIS	READMODE[1:0]	
0x07		31:24								
0x08	PARAM	7:0	NVMP[7:0]							
0x09		15:8	NVMP[15:8]							
0x0A		23:16						PSZ[2:0]		
0x0B		31:24								
0x0C	INTENCLR	7:0							ERROR	READY
0x0D	Reserved									
0x0E	Reserved									
0x0F	Reserved									
0x10	INTENSET	7:0							ERROR	READY
0x11	Reserved									
0x12	Reserved									
0x13	Reserved									
0x14	INTFLAG	7:0							ERROR	READY
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	STATUS	7:0				NVME	LOCKE	PROGE	LOAD	PRM
0x19		15:8								SB
0x1A	Reserved									
0x1B	Reserved									
0x1C	ADDR	7:0	ADDR[7:0]							
0x1D		15:8	ADDR[15:8]							
0x1E		23:16			ADDR[21:16]					
0x1F		31:24								
0x20	LOCK	7:0	LOCK[7:0]							
0x21		15:8	LOCK[15:8]							
0x22	Reserved									
0x23	Reserved									

20.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to the “Register Access Protection” on page 255 and the “PAC – Peripheral Access Controller” on page 27 for details.

20.8.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x0000

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
	CMDEX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		CMD[6:0]						
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:8 – CMDEX: Command Execution**
  This bit group should be written with the key value 0xA5 to enable the command written to CMD to be executed. If the bit group is written with a different key value, the write is not performed and the PROGE status bit is set. PROGE is also set if the a previously written command is not complete.

  The key value must be written at the same time as CMD. If a command is issued through the APB bus on the same cycle as an AHB bus access, the AHB bus access will be given priority. The command will then be executed when the NVM block and the AHB bus are idle.

  The READY status must be one when the command is issued.

  Bit 0 of the CMDEX bit group will read back as one until the command is issued.

- **Bit 7 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 6:0 – CMD: Command**
  These bits define the command to be executed when the CMDEX key is written, as shown in Table 20-4.

Table 20-4. Command Bit Description

CMD[4:0]	Group Configuration	Description
0x00-0x01	-	Reserved
0x02	ER	Erase Row - Erases the row addressed by the ADDR register.
0x03	-	Reserved
0x04	WP	Write Page - Writes the contents of the page buffer to the page addressed by the ADDR register.
0x05	EAR	Erase Auxiliary Row - Erases the auxiliary row addressed by the ADDR register. This command can be given only when the security bit is not set and only to the user configuration row.

Table 20-4. Command Bit Description (Continued)

CMD[4:0]	Group Configuration	Description
0x06	WAP	Write Auxiliary Page - Writes the contents of the page buffer to the page addressed by the ADDR register. This command can be given only when the security bit is not set and only to the user configuration row.
0x07-0x3F	-	Reserved
0x40	LR	Lock Region - Locks the region containing the address location in the ADDR register.
0x41	UR	Unlock Region - Unlocks the region containing the address location in the ADDR register.
0x42	SPRM	Sets the power reduction mode.
0x43	CPRM	Clears the power reduction mode.
0x44	PBC	Page Buffer Clear - Clears the page buffer.
0x45	SSB	Set Security Bit - Sets the security bit by writing 0x00 to the first byte in the lockbit row.
0x46	INVALL	Invalidates all cache lines.
0x46-0x7F	-	Reserved

20.8.2 Control B

Name: CTRLB

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
						CACHEDIS	READMODE[1:0]	
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
							SLEEPPRM[1:0]	
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MANW			RWS[3:0]				
Access	R/W	R	R	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:19 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 18 – CACHEDIS: Cache Disable**
  This bit is used to disable the cache.
  0: The cache is enabled.
  1: The cache is disabled.

- **Bits 17:16 – READMODE: NVMCTRL Read Mode**

READMODE	Name	Description
0x0	NO_MISS_PENALTY	The NVM Controller (cache system) does not insert wait states on a cache miss. Gives the best system performance.
0x1	LOW_POWER	Reduces power consumption of the cache system, but inserts a wait state each time there is a cache miss. This mode may not be relevant if CPU performance is required, as the application will be stalled and may lead to increase run time.
0x2	DETERMINISTIC	The cache system ensures that a cache hit or miss takes the same amount of time, determined by the number of programmed flash wait states. This mode can be used for real-time applications that require deterministic execution timings.
0x3	Reserved	

- **Bits 15:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 9:8 – SLEEPPRM: Power Reduction Mode during Sleep**
  Indicates the power reduction mode during sleep.

Table 20-5. Table 1-7. Power Reduction Mode during Sleep

SLEEPPRM[1:0]	Name	Description
0x0	WAKEONACCESS	NVM block enters low-power mode when entering sleep. NVM block exits low-power mode upon first access.
0x1	WAKEUPINSTANT	NVM block enters low-power mode when entering sleep. NVM block exits low-power mode when exiting sleep.
0x2	Reserved	
0x3	DISABLED	Auto power reduction disabled.

- **Bit 7 – MANW: Manual Write**
  0: Writing to the last word in the page buffer will initiate a write operation to the page addressed by the last write operation. This includes writes to memory and auxiliary rows.
  1: Write commands must be issued through the CMD register.

- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 4:1 – RWS: NVM Read Wait States**
  These bits give the number of wait states for a read operation. Zero indicates zero wait states, one indicates one wait state, etc., up to 15 wait states.
  This register is initialized to 0 wait states. Software can change this value based on the NVM access time and system frequency.

- **Bit 0 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

20.8.3 NVM Parameter

Name: PARAM

Offset: 0x08

Reset: 0x000XXXXX

Property: –

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
						PSZ[2:0]		
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	X	X	X

Bit	15	14	13	12	11	10	9	8
	NVMP[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

Bit	7	6	5	4	3	2	1	0
	NVMP[7:0]							
Access	R	R	R	R	R		R	R
Reset	X	X	X	X	X	X	X	X

- **Bits 31:19 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 18:16 – PSZ: Page Size**
  Indicates the page size. Not all device families will provide all the page sizes indicated in the table.

Table 20-6. Page Size

PSZ[2:0]	Name	Description
0x0	8	8 bytes
0x1	16	16 bytes
0x2	32	32 bytes
0x3	64	64 bytes
0x4	128	128 bytes
0x5	256	256 bytes
0x6	512	512 bytes
0x7	1024	1024 bytes

- **Bits 15:0 – NVMP: NVM Pages**
  Indicates the number of pages in the NVM main address space.

20.8.4 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							ERROR	**READY**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – ERROR: Error Interrupt Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the ERROR interrupt enable.
  This bit will read as the current value of the ERROR interrupt enable.

- **Bit 0 – READY: NVM Ready Interrupt Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the READY interrupt enable.
  This bit will read as the current value of the READY interrupt enable.

20.8.5 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x10

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							ERROR	**READY**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – ERROR: Error Interrupt Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit sets the ERROR interrupt enable.
  This bit will read as the current value of the ERROR interrupt enable.

- **Bit 0 – READY: NVM Ready Interrupt Enable**
  Writing a zero to this bit has no effect.
  Writing a one to this bit sets the READY interrupt enable.
  This bit will read as the current value of the READY interrupt enable.

20.8.6 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x14

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
							ERROR	**READY**
Access	R	R	R	R	R	R	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – ERROR: Error**
  This flag is set on the occurrence of an NVME, LOCKE or PROGE error.
  0: No errors have been received since the last clear.
  1: At least one error has occurred since the last clear.
  This bit can be cleared by writing a one to its bit location.

- **Bit 0 – READY: NVM Ready**
  0: The NVM controller is busy programming or erasing.
  1: The NVM controller is ready to accept a new command.

20.8.7 Status

Name: STATUS

Offset: 0x18

Reset: 0x0X00

Property: –

Bit	15	14	13	12	11	10	9	8
								SB
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	X

Bit	7	6	5	4	3	2	1	0
				NVME	LOCKE	PROGE	LOAD	PRM
Access	R	R	R	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 15:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 8 – SB: Security Bit Status**
  0: The Security bit is inactive.
  1: The Security bit is active.
- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 4 – NVME: NVM Error**
  0: No programming or erase errors have been received from the NVM controller since this bit was last cleared.
  1: At least one error has been registered from the NVM Controller since this bit was last cleared.
  This bit can be cleared by writing a one to its bit location.
- **Bit 3 – LOCKE: Lock Error Status**
  0: No programming of any locked lock region has happened since this bit was last cleared.
  1: Programming of at least one locked lock region has happened since this bit was last cleared.
  This bit can be cleared by writing a one to its bit location.
- **Bit 2 – PROGE: Programming Error Status**
  0: No invalid commands or bad keywords were written in the NVM Command register since this bit was last cleared.
  1: An invalid command and/or a bad keyword was/were written in the NVM Command register since this bit was last cleared.
  This bit can be cleared by writing a one to its bit location.
- **Bit 1 – LOAD: NVM Page Buffer Active Loading**
  This bit indicates that the NVM page buffer has been loaded with one or more words. Immediately after an NVM load has been performed, this flag is set, and it remains set until a page write or a page buffer clear (PBCLR) command is given.

This bit can be cleared by writing a one to its bit location.

- **Bit 0 – PRM: Power Reduction Mode**
  This bit indicates the current NVM power reduction state. The NVM block can be set in power reduction mode in two ways: through the command interface or automatically when entering sleep with SLEEPPRM set accordingly.
  PRM can be cleared in three ways: through AHB access to the NVM block, through the command interface (SPRM and CPRM) or when exiting sleep with SLEEPPRM set accordingly.
  0: NVM is not in power reduction mode.
  1: NVM is in power reduction mode.

20.8.8 Address

Name: ADDR

Offset: 0x1C

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			ADDR[21:16]					
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 21:0 – ADDR: NVM Address**
  ADDR drives the hardware (16-bit) address to the NVM when a command is executed using CMDEX. 8-bit addresses must be shifted one bit to the right before writing to this register.

  This register is automatically updated when writing to the page buffer, and can also be manually written. This register holds the address offset for the section addressed.

20.8.9 Lock Section

Name: LOCK

Offset: 0x20

Reset: 0xXXXX

Property: –

Bit	15	14	13	12	11	10	9	8
	LOCK[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

Bit	7	6	5	4	3	2	1	0
	LOCK[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	X	X	X	X	X	X	X	X

- **Bits 15:0 – LOCK: Region Lock Bits**
  In order to set or clear these bits, the CMD register must be used.
  0: The corresponding lock region is locked.
  1: The corresponding lock region is not locked.

21. PORT

21.1 Overview

The Port (PORT) controls the I/O pins of the microcontroller. The I/O pins are organized in a series of groups, collectively referred to as a line bundle, and each group can have up to 32 pins that can be configured and controlled individually or as a group. Each pin may either be used for general-purpose I/O under direct application control or assigned to an embedded device peripheral. When used for general-purpose I/O, each pin can be configured as input or output, with highly configurable driver and pull settings.

All I/O pins have true read-modify-write functionality when used for general-purpose I/O; the direction or the output value of one or more pins may be changed (set, reset or toggled) without unintentionally changing the state of any other pins in the same line bundle via a single, atomic 8-, 16- or 32-bit write.

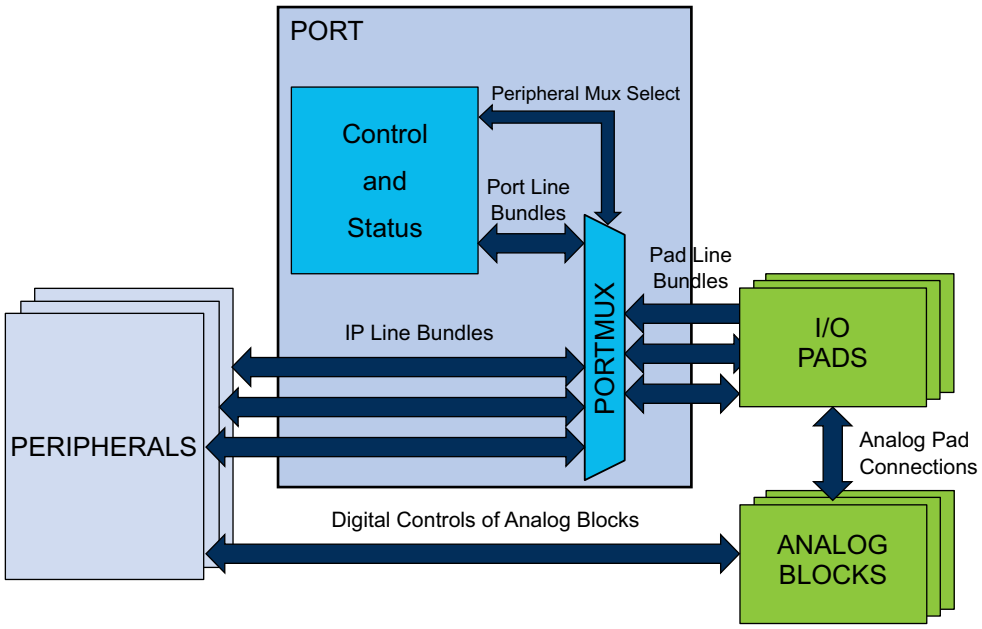
The PORT is connected to the high-speed bus matrix through an AHB/APB bridge. The Pin Direction, Data Output Value and Data Input Value registers may also be accessed using the low-latency CPU local bus (IOBUS; ARM® single-cycle I/O port).

21.2 Features

- Selectable input and output configuration individually for each pin
- Software-controlled multiplexing of peripheral functions on I/O pins
- Flexible pin configuration through a dedicated Pin Configuration register
- Configurable output driver and pull settings:
  - Totem-pole (push-pull)
  - Pull configuration
- Configurable input buffer and pull settings:
  - Internal pull-up or pull-down
  - Input sampling criteria
  - Input buffer can be disabled if not needed for lower power consumption
- Read-modify-write support for pin configuration, output value and pin direction

21.3 Block Diagram

Figure 21-1. PORT Block Diagram



21.4 Signal Description

Signal Name	Type	Description
Pxy	Digital I/O	General-purpose I/O pin y

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

21.5 Product Dependencies

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

21.5.1 I/O Lines

The I/O lines of the PORT are mapped to pins of the physical device package according to a simple naming scheme. Each line bundle of up to 32 pins is assigned a letter identifier, starting with A, that monotonically increases through the alphabet for each subsequent line bundle. Within each line bundle, each pin is assigned a numerical identifier according to its bit position.

The resulting PORT pins are mapped as Pxy, where x=A, B, C,… and y=00, 01, …, 31 to uniquely identify each pin in the device, e.g., PA24, PC03, etc.

Each pin may have one or more peripheral multiplexer settings, which allow the pad to be routed internally to a dedicated peripheral function. When enabled, the selected peripheral is given control over the output state of the pad, as well as the ability to read the current physical pad state. Refer to "I/O Multiplexing and Considerations" on page 11 for details.

Device-specific configurations may result in some pins (and the corresponding Pxy pin) not being implemented.

21.5.2 Power Management

During reset, all PORT lines are configured as inputs with input buffers, output buffers and pull disabled.

If the PORT peripheral is shut down, the latches contained in the pad will retain their current configuration, such as the output value and pull settings. However, the PORT configuration registers and input synchronizers will lose their contents, and these will not be restored when PORT is powered up again. The user must, therefore, reconfigure the PORT peripheral at power up to ensure it is in a well-defined state before use.

The PORT will continue to operate in any sleep mode where the selected module source clock is running.

21.5.3 Clocks

The PORT bus clock (CLK_PORT_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_PORT_APB can be found in the Peripheral Clock Masking section in the “PM – Power Manager” on page 100.

The PORT is fed by two different clocks: a CPU main clock, which allows the CPU to access the PORT through the low-latency CPU local bus (IOBUS), and an APB clock, which is a divided clock of the CPU main clock and allows the CPU to acces the PORT registers through the high-speed matrix and the AHB/APB bridge.

IOBUS accesses have priority over APB accesses. The latter must insert wait states in the event of concurrent PORT accesses.

The PORT input synchronizers use the CPU main clock so that the resynchronization delay is minimized with respect to the APB clock.

21.5.4 DMA

Not applicable.

21.5.5 Interrupts

Not applicable.

21.5.6 Events

Not applicable.

21.5.7 Debug Operation

When the CPU is halted in debug mode, the PORT continues normal operation. If the PORT 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.

21.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC).

Write-protection is denoted by the Write-Protected property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

21.5.9 Analog Connections

Analog functions are connected directly between the analog blocks and the I/O pads using analog buses. However, selecting an analog peripheral function for a given pin will disable the corresponding digital features of the pad.

21.5.10 CPU Local Bus

The CPU local bus (IOBUS) is an interface that connects the CPU directly to the PORT. It is a single-cycle bus interface, and does not support wait states. It supports byte, half word and word sizes.

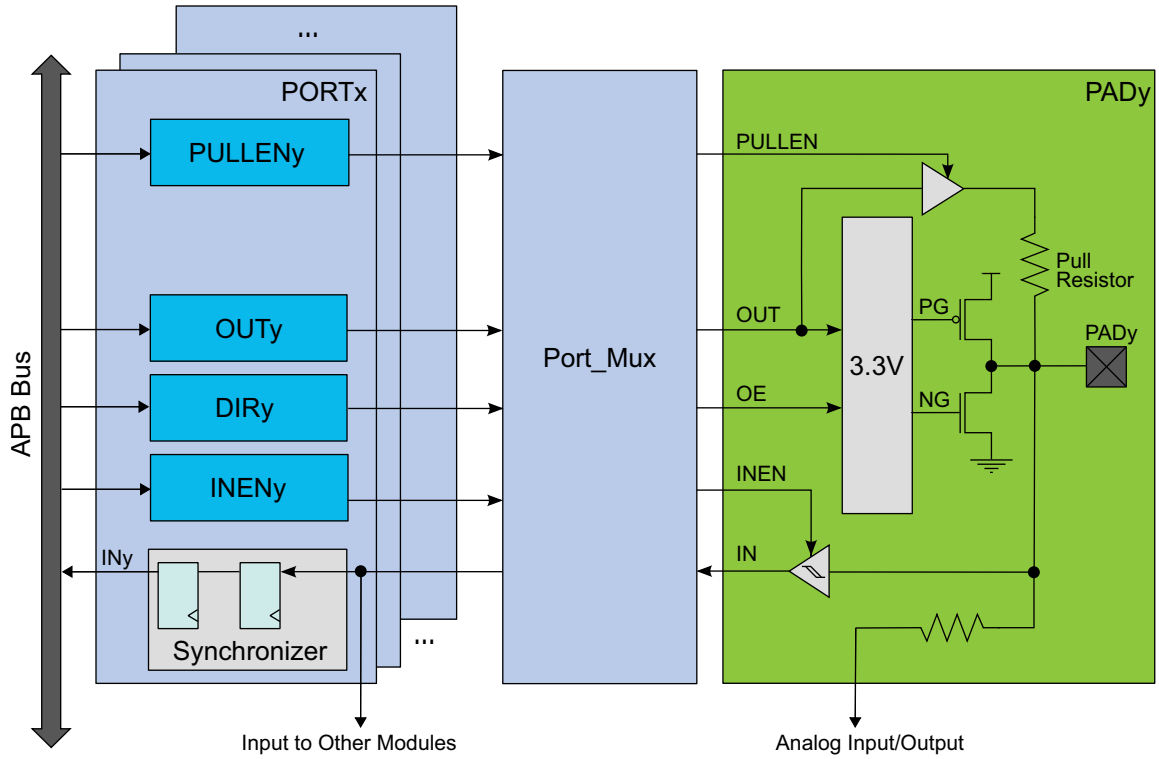
The CPU accesses the PORT module through the IOBUS when it performs read or write from address 0x60000000. The PORT register map is equivalent to the one described in the register description section.

This bus is generally used for low latency. The Data Direction (DIR) and Data Output Value (OUT) registers can be read, written, set, cleared or toggled using this bus, and the Data Input Value (IN) registers can be read.

Since the IOBUS cannot wait for IN register resynchronization, the Control register (CTRL) must be configured to enable continuous sampling of all pins that will need to be read via the IOBUS to prevent stale data from being read.

21.6 Functional Description

Figure 21-2. Overview of the PORT



21.6.1 Principle of Operation

The I/O pins of the device are controlled by reads and writes of the PORT peripheral registers. For each port pin, a corresponding bit in the Data Direction (DIR) and Data Output Value (OUT) registers are used to enable that pin as an output and to define the output state.

The direction of each pin in a port bundle is configured via the DIR register. If a bit in DIR is written to one, the corresponding pin is configured as an output pin. If a bit in DIR is written to zero, the corresponding pin is configured as an input pin.

When the direction is set as output, the corresponding bit in the OUT register is used to set the level of the pin. If bit y of OUT is written to one, pin y is driven high. If bit y of OUT is written to zero, pin y is driven low.

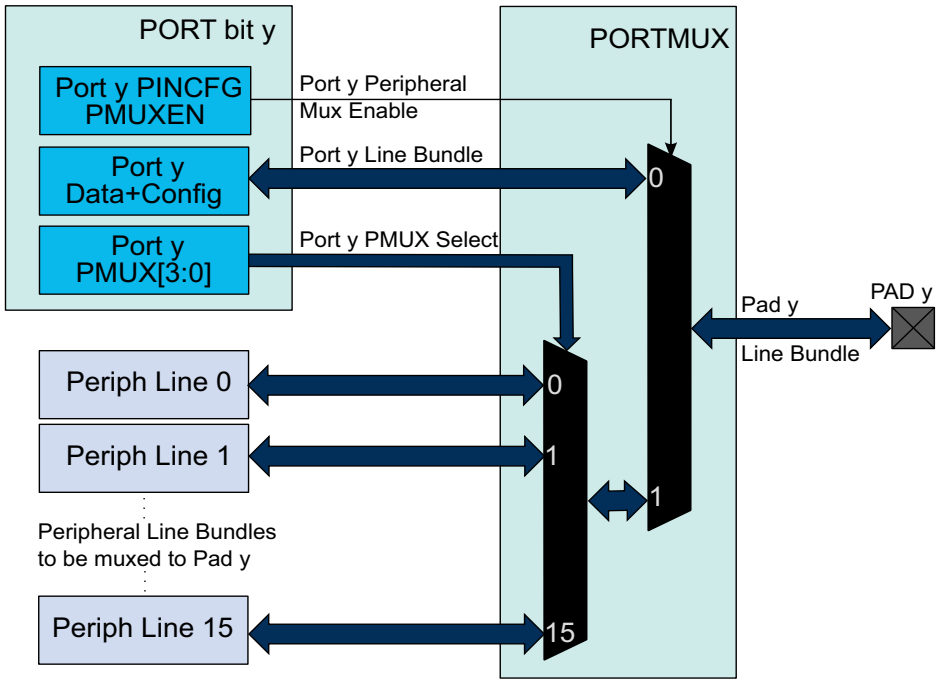
Additional pin configuration can be set by writing to the Pin Configuration (PINCFGy) registers.

The Data Input Value bit (IN) is used to read the port pin with resynchronization to the PORT clock. By default, these input synchronizers are clocked only when an input value read is requested in order to reduce power consumption. Input value can always be read, whether the pin is configured as input or output, except if digital input is disabled by writing a zero to the INEN bit in the Pin Configuration registers (PINCFGy).

The PORT also allows peripheral functions to be connected to individual I/O pins by writing a one to the corresponding PMUXEN bit in the PINCFGy registers and by writing the chosen selection to the Peripheral Multiplexing registers (PMUXn) for that pin. This will override the connection between the PORT and that I/O pin, and connect the selected peripheral line bundle to the pad instead of the PORT line bundle.

Each group of up to 32 pins is controlled by a set of registers, as described in Figure 21-3. This set of registers is duplicated for each group of pins, with increasing base addresses.

Figure 21-3. Overview of the Peripheral Functions Multiplexing



21.6.2 Basic Operation

21.6.2.1 Initialization

After reset, all standard-function device I/O pads are connected to the PORT with outputs tri-stated and input buffers disabled, even if no clocks are running. Specific pins, such as the ones used for connection to a debugger, may be configured differently, as required by their special function.

21.6.3 Basic Operation

Each I/O pin y can be configured and accessed by reading or writing PORT registers. Because PORT registers are grouped into sets of registers for each group of up to 32 pins, the base address of the register set for pin y is at byte address PORT + (y / 32) * 0x80. (y / 32) will be used as the index within that register set.

To use pin y as an output, configure it as output by writing the (y / 32) bit in the DIR register to one. To avoid disturbing the configuration of other pins in that group, this can also be done by writing the (y / 32) bit in the DIRSET register to one. The desired output value can be set by writing the (y / 32) bit to that value in register OUT.

Similarly, writing an OUTSET bit to one will set the corresponding bit in the OUT register to one, while writing an OUTCLR bit to one will set it to zero, and writing an OUTTGL bit to one will toggle that bit in OUT.

To use pin y as an input, configure it as input by writing the (y / 32) bit in the DIR register to zero. To avoid disturbing the configuration of other pins in that group, this can also be done by writing the (y / 32) bit in DIRCLR register to one. The desired input value can be read from the (y / 32) bit in register IN as soon as the INEN bit in the Pin Configuration register (PINCFGy) is written to one. Refer to “I/O Multiplexing and Considerations” on page 11 for details on pin configuration.

By default, the input synchronizer is clocked only when an input read is requested, which will delay the read operation by two CLK_PORT cycles. To remove that delay, the input synchronizers for each group of eight pins can be configured to be always active, but this comes at the expense of higher power consumption. This is controlled by writing a one to the corresponding SAMPLINGn bit group of the CTRL register, where n = (y / 32) / 8.

To use pin y as one of the available peripheral functions for that pin, configure it by writing a one to the corresponding PMUXEN bit of the PINCFGy register. The PINCFGy register for pin y is at byte offset (PINCFG0 + (y / 32)).

The peripheral function can be selected by writing to the PMUXO or PMUXE bit group in the PMUXn register. The PMUXO/PMUXE bit group is at byte offset (PMUX0 + (y / 32) / 2), in bits 3:0 if y is even and in bits 7:4 if y is odd.

The chosen peripheral must also be configured and enabled.

21.6.4 I/O Pin Configuration

The Pin Configuration register (PINCFGy) is used for additional I/O pin configuration. A pin can be set in a totem-pole, open-drain or pull configuration.

Because pull configuration is done through the Pin Configuration register, all intermediate PORT states during switching of pin direction and pin values are avoided.

The I/O pin configurations are described further in this chapter, and summarized in Table 21-1.

21.6.4.1 Pin Configurations Summary

Table 21-1. Pin Configurations Summary

DIR	INEN	PULLEN	OUT	Configuration
0	0	0	X	Reset or analog I/O; all digital disabled
0	0	1	0	Pull-down; input disabled
0	0	1	1	Pull-up; input disabled
0	1	0	X	Input
0	1	1	0	Input with pull-down
0	1	1	1	Input with pull-up
1	0	X	X	Output; input disabled
1	1	X	X	Output; input enabled

21.6.4.2 Input Configuration

Figure 21-4. I/O Configuration - Standard Input

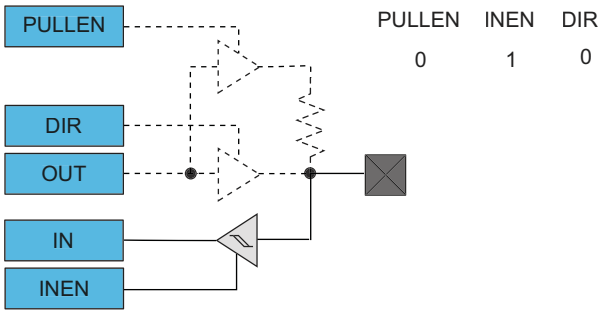
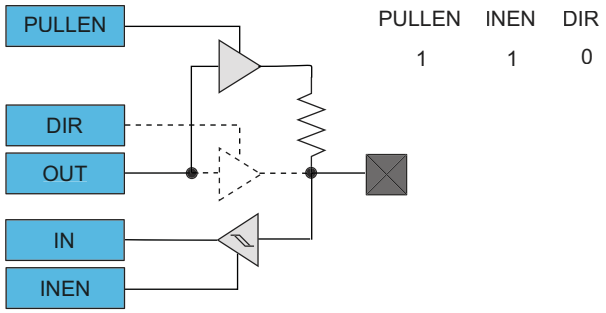


Figure 21-5. I/O Configuration - Input with Pull



Note that when pull is enabled, the pull value is defined by the OUTx value.

21.6.4.3 Totem-Pole Output

When configured for totem-pole (push-pull) output, the pin is driven low or high according to the corresponding bit setting in the OUT register. In this configuration, there is no current limitation for sink or source other than what the pin is capable of. If the pin is configured for input, the pin will float if no external pull is connected. Note, that enabling the output driver automatically disables pull.

Figure 21-6. I/O Configuration - Totem-Pole Output with Disabled Input

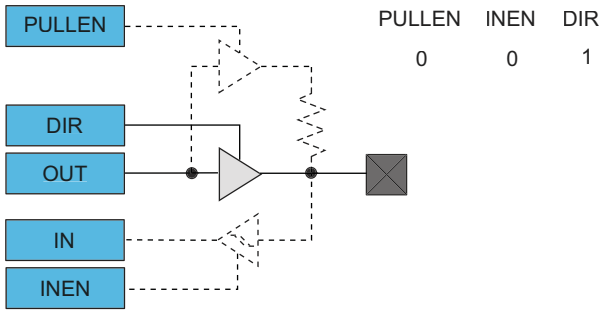


Figure 21-7. I/O Configuration - Totem-Pole Output with Enabled Input

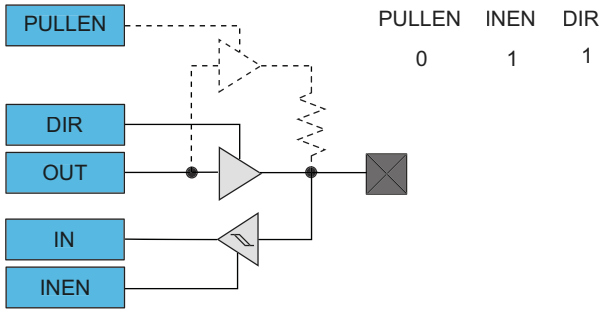
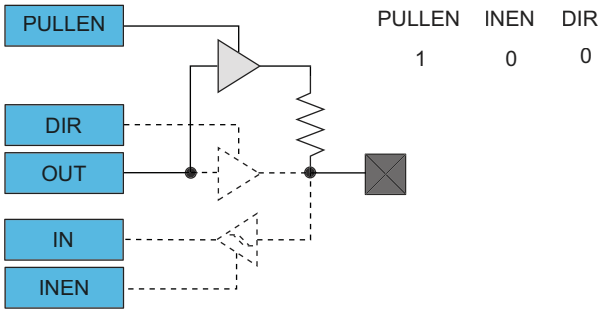
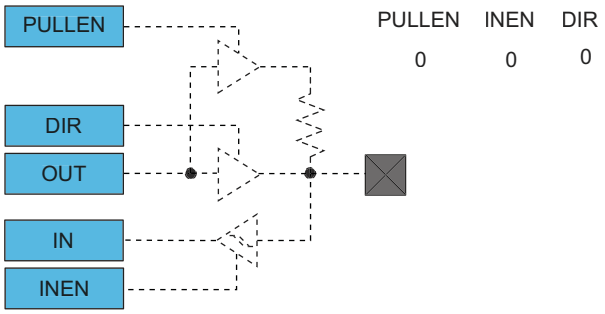


Figure 21-8. I/O Configuration - Output with Pull



21.6.4.4 Digital Functionality Disabled

Figure 21-9. I/O Configuration - Reset or Analog I/O: Digital Output, Input and Pull Disabled



21.7 Register Summary

The I/O pins are organized in groups with up to 32 pins. Group 0 consists of the PA pins, group 1 the PB pins, etc. Each group has its own set of registers. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, while the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

Offset	Name	Bit Pos.							
0x00	DIR	7:0	DIR[7:0]						
0x01		15:8	DIR[15:8]						
0x02		23:16	DIR[23:16]						
0x03		31:24	DIR[31:24]						
0x04	DIRCLR	7:0	DIRCLR[7:0]						
0x05		15:8	DIRCLR[15:8]						
0x06		23:16	DIRCLR[23:16]						
0x07		31:24	DIRCLR[31:24]						
0x08	DIRSET	7:0	DIRSET[7:0]						
0x09		15:8	DIRSET[15:8]						
0x0A		23:16	DIRSET[23:16]						
0x0B		31:24	DIRSET[31:24]						
0x0C	DIRTGL	7:0	DIRTGL[7:0]						
0x0D		15:8	DIRTGL[15:8]						
0x0E		23:16	DIRTGL[23:16]						
0x0F		31:24	DIRTGL[31:24]						
0x10	OUT	7:0	OUT[7:0]						
0x11		15:8	OUT[15:8]						
0x12		23:16	OUT[23:16]						
0x13		31:24	OUT[31:24]						
0x14	OUTCLR	7:0	OUTCLR[7:0]						
0x15		15:8	OUTCLR[15:8]						
0x16		23:16	OUTCLR[23:16]						
0x17		31:24	OUTCLR[31:24]						
0x18	OUTSET	7:0	OUTSET[7:0]						
0x19		15:8	OUTSET[15:8]						
0x1A		23:16	OUTSET[23:16]						
0x1B		31:24	OUTSET[31:24]						
0x1C	OUTTGL	7:0	OUTTGL[7:0]						
0x1D		15:8	OUTTGL[15:8]						
0x1E		23:16	OUTTGL[23:16]						
0x1F		31:24	OUTTGL[31:24]						
0x20	IN	7:0	IN[7:0]						
0x21		15:8	IN[15:8]						
0x22		23:16	IN[23:16]						
0x23		31:24	IN[31:24]						
0x24	CTRL	7:0	SAMPLING[7:0]						
0x25		15:8	SAMPLING[15:8]						
0x26		23:16	SAMPLING[23:16]						
0x27		31:24	SAMPLING[31:24]						

Offset	Name	Bit Pos.								
0x28	WRCONFIG	7:0	PINMASK[7:0]							
0x29		15:8	PINMASK[15:8]							
0x2A		23:16						PULLEN	INEN	PMUXEN
0x2B		31:24	HWSEL	WRPINCFG		WRPMUX	PMUX[3:0]			
0x2C	Reserved									
0x2D										
0x2E										
0x2F										
0x30	PMUX0	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x31	PMUX1	7:0	PMUXO[3:0]				PMUXE[3:0]			
...	...	...								
0x3E	PMUX14	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x3F	PMUX15	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x40	PINCFG0	7:0						PULLEN	INEN	PMUXEN
0x41	PINCFG1	7:0						PULLEN	INEN	PMUXEN
...	...	...								
0x5E	PINCFG30	7:0						PULLEN	INEN	PMUXEN
0x5F	PINCFG31	7:0						PULLEN	INEN	PMUXEN

21.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 278 for details.

21.8.1 Data Direction

Name: DIR

Offset: 0x00+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	DIR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DIR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DIR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – DIR[31:0]: Port Data Direction**
  These bits set the data direction for the individual I/O pins in the PORT group.
  0: The corresponding I/O pin in the group is configured as an input.
  1: The corresponding I/O pin in the group is configured as an output.

21.8.2 Data Direction Clear

This register allows the user to set one or more I/O pins as an input, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Direction (DIR), Data Direction Toggle (DIRTGL) and Data Direction Set (DIRSET) registers.

Name: DIRCLR

Offset: 0x04+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	DIRCLR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIRCLR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DIRCLR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DIRCLR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – DIRCLR[31:0]: Port Data Direction Clear**
  0: The I/O pin direction is cleared.
  1: The I/O pin direction is set.
  Writing a zero to a bit has no effect.
  Writing a one to a bit will clear the corresponding bit in the DIR register, which configures the I/O pin as an input.

21.8.3 Data Direction Set

This register allows the user to set one or more I/O pins as an output, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Direction (DIR), Data Direction Toggle (DIRTGL) and Data Direction Clear (DIRCLR) registers.

Name: DIRSET

Offset: 0x08+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	DIRSET[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIRSET[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DIRSET[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DIRSET[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – DIRSET[31:0]: Port Data Direction Set**
  0: The I/O pin direction is cleared.
  1: The I/O pin direction is set.
  Writing a zero to a bit has no effect.
  Writing a one to a bit will set the corresponding bit in the DIR register, which configures the I/O pin as an output.

21.8.4 Data Direction Toggle

This register allows the user to toggle the direction of one or more I/O pins, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Direction (DIR), Data Direction Set (DIRSET) and Data Direction Clear (DIRCLR) registers.

Name: DIRTGL

Offset: 0x0C+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	DIRTGL[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIRTGL[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DIRTGL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DIRTGL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – DIRTGL[31:0]: Port Data Direction Toggle**
  0: The I/O pin direction is cleared.
  1: The I/O pin direction is set.
  Writing a zero to a bit has no effect.
  Writing a one to a bit will toggle the corresponding bit in the DIR register, which reverses the direction of the I/O pin.

21.8.5 Data Output Value

This register sets the data output drive value for the individual I/O pins in the PORT.

Name: OUT

Offset: 0x10+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	OUT[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	OUT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	OUT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – OUT[31:0]: Port Data Output Value**
  These bits set the logical output drive level of I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via the Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will set the input pull direction.
  0: The I/O pin output is driven low, or the input is connected to an internal pull-down.
  1: The I/O pin output is driven high, or the input is connected to an internal pull-up.

21.8.6 Data Output Value Clear

This register allows the user to set one or more output I/O pin drive levels low, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Toggle (OUTTGL) and Data Output Value Set (OUTSET) registers.

Name: OUTCLR

Offset: 0x14+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	OUTCLR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	OUTCLR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	OUTCLR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUTCLR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – OUTCLR[31:0]: Port Data Output Value Clear**
  0: The I/O pin output is driven low.
  1: The I/O pin output is driven high.
  Writing a zero to a bit has no effect.
  Writing a one to a bit will clear the corresponding bit in the OUT register, which sets the output drive level low for I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via the Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will set the input pull direction to an internal pull-down.

21.8.7 Data Output Value Set

This register allows the user to set one or more output I/O pin drive levels high, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Toggle (OUTTGL) and Data Output Value Clear (OUTCLR) registers.

Name: OUTSET

Offset: 0x18+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	OUTSET[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	OUTSET[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	OUTSET[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUTSET[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – OUTSET[31:0]: Port Data Output Value Set**
  0: The I/O pin output is driven low.
  1: The I/O pin output is driven high.
  Writing a zero to a bit has no effect.
  Writing a one to a bit will set the corresponding bit in the OUT register, which sets the output drive level high for I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via the Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will set the input pull direction to an internal pull-up.

21.8.8 Data Output Value Toggle

This register allows the user to toggle the drive level of one or more output I/O pins, without doing a read-modify-write operation. Changes in this register will also be reflected in the Data Output Value (OUT), Data Output Value Set (OUTSET) and Data Output Value Clear (OUTCLR) registers.

Name: OUTTGL

Offset: 0x1C+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	OUTTGL[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	OUTTGL[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	OUTTGL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUTTGL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – OUTTGL[31:0]: Port Data Output Value Toggle**
  0: The I/O pin output is driven low.
  1: The I/O pin output is driven high.
  Writing a zero to a bit has no effect.
  Writing a one to a bit will toggle the corresponding bit in the OUT register, which inverts the output drive level for I/O pins configured as outputs via the Data Direction register (DIR). For pins configured as inputs via the Data Direction register (DIR) with pull enabled via the Pull Enable register (PULLEN), these bits will toggle the input pull direction.

21.8.9 Data Input Value

Name: IN

Offset: 0x20+x*0x80 [x=0..1]

Reset: 0x00000000

Property: -

Bit	31	30	29	28	27	26	25	24
	IN[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IN[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IN[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IN[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – IN[31:0]: Port Data Input Value**
  These bits are cleared when the corresponding I/O pin input sampler detects a logical low level on the input pin.
  These bits are set when the corresponding I/O pin input sampler detects a logical high level on the input pin.

21.8.10 Control

Name: CTRL

Offset: 0x24+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	SAMPLING[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SAMPLING[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SAMPLING[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SAMPLING[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – SAMPLING[31:0]: Input Sampling Mode**
  Configures the input sampling functionality of the I/O pin input samplers for pins configured as inputs via the Data Direction register (DIR).

  0: The I/O pin input synchronizer is disabled.

  1: The I/O pin input synchronizer is enabled.

  The input samplers are enabled and disabled in sub-groups of eight. Thus, if any pins within a byte request continuous sampling, all pins in that eight pin sub-group will be continuously sampled.

21.8.11 Write Configuration

This write-only register is used to configure several pins simultaneously with the same configuration and/or peripheral multiplexing.

In order to avoid the side effect of non-atomic access, 8-bit or 16-bit writes to this register will have no effect. Reading this register always returns zero.

Name: WRCONFIG

Offset: 0x28+x*0x80 [x=0..1]

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
	HWSEL	WRPINCFG		WRPMUX	PMUX[3:0]			
Access	W	W	R	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
						PULLEN	INEN	PMUXEN
Access	R	W	R	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PINMASK[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PINMASK[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

- **Bit 31 – HWSEL: Half-Word Select**
  This bit selects the half-word field of a 32-pin group to be reconfigured in the atomic write operation.
  0: The lower 16 pins of the PORT group will be configured.
  1: The upper 16 pins of the PORT group will be configured.
  This bit will always read as zero.

- **Bit 30 – WRPINCFG: Write PINCFG**
  This bit determines whether the atomic write operation will update the Pin Configuration register (PINCFGy) or not for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits.
  0: The PINCFGy registers of the selected pins will not be updated.
  1: The PINCFGy registers of the selected pins will be updated.
  Writing a zero to this bit has no effect.

Writing a one to this bit updates the configuration of the selected pins with the written WRCONFIG.PULLEN, WRCONFIG.INEN, WRCONFIG.PMUXEN and WRCONFIG.PINMASK values.

This bit will always read as zero.

- **Bit 29 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 28 – WRPMUX: Write PMUX**
  This bit determines whether the atomic write operation will update the Peripheral Multiplexing register (PMUXn) or not for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits.

  0: The PMUXn registers of the selected pins will not be updated.

  1: The PMUXn registers of the selected pins will be updated.

  Writing a zero to this bit has no effect.

  Writing a one to this bit updates the pin multiplexer configuration of the selected pins with the written WRCONFIG.PMUX value.

  This bit will always read as zero.

- **Bits 27:24 – PMUX[3:0]: Peripheral Multiplexing**
  These bits determine the new value written to the Peripheral Multiplexing register (PMUXn) for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPMUX bit is set.

  These bits will always read as zero.

- **Bits 23:19 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 18 – PULLEN: Pull Enable**
  This bit determines the new value written to PINCFGy.PULLEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits when the WRCONFIG.WRPINCFG bit is set.

  This bit will always read as zero.

- **Bit 17 – INEN: Input Enable**
  This bit determines the new value written to PINCFGy.DRVSTR for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits when the WRCONFIG.WRPINCFG bit is set.

  This bit will always read as zero.

- **Bit 16 – PMUXEN: Peripheral Multiplexer Enable**
  This bit determines the new value written to PINCFGy.PMUXEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits when the WRCONFIG.WRPINCFG bit is set.

  This bit will always read as zero.

- **Bits 15:0 – PINMASK[15:0]: Pin Mask for Multiple Pin Configuration**
  These bits select the pins to be configured within the half-word group selected by the WRCONFIG.HWSEL bit.

  0: The configuration of the corresponding I/O pin in the half-word group will be left unchanged.

  1: The configuration of the corresponding I/O pin in the half-word pin group will be updated.

  These bits will always read as zero.

21.8.12 Peripheral Multiplexing n

There are up to 16 Peripheral Multiplexing registers in each group, one for every set of two subsequent I/O lines. The n denotes the number of the set of I/O lines, while the m denotes the number of the group.

Name: PMUXn

Offset: 0x30+n*0x1+x*0x80 [n=0..15] [x=0..1]

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	PMUXO[3:0]				PMUXE[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:4 – PMUXO[3:0]: Peripheral Multiplexing Odd**
  These bits select the peripheral function for odd-numbered pins (2*n + 1) of a PORT group, if the corresponding PINCFGy.PMUXEN bit is one.
  Not all possible values for this selection may be valid. For more details, refer to “I/O Multiplexing and Considerations” on page 11.

- **Bits 3:0 – PMUXE[3:0]: Peripheral Multiplexing Even**
  These bits select the peripheral function for even-numbered pins (2*n) of a PORT group, if the corresponding PINCFGy.PMUXEN bit is one.
  Not all possible values for this selection may be valid. For more details, refer to “I/O Multiplexing and Considerations” on page 11.

Value	Name	Description
0x0	A	Peripheral function A selected
0x1	B	Peripheral function B selected
0x2	C	Peripheral function C selected
0x3	D	Peripheral function D selected
0x4	E	Peripheral function E selected
0x5	F	Peripheral function F selected
0x6	G	Peripheral function G selected
0x7	H	Peripheral function H selected
0x8-0xF	Reserved	

21.8.13 Pin Configuration y

There are up to 32 Pin Configuration registers in each group, one for each I/O line. The y denotes the number of the I/O line, while the x denotes the number of the group.

Name: PINCFGy

Offset: 0x40+y*0x1+x*0x80 [y=0..31] [x=0..1]

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						PULLEN	**INEN**	**PMUXEN**
Access	R	R/W	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – PULLEN: Pull Enable**
  This bit enables the internal pull-up or pull-down resistor of an I/O pin configured as an input.
  0: Internal pull resistor is disabled, and the input is in a high-impedance configuration.
  1: Internal pull resistor is enabled, and the input is driven to a defined logic level in the absence of external input.

- **Bit 1 – INEN: Input Enable**
  This bit controls the input buffer of an I/O pin configured as either an input or output.
  0: Input buffer for the I/O pin is disabled, and the input value will not be sampled.
  1: Input buffer for the I/O pin is enabled, and the input value will be sampled when required.
  Writing a zero to this bit disables the input buffer completely, preventing read-back of the physical pin state when the pin is configured as either an input or output.

- **Bit 0 – PMUXEN: Peripheral Multiplexer Enable**
  This bit enables or disables the peripheral multiplexer selection set in the Peripheral Multiplexing register (PMUXn) to enable or disable alternative peripheral control over an I/O pin direction and output drive value.
  0: The peripheral multiplexer selection is disabled, and the PORT registers control the direction and output drive value.
  1: The peripheral multiplexer selection is enabled, and the selected peripheral controls the direction and output drive value.
  Writing a zero to this bit allows the PORT to control the pad direction via the Data Direction register (DIR) and output drive value via the Data Output Value register (OUT). The peripheral multiplexer value in PMUXn is ignored.
  Writing a one to this bit enables the peripheral selection in PMUXn to control the pad. In this configuration, the physical pin state may still be read from the Data Input Value register (IN) if PINCFGy.INEN is set.

22. EVSYS – Event System

22.1 Overview

The Event System (EVSYS) allows autonomous, low-latency and configurable communication between peripherals.

Several peripherals can be configured to emit and/or respond to signals known as events. The exact condition to generate an event, or the action taken upon receiving an event, is specific to each module. Peripherals that respond to events are called event users. Peripherals that emit events are called event generators. A peripheral can have one or more event generators and can have one or more event users.

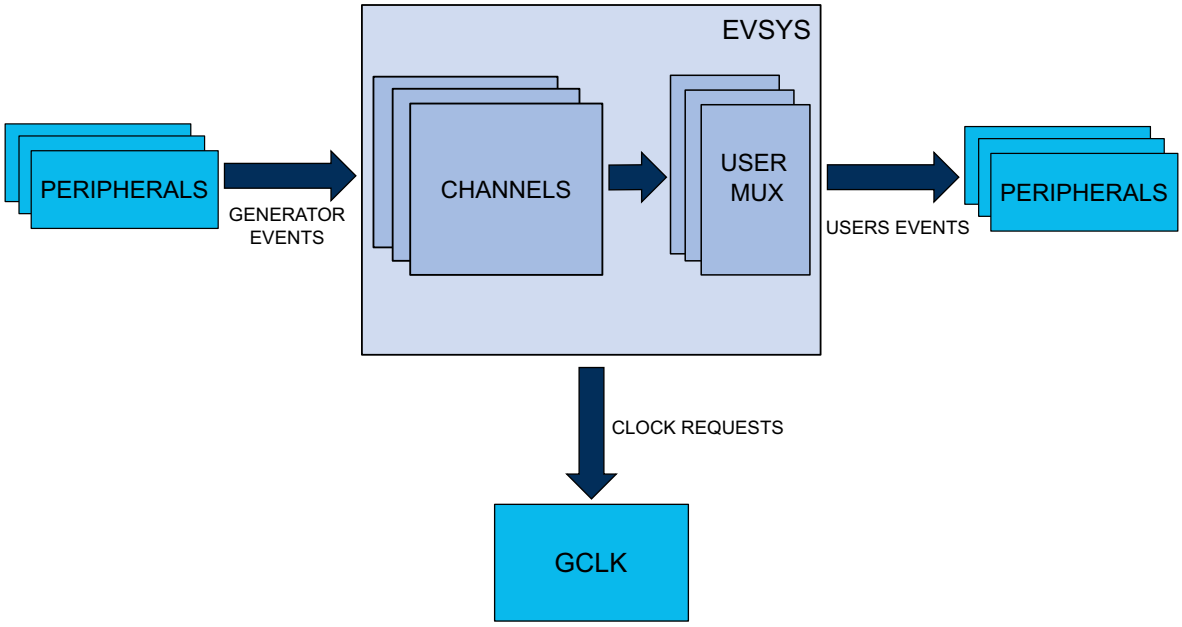
Communication is made without CPU intervention and without consuming system resources such as bus or RAM bandwidth. This reduces the load on the CPU and other system resources, compared to a traditional interrupt-based system.

22.2 Features

- System for direct peripheral-to-peripheral communication and signaling
- Eight configurable event channels, where each channel can:
  - Be connected to any event generator
  - Provide a pure asynchronous, resynchronized or synchronous path
- 58 event generators
- 14 event users
- Configurable edge detector
- Peripherals can be event generators, event users or both
- SleepWalking and interrupt for operation in low-power modes
- Software event generation
- Each event user can choose which channel to listen to

22.3 Block Diagram

Figure 22-1. Event System Block Diagram



22.4 Signal Description

Not applicable.

22.5 Product Dependencies

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

22.5.1 I/O Lines

Not applicable.

22.5.2 Power Management

The EVSYS can be used to wake up the CPU from all sleep modes, even if the clock used by the EVSYS channel and the EVSYS bus clock are disabled. Refer to “PM – Power Manager” on page 100 for details on the different sleep modes.

In all power save modes where the clock for the EVSYS is stopped, the device can wake up the EVSYS clock.

Some event generators can generate an event when the system clock is stopped. The generic clock (GCLK) for this channel will be restarted if the channel uses a synchronized path or a resynchronized path, without waking the system from sleep. The clock remains active only as long as necessary to handle the event. After the event has been handled, the clock will be turned off and the system will remain in the original sleep mode. This is known as SleepWalking. When an asynchronous path is used, there is no need for the clock to be activated for the event to be propagated to the user.

On a software reset, all registers are set to their reset values and any ongoing events are canceled.

22.5.3 Clocks

The EVSYS bus clock (CLK_EVSYS_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_EVSYS_APB can be found in the Peripheral Clock Masking section in “PM – Power Manager” on page 100.

Each EVSYS channel has a dedicated generic clock (GCLK_EVSYS_x). These are used for detection and propagation of events for each channel. These clocks must be configured and enabled in the generic clock controller before using the EVSYS. Refer to “GCLK – Generic Clock Controller” on page 78 for details.

22.5.4 DMA

Not applicable.

22.5.5 Interrupts

The interrupt request line is connected to the interrupt controller. Using the EVSYS interrupts requires the interrupt controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

22.5.6 Events

Not applicable.

22.5.7 Debug Operation

When the CPU is halted in debug mode, the EVSYS continues normal operation. If the EVSYS 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.

22.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following register:

- Interrupt Flag Status and Clear register (INTFLAG)

Write-protection is denoted by the Write-Protected property in the register description.

Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

22.5.9 Analog Connections

Not applicable.

22.6 Functional Description

22.6.1 Principle of Operation

Event users are connected to multiplexers that have all available event channels as input. The multiplexer must be configured to select one of these channels. The channels can be configured to route signals from any event generator, but cannot be connected to multiple event generators.

22.6.2 Basic Operation

22.6.2.1 Initialization

The peripheral that is to act as event generator should be configured to be able to generate events. The peripheral to act as event user should be configured to handle incoming events.

When this has been done, the event system is ready to be configured. The configuration must follow this order:

1. Configure the event user by performing a single 16-bit write to the User Multiplexer register (USER) with:
   1.1. The channel that is to be connected to a user written to the Channel bit group (USER.CHANNEL)
   1.2. The user to connect the channel to written to the User bit group (USER.USER)
2. Configure the channel by performing a single 32-bit write to the Channel (CHANNEL) register with:
   2.1. The channel to be configured written to the Channel Selection bit group (CHANNEL.CHANNEL)

2.2. The path to be used written to the Path Selection bit group (CHANNEL.PATH)
2.3. The type of edge detection to use on the channel written to the Edge Selection bit group (CHANNEL.EDGSEL)
2.4. The event generator to be used written to the Event Generator bit group (CHANNEL.EVGEN)

22.6.2.2 Enabling, Disabling and Resetting

The EVSYS is always enabled.

The EVSYS is reset by writing a one to the Software Reset bit in the Control register (CTRL.SWRST). All registers in the EVSYS will be reset to their initial state. Refer to the CTRL register for details.

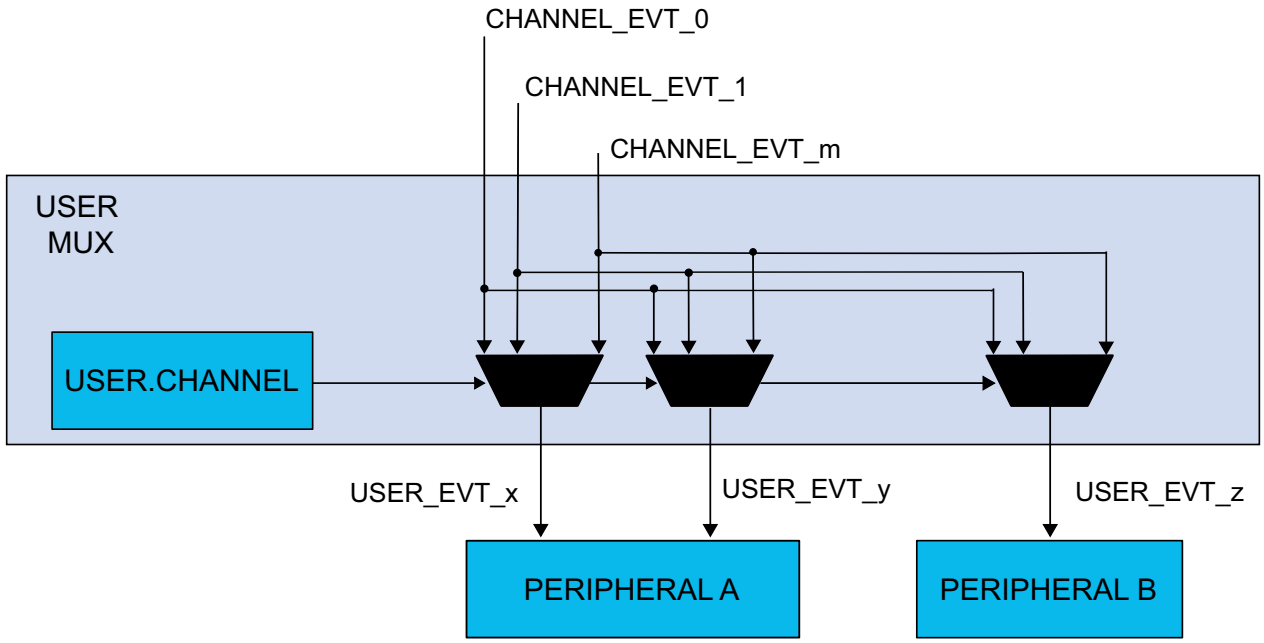
22.6.2.3 User Multiplexer Setup

Each user multiplexer is dedicated to one event user. A user multiplexer receives all event channel outputs and must be configured to select one of these channels. The user must always be configured before the channel is configured. A full list of selectable users can be found in the User Multiplexer register (USER) description. Refer to Table 22-6 for details.

To configure a user multiplexer, the USER register must be written in a single 16-bit write.

It is possible to read out the configuration of a user by first selecting the user by writing to USER.USER using an 8-bit write and then performing a read of the USER register.

Figure 22-2. User MUX



22.6.2.4 Channel Setup

The channel to be used with an event user must be configured with an event generator. The path of the channel should be configured, and when using a synchronous path or resynchronized path, the edge selection should be configured. All these configurations are available in the Channel register (CHANNEL).

To configure a channel, the Channel register must be written in a single, 32-bit write.

It is possible to read out the configuration of a channel by first selecting the channel by writing to CHANNEL.CHANNEL using a, 8-bit write, and then performing a read of the CHANNEL register.

Event Generators

The event generator is selected by writing to the Event Generator bit group in the Channel register (CHANNEL.EVGEN).

A full list of selectable generators can be found in the CHANNEL register description. Refer to Table 22-3 for details.

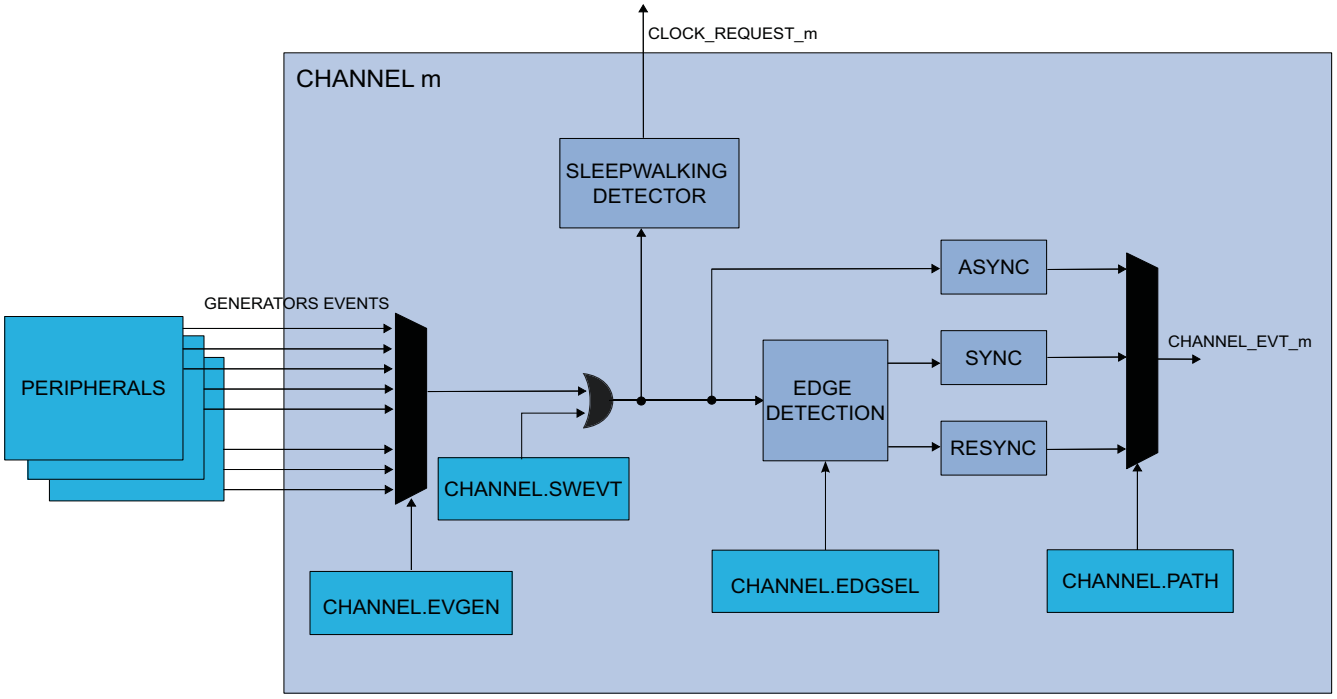
The channels are not connected to any of the event generators (CHANNEL.EVGEN = 0x00) by default.

22.6.2.5 Channel Path

There are three different ways to propagate the event provided by an event generator:

- Asynchronous path
- Synchronous path
- Resynchronized path

Figure 22-3. Channel



The path is selected by writing to the Path Selection bit group in the Channel register (CHANNEL.PATH).

Asynchronous Path

When using the asynchronous path, the events are propagated from the event generator to the event user with no intervention from the event system. This means that if the GCLK_EVSYS_x for the channel used is inactive, the event will still be propagated to the user.

Events propagated in the asynchronous path cannot generate any interrupts, and no channel status bits will indicate the state of the channel. No edge detection is available; this must be handled in the event user.

When the event generator and the event user share the same generic clock, using the asynchronous path will propagate the event with the least amount of latency.

Synchronous Path

The synchronous path should be used when the event generator and the event channel share the same generic clock. If they do not share the same clock, a logic change from the event generator to the event channel might not be detected in the channel, which means that the event will not be propagated to the event user.

When using the synchronous path, the channel is capable of generating interrupts. The channel status bits in the Channel Status register (CHSTATUS) are also updated and available for use.

If the Generic Clocks Request bit in the Control register (CTRL.GCLKREQ) is zero, the channel operates in SleepWalking mode and request the configured generic clock only when an event is to be propagated through the channel. If CTRL.GCLKREQ is one, the generic clock will always be on for the configured channel.

Resynchronized Path

The resynchronized path should be used when the event generator and the event channel do not share the same clock. When the resynchronized path is used, resynchronization of the event from the event generator is done in the channel.

When the resynchronized path is used, the channel is capable of generating interrupts. The channel status bits in the Channel Status register (CHSTATUS) are also updated and available for use.

If the Generic Clocks Request bit in the Control register (CTRL.GCLKREQ) is zero, the channel operates in SleepWalking mode and request the configured generic clock only when an event is to be propagated through the channel. If CTRL.GCLKREQ is one, the generic clock will always be on for the configured channel.

22.6.2.6 Edge Detection

When synchronous or resynchronized paths are used, edge detection must be used. The event system can perform edge detection in three different ways:

- Generate an event only on the rising edge
- Generate an event only on the falling edge
- Generate an event on rising and falling edges.

Edge detection is selected by writing to the Edge Selection bit group in the Channel register (CHANNEL.EDGSEL).

If the generator event is a pulse, the Both Edges method must not be selected. Use the Rising Edge or Falling Edge detection method, depending on the generator event default level.

22.6.2.7 The Overrun Channel x Interrupt

The Overrun Channel x interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVRx) is set and the optional interrupt is generated in the following two cases:

- At least one of the event users on channel x is not ready when a new event occurs
- An event occurs when the previous event on channel x has not yet been handled by all event users

INTFLAG.OVRx will be set when using a synchronous or resynchronized path, but not when using an asynchronous path.

22.6.2.8 The Event Detected Channel x Interrupt

The Event Detected Channel x interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.EVDx) is set when an event coming from the event generator configured on channel x is detected.

INTFLAG.EVDx will be set when using a synchronous and resynchronized path, but not when using an asynchronous path.

22.6.2.9 Channel Status

The Channel Status register (CHSTATUS) updates the status of the channels when a synchronous or resynchronized path is in use. There are two different status bits in CHSTATUS for each of the available channels: The CHSTATUS.CHBUSYx bit is set to one if an event on the corresponding channel x has not been handled by all event users connected to that channel.

The CHSTATUS.USRRDYx bit is set to one if all event users connected to the corresponding channel x are ready to handle incoming events on that channel.

22.6.2.10 Software Event

A software event can be initiated on a channel by writing a one to the Software Event bit in the Channel register (CHANNEL.SWEVT) together with the Channel bits (CHANNEL.CHANNEL). This will generate a software event on the selected channel.

The software event can be used for application debugging, and functions like any event generator. To use the software event, the event path must be configured to either a synchronous path or resynchronized path (CHANNEL.PATH = 0x0 or 0x1), edge detection must be configured to rising-edge detection (CHANNEL.EDGSEL= 0x1) and the Generic Clock Request bit must be set to one (CTRL.GCLKREQ=0x1).

22.6.3 Interrupts

The EVSYS has the following interrupt sources:

- Overrun Channel x interrupt (INTFLAG)
- Event Detected Channel x interrupt (INTFLAG)

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the EVSYS is reset.

See the INTFLAG register for details on how to clear interrupt flags. The EVSYS has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated.

Refer to “Nested Vector Interrupt Controller” on page 24 for details.

22.6.4 Sleep Mode Operation

The EVSYS can generate interrupts to wake up the device from any sleep mode.

22.7 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRL	7:0				GCLKREQ				SWRST
0x01	Reserved									
0x02	Reserved									
0x03	Reserved									
0x04	CHANNEL	7:0	CHANNEL[7:0]							
0x05		15:8								SWEVT
0x06		23:16	EVGEN[7:0]							
0x07		31:24					EDGSEL[1:0]		PATH[1:0]	
0x08	USER	7:0	USER[7:0]							
0x09		15:8	CHANNEL[7:0]							
0x0A	Reserved									
0x0B	Reserved									
0x0C	CHSTATUS	7:0	USRRDY7	USRRDY6	USRRDY5	USRRDY4	USRRDY3	USRRDY2	USRRDY1	USRRDY0
0x0D		15:8	CHBUSY7	CHBUSY6	CHBUSY5	CHBUSY4	CHBUSY3	CHBUSY2	CHBUSY1	CHBUSY0
0x0E		23:16								
0x0F		31:24								
0x10	INTENCLR	7:0	OVR7	OVR6	OVR5	OVR4	OVR3	OVR2	OVR1	OVR0
0x11		15:8	EVD7	EVD6	EVD5	EVD4	EVD3	EVD2	EVD1	EVD0
0x12		23:16								
0x13		31:24								
0x14	INTENSET	7:0	OVR7	OVR6	OVR5	OVR4	OVR3	OVR2	OVR1	OVR0
0x15		15:8	EVD7	EVD6	EVD5	EVD4	EVD3	EVD2	EVD1	EVD0
0x16		23:16								
0x17		31:24								
0x18	INTFLAG	7:0	OVR7	OVR6	OVR5	OVR4	OVR3	OVR2	OVR1	OVR0
0x19		15:8	EVD7	EVD6	EVD5	EVD4	EVD3	EVD2	EVD1	EVD0
0x1A		23:16								
0x1B		31:24								

22.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 303 and “PAC – Peripheral Access Controller” on page 27 for details.

22.8.1 Control

Name: CTRL

Offset: 0x00

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				GCLKREQ				SWRST
Access	R	R	R	R/W	R	R	R	W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – GCLKREQ: Generic Clock Requests**
  This bit is used to determine whether the generic clocks used for the different channels should be on all the time or only when an event needs the generic clock. Events propagated trough asynchronous paths will not need a generic clock.
  0: Generic clock is requested and turned on only if an event is detected.
  1: Generic clock for a channel is always on.

- **Bits 3:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – SWRST: Software Reset**
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the EVSYS to their initial state.
  Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

22.8.2 Channel

This register allows the user to configure the channel specified in the CHANNEL bit group. To write to this register, do a single, 32-bit write of all the configuration and channel selection data.

To read from this register, first do an 8-bit write to the CHANNEL.CHANNEL bit group specifying the channel configuration to be read, and then read the Channel register (CHANNEL).

Name: CHANNEL

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
					EDGSEL[1:0]		PATH[1:0]	
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EVGEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								SWEVT
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CHANNEL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:28 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 27:26 – EDGSEL: Edge Detection Selection**
  These bits set the type of edge detection to be used on the channel.
  These bits must be written to zero when using the asynchronous path.

Table 22-1. Edge Detection Selection

Value	Name	Description
0x0	NO_EVT_OUTPUT	No event output when using the resynchronized or synchronous path
0x1	RISING_EDGE	Event detection only on the rising edge of the signal from the event generator
0x2	FALLING_EDGE	Event detection only on the falling edge of the signal from the event generator
0x3	BOTH_EDGES	Event detection on rising and falling edges of the signal from the event generator

- **Bits 25:24 – PATH: Path Selection**
  These bits are used to chose which path will be used by the selected channel.
  The path choice can be limited by the channel source, see Table 22-6.

Table 22-2. Path Selection

Value	Name	Description
0x0	SYNCHRONOUS	Synchronous path
0x1	RESYNCHRONIZED	Resynchronized path
0x2	ASYNCHRONOUS	Asynchronous path
0x3	-	Reserved

- **Bits 23:16 – EVGEN: Event Generator**
  These bits are used to choose the event generator to connect to the selected channel.

Table 22-3. Event Generator Selection

Value	Event Generator	Description
0x00	NONE	No event generator selected
0x01	RTC CMP0	Compare 0 (mode 0 and 1) or Alarm 0 (mode 2)
0x02	RTC CMP1	Compare 1
0x03	RTC OVF	Overflow
0x04	RTC PER0	Period 0
0x05	RTC PER1	Period 1
0x06	RTC PER2	Period 2
0x07	RTC PER3	Period 3
0x08	RTC PER4	Period 4
0x09	RTC PER5	Period 5
0x0A	RTC PER6	Period 6
0x0B	RTC PER7	Period 7
0x0C	EIC EXTINT0	External Interrupt 0
0x0D	EIC EXTINT1	External Interrupt 1

Table 22-3. Event Generator Selection (Continued)

Value	Event Generator	Description
0x0E	EIC EXTINT2	External Interrupt 2
0x0F	EIC EXTINT3	External Interrupt 3
0x10	EIC EXTINT4	External Interrupt 4
0x11	EIC EXTINT5	External Interrupt 5
0x12	EIC EXTINT6	External Interrupt 6
0x13	EIC EXTINT7	External Interrupt 7
0x14	EIC EXTINT8	External Interrupt 8
0x15	EIC EXTINT9	External Interrupt 9
0x16	EIC EXTINT10	External Interrupt 10
0x17	EIC EXTINT11	External Interrupt 11
0x18	EIC EXTINT12	External Interrupt 12
0x19	EIC EXTINT13	External Interrupt 13
0x1A	EIC EXTINT14	External Interrupt 14
0x1B	EIC EXTINT15	External Interrupt 15
0x1C	TC0 OVF	Overflow/Underflow
0x1D	TC0 MC0	Match/Capture 0
0x1E	TC0 MC1	Match/Capture 1
0x1F	TC1 OVF	Overflow/Underflow
0x20	TC1 MC0	Match/Capture 0
0x21	TC1 MC1	Match/Capture 1
0x22	TC2 OVF	Overflow/Underflow
0x23	TC2 MC0	Match/Capture 0
0x24	TC2 MC1	Match/Capture 1
0x25	TC3 OVF	Overflow/Underflow
0x26	TC3 MC0	Match/Capture 0
0x27	TC3 MC1	Match/Capture 1
0x28	TC4 OVF	Overflow/Underflow
0x29	TC4 MC0	Match/Capture 0
0x2A	TC4 MC1	Match/Capture 1
0x2B	TC5 OVF	Overflow/Underflow
0x2C	TC5 MC0	Match/Capture 0
0x2D	TC5 MC1	Match/Capture 1
0x2E	TC6 OVF	Overflow/Underflow

Table 22-3. Event Generator Selection (Continued)

Value	Event Generator	Description
0x2F	TC6 MC0	Match/Capture 0
0x30	TC6 MC1	Match/Capture 1
0x31	TC7 OVF	Overflow/Underflow
0x32	TC7 MC0	Match/Capture 0
0x33	TC7 MC1	Match/Capture 1
0x34	ADC RESRDY	Result Ready
0x35	ADC WINMON	Window Monitor
0x36	AC COMP0	Comparator 0
0x37	AC COMP1	Comparator 1
0x38	AC WIN	Window 0
0x39	DAC EMPTY	Data Buffer Empty
0x3A	PTC EOC	End of Conversion
0x3B	PTC WCOMP	Window Comparator
0x3C-0xFF	Reserved	

- **Bits 15:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 8 – SWEVT: Software Event**
  This bit is used to insert a software event on the channel selected by the CHANNEL.CHANNEL bit group.
  This bit must be written together with CHANNEL.CHANNELusing a 16-bit write.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will trigger a software event for the corresponding channel.
  This bit will always return zero when read.

- **Bits 7:0 – CHANNEL: Channel Selection**
  These bits are used to select the channel to be set up or read from.

Table 22-4. Channel Selection

Value	Channel Number
0x00	0
0x01	1
0x02	2
0x03	3
0x04	4
0x05	5
0x06	6
0x07	7
0x08-0xFF	Reserved

22.8.3 User Multiplexer

This register is used to configure a specified event user. To write to this register, do a single, 16-bit write of all the configuration and event user selection data.

To read from this register, first do an 8-bit write to the USER.USER bit group specifying the event user configuration to be read, and then read USER.

Name: USER

Offset: 0x08

Reset: 0x0000

Property: Write-protected

Bit	15	14	13	12	11	10	9	8
	CHANNEL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	USER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:8 – CHANNEL: Channel Event Selection**
  These bits are used to select the channel to connect to the event user.
  Note that to select channel n, the value (n+1) must be written to the USER.CHANNEL bit group.

Table 22-5. Channel Event Selection

Value	Channel Number
0x00	No channel output selected
0x01	0
0x02	1
0x03	2
0x04	3
0x05	4
0x06	5
0x07	6
0x08	7
0x09-0xFF	Reserved

- **Bits 7:0 – USER: User Multiplexer Selection**
  These bits select the event user to be configured with a channel, or the event user to read the channel value from.

Table 22-6. User Multiplexer Selection

USER[7:0]	User Multiplexer	Description	Path Type
0x00	TC0		Asynchronous, synchronous and resynchronized paths
0x01	TC1		Asynchronous, synchronous and resynchronized paths
0x02	TC2		Asynchronous, synchronous and resynchronized paths
0x03	TC3		Asynchronous, synchronous and resynchronized paths
0x04	TC4		Asynchronous, synchronous and resynchronized paths
0x05	TC5		Asynchronous, synchronous and resynchronized paths
0x06	TC6		Asynchronous, synchronous and resynchronized paths
0x07	TC7		Asynchronous, synchronous and resynchronized paths
0x08	ADC START	ADC start conversion	Asynchronous path only
0x09	ADC SYNC	Flush ADC	Asynchronous path only
0x0A	AC COMP0	Start comparator 0	Asynchronous path only
0x0B	AC COMP1	Start comparator 1	Asynchronous path only
0x0C	DAC START	DAC start conversion	Asynchronous path only
0x0D	PTC STCONV	PTC start conversion	Asynchronous path only
0x0E-0xFF	Reserved		Reserved

22.8.4 Channel Status

Name: CHSTATUS

Offset: 0x0C

Reset: 0x000000FF

Property: –

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CHBUSY7	CHBUSY6	CHBUSY5	CHBUSY4	CHBUSY3	CHBUSY2	CHBUSY1	CHBUSY0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	USRRDY7	USRRDY6	USRRDY5	USRRDY4	USRRDY3	USRRDY2	USRRDY1	USRRDY0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 15:8 – CHBUSYx: Channel Busy x**
  This bit is cleared when channel x is idle
  This bit is set if an event on channel x has not been handled by all event users connected to channel x.
- **Bits 7:0 – USRRDYx: User Ready for Channel x**
  This bit is cleared when at least one of the event users connected to the channel is not ready.
  This bit is set when all event users connected to channel x are ready to handle incoming events on channel x.

22.8.5 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x10

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EVD7	**EVD6**	**EVD5**	**EVD4**	**EVD3**	**EVD2**	**EVD1**	**EVD0**
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OVR7	**OVR6**	**OVR5**	**OVR4**	**OVR3**	**OVR2**	**OVR1**	**OVR0**
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:8 – EVDx: Event Detected Channel x Interrupt Enable**
  0: The Event Detected Channel x interrupt is disabled.
  1: The Event Detected Channel x interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Event Detected Channel x Interrupt Enable bit, which disables the Event Detected Channel x interrupt.

- **Bits 7:0 – OVRx: Overrun Channel x Interrupt Enable**
  0: The Overrun Channel x interrupt is disabled.
  1: The Overrun Channel x interrupt is enabled.

Writing a zero to this bit has no effect.
Writing a one to this bit will clear the Overrun Channel x Interrupt Enable bit, which disables the Overrun Channel x interrupt.

22.8.6 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x14

Reset: 0x00000000

Property: Write-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EVD7	**EVD6**	**EVD5**	**EVD4**	**EVD3**	**EVD2**	**EVD1**	**EVD0**
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OVR7	**OVR6**	**OVR5**	**OVR4**	**OVR3**	**OVR2**	**OVR1**	**OVR0**
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:8 – EVDx: Event Detected Channel x Interrupt Enable**
  0: The Event Detected Channel x interrupt is disabled.
  1: The Event Detected Channel x interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Event Detected Channel x Interrupt Enable bit, which enables the Event Detected Channel x interrupt.

- **Bits 7:0 – OVRx: Overrun Channel x Interrupt Enable**
  0: The Overrun Channel x interrupt is disabled.
  1: The Overrun Channel x interrupt is enabled.

Writing a zero to this bit has no effect.
Writing a one to this bit will set the Overrun Channel x Interrupt Enable bit, which enables the Overrun Channel x interrupt.

22.8.7 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x18

Reset: 0x00000000

Property: –

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EVD7	**EVD6**	**EVD5**	**EVD4**	**EVD3**	**EVD2**	**EVD1**	**EVD0**
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OVR7	**OVR6**	**OVR5**	**OVR4**	**OVR3**	**OVR2**	**OVR1**	**OVR0**
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:16 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 15:8 – EVDx: Event Detected Channel x**
  This flag is set on the next CLK_EVSYS_APB cycle when an event is being propagated through the channel, and an interrupt request will be generated if INTENCLR/SET.EVDx is one.
  When the event channel path is asynchronous, the EVDx interrupt flag will not be set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Event Detected Channel n interrupt flag.

- **Bits 7:0 – OVRx: Overrun Channel x**
  This flag is set on the next CLK_EVSYS cycle after an overrun channel condition occurs, and an interrupt request will be generated if INTENCLR/SET.OVRx is one.
  There are two possible overrun channel conditions:
  - One or more of the event users on channel x are not ready when a new event occurs
  - An event happens when the previous event on channel x has not yet been handled by all event users

  When the event channel path is asynchronous, the OVRx interrupt flag will not be set.

Writing a zero to this bit has no effect.
Writing a one to this bit will clear the Overrun Channel x interrupt flag.

23. SERCOM – Serial Communication Interface

23.1 Overview

The serial communication interface (SERCOM) can be configured to support a number of modes; I^2C, SPI and USART. Once configured and enabled, all SERCOM resources are dedicated to the selected mode.

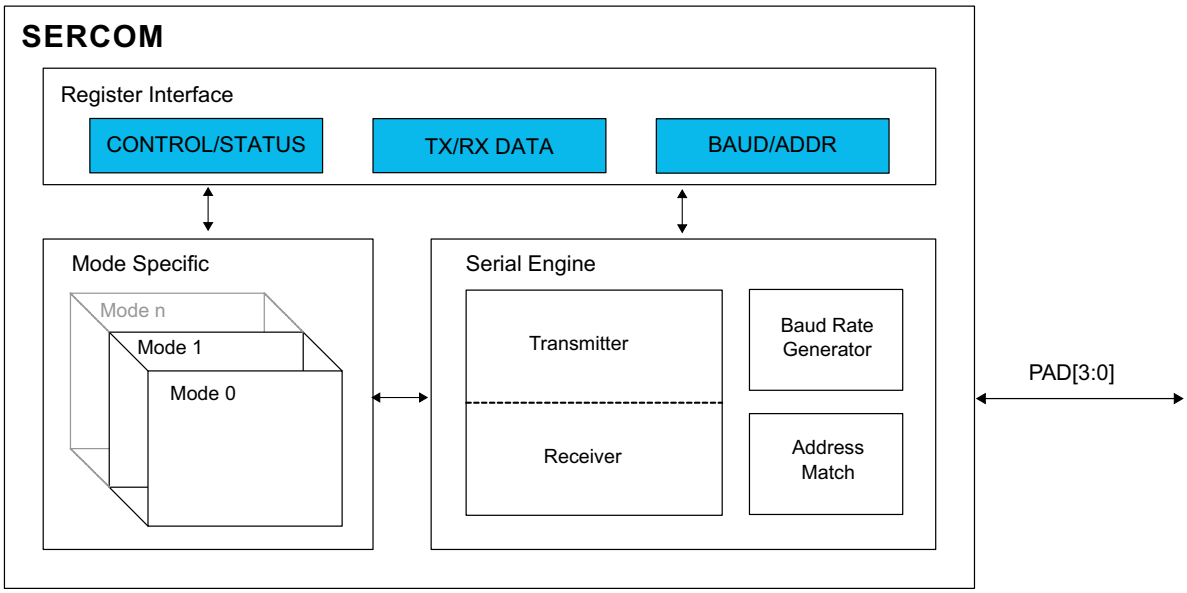
The SERCOM serial engine consists of a transmitter and receiver, baud-rate generator and address matching functionality. It can be configured to use the internal generic clock or an external clock, making operation in all sleep modes possible.

23.2 Features

- Combined interface configurable as one of the following:
  - I^2C – Two-wire serial interface
    - SMBus™ compatible.
  - SPI – Serial peripheral interface
  - USART – Universal synchronous and asynchronous serial receiver and transmitter
- Single transmit buffer and double receive buffer
- Baud-rate generator
- Address match/mask logic
- Operational in all sleep modes

23.3 Block Diagram

Figure 23-1. SERCOM Block Diagram



23.4 Signal Description

See the respective SERCOM mode chapters for details:

- “SERCOM USART – SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter” on page 332
- “SERCOM SPI – SERCOM Serial Peripheral Interface” on page 357
- “SERCOM I2C – SERCOM Inter-Integrated Circuit” on page 382

23.5 Product Dependencies

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

23.5.1 I/O Lines

Using the SERCOM I/O lines requires the I/O pins to be configured using port configuration (PORT). Refer to “PORT” on page 276 for details.

From Figure 23-1 one can see that the SERCOM has four internal pads, PAD[3:0]. The signals from I^2C, SPI and USART are routed through these SERCOM pads via a multiplexer. The configuration of the multiplexer is available from the different SERCOM modes. Refer to the mode specific chapters for details:

- “SERCOM USART – SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter” on page 332
- “SERCOM SPI – SERCOM Serial Peripheral Interface” on page 357
- “SERCOM I2C – SERCOM Inter-Integrated Circuit” on page 382

23.5.2 Power Management

The SERCOM can operate in any sleep mode.SERCOM interrupts can be used to wake up the device from sleep modes. Refer to “PM – Power Manager” on page 100 for details on the different sleep modes.

23.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) is enabled by default, and can be enabled and disabled in the Power Manager. Refer to “PM – Power Manager” on page 100 for details.

Two generic clocks are used by the SERCOM: GCLK_SERCOMx_CORE and GCLK_SERCOMx_SLOW. The core clock (GCLK_SERCOMx_CORE) is required to clock the SERCOM while operating as a master, while the slow clock (GCLK_SERCOMx_SLOW) is only required for certain functions. See specific mode chapters for details.

These clocks must be configured and enabled in the Generic Clock Controller (GCLK) before using the SERCOM. Refer to “GCLK – Generic Clock Controller” on page 78 for details.

These generic clocks are asynchronous to the user interface clock (CLK_SERCOMx_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to “Synchronization” on page 331 for further details.

23.5.4 DMA

Not applicable.

23.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the SERCOM interrupts requires the Interrupt Controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

23.5.6 Events

Not applicable.

23.5.7 Debug Operation

When the CPU is halted in debug mode, the SERCOM continues normal operation. If the SERCOM 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. The SERCOM can be forced to halt operation during debugging.

23.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Address register (ADDR)
- Data register (DATA)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

23.5.9 Analog Connections

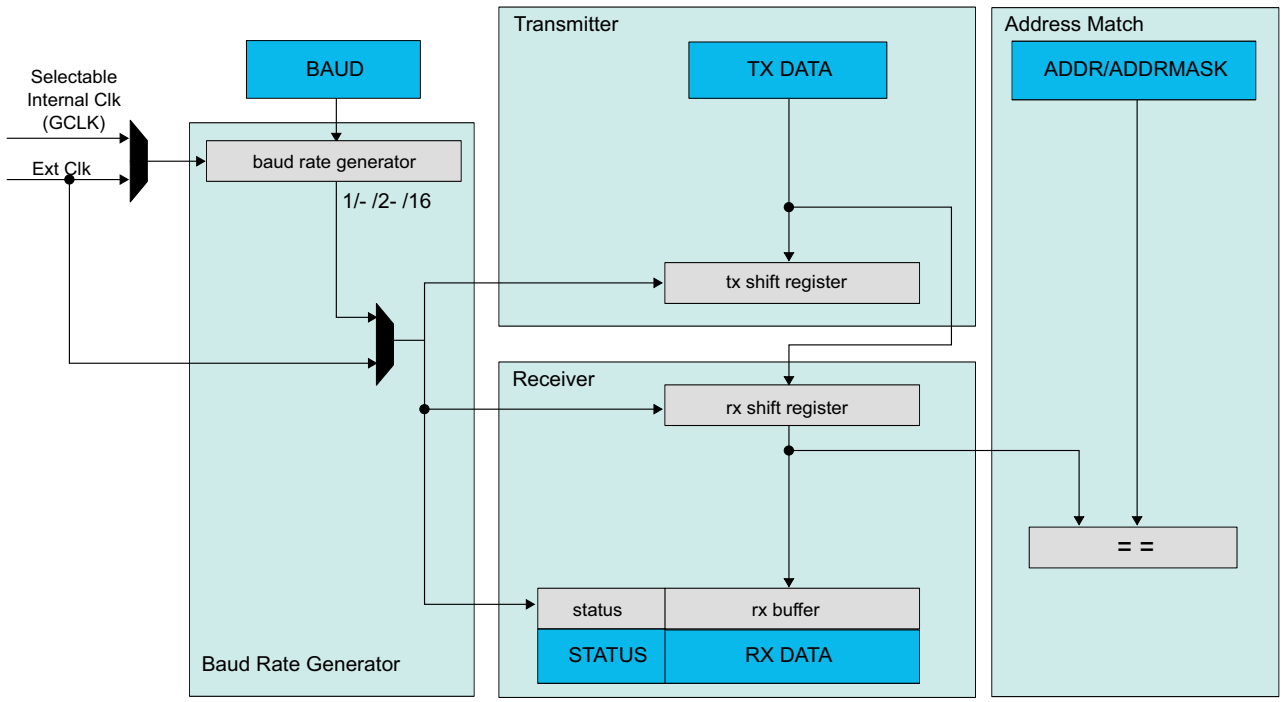
Not applicable.

23.6 Functional Description

23.6.1 Principle of Operation

The basic structure of the SERCOM serial engine is shown in Figure 23-2. Fields shown in capital letters are synchronous to the system clock and accessible by the CPU, while fields with lowercase letters can be configured to run on the GCLK_SERCOMx_CORE clock or an external clock.

Figure 23-2. SERCOM Serial Engine



The transmitter consists of a single write buffer and a shift register. The receiver consists of a two-level receive buffer and a shift register. The baud-rate generator is capable of running on the GCLK_SERCOMx_CORE clock or an external clock. Address matching logic is included for SPI and I^2C operation.

23.6.2 Basic Operation

23.6.2.1 Initialization

The SERCOM must be configured to the desired mode by writing to the Operating Mode bits in the Control A register (CTRLA.MODE). Refer to Figure 23-1 for details.

Table 23-1. SERCOM Modes

CTRLA.MODE	Description
0x0	USART with external clock
0x1	USART with internal clock
0x2	SPI in slave operation
0x3	SPI in master operation
0x4	I^2C slave operation
0x5	I^2C master operation
0x6-0x7	Reserved

For further initialization information, see the respective SERCOM mode chapters.

23.6.2.2 Enabling, Disabling and Resetting

The SERCOM is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The SERCOM is disabled by writing a zero to CTRLA.ENABLE.

The SERCOM is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the SERCOM, except DBGCTRL, will be reset to their initial state, and the SERCOM will be disabled. Refer to the CTRLA register descriptions for details.

23.6.2.3 Clock Generation – Baud-Rate Generator

The baud-rate generator, as shown in Figure 23-3, is used for internal clock generation for asynchronous and synchronous communication. The generated output frequency (f_{BAUD}) is determined by the Baud register (BAUD) setting and the baud reference frequency (f_{REF}). The baud reference clock is the serial engine clock, and it can be internal or external.

For asynchronous operation, the /16 (divide-by-16) output is used when transmitting and the /1 (divide-by-1) output is used when receiving. For synchronous operation the /2 (divide-by-2) output is used. This functionality is automatically configured, depending on the selected operating mode.

Figure 23-3. Baud Rate Generator

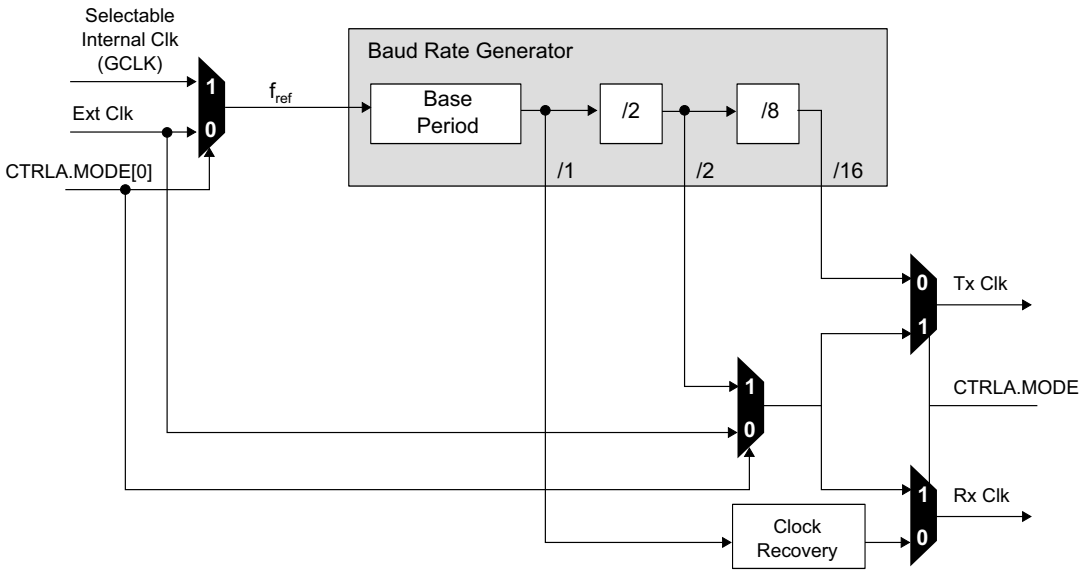


Table 23-2 contains equations for calculating the baud rate (in bits per second) and for calculating the BAUD register value for each mode of operation.

For asynchronous mode, the BAUD register value is 16 bits (0 to 65,535), while for synchronous mode, the BAUD register value is 8 bits (0 to 255).

Table 23-2. Baud Rate Equations

Operating Mode	Condition	Baud Rate (Bits Per Second)	BAUD Register Value Calculation
Asynchronous	$f_{BAUD} \leq \frac{f_{REF}}{16}$	$f_{BAUD} = \frac{f_{REF}}{16}\left(1 - \frac{BAUD}{65{,}536}\right)$	$BAUD = 65{,}536\left(1 - 16\frac{f_{BAUD}}{f_{REF}}\right)$
Synchronous	$f_{BAUD} \leq \frac{f_{REF}}{2}$	$f_{BAUD} = \frac{f_{REF}}{2(BAUD+1)}$	$BAUD = \frac{f_{REF}}{2f_{BAUD}} - 1$

Asynchronous Mode BAUD Value Selection

The formula given for f_{BAUD} calculates the average frequency over 65,536 f_{REF} cycles. Although the BAUD register can be set to any value between 0 and 65,536, the values that will change the average frequency of f_{BAUD} over a single frame are more constrained. The BAUD register values that will affect the average frequency over a single frame lead to an integer increase in the cycles per frame (CPF)

$$CPF = \frac{f_{REF}}{f_{BAUD}}(D+S)$$

where

- D represent the data bits per frame
- S represent the sum of start and first stop bits, if present

Table 23-3 shows the BAUD register value versus baud frequency at a serial engine frequency of 48MHz. This assumes a D value of 8 bits and an S value of 2 bits (10 bits, including start and stop bits).

Table 23-3. BAUD Register Value vs. Baud Frequency

BAUD Register Value	Serial Engine CPF	f_{BAUD} at 48MHz Serial Engine Frequency (f_{REF})
0 – 406	160	3MHz
407 – 808	161	2.981MHz
809 – 1205	162	2.963MHz
...		
65206	31775	15.11kHz
65207	31871	15.06kHz
65208	31969	15.01kHz

23.6.3 Additional Features

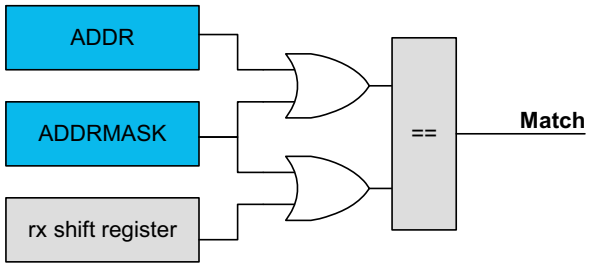
23.6.3.1 Address Match and Mask

The SERCOM address match and mask feature is capable of matching one address with a mask, two unique addresses or a range of addresses, based on the mode selected. The match uses seven or eight bits, depending on the mode.

Address With Mask

An address written to the Address bits in the Address register (ADDR.ADDR) with a mask written to the Address Mask bits in the Address register (ADDR.ADDRMASK) will yield an address match. All bits that are masked are not included in the match. Note that setting the ADDR.ADDRMASK to all zeros will match a single unique address, while setting ADDR.ADDRMASK to all ones will result in all addresses being accepted.

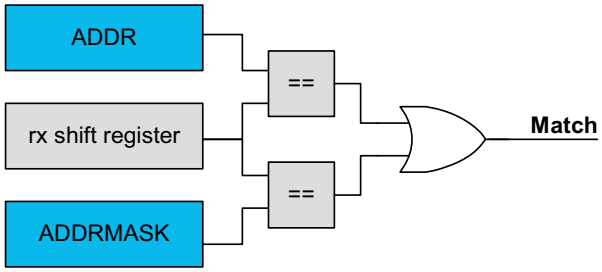
Figure 23-4. Address With Mask



Two Unique Addresses

The two addresses written to ADDR and ADDRMASK will cause a match.

Figure 23-5. Two Unique Addresses



Address Range

The range of addresses between and including ADDR.ADDR and ADDR.ADDRMASK will cause a match. ADDR.ADDR and ADDR.ADDRMASK can be set to any two addresses, with ADDR.ADDR acting as the upper limit and ADDR.ADDRMASK acting as the lower limit.

Figure 23-6. Address Range

23.6.4 DMA Operation

Not applicable.

23.6.5 Interrupts

Interrupt sources are mode-specific. See the respective SERCOM mode chapters for details.

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the SERCOM is reset. See the register description for details on how to clear interrupt flags.

The SERCOM has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

23.6.6 Events

Not applicable.

23.6.7 Sleep Mode Operation

The peripheral can operate in any sleep mode where the selected serial clock is running. This clock can be external or generated by the internal baud-rate generator.

The SERCOM interrupts can be used to wake up the device from sleep modes. Refer to the different SERCOM mode chapters for details.

23.6.8 Synchronization

Due to the asynchronicity between CLK_SERCOMx_APB and GCLK_SERCOMx_CORE, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The Synchronization Ready interrupt can be used to signal when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

24. SERCOM USART – SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter

24.1 Overview

The universal synchronous and asynchronous receiver and transmitter (USART) is one of the available modes in the Serial Communication Interface (SERCOM).

Refer to “SERCOM – Serial Communication Interface” on page 324 for details.

The USART uses the SERCOM transmitter and receiver configured as shown in Figure 24-1. Fields shown in capital letters are synchronous to the CLK_SERCOMx_APB and accessible by the CPU, while fields with lowercase letters can be configured to run on the internal generic clock or an external clock.

The transmitter consists of a single write buffer, a shift register and control logic for handling different frame formats. The write buffer allows continuous data transmission without any delay between frames.

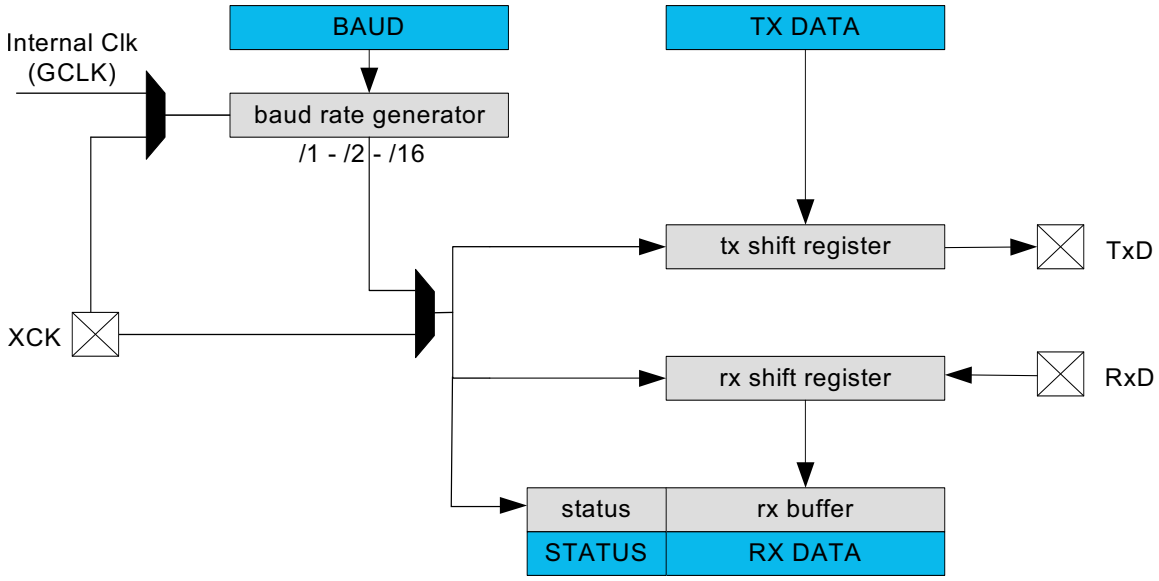
The receiver consists of a two-level receive buffer and a shift register. Status information for the received data is available for error checking. Data and clock recovery units ensure robust synchronization and noise filtering during asynchronous data reception.

24.2 Features

- Full-duplex operation
- Asynchronous (with clock reconstruction) or synchronous operation
- Internal or external clock source for asynchronous and synchronous operation
- Baud-rate generator
- Supports serial frames with 5, 6, 7, 8 or 9 data bits and 1 or 2 stop bits
- Odd or even parity generation and parity check
- Selectable LSB- or MSB-first data transfer
- Buffer overflow and frame error detection
- Noise filtering, including false start-bit detection and digital low-pass filter
- Can operate in all sleep modes
- Operation at speeds up to half the system clock for internally generated clocks
- Operation at speeds up to the system clock for externally generated clocks

24.3 Block Diagram

Figure 24-1. USART Block Diagram



24.4 Signal Description

Signal Name	Type	Description
PAD[3:0]	Digital I/O	General SERCOM pins

Refer to “I/O Multiplexing and Considerations” on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

24.5 Product Dependencies

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

24.5.1 I/O Lines

Using the USART’s I/O lines requires the I/O pins to be configured using port configuration (PORT).

Refer to “PORT” on page 276 for details.

When the SERCOM is used in USART mode, the pins should be configured according to Table 24-1. If the receiver or transmitter is disabled, these pins can be used for other purposes.

Table 24-1. USART Pin Configuration

Pin	Pin Configuration
TxD	Output
RxD	Input
XCK	Output or input

The combined configuration of PORT and the Transmit Data Pinout and Receive Data Pinout bit groups (refer to the Control A register description) will define the physical position of the USART signals in Table 24-1.

24.5.2 Power Management

The USART can continue to operate in any sleep mode where the selected source clock is running. The USART interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

24.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB, where x represents the specific SERCOM instance number) can be enabled and disabled in the Power Manager, and the default state of CLK_SERCOMx_APB can be found in the Peripheral Clock Masking section in "PM – Power Manager" on page 100.

A generic clock (GCLK_SERCOMx_CORE) is required to clock the SERCOMx_CORE. This clock must be configured and enabled in the Generic Clock Controller before using the SERCOMx_CORE. Refer to "GCLK – Generic Clock Controller" on page 78 for details.

This generic clock is asynchronous to the bus clock (CLK_SERCOMx_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to "Synchronization" on page 340 for further details.

24.5.4 DMA

Not applicable.

24.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the USART interrupts requires the Interrupt Controller to be configured first. Refer to "Nested Vector Interrupt Controller" on page 24 for details.

24.5.6 Events

Not applicable.

24.5.7 Debug Operation

When the CPU is halted in debug mode, the USART continues normal operation. If the USART 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. The USART can be forced to halt operation during debugging.

Refer to DBGCTRL for details.

24.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Status register (STATUS)
- Data register (DATA)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

24.5.9 Analog Connections

Not applicable.

24.6 Functional Description

24.6.1 Principle of Operation

The USART uses three communication lines for data transfer:

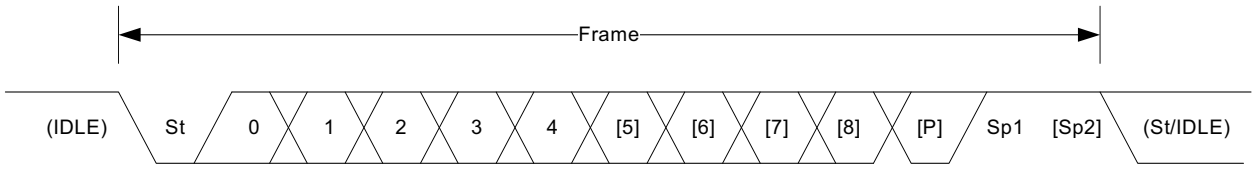
- RxD for receiving
- TxD for transmitting
- XCK for the transmission clock in synchronous operation

USART data transfer is frame based, where a serial frame consists of:

- 1 start bit
- 5, 6, 7, 8 or 9 data bits
- MSB or LSB first
- No, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by one character of data bits. If enabled, the parity bit is inserted after the data bits and before the first stop bit. One frame can be directly followed by a new frame, or the communication line can return to the idle (high) state. Figure 24-2 illustrates the possible frame formats. Bits inside brackets are optional.

Figure 24-2. Frame Formats



St Start bit; always low

(n) Data bits; 0 to 8

P Parity bit; odd or even

Sp Stop bit; always high

IDLE No transfers on the communication line; always high in this state

24.6.2 Basic Operation

24.6.2.1 Initialization

The following registers are enable-protected, meaning they can only be written when the USART is disabled (CTRL.ENABLE is zero):

- Control A register (CTRLA), except the Enable (ENABLE) and Software Reset (SWRST) bits
- Control B register (CTRLB), except the Receiver Enable (RXEN) and Transmitter Enable (TXEN) bits
- Baud register (BAUD)

Any writes to these registers when the USART is enabled or is being enabled (CTRL.ENABLE is one) will be discarded. Writes to these registers) while the peripheral is being disabled will be completed after the disabling is complete.

Before the USART is enabled, it must be configured, as outlined in the following steps:

- USART mode with external or internal clock must be selected first by writing 0x0 or 0x1 to the Operating Mode bit group in the Control A register (CTRLA.MODE)

- Communication mode (asynchronous or synchronous) must be selected by writing to the Communication Mode bit in the Control A register (CTRLA.CMODE)
- SERCOM pad to use for the receiver must be selected by writing to the Receive Data Pinout bit group in the Control A register (CTRLA.RXPO)
- SERCOM pads to use for the transmitter and external clock must be selected by writing to the Transmit Data Pinout bit in the Control A register (CTRLA.TXPO)
- Character size must be selected by writing to the Character Size bit group in the Control B register (CTRLB.CHSIZE)
- MSB- or LSB-first data transmission must be selected by writing to the Data Order bit in the Control A register (CTRLA.DORD)
- When parity mode is to be used, even or odd parity must be selected by writing to the Parity Mode bit in the Control B register (CTRLB.PMODE) and enabled by writing 0x1 to the Frame Format bit group in the Control A register (CTRLA.FORM)
- Number of stop bits must be selected by writing to the Stop Bit Mode bit in the Control B register (CTRLB.SBMODE)
- When using an internal clock, the Baud register (BAUD) must be written to generate the desired baud rate
- The transmitter and receiver can be enabled by writing ones to the Receiver Enable and Transmitter Enable bits in the Control B register (CTRLB.RXEN and CTRLB.TXEN)

24.6.2.2 Enabling, Disabling and Resetting

The USART is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The USART is disabled by writing a zero to CTRLA.ENABLE.

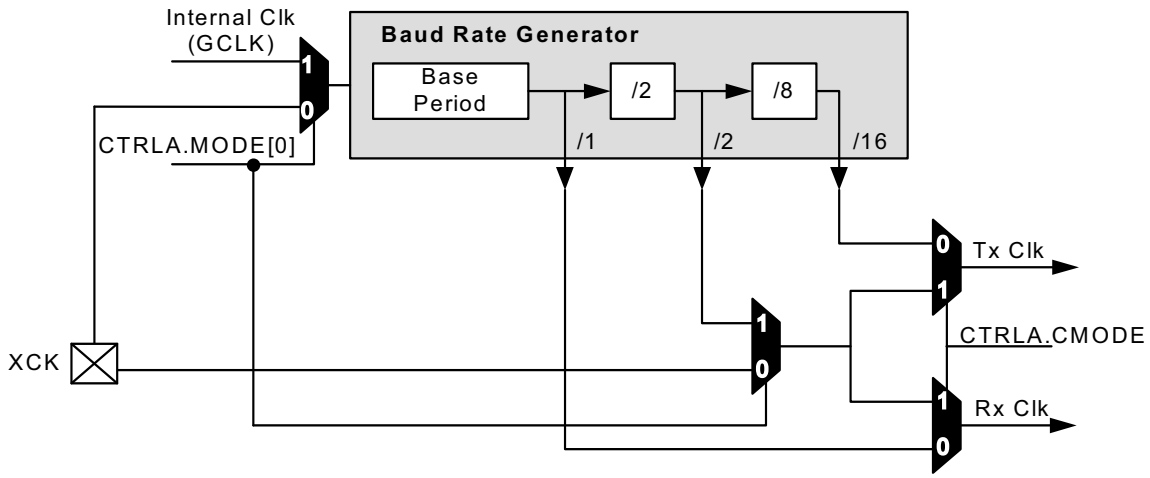
The USART is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the USART, except DBGCTRL, will be reset to their initial state, and the USART will be disabled. Refer to the CTRLA register for details.

24.6.2.3 Clock Generation and Selection

For both synchronous and asynchronous modes, the clock used for shifting and sampling data can be generated internally by the SERCOM baud-rate generator or supplied externally through the XCK line. Synchronous mode is selected by writing a one to the Communication Mode bit in the Control A register (CTRLA.CMODE) and asynchronous mode is selected by writing a zero to CTRLA.CMODE. The internal clock source is selected by writing 0x1 to the Operation Mode bit group in the Control A register (CTRLA.MODE) and the external clock source is selected by writing 0x0 to CTRLA.MODE.

The SERCOM baud-rate generator is configured as shown in Figure 24-3. When CTRLA.CMODE is zero, the baud-rate generator is automatically set to asynchronous mode and the 16-bit Baud register value is used. When CTRLA.CMODE is one, the baud-rate generator is automatically set to synchronous mode and the eight LSBs of the Baud register are used. Refer to “Clock Generation – Baud-Rate Generator” on page 327 for details on configuring the baud rate.

Figure 24-3. Clock Generation

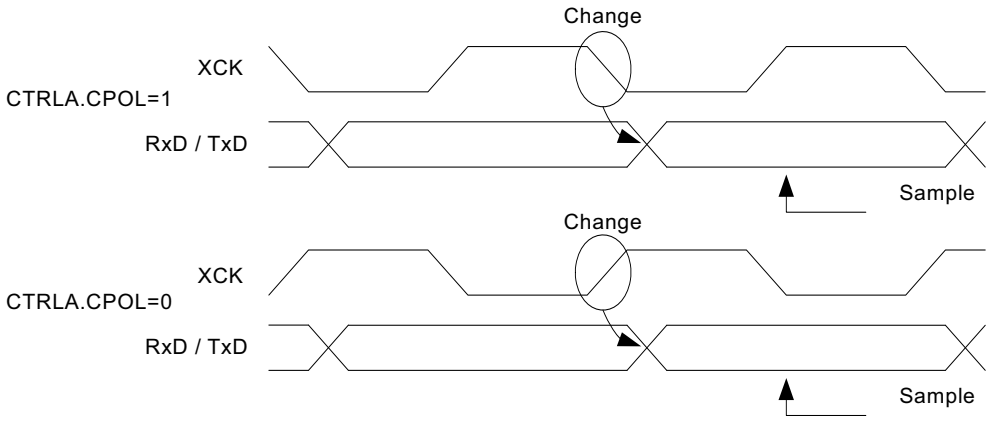


Synchronous Clock Operation

When synchronous mode is used, the CTRLA.MODE bit group controls whether the transmission clock (XCK line) is an input or output. The dependency between the clock edges and data sampling or data change is the same for internal and external clocks. Data input on the RxD pin is sampled at the opposite XCK clock edge as data is driven on the TxD pin.

The Clock Polarity bit in the Control A register (CTRLA.CPOL) selects which XCK clock edge is used for RxD sampling and which is used for TxD change. As shown in Figure 24-4, when CTRLA.CPOL is zero, the data will be changed on the rising XCK edge and sampled on the falling XCK edge. If CTRLA.CPOL is one, the data will be changed on the falling edge of XCK and sampled on the rising edge of XCK.

Figure 24-4. Synchronous Mode XCK Timing



When the clock is provided through XCK (CTRLA.MODE is 0x0), the shift registers operate directly on the XCK clock. This means that XCK is not synchronized with the system clock and, therefore, can operate at frequencies up to the system frequency.

24.6.2.4 Data Register

The USART Transmit Data register (TxDATA) and USART Receive Data register(RxDATA) share the same I/O address, referred to as the Data register (DATA). Writing the DATA register will update the Transmit Data register. Reading the DATA register will return the contents of the Receive Data register.

24.6.2.5 Data Transmission

A data transmission is initiated by loading the DATA register with the data to be sent. The data in TxDATA is moved to the shift register when the shift register is empty and ready to send a new frame. When the shift register is loaded with data, one complete frame will be transmitted.

The Transmit Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) is set, and the optional interrupt is generated, when the entire frame plus stop bit(s) have been shifted out and there is no new data written to the DATA register.

The DATA register should only be written when the Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set, which indicates that the register is empty and ready for new data.

Disabling the Transmitter

Disabling the transmitter will not become effective until any ongoing and pending transmissions are completed, i.e., when the transmit shift register and TxDATA do not contain data to be transmitted. The transmitter is disabled by writing a zero to the Transmitter Enable bit in the Control B register (CTRLB.TXEN).

24.6.2.6 Data Reception

The receiver starts data reception when a valid start bit is detected. Each bit that follows the start bit will be sampled at the baud rate or XCK clock, and shifted into the receive shift register until the first stop bit of a frame is received. When the first stop bit is received and a complete serial frame is present in the receive shift register, the contents of the shift register will be moved into the two-level receive buffer. The Receive Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set, and the optional interrupt is generated. A second stop bit will be ignored by the receiver.

The received data can be read by reading the DATA register. DATA should not be read unless the Receive Complete interrupt flag is set.

Disabling the Receiver

Disabling the receiver by writing a zero to the Receiver Enable bit in the Control B register (CTRLB.RXEN) will flush the two-level receive buffer, and data from ongoing receptions will be lost.

Error Bits

The USART receiver has three error bits. The Frame Error (FERR), Buffer Overflow (BUFOVF) and Parity Error (PERR) bits can be read from the Status (STATUS) register. Upon error detection, the corresponding bit will be set until it is cleared by writing a one to it. These bits are also automatically cleared when the receiver is disabled.

There are two methods for buffer overflow notification. When the immediate buffer overflow notification bit (CTRLA.IBON) is set, STATUS.BUFOVF is raised immediately upon buffer overflow. Software can then empty the receive FIFO by reading RxDATA until the receive complete interrupt flag (INTFLAG.RXC) goes low.

When CTRLA.IBON is zero, the buffer overflow condition travels with data through the receive FIFO. After the received data is read, STATUS.BUFOVF will be set along with INTFLAG.RXC.

Asynchronous Data Reception

The USART includes a clock recovery and data recovery unit for handling asynchronous data reception. The clock recovery logic is used to synchronize the incoming asynchronous serial frames at the RxD pin to the internally generated baud-rate clock. The data recovery logic samples and applies a low-pass filter to each incoming bit, thereby improving the noise immunity of the receiver. The asynchronous reception operational range depends on the accuracy of the internal baud-rate clock, the rate of the incoming frames and the frame size (in number of bits).

Asynchronous Operational Range

The operational range of the receiver depends on the difference between the received bit rate and the internally generated baud rate. If the baud rate of an external transmitter is too high or too low compared to the internally generated baud rate, the receiver will not be able to synchronize the frames to the start bit.

There are two possible sources for a mismatch in baud rate. The reference clock will always have some minor instability. In addition, the baud-rate generator can not always do an exact division of the reference clock frequency to get the baud

rate desired. In this case, the BAUD register value should be selected to give the lowest possible error. Refer to "Asynchronous Mode BAUD Value Selection" on page 328 for details.

Recommended maximum receiver baud-rate errors for various character sizes are shown in the table below.

Table 24-2. Asynchronous Receiver Error

D (Data bits + Parity)	R_{SLOW}(%)	R_{FAST}(%)	Max Total Error (%)	Recommended Max Rx Error (%)
5	94.12	107.69	+5.88/-7.69	±2.5
6	94.92	106.67	+5.08/-6.67	±2.0
7	95.52	105.88	+4.48/-5.88	±2.0
8	96.00	105.26	+4.00/-5.26	±2.0
9	96.39	104.76	+3.61/-4.76	±1.5
10	96.70	104.35	+3.30/-4.35	±1.5

The recommended maximum receiver baud-rate error assumes that the receiver and transmitter equally divide the maximum total error.

The following equations can be used to calculate the ratio of the incoming data rate and internal receiver baud rate:

$$R_{SLOW} = \frac{16(D+1)}{16(D+1)+6} \qquad R_{FAST} = \frac{16(D+2)}{16(D+1)+8}$$

where:

- D is the sum of character size and parity size (D = 5 to 10 bits)
- R_{SLOW} is the ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate
- R_{FAST} is the ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate

24.6.3 Additional Features

24.6.3.1 Parity

Even or odd parity can be selected for error checking by writing 0x1 to the Frame Format bit group in the Control A register (CTRLA.FORM). If even parity is selected by writing a zero to the Parity Mode bit in the Control B register (CTRLB.PMODE), the parity bit of the outgoing frame is set to one if the number of data bits that are one is odd (making the total number of ones even). If odd parity is selected by writing a one to CTRLB.PMODE, the parity bit of the outgoing frame is set to one if the number of data bits that are one is even (making the total number of ones odd).

When parity checking is enabled, the parity checker calculates the parity of the data bits in incoming frames and compares the result with the parity bit of the corresponding frame. If a parity error is detected, the Parity Error bit in the Status register (STATUS.PERR) is set.

24.6.3.2 Loop-back Mode

By configuring the Receive Data Pinout (CTRLA.RXPO) and Transmit Data Pinout (CTRLA.TXPO) to use the same data pins for transmit and receive, loop-back is achieved. The loop-back is through the pad, so the signal is also available externally.

24.6.4 Interrupts

The USART has the following interrupt sources:

- Receive Complete
- Transmit Complete
- Data Register Empty

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the USART is reset. See the register description for details on how to clear interrupt flags.

The USART has one common interrupt request line for all the interrupt sources. The user must read INTFLAG to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

24.6.5 Events

Not applicable.

24.6.6 Sleep Mode Operation

When using internal clocking, writing the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY) to one will allow GCLK_SERCOMx_CORE to be enabled in all sleep modes. Any interrupt can wake up the device.

When using external clocking, writing a one to CTRLA.RUNSTDBY will allow the Receive Complete interrupt.to wake up the device.

If CTRLA.RUNSTDBY is zero, the internal clock will be disabled when any ongoing transfer is finished. A Transfer Complete interrupt can wake up the device. When using external clocking, this will be disconnected when any ongoing transfer is finished, and all reception will be dropped.

24.6.7 Synchronization

Due to the asynchronicity between CLK_SERCOMx_APB and GCLK_SERCOMx_CORE, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)
- Receiver Enable bit in the Control B register (CTRLB.RXEN)
- Transmitter Enable bit in the Control B register (CTRLB.TXEN)

CTRLB.RXEN and CTRLB.TXEN behave somewhat differently than described above. Refer to CTRLB register description for details.

Synchronization is denoted by the Write-Synchronized property in the register description.

24.7 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
0x01		15:8								IBON
0x02		23:16			RXPO[1:0]					TXPO
0x03		31:24		DORD	CPOL	CMODE	FORM[3:0]			
0x04	CTRLB	7:0		SBMODE				CHSIZE[2:0]		
0x05		15:8			PMODE					
0x06		23:16							RXEN	TXEN
0x07		31:24								
0x08	DBGCTRL	7:0								DBGSTOP
0x09	Reserved									
0x0A	BAUD	7:0	BAUD[7:0]							
0x0B		15:8	BAUD[15:8]							
0x0C	INTENCLR	7:0						RXC	TXC	DRE
0x0D	INTENSET	7:0						RXC	TXC	DRE
0x0E	INTFLAG	7:0						RXC	TXC	DRE
0x0F	Reserved									
0x10	STATUS	7:0						BUFOVF	FERR	PERR
0x11		15:8	SYNCBUSY							
0x12	Reserved									
0x13	Reserved									
0x14	Reserved									
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	DATA	7:0	DATA[7:0]							
0x19		15:8								DATA[8]
0x1A	Reserved									
0x1B	Reserved									
0x1C	Reserved									
0x1D	Reserved									
0x1E	Reserved									
0x1F	Reserved									

24.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 334 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Synchronized property in each individual register description. Refer to “Synchronization” on page 340 for details.

Some registers are enable-protected, meaning they can only be written when the USART is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

24.8.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x00000000

Property: Enable-Protected, Write-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		DORD	CPOL	CMODE	FORM[3:0]			
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			RXPO[1:0]					TXPO
Access	R	R	R/W	R/W	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								IBON
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
Access	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 31 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 30 – DORD: Data Order**
  This bit indicates the data order when a character is shifted out from the Data register.
  0: MSB is transmitted first.
  1: LSB is transmitted first.
  This bit is not synchronized.

- **Bit 29 – CPOL: Clock Polarity**
  This bit indicates the relationship between data output change and data input sampling in synchronous mode.
  This bit is not synchronized.

Table 24-3. Clock Polarity

CPOL	TxD Change	RxD Sample
0x0	Rising XCK edge	Falling XCK edge
0x1	Falling XCK edge	Rising XCK edge

- **Bit 28 – CMODE: Communication Mode**
  This bit indicates asynchronous or synchronous communication.
  0: Asynchronous communication.
  1: Synchronous communication.
  This bit is not synchronized.

- **Bits 27:24 – FORM[3:0]: Frame Format**
  These bits define the frame format.
  These bits are not synchronized.

Table 24-4. Frame Format

FORM[3:0]	Description
0x0	USART frame
0x1	USART frame with parity
0x2-0xF	Reserved

- **Bits 23:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 21:20 – RXPO[1:0]: Receive Data Pinout**
  These bits define the receive data (RxD) pin configuration.
  These bits are not synchronized.

Table 24-5. Receive Data Pinout

RXPO[1:0]	Name	Description
0x0	PAD[0]	SERCOM PAD[0] is used for data reception
0x1	PAD[1]	SERCOM PAD[1] is used for data reception
0x2	PAD[2]	SERCOM PAD[2] is used for data reception
0x3	PAD[3]	SERCOM PAD[3] is used for data reception

- **Bits 19:17 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 16 – TXPO: Transmit Data Pinout**
  This bit defines the transmit data (TxD) and XCK pin configurations.
  This bit is not synchronized.

Table 24-6. Transmit Data Pinout

TXPO	TxD Pin Location	XCK Pin Location (When Applicable)
0x0	PAD[0]	PAD[1]
0x1	PAD[2]	PAD[3]

- **Bits 15:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 8 – IBON: Immediate Buffer Overflow Notification**
  This bit controls when the buffer overflow status bit (STATUS.BUFOVF) is asserted when a buffer overflow occurs.
  0: STATUS.BUFOVF is asserted when it occurs in the data stream.
  1: STATUS.BUFOVF is asserted immediately upon buffer overflow.
- **Bit 7 – RUNSTDBY: Run In Standby**
  This bit defines the functionality in standby sleep mode.
  This bit is not synchronized.

Table 24-7. Run In Standby

RUNSTDBY	External Clock	Internal Clock
0x0	External clock is disconnected when ongoing transfer is finished. All reception is dropped.	
0x1		Generic clock is enabled in all sleep modes. Any interrupt can wake up the device.

- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 4:2 – MODE: Operating Mode**

These bits must be written to 0x0 or 0x1 to select the USART serial communication interface of the SERCOM.

0x0: USART with external clock.

0x1: USART with internal clock.

These bits are not synchronized.

- **Bit 1 – ENABLE: Enable**

  0: The peripheral is disabled or being disabled.

  1: The peripheral is enabled or being enabled.

  Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY is cleared when the operation is complete.

  This bit is not enable-protected.

- **Bit 0 – SWRST: Software Reset**

  0: There is no reset operation ongoing.

  1: The reset operation is ongoing.

  Writing a zero to this bit has no effect.

  Writing a one to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

  Writing a one to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

  Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

  This bit is not enable-protected.

24.8.2 Control B

Name: CTRLB

Offset: 0x04

Reset: 0x00000000

Property: Enable-Protected, Write-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
							RXEN	TXEN
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
			PMODE					
Access	R	R	R/W	R	R	R	R/W	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		SBMODE				CHSIZE[2:0]		
Access	R	R/W	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:18 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 17 – RXEN: Receiver Enable**
  0: The receiver is disabled or being enabled.
  1: The receiver is enabled or will be enabled when the USART is enabled.
  Writing a zero to this bit will disable the USART receiver. Disabling the receiver will flush the receive buffer and clear the FERR, PERR and BUFOVF bits in the STATUS register.
  Writing a one to CTRLB.RXEN when the USART is disabled will set CTRLB.RXEN immediately. When the USART is enabled, CTRLB.RXEN will be cleared, and STATUS.SYNCBUSY will be set and remain set until the receiver is enabled. When the receiver is enabled, CTRLB.RXEN will read back as one.
  Writing a one to CTRLB.RXEN when the USART is enabled will set STATUS.SYNCBUSY, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as one.
  This bit is not enable-protected.

- **Bit 16 – TXEN: Transmitter Enable**
  0: The transmitter is disabled or being enabled.

1: The transmitter is enabled or will be enabled when the USART is enabled.

Writing a zero to this bit will disable the USART transmitter. Disabling the transmitter will not become effective until ongoing and pending transmissions are completed.

Writing a one to CTRLB.TXEN when the USART is disabled will set CTRLB.TXEN immediately. When the USART is enabled, CTRLB.TXEN will be cleared, and STATUS.SYNCBUSY will be set and remain set until the transmitter is enabled. When the transmitter is enabled, CTRLB.TXEN will read back as one.

Writing a one to CTRLB.TXEN when the USART is enabled will set STATUS.SYNCBUSY, which will remain set until the receiver is enabled, and CTRLB.TXEN will read back as one.

This bit is not enable-protected.

- **Bits 15:14 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 13 – PMODE: Parity Mode**
  This bit selects the type of parity used when parity is enabled (CTRLA.FORM is one). The transmitter will automatically generate and send the parity of the transmitted data bits within each frame. The receiver will generate a parity value for the incoming data and parity bit, compare it to the parity mode and, if a mismatch is detected, STATUS.PERR will be set.

  0: Even parity.

  1: Odd parity.

  This bit is not synchronized.

- **Bits 12:7 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 6 – SBMODE: Stop Bit Mode**
  This bit selects the number of stop bits transmitted.

  0: One stop bit.

  1: Two stop bits.

  This bit is not synchronized.

- **Bits 5:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – CHSIZE[2:0]: Character Size**
  These bits select the number of bits in a character.

These bits are not synchronized.

Table 24-8. Character Size

CHSIZE[2:0]	Description
0x0	8 bits
0x1	9 bits
0x2-0x4	Reserved
0x5	5 bits
0x6	6 bits
0x7	7 bits

24.8.3 Debug Control

Name: DBGCTRL

Offset: 0x08

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								DBGSTOP
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – DBGSTOP: Debug Stop Mode**
  This bit controls the baud-rate generator functionality when the CPU is halted by an external debugger.
  0: The baud-rate generator continues normal operation when the CPU is halted by an external debugger.
  1: The baud-rate generator is halted when the CPU is halted by an external debugger.

24.8.4 Baud

Name: BAUD

Offset: 0x0A

Reset: 0x0000

Property: Enable-Protected, Write-Protected

Bit	15	14	13	12	11	10	9	8
	BAUD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BAUD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – BAUD: Baud Value**
  These bits control the clock generation, as described in the SERCOM Baud Rate section.

24.8.5 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						RXC	**TXC**	**DRE**
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RXC: Receive Complete Interrupt Enable**
  0: Receive Complete interrupt is disabled.
  1: Receive Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

- **Bit 1 – TXC: Transmit Complete Interrupt Enable**
  0: Transmit Complete interrupt is disabled.
  1: Transmit Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Transmit Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

- **Bit 0 – DRE: Data Register Empty Interrupt Enable**
  0: Data Register Empty interrupt is disabled.
  1: Data Register Empty interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

24.8.6 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR) .

Name: INTENSET

Offset: 0x0D

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						RXC	**TXC**	**DRE**
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RXC: Receive Complete Interrupt Enable**
  0: Receive Complete interrupt is disabled.
  1: Receive Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

- **Bit 1– TXC: Transmit Complete Interrupt Enable**
  0: Transmit Complete interrupt is disabled.
  1: Transmit Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

- **Bit 0 – DRE: Data Register Empty Interrupt Enable**
  0: Data Register Empty interrupt is disabled.
  1: Data Register Empty interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

24.8.7 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x0E

Reset: 0x00

Property:

Bit	7	6	5	4	3	2	1	0
						RXC	TXC	DRE
Access	R	R	R	R	R	R	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- --**Bit 2 – RXC: Receive Complete**
  This flag is cleared by reading the Data register (DATA) or by disabling the receiver.
  This flag is set when there are unread data in DATA.
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.

- **Bit 1 – TXC: Transmit Complete**
  This flag is cleared by writing a one to it or by writing new data to DATA.
  This flag is set when the entire frame in the transmit shift register has been shifted out and there are no new data in DATA.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the flag.

- **Bit 0 – DRE: Data Register Empty**
  This flag is cleared by writing new data to DATA.
  This flag is set when DATA is empty and ready to be written.
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.

24.8.8 Status

Name: STATUS

Offset: 0x10

Reset: 0x0000

Property:

Bit	15	14	13	12	11	10	9	8
	SYNCBUSY							
Access	R/W	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						BUFOVF	FERR	PERR
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 14:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – BUFOVF: Buffer Overflow**
  Reading this bit before reading the Data register will indicate the error status of the next character to be read.
  This bit is cleared by writing a one to the bit or by disabling the receiver.
  This bit is set when a buffer overflow condition is detected. A buffer overflow occurs when the receive buffer is full, there is a new character waiting in the receive shift register and a new start bit is detected.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear it.

- **Bit 1 – FERR: Frame Error**
  Reading this bit before reading the Data register will indicate the error status of the next character to be read.
  This bit is cleared by writing a one to the bit or by disabling the receiver.
  This bit is set if the received character had a frame error, i.e., when the first stop bit is zero.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear it.

- **Bit 0 – PERR: Parity Error**
  Reading this bit before reading the Data register will indicate the error status of the next character to be read.
  This bit is cleared by writing a one to the bit or by disabling the receiver.
  This bit is set if parity checking is enabled (CTRLA.FORM is one) and a parity error is detected.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear it.

24.8.9 Data

Name: DATA

Offset: 0x18

Reset: 0x0000

Property: -

Bit	15	14	13	12	11	10	9	8
								DATA[8]
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 8:0 – DATA[8:0]: Data**
  Reading these bits will return the contents of the Receive Data register. The register should be read only when the Receive Complete Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set. The status bits in STATUS should be read before reading the DATA value in order to get any corresponding error.

  Writing these bits will write the Transmit Data register. This register should be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

25. SERCOM SPI – SERCOM Serial Peripheral Interface

25.1 Overview

The serial peripheral interface (SPI) is one of the available modes in the Serial Communication Interface (SERCOM). Refer to "SERCOM – Serial Communication Interface" on page 324 for details.

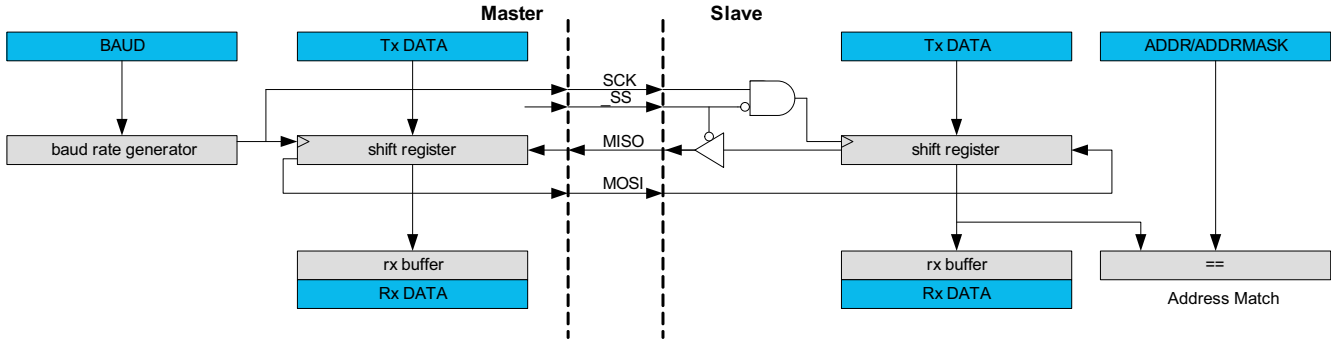
The SPI uses the SERCOM transmitter and receiver configured as shown in "Full-Duplex SPI Master Slave Interconnection" on page 357. Each side, master and slave, depicts a separate SPI containing a shift register, a transmit buffer and two receive buffers. In addition, the SPI master uses the SERCOM baud-rate generator, while the SPI slave can use the SERCOM address match logic. Fields shown in capital letters are synchronous to CLK_SERCOMx_APB and accessible by the CPU, while fields with lowercase letters are synchronous to the SCK clock.

25.2 Features

- Full-duplex, four-wire interface (MISO, MOSI, SCK, _SS)
- Single-buffered transmitter, double-buffered receiver
- Supports all four SPI modes of operation
- Single data direction operation allows alternate function on MISO or MOSI pin
- Selectable LSB- or MSB-first data transfer
- Master operation:
  - Serial clock speed up to half the system clock
  - 8-bit clock generator
- Slave operation:
  - Serial clock speed up to the system clock
  - Optional 8-bit address match operation
  - Operation in all sleep modes

25.3 Block Diagram

Figure 25-1. Full-Duplex SPI Master Slave Interconnection



25.4 Signal Description

Signal Name	Type	Description
PAD[3:0]	Digital I/O	General SERCOM pins

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped to one of several pins.

25.5 Product Dependencies

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

25.5.1 I/O Lines

Using the SERCOM's I/O lines requires the I/O pins to be configured using port configuration (PORT). Refer to "PORT" on page 276 for details.

When the SERCOM is configured for SPI operation, the pins should be configured according to Table 25-1. If the receiver is disabled, the data input pin can be used for other purposes. In master mode the slave select line (_SS) is controlled by software.

Table 25-1. SPI Pin Configuration

Pin	Master SPI	Slave SPI
MOSI	Output	Input
MISO	Input	Output
SCK	Output	Input
_SS	User defined output enable	Input

The combined configuration of PORT and the Data In/Data Out and Data Out Pinout bit groups in Control A register will define the physical position of the SPI signals in Table 25-1.

25.5.2 Power Management

The SPI can continue to operate in any sleep mode. The SPI interrupts can be used to wake up the device from sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

25.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_SERCOMx_APB can be found in the Peripheral Clock Masking section in the "PM – Power Manager" on page 100.

A generic clock (GCLK_SERCOMx_CORE) is required to clock the SPI. This clock must be configured and enabled in the Generic Clock Controller before using the SPI. Refer to "GCLK – Generic Clock Controller" on page 78 for details.

This generic clock is asynchronous to the bus clock (CLK_SERCOMx_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to "Synchronization" on page 365 for further details.

25.5.4 DMA

Not applicable.

25.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the SPI, interrupts requires the Interrupt Controller to be configured first. Refer to "Nested Vector Interrupt Controller" on page 24 for details.

25.5.6 Events

Not applicable.

25.5.7 Debug Operation

When the CPU is halted in debug mode, the SPI continues normal operation. If the SPI 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. The SPI can be forced to halt operation during debugging. Refer to the Debug Control (DBGCTRL) register for details.

25.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Clear and Status register (INTFLAG)
- Status register (STATUS)
- Data register (DATA)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

25.5.9 Analog Connections

Not applicable.

25.6 Functional Description

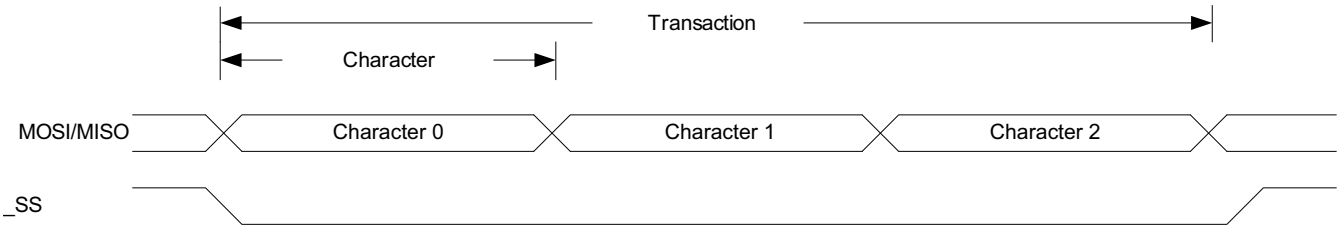
25.6.1 Principle of Operation

The SPI is a high-speed synchronous data transfer interface. It allows fast communication between the device and peripheral devices.

The SPI can operate as master or slave. As master, the SPI initiates and controls all data transactions. The SPI is single buffered for transmitting and double buffered for receiving. When transmitting data, the Data register can be loaded with the next character to be transmitted while the current transmission is in progress. For receiving, this means that the data is transferred to the two-level receive buffer upon reception, and the receiver is ready for a new character.

The SPI transaction format is shown in Figure 25-2, where each transaction can contain one or more characters. The character size is configurable, and can be either 8 or 9 bits.

Figure 25-2. SPI Transaction Format



The SPI master must initiate a transaction by pulling low the slave select line (_SS) of the desired slave. The master and slave prepare data to be sent in their respective shift registers, and the master generates the serial clock on the SCK line. Data are always shifted from master to slave on the master output, slave input line (MOSI), and from slave to master on the master input, slave output line (MISO). The master signals the end of the transaction by pulling the _SS line high.

As each character is shifted out from the master, another character is shifted in from the slave.

25.6.2 Basic Operation

25.6.2.1 Initialization

The following registers are enable-protected, meaning that they can only be written when the SPI is disabled (CTRL.ENABLE is zero):

- Control A register (CTRLA), except Enable (CTRLA.ENABLE) and Software Reset (CTRLA.SWRST)
- Control B register (CTRLB), except Receiver Enable (RXEN)
- Baud register (BAUD)
- Address register (ADDR)

Any writes to these registers when the SPI is enabled or is being enabled (CTRL.ENABLE is one) will be discarded. Writes to these registers while the SPI is being disabled will be completed after the disabling is complete.

Enable-protection is denoted by the Enable-Protection property in the register description.

Before the SPI is enabled, it must be configured, as outlined by the following steps:

- SPI mode in master or slave operation must be selected by writing 0x2 or 0x3 to the Operating Mode bit group in the Control A register (CTRLA.MODE)
- Transfer mode must be selected by writing the Clock Polarity bit and the Clock Phase bit in the Control A register (CTRLA.CPOL and CTRLA.CPHA)
- Transaction format must be selected by writing the Frame Format bit group in the Control A register (CTRLA.FORM)
- SERCOM pad to use for the receiver must be selected by writing the Data In Pinout bit in the Control A register (CTRLA.DIPO)
- SERCOM pads to use for the transmitter, slave select and serial clock must be selected by writing the Data Out Pinout bit group in the Control A register (CTRLA.DOPO)
- Character size must be selected by writing the Character Size bit in the Control B register (CTRLB.CHSIZE)
- Data direction must be selected by writing the Data Order bit in the Control A register (CTRLA.DORD)
- If the SPI is used in master mode, the Baud register (BAUD) must be written to generate the desired baud rate
- The receiver can be enabled by writing a one to the Receiver Enable bit in the Control B register (CTRLB.RXEN)

25.6.2.2 Enabling, Disabling and Resetting

The SPI is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The SPI is disabled by writing a zero to CTRLA.ENABLE.

The SPI is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the SPI, except DBGCTRL, will be reset to their initial state, and the SPI will be disabled. Refer to CTRLA for details.

25.6.2.3 Clock Generation

In SPI master operation (CTRLA.MODE is 0x3), the serial clock (SCK) is generated internally using the SERCOM baud-rate generator. When used in SPI mode, the baud-rate generator is set to synchronous mode, and the 8-bit Baud register (BAUD) value is used to generate SCK, clocking the shift register. Refer to “Clock Generation – Baud-Rate Generator” on page 327 for more details.

In SPI slave operation (CTRLA.MODE is 0x2), the clock is provided by an external master on the SCK pin. This clock is used to directly clock the SPI shift register.

25.6.2.4 Data Register

The SPI Transmit Data register (TxDATA) and SPI Receive Data register (RxDATA) share the same I/O address, referred to as the SPI Data register (DATA). Writing the DATA register will update the Transmit Data register. Reading the DATA register will return the contents of the Receive Data register.

25.6.2.5 SPI Transfer Modes

There are four combinations of SCK phase and polarity with respect to the serial data. The SPI data transfer modes are shown in Table 25-2 and Figure 25-3. SCK phase is selected by the Clock Phase bit in the Control A register

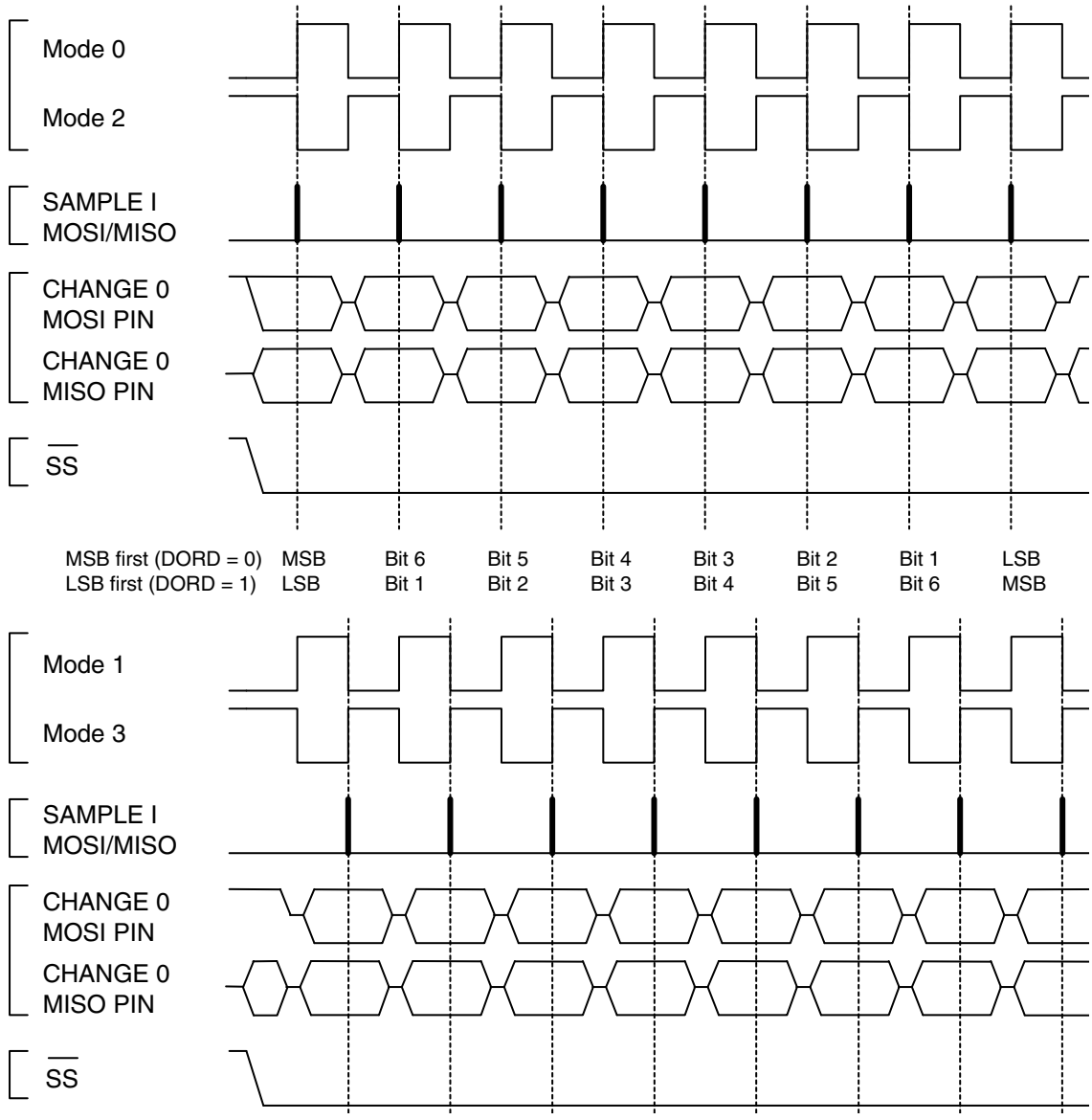
(CTRLA.CPHA). SCK polarity is selected by the Clock Polarity bit in the Control A register (CTRLA.CPOL). Data bits are shifted out and latched in on opposite edges of the SCK signal, ensuring sufficient time for the data signals to stabilize.

Table 25-2. SPI Transfer Modes

Mode	CPOL	CPHA	Leading Edge	Trailing Edge
0	0	0	Rising, sample	Falling, setup
1	0	1	Rising, setup	Falling, sample
2	1	0	Falling, sample	Rising, setup
3	1	1	Falling, setup	Rising, sample

Leading edge is the first clock edge in a clock cycle, while trailing edge is the second clock edge in a clock cycle.

Figure 25-3. SPI Transfer Modes



25.6.2.6 Transferring Data

Master

When configured as a master (CTRLA.MODE is 0x3), the _SS line can be located at any general purpose I/O pin, and must be configured as an output. When the SPI is ready for a data transaction, software must pull the _SS line low.

When writing a character to the Data register (DATA), the character will be transferred to the shift register when the shift register is empty. Once the contents of TxDATA have been transferred to the shift register, the Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set, and a new character can be written to DATA.

As each character is shifted out from the master, another character is shifted in from the slave. If the receiver is enabled (CTRLA.RXEN is one), the contents of the shift register will be transferred to the two-level receive buffer. The transfer takes place in the same clock cycle as the last data bit is shifted in, and the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) will be set. The received data can be retrieved by reading DATA.

When the last character has been transmitted and there is no valid data in DATA, the Transmit Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) is set. When the transaction is finished, the master must indicate this to the slave by pulling the _SS line high.

Slave

When configured as a slave (CTRLA.MODE is 0x2), the SPI interface will remain inactive, with the MISO line tri-stated as long as the _SS pin is pulled high. Software may update the contents of DATA at any time, as long as the Data Register Empty flag in the Interrupt Status and Clear register (INTFLAG.DRE) is set.

When _SS is pulled low and SCK is running, the slave will sample and shift out data according to the transaction mode set. When the contents of TxDATA have been loaded into the shift register, INTFLAG.DRE is set, and new data can be written to DATA. Similar to the master, the slave will receive one character for each character transmitted. On the same clock cycle as the last data bit of a character is received, the character will be transferred into the two-level receive buffer. The received character can be retrieved from DATA when INTFLAG.RCX is set.

When the master pulls the _SS line high, the transaction is done and the Transmit Complete Interrupt flag in the Interrupt Flag Status and Clear register (TXC) is set.

Once DATA is written, it takes three SCK clocks to load the shift register. After the DATA register is empty, it takes three CLK_SERCOM_APB cycles for INTFLAG.DRE to be set.

25.6.2.7 Receiver Error Bit

The SPI receiver has one error bit: the Buffer Overflow bit (BUFOVF), which can be read from the Status register (STATUS). Upon error detection, the bit will be set until it is cleared by writing a one to it. The bit is also automatically cleared when the receiver is disabled.

There are two methods for buffer overflow notification. When the immediate buffer overflow notification bit (CTRLA.IBON) is set, STATUS.BUFOVF is raised immediately upon buffer overflow. Software can then empty the receive FIFO by reading RxDATA until the receive complete interrupt flag (INTFLAG.RXC) goes low.

When CTRLA.IBON is zero, the buffer overflow condition travels with data through the receive FIFO. After the received data is read, STATUS.BUFOVF will be set along with INTFLAG.RXC, and RxDATA will be zero.

25.6.3 Additional Features

25.6.3.1 Address Recognition

When the SPI is configured for slave operation (CTRLA.MODE is 0x2) with address recognition (CTRLA.FORM is 0x2), the SERCOM address recognition logic is enabled. When address recognition is enabled, the first character in a transaction is checked for an address match. If there is a match, then the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set, the MISO output is enabled and the transaction is processed. If there is no match, the transaction is ignored.

If the device is in sleep mode, an address match can wake up the device in order to process the transaction. If the address does not match, then the complete transaction is ignored. If a 9-bit frame format is selected, only the lower 8 bits of the shift register are checked against the Address register (ADDR).

Refer to “Address Match and Mask” on page 330 for further details.

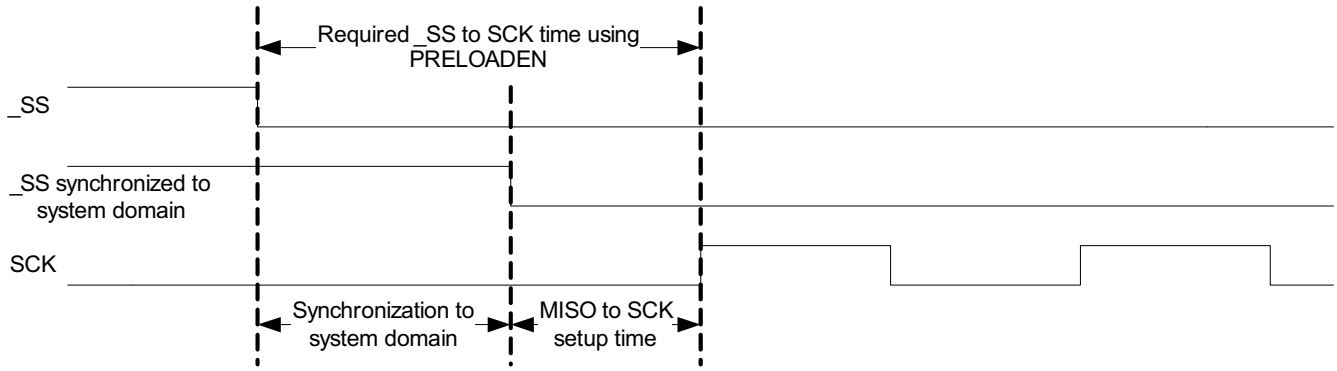
25.6.3.2 Preloading of the Slave Shift Register

When starting a transaction, the slave will first transmit the contents of the shift register before loading new data from DATA. The first character sent can be either the reset value of the shift register (if this is the first transmission since the last reset) or the last character in the previous transmission. Preloading can be used to preload data to the shift register while _SS is high and eliminate sending a dummy character when starting a transaction.

In order to guarantee enough set-up time before the first SCK edge, enough time must be given between _SS going low and the first SCK sampling edge, as shown in Figure 25-4.

Preloading is enabled by setting the Slave Data Preload Enable bit in the Control B register (CTRLB.PLOADEN).

Figure 25-4. Timing Using Preloading

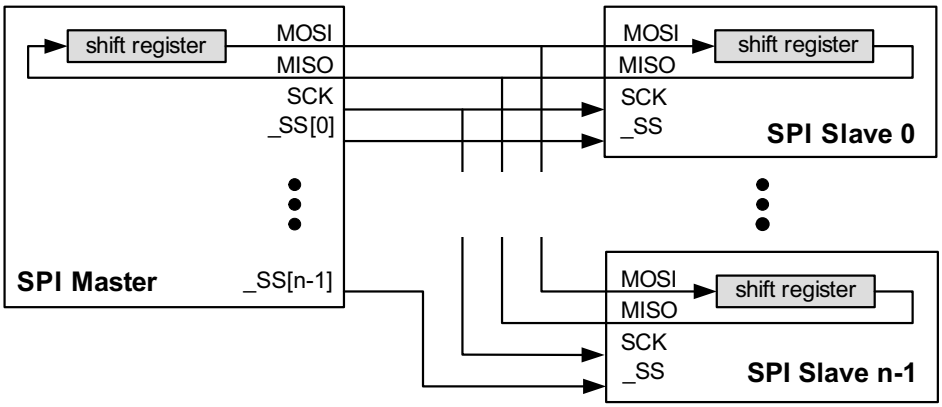


Only one data character written to DATA will be preloaded into the shift register while the synchronized _SS signal (see Figure 25-4) is high. The next character written to DATA before _SS is pulled low will be stored in DATA until transfer begins. If the shift register is not preloaded, the current contents of the shift register will be shifted out.

25.6.3.3 Master with Several Slaves

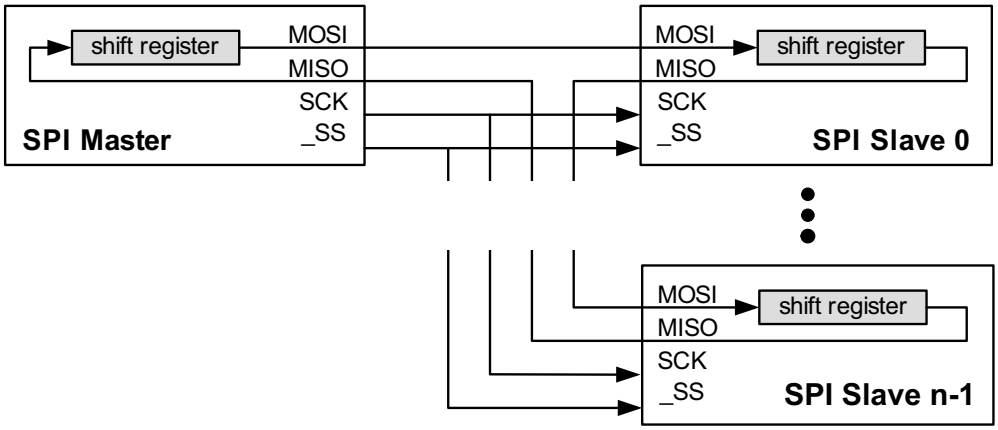
If the bus consists of several SPI slaves, an SPI master can use general purpose I/O pins to control the _SS line to each of the slaves on the bus, as shown in Figure 25-5. In this configuration, the single selected SPI slave will drive the tri-state MISO line.

Figure 25-5. Multiple Slaves in Parallel



An alternate configuration is shown in Figure 25-6. In this configuration, all n attached slaves are connected in series. A common _SS line is provided to all slaves, enabling them simultaneously. The master must shift n characters for a complete transaction.

Figure 25-6. Multiple Slaves in Series



25.6.3.4 Loop-back Mode

By configuring the Data In Pinout (CTRLA.DIPO) and Data Out Pinout (CTRLA.DOPO) to use the same data pins for transmit and receive, loop-back is achieved. The loop-back is through the pad, so the signal is also available externally.

25.6.4 Interrupts

The SPI has the following interrupt sources:

- Receive Complete
- Transmit Complete
- Data Register Empty

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the SPI is reset. See the register description for details on how to clear interrupt flags.

The SPI has one common interrupt request line for all the interrupt sources. The user must read INTFLAG to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

For details on clearing interrupt flags, refer to INTFLAG.

25.6.5 Events

Not applicable.

25.6.6 Sleep Mode Operation

During master operation, the generic clock will continue to run in idle sleep mode. If the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY) is one, the GCLK_SERCOM_CORE will also be enabled in standby sleep mode. Any interrupt can wake up the device.

If CTRLA.RUNSTDBY is zero during master operation, GLK_SERCOMx_CORE will be disabled when the ongoing transaction is finished. Any interrupt can wake up the device.

During slave operation, writing a one to CTRLA.RUNSTDBY will allow the Receive Complete interrupt to wake up the device.

If CTRLA.RUNSTDBY is zero during slave operation, all reception will be dropped, including the ongoing transaction.

25.6.7 Synchronization

Due to the asynchronicity between CLK_SERCOMx_APB and GCLK_SERCOMx_CORE, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)
- Receiver Enable bit in the Control B register (CTRLB.RXEN)

CTRLB.RXEN behaves somewhat differently than described above. Refer to CTRLB for details.

Write-synchronization is denoted by the Write-Synchronized property in the register description.

25.7 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
0x01		15:8								IBON
0x02		23:16			DIPO[1:0]				DOPO[1:0]	
0x03		31:24		DORD	CPOL	CPHA	FORM[3:0]			
0x04	CTRLB	7:0		PLOADEN				CHSIZE[2:0]		
0x05		15:8	AMODE[1:0]							
0x06		23:16							RXEN	
0x07		31:24								
0x08	DBGCTRL	7:0								DBGSTOP
0x09	Reserved									
0x0A	BAUD	7:0	BAUD[7:0]							
0x0B	Reserved									
0x0C	INTENCLR	7:0						RXC	TXC	DRE
0x0D	INTENSET	7:0						RXC	TXC	DRE
0x0E	INTFLAG	7:0						RXC	TXC	DRE
0x0F	Reserved									
0x10	STATUS	7:0						BUFOVF		
0x11		15:8	SYNCBUSY							
0x12	Reserved									
0x13	Reserved									
0x14	ADDR	7:0	ADDR[7:0]							
0x15		15:8								
0x16		23:16	ADDRMASK[7:0]							
0x17		31:24								
0x18	DATA	7:0	DATA[7:0]							
0x19		15:8								DATA[8]
0x1A	Reserved									
0x1B	Reserved									
0x1C	Reserved									
0x1D	Reserved									
0x1E	Reserved									
0x1F	Reserved									

25.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 359 for details.

Some registers require synchronization when read and/or written. Write-synchronization is denoted by the Write-Synchronized property in each individual register description. Refer to “Synchronization” on page 365 for details.

Some registers are enable-protected, meaning they can only be written when the USART is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

25.8.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x00000000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		DORD	CPOL	CPHA	FORM[3:0]			
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			DIPO[1:0]				DOPO[1:0]	
Access	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								IBON
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
Access	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 31 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 30 – DORD: Data Order**
  This bit indicates the data order when a character is shifted out from the Data register.
  0: MSB is transferred first.
  1: LSB is transferred first.
  This bit is not synchronized.

- **Bit 29 – CPOL: Clock Polarity**
  In combination with the Clock Phase bit (CPHA), this bit determines the SPI transfer mode.
  0: SCK is low when idle. The leading edge of a clock cycle is a rising edge, while the trailing edge is a falling edge.
  1: SCK is high when idle. The leading edge of a clock cycle is a falling edge, while the trailing edge is a rising edge.
  This bit is not synchronized.

- **Bit 28 – CPHA: Clock Phase**
  In combination with the Clock Polarity bit (CPOL), this bit determines the SPI transfer mode.

0: The data is sampled on a leading SCK edge and changed on a trailing SCK edge.

1: The data is sampled on a trailing SCK edge and changed on a leading SCK edge.

This bit is not synchronized.

Table 25-3. SPI Transfer Modes

Mode	CPOL	CPHA	Leading Edge	Trailing Edge
0x0	0	0	Rising, sample	Falling, change
0x1	0	1	Rising, change	Falling, sample
0x2	1	0	Falling, sample	Rising, change
0x3	1	1	Falling, change	Rising, sample

- **Bits 27:24 – FORM[3:0]: Frame Format**
  Table 25-4 shows the various frame formats supported by the SPI. When a frame format with address is selected, the first byte received is checked against the ADDR register.

Table 25-4. Frame Format

FORM[3:0]	Name	Description
0x0	SPI	SPI frame
0x1	-	Reserved
0x2	SPI_ADDR	SPI frame with address
0x3-0xF	-	Reserved

- **Bits 23:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 21:20 – DIPO[1:0]: Data In Pinout**
  These bits define the data in (DI) pad configurations.

  In master operation, DI is MISO.

  In slave operation, DI is MOSI.

  These bits are not synchronized.

Table 25-5. Data In Pinout

DIPO[1:0]	Name	Description
0x0	PAD[0]	SERCOM PAD[0] is used as data input
0x1	PAD[1]	SERCOM PAD[1] is used as data input
0x2	PAD[2]	SERCOM PAD[2] is used as data input
0x3	PAD[3]	SERCOM PAD[3] is used as data input

- **Bits 19:18 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 17:16 – DOPO: Data Out Pinout**
  This bit defines the available pad configurations for data out (DO) and the serial clock (SCK). In slave operation, the slave select line (_SS) is controlled by DOPO, while in master operation the _SS line is controlled by the port configuration.

  In master operation, DO is MOSI.

  In slave operation, DO is MISO.

  These bits are not synchronized.

Table 25-6. Data Out Pinout

DOPO	DO	SCK	Slave _SS	Master _SS
0x0	PAD[0]	PAD[1]	PAD[2]	System configuration
0x1	PAD[2]	PAD[3]	PAD[1]	System configuration
0x2	PAD[3]	PAD[1]	PAD[2]	System configuration
0x3	PAD[0]	PAD[3]	PAD[1]	System configuration

- **Bits 15:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 8 – IBON: Immediate Buffer Overflow Notification**
  This bit controls when the buffer overflow status bit (STATUS.BUFOVF) is asserted when a buffer overflow occurs.

  0: STATUS.BUFOVF is asserted when it occurs in the data stream.

  1: STATUS.BUFOVF is asserted immediately upon buffer overflow.

  This bit is not synchronized.

- **Bit 7 – RUNSTDBY: Run In Standby**
  This bit defines the functionality in standby sleep mode.

  These bits are not synchronized.

Table 25-7. Run In Standby Configuration

RUNSTDBY	Slave	Master
0x0	Disabled. All reception is dropped, including the ongoing transaction.	Generic clock is disabled when ongoing transaction is finished. All interrupts can wake up the device.
0x1	Wake on Receive Complete interrupt.	Generic clock is enabled while in sleep modes. All interrupts can wake up the device.

- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 4:2 – MODE: Operating Mode**
  These bits must be written to 0x2 or 0x3 to select the SPI serial communication interface of the SERCOM.

  0x2: SPI slave operation

  0x3: SPI master operation

  These bits are not synchronized.

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled or being disabled.

1: The peripheral is enabled or being enabled.

Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY is cleared when the operation is complete.

This bit is not enable-protected.

- **Bit 0 – SWRST: Software Reset**

  0: There is no reset operation ongoing.

  1: The reset operation is ongoing.

  Writing a zero to this bit has no effect.

  Writing a one to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

  Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

  Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

  This bit is not enable-protected.

25.8.2 Control B

Name: CTRLB

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
							RXEN	
Access	R/W	R/W	R/W	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	AMODE[1:0]							
Access	R/W	R/W	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		PLOADEN				CHSIZE[2:0]		
Access	R	R/W	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:18 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 17 – RXEN: Receiver Enable**
  0: The receiver is disabled or being enabled.
  1: The receiver is enabled or it will be enabled when SPI is enabled.
  Writing a zero to this bit will disable the SPI receiver immediately. The receive buffer will be flushed, data from ongoing receptions will be lost and STATUS.BUFOVF will be cleared.
  Writing a one to CTRLB.RXEN when the SPI is disabled will set CTRLB.RXEN immediately. When the SPI is enabled, CTRLB.RXEN will be cleared, STATUS.SYNCBUSY will be set and remain set until the receiver is enabled. When the receiver is enabled CTRLB.RXEN will read back as one.
  Writing a one to CTRLB.RXEN when the SPI is enabled will set STATUS.SYNCBUSY, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as one.
  This bit is not enable-protected.

- **Bit 16 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 15:14 – AMODE: Address Mode**
  These bits set the slave addressing mode when the frame format (CTRLA.FORM) with address is used. They are unused in master mode.

Table 25-8. Address Mode

AMODE[1:0]	Name	Description
0x0	MASK	ADDRMASK is used as a mask to the ADDR register
0x1	2_ADDRS	The slave responds to the two unique addresses in ADDR and ADDRMASK
0x2	RANGE	The slave responds to the range of addresses between and including ADDR and ADDRMASK. ADDR is the upper limit
0x3		Reserved

- **Bits 13:7 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 6 – PLOADEN: Slave Data Preload Enable**
  Setting this bit will enable preloading of the slave shift register when there is no transfer in progress. If the _SS line is high when DATA is written, it will be transferred immediately to the shift register.

- **Bits 5:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – CHSIZE[2:0]: Character Size**

Table 25-9. Character Size

CHSIZE[2:0]	Name	Description
0x0	8BIT	8 bits
0x1	9BIT	9 bits
0x2-0x7		Reserved

25.8.3 Debug Control

Name: DBGCTRL

Offset: 0x08

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								DBGSTOP
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – DBGSTOP: Debug Stop Mode**
  This bit controls the functionality when the CPU is halted by an external debugger.
  0: The baud-rate generator continues normal operation when the CPU is halted by an external debugger.
  1: The baud-rate generator is halted when the CPU is halted by an external debugger.

25.8.4 Baud Rate

Name: BAUD

Offset: 0x0A

Reset: 0x00

Property: Write-Protected, Enable-Protected

Bit	7	6	5	4	3	2	1	0
	BAUD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:0 – BAUD: Baud Register**
  These bits control the clock generation, as described in the SERCOM “Clock Generation – Baud-Rate Generator” on page 327.

25.8.5 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						RXC	TXC	DRE
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RXC: Receive Complete Interrupt Enable**
  0: Receive Complete interrupt is disabled.
  1: Receive Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

- **Bit 1 – TXC: Transmit Complete Interrupt Enable**
  0: Transmit Complete interrupt is disabled.
  1: Transmit Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Transmit Complete Interrupt Enable bit, which disable the Transmit Complete interrupt.

- **Bit 0 – DRE: Data Register Empty Interrupt Enable**
  0: Data Register Empty interrupt is disabled.
  1: Data Register Empty interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

25.8.6 Interrupt Enable Set

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x0D

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						RXC	**TXC**	**DRE**
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RXC: Receive Complete Interrupt Enable**
  0: Receive Complete interrupt is disabled.
  1: Receive Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

- **Bit 1 – TXC: Transmit Complete Interrupt Enable**
  0: Transmit Complete interrupt is disabled.
  1: Transmit Complete interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

- **Bit 0 – DRE: Data Register Empty Interrupt Enable**
  0: Data Register Empty interrupt is disabled.
  1: Data Register Empty interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

25.8.7 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x0E

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
						RXC	TXC	DRE
Access	R	R	R	R	R	R	R/W	R
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RXC: Receive Complete**
  This flag is cleared by reading the Data (DATA) register or by disabling the receiver.
  This flag is set when there are unread data in the receive buffer. If address matching is enabled, the first data received in a transaction will be an address.
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.

- **Bit 1 – TXC: Transmit Complete**
  This flag is cleared by writing a one to it or by writing new data to DATA.
  In master mode, this flag is set when the data have been shifted out and there are no new data in DATA.
  In slave mode, this flag is set when the _SS pin is pulled high. If address matching is enabled, this flag is only set if the transaction was initiated with an address match.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the flag.

- **Bit 0 – DRE: Data Register Empty**
  This flag is cleared by writing new data to DATA.
  This flag is set when DATA is empty and ready for new data to transmit.
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.

25.8.8 Status

Name: STATUS

Offset: 0x10

Reset: 0x0000

Property: –

Bit	15	14	13	12	11	10	9	8
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						BUFOVF		
Access	R	R	R	R	R	R/W	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is in progress.

- **Bits 14:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – BUFOVF: Buffer Overflow**
  Reading this bit before reading DATA will indicate the error status of the next character to be read.
  This bit is cleared by writing a one to the bit or by disabling the receiver.
  This bit is set when a buffer overflow condition is detected. An overflow condition occurs if the two-level receive buffer is full when the last bit of the incoming character is shifted into the shift register. All characters shifted into the shift registers before the overflow condition is eliminated by reading DATA will be lost.
  When set, the corresponding RxDATA will be 0.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear it.

- **Bits 1:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

25.8.9 Address

Name: ADDR

Offset: 0x14

Reset: 0x00000000

Property: Write-Protected, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRMASK[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:24 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 23:16 – ADDRMASK[7:0]: Address Mask**
  These bits hold the address mask when the transaction format (CTRLA.FORM) with address is used.
- **Bits 15:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:0 – ADDR[7:0]: Address**
  These bits hold the address when the transaction format (CTRLA.FORM) with address is used.

25.8.10 Data

Name: DATA

Offset: 0x18

Reset: 0x0000

Property: –

Bit	15	14	13	12	11	10	9	8
								DATA[8]
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 8:0 – DATA[8:0]: Data**
  Reading these bits will return the contents of the receive data buffer. The register should be read only when the Receive Complete Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set.
  Writing these bits will write the transmit data buffer. This register should be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

26. SERCOM I²C – SERCOM Inter-Integrated Circuit

26.1 Overview

The inter-integrated circuit (I²C) interface is one of the available modes in the serial communication interface (SERCOM). Refer to "SERCOM – Serial Communication Interface" on page 324 for details.

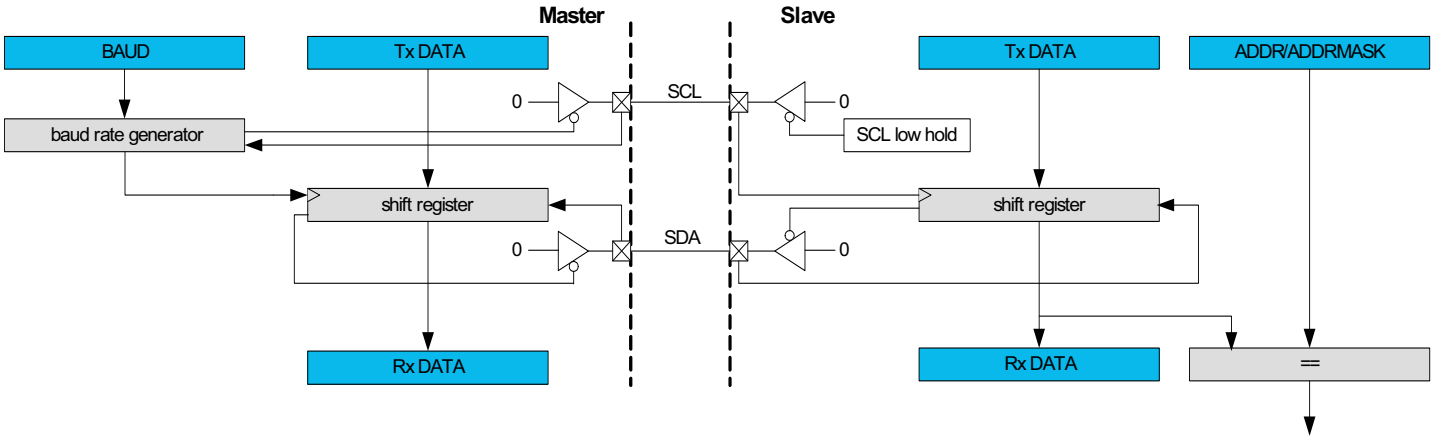
The I²C interface uses the SERCOM transmitter and receiver configured as shown in Figure 26-1. Fields shown in capital letters are registers accessible by the CPU, while lowercase fields are internal to the SERCOM. Each side, master and slave, depicts a separate I²C interface containing a shift register, a transmit buffer and a receive buffer. In addition, the I²C master uses the SERCOM baud-rate generator, while the I²C slave uses the SERCOM address match logic.

26.2 Features

- Master or slave operation
- Philips I²C compatible
- SMBus™ compatible
- 100kHz and 400kHz support at low system clock frequencies
- Physical interface includes:
  - Slew-rate limited outputs
  - Filtered inputs
- Slave operation:
  - Operation in all sleep modes
  - Wake-up on address match
  - Address match in hardware for:
    - 7-bit unique address and/or 7-bit general call address
    - 7-bit address range
    - Two unique 7-bit addresses

26.3 Block Diagram

Figure 26-1. I²C Single-Master Single-Slave Interconnection



26.4 Signal Description

Signal Name	Type	Description
PAD[0]	Digital I/O	SDA
PAD[1]	Digital I/O	SCL
PAD[2]	Digital I/O	SDA_OUT (4-wire)
PAD[3]	Digital I/O	SDC_OUT (4-wire)

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins. Note that not all the pins are I^2C pins. Refer to Table 5-1 for details on the pin type for each pin.

26.5 Product Dependencies

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

26.5.1 I/O Lines

Using the SERCOM's I/O lines requires the I/O pins to be configured. Refer to "PORT" on page 276 for details.

26.5.2 Power Management

The I^2C will continue to operate in any sleep mode where the selected source clock is running. I^2C interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

26.5.3 Clocks

The SERCOM bus clock (CLK_SERCOMx_APB, where i represents the specific SERCOM instance number) is enabled by default, and can be enabled and disabled in the Power Manager. Refer to "PM – Power Manager" on page 100 for details.

The SERCOM bus clock (CLK_SERCOMx_APB) is enabled by default, and can be enabled and disabled in the Power Manager. Refer to "PM – Power Manager" on page 100 for details.

Two generic clocks are used by the SERCOM (GCLK_SERCOMx_CORE and GCLK_SERCOM_SLOW). The core clock (GCLK_SERCOMx_CORE) is required to clock the SERCOM while operating as a master, while the slow clock (GCLK_SERCOM_SLOW) is required only for certain functions. These clocks must be configured and enabled in the Generic Clock Controller (GCLK) before using the SERCOM. Refer to "GCLK – Generic Clock Controller" on page 78 for details.

These generic clocks are asynchronous to the SERCOM bus clock (CLK_SERCOMx_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to the "Synchronization" on page 395 section for further details.

26.5.4 DMA

Not applicable.

26.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the I^2C interrupts requires the Interrupt Controller to be configured first. Refer to "Nested Vector Interrupt Controller" on page 24 for details.

26.5.6 Events

Not applicable.

26.5.7 Debug Operation

When the CPU is halted in debug mode, the I^{2}C interface continues normal operation. If the I^{2}C interface 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. The I^{2}C interface can be forced to halt operation during debugging.

Refer to the DBGCTRL register for details.

26.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Interrupt Flag Status and Clear register (INTFLAG)
- Status register (STATUS)
- Address register (ADDR)
- Data register (DATA)

Write-protection is denoted by the Write-Protected property in the register description.

Write-protection does not apply to accesses through en external debugger. Refer to"PAC – Peripheral Access Controller" on page 27 for details.

26.5.9 Analog Connections

Not applicable.

26.6 Functional Description

26.6.1 Principle of Operation

The I^{2}C interface uses two physical lines for communication:

- Serial Data Line (SDA) for packet transfer
- Serial Clock Line (SCL) for the bus clock

A transaction starts with the start condition, followed by a 7-bit address and a direction bit (read or write) sent from the I^{2}C master. The addressed I^{2}C slave will then acknowledge (ACK) the address, and data packet transactions can commence. Every 9-bit data packet consists of 8 data bits followed by a one-bit reply indicating whether the data was acknowledged or not. In the event that a data packet is not acknowledged (NACK), whether sent from the I^{2}C slave or master, it will be up to the I^{2}C master to either terminate the connection by issuing the stop condition, or send a repeated start if more data is to be transceived.

Figure 26-2 illustrates the possible transaction formats and Figure 26-3 explains the legend used.

Figure 26-2. Basic I²C Transaction Diagram

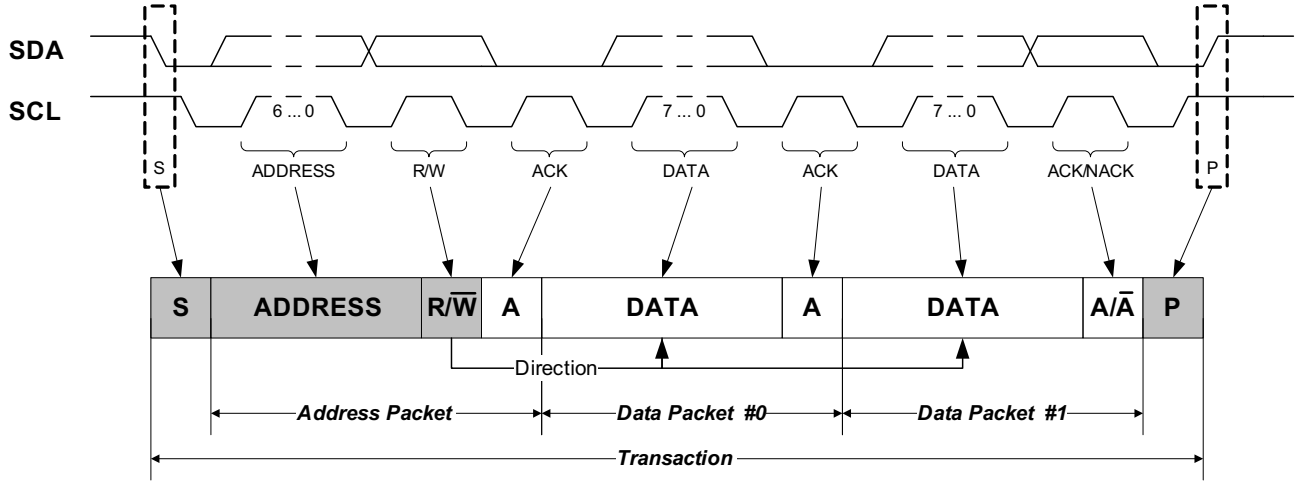
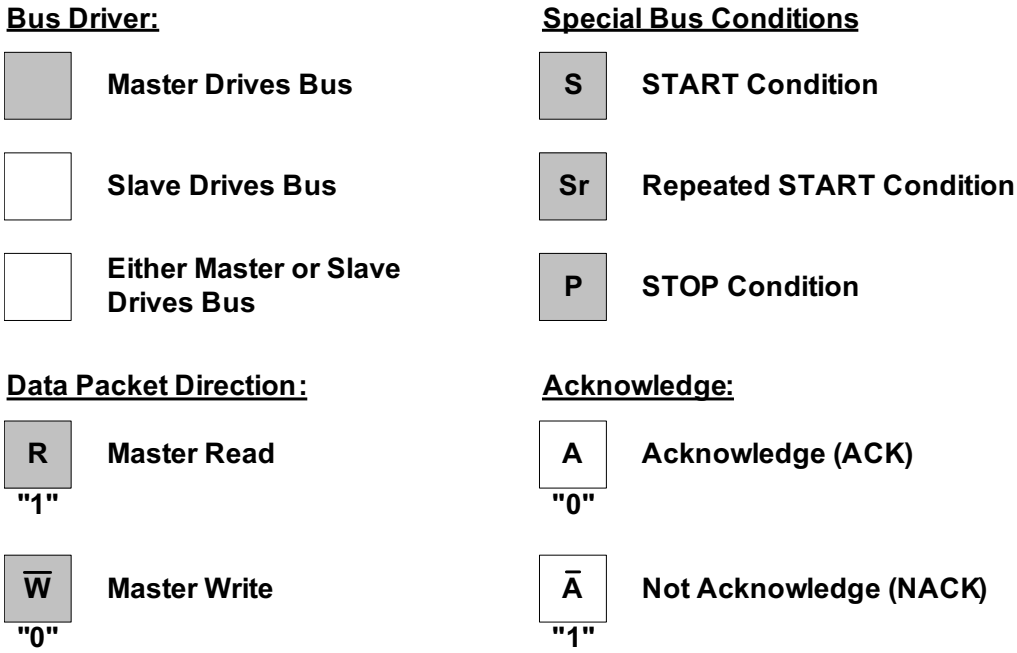


Figure 26-3. Transaction Diagram Syntax



26.6.2 Basic Operation

26.6.2.1 Initialization

The following registers are enable-protected, meaning they can be written only when the I²C interface is disabled (CTRLA.ENABLE is zero):

- Control A register (CTRLA), except Enable (CTRLA.ENABLE) and Software Reset (CTRLA.SWRST)
- Control B register (CTRLB), except Acknowledge Action (CTRLB.ACKACT) and Command (CTRLB.CMD)
- Baud Rate register (BAUD)
- Address register (ADDR) while in slave operation

Any writes to these bits or registers when the I²C interface is enabled or is being enabled (CTRLA.ENABLE is one) will be discarded. Writes to these registers while the I²C interface is being disabled will be completed after the disabling is complete.

Enable-protection is denoted by the Enable-Protection property in the register description.

Before the I²C interface is enabled, it must be configured as outlined by the following steps:

I²C mode in master or slave operation must be selected by writing 0x4 or 0x5 to the Operating Mode bit group in the Control A register (CTRLA.MODE)

- SCL low time-out can be enabled by writing to the SCL Low Time-Out bit in the Control A register (CTRLA.LOWTOUT)
- In master operation, the inactive bus time-out can be set in the Inactive Time-Out bit group in the Control A register (CTRLA.INACTOUT)
- Hold time for SDA can be set in the SDA Hold Time bit group in the Control A register (CTRLA.SDAHOLD)
- Smart operation can be enabled by writing to the Smart Mode Enable bit in the Control B register (CTRLB.SMEN)
- In slave operation, the address match configuration must be set in the Address Mode bit group in the Control B register (CTRLB.AMODE)
- In slave operation, the addresses must be set, according to the selected address configuration, in the Address and Address Mask bit groups in the Address register (ADDR.ADDR and ADDR.ADDRMASK)
- In master operation, the Baud Rate register (BAUD) must be written to generate the desired baud rate

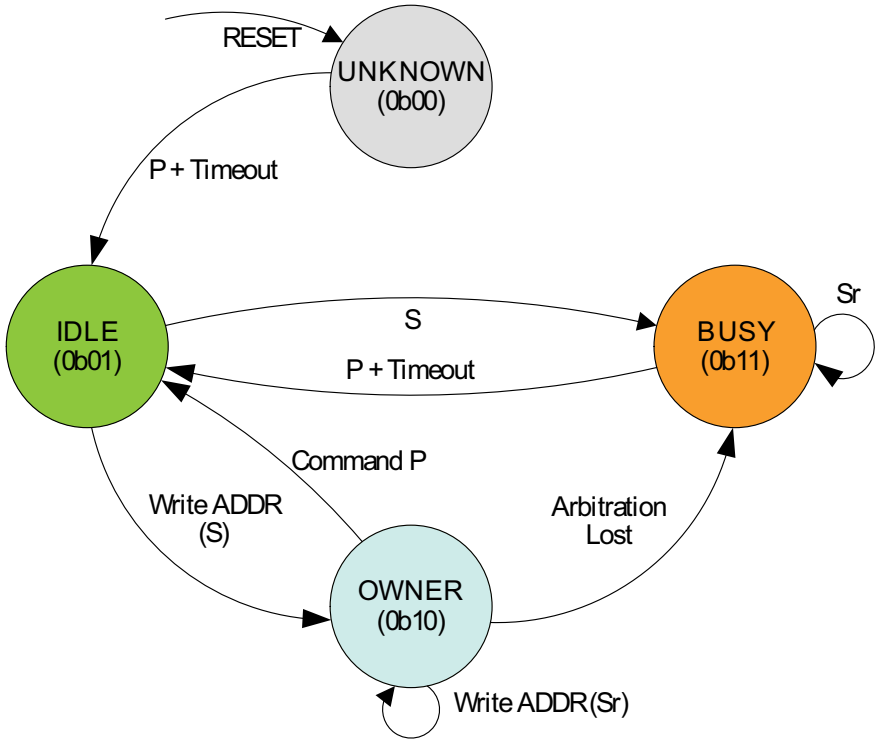
26.6.2.2 Enabling, Disabling and Resetting

The I²C interface is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The I²C interface is disabled by writing a zero to CTRLA.ENABLE. The I²C interface is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the I²C interface, except DBGCTRL, will be reset to their initial state, and the I²C interface will be disabled. Refer to CTRLA for details.

26.6.2.3 I²C Bus State Logic

The bus state logic includes several logic blocks that continuously monitor the activity on the I²C bus lines in all sleep modes. The start and stop detectors and the bit counter are all essential in the process of determining the current bus state. The bus state is determined according to the state diagram shown in Figure 26-4. Software can get the current bus state by reading the Master Bus State bits in the Status register (STATUS.BUSSTATE). The value of STATUS.BUSSTATE in the figure is shown in binary.

Figure 26-4. Bus State Diagram



The bus state machine is active when the I^{2}C master is enabled. After the I^{2}C master has been enabled, the bus state is unknown. From the unknown state, the bus state machine can be forced to enter the idle state by writing to STATUS.BUSSTATE accordingly. However, if no action is taken by software, the bus state will become idle if a stop condition is detected on the bus. If the inactive bus time-out is enabled, the bus state will change from unknown to idle on the occurrence of a time-out. Note that after a known bus state is established, the bus state logic will not re-enter the unknown state from either of the other states.

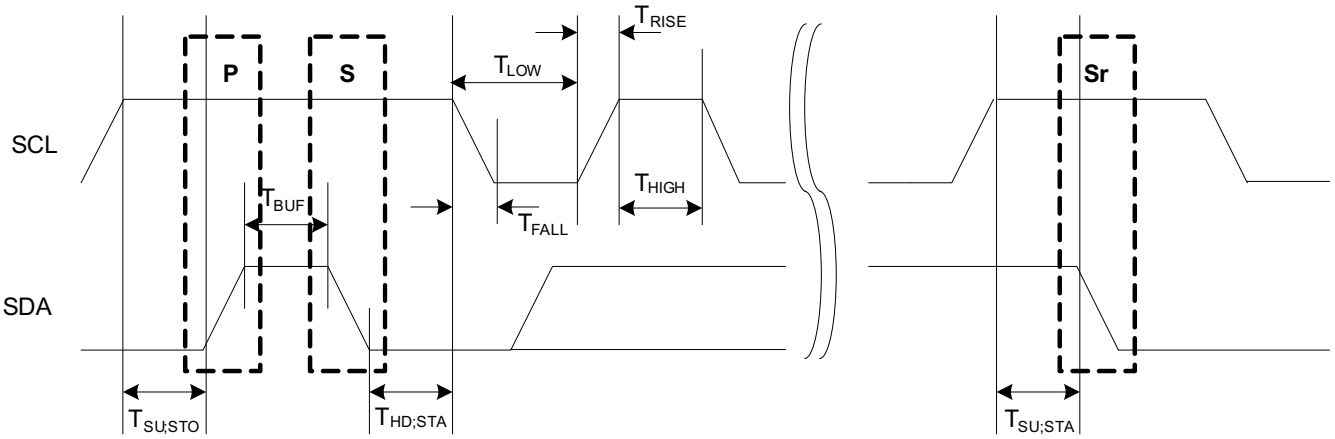
When the bus is idle it is ready for a new transaction. If a start condition is issued on the bus by another I^{2}C master in a multimaster setup, the bus becomes busy until a stop condition is detected. The stop condition will cause the bus to re-enter the IDLE state. If the inactive bus time-out (SMBus) is enabled, the bus state will change from busy to idle on the occurrence of a time-out. If a start condition is generated internally by writing the Address bit group in the Address register (ADDR.ADDR) while in idle state, the owner state is entered. If the complete transaction was performed without interference, i.e., arbitration not lost, the I^{2}C master is allowed to issue a stop condition, which in turn will cause a change of the bus state back to idle. However, if a packet collision is detected when in the owner state, the arbitration is assumed lost and the bus state becomes busy until a stop condition is detected.

A repeated start condition will change the bus state only if arbitration is lost while issuing a repeated start.

26.6.2.4 Clock Generation

The Master I^{2}C clock (SCL) frequency is determined by a number of factors. The low (T_{LOW}) and high ($T_{\_HIGH}$) times are determined by the Baud Rate register (BAUD), while the rise (T_{RISE}) and fall (T_{FALL}) times are determined by the bus topology. Because of the wired-AND logic of the bus, T_{FALL} will be considered as part of T_{LOW}. Likewise, T_{RISE} will be in a state between T_{LOW} and T_{HIGH} until a high state has been detected.

Figure 26-5. SCL Timing



The following parameters are timed using the SCL low time period. This comes from the Master Baud Rate Low bit group in the Baud Rate register (BAUD.BAUDLOW) when non-zero, or the Master Baud Rate bit group in the Baud Rate register (BAUD.BAUD) when BAUD.BAUDLOW is zero.

- T_{LOW} – Low period of SCL clock
- $T_{SU;STO}$ – Set-up time for stop condition
- T_{BUF} – Bus free time between stop and start conditions
- $T_{HD;STA}$ – Hold time (repeated) start condition
- $T_{SU;STA}$ – Set-up time for repeated start condition
- T_{HIGH} is timed using the SCL high time count from BAUD.BAUD
- T_{RISE} is determined by the bus impedance; for internal pull-ups. Refer to “Electrical Characteristics” on page 558 for details.
- T_{FALL} is determined by the open-drain current limit and bus impedance; can typically be regarded as zero. Refer to “Electrical Characteristics” on page 558 for details.

The SCL frequency is given by:

$$f_{SCL} = \frac{1}{T_{LOW} + T_{HIGH} + T_{RISE}}$$

When BAUD.BAUDLOW is zero, the BAUD.BAUD value is used to time both SCL high and SCL low. In this case the following formula will give the SCL frequency:

$$f_{SCL} = \frac{f_{GCLK}}{2(5 + BAUD) + f_{GCLK} T_{RISE}}$$

When BAUD.BAUDLOW is non-zero, the following formula is used to determine the SCL frequency:

$$f_{SCL} = \frac{f_{GCLK}}{10 + BAUD + BAUDLOW + f_{GCLK} T_{RISE}}$$

When BAUDLOW is non-zero, the following formula can be used to determine the SCL frequency:

$$f_{SCL} = \frac{f_{GCLK}}{10 + BAUD + BAUDLOW + f_{GCLK} T_{RISE}}$$

The following formulas can be used to determine the SCL T_{LOW} and T_{HIGH} times:

$$T_{low} = \frac{BAUD.BAUDLOW + 5}{f_{GCLK}}$$

$$T_{HIGH} = \frac{BAUD.BAUD + 5}{f_{GCLK}}$$

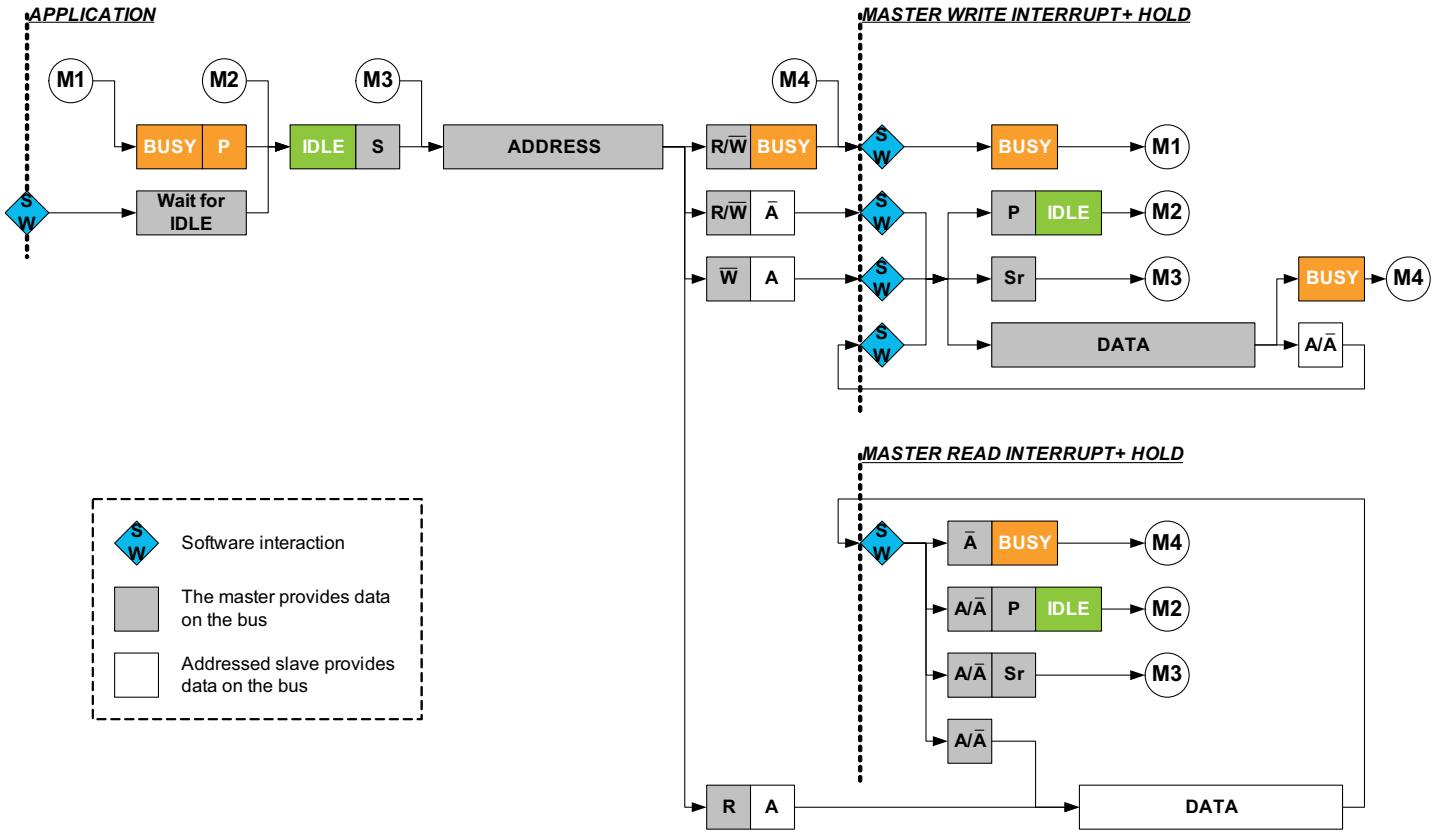
26.6.2.5 I²C Master Operation

The I²C master is byte-oriented and interrupt based. The number of interrupts generated is kept at a minimum by automatic handling of most events. Auto-triggering of operations and a special smart mode, which can be enabled by writing a one to the Smart Mode Enable bit in the Control A register (CTRLA.SMEN), are included to reduce software driver complexity and code size.

The I²C master operates according to the behavior diagram shown in Figure 26-6. The circles with a capital letter M followed by a number (M1, M2... etc.) indicate which node in the figure the bus logic can jump to based on software or hardware interaction.

This diagram is used as reference for the description of the I²C master operation throughout the document.

Figure 26-6. I²C Master Behavioral Diagram



Transmitting Address Packets

The I²C master starts a bus transaction by writing ADDR.ADDR with the I²C slave address and the direction bit. If the bus is busy, the I²C master will wait until the bus becomes idle before continuing the operation. When the bus is idle, the I²C master will issue a start condition on the bus. The I²C master will then transmit an address packet using the address written to ADDR.ADDR.

After the address packet has been transmitted by the I²C master, one of four cases will arise, based on arbitration and transfer direction.

Case 1: Arbitration lost or bus error during address packet transmission

If arbitration was lost during transmission of the address packet, the Master on Bus bit in the Interrupt Flag register (INTFLAG.MB) and the Arbitration Lost bit in the Status register (STATUS.ARBLOST) are both set. Serial data output to SDA is disabled, and the SCL is released, which disables clock stretching. In effect the I²C master is no longer allowed to perform any operation on the bus until the bus is idle again. A bus error will behave similarly to the arbitration lost condition. In this case, the MB interrupt flag and Master Bus Error bit in the Status register (STATUS.BUSERR) are both set in addition to STATUS.ARBLOST.

The Master Received Not Acknowledge bit in the Status register (STATUS.RXNACK) will always contain the last successfully received acknowledge or not acknowledge indication.

In this case, software will typically inform the application code of the condition and then clear the interrupt flag before exiting the interrupt routine. No other flags have to be cleared at this point, because all flags will be cleared automatically the next time the ADDR.ADDR register is written.

Case 2: Address packet transmit complete – No ACK received

If no I²C slave device responds to the address packet, then the INTFLAG.MB interrupt flag is set and STATUS.RXNACK is set. The clock hold is active at this point, preventing further activity on the bus.

The missing ACK response can indicate that the I²C slave is busy with other tasks or sleeping and, therefore, not able to respond. In this event, the next step can be either issuing a stop condition (recommended) or resending the address packet by using a repeated start condition. However, the reason for the missing acknowledge can be that an invalid I²C slave address has been used or that the I²C slave is for some reason disconnected or faulty. If using SMBus logic, the slave must ACK the address, and hence no action means the slave is not available on the bus.

Case 3: Address packet transmit complete – Write packet, Master on Bus set

If the I²C master receives an acknowledge response from the I²C slave, INTFLAG.MB is set and STATUS.RXNACK is cleared. The clock hold is active at this point, preventing further activity on the bus.

In this case, the software implementation becomes highly protocol dependent. Three possible actions can enable the I²C operation to continue. The three options are:

- The data transmit operation is initiated by writing the data byte to be transmitted into DATA.DATA.
- Transmit a new address packet by writing ADDR.ADDR. A repeated start condition will automatically be inserted before the address packet.
- Issue a stop condition, consequently terminating the transaction.

Case 4: Address packet transmit complete – Read packet, Slave on Bus set

If the I²C master receives an ACK from the I²C slave, the I²C master proceeds to receive the next byte of data from the I²C slave. When the first data byte is received, the Slave on Bus bit in the Interrupt Flag register (INTFLAG.SB) is set and STATUS.RXNACK is cleared. The clock hold is active at this point, preventing further activity on the bus.

In this case, the software implementation becomes highly protocol dependent. Three possible actions can enable the I²C operation to continue. The three options are:

- Let the I²C master continue to read data by first acknowledging the data received. This is automatically done when reading DATA.DATA if the smart mode is enabled.
- Transmit a new address packet.
- Terminate the transaction by issuing a stop condition.

An ACK or NACK will be automatically transmitted for the last two alternatives if smart mode is enabled. The Acknowledge Action bit in the Control B register (CTRLB.ACKACT) determines whether ACK or NACK should be sent.

Transmitting Data Packets

When an address packet with direction set to write (STATUS.DIR is zero) has been successfully transmitted, INTFLAG.MB will be set and the I²C master can start transmitting data by writing to DATA.DATA. The I²C master transmits data via the I²C bus while continuously monitoring for packet collisions. If a collision is detected, the I²C master looses arbitration and STATUS.ARBLOST is set. If the transmit was successful, the I²C master automatically receives an ACK bit from the I²C slave and STATUS.RXNACK will be cleared. INTFLAG.MB will be set in both cases, regardless of arbitration outcome.

Testing STATUS.ARBLOST and handling the arbitration lost condition in the beginning of the I²C Master on Bus interrupt is recommended. This can be done, as there is no difference between handling address and data packet arbitration.

STATUS.RXNACK must be checked for each data packet transmitted before the next data packet transmission can commence. The I²C master is not allowed to continue transmitting data packets if a NACK is given from the I²C slave.

Receiving Data Packets

When INTFLAG.SB is set, the I²C master will already have received one data packet. The I²C master must respond by sending either an ACK or NACK. Sending a NACK might not be successfully executed as arbitration can be lost during the transmission. In this case, a loss of arbitration will cause INTFLAG.SB to not be set on completion. Instead, INTFLAG.MB will be used to indicate a change in arbitration. Handling of lost arbitration is the same as for data bit transmission.

26.6.2.6 I²C Slave Operation

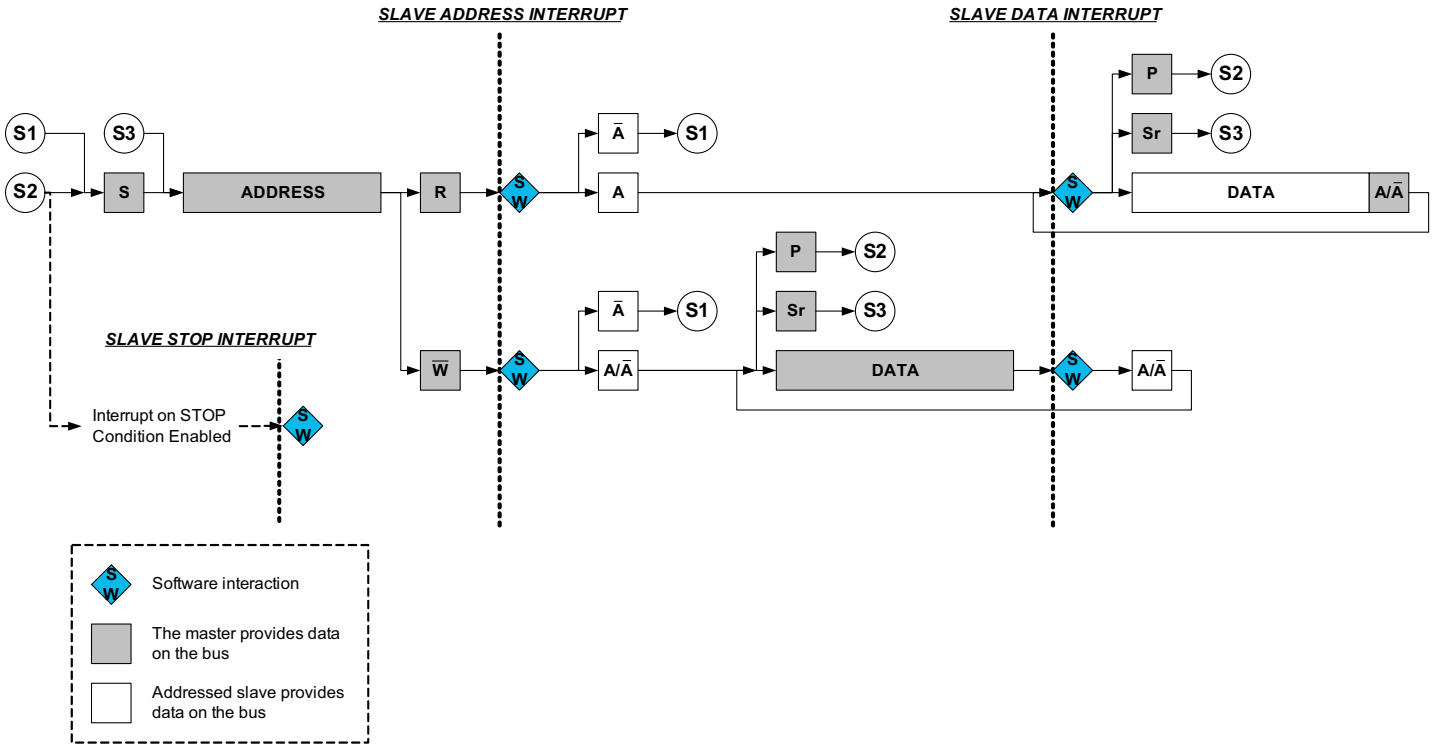
The I²C slave is byte-oriented and interrupt-based. The number of interrupts generated is kept at a minimum by automatic handling of most events. Auto triggering of operations and a special smart mode, which can be enabled by

writing a 1 to the Smart Mode Enable bit in the Control A register (CTRLA.SMEN), are included to reduce software's complexity and code size.

The I²C slave operates according to the behavior diagram shown in Figure 26-7. The circles with a capital S followed by a number (S1, S2... etc.) indicate which node in the figure the bus logic can jump to based on software or hardware interaction.

This diagram is used as reference for the description of the I²C slave operation throughout the document.

Figure 26-7. I²C Slave Behavioral Diagram



Receiving Address Packets

When the I²C slave is properly configured, it will wait for a start condition to be detected. When a start condition is detected, the successive address packet will be received and checked by the address match logic. If the received address is not a match, the packet is rejected and the I²C slave waits for a new start condition. The I²C slave Address Match bit in the Interrupt Flag register (INTFLAG.AMATCH) is set when a start condition followed by a valid address packet is detected. SCL will be stretched until the I²C slave clears INTFLAG.AMATCH. Because the I²C slave holds the clock by forcing SCL low, the software is given unlimited time to respond to the address.

The direction of a transaction is determined by reading the Read / Write Direction bit in the Status register (STATUS.DIR), and the bit will be updated only when a valid address packet is received.

If the Transmit Collision bit in the Status register (STATUS.COLL) is set, this indicates that the last packet addressed to the I²C slave had a packet collision. A collision causes the SDA and SCL lines to be released without any notification to software. The next AMATCH interrupt is, therefore, the first indication of the previous packet's collision. Collisions are intended to follow the SMBus Address Resolution Protocol (ARP).

After the address packet has been received from the I²C master, one of two cases will arise based on transfer direction.

Case 1: Address packet accepted – Read flag set

The STATUS.DIR bit is one, indicating an I²C master read operation. The SCL line is forced low, stretching the bus clock. If an ACK is sent, I²C slave hardware will set the Data Ready bit in the Interrupt Flag register (INTFLAG.DRDY),

indicating data are needed for transmit. If not acknowledge is sent, the I²C slave will wait for a new start condition and address match.

Typically, software will immediately acknowledge the address packet by sending an ACK/NACK bit. The I²C slave command CTRLB.CMD = 3 can be used for both read and write operation as the command execution is dependent on the STATUS.DIR bit.

Writing a one to INTFLAG.AMATCH will also cause an ACK/NACK to be sent corresponding to the CTRLB.ACKACT bit.

Case 2: Address packet accepted – Write flag set

The STATUS.DIR bit is cleared, indicating an I²C master write operation. The SCL line is forced low, stretching the bus clock. If an ACK is sent, the I²C slave will wait for data to be received. Data, repeated start or stop can be received.

If not acknowledge is sent, the I²C slave will wait for a new start condition and address match.

Typically, software will immediately acknowledge the address packet by sending an ACK/NACK bit. The I²C slave command CTRLB.CMD = 3 can be used for both read and write operation as the command execution is dependent on STATUS.DIR.

Writing a one to INTFLAG.AMATCH will also cause an ACK/NACK to be sent corresponding to the CTRLB.ACKACT bit.

Receiving and Transmitting Data Packets

After the I²C slave has received an address packet, it will respond according to the direction either by waiting for the data packet to be received or by starting to send a data packet by writing to DATA.DATA. When a data packet is received or sent, INTFLAG.DRDY will be set. Then, if the I²C slave was receiving data, it will send an acknowledge according to CTRLB.ACKACT.

Case 1: Data received

INTFLAG.DRDY is set, and SCL is held low pending SW interaction.

Case 2: Data sent

When a byte transmission is successfully completed, the INTFLAG.DRDY interrupt flag is set. If NACK is received, the I²C slave must expect a stop or a repeated start to be received. The I²C slave must release the data line to allow the I²C master to generate a stop or repeated start.

Upon stop detection, the Stop Received bit in the Interrupt Flag register (INTFLAG.PREC) will be set and the I²C slave will return to the idle state.

26.6.3 Additional Features

26.6.3.1 SMBus

The I²C hardware incorporates hardware SCL low time-out, which allows a time-out to occur if the clock line is held low too long. This time-out is driven by the GCLK_SERCOM_SLOW clock. The I²C interface also allows for a SMBus compatible SDA hold time.

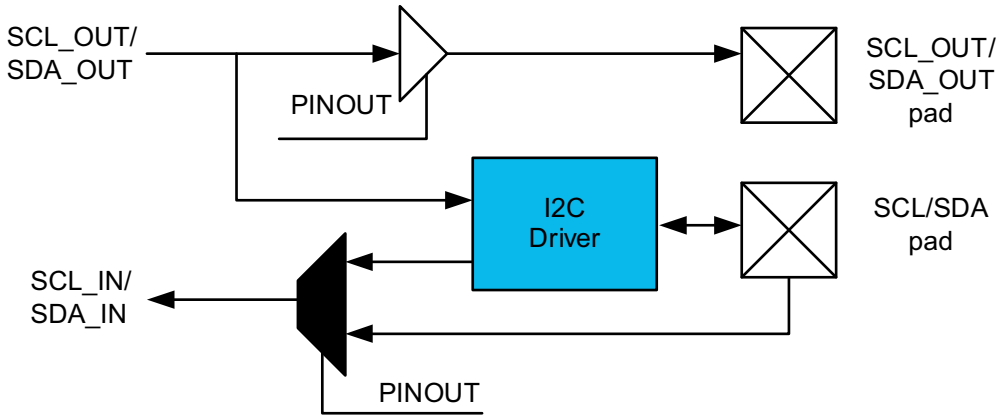
26.6.3.2 Smart Mode

The I²C interface incorporates a special smart mode that simplifies application code and minimizes the user interaction needed to keep hold of the I²C protocol. The smart mode accomplishes this by letting the reading of DATA.DATA automatically issue an ACK or NACK based on the state of CTRLB.ACKACT.

26.6.3.3 4-Wire Mode

Setting the Pin Usage bit in the Control A register (CTRLA.PINOUT) for master or slave to 4-wire mode enables operation as shown in Figure 26-8. In this mode, the internal I²C tri-state drivers are bypassed, and an external, I²C-compliant tri-state driver is needed when connecting to an I²C bus.

Figure 26-8. I2C Pad Interface



26.6.3.4 Quick Command

Setting the Quick Command Enable bit in the Control B register (CTRLB.QCEN) enables quick command. When quick command is enabled, the corresponding interrupt flag is set immediately after the slave acknowledges the address. At this point, the software can either issue a stop command or a repeated start by writing CTRLB.CMD or ADDR.ADDR.

26.6.4 Interrupts

The I²C slave has the following interrupt sources:

- Data Ready
- Address Match
- Stop Received

The I²C master has the following interrupt sources:

- Slave on Bus
- Master on Bus

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the I²C is reset. See INTFLAG for details on how to clear interrupt flags.

The I²C has one common interrupt request line for all the interrupt sources. The user must read INTFLAG to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

26.6.5 Sleep Mode Operation

During I²C master operation, the generic clock (GCLK_SERCOMx_CORE) will continue to run in idle sleep mode. If the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY) is one, the GLK_SERCOMx_CORE will also run in standby sleep mode. Any interrupt can wake up the device.

If CTRLA.RUNSTDBY is zero during I²C master operation, the GLK_SERCOMx_CORE will be disabled when an ongoing transaction is finished. Any interrupt can wake up the device.

During I²C slave operation, writing a one to CTRLA.RUNSTDBY will allow the Address Match interrupt to wake up the device.

In I²C slave operation, all receptions will be dropped when CTRLA.RUNSTDBY is zero.

26.6.6 Synchronization

Due to the asynchronicity between CLK_SERCOMx_APB and GCLK_SERCOMx_CORE, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following register needs synchronization when written:

- Data (DATA) when in smart mode

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)
- Write to Bus State bits in the Status register (STATUS.BUSSTATE)
- Address bits in the Address register (ADDR.ADDR) when in master operation

Write-synchronization is denoted by the Write-Synchronized property in the register description.

The following register needs synchronization when read:

- Data (DATA) when in smart mode

Read-synchronization is denoted by the Read-Synchronized property in the register description.

26.7 Register Summary

Table 26-1. Register Summary – Slave Mode

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]=100			ENABLE	SWRST
0x01		15:8								
0x02		23:16			SDAHOLD[1:0]					PINOUT
0x03		31:24		LOWTOUT						
0x04	CTRLB	7:0								
0x05		15:8	AMODE[1:0]							SMEN
0x06		23:16						ACKACT	CMD[1:0]	
0x07		31:24								
0x08	Reserved									
...	Reserved									
0x0B	Reserved									
0x0C	INTENCLR	7:0						DRDY	AMATCH	PREC
0x0D	INTENSET	7:0						DRDY	AMATCH	PREC
0x0E	INTFLAG	7:0						DRDY	AMATCH	PREC
0x0F	Reserved									

Table 26-1. Register Summary – Slave Mode (Continued)

Offset	Name	Bit Pos.								
0x10	STATUS	7:0	CLKHOLD	LOWTOUT		SR	DIR	RXNACK	COLL	BUSERR
0x11		15:8	SYNCBUSY							
0x12	Reserved									
0x13	Reserved									
0x14	ADDR	7:0	ADDR[6:0]							GENCEN
0x15		15:8								
0x16		23:16	ADDRMASK[6:0]							
0x17		31:24								
0x18	DATA	7:0	DATA[7:0]							
0x19		15:8								

Table 26-2. Register Summary – Master Mode

Offset	Name	Bit Pos								
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]=101			ENABLE	SWRST
0x01		15:8								
0x02		23:16			SDAHOLD[1:0]					PINOUT
0x03		31:24		LOWTOUT	INACTOUT[1:0]					
0x04	CTRLB	7:0								
0x05		15:8							QCEN	SMEN
0x06		23:16						ACKACT	CMD[1:0]	
0x07		31:24								
0x08	DBGCTRL	7:0								DBGSTOP
0x09	Reserved									
0x0A	BAUD	7:0	BAUD[7:0]							
0x0B		15:8	BAUDLOW[7:0]							
0x0C	INTENCLR	7:0							SB	MB
0x0D	INTENSET	7:0							SB	MB
0x0E	INTFLAG	7:0							SB	MB
0x0F	Reserved									

Table 26-2. Register Summary – Master Mode (Continued)

Offset	Name	Bit Pos								
0x10	STATUS	7:0	CLKHOLD	LOWTOUT	BUSSTATE[1:0]			RXNACK	ARBLOST	BUSERR
0x11		15:8	SYNCBUSY							
0x12	Reserved									
0x13	Reserved									
0x14	ADDR	7:0	ADDR[7:0]							
0x15		15:8								
0x16	Reserved									
0x17	Reserved									
0x18	DATA	7:0	DATA[7:0]							
0x19		15:8								

26.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to“Register Access Protection” on page 384 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 395 for details.

Some registers are enable-protected, meaning they can only be written when the I^2C is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

26.8.1 I²C Slave Register Description

26.8.1.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x00000000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		LOWTOUT						
Access	R	R/W	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			SDAHOLD[1:0]					PINOUT
Access	R	R	R/W	R/W	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]=100			ENABLE	SWRST
Access	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 31 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 30 – LOWTOUT: SCL Low Time-Out**
  This bit enables the SCL low time-out. If SCL is held low for 25ms-35ms, the slave will release its clock hold, if enabled, and reset the internal state machine. Any interrupts set at the time of time-out will remain set.
  0: Time-out disabled.
  1: Time-out enabled.
  This bit is not synchronized.

- **Bits 29:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 21:20 – SDAHOLD[1:0]: SDA Hold Time**
  These bits define the SDA hold time with respect to the negative edge of SCL.

Table 26-3. SDA Hold Time

Value	Name	Description
0x0	DIS	Disabled
0x1	75	50-100ns hold time
0x2	450	300-600ns hold time
0x3	600	400-800ns hold time

These bits are not synchronized.

- **Bits 19:17 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 16 – PINOUT: Pin Usage**
  This bit sets the pin usage to either two- or four-wire operation:
  0: 4-wire operation disabled
  1: 4-wire operation enabled
  This bit is not synchronized.
- **Bits 15:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 7 – RUNSTDBY: Run in Standby**
  This bit defines the functionality in standby sleep mode.
  0: Disabled – All reception is dropped.
  1: Wake on address match, if enabled.
  This bit is not synchronized.
- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 4:2 – MODE[2:0]: Operating Mode**
  These bits must be written to 0x04 to select the I^2C slave serial communication interface of the SERCOM.
  These bits are not synchronized.
- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled.
  1: The peripheral is enabled.
  Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.
  This bit is not enable-protected.
- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.

Writing a one to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing a one to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

This bit is not enable-protected.

26.8.1.2 Control B

Name: CTRLB

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
						ACKACT	CMD[1:0]	
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	AMODE[1:0]							SMEN
Access	R/W	R/W	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:19 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 18 – ACKACT: Acknowledge Action**
  0: Send ACK
  1: Send NACK
  The Acknowledge Action (ACKACT) bit defines the slave's acknowledge behavior after an address or data byte is received from the master. The acknowledge action is executed when a command is written to the CMD bits. If smart mode is enabled (CTRLB.SMEN is one), the acknowledge action is performed when the DATA register is read.
  This bit is not enable-protected.
  This bit is not write-synchronized.

- **Bits 17:16 – CMD[1:0]: Command**
  Writing the Command bits (CMD) triggers the slave operation as defined in Table 26-4. The CMD bits are strobe bits, and always read as zero. The operation is dependent on the slave interrupt flags, INTFLAG.DRDY and INTFLAG.AMATCH, in addition to STATUS.DIR (See Table 26-4).

All interrupt flags (INTFLAG.DRDY, INTFLAG.AMATCH and INTFLAG.PREC) are automatically cleared when a command is given.

This bit is not enable-protected.

Table 26-4. Command Description

CMD[1:0]	DIR	Action
0x0	X	(No action)
0x1	X	(Reserved)
0x2	Used to complete a transaction in response to a data interrupt (DRDY)	
	0 (Master write)	Execute acknowledge action succeeded by waiting for any start (S/Sr) condition
	1 (Master read)	Wait for any start (S/Sr) condition
0x3	Used in response to an address interrupt (AMATCH)	
	0 (Master write)	Execute acknowledge action succeeded by reception of next byte
	1 (Master read)	Execute acknowledge action succeeded by slave data interrupt
	Used in response to a data interrupt (DRDY)	
	0 (Master write)	Execute acknowledge action succeeded by reception of next byte
	1 (Master read)	Execute a byte read operation followed by ACK/NACK reception

- **Bits 15:14 – AMODE[1:0]: Address Mode**
  These bits set the addressing mode according to Table 26-5.

Table 26-5. Address Mode Description

Value	Name	Description
0x0	MASK	The slave responds to the address written in ADDR.ADDR masked by the value in ADDR.ADDRMASK(1).
0x1	2_ADDRS	The slave responds to the two unique addresses in ADDR.ADDR and ADDR.ADDRMASK.
0x2	RANGE	The slave responds to the range of addresses between and including ADDR.ADDR and ADDR.ADDRMASK. ADDR.ADDR is the upper limit.
0x3	-	Reserved.

Note: 1. See "SERCOM – Serial Communication Interface" on page 324 for additional information.

These bits are not write-synchronized.

- **Bits 13:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 8 – SMEN: Smart Mode Enable**
  This bit enables smart mode. When smart mode is enabled, acknowledge action is sent when DATA.DATA is read.

  0: Smart mode is disabled.

  1: Smart mode is enabled.

  This bit is not write-synchronized.

- **Bits 7:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

26.8.1.3 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						DRDY	AMATCH	PREC
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – DRDY: Data Ready Interrupt Enable**
  0: The Data Ready interrupt is disabled.
  1: The Data Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Data Ready bit, which disables the Data Ready interrupt.

- **Bit 1 – AMATCH: Address Match Interrupt Enable**
  0: The Address Match interrupt is disabled.
  1: The Address Match interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Address Match Interrupt Enable bit, which disables the Address Match interrupt.

- **Bit 0 – PREC: Stop Received Interrupt Enable**
  0: The Stop Received interrupt is disabled.
  1: The Stop Received interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Stop Received bit, which disables the Stop Received interrupt.

26.8.1.4 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x0D

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						DRDY	AMATCH	PREC
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 2 – DRDY: Data Ready Interrupt Enable**
  0: The Data Ready interrupt is disabled.
  1: The Data Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Data Ready bit, which enables the Data Ready interrupt.
- **Bit 1 – AMATCH: Address Match Interrupt Enable**
  0: The Address Match interrupt is disabled.
  1: The Address Match interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Address Match Interrupt Enable bit, which enables the Address Match interrupt.
- **Bit 0 – PREC: Stop Received Interrupt Enable**
  0: The Stop Received interrupt is disabled.
  1: The Stop Received interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Stop Received bit, which enables the Stop Received interrupt.

26.8.1.5 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x0E

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
						DRDY	AMATCH	PREC
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – DRDY: Data Ready**
  This flag is set when a I^2C slave byte transmission is successfully completed.
  The flag is cleared by hardware when either:
  - Writing to the DATA register.
  - Reading the DATA register with smart mode enabled.
  - Writing a valid command to the CMD register.

  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Data Ready interrupt flag. Optionally, the flag can be cleared manually by writing a one to INTFLAG.DRDY.

- **Bit 1 – AMATCH: Address Match**
  This flag is set when the I^2C slave address match logic detects that a valid address has been received.
  The flag is cleared by hardware when CTRL.CMD is written.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Address Match interrupt flag. Optionally the flag can be cleared manually by writing a one to INTFLAG.AMATCH. When cleared, an ACK/NACK will be sent according to CTRLB.ACKACT.

- **Bit 0 – PREC: Stop Received**
  This flag is set when a stop condition is detected for a transaction being processed. A stop condition detected between a bus master and another slave will not set this flag.
  This flag is cleared by hardware after a command is issued on the next address match.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Stop Received interrupt flag. Optionally, the flag can be cleared manually by writing a one to INTFLAG.PREC.

26.8.1.6 Status

Name: STATUS

Offset: 0x10

Reset: 0x0000

Property: -

Bit	15	14	13	12	11	10	9	8
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CLKHOLD	LOWTOUT		SR	DIR	RXNACK	COLL	BUSERR
Access	R	R/W	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – SYNCBUSY: Synchronization Busy**
  This bit is set when the synchronization of registers between clock domains is started.
  This bit is cleared when the synchronization of registers between the clock domains is complete.

- **Bits 14:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 7 – CLKHOLD: Clock Hold**
  The slave Clock Hold bit (STATUS.CLKHOLD) is set when the slave is holding the SCL line low, stretching the I^2C clock. Software should consider this bit a read-only status flag that is set when INTFLAG.DRDY or INTFLAG.AMATCH is set.
  This bit is automatically cleared when the corresponding interrupt is also cleared.

- **Bit 6 – LOWTOUT: SCL Low Time-out**
  This bit is set if an SCL low time-out occurs.
  This bit is cleared automatically if responding to a new start condition with ACK or NACK (write 3 to CTRLB.CMD) or when INTFLAG.AMATCH is cleared.
  0: No SCL low time-out has occurred.
  1: SCL low time-out has occurred.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the status.

- **Bit 5 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 4 – SR: Repeated Start**
  When INTFLAG.AMATCH is raised due to an address match, SR indicates a repeated start or start condition.
  0: Start condition on last address match
  1: Repeated start condition on last address match
  This flag is only valid while the INTFLAG.AMATCH flag is one.

- **Bit 3 – DIR: Read / Write Direction**
  The Read/Write Direction (STATUS.DIR) bit stores the direction of the last address packet received from a master.
  0: Master write operation is in progress.
  1: Master read operation is in progress.

- **Bit 2 – RXNACK: Received Not Acknowledge**
  This bit indicates whether the last data packet sent was acknowledged or not.
  0: Master responded with ACK.
  1: Master responded with NACK.

- **Bit 1 – COLL: Transmit Collision**
  If set, the I^2C slave was not able to transmit a high data or NACK bit, the I^2C slave will immediately release the SDA and SCL lines and wait for the next packet addressed to it.
  This flag is intended for the SMBus address resolution protocol (ARP). A detected collision in non-ARP situations indicates that there has been a protocol violation, and should be treated as a bus error.
  Note that this status will not trigger any interrupt, and should be checked by software to verify that the data were sent correctly. This bit is cleared automatically if responding to an address match with an ACK or a NACK (writing 0x3 to CTRLB.CMD), or INTFLAG.AMATCH is cleared.
  0: No collision detected on last data byte sent.
  1: Collision detected on last data byte sent.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the status.

- **Bit 0 – BUSERR: Bus Error**
  The Bus Error bit (STATUS.BUSERR) indicates that an illegal bus condition has occurred on the bus, regardless of bus ownership. An illegal bus condition is detected if a protocol violating start, repeated start or stop is detected on the I^2C bus lines. A start condition directly followed by a stop condition is one example of a protocol violation. If a time-out occurs during a frame, this is also considered a protocol violation, and will set STATUS.BUSERR.
  This bit is cleared automatically if responding to an address match with an ACK or a NACK (writing 0x3 to CTRLB.CMD) or INTFLAG.AMATCH is cleared.
  0: No bus error detected.
  1: Bus error detected.
  Writing a one to this bit will clear the status.
  Writing a zero to this bit has no effect.

26.8.1.7 Address

Name: ADDR

Offset: 0x14

Reset: 0x00000000

Property: Write-Protected, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRMASK[6:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[6:0]							GENCEN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:24 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 23:17 – ADDRMASK[6:0]: Address Mask**
  The ADDRMASK bits acts as a second address match register, an address mask register or the lower limit of an address range, depending on the CTRLB.AMODE setting.
- **Bits 16:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 7:1 – ADDR[6:0]: Address**
  The slave address (ADDR) bits contain the I^2C slave address used by the slave address match logic to determine if a master has addressed the slave. When using 7-bit addressing, the address register (ADDR.ADDR) represents the slave address.

  If using 10-bit addressing, the address match logic only supports hardware address recognition of the first 2 bits of a 10-bit address. If writing ADDR.ADDR = "0b1111 0xx," 'xx' represents bits 9 and 8 or the slave address. The next byte received is bits 7 to 0 in the 10-bit address, and this must be handled by software.

When the address match logic detects a match, INTFLAG.AMATCH is set and STATUS.DIR is updated to indicate whether it is a read or a write transaction.

- **Bit 0 – GENCEN: General Call Address Enable**
  Writing a one to GENCEN enables general call address recognition. A general call address is an address of all zeroes with the direction bit written to zero (master write).

  0: General call address recognition disabled.

  1: General call address recognition enabled.

26.8.1.8 Data

Name: DATA

Offset: 0x18

Reset: 0x0000

Property: Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 7:0 – DATA[7:0]: Data**
  The slave data register I/O location (DATA.DATA) provides access to the master transmit and receive data buffers. Reading valid data or writing data to be transmitted can be successfully done only when SCL is held low by the slave (STATUS.CLKHOLD is set). An exception occurs when reading the last data byte after the stop condition has been received.
  Accessing DATA.DATA auto-triggers I^2C bus operations. The operation performed depends on the state of CTRLB.ACKACT, CTRLB.SMEN and the type of access (read/write).
  Writing or reading DATA.DATA when not in smart mode does not require synchronization.

26.8.2 I²C Master Register Description

26.8.2.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x00000000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		LOWTOUT	INACTOUT[1:0]					
Access	R	R/W	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			SDAHOLD[1:0]					PINOUT
Access	R	R	R/W	R/W	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]=101			ENABLE	SWRST
Access	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 31 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 30 – LOWTOUT: SCL Low Time-Out**
  This bit enables the SCL low time-out. If SCL is held low for 25ms-35ms, the master will release its clock hold, if enabled, and complete the current transaction. A stop condition will automatically be transmitted.
  INTFLAG.SB or INTFLAG.MB will be set as normal, but the clock hold will be released. The STATUS.LOWTOUT and STATUS.BUSERR status bits will be set.
  0: Time-out disabled.
  1: Time-out enabled.
  This bit is not synchronized.

- **Bits 29:28 – INACTOUT[1:0]: Inactive Time-Out**
  If the inactive bus time-out is enabled and the bus is inactive for longer than the time-out setting, the bus state logic will be set to idle. An inactive bus arise when either an I²C master or slave is holding the SCL low. The available time-outs are given in Table 26-6.

Enabling this option is necessary for SMBus compatibility, but can also be used in a non-SMBus set-up.

Table 26-6. Inactive Timout

Value	Name	Description
0x0	DIS	Disabled
0x1	55US	5-6 SCL cycle time-out (50-60µs)
0x2	105US	10-11 SCL cycle time-out (100-110µs)
0x3	205US	20-21 SCL cycle time-out (200-210µs)

Calculated time-out periods are based on a 100kHz baud rate.

These bits are not synchronized.

- **Bits 27:22 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 21:20 – SDAHOLD[1:0]: SDA Hold Time**
  These bits define the SDA hold time with respect to the negative edge of SCL.

Table 26-7. SDA Hold Time

Value	Name	Description
0x0	DIS	Disabled
0x1	75NS	50-100ns hold time
0x2	450NS	300-600ns hold time
0x3	600NS	400-800ns hold time

These bits are not synchronized.

- **Bits 19:17 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 16 – PINOUT: Pin Usage**
  This bit set the pin usage to either two- or four-wire operation:
  0: 4-wire operation disabled.
  1: 4-wire operation enabled.
  This bit is not synchronized.
- **Bits 15:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 7 – RUNSTDBY: Run in Standby**
  This bit defines the functionality in standby sleep mode.
  0: GCLK_SERCOMx_CORE is disabled and the I^2C master will not operate in standby sleep mode.
  1: GCLK_SERCOMx_CORE is enabled in all sleep modes allowing the master to operate in standby sleep mode.
  This bit is not synchronized.

- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 4:2 – MODE[2:0]: Operating Mode**
  These bits must be written to 0x5 to select the I^2C master serial communication interface of the SERCOM.
  These bits are not synchronized.

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled.
  1: The peripheral is enabled.
  Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.
  This bit is not enable-protected.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.
  Writing a one to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.
  Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.
  This bit is not enable-protected.

26.8.2.2 Control B

Name: CTRLB

Offset: 0x04

Reset: 0x00000000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
						ACKACT	CMD[1:0]	
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
							QCEN	SMEN
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 31:19 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 18 – ACKACT: Acknowledge Action**
  The Acknowledge Action (ACKACT) bit defines the I^2C master's acknowledge behavior after a data byte is received from the I^2C slave. The acknowledge action is executed when a command is written to CTRLB.CMD, or if smart mode is enabled (CTRLB.SMEN is written to one), when DATA.DATA is read.

  0: Send ACK.

  1: Send NACK.

  This bit is not enable-protected.

  This bit is not write-synchronized.

- **Bits 17:16 – CMD[1:0]: Command**
  Writing the Command bits (CMD) triggers the master operation as defined in Table 26-8. The CMD bits are strobe bits, and always read as zero. The acknowledge action is only valid in master read mode. In master write mode, a command will only result in a repeated start or stop condition. The CTRLB.ACKACT bit and the CMD bits can be written at the same time, and then the acknowledge action will be updated before the command is triggered.

Commands can only be issued when the Slave on Bus interrupt flag (INTFLAG.SB) or Master on Bus interrupt flag (INTFLAG.MB) is one.

If CMD 0x1 is issued, a repeated start will be issued followed by the transmission of the current address in ADDR.ADDR. If another address is desired, ADDR.ADDR must be written instead of the CMD bits. This will trigger a repeated start followed by transmission of the new address.

Issuing a command will set STATUS.SYNCBUSY.

Table 26-8. Command Description

CMD[1:0]	DIR	Action
0x0	X	(No action)
0x1	X	Execute acknowledge action succeeded by repeated Start
0x2	0 (Write)	No operation
	1 (Read)	Execute acknowledge action succeeded by a byte read operation
0x3	X	Execute acknowledge action succeeded by issuing a stop condition

These bits are not enable-protected.

- **Bits 15:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 9 – QCEN: Quick Command Enable**
  Setting the Quick Command Enable bit (QCEN) enables quick command.
  0: Quick Command is disabled.
  1: Quick Command is enabled.
  This bit is not write-synchronized.
- **Bit 8 – SMEN: Smart Mode Enable**
  This bit enables smart mode. When smart mode is enabled, acknowledge action is sent when DATA.DATA is read.
  0: Smart mode is disabled.
  1: Smart mode is enabled.
  This bit is not write-synchronized.
- **Bits 7:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

26.8.2.3 Debug Control

Name: DBGCTRL

Offset: 0x08

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								DBGSTOP
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – DBGSTOP: Debug Stop Mode**
  This bit controls functionality when the CPU is halted by an external debugger.
  0: The baud-rate generator continues normal operation when the CPU is halted by an external debugger.
  1: The baud-rate generator is halted when the CPU is halted by an external debugger.

26.8.2.4 Baud Rate

Name: BAUD

Offset: 0x0A

Reset: 0x0000

Property: Write-Protected, Enable-Protected

Bit	15	14	13	12	11	10	9	8
	BAUDLOW[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BAUD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:8 – BAUDLOW[7:0]: Master Baud Rate Low**
  If the Master Baud Rate Low bit group (BAUDLOW) has a non-zero value, the SCL low time will be described by the value written.
  For more information on how to calculate the frequency, see “SERCOM I2C – SERCOM Inter-Integrated Circuit” on page 382.

- **Bits 7:0 – BAUD[7:0]: Master Baud Rate**
  The Master Baud Rate bit group (BAUD) is used to derive the SCL high time if BAUD.BAUDLOW is non-zero. If BAUD.BAUDLOW is zero, BAUD will be used to generate both high and low periods of the SCL.
  For more information on how to calculate the frequency, see “SERCOM I2C – SERCOM Inter-Integrated Circuit” on page 382.

26.8.2.5 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							SB	**MB**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – SB: Slave on Bus Interrupt Enable**
  0: The Slave on Bus interrupt is disabled.
  1: The Slave on Bus interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Slave on Bus Interrupt Enable bit, which disables the Slave on Bus interrupt.

- **Bit 0 – MB: Master on Bus Interrupt Enable**
  0: The Master on Bus interrupt is disabled.
  1: The Master on Bus interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Master on Bus Interrupt Enable bit, which disables the Master on Bus interrupt.

26.8.2.6 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x0D

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							SB	**MB**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – SB: Slave on Bus Interrupt Enable**
  0: The Slave on Bus interrupt is disabled.
  1: The Slave on Bus interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Slave on Bus Interrupt Enable bit, which enables the Slave on Bus interrupt.

- **Bit 0 – MB: Master on Bus Interrupt Enable**
  0: The Master on Bus interrupt is disabled.
  1: The Master on Bus interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Master on Bus Interrupt Enable bit, which enables the Master on Bus interrupt.

26.8.2.7 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x0E

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
							SB	**MB**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – SB: Slave on Bus**
  The Slave on Bus flag (SB) is set when a byte is successfully received in master read mode, i.e., no arbitration lost or bus error occurred during the operation. When this flag is set, the master forces the SCL line low, stretching the I^2C clock period. The SCL line will be released and SB will be cleared on one of the following actions:
  - Writing to ADDR.ADDR
  - Writing to DATA.DATA
  - Reading DATA.DATA when smart mode is enabled (CTRLB.SMEN)
  - Writing a valid command to CTRLB.CMD

  Writing a one to this bit location will clear the SB flag. The transaction will not continue or be terminated until one of the above actions is performed.

  Writing a zero to this bit has no effect.

- **Bit 0 – MB: Master on Bus**
  The Master on Bus flag (MB) is set when a byte is transmitted in master write mode. The flag is set regardless of the occurrence of a bus error or an arbitration lost condition. MB is also set when arbitration is lost during sending of NACK in master read mode, and when issuing a start condition if the bus state is unknown. When this flag is set and arbitration is not lost, the master forces the SCL line low, stretching the I^2C clock period. The SCL line will be released and MB will be cleared on one of the following actions:
  - Writing to ADDR.ADDR
  - Writing to DATA.DATA
  - Reading DATA.DATA when smart mode is enabled (CTRLB.SMEN)
  - Writing a valid command to CTRLB.CMD

  If arbitration is lost, writing a one to this bit location will clear the MB flag.

  If arbitration is not lost, writing a one to this bit location will clear the MB flag. The transaction will not continue or be terminated until one of the above actions is performed.

  Writing a zero to this bit has no effect.

26.8.2.8 Status

Name: STATUS

Offset: 0x10

Reset: 0x0000

Property: Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CLKHOLD	LOWTOUT	BUSSTATE[1:0]			RXNACK	ARBLOST	BUSERR
Access	R	R/W	R	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 14:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 7 – CLKHOLD: Clock Hold**
  The Master Clock Hold flag (STATUS.CLKHOLD) is set when the master is holding the SCL line low, stretching the I^2C clock. Software should consider this bit a read-only status flag that is set when INTFLAG.SB or INTFLAG.MB is set. When the corresponding interrupt flag is cleared and the next operation is given, this bit is automatically cleared.
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.
  This bit is not write-synchronized.

- **Bit 6 – LOWTOUT: SCL Low Time-Out**
  This bit is set if an SCL low time-out occurs.
  Writing a one to this bit location will clear STATUS.LOWTOUT. Normal use of the I^2C interface does not require the LOWTOUT flag to be cleared by this method. This flag is automatically cleared when writing to the ADDR register.
  Writing a zero to this bit has no effect.
  This bit is not write-synchronized.

- **Bits 5:4 – BUSSTATE[1:0]: Bus State**
  These bits indicate the current I^2C bus state as defined in Table 26-9. After enabling the SERCOM as an I^2C master, the bus state will be unknown.

Table 26-9. Bus State

Value	Name	Description
0x0	Unknown	The bus state is unknown to the I^{2}C master and will wait for a stop condition to be detected or wait to be forced into an idle state by software
0x1	Idle	The bus state is waiting for a transaction to be initialized
0x2	Owner	The I^{2}C master is the current owner of the bus
0x3	Busy	Some other I^{2}C master owns the bus

When the master is disabled, the bus-state is unknown. When in the unknown state, writing 0x1 to BUSSTATE forces the bus state into the idle state. The bus state cannot be forced into any other state.

Writing STATUS.BUSSTATE to idle will set STATUS.SYNCBUSY.

- **Bit 3 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 2 – RXNACK: Received Not Acknowledge**
  This bit indicates whether the last address or data packet sent was acknowledged or not.
  0: Slave responded with ACK.
  1: Slave responded with NACK.
  Writing a zero to this bit has no effect.
  Writing a one to this bit has no effect.
  This bit is not write-synchronized.

- **Bit 1 – ARBLOST: Arbitration Lost**
  The Arbitration Lost flag (STATUS.ARBLOST) is set if arbitration is lost while transmitting a high data bit or a NACK bit, or while issuing a start or repeated start condition on the bus. The Master on Bus interrupt flag (INTFLAG.MB) will be set when STATUS.ARBLOST is set.
  Writing the ADDR.ADDR register will automatically clear STATUS.ARBLOST.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear it.
  This bit is not write-synchronized.

- **Bit 0 – BUSERR: Bus Error**
  The Bus Error bit (STATUS.BUSERR) indicates that an illegal bus condition has occurred on the bus, regardless of bus ownership. An illegal bus condition is detected if a protocol violating start, repeated start or stop is detected on the I^{2}C bus lines. A start condition directly followed by a stop condition is one example of a protocol violation. If a time-out occurs during a frame, this is also considered a protocol violation, and will set BUSERR.
  If the I^{2}C master is the bus owner at the time a bus error occurs, STATUS.ARBLOST and INTFLAG.MB will be set in addition to BUSERR.
  Writing the ADDR.ADDR register will automatically clear the BUSERR flag.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear it.
  This bit is not write-synchronized.

26.8.2.9 Address

Name: ADDR

Offset: 0x14

Reset: 0x0000

Property: Write-Synchronized

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 7:0 – ADDR[7:0]: Address**
  When ADDR is written, the consecutive operation will depend on the bus state:
  Unknown: INTFLAG.MB and STATUS.BUSERR are set, and the operation is terminated.
  Busy: The I^2C master will await further operation until the bus becomes idle.
  Idle: The I^2C master will issue a start condition followed by the address written in ADDR. If the address is acknowledged, SCL is forced and held low, and STATUS.CLKHOLD and INTFLAG.MB are set.
  Owner: A repeated start sequence will be performed. If the previous transaction was a read, the acknowledge action is sent before the repeated start bus condition is issued on the bus. Writing ADDR to issue a repeated start is performed while INTFLAG.MB or INTFLAG.SB is set.
  Regardless of winning or loosing arbitration, the entire address will be sent. If arbitration is lost, only ones are transmitted from the point of loosing arbitration and the rest of the address length.
  STATUS.BUSERR, STATUS.ARBLOST, INTFLAG.MB and INTFLAG.SB will be cleared when ADDR is written.
  The ADDR register can be read at any time without interfering with ongoing bus activity, as a read access does not trigger the master logic to perform any bus protocol related operations.
  The I^2C master control logic uses bit 0 of ADDR as the bus protocol's read/write flag (R/W); 0 for write and 1 for read.

26.8.2.10 Data

Name: DATA

Offset: 0x18

Reset: 0x0000

Property: Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:8 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 7:0 – DATA[7:0]: Data**
  The master data register I/O location (DATA) provides access to the master transmit and receive data buffers. Reading valid data or writing data to be transmitted can be successfully done only when SCL is held low by the master (STATUS.CLKHOLD is set). An exception occurs when reading the last data byte after the stop condition has been sent.
  Accessing DATA.DATA auto-triggers I^2C bus operations. The operation performed depends on the state of CTRLB.ACKACT, CTRLB.SMEN and the type of access (read/write).
  Writing or reading DATA.DATA when not in smart mode does not require synchronization.

27. TC – Timer/Counter

27.1 Overview

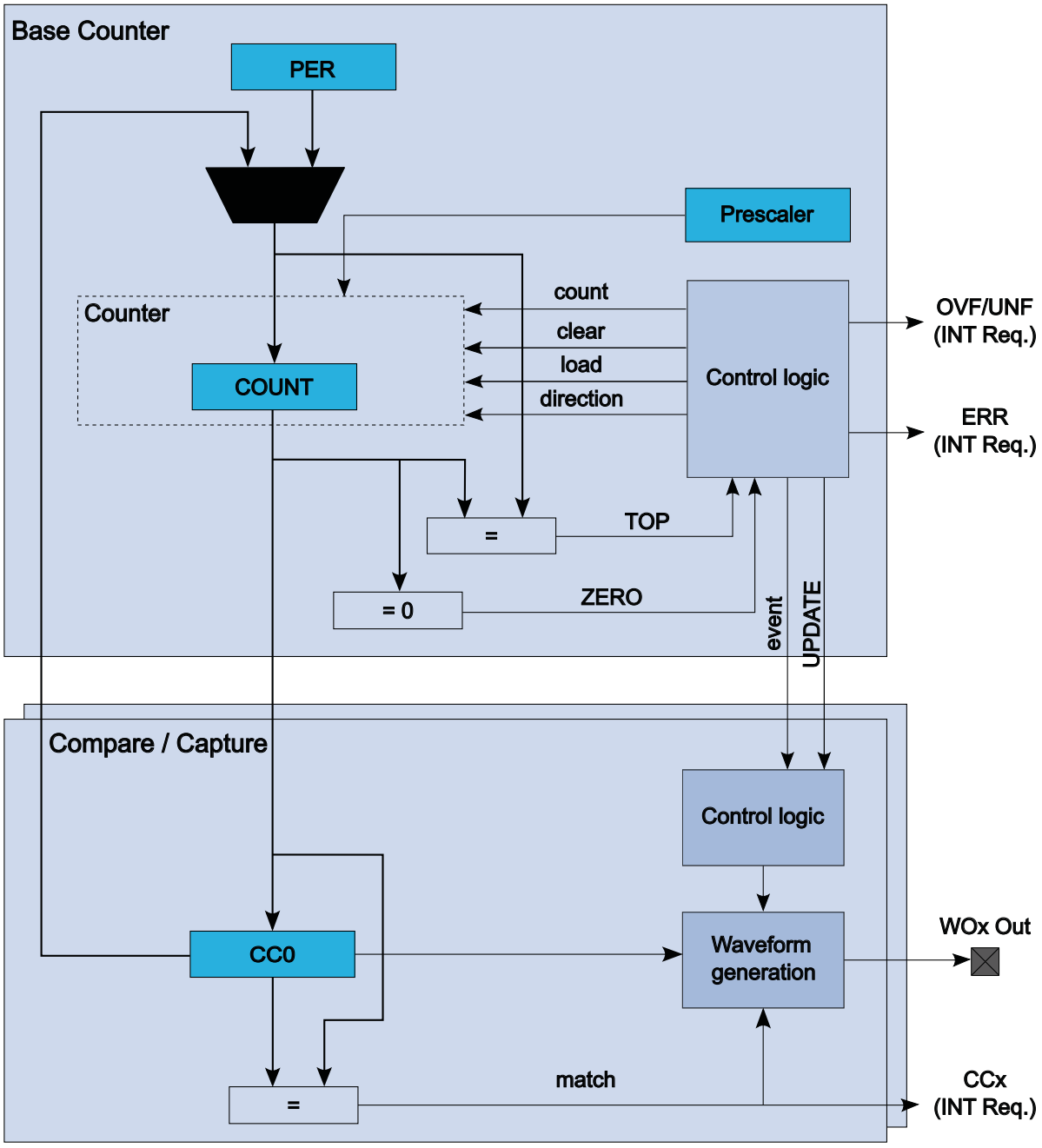
The TC consists of a counter, a prescaler, compare/capture channels and control logic. The counter can be set to count events, or it can be configured to count clock pulses. The counter, together with the compare/capture channels, can be configured to timestamp input events, allowing capture of frequency and pulse width. It can also perform waveform generation, such as frequency generation and pulse-width modulation (PWM).

27.2 Features

- Selectable configuration
  - 8-, 16- or 32-bit TC, with compare/capture channels
- Waveform generation
  - Frequency generation
  - Single-slope pulse-width modulation
- Input capture
  - Event capture
  - Frequency capture
  - Pulse-width capture
- One input event
- Interrupts/output events on:
  - Counter overflow/underflow
  - Compare match or capture
- Internal prescaler

27.3 Block Diagram

Figure 27-1. Timer/Counter Block Diagram



27.4 Signal Description

Signal Name	Type	Description
WO[1:0]	Digital output	Waveform output

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

27.5 Product Dependencies

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

27.5.1 I/O Lines

Using the TC's I/O lines requires the I/O pins to be configured. Refer to "PORT" on page 276 for details.

27.5.2 Power Management

The TC can continue to operate in any sleep mode where the selected source clock is running. The TC interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

27.5.3 Clocks

The TC bus clock (CLK_TCx_APB, where x represents the specific TC instance number) can be enabled and disabled in the Power Manager, and the default state of CLK_TCx_APB can be found in the Peripheral Clock Masking section in "PM – Power Manager" on page 100.

The different TC instances are paired, even and odd, starting from TC0, and use the same generic clock, GCLK_TCx. This means that the TC instances in a TC pair cannot be set up to use different GCLK_TCx clocks.

This generic clock is asynchronous to the user interface clock (CLK_TCx_APB). Due to this asynchronicity, accessing certain registers will require synchronization between the clock domains. Refer to "Synchronization" on page 440 for further details.

27.5.4 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the TC interrupts requires the interrupt controller to be configured first. Refer to "Nested Vector Interrupt Controller" on page 24 for details.

27.5.5 Events

To use the TC event functionality, the corresponding events need to be configured in the event system. Refer to "EVSYS – Event System" on page 301 for details.

27.5.6 Debug Operation

When the CPU is halted in debug mode the TC will halt normal operation. The TC can be forced to continue operation during debugging. Refer to the DBGCTRL register for details.

27.5.7 Register Access Protection

All registers with write-access are optionally write-protected by the peripheral access controller (PAC), except the following registers:

- Interrupt Flag register (INTFLAG)
- Status register (STATUS)

- Read Request register (READREQ)
- Count register (COUNT)
- Period register (PER)
- Compare/Capture Value registers (CCx)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

27.5.8 Analog Connections

Not applicable.

27.6 Functional Description

27.6.1 Principle of Operation

The counter in the TC can be set to count on events from the Event System, or on the GCLK_TCx frequency. The pulses from GCLK_TCx will go through the prescaler, where it is possible to divide the frequency down.

The value in the counter is passed to the compare/capture channels, where it can either be compared with user defined values or captured on a predefined event.

The TC can be configured as an 8-, 16- or 32-bit counter. Which mode is chosen will determine the maximum range of the counter. The counter range combined with the operating frequency will determine the maximum time resolution achievable with the TC peripheral.

The TC can be set to count up or down. By default, the counter will operate in a continuous mode, where the counter will wrap to the zero respectively top value it counts from when reaching the top respectively zero.

When one of the compare/capture channels is used in compare mode, the TC can be used for waveform generation. Upon a match between the counter and the value in one or more of the Compare/Capture Value registers (CCx), one or more output pins on the device can be set to toggle. The CCx registers and the counter can thereby be used in frequency generation and PWM generation.

Capture mode can be used to automatically capture the period and pulse width of signals.

27.6.2 Basic Operation

27.6.2.1 Initialization

The following register is enable-protected, meaning that it can only be written when the TC is disabled (CTRLA.ENABLE is zero):

- Control A register (CTRLA), except the Run Standby (RUNSTDBY), Enable (ENABLE) and Software Reset (SWRST) bits

The following bits are enable-protected:

- Event Action bits in the Event Control register (EVCTRL.EVACT)

Enable-protected bits in the CTRLA register can be written at the same time as CTRLA.ENABLE is written to one, but not at the same time as CTRLA.ENABLE is written to zero.

Before the TC is enabled, it must be configured, as outlined by the following steps:

- The TC bus clock (CLK_TCx_APB) must be enabled
- The mode (8, 16 or 32 bits) of the TC must be selected in the TC Mode bit group in the Control A register (CTRLA.MODE). The default mode is 16 bits
- One of the wavegen modes must be selected in the Waveform Generation Operation bit group in the Control A register (CTRLA.WAVEGEN)

- If the GCLK_TCx frequency used should be prescaled, this can be selected in the Prescaler bit group in the Control A register (CTRLA.PRESCALER)
- If the prescaler is used, one of the presync modes must be chosen in the Prescaler and Counter Synchronization bit group in the Control A register (CTRLA.PRESYNC)
- One-shot mode can be selected by writing a one to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT)
- If the counter should count down from the top value, write a one to the Counter Direction bit in the Control B Set register (CTRLBSET.DIR)
- If capture operations are to be used, the individual channels must be enabled for capture in the Capture Channel x Enable bit group in the Control C register (CTRLC.CPTEN)
- The waveform output for individual channels can be inverted using the Output Waveform Invert Enable bit group in the Control C register (CTRLC.INVEN)

27.6.2.2 Enabling, Disabling and Resetting

The TC is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The TC is disabled by writing a zero to CTRLA.ENABLE.

The TC is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the TC, except DBGCTRL, will be reset to their initial state, and the TC will be disabled. Refer to the CTRLA register for details.

The TC should be disabled before the TC is reset to avoid undefined behavior.

27.6.2.3 Prescaler Selection

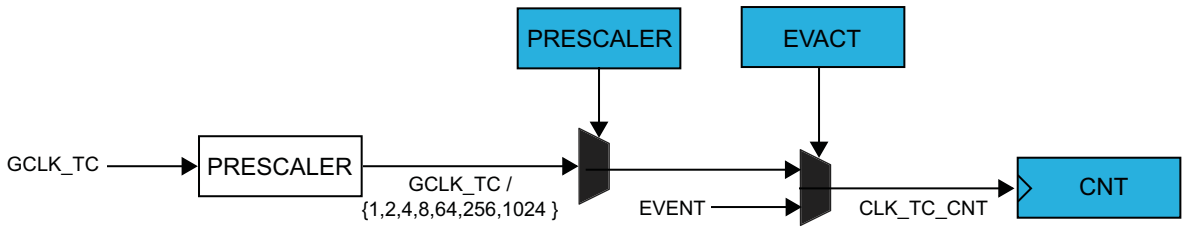
As seen in Figure 27-2, the GCLK_TC clock is fed into the internal prescaler. Prescaler output intervals from 1 to 1/1024 are available. For a complete list of available prescaler outputs, see the register description for the Prescaler bit group in the Control A register (CTRLA.PRESCALER).

The prescaler consists of a counter that counts to the selected prescaler value, whereupon the output of the prescaler toggles.

When the prescaler is set to a value greater than one, it is necessary to choose whether the prescaler should reset its value to zero or continue counting from its current value on the occurrence of an overflow or underflow. It is also necessary to choose whether the TC counter should wrap around on the next GCLK_TC clock pulse or the next prescaled clock pulse (CLK_TC_CNT of Figure 27-2). To do this, use the Prescaler and Counter Synchronization bit group in the Control A register (CTRLA.PRESYNC).

If the counter is set to count events from the event system, these will not pass through the prescaler, as seen in Figure 27-2.

Figure 27-2. Prescaler



27.6.2.4 TC Mode

The counter mode is selected with the TC Mode bit group in the Control A register (CTRLA.MODE). By default, the counter is enabled in the 16-bit counter mode.

Three counter modes are available:

- COUNT8: The 8-bit TC has its own Period register (PER). This register is used to store the period value that can be used as the top value for waveform generation.
- COUNT16: This is the default counter mode. There is no dedicated period register in this mode.
- COUNT32: This mode is achieved by pairing two 16-bit TC peripherals. This pairing is explained in “Clocks” on page 431. The even-numbered TC instance will act as master to the odd-numbered TC peripheral, which will act as a slave. The slave status of the slave is indicated by reading the Slave bit in the Status register (STATUS.SLAVE). The registers of the slave will not reflect the registers of the 32-bit counter. Writing to any of the slave registers will not affect the 32-bit counter. Normal access to the slave COUNT and CCx registers is not allowed.

27.6.2.5 Counter Operations

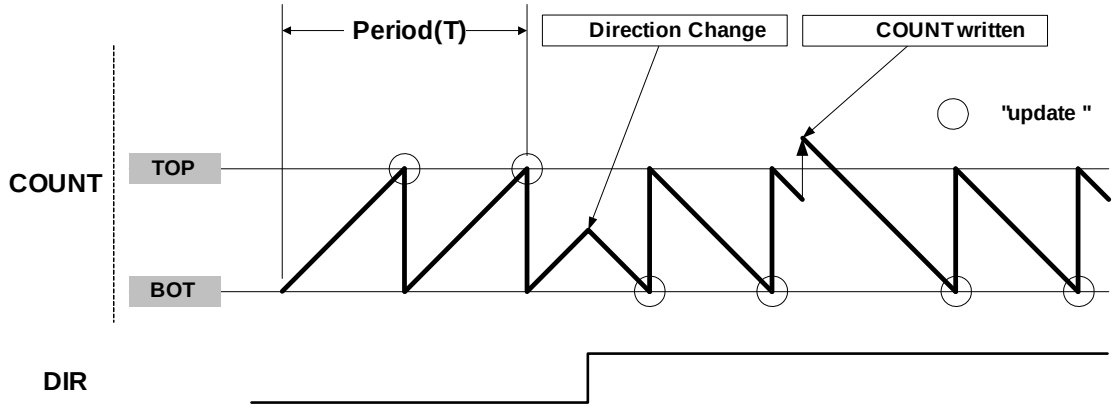
The counter can be set to count up or down. When the counter is counting up and the top value is reached, the counter will wrap around to zero on the next clock cycle. When counting down, the counter will wrap around to the top value when zero is reached. For one-shot mode, the counter will continue to count after a wraparound occurs.

To set the counter to count down, write a one to the Direction bit in the Control B Set register (CTRLBSET.DIR). To count up, write a one to the Direction bit in the Control B Clear register (CTRLBCLR.DIR).

Each time the counter reaches the top value or zero, it will set the Overflow Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF). It is also possible to generate an event on overflow or underflow when the Overflow/Underflow Event Output Enable bit in the Event Control register (EVCTRL.OVFEO) is one.

The counter value can be read from the Counter Value register (COUNT) or a new value can be written to the COUNT register. Figure 27-3 gives an example of writing a new counter value. The COUNT value will always be zero when starting the TC, unless some other value has been written to it or the TC has been stopped at some value other than zero.

Figure 27-3. Counter Operation



Stop Command

On the stop command, which can be evoked in the Command bit group in the Control B Set register (CTRLBSET.CMD), the counter will retain its current value. All waveforms are cleared. The counter stops counting, and the Stop bit in the Status register is set (STATUS.STOP).

Retrigger Command and Event Action

Retriggering can be evoked either as a software command, using the Retrigger command in the Control B Set register (CTRLBSET.CMD), or as a retrigger event action, using the Event Action bit group in the Event Control register (EVCTRL.EVACT).

When a retrigger is evoked while the counter is running, the counter will wrap to the top value or zero, depending on the counter direction..

When a retrigger is evoked with the counter stopped, the counter will continue counting from the value in the COUNT register.

Note: When retrigger event action is configured and enabled as an event action, enabling the counter will not start the counter. The counter will start at the next incoming event and restart on any following event.

Count Event Action

When the count event action is configured, every new incoming event will make the counter increment or decrement, depending on the state of the direction bit (CTRLBSET.DIR).

Start Event Action

When the TC is configured with a start event action in the EVCTRL.EVACT bit group, enabling the TC does not make the counter start; the start is postponed until the next input event or software retrigger action. When the counter is running, an input event has not effect on the counter.

27.6.2.6 Compare Operations

When using the TC with the Compare/Capture Value registers (CCx) configured for compare operation, the counter value is continuously compared to the values in the CCx registers. This can be used for timer or waveform operation.

Waveform Output Operations

The compare channels can be used for waveform generation on the corresponding I/O pins. To make the waveform visible on the connected pin, the following requirements must be fulfilled:

- Choose a waveform generation operation
- Optionally, invert the waveform output by writing the corresponding Output Waveform Invert Enable bit in the Control C register (CTRLC.INVx)
- Enable the corresponding multiplexor in the PORT

The counter value is continuously compared with each CCx available. When a compare match occurs, the Match or Capture Channel x interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.MCx) is set on the next zero-to-one transition of CLK_TC_CNT (see Figure 27-4). An interrupt and/or event can be generated on such a condition when INTENSET.MCx and/or EVCTRL.MCEOx is one.

One of four configurations in the Waveform Generation Operation bit group in the Control A register (CTRLA.WAVEGEN) must be chosen to perform waveform generation. This will influence how the waveform is generated and impose restrictions on the top value. The four configurations are:

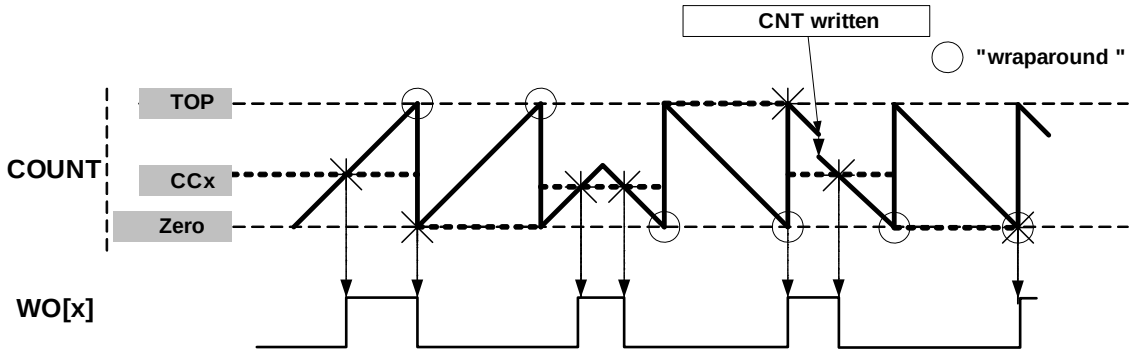
- Normal frequency (NFRQ)
- Match frequency (MFRQ)
- Normal PWM (NPWM)
- Match PWM (MPWM)

When using NPWM or NFRQ, the top value is determined by the counter mode. In 8-bit mode, the Period register (PER) is used as the top value and the top value can be changed by writing to the PER register. In 16- and 32-bit mode, the top value is fixed to the maximum value of the counter.

Frequency Operation

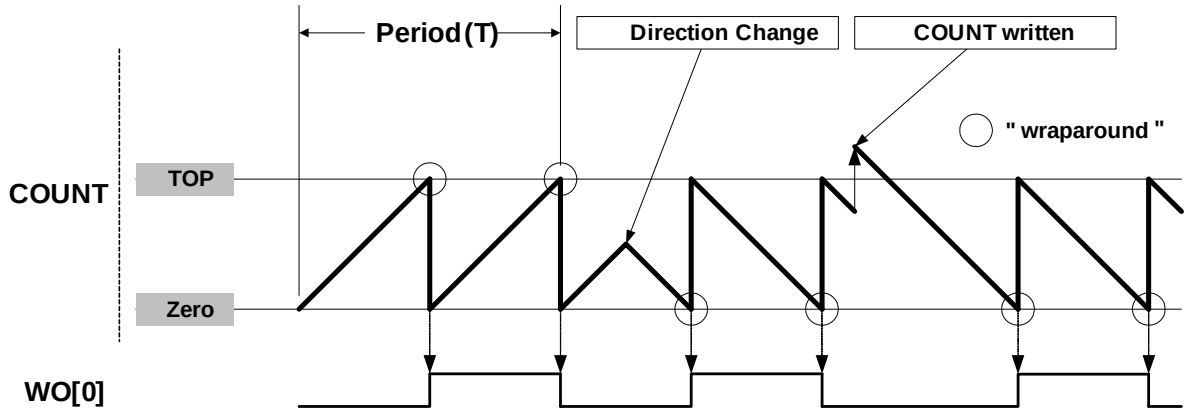
When NFRQ is used, the output waveform (WO[x]) toggles every time CCx and the counter are equal, and the interrupt flag corresponding to that channel will be set.

Figure 27-4. Normal Frequency Operation



When MFRQ is used, the value in CC0 will be used as the top value and WO[0] will toggle on every overflow/underflow.

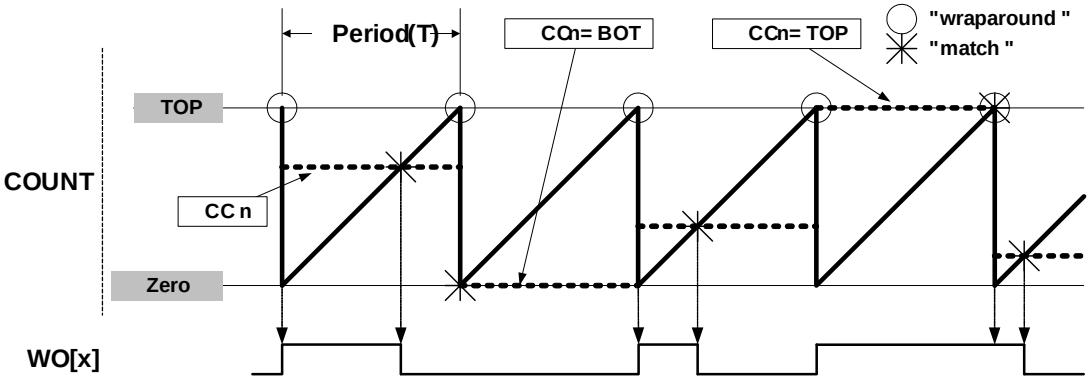
Figure 27-5. Match Frequency Operation



PWM Operation

In PWM operation, the CCx registers control the duty cycle of the waveform generator output. Figure 27-6 shows how the WO[x] output is set at a start or a compare match between the COUNT value and the top value, and cleared on the compare match between the COUNT value and CCx register value.

Figure 27-6. Normal PWM Operation



In match operation, Compare/Capture register CC0 is used as the top value, and in this case WO[0] will toggle on every overflow/underflow.

The following equation is used to calculate the exact period for a single-slope PWM (R_{PWM_SS}) waveform:

$$R_{PWM\_SS} = \frac{\log(TOP+1)}{\log(2)}$$

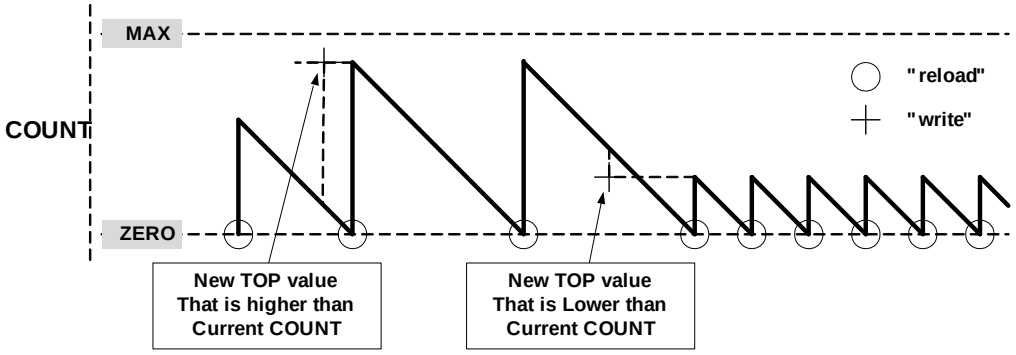
$$f_{PWM\_SS} = \frac{f_{CLK\_TC}}{N(TOP+1)}$$

where N represent the prescaler divider used (1, 2, 4, 8, 16, 64, 256, 1024).

Changing the Top Value

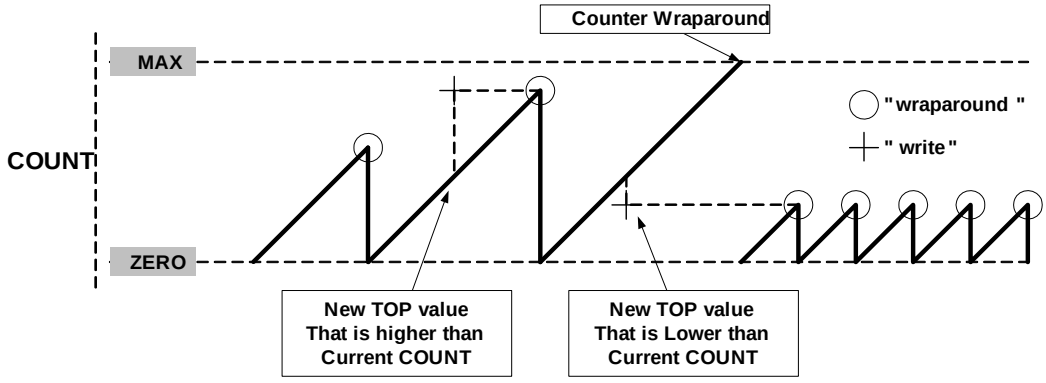
Changing the top value while the counter is running is possible. If a new top value is written when the counter value is close to zero and counting down, the counter can be reloaded with the previous top value, due to synchronization delays. If this happens, the counter will count one extra cycle before the new top value is used.

Figure 27-7. Changing the Top Value when Counting Down



When counting up a change from a top value that is lower relative to the old top value can make the counter miss this change if the counter value is larger than the new top value when the change occurred. This will make the counter count to the max value. An example of this can be seen in Figure 27-8.

Figure 27-8. Changing the Top Value when Counting Up



27.6.2.7 Capture Operations

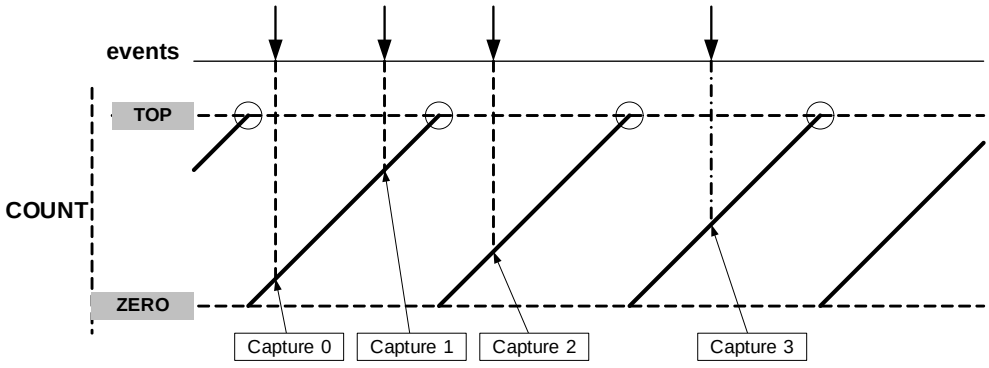
To enable and use capture operations, the event line into the TC must be enabled using the TC Event Input bit in the Event Control register (EVCTRL.TCEI). The capture channels to be used must also be enabled in the Capture Channel x Enable bit group in the Control C register (CTRLC.CPTENx) before capture can be performed.

Event Capture Action

The compare/capture channels can be used as input capture channels to capture any event from the Event System and give them a timestamp. Because all capture channels use the same event line, only one capture channel should be enabled at a time when performing event capture.

Figure 27-9 shows four capture events for one capture channel.

Figure 27-9. Input Capture Timing



When the Capture Interrupt flag is set and a new capture event is detected, there is nowhere to store the new timestamp. As a result, the Error Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.ERR) is set.

Period and Pulse-Width Capture Action

The TC can perform two input captures and restart the counter on one of the edges. This enables the TC to measure the pulse width and period. This can be used to characterize the frequency and duty cycle of an input signal:

$$f = \frac{1}{T}$$

$$dutyCycle = \frac{t_p}{T}$$

When using PPW event action, the period (T) will be captured into CC0 and the pulse width (tp) in CC1. In PWP event action, the pulse width (tp) will be captured in CC0 and the period (T) in CC1.

Selecting PWP (pulse-width, period) or PPW (period, pulse-width) in the Event Action bit group in the Event Control register (EVCTRL.EVACT) enables the TC to performs two capture actions, one on the rising edge and one on the falling edge.

The TC Inverted Event Input in the Event Control register (EVCTRL.TCINV) is used to select whether the wraparound should occur on the rising edge or the falling edge. If EVCTRL.TCINV is written to one, the wraparound will happen on the falling edge. The event source to be captured must be an asynchronous event.

To fully characterize the frequency and duty cycle of the input signal, activate capture on CC0 and CC1 by writing 0x3 to the Capture Channel x Enable bit group in the Control C register (CTRLC.CPTEN). When only one of these measurements is required, the second channel can be used for other purposes.

The TC can detect capture overflow of the input capture channels. When the Capture Interrupt flag is set and a new capture event is detected, there is nowhere to store the new timestamp. Asa result, INTFLAG.ERR is set.

27.6.3 Additional Features

27.6.3.1 One-Shot Operation

When one-shot operation is enabled, the counter automatically stops on the next counter overflow or underflow condition. When the counter is stopped, STATUS.STOP is automatically set by hardware and the waveform outputs are set to zero.

One-shot operation can be enabled by writing a one into the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) and disabled by writing a one to the One-Shot bit in the Control B Clear register (CTRLBCLR.ONESHOT). When enabled, it will count until an overflow or underflow occurs. The one-shot operation can be restarted with a retrigger command, a retrigger event or a start event.

When the counter restarts its operation, the Stop bit in the Status register (STATUS.STOP) is automatically cleared by hardware.

27.6.4 Interrupts

The TC has the following interrupt sources:

- Overflow/Underflow: OVF
- Compare or Capture Channels
- Capture Overflow Error: ERR
- Synchronization Ready: SYNCRDY

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one

to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the TC is reset. See the INTFLAG register for details on how to clear interrupt flags.

The TC has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present. Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

27.6.5 Events

The TC can generate the following output events:

- Overflow/Underflow (OVF)
- Match or Capture (MC)

Writing a one to an Event Output bit in the Event Control register (EVCTRL**.**MCEO) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event.

To enable one of the following event actions, write to the Event Action bit group (EVCTRL.EVACT).

- Start the counter
- Retrigger counter
- Increment or decrement counter (depends on counter direction)
- Capture event
- Capture period
- Capture pulse width

Writing a one to the TC Event Input bit in the Event Control register (EVCTRL.TCEI) enables input events to the TC. Writing a zero to this bit disables input events to the TC. Refer to “EVSYS – Event System” on page 301 for details on configuring the Event System.

27.6.6 Sleep Mode Operation

The TC can be configured to operate in any sleep mode. To be able to run in standby, the RUNSTDBY bit in the Control A register (CTRLA.RUNSTDBY) must be written to one. The TC can wake up the device using interrupts from any sleep mode or perform actions through the Event System.

27.6.7 Synchronization

Due to the asynchronicity between CLK_TCx_APB and GCLK_TCx some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register(STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The synchronization Ready interrupt can be used to signal when sync is complete. This can be accessed via the Synchronization Ready Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.SYNCRDY).

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)

Write-synchronization is denoted by the Write-Synchronized property in the register description.

The following registers need synchronization when written:

- Control B Clear register (CTRLBCLR)
- Control B Set register (CTRLBSET)
- Control C register (CTRLC)
- Count Value register (COUNT)
- Period Value register (PERIOD)
- Compare/Capture Value registers (CCx)

Write-synchronization is denoted by the Write-Synchronized property in the register description.

The following registers need synchronization when read:

- Control B Clear register (CTRLBCLR)
- Control B Set register (CTRLBSET)
- Control C register (CTRLC)
- Count Value register (COUNT)
- Period Value register (PERIOD)
- Compare/Capture Value registers (CCx)

Read-synchronization is denoted by the Read-Synchronized property in the register description.

27.7 Register Summary

Table 27-1. Register Summary – 8-Bit Mode Registers

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0		WAVEGEN[1:0]			MODE[1:0]		ENABLE	SWRST
0x01		15:8			PRESCSYNC[1:0]		RUNSTDBY	PRESCALER[2:0]		
0x02	READREQ	7:0				ADDR[4:0]				
0x03		15:8	RREQ	RCONT						
0x04	CTRLBCLR	7:0	CMD[1:0]					ONESHOT		DIR
0x05	CTRLBSET	7:0	CMD[1:0]					ONESHOT		DIR
0x06	CTRLC	7:0			CPTEN1	CPTEN0			INVEN1	INVEN0
0x07	Reserved									
0x08	DBGCTRL	7:0								DBGRUN
0x09	Reserved									
0x0A	EVCTRL	7:0			TCEI	TCINV		EVACT[2:0]		
0x0B		15:8			MCEO1	MCEO0				OVFEO
0x0C	INTENCLR	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0D	INTENSET	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0E	INTFLAG	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0F	STATUS	7:0	SYNCBUSY			SLAVE	STOP			
0x10	COUNT	7:0	COUNT[7:0]							
0x11	Reserved									
0x12	Reserved									
0x13	Reserved									
0x14	PER	7:0	PER[7:0]							
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	CC0	7:0	CC[7:0]							
0x19	CC1	7:0	CC[7:0]							
0x1A	Reserved									
0x1B	Reserved									
0x1C	Reserved									
0x1D	Reserved									
0x1E	Reserved									
0x1F	Reserved									

Table 27-2. Register Summary – 16-Bit Mode Registers

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0		WAVEGEN[1:0]			MODE[1:0]		ENABLE	SWRST
0x01		15:8			PRESCSYNC[1:0]		RUNSTDBY	PRESCALER[2:0]		
0x02	READREQ	7:0				ADDR[4:0]				
0x03		15:8	RREQ	RCONT						
0x04	CTRLBCLR	7:0	CMD[1:0]					ONESHOT		DIR
0x05	CTRLBSET	7:0	CMD[1:0]					ONESHOT		DIR
0x06	CTRLC	7:0			CPTEN1	CPTEN0			INVEN1	INVEN0
0x07	Reserved									
0x08	DBGCTRL	7:0								DBGRUN
0x09	Reserved									
0x0A	EVCTRL	7:0			TCEI	TCINV		EVACT[2:0]		
0x0B		15:8			MCEO1	MCEO0				OVFEO
0x0C	INTENCLR	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0D	INTENSET	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0E	INTFLAG	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0F	STATUS	7:0	SYNCBUSY			SLAVE	STOP			
0x10	COUNT	7:0	COUNT[7:0]							
0x11		15:8	COUNT[15:8]							
0x12	Reserved									
0x13	Reserved									
0x14	Reserved									
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	CC0	7:0	CC[7:0]							
0x19		15:8	CC[15:8]							
0x1A	CC1	7:0	CC[7:0]							
0x1B		15:8	CC[15:8]							
0x1C	Reserved									
0x1D	Reserved									
0x1E	Reserved									
0x1F	Reserved									

Table 27-3. Register Summary – 32-Bit Mode Registers

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0		WAVEGEN[1:0]			MODE[1:0]		ENABLE	SWRST
0x01		15:8			PRESCSYNC[1:0]		RUNSTDBY	PRESCALER[2:0]		
0x02	READREQ	7:0				ADDR[4:0]				
0x03		15:8	RREQ	RCONT						
0x04	CTRLBCLR	7:0	CMD[1:0]					ONESHOT		DIR
0x05	CTRLBSET	7:0	CMD[1:0]					ONESHOT		DIR
0x06	CTRLC	7:0			CPTEN1	CPTEN0			INVEN1	INVEN0
0x07	Reserved									
0x08	DBGCTRL	7:0								DBGRUN
0x09	Reserved									
0x0A	EVCTRL	7:0			TCEI	TCINV		EVACT[2:0]		
0x0B		15:8			MCEO1	MCEO0				OVFEO
0x0C	INTENCLR	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0D	INTENSET	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0E	INTFLAG	7:0			MC1	MC0	SYNCRDY		ERR	OVF
0x0F	STATUS	7:0	SYNCBUSY			SLAVE	STOP			
0x10	COUNT	7:0	COUNT[7:0]							
0x11		15:8	COUNT[15:8]							
0x12		23:16	COUNT[23:16]							
0x13		31:24	COUNT[31:24]							
0x14	Reserved									
0x15	Reserved									
0x16	Reserved									
0x17	Reserved									
0x18	CC0	7:0	CC[7:0]							
0x19		15:8	CC[15:8]							
0x1A		23:16	CC[23:16]							
0x1B		31:24	CC[31:24]							
0x1C	CC1	7:0	CC[7:0]							
0x1D		15:8	CC[15:8]							
0x1E		23:16	CC[23:16]							
0x1F		31:24	CC[31:24]							

27.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to the “Register Access Protection” on page 431 and the “PAC – Peripheral Access Controller” on page 27 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 440 for details.

Some registers are enable-protected, meaning they can only be written when the TC is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

27.8.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x0000

Property: Write-Protected, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
			PRESCSYNC[1:0]		RUNSTDBY	PRESCALER[2:0]		
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		WAVEGEN[1:0]			MODE[1:0]		ENABLE	SWRST
Access	R	R/W	R/W	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:14 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 13:12 – PRESCSYNC[1:0]: Prescaler and Counter Synchronization**
  These bits select whether the counter should wrap around on the next GCLK_TCx clock or the next prescaled GCLK_TCx clock. It also makes it possible to reset the prescaler.
  The options are as shown in Table 27-4.
  These bits are not synchronized.

Table 27-4. Prescaler and Counter Synchronization

Value	Name	Description
0x0	GCLK	Reload or reset the counter on next generic clock
0x1	PRESC	Reload or reset the counter on next prescaler clock
0x2	RESYNC	Reload or reset the counter on next generic clock. Reset the prescaler counter
0x3	-	Reserved

- **Bit 11 – RUNSTDBY: Run in Standby**
  This bit is used to keep the TC running in standby mode:
  0: The TC is halted in standby.
  1: The TC continues to run in standby.
  This bit is not synchronized.

- **Bits 10:8 – PRESCALER[2:0]: Prescaler**
  These bits select the counter prescaler factor, as shown in Table 27-5.
  These bits are not synchronized.

Table 27-5. Prescaler

Value	Name	Description
0x0	DIV1	Prescaler: GCLK_TC
0x1	DIV2	Prescaler: GCLK_TC/2
0x2	DIV4	Prescaler: GCLK_TC/4
0x3	DIV8	Prescaler: GCLK_TC/8
0x4	DIV16	Prescaler: GCLK_TC/16
0x5	DIV64	Prescaler: GCLK_TC/64
0x6	DIV256	Prescaler: GCLK_TC/256
0x7	DIV1024	Prescaler: GCLK_TC/1024

- **Bit 7 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 6:5 – WAVEGEN[1:0]: Waveform Generation Operation**
  These bits select the waveform generation operation. They affect the top value, as shown in “Waveform Output Operations” on page 435. It also controls whether frequency or PWM waveform generation should be used. How these modes differ can also be seen from “Waveform Output Operations” on page 435.
  These bits are not synchronized.

Table 27-6. Waveform Generation Operation

Value	Name	Operation	Top Value	Output Waveform on Match	Output Waveform on Wraparound
0x0	NFRQ	Normal frequency	PER(1)/Max	Toggle	No action
0x1	MFRQ	Match frequency	CC0	Toggle	No action
0x2	NPWM	Normal PWM	PER(1)/Max	Set	Clear
0x3	MPWM	Match PWM	CC0	Set	Clear

Note: 1. This depends on the TC mode. In 8-bit mode, the top value is the Period Value register (PER). In 16- and 32-bit mode it is the maximum value.

- **Bit 4 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 3:2 – MODE[1:0]: TC Mode**
  These bits select the TC mode, as shown in Table 27-7.
  These bits are not synchronized.

Table 27-7. TC Mode

Value	Name	Description
0x0	COUNT16	Counter in 16-bit mode
0x1	COUNT8	Counter in 8-bit mode
0x2	COUNT32	Counter in 32-bit mode
0x3	-	Reserved

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled.
  1: The peripheral is enabled.
  Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately, and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the TC, except DBGCTRL, to their initial state, and the TC will be disabled.
  Writing a one to CTRLA.SWRST will always take precedence; all other writes in the same write-operation will be discarded.
  Due to synchronization there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

27.8.2 Read Request

For a detailed description of this register and its use, refer to the"Synchronization" on page 440.

Name: READREQ

Offset: 0x02

Reset: 0x0000

Property: -

Bit	15	14	13	12	11	10	9	8
	RREQ	RCONT						
Access	W	R/W	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				ADDR[4:0]				
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 15 – RREQ: Read Request**
  Writing a zero to this bit has no effect.
  This bit will always read as zero.
  Writing a one to this bit requests synchronization of the register pointed to by the Address bit group (READREQ.ADDR) and sets the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY).

- **Bit 14 – RCONT: Read Continuously**
  0: Continuous synchronization is disabled.
  1: Continuous synchronization is enabled.
  When continuous synchronization is enabled, the register pointed to by the Address bit group (READREQ.ADDR) will be synchronized automatically every time the register is updated.

- **Bits 13:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 4:0 – ADDR[4:0]: Address**
  These bits select the offset of the register that needs read synchronization. In the TC, only COUNT and CCx are available for read synchronization.

27.8.3 Control B Clear

This register allows the user to change this register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set (CTRLBSET) register.

Name: CTRLBCLR

Offset: 0x04

Reset: 0x00

Property: Write-Protected, Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	CMD[1:0]					ONESHOT		DIR
Access	R/W	R/W	R	R	R	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – CMD[1:0]: Command**
  These bits are used for software control of retriggering and stopping the TC. When a command has been executed, the CMD bit group will read back as zero. The commands are executed on the next prescaled GCLK_TC clock cycle.
  Writing a zero to one of these bits has no effect.
  Writing a one to one of these bits will clear the pending command.

Table 27-8. Command

Value	Name	Description
0x0	NONE	No action
0x1	RETRIGGER	Force a start, restart or retrigger
0x2	STOP	Force a stop
0x3	-	Reserved

- **Bits 5:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – ONESHOT: One-Shot**
  This bit controls one-shot operation of the TC. When in one-shot mode, the TC will stop counting on the next overflow/underflow condition or a stop command.
  0: The TC will wrap around and continue counting on an overflow/underflow condition.
  1: The TC will wrap around and stop on the next underflow/overflow condition.
  Writing a zero to this bit has no effect
  Writing a one to this bit will disable one-shot operation.

- **Bit 1 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 0 – DIR: Counter Direction**
  This bit is used to change the direction of the counter.
  0: The timer/counter is counting up (incrementing).

1: The timer/counter is counting down (decrementing).
Writing a zero to this bit has no effect.
Writing a one to this bit will make the counter count up.

27.8.4 Control B Set

This register allows the user to change this register without doing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set (CTRLBCLR) register.

Name: CTRLBSET

Offset: 0x05

Reset: 0x00

Property: Write-Protected, Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	CMD[1:0]					ONESHOT		DIR
Access	R/W	R/W	R	R	R	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – CMD[1:0]: Command**
  These bits is used for software control of retriggering and stopping the TC. When a command has been executed, the CMD bit group will be read back as zero. The commands are executed on the next prescaled GCLK_TC clock cycle.
  Writing a zero to one of these bits has no effect.
  Writing a one to one of these bits will set a command.

Table 27-9. Command

Value	Name	Description
0x0	NONE	No action
0x1	RETRIGGER	Force a start, restart or retrigger
0x2	STOP	Force a stop
0x3	-	Reserved

- **Bits 5:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – ONESHOT: One-Shot**
  This bit controls one-shot operation of the TC. When active, the TC will stop counting on the next overflow/underflow condition or a stop command.
  0: The TC will wrap around and continue counting on an overflow/underflow condition.
  1: The timer/counter will wrap around and stop on the next underflow/overflow condition.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will enable one-shot operation.

- **Bit 1 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 0 – DIR: Counter Direction**
  This bit is used to change the direction of the counter.
  0: The timer/counter is counting up (incrementing).

1: The timer/counter is counting down (decrementing).
Writing a zero to this bit has no effect
Writing a one to this bit will make the counter count down.

27.8.5 Control C

Name: CTRLC

Offset: 0x06

Reset: 0x00

Property: Write-Protected, Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
			CPTEN1	CPTEN0			INVEN1	INVEN0
Access	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – CPTENx: Capture Channel x Enable**
  These bits are used to select whether channel x is a capture or a compare channel.
  Writing a one to CPTENx enables capture on channel x.
  Writing a zero to CPTENx disables capture on channel x.

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – INVENx: Output Waveform x Invert Enable**
  These bits are used to select inversion on the output of channel x.
  Writing a one to INVENx inverts the output from WO[x].
  Writing a zero to INVENx disables inversion of the output from WO[x].

27.8.6 Debug Control

Name: DBGCTRL

Offset: 0x08

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								DBGRUN
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – DBGRUN: Debug Run Mode**
  This bit is not affected by a software reset, and should not be changed by software while the TC is enabled.
  0: The TC is halted when the device is halted in debug mode.
  1: The TC continues normal operation when the device is halted in debug mode.

27.8.7 Event Control

Name: EVCTRL

Offset: 0x0A

Reset: 0x0000

Property: Write-Protected, Enable-Protected

Bit	15	14	13	12	11	10	9	8
			MCEO1	MCEO0				OVFEO
Access	R	R	R/W	R/W	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			TCEI	TCINV		EVACT[2:0]		
Access	R	R	R/W	R/W	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:14 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 13:12 – MCEOx: Match or Capture Channel x Event Output Enable**
  These bits control whether event match or capture on channel x is enabled or not and generated for every match or capture.
  0: Match/Capture event on channel x is disabled and will not be generated.
  1: Match/Capture event on channel x is enabled and will be generated for every compare/capture.
  These bits are not enable-protected.
- **Bits 11:9 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 8 – OVFEO: Overflow/Underflow Event Output Enable**
  This bit is used to enable the Overflow/Underflow event. When enabled an event will be generated when the counter overflows/underflows.
  0: Overflow/Underflow event is disabled and will not be generated.
  1: Overflow/Underflow event is enabled and will be generated for every counter overflow/underflow.
  This bit is not enable-protected.
- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 5 – TCEI: TC Event Input**
  This bit is used to enable input events to the TC.
  0: Incoming events are disabled.
  1: Incoming events are enabled.
  This bit is not enable-protected.

- **Bit 4 – TCINV: TC Inverted Event Input**
  This bit inverts the input event source when used in PWP or PPW measurement.
  0: Input event source is not inverted.
  1: Input event source is inverted.
  This bit is not enable-protected.

- **Bit 3 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 2:0 – EVACT[2:0]: Event Action**
  These bits define the event action the TC will perform on an event,as shown in Table 27-10.

Table 27-10. Event Action

Value	Name	Description
0x0	OFF	Event action disabled
0x1	RETRIGGER	Start, restart or retrigger TC on event
0x2	COUNT	Count on event
0x3	START	Start TC on event
0x4	-	Reserved
0x5	PPW	Period captured in CC0, pulse width in CC1
0x6	PWP	Period captured in CC1, pulse width in CC0
0x7	-	Reserved

27.8.8 Interrupt Enable Clear

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x0C

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
			MC1	MC0	SYNCRDY		ERR	OVF
Access	R	R/W	R/W	R/W	R/W	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – MCx: Match or Capture Channel x Interrupt Enable**
  0: The Match or Capture Channel x interrupt is disabled.
  1: The Match or Capture Channel x interrupt is enabled.
  Writing a zero to MCx has no effect.
  Writing a one to MCx will clear the corresponding Match or Capture Channel x Interrupt Disable/Enable bit, which disables the Match or Capture Channel x interrupt.

- **Bit 3 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The synchronization ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready Interrupt Disable/Enable bit, which disables the Synchronization Ready interrupt.

- **Bit 2 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 1 – ERR: Error Interrupt Enable**
  0: The Error interrupt is disabled.
  1: The Error interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Error Interrupt Disable/Enable bit, which disables the Compare interrupt.

- **Bit 0 – OVF: Overflow Interrupt Enable**
  0: The Overflow interrupt is disabled.
  1: The Overflow interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Overflow Interrupt Disable/Enable bit, which disables the Overflow interrupt.

27.8.9 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x0D

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
			MC1	MC0	SYNCRDY		ERR	OVF
Access	R	R	R/W	R/W	R/W	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – MCx: Match or Capture Channel x Interrupt Enable**
  0: The Match or Capture Channel x interrupt is disabled.
  1: The Match or Capture Channel x interrupt is enabled.
  Writing a zero to MCx has no effect.
  Writing a one to MCx will set the corresponding Match or Capture Channel x Interrupt Enable bit, which enables the Match or Capture Channel x interrupt.

- **Bit 3 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Synchronization Ready Interrupt Disable/Enable bit, which enables the Synchronization Ready interrupt.

- **Bit 2 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 1 – ERR: Error Interrupt Enable**
  0: The Error interrupt is disabled.
  1: The Error interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Error Interrupt bit, which enables the Error interrupt.

- **Bit 0 – OVF: Overflow Interrupt Enable**
  0: The Overflow interrupt is disabled.
  1: The Overflow interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Overflow Interrupt Enable bit, which enables the Overflow interrupt.

27.8.10 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x0E

Reset: 0x00

Property: -

Bit	7	6	5	4	3	2	1	0
			MC1	MC0	SYNCRDY		ERR	OVF
Access	R	R	R/W	R/W	R/W	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – MCx: Match or Capture Channel x**
  This flag is set on the next CLK_TC_CNT cycle after a match with the compare condition or once CCx register contain a valid capture value, and will generate an interrupt request if the corresponding Match or Capture Channel x Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.MCx) is one.
  Writing a zero to one of these bits has no effect.
  Writing a one to one of these bits will clear the corresponding Match or Capture Channel x interrupt flag
  In capture mode, this flag is automatically cleared when CCx register is read.

- **Bit 3 – SYNCRDY: Synchronization Ready**
  This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when the transition is caused by an enable or software reset, and will generate an interrupt request if the Synchronization Ready Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.SYNCRDY) is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready interrupt flag

- **Bit 2 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bit 1 – ERR: Error**
  This flag is set if a new capture occurs on a channel when the corresponding Match or Capture Channel x interrupt flag is one, in which case there is nowhere to store the new capture.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Error interrupt flag.

- **Bit 0 – OVF: Overflow**
  This flag is set on the next CLK_TC_CNT cycle after an overflow condition occurs, and will generate an interrupt if INTENCLR/SET.OVF is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Overflow interrupt flag.

27.8.11 Status

Name: STATUS

Offset: 0x0F

Reset: 0x08

Property: -

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY			SLAVE	STOP			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.
- **Bits 6:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 4 – SLAVE: Slave**
  This bit is set when the even-numbered master TC is set to run in 32-bit mode. The odd-numbered TC will be the slave.
- **Bit 3 – STOP: Stop**
  This bit is set when the TC is disabled, on a Stop command or on an overflow or underflow condition when the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) is one.
  0: Counter is running.
  1: Counter is stopped.
- **Bits 2:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

27.8.12 Counter Value

27.8.12.1 8-Bit Mode

Name: COUNT

Offset: 0x10

Reset: 0x00

Property: Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:0 – COUNT[7:0]: Counter Value**
  These bits contain the current counter value.

27.8.12.2 16-Bit Mode

Name: COUNT

Offset: 0x10

Reset: 0x0000

Property: Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – COUNT[15:0]: Counter Value**
  These bits contain the current counter value.

27.8.12.3 32-Bit Mode

Name: COUNT

Offset: 0x10

Reset: 0x00000000

Property: Write-Synchronized, Read-Synchronized

Bit	31	30	29	28	27	26	25	24
	COUNT[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COUNT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – COUNT[31:0]: Counter Value**
  These bits contain the current counter value.

27.8.13 Period Value

The Period Value register is available only in 8-bit TC mode. It is not available in 16-bit and 32-bit TC modes.

27.8.13.1 8-Bit Mode

Name: PER

Offset: 0x14

Reset: 0xFF

Property: Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	PER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

- **Bits 7:0 – PER[7:0]: Period Value**
  These bits contain the counter period value in 8-bitTC mode.

27.8.14 Compare/Capture

27.8.14.1 8-Bit Mode

Name: CCx

Offset: 0x18+i*0x1 [i=0..3]

Reset: 0x00

Property: Write-Synchronized, Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	CC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:0 – CC[7:0]: Compare/Capture Value**
  These bits contain the compare/capture value in 8-bit TC mode. In frequency or PWM waveform match operation (CTRLA.WAVEGEN), the CC0 register is used as a period register.

27.8.14.2 16-Bit Mode

Name: CCx

Offset: 0x18+i*0x2 [i=0..3]

Reset: 0x0000

Property: Write-Synchronized, Read-Synchronized

Bit	15	14	13	12	11	10	9	8
	CC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – CC[15:0]: Compare/Capture Value**
  These bits contain the compare/capture value in 16-bit TC mode. In frequency or PWM waveform match operation (CTRLA.WAVEGEN), the CC0 register is used as a period register.

27.8.14.3 32-Bit Mode

Name: CCx

Offset: 0x18+i*0x4 [i=0..3]

Reset: 0x00000000

Property: Write-Synchronized, Read-Synchronized

Bit	31	30	29	28	27	26	25	24
	CC[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CC[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:0 – CC[31:0]: Compare/Capture Value**

  These bits contain the compare/capture value in 32-bit TC mode. In frequency or PWM waveform match operation (CTRLA.WAVEGEN), the CC0 register is used as a period register.

28. ADC – Analog-to-Digital Converter

28.1 Overview

The Analog-to-Digital Converter (ADC) converts analog signals to digital values. The ADC has 12-bit resolution, and is capable of converting up to 350ksps. The input selection is flexible, and both differential and single-ended measurements can be performed. An optional gain stage is available to increase the dynamic range. In addition, several internal signal inputs are available. The ADC can provide both signed and unsigned results.

ADC measurements can be started by either application software or an incoming event from another peripheral in the device. ADC measurements can be started with predictable timing, and without software intervention.

Both internal and external reference voltages can be used.

An integrated temperature sensor is available for use with the ADC. The bandgap voltage as well as the scaled I/O and core voltages can also be measured by the ADC.

The ADC has a compare function for accurate monitoring of user-defined thresholds, with minimum software intervention required.

The ADC may be configured for 8-, 10- or 12-bit results, reducing the conversion time. ADC conversion results are provided left- or right-adjusted, which eases calculation when the result is represented as a signed value.

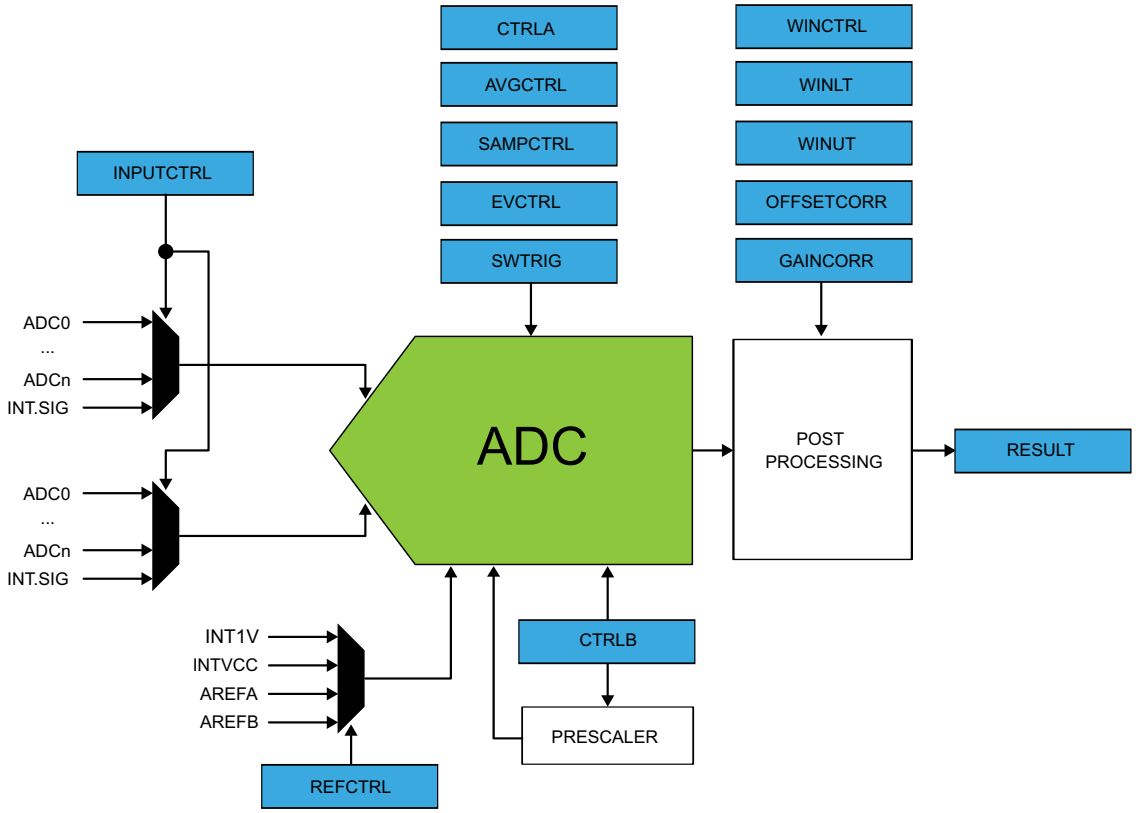
28.2 Features

- 8-, 10- or 12-bit resolution
- Up to 350,000 samples per second (350ksps)
- Differential and single-ended inputs
  - Up to 32 analog inputs
    - 25 positive and 10 negative, including internal and external
- Five internal inputs
  - Bandgap
  - Temperature sensor
  - DAC
  - Scaled core supply
  - Scaled I/O supply
- 1/2x to 16x gain
- Single, continuous and pin-scan conversion options
- Windowing monitor with selectable channel
- Conversion range:
  - V_{ref} [1v to V_{DDANA} - 0.6V]
  - ADCx * GAIN [0V to -V_{ref}]
- Built-in internal reference and external reference options
  - Four bits for reference selection

- Event-triggered conversion for accurate timing (one event input)
- Hardware gain and offset compensation
- Averaging and oversampling with decimation to support, up to 16-bit result
- Selectable sampling time

28.3 Block Diagram

Figure 28-1. ADC Block Diagram



28.4 Signal Description

Signal Name	Type	Description
AREFA	Analog input	External reference voltage A
AREFB	Analog input	External reference voltage B
ADC[19..0](1)	Analog input	Analog input channels

Note: 1. Refer to "Configuration Summary" on page 3 for details on exact number of analog input channels.

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

28.5 Product Dependencies

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

28.5.1 I/O Lines

Using the ADC's I/O lines requires the I/O pins to be configured using the port configuration (PORT).

Refer to “PORT” on page 276 for details.

28.5.2 Power Management

The ADC will continue to operate in any sleep mode where the selected source clock is running. The ADC’s interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting the sleep modes. Refer to “PM – Power Manager” on page 100 for details on the different sleep modes.

28.5.3 Clocks

The ADC bus clock (CLK_ADC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_ADC_APB can be found in the Table 15-1.

A generic clock (GCLK_ADC) is required to clock the ADC. This clock must be configured and enabled in the Generic Clock Controller (GCLK) before using the ADC. Refer “GCLK – Generic Clock Controller” on page 78 for details.

This generic clock is asynchronous to the bus clock (CLK_ADC_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to “Synchronization” on page 479 for further details.

28.5.4 Interrupts

The interrupt request line is connected to the interrupt controller. Using ADC interrupts requires the interrupt controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

28.5.5 Events

Events are connected to the Event System. Refer to “EVSYS – Event System” on page 301 for details.

28.5.6 Debug Operation

When the CPU is halted in debug mode, the ADC will halt normal operation. The ADC can be forced to continue operation during debugging. Refer to the Debug Control register (DBGCTRL) for details.

28.5.7 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following register:

- Interrupt Flag Status and Clear register (INTFLAG)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode or the CPU reset is extended, all write-protection is automatically disabled. Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

28.5.8 Analog Connections

I/O-pins AIN0 to AIN19 as well as the AREFA/AREFB reference voltage pin are analog inputs to the ADC.

28.5.9 Calibration

The values BIAS_CAL and LINEARITY_CAL from the production test must be loaded from the NVM Software Calibration Row into the ADC Calibration register (CALIB) by software to achieve specified accuracy.

Refer to “NVM Software Calibration Row Mapping” on page 22 for more details.

28.6 Functional Description

28.6.1 Principle of Operation

By default, the ADC provides results with 12-bit resolution. 8-bit or 10-bit results can be selected in order to reduce the conversion time. The ADC has an oversampling with decimation option that can extend the resolution to 16 bits. The input values can be either internal (e.g., internal temperature sensor) or external (connected I/O pins). The user can also configure whether the conversion should be single-ended or differential.

28.6.2 Basic Operation

28.6.2.1 Initialization

Before enabling the ADC, the asynchronous clock source must be selected and enabled, and the ADC reference must be configured. The first conversion after the reference is changed must not be used. All other configuration registers must be stable during the conversion. The source for GCLK_ADC is selected and enabled in the System Controller (SYSCTRL). Refer to “SYSCTRL – System Controller” on page 127 for more details.

When GCLK_ADC is enabled, the ADC can be enabled by writing a one to the Enable bit in the Control Register A (CTRLA.ENABLE).

28.6.2.2 Enabling, Disabling and Reset

The ADC is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The ADC is disabled by writing a zero to CTRLA.ENABLE.

The ADC is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the ADC, except DBGCTRL, will be reset to their initial state, and the ADC will be disabled. Refer to the CTRLA register for details.

The ADC must be disabled before it is reset.

28.6.2.3 Basic Operation

In the most basic configuration, the ADC sample values from the configured internal or external sources (INPUTCTRL register). The rate of the conversion is dependent on the combination of the GCLK_ADC frequency and the clock prescaler.

To convert analog values to digital values, the ADC needs first to be initialized, as described in “Initialization” on page 472. Data conversion can started either manually, by writing a one to the Start bit in the Software Trigger register (SWTRIG.START), or automatically, by configuring an automatic trigger to initiate the conversions. A free-running mode could be used to continuously convert an input channel. There is no need for a trigger to start the conversion. It will start automatically at the end of previous conversion.

The automatic trigger can be configured to trigger on many different conditions.

The result of the conversion is stored in the Result register (RESULT) as it becomes available, overwriting the result from the previous conversion.

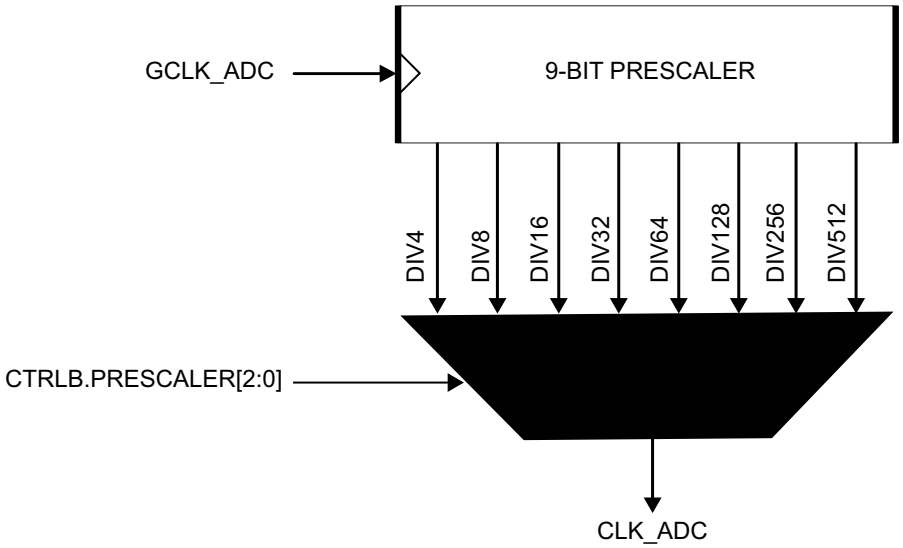
To avoid data loss if more than one channel is enabled, the conversion result must be read as it becomes available (INTFLAG.RESRDY). Failing to do so will result in an overrun error condition, indicated by the OVERRUN bit in the Interrupt Flag Status and Clear register (INTFLAG.OVERRUN).

To use an interrupt handler, the corresponding bit in the Interrupt Enable Set register (INTENSET) must be written to one.

28.6.3 Prescaler

The ADC is clocked by GCLK_ADC. There is also a prescaler in the ADC to enable conversion at lower clock rates. Refer to CTRLB for details on prescaler settings.

Figure 28-2. ADC Prescaler



The propagation delay of an ADC measurement is given by:

$$PropagationDelay = \frac{1 + \frac{Resolution}{2} + DelayGain}{f_{ADC}}$$

Table 28-1. Delay Gain

INTPUTCTRL.GAIN[3:0]	Delay Gain (in CLK_ADC Period)	
	Differential Mode	Single-Ended Mode
0x0	0	0
0x1	0	1
0x2	1	1
0x3	1	2
0x4	2	2
0x5 ... 0xE	Reserved	Reserved
0xF	0	1

28.6.4 ADC Resolution

The ADC supports 8-bit, 10-bit and 12-bit resolutions. Resolution can be changed by writing the Resolution bit group in the Control B register (CTRLB.RESSEL). After a reset, the resolution is set to 12 bits by default.

28.6.5 Differential and Single-Ended Conversions

The ADC has two conversion options: differential and single-ended. When measuring signals where the positive input is always at a higher voltage than the negative input, the single-ended conversion should be used in order to have full 12-bit resolution in the conversion, which has only positive values. The negative input must be connected to ground. This ground could be the internal GND, IOGND or an external ground connected to a pin. Refer to INPUTCTRL for selection details. If 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 is done in the Differential Mode bit in the Control B register (CTRLB.DIFFMODE). These two types of conversion could be run in single mode or in free-running mode. When set up in free-running mode, an ADC input will continuously sample and do new conversions. The INTFLAG.RESRDY bit will be set at the end of each conversion.

28.6.5.1 Conversion Timing

Figure 28-3 shows the ADC timing for a single conversion without gain. The writing of the ADC Start Conversion bit (SWTRIG.START) or Start Conversion Event In bit (EVCTRL.STARTEI) must occur at least one CLK_ADC_APB cycle before the CLK_ADC cycle on which the conversion starts. The input channel is sampled in the first half CLK_ADC period. The sampling time can be increased by using the Sampling Time Length bit group in the Sampling Time Control register (SAMPCTRL.SAMPLEN). Refer to Figure 28-4 for example on increased sampling time.

Figure 28-3. ADC Timing for One Conversion in Differential Mode without Gain

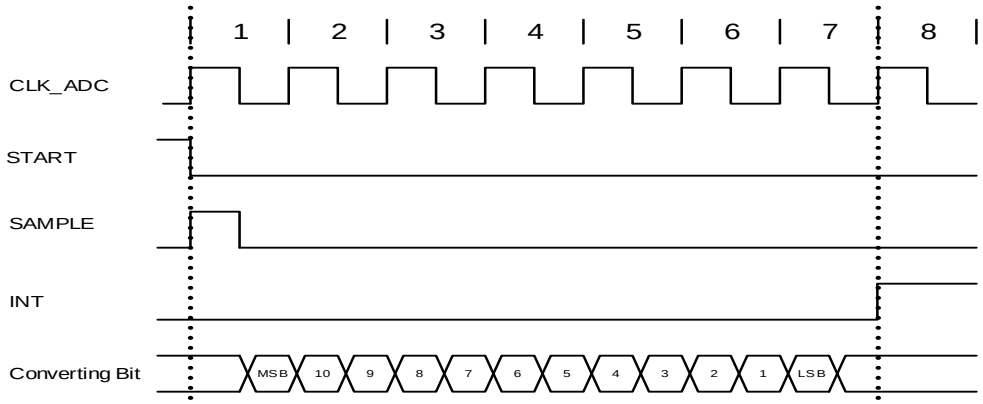


Figure 28-4. ADC Timing for One Conversion in Differential Mode without Gain, but with Increased Sampling Time

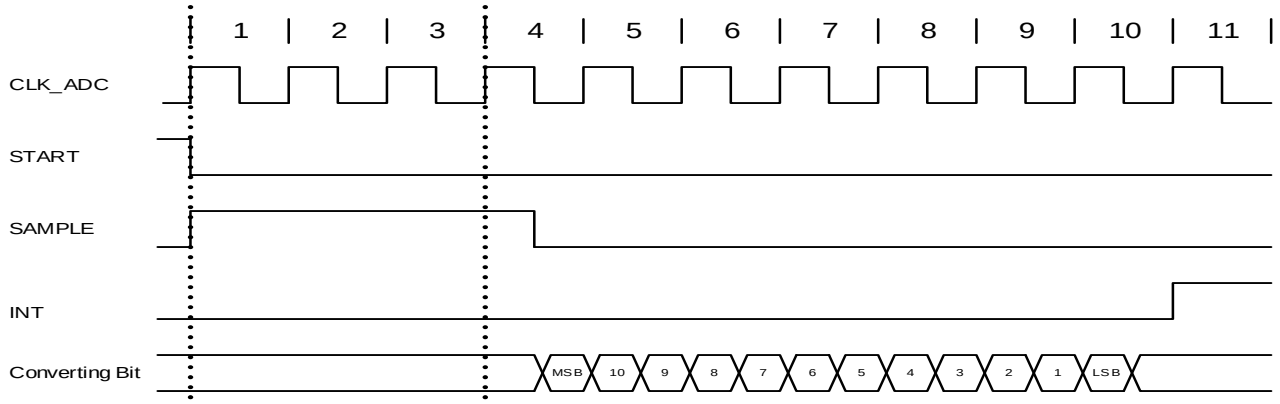


Figure 28-5. ADC Timing for Free Running in Differential Mode without Gain

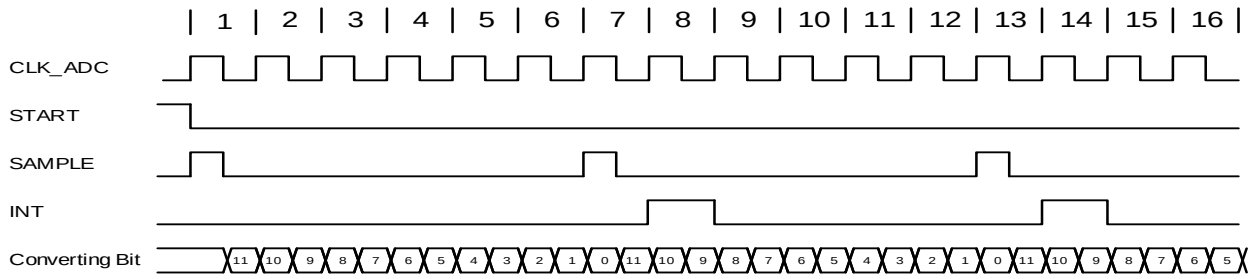


Figure 28-6. ADC Timing for One Conversion in Single-Ended Mode without Gain

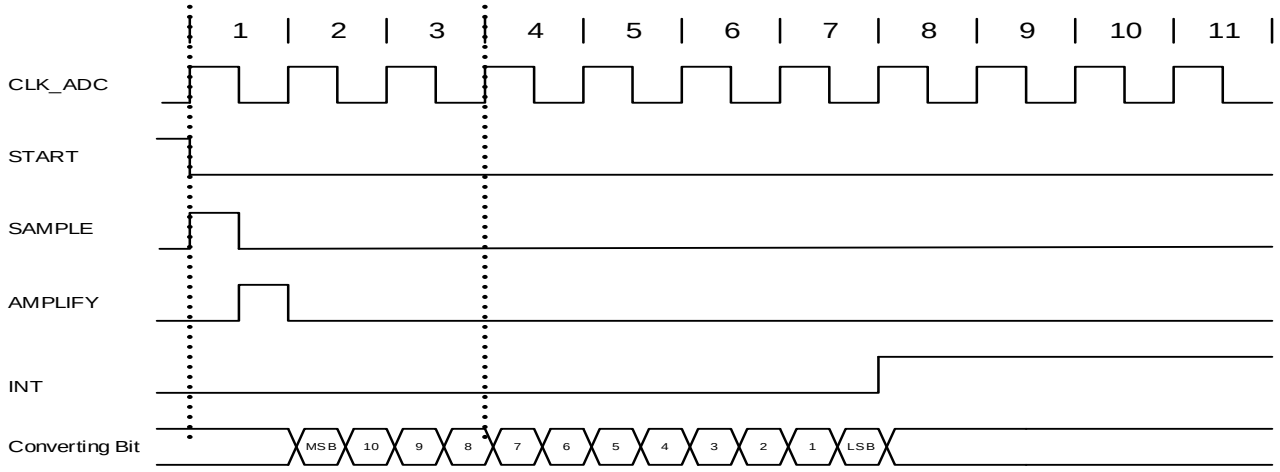
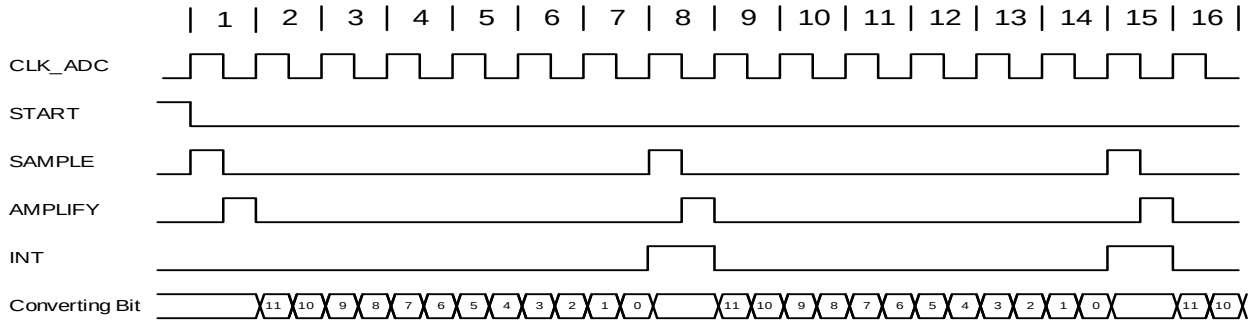


Figure 28-7. ADC Timing for Free Running in Single-Ended Mode without Gain



28.6.6 Accumulation

The result from multiple consecutive conversions can be accumulated. The number of samples to be accumulated is specified by writing to the Number of Samples to be Collected field in the Average Control register (AVGCTRL.SAMPLENUM) as described in Table 28-2. When accumulating more than 16 samples, the result will be too large for the 16-bit RESULT register. To avoid overflow, the result is shifted right automatically to fit within the 16 available bits. The number of automatic right shifts are specified in Table 28-2. Note that to be able to perform the accumulation of two or more samples, the Conversion Result Resolution field in the Control B register (CTRLB.RESSEL) must be written to one.

Table 28-2. Accumulation

Number of Accumulated Samples	AVGCTRL. SAMPLENUM	Intermediate Result Precision	Number of Automatic Right Shifts	Final Result Precision	Automatic Division Factor
1	0x0	12 bits	0	12 bits	0
2	0x1	13 bits	0	13 bits	0
4	0x2	14 bits	0	14 bits	0
8	0x3	15 bits	0	15 bits	0
16	0x4	16 bits	0	16 bits	0
32	0x5	17 bits	1	16 bits	2
64	0x6	18 bits	2	16 bits	4
128	0x7	19 bits	3	16 bits	8
256	0x8	20 bits	4	16 bits	16
512	0x9	21 bits	5	16 bits	32
1024	0xA	22 bits	6	16 bits	64
Reserved	0xB –0xF	12 bits		12 bits	0

28.6.7 Averaging

Averaging is a feature that increases the sample accuracy, though at the cost of reduced sample rate. This feature is suitable when operating in noisy conditions. Averaging is done by accumulating *m* samples, as described in “Accumulation” on page 476, and divide the result by *m*. The averaged result is available in the RESULT register. The number of samples to be accumulated is specified by writing to AVGCTRL.SAMPLENUM as described in Table 28-3. The division is obtained by a combination of the automatic right shift described above, and an additional right shift that must be specified by writing to the Adjusting Result/Division Coefficient field in AVGCTRL (AVGCTRL.ADJRES) as described in Table 28-3. Note that to be able to perform the averaging of two or more samples, the Conversion Result Resolution field in the Control B register (CTRLB.RESSEL) must be written to one.

Averaging AVGCTRL.SAMPLENUM samples will reduce the effective sample rate by $\frac{1}{\text{AVGCTRL.SAMPLENUM}}$.

When the required average is reached, the INTFLAG.RESRDY bit is set.

Table 28-3. Averaging

Number of Accumulated Samples	AVGCTRL. SAMPLENUM	Intermediate Result Precision	Number of Automatic Right Shifts	Division Factor	AVGCTRL. ADJRES	Total Number of Right Shifts	Final Result Precision	Automatic Division Factor
1	0x0	12 bits	0	1	0x0		12 bits	0
2	0x1	13	0	2	0x1	1	12 bits	0
4	0x2	14	0	4	0x2	2	12 bits	0
8	0x3	15	0	8	0x3	3	12 bits	0
16	0x4	16	0	16	0x4	4	12 bits	0
32	0x5	17	1	16	0x4	5	12 bits	2
64	0x6	18	2	16	0x4	6	12 bits	4
128	0x7	19	3	16	0x4	7	12 bits	8
256	0x8	20	4	16	0x4	8	12 bits	16
512	0x9	21	5	16	0x4	9	12 bits	32
1024	0xA	22	6	16	0x4	10	12 bits	64
Reserved	0xB –0xF				0x0		12 bits	0

28.6.8 Oversampling and Decimation

By using oversampling and decimation, the ADC resolution can be increased from 12 bits to up to 16 bits. To increase the resolution by *n* bits, 4^n samples must be accumulated. The result must then be shifted right by *n* bits. This right shift is a combination of the automatic right shift and the value written to AVGCTRL.ADJRES. To obtain the correct resolution, the ADJRES must be configured as described in the table below. This method will result in *n* bit extra LSB resolution.

Table 28-4. Configuration Required for Oversampling and Decimation

Result Resolution	Number of Samples to Average	AVGCTRL.SAMPLENUM[3:0]	Number of Automatic Right Shifts	AVGCTRL.ADJRES[2:0]
13 bits	$4^1 = 4$	0x2	0	0x1
14 bits	$4^2 = 16$	0x4	0	0x2
15 bits	$4^3 = 64$	0x6	2	0x1
16 bits	$4^4 = 256$	0x8	4	0x0

28.6.9 Window Monitor

The window monitor allows the conversion result to be compared to some predefined threshold values. Supported modes are selected by writing the Window Monitor Mode bit group in the Window Monitor Control register (WINCTRL.WINMODE[2:0]). Thresholds are given by writing the Window Monitor Lower Threshold register (WINLT) and Window Monitor Upper Threshold register (WINUT).

If differential input is selected, the WINLT and WINUT are evaluated as signed values. Otherwise they are evaluated as unsigned values.

Another important point is that the significant WINLT and WINUT bits are given by the precision selected in the Conversion Result Resolution bit group in the Control B register (CTRLB.RESSEL). This means that if 8-bit mode is selected, only the eight lower bits will be considered. In addition, in differential mode, the eighth bit will be considered as the sign bit even if the ninth bit is zero.

The INTFLAG.WINMON interrupt flag will be set if the conversion result matches the window monitor condition.

28.6.10 Offset and Gain Correction

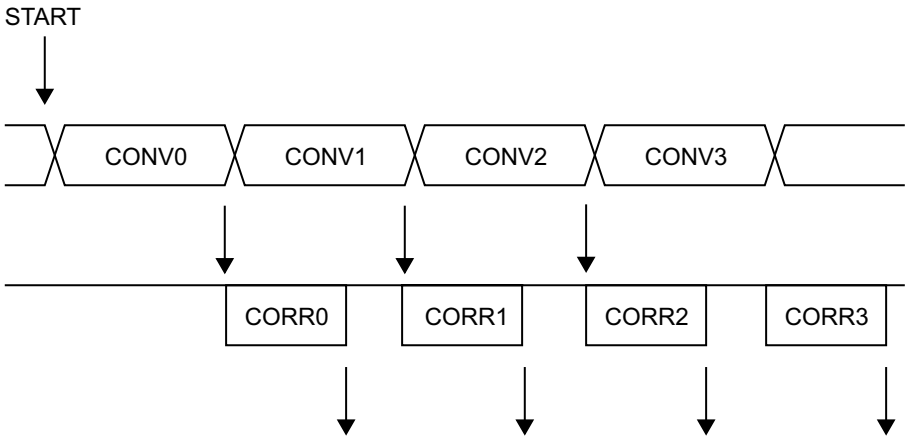
Inherent gain and offset errors affect the absolute accuracy of the ADC. The offset error is defined as the deviation of the actual ADC's transfer function from an ideal straight line at zero input voltage. The offset error cancellation is handled by the Offset Correction register (OFFSETCORR). The offset correction value is subtracted from the converted data before writing the Result register (RESULT). The gain error is defined as the deviation of the last output step's midpoint from the ideal straight line, after compensating for offset error. The gain error cancellation is handled by the Gain Correction register (GAINCORR). To correct these two errors, the Digital Correction Logic Enabled bit in the Control B register (CTRLB.CORREN) must be written to one.

Offset and gain error compensation results are both calculated according to:

$$\text{Result} = (\text{Conversion value} - \text{OFFSETCORR}) \cdot \text{GAINCORR}$$

In single conversion, a latency of 13 GCLK_ADC is added to the availability of the final result. Since the correction time is always less than the propagation delay, this latency appears in free-running mode only during the first conversion. After that, a new conversion will be initialized when a conversion completes. All other conversion results are available at the defined sampling rate.

Figure 28-8. ADC Timing Correction Enabled



28.6.11 Interrupts

The ADC has the following interrupt sources:

- Result Conversion Ready: RESRDY
- Overrun: OVERRUN
- Window Monitor: WINMON
- Synchronization Ready: SYNCRDY

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one

to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR) register. An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the peripheral is reset. An interrupt flag is cleared by writing a one to the corresponding bit in the INTFLAG register. Each peripheral can have one interrupt request line per interrupt source or one common interrupt request line for all the interrupt sources. This is device dependent.

Refer to “Nested Vector Interrupt Controller” on page 24 for details. If the peripheral has one common interrupt request line for all the interrupt sources, the user must read the INTFLAG register to determine which interrupt condition is present.

28.6.12 Events

The peripheral can generate the following output events:

- Result Ready (RESRDY)
- Window Monitor (WINMON)

Output events must be enabled to be generated. Writing a one to an Event Output bit in the Event Control register (EVCTRL.xxEO) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. The events must be correctly routed in the Event System. Refer to “EVSYS – Event System” on page 301 for details.

The peripheral can take the following actions on an input event:

- ADC start conversion (START)
- ADC conversion flush (FLUSH)

Input events must be enabled for the corresponding action to be taken on any input event. Writing a one to an Event Input bit in the Event Control register (EVCTRL.xxEI) enables the corresponding action on the input event. Writing a zero to this bit disables the corresponding action on the input event. Note that if several events are connected to the peripheral, the enabled action will be taken on any of the incoming events. The events must be correctly routed in the Event System. Refer to “EVSYS – Event System” on page 301 for details.

28.6.13 Sleep Mode Operation

The Run in Standby bit in the Control A register (CTRLA.RUNSTDBY) controls the behavior of the ADC during standby sleep mode. When the bit is zero, the ADC is disabled during sleep, but maintains its current configuration. When the bit is one, the ADC continues to operate during sleep. Note that when RUNSTDBY is zero, the analog blocks are powered off for the lowest power consumption. This necessitates a start-up time delay when the system returns from sleep.
When RUNSTDBY is one, any enabled ADC interrupt source can wake up the CPU. However, ADC conversion will be triggerable by events only while the CPU is idle.

28.6.14 Synchronization

Due to the asynchronicity between CLK_ADC_APB and GCLK_ADC, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The Synchronization Ready interrupt can be used to signal when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)

The following registers need synchronization when written:

- Control B (CTRLB)
- Software Trigger (SWTRIG)
- Window Monitor Control (WINCTRL)
- Input Control (INPUTCTRL)
- Window Upper/Lower Threshold (WINUT/WINLT)

Write-synchronization is denoted by the Write-Synchronized property in the register description.

The following registers need synchronization when read:

- Software Trigger (SWTRIG)
- Input Control (INPUTCTRL)
- Result (RESULT)

Read-synchronization is denoted by the Read-Synchronized property in the register description.

28.7 Register Summary

Offset	Name	Bit pos.								
0x00	CTRLA	7:0						RUNSTDBY	ENABLE	SWRST
0x01	REFCTRL	7:0	REFCOMP				REFSEL[3:0]			
0x02	AVGCTRL	7:0		ADJRES[2:0]			SAMPLENUM[3:0]			
0x03	SAMPCTRL	7:0			SAMPLEN[5:0]					
0x04	CTRLB	7:0			RESSEL[1:0]		CORREN	FREERUN	LEFTADJ	DIFFMODE
0x05		15:8						PRESCALER[2:0]		
0x06	Reserved									
0x07	Reserved									
0x08	WINCTRL	7:0						WINMODE[2:0]		
0x09	Reserved									
0x0A	Reserved									
0x0B	Reserved									
0x0C	SWTRIG	7:0							START	FLUSH
0x0D	Reserved									
0x0E	Reserved									
0x0F	Reserved									
0x10	INPUTCTRL	7:0				MUXPOS[4:0]				
0x11		15:8				MUXNEG[4:0]				
0x12		23:16	INPUTOFFSET[3:0]				INPUTSCAN[3:0]			
0x13		31:24					GAIN[3:0]			
0x14	EVCTRL	7:0			WINMONEO	RESRDYEO			SYNCEI	STARTEI
0x15	Reserved									
0x16	INTENCLR	7:0					SYNCRDY	WINMON	OVERRUN	RESRDY
0x17	INTENSET	7:0					SYNCRDY	WINMON	OVERRUN	RESRDY
0x18	INTFLAG	7:0					SYNCRDY	WINMON	OVERRUN	RESRDY
0x19	STATUS	7:0	SYNCBUSY							
0x1A	RESULT	7:0	RESULT[7:0]							
0x1B		15:8	RESULT[15:8]							
0x1C	WINLT	7:0	WINLT[7:0]							
0x1D		15:8	WINLT[15:8]							
0x1E	Reserved									
0x1F	Reserved									
0x20	WINUT	7:0	WINUT[7:0]							
0x21		15:8	WINUT[15:8]							
0x22	Reserved									
0x23	Reserved									

Offset	Name	Bit pos.								
0x24	GAINCORR	7:0	GAINCORR[7:0]							
0x25		15:8					GAINCORR[11:8]			
0x26	OFFSETCORR	7:0	OFFSETCORR[7:0]							
0x27		15:8					OFFSETCORR[11:8]			
0x28	CALIB	7:0	LINEARITY_CAL[7:0]							
0x29		15:8						BIAS_CAL[2:0]		
0x2A	DBGCTRL	7:0								DBGRUN

28.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 471 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 479 for details.

Some registers are enable-protected, meaning they can be written only when the ADC is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

28.8.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						RUNSTDBY	**ENABLE**	**SWRST**
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RUNSTDBY: Run in Standby**
  This bit indicates whether the ADC will continue running in standby sleep mode or not:
  0: The ADC is halted during standby sleep mode.
  1: The ADC continues normal operation during standby sleep mode.

- **Bit 1 – ENABLE: Enable**
  0: The ADC is disabled.
  1: The ADC is enabled.
  Due to synchronization, there is a delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY will be cleared when the operation is complete.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the ADC, except DBGCTRL, to their initial state, and the ADC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

28.8.2 Reference Control

Name: REFCTRL

Offset: 0x01

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	REFCOMP				REFSEL[3:0]			
Access	R/W	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – REFCOMP: Reference Buffer Offset Compensation Enable**
  The accuracy of the gain stage can be increased by enabling the reference buffer offset compensation. This will decrease the input impedance and thus increase the start-up time of the reference.
  0: Reference buffer offset compensation is disabled.
  1: Reference buffer offset compensation is enabled.

- **Bits 6:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 3:0 – REFSEL[3:0]: Reference Selection**
  These bits select the reference for the ADC according to Table 28-5.

Table 28-5. Reference Selection

Value	Name	Description
0x0	INT1V	1.0V voltage reference
0x1	INTVCC0	1/1.48 VDDANA
0x2	INTVCC1	1/2 VDDANA (only for VDDANA > 2.0V)
0x3	AREFA	External reference
0x4	AREFB	External reference
0x5-0xF	Reserved	Reserved

28.8.3 Average Control

Name: AVGCTRL

Offset: 0x02

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
		ADJRES[2:0]			SAMPLENUM[3:0]			
Access	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.
- **Bits 6:4 – ADJRES[2:0]: Adjusting Result / Division Coefficient**
  These bits define the division coefficient in 2^n steps.
- **Bits 3:0 – SAMPLENUM[3:0]: Number of Samples to be Collected**
  These bits define how many samples should be added together.The result will be available in the Result register (RESULT). Note: if the result width increases, CTRLB.RESSEL must be changed.

Table 28-6. Number of Samples to be Collected

Value	Name	Description
0x0		1 sample
0x1		2 samples
0x2		4 samples
0x3		8 samples
0x4		16 samples
0x5		32 samples
0x6		64 samples
0x7		128 samples
0x8		256 samples
0x9		512 samples
0xA		1024 samples
0xB-0xF		Reserved

28.8.4 Sampling Time Control

Name: SAMPCTRL

Offset: 0x03

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
			SAMPLEN[5:0]					
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:0 – SAMPLEN[5:0]: Sampling Time Length**
  These bits control the ADC sampling time in number of half CLK_ADC cycles, depending of the prescaler value, thus controlling the ADC input impedance. Sampling time is set according to the equation:

$$\text{Sampling time} = (SAMPLEN + 1) \cdot \left(\frac{CLK_{ADC}}{2}\right)$$

28.8.5 Control B

Name: CTRLB

Offset: 0x04

Reset: 0x0000

Property: Write-Synchronized, Write-Protected

Bit	15	14	13	12	11	10	9	8
						PRESCALER[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			RESSEL[1:0]		CORREN	FREERUN	LEFTADJ	DIFFMODE
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:11 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 10:8 – PRESCALER[2:0]: Prescaler Configuration**
  These bits define the ADC clock relative to the peripheral clock according to Table 28-7. These bits can only be written while the ADC is disabled.

Table 28-7. Prescaler Configuration

Value	Name	Description
0x0	DIV4	Peripheral clock divided by 4
0x1	DIV8	Peripheral clock divided by 8
0x2	DIV16	Peripheral clock divided by 16
0x3	DIV32	Peripheral clock divided by 32
0x4	DIV64	Peripheral clock divided by 64
0x5	DIV128	Peripheral clock divided by 128
0x6	DIV256	Peripheral clock divided by 256
0x7	DIV512	Peripheral clock divided by 512

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – RESSEL[1:0]: Conversion Result Resolution**
  These bits define whether the ADC completes the conversion at 12-, 10- or 8-bit result resolution. These bits can be written only while the ADC is disabled.

Table 28-8. Conversion Result Resolution

Value	Name	Description
0x0	12BIT	12-bit result
0x1	16BIT	For averaging mode output
0x2	10BIT	10-bit result
0x3	8BIT	8-bit result

- **Bit 3 – CORREN: Digital Correction Logic Enabled**
  0: Disable the digital result correction.
  1: Enable the digital result correction. The ADC conversion result in the RESULT register is then corrected for gain and offset based on the values in the GAINCAL and OFFSETCAL registers. Conversion time will be increased by X cycles according to the value in the Offset Correction Value bit group in the Offset Correction register.
  This bit can be changed only while the ADC is disabled.

- **Bit 2 – FREERUN: Free Running Mode**
  0: The ADC run is single conversion mode.
  1: The ADC is in free running mode and a new conversion will be initiated when a previous conversion completes.
  This bit can be changed only while the ADC is disabled.

- **Bit 1 – LEFTADJ: Left-Adjusted Result**
  0: The ADC conversion result is right-adjusted in the RESULT register.
  1: The ADC conversion result is left-adjusted in the RESULT register. The high byte of the 12-bit result will be present in the upper part of the result register. Writing this bit to zero (default) will right-adjust the value in the RESULT register.
  This bit can be changed only while the ADC is disabled.

- **Bit 0 – DIFFMODE: Differential Mode**
  0: The ADC is running in singled-ended mode.
  1: The ADC is running in differential mode. In this mode, the voltage difference between the MUXPOS and MUX-NEG inputs will be converted by the ADC.
  This bit can be changed only while the ADC is disabled.

28.8.6 Window Monitor Control

Name: WINCTRL

Offset: 0x08

Reset: 0x00

Property: Write-Synchronized, Write-Protected

Bit	7	6	5	4	3	2	1	0
						WINMODE[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:0 – WINMODE[2:0]: Window Monitor Mode**
  These bits enable and define the window monitor mode. Table 28-9 shows the mode selections.

Table 28-9. Window Monitor Mode

Value	Name	Description
0x0		No window mode (default)
0x1		Mode 1: RESULT > WINLT
0x2		Mode 2: RESULT < WINUT
0x3		Mode 3: WINLT < RESULT < WINUT
0x4		Mode 4:!(WINLT < RESULT < WINUT)
0x5-0x7		Reserved

28.8.7 Software Trigger

Name: SWTRIG

Offset: 0x0C

Reset: 0x00

Property: Write-Synchronized, Write-Protected

Bit	7	6	5	4	3	2	1	0
							START	**FLUSH**
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – START: ADC Start Conversion**
  0: The ADC will not start a conversion.
  1: The ADC will start a conversion. The bit is cleared by hardware when the conversion has started. Setting this bit when it is already set has no effect.
  Writing this bit to zero will have no effect.

- **Bit 0 – FLUSH: ADC Conversion Flush**
  0: No flush action.
  1: The ADC pipeline will be flushed. A flush will restart the ADC clock on the next peripheral clock edge, and all conversions in progress will be aborted and lost. This bit is cleared until the ADC has been flushed.
  After the flush, the ADC will resume where it left off; i.e., if a conversion was pending, the ADC will start a new conversion.
  Writing this bit to zero will have no effect.

28.8.8 Input Control

Name: INPUTCTRL

Offset: 0x10

Reset: 0x00000000

Property: Write-Synchronized, Write-Protected

Bit	31	30	29	28	27	26	25	24
					GAIN[3:0]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	INPUTOFFSET[3:0]				INPUTSCAN[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MUXNEG[4:0]				
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				MUXPOS[4:0]				
Access	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:28 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 27:24 – GAIN[3:0]: Gain Factor Selection**
  These bits set the gain factor of the ADC gain stage according to the values shown in Table 28-10.

Table 28-10. Gain Factor Selection

Value	Name	Description
0x0	1X	1x
0x1	2X	2x
0x2	4X	4x
0x3	8X	8x
0x4	16X	16x
0x5-0xE	–	Reserved
0xF	DIV2	1/2x

- **Bits 23:20 – INPUTOFFSET[3:0]: Positive Mux Setting Offset**
  The pin scan is enabled when INPUTSCAN != 0. Writing these bits to a value other than zero causes the first conversion triggered to be converted using a positive input equal to MUXPOS + INPUTOFFSET. Setting this register to zero causes the first conversion to use a positive input equal to MUXPOS.

  After a conversion, the INPUTOFFSET register will be incremented by one, causing the next conversion to be done with the positive input equal to MUXPOS + INPUTOFFSET. The sum of MUXPOS and INPUTOFFSET gives the input that is actually converted.

- **Bits 19:16 – INPUTSCAN[3:0]: Number of Input Channels Included in Scan**
  This register gives the number of input sources included in the pin scan. The number of input sources included is INPUTSCAN + 1. The input channels included are in the range from MUXPOS + INPUTOFFSET to MUXPOS + INPUTOFFSET + INPUTSCAN.

  The range of the scan mode must not exceed the number of input channels available on the device.

- **Bits 15:13 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 12:8 – MUXNEG[4:0]: Negative Mux Input Selection**
  These bits define the Mux selection for the negative ADC input. Table 28-11 shows the possible input selections.

Table 28-11. Negative Mux Input Selection

Value	Name	Description
0x00	PIN0	ADC AIN0 pin
0x01	PIN1	ADC AIN1 pin

Table 28-11. Negative Mux Input Selection (Continued)

Value	Name	Description
0x02	PIN2	ADC AIN2 pin
0x03	PIN3	ADC AIN3 pin
0x04	PIN4	ADC AIN4 pin
0x05	PIN5	ADC AIN5 pin
0x06	PIN6	ADC AIN6 pin
0x07	PIN7	ADC AIN7 pin
0x08-0x17	–	Reserved
0x18	GND	Internal ground
0x19	IOGND	I/O ground
0x1A-0x1F	–	Reserved

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 4:0 – MUXPOS[4:0]: Positive Mux Input Selection**
  These bits define the Mux selection for the positive ADC input. Table 28-12 shows the possible input selections. If the internal bandgap voltage or temperature sensor input channel is selected, then the Sampling Time Length bit group in the Sampling Control register must be written with a corresponding value.

Table 28-12. Positive Mux Input Selection

MUXPOS[4:0]	Group configuration	Description
0x00	PIN0	ADC AIN0 pin
0x01	PIN1	ADC AIN1 pin
0x02	PIN2	ADC AIN2 pin
0x03	PIN3	ADC AIN3 pin
0x04	PIN4	ADC AIN4 pin
0x05	PIN5	ADC AIN5 pin
0x06	PIN6	ADC AIN6 pin
0x07	PIN7	ADC AIN7 pin
0x08	PIN8	ADC AIN8 pin
0x09	PIN9	ADC AIN9 pin
0x0A	PIN10	ADC AIN10 pin
0x0B	PIN11	ADC AIN11 pin
0x0C	PIN12	ADC AIN12 pin
0x0D	PIN13	ADC AIN13 pin

Table 28-12. Positive Mux Input Selection (Continued)

MUXPOS[4:0]	Group configuration	Description
0x0E	PIN14	ADC AIN14 pin
0x0F	PIN15	ADC AIN15 pin
0x10	PIN16	ADC AIN16 pin
0x11	PIN17	ADC AIN17 pin
0x12	PIN18	ADC AIN18 pin
0x13	PIN19	ADC AIN19 pin
0x14-0x17		Reserved
0x18	TEMP	Temperature reference
0x19	BANDGAP	Bandgap voltage
0x1A	SCALEDCOREVCC	1/4 scaled core supply
0x1B	SCALEDIOVCC	1/4 scaled I/O supply
0x1C	DAC	DAC output
0x1D-0x1F		Reserved

28.8.9 Event Control

Name: EVCTRL

Offset: 0x14

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
			WINMONEO	RESRDYEO			SYNCEI	STARTEI
Access	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 5 – WINMONEO: Window Monitor Event Out**
  This bit indicates whether the Window Monitor event output is enabled or not and an output event will be generated when the window monitor detects something.
  0: Window Monitor event output is disabled and an event will not be generated.
  1: Window Monitor event output is enabled and an event will be generated.

- **Bit 4 – RESRDYEO: Result Ready Event Out**
  This bit indicates whether the Result Ready event output is enabled or not and an output event will be generated when the conversion result is available.
  0: Result Ready event output is disabled and an event will not be generated.
  1: Result Ready event output is enabled and an event will be generated.

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – SYNCEI: Synchronization Event In**
  0: A flush and new conversion will not be triggered on any incoming event.
  1: A flush and new conversion will be triggered on any incoming event.

- **Bit 0 – STARTEI: Start Conversion Event In**
  0: A new conversion will not be triggered on any incoming event.
  1: A new conversion will be triggered on any incoming event.

28.8.10 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x16

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
					SYNCRDY	WINMON	OVERRUN	RESRDY
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 3 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled, and an interrupt request will be generated when the Synchronization Ready interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit and the corresponding interrupt request.

- **Bit 2 – WINMON: Window Monitor Interrupt Enable**
  0: The window monitor interrupt is disabled.
  1: The window monitor interrupt is enabled, and an interrupt request will be generated when the Window Monitor interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Window Monitor Interrupt Enable bit and the corresponding interrupt request.

- **Bit 1 – OVERRUN: Overrun Interrupt Enable**
  0: The Overrun interrupt is disabled.
  1: The Overrun interrupt is enabled, and an interrupt request will be generated when the Overrun interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Overrun Interrupt Enable bit and the corresponding interrupt request.

- **Bit 0 – RESRDY: Result Ready Interrupt Enable**
  0: The Result Ready interrupt is disabled.
  1: The Result Ready interrupt is enabled, and an interrupt request will be generated when the Result Ready interrupt flag is set.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Result Ready Interrupt Enable bit and the corresponding interrupt request.

28.8.11 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x17

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
					SYNCRDY	WINMON	OVERRUN	RESRDY
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 3 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit, which enables the Synchronization Ready interrupt.

- **Bit 2 – WINMON: Window Monitor Interrupt Enable**
  0: The Window Monitor interrupt is disabled.
  1: The Window Monitor interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Window Monitor Interrupt bit and enable the Window Monitor interrupt.

- **Bit 1 – OVERRUN: Overrun Interrupt Enable**
  0: The Overrun interrupt is disabled.
  1: The Overrun interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Overrun Interrupt bit and enable the Overrun interrupt.

- **Bit 0 – RESRDY: Result Ready Interrupt Enable**
  0: The Result Ready interrupt is disabled.
  1: The Result Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Result Ready Interrupt bit and enable the Result Ready interrupt.

28.8.12 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x18

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
					SYNCRDY	WINMON	OVERRUN	RESRDY
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 3 – SYNCRDY: Synchronization Ready**
  This flag is cleared by writing a one to the flag.
  This flag is set on a one-to-zero transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when caused by an enable or software reset, and will generate an interrupt request if INTENCLR/SET.SYNCRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Synchronization Ready interrupt flag.

- **Bit 2 – WINMON: Window Monitor**
  This flag is cleared by writing a one to the flag or by reading the RESULT register.
  This flag is set on the next GCLK_ADC cycle after a match with the window monitor condition, and an interrupt request will be generated if INTENCLR/SET.WINMON is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Window Monitor interrupt flag.

- **Bit 1 – OVERRUN: Overrun**
  This flag is cleared by writing a one to the flag.
  This flag is set if RESULT is written before the previous value has been read by CPU, and an interrupt request will be generated if INTENCLR/SET.OVERRUN is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Overrun interrupt flag.

- **Bit 0 – RESRDY: Result Ready**
  This flag is cleared by writing a one to the flag or by reading the RESULT register.
  This flag is set when the conversion result is available, and an interrupt will be generated if INTENCLR/SET.RESRDY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Result Ready interrupt flag.

28.8.13 Status

Name: STATUS

Offset: 0x19

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

28.8.14 Result

Name: RESULT

Offset: 0x1A

Reset: 0x0000

Property: Read-Synchronized

Bit	15	14	13	12	11	10	9	8
	RESULT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RESULT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – RESULT[15:0]: Result Conversion Value**

  These bits will hold up to a 16-bit ADC result, depending on the configuration.

  In single-ended without averaging mode, the ADC conversion will produce a 12-bit result, which can be left- or right-shifted, depending on the setting of CTRLB.LEFTADJ.

  If the result is left-adjusted (CTRLB.LEFTADJ), the high byte of the result will be in bit position [15:8], while the remaining 4 bits of the result will be placed in bit locations [7:4]. This can be used only if an 8-bit result is required; i.e., one can read only the high byte of the entire 16-bit register.

  If the result is not left-adjusted (CTRLB.LEFTADJ) and no oversampling is used, the result will be available in bit locations [11:0], and the result is then 12 bits long.

  If oversampling is used, the result will be located in bit locations [15:0], depending on the settings of the Average Control register (AVGCTRL).

28.8.15 Window Monitor Lower Threshold

Name: WINLT

Offset: 0x1C

Reset: 0x0000

Property: Write-Synchronized, Write-Protected

Bit	15	14	13	12	11	10	9	8
	WINLT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WINLT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – WINLT[15:0]: Window Lower Threshold**
  If the window monitor is enabled, these bits define the lower threshold value.

28.8.16 Window Monitor Upper Threshold

Name: WINUT

Offset: 0x20

Reset: 0x0000

Property: Write-Synchronized, Write-Protected

Bit	15	14	13	12	11	10	9	8
	WINUT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WINUT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – WINUT[15:0]: Window Upper Threshold**
  If the window monitor is enabled, these bits define the upper threshold value.

28.8.17 Gain Correction

Name: GAINCORR

Offset: 0x24

Reset: 0x0000

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
					GAINCORR[11:8]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	GAINCORR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 11:0 – GAINCORR[11:0]: Gain Correction Value**
  If the CTRLB.CORREN bit is one, these bits define how the ADC conversion result is compensated for gain error before being written to the result register. The gain correction is a fractional value, a 1-bit integer plus an 11-bit fraction, and therefore ½ <= GAINCORR < 2. GAINCORR values range from 0.10000000000 to 1.11111111111.

28.8.18 Offset Correction

Name: OFFSETCORR

Offset: 0x26

Reset: 0x0000

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
					OFFSETCORR[11:8]			
Access	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OFFSETCORR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:12 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 11:0 – OFFSETCORR[11:0]: Offset Correction Value**
  If the CTRLB.CORREN bit is one, these bits define how the ADC conversion result is compensated for offset error before being written to the Result register. This OFFSETCORR value is in two's complement format.

28.8.19 Calibration

Name: CALIB

Offset: 0x28

Reset: 0x0000

Property: Write-Protected

Bit	15	14	13	12	11	10	9	8
						BIAS_CAL[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LINEARITY_CAL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:11 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 10:8 – BIAS_CAL[2:0]: Bias Calibration Value**
  This value from production test must be loaded from the NVM software calibration row into the CALIB register by software to achieve the specified accuracy.
  The value must be copied only, and must not be changed.
- **Bits 7:0 – LINEARITY_CAL[7:0]: Linearity Calibration Value**
  This value from production test must be loaded from the NVM software calibration row into the CALIB register by software to achieve the specified accuracy.
  The value must be copied only, and must not be changed.

28.8.20 Debug Control

Name: DBGCTRL

Offset: 0x2A

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
								DBGRUN
Access	R	R	R	R	R	R	R	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:1 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 0 – DBGRUN: Debug Run**
  0: The ADC is halted during debug mode.
  1: The ADC continues normal operation during debug mode.
  This bit can be changed only while the ADC is disabled.
  This bit should be written only while a conversion is not ongoing.

29. AC – Analog Comparators

29.1 Overview

The Analog Comparator (AC) supports two individual comparators. Each comparator (COMP) compares the voltage levels on two inputs, and provides a digital output based on this comparison. Each comparator may be configured to generate interrupt requests and/or peripheral events upon several different combinations of input change.

Hysteresis and propagation delay are two important properties of the comparators; dynamic behavior. Both parameters may be adjusted to achieve the optimal operation for each application.

The input selection includes four shared analog port pins and several internal signals. Each comparator output state can also be output on a pin for use by external devices.

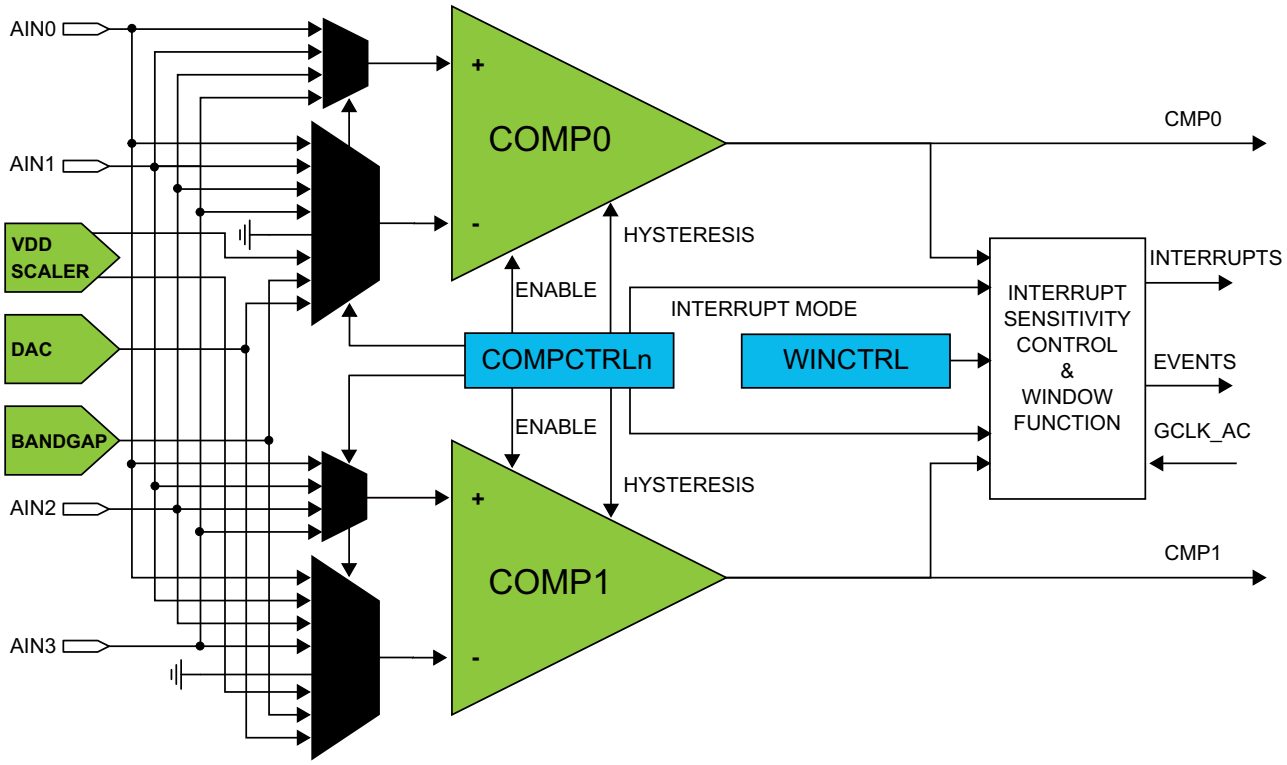
The comparators are always grouped in pairs on each port. The AC module may implement one pair. These are called Comparator 0 (COMP0) and Comparator 1 (COMP1). They have identical behaviors, but separate control registers. The pair can be set in window mode to compare a signal to a voltage range instead of a single voltage level.

29.2 Features

- Two individual comparators
- Selectable propagation delay versus current consumption
- Selectable hysteresis
  - On/Off
- Analog comparator outputs available on pins
  - Asynchronous or synchronous
- Flexible input selection
  - Four pins selectable for positive or negative inputs
  - Ground (for zero crossing)
  - Bandgap reference voltage
  - 64-level programmable VDD scaler per comparator
  - DAC
- Interrupt generation on:
  - Rising or falling edge
  - Toggle
  - End of comparison
- Window function interrupt generation on:
  - Signal above window
  - Signal inside window
  - Signal below window
  - Signal outside window
- Event generation on:
  - Comparator output
  - Window function inside/outside window
- Optional digital filter on comparator output
- Low-power option
  - Single-shot support

29.3 Block Diagram

Figure 29-1. Analog Comparator Block Diagram



29.4 Signal Description

Signal Name	Type	Description
AIN[3..0]	Analog input	Comparator inputs
CMP[1..0]	Digital output	Comparator outputs

Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

29.5 Product Dependencies

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

29.5.1 I/O Lines

Using the AC's I/O lines requires the I/O pins to be configured. Refer to the PORT chapter for details.

Refer to "PORT" on page 276 for details.

29.5.2 Power Management

The AC will continue to operate in any sleep mode where the selected source clock is running. The AC's interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

29.5.3 Clocks

The AC bus clock (CLK_AC_APB) can be enabled and disabled in the Power Manager, and the default state of the CLK_AC_APB can be found in the Peripheral Clock Masking section of “PM – Power Manager” on page 100.

Two generic clocks (GCLK_AC_DIG and GCLK_AC_ANA) are used by the AC. The digital clock (GCLK_AC_DIG) is required to provide the sampling rate for the comparators, while the analog clock (GCLK_AC_ANA) is required for low-voltage operation (VDD < 2.5V) to ensure that the resistance of the analog input multiplexors remains low. These clocks must be configured and enabled in the Generic Clock Controller before using the peripheral.

Refer to “GCLK – Generic Clock Controller” on page 78 for details.

These generic clocks are asynchronous to the CLK_AC_APB clock. Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to “Synchronization” on page 518 for further details.

29.5.4 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the AC interrupts requires the Interrupt Controller to be configured first. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

29.5.5 Events

The events are connected to the Event System. Using the events requires the Event System to be configured first. Refer to “EVSYS – Event System” on page 301 for details.

29.5.6 Debug Operation

When the CPU is halted in debug mode, the peripheral continues normal operation. If the peripheral 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.

29.5.7 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following registers:

- Control B register (CTRLB)
- Interrupt Flag register (INTFLAG)

Write-protection is denoted by the Write-Protected property in the register description.

Write-protection does not apply for accesses through an external debugger. Refer to “PAC – Peripheral Access Controller” on page 27 for details.

29.5.8 Analog Connections

Each comparator has up to four I/O pins that can be used as analog inputs. Each pair of comparators shares the same four pins. These pins must be configured for analog operation before using them as comparator inputs.

Any internal reference source, such as a bandgap reference voltage or the DAC, must be configured and enabled prior to its use as a comparator input.

29.5.9 Other Dependencies

Not applicable.

29.6 Functional Description

29.6.1 Principle of Operation

Each comparator has one positive input and one negative input. Each positive input may be chosen from a selection of analog input pins. Each negative input may be chosen from a selection of analog input pins or internal inputs, such as a

bandgap reference voltage. The digital output from the comparator is one when the difference between the positive and the negative input voltage is positive, and zero otherwise.

The individual comparators can be used independently (normal mode) or grouped in pairs to generate a window comparison (window mode).

29.6.2 Basic Operation

29.6.2.1 Initialization

Before enabling the AC, the input and output events must be configured in the Event Control register (EVCTRL). These settings cannot be changed while the AC is enabled.

Each individual comparator must also be configured by its respective Comparator Control register (COMPCTRLx) before that comparator is enabled. These settings cannot be changed while the comparator is enabled.

- Select the desired measurement mode with COMPCTRLx.SINGLE. See “Starting a Comparison” on page 511 for more details
- Select the desired hysteresis with COMPCTRLx.HYST. See “Input Hysteresis” on page 515 for more details
- Select the comparator speed versus power with COMPCTRLx.SPEED. See “Propagation Delay vs. Power Consumption” on page 515 for more details
- Select the interrupt source with COMPCTRLx.INTSEL
- Select the positive and negative input sources with the COMPCTRLx.MUXPOS and COMPCTRLx.MUXNEG bits. See section “Selecting Comparator Inputs” on page 513 for more details
- Select the filtering option with COMPCTRLx.FLEN

29.6.2.2 Enabling, Disabling and Resetting

The AC is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The individual comparators must be also enabled by writing a one to the Enable bit in the Comparator x Control registers (COMPCTRLx.ENABLE). The AC is disabled by writing a zero to CTRLA.ENABLE. This will also disable the individual comparators, but will not clear their COMPCTRLx.ENABLE bits.

The AC is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the AC, except DEBUG, will be reset to their initial state, and the AC will be disabled. Refer to the CTRLA register for details.

29.6.2.3 Starting a Comparison

Each comparator channel can be in one of two different measurement modes, determined by the Single bit in the Comparator x Control register (COMPCTRLx.SINGLE):

- Continuous measurement
- Single-shot

After being enabled, a start-up delay is required before the result of the comparison is ready. This start-up time is measured automatically to account for environmental changes, such as temperature or voltage supply level, and is specified in “Electrical Characteristics” on page 558.

During the start-up time, the COMP output is not available. If the supply voltage is below 2.5V, the start-up time is also dependent on the voltage doubler. If the supply voltage is guaranteed to be above 2.5V, the voltage doubler can be disabled by writing the Low-Power Mux bit in the Control A register (CTRLA.LPMUX) to one.

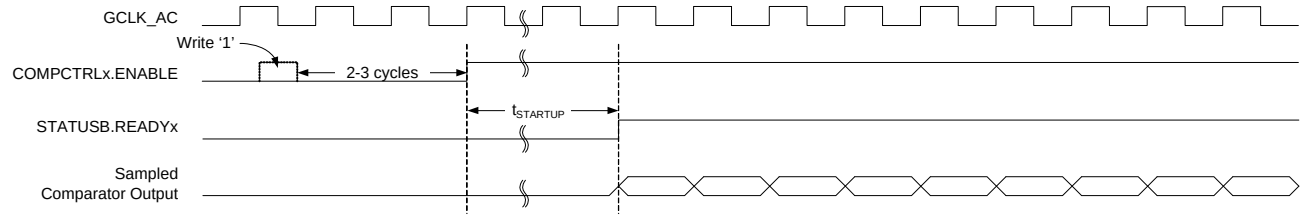
The comparator can be configured to generate interrupts when the output toggles, when the output changes from zero to one (rising edge), when the output changes from one to zero (falling edge) or at the end of the comparison. An end-of-comparison interrupt can be used with the single-shot mode to chain further events in the system, regardless of the state of the comparator outputs. The interrupt mode is set by the Interrupt Selection bit group in the Comparator Control register (COMPCTRLx.INTSEL). Events are generated using the comparator output state, regardless of whether the interrupt is enabled or not.

Continuous Measurement

Continuous measurement is selected by writing COMPCTRLx.SINGLE to zero. In continuous mode, the comparator is continuously enabled and performing comparisons. This ensures that the result of the latest comparison is always available in the Current State bit in the Status A register (STATUSA.STATEx). After the start-up time has passed, a comparison is done and STATUSA is updated. The Comparator x Ready bit in the Status B register (STATUSB.READYx) is set, and the appropriate peripheral events and interrupts are also generated. New comparisons are performed continuously until the COMPCTRLx.ENABLE bit is written to zero. The start-up time applies only to the first comparison.

In continuous operation, edge detection of the comparator output for interrupts is done by comparing the current and previous sample. The sampling rate is the GCLK_AC_DIG frequency. An example of continuous measurement is shown in Figure 29-2.

Figure 29-2. Continuous Measurement Example



For low-power operation, comparisons can be performed during sleep modes without a clock. The comparator is enabled continuously, and changes in the state of the comparator are detected asynchronously. When a toggle occurs, the Power Manager will start GCLK_AC_DIG to register the appropriate peripheral events and interrupts. The GCLK_AC_DIG clock is then disabled again automatically, unless configured to wake up the system from sleep.

Single-Shot

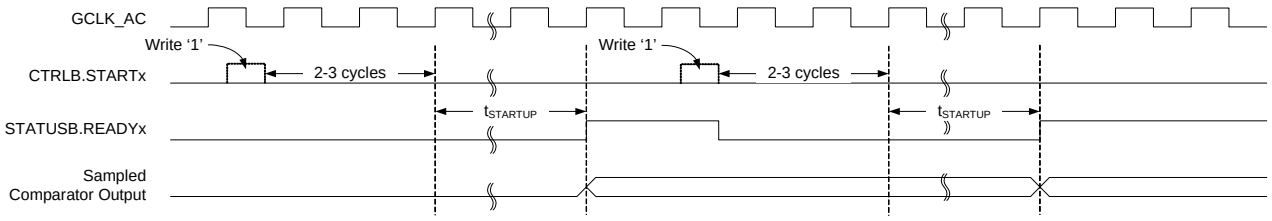
Single-shot operation is selected by writing COMPCTRLx.SINGLE to one. During single-shot operation, the comparator is normally idle. The user starts a single comparison by writing a one to the respective Start Comparison bit in the write-only Control B register (CTRLB.STARTx). The comparator is enabled, and after the start-up time has passed, a single comparison is done and STATUSA is updated. Appropriate peripheral events and interrupts are also generated. No new comparisons will be performed.

Writing a one to CTRLB.STARTx also clears the Comparator x Ready bit in the Status B register (STATUSB.READYx). STATUSB.READYx is set automatically by hardware when the single comparison has completed. To remove the need for polling, an additional means of starting the comparison is also available. A read of the Status C register (STATUSC) will start a comparison on all comparators currently configured for single-shot operation. The read will stall the bus until all enabled comparators are ready. If a comparator is already busy with a comparison, the read will stall until the current comparison is compete, and a new comparison will not be started.

A single-shot measurement can also be triggered by the Event System. Writing a one to the Comparator x Event Input bit in the Event Control Register (EVCTRL.COMPEIx) enables triggering on incoming peripheral events. Each comparator can be triggered independently by separate events. Event-triggered operation is similar to user-triggered operation; the difference is that a peripheral event from another hardware module causes the hardware to automatically start the comparison and clear STATUSB.READYx.

To detect an edge of the comparator output in single-shot operation for the purpose of interrupts, the result of the current measurement is compared with the result of the previous measurement (one sampling period earlier). An example of single-shot operation is shown in Figure 29-3.

Figure 29-3. Single-Shot Example



For low-power operation, event-triggered measurements can be performed during sleep modes. When the event occurs, the Power Manager will start GCLK_AC_DIG. The comparator is enabled, and after the startup time has passed, a comparison is done and appropriate peripheral events and interrupts are also generated. The comparator and GCLK_AC_DIG are then disabled again automatically, unless configured to wake up the system from sleep.

29.6.3 Selecting Comparator Inputs

Each comparator has one positive and one negative input. The positive input is fed from an external input pin (AINx). The negative input can be fed either from an external input pin (AINx) or from one of the several internal reference voltage sources common to all comparators. The user selects the input source as follows:

- The positive input is selected by the Positive Input MUX Select bit group in the Comparator Control register (COMPCTRLx.MUXPOS)
- The negative input is selected by the Negative Input MUX Select bit group in the Comparator Control register (COMPCTRLx.MUXNEG)

In the case of using an external I/O pin, the selected pin must be configured for analog usage in the PORT Controller by disabling the digital input and output. The switching of the analog input multiplexors is controlled to minimize crosstalk between the channels. The input selection must be changed only while the individual comparator is disabled.

29.6.4 Window Operation

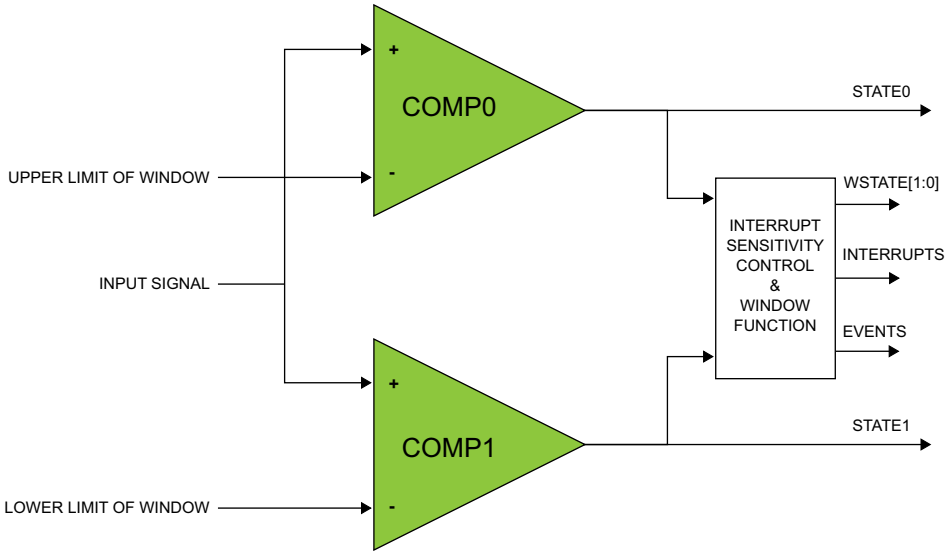
Each comparator pair can be configured to work together in window mode. In this mode, a voltage range is defined, and the comparators give information about whether an input signal is within this range or not. Window mode is enabled by the Window Enable x bit in the Window Control register (WINCTRL.WENx). Both comparators in a pair must have the same measurement mode setting in their respective Comparator Control Registers (COMPCTRLx.SINGLE).

To physically configure the pair of comparators for window mode, the same I/O pin should be chosen for each comparator's positive input to create the shared input signal. The negative inputs define the range for the window. In Figure 29-4, COMP0 defines the upper limit and COMP1 defines the lower limit of the window, as shown but the window will also work in the opposite configuration with COMP0 lower and COMP1 higher. The current state of the window function is available in the Window x State bit group of the Status register (STATUS.WSTATEx).

Window mode can be configured to generate interrupts when the input voltage changes to below the window, when the input voltage changes to above the window, when the input voltage changes into the window or when the input voltage changes outside the window. The interrupt selections are set by the Window Interrupt Selection bit group in the Window Control register (WINCTRL.WINTSELx[1:0]). Events are generated using the inside/outside state of the window, regardless of whether the interrupt is enabled or not. Note that the individual comparator outputs, interrupts and events continue to function normally during window mode.

When the comparators are configured for window mode and single-shot mode, measurements are performed simultaneously on both comparators. Writing a one to either Start Comparison bit in the Control B register (CTRLB.STARTx) starts a measurement. Likewise either peripheral event can start a measurement.

Figure 29-4. Comparators in Window Mode



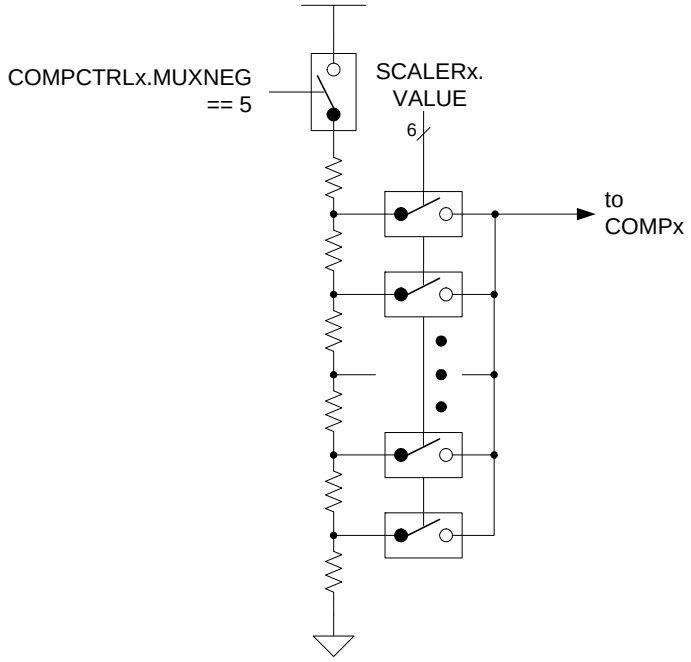
29.6.5 Voltage Doubler

The AC contains a voltage doubler that can reduce the resistance of the analog multiplexors when the supply voltage is below 2.5V. The voltage doubler is normally switched on/off automatically based on the supply level. When enabling the comparators, additional start-up time is required for the voltage doubler to settle. If the supply voltage is guaranteed to be above 2.5V, the voltage doubler can be disabled by writing the Low-Power Mux bit in the Control A register (CTRLA.LPMUX) to one. Disabling the voltage doubler saves power and reduces the start-up time.

29.6.6 VDD Scaler

The VDD scaler generates a reference voltage that is a fraction of the device's supply voltage, with 64 levels. One independent voltage channel is dedicated for each comparator. The scaler is enabled when a comparator's Negative Input Mux bit group in its Comparator Control register (COMPCTRLx.MUXNEG) is set to five and the comparator is enabled. The voltage of each channel is selected by the Value bit group in the Scaler x registers (SCALERx.VALUE[5:0]).

Figure 29-5. VDD Scaler



29.6.7 Input Hysteresis

Application software can selectively enable/disable hysteresis for the comparison. Applying hysteresis will help prevent constant toggling of the output, which can be caused by noise when the input signals are close to each other. Hysteresis is enabled for each comparator individually by the Hysteresis Mode bit in the Comparator x Control register (COMPCTRLx.HYST). Hysteresis is available only in continuous mode (COMPCTRLx.SINGLE=0).

29.6.8 Propagation Delay vs. Power Consumption

It is possible to trade off comparison speed for power efficiency to get the shortest possible propagation delay or the lowest power consumption. The speed setting is configured for each comparator individually by the Speed bit group in the Comparator x Control register (COMPCTRLx.SPEED). The Speed bits select the amount of bias current provided to the comparator, and as such will also affect the start-up time.

29.6.9 Filtering

The output of the comparators can be digitally filtered to reduce noise using a simple digital filter. The filtering is determined by the Filter Length bits in the Comparator Control x register (COMPCTRLx.FLEN), and is independent for each comparator. Filtering is selectable from none, 3-bit majority (N=3) or 5-bit majority (N=5) functions. Any change in the comparator output is considered valid only if N/2+1 out of the last N samples agree. The filter sampling rate is the CLK_AC frequency scaled by the prescaler setting in the Control A register (CTRLA.PRESCALER).

Note that filtering creates an additional delay of N-1 sampling cycles from when a comparison is started until the comparator output is validated. For continuous mode, the first valid output will occur when the required number of filter samples is taken. Subsequent outputs will be generated every cycle based on the current sample plus the previous N-1 samples, as shown in Figure 29-6. For single-shot mode, the comparison completes after the Nth filter sample, as shown in Figure 29-7.

Figure 29-6. Continuous Mode Filtering

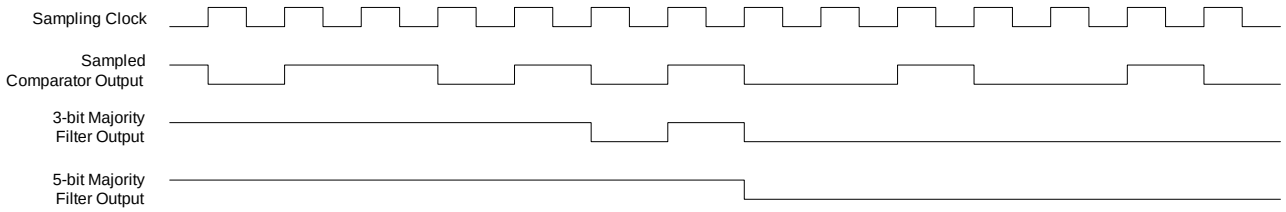
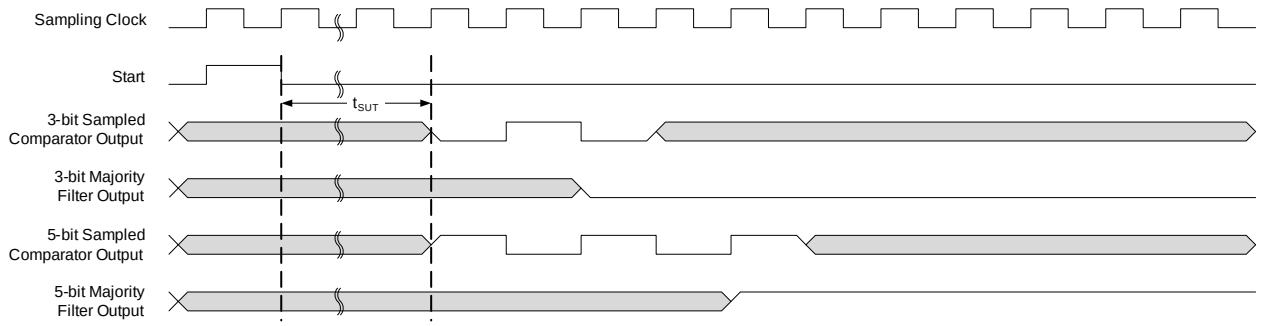


Figure 29-7. Single-Shot Filtering



During sleep modes, filtering is supported only for single-shot measurements. Filtering must be disabled if continuous measurements will be done during sleep modes, or the resulting interrupt/event may be generated incorrectly.

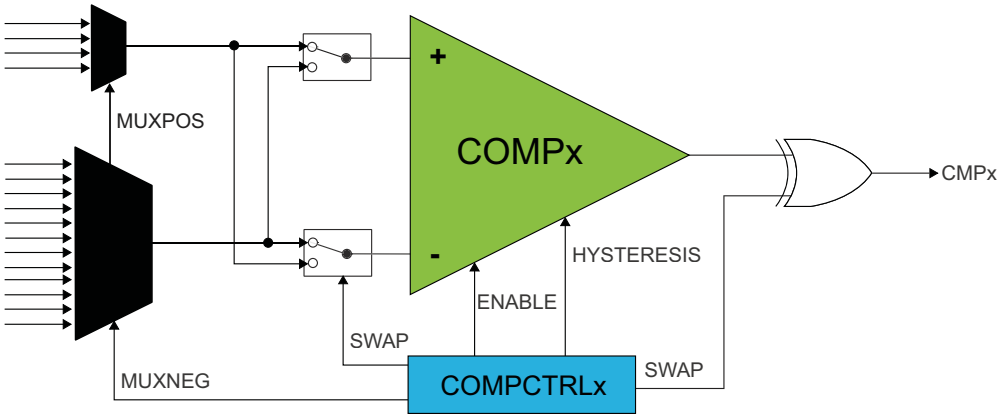
29.6.10 Comparator Output

The output of each comparator can be routed to an I/O pin by setting the Output bit group in the Comparator Control x register (COMPCTRLx.OUT). This allows the comparator to be used by external circuitry. Either the raw, non-synchronized output of the comparator or the CLK_AC-synchronized version, including filtering, can be used as the I/O signal source. The output appears on the corresponding CMP[x] pin.

29.6.11 Offset Compensation

The Swap bit in the Comparator Control registers (COMPCTRLx.SWAP) controls switching of the input signals to a comparator's positive and negative terminals. When the comparator terminals are swapped, the output signal from the comparator is also inverted, as shown in Figure 29-8. This allows the user to measure or compensate for the comparator input offset voltage. As part of the input selection, COMPCTRLx.SWAP can be changed only while the comparator is disabled.

Figure 29-8. Input Swapping for Offset Compensation



29.7 Additional Features

29.7.1 Interrupts

The peripheral has the following interrupt sources:

- Comparator: COMP0, COMP1(INTENCLR, INTSET, INTFLAG)
- Window: WIN0(INTENCLR, INTSET, INTFLAG)

Comparator interrupts are generated based on the conditions selected by the Interrupt Selection bit group in the Comparator Control registers (COMPCTRLx.INTSEL). Window interrupts are generated based on the conditions selected by the Window Interrupt Selection bit group in the Window Control register (WINCTRL.WINTSEL[1:0]).

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the peripheral is reset. An interrupt flag is cleared by writing a one to the corresponding bit in the INTFLAG register.

Each peripheral can have one interrupt request line per interrupt source or one common interrupt request line for all the interrupt sources. If the peripheral has one common interrupt request line for all the interrupt sources, the user must read the INTFLAG register to determine which interrupt condition is present.

For details on clearing interrupt flags, refer to the INTFLAG register description.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

29.7.2 Events

The peripheral can generate the following output events:

- Comparator: COMPEO0, COMPEO1(EVCTRL)
- Window: WINEO0(EVCTRL)

Output events must be enabled to be generated. Writing a one to an Event Output bit in the Event Control register (EVCTRL.COMPEOx) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. The events must be correctly routed in the Event System. Refer to “EVSYS – Event System” on page 301 for details.

The peripheral can take the following actions on an input event:

- Single-shot measurement
- Single-shot measurement in window mode

Input events must be enabled for the corresponding action to be taken on any input event. Writing a one to an Event Input bit in the Event Control register (EVCTRL.COMPEIx) enables the corresponding action on input event. Writing a zero to a bit disables the corresponding action on input event. Note that if several events are connected to the peripheral, the enabled action will be taken on any of the incoming events. The events must be correctly routed in the Event System. Refer to “EVSYS – Event System” on page 301 for details.

When EVCTRL.COMPEIx is one, the event will start a comparison on COMPx after the start-up time delay. In normal mode, each comparator responds to its corresponding input event independently. For a pair of comparators in window mode, either comparator event will trigger a comparison on both comparators simultaneously.

29.7.3 Sleep Mode Operation

The Run in Standby bit in the Control A register (CTRLA.RUNSTDBY) controls the behavior of the AC during standby sleep mode. When the bit is zero, the comparator pair is disabled during sleep, but maintains its current configuration. When the bit is one, the comparator pair continues to operate during sleep. Note that when RUNSTDBY is zero, the

analog blocks are powered off for the lowest power consumption. This necessitates a start-up time delay when the system returns from sleep.

When RUNSTDBY is one, any enabled AC interrupt source can wake up the CPU. However, single-shot comparisons will be triggerable by events only while the CPU is idle. The AC can also be used during sleep modes where the clock used by the AC is disabled, provided that the AC is still powered (not in shutdown). In this case, the behavior is slightly different and depends on the measurement mode, as listed in Table 29-1.

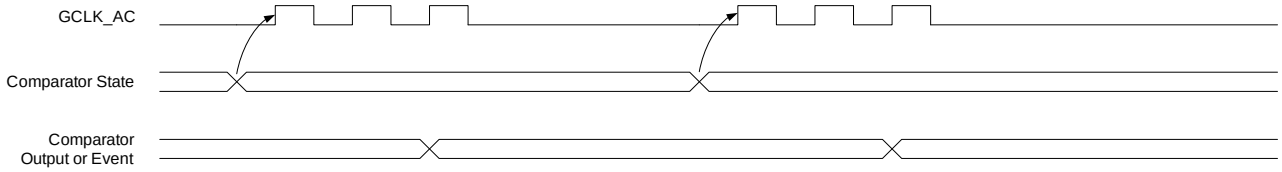
Table 29-1. Sleep Mode Operation

COMPCTRLx.MODE	RUNSTDBY=0	RUNSTDBY=1
0 (Continuous)	COMPx disabled	GCLK_AC_DIG stopped, COMPx enabled
1 (Single-shot)	COMPx disabled	GCLK_AC_DIG stopped, COMPx enabled only when triggered by an input event

29.7.3.1 Continuous Measurement during Sleep

When a comparator is enabled in continuous measurement mode and GCLK_AC_DIG is disabled during sleep, the comparator will remain continuously enabled and will function asynchronously. The current state of the comparator is asynchronously monitored for changes. If an edge matching the interrupt condition is found, GCLK_AC_DIG is started to register the interrupt condition and generate events. If the interrupt is enabled in the Interrupt Enable registers (INTENCLR/SET), the AC can wake up the device; otherwise GCLK_AC_DIG is disabled until the next edge detection. Filtering is not possible with this configuration.

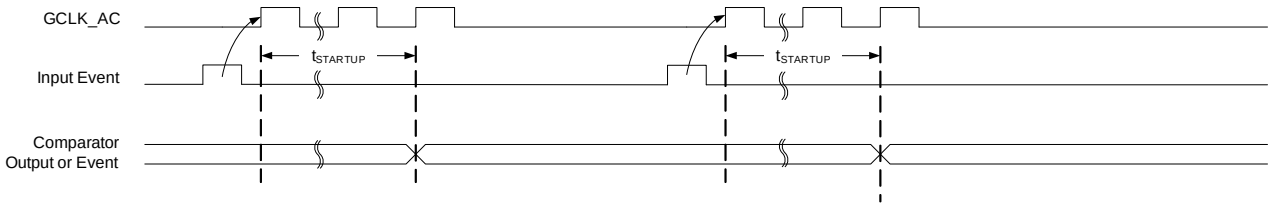
Figure 29-9. Continuous Mode SleepWalking



29.7.3.2 Single-Shot Measurement during Sleep

For low-power operation, event-triggered measurements can be performed during sleep modes. When the event occurs, the Power Manager will start GCLK_AC_DIG. The comparator is enabled, and after the start-up time has passed, a comparison is done, with filtering if desired, and the appropriate peripheral events and interrupts are also generated, as shown in Figure 29-10 The comparator and GCLK_AC_DIG are then disabled again automatically, unless configured to wake the system from sleep. Filtering is allowed with this configuration.

Figure 29-10.Single-Shot SleepWalking



29.7.4 Synchronization

Due to the asynchronicity between CLK_MODULE_APB and GCLK_MODULE, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in Control A register (CTRLA.SWRST)
- Enable bit in Control A register (CTRLA.ENABLE)
- Enable bit in Comparator Control register (COMPCTRLn.ENABLE)

The following register need synchronization when written:

- Window Control register (WINCTRL)

Refer to the Synchronization chapter for further details.

29.8 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0						RUNSTDBY	ENABLE	SWRST
0x01	CTRLB	7:0							START1	START0
0x02	EVCTRL	7:0				WINEO0			COMPEO1	COMPEO0
0x03		15:8							COMPEI1	COMPEI0
0x04	INTENCLR	7:0				WIN0			COMP1	COMP0
0x05	INTENSET	7:0				WIN0			COMP1	COMP0
0x06	INTFLAG	7:0				WIN0			COMP1	COMP0
0x07	Reserved									
0x08	STATUSA	7:0			WSTATE0[1:0]				STATE1	STATE0
0x09	STATUSB	7:0	SYNCBUSY						READY1	READY0
0x0A	STATUSC	7:0			WSTATE0[1:0]				STATE1	STATE0
0x0B	Reserved									
0x0C	WINCTRL	7:0						WINTSEL0[1:0]		WEN0
0x0D	Reserved									
0x0E	Reserved									
0x0F	Reserved									
0x10	COMPCTRL0	7:0		INTSEL[1:0]			SPEED[1:0]		SINGLE	ENABLE
0x11		15:8	SWAP		MUXPOS[1:0]			MUXNEG[2:0]		
0x12		23:16					HYST		OUT[1:0]	
0x13		31:24						FLEN[2:0]		
0x14	COMPCTRL1	7:0		INTSEL[1:0]			SPEED[1:0]		SINGLE	ENABLE
0x15		15:8	SWAP		MUXPOS[1:0]			MUXNEG[2:0]		
0x16		23:16					HYST		OUT[1:0]	
0x17		31:24						FLEN[2:0]		
0x18	Reserved									
...	Reserved									
0x1F	Reserved									
0x20	SCALER0	7:0	VALUE[5:0]							
0x21	SCALER1	7:0	VALUE[5:0]							

29.9 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 510 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Write-Synchronized or the Read-Synchronized property in each individual register description. Refer to “Synchronization” on page 518 for details.

Some registers are enable-protected, meaning they can be written only when the AC is disabled. Enable-protection is denoted by the Enable-Protected property in each individual register description.

29.9.1 Control A

Name: CTRLA

Offset: 0x00

Reset: 0x00

Property: Write-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
						RUNSTDBY	ENABLE	SWRST
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RUNSTDBY: Run in Standby**
  This bit controls the behavior of the comparators during standby sleep mode.
  0: The comparator pair is disabled during sleep.
  1: The comparator pair continues to operate during sleep.
  This bit is not synchronized

- **Bit 1 – ENABLE: Enable**
  0: The AC is disabled.
  1: The AC is enabled. Each comparator must also be enabled individually by the Enable bit in the Comparator Control register (COMPCTRLn.ENABLE).
  Due to synchronization, there is delay from updating the register until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately after being written. STATUS.SYNCBUSY is set. STATUS.SYNCBUSY is cleared when the peripheral is enabled/disabled.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets all registers in the AC to their initial state, and the AC will be disabled.

Writing a one to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

29.9.2 Control B

Name: CTRLB

Offset: 0x01

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
							START1	**START0**
Access	R	R	R	R	R	R	W	W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – STARTx: Comparator x Start Comparison**
  Writing a zero to this field has no effect.
  Writing a one to STARTx starts a single-shot comparison on COMPx if both the Single-Shot and Enable bits in the Comparator x Control Register are one (COMPCTRLx.SINGLE and COMPCTRLx.ENABLE). If comparator x is not implemented, or if it is not enabled in single-shot mode, writing a one has no effect.
  This bit always reads as zero.

29.9.3 Event Control

Name: EVCTRL

Offset: 0x02

Reset: 0x0000

Property: Write-Protected, Enable-Protected

Bit	15	14	13	12	11	10	9	8
							COMPEI1	COMPEI0
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				WINEO0			COMPEO1	COMPEO0
Access	R	R	R	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:10 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 9:8 – COMPEIx: Comparator x Event Input**
  Note that several actions can be enabled for incoming events. If several events are connected to the peripheral, the enabled action will be taken for any of the incoming events. There is no way to tell which of the incoming events caused the action.

  These bits indicate whether a comparison will start or not on any incoming event.

  0: Comparison will not start on any incoming event.

  1: Comparison will start on any incoming event.

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – WINEO0: Window 0 Event Output Enable**
  This bit indicates whether the window 0 function can generate a peripheral event or not.

  0: Window 0 event is disabled.

  1: Window 0 event is enabled.

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – COMPEOx: Comparator x Event Output Enable**
  These bits indicate whether the comparator x output can generate a peripheral event or not.

  0: COMPx event generation is disabled.

  1: COMPx event generation is enabled.

29.9.4 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x04

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				WIN0			**COMP1**	**COMP0**
Access	R	R	R	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – WIN0: Window 0 Interrupt Enable**
  Reading this bit returns the state of the Window 0 interrupt enable.
  0: The Window 0 interrupt is disabled.
  1: The Window 0 interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the Window 0 interrupt.

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – COMPx: Comparator x Interrupt Enable**
  Reading this bit returns the state of the Comparator x interrupt enable.
  0: The Comparator x interrupt is disabled.
  1: The Comparator x interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit disables the Comparator x interrupt.

29.9.5 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x05

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
				WIN0			COMP1	COMP0
Access	R	R	R	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – WIN0: Window 0 Interrupt Enable**
  Reading this bit returns the state of the Window 0 interrupt enable.
  0: The Window 0 interrupt is disabled.
  1: The Window 0 interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit enables the Window 0 interrupt.

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – COMPx: Comparator x Interrupt Enable**
  Reading this bit returns the state of the Comparator x interrupt enable.
  0: The Comparator x interrupt is disabled.
  1: The Comparator x interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Ready interrupt bit and enable the Ready interrupt.

29.9.6 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x06

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
				WIN0			COMP1	COMP0
Access	R	R	R	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 4 – WIN0: Window 0**
  This flag is set according to the Window 0 Interrupt Selection bit group in the WINCTRL register (WINCTRL.WINTSEL0) and will generate an interrupt if INTENCLR/SET.WIN0 is also one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Window 0 interrupt flag.

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – COMPx: Comparator x**
  Reading this bit returns the status of the Comparator x interrupt flag. If comparator x is not implemented, COMPx always reads as zero.
  This flag is set according to the Interrupt Selection bit group in the Comparator x Control register (COMPCTRLx.INTSEL) and will generate an interrupt if INTENCLR/SET.COMPx is also one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit clears the Comparator x interrupt flag.

29.9.7 Status A

Name: STATUSA

Offset: 0x08

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
			WSTATE0[1:0]				STATE1	STATE0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – WSTATE0[1:0]: Window 0 Current State**
  These bits show the current state of the signal if the window 0 mode is enabled, according to Table 29-2. If the window 0 function is not implemented, WSTATE0 always reads as zero.

Table 29-2. Window Mode Current State

Value	Name	Description
0x0	ABOVE	Signal is above window
0x1	INSIDE	Signal is inside window
0x2	BELOW	Signal is below window
0x3	–	Reserved

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – STATEx: Comparator x Current State**
  This bit shows the current state of the output signal from COMPx. STATEx is valid only when STATUSB.READYx is one.

29.9.8 Status B

Name: STATUSB

Offset: 0x09

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY						READY1	READY0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.
- **Bits 6:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bits 1:0 – READYx: Comparator x Ready**
  This bit is cleared when the comparator x output is not ready.
  This bit is set when the comparator x output is ready.

29.9.9 Status C

STATUSC is a copy of STATUSA (see STATUSA register), with the additional feature of automatically starting single-shot comparisons. A read of STATUSC will start a comparison on all comparators currently configured for single-shot operation. The read will stall the bus until all enabled comparators are ready. If a comparator is already busy with a comparison, the read will stall until the current comparison is compete, and a new comparison will not be started.

Name: STATUSC

Offset: 0x0A

Reset: 0x00

Property: –

Bit	7	6	5	4	3	2	1	0
			WSTATE0[1:0]				STATE1	STATE0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:4 – WSTATE0[1:0]: Window 0 Current State**
  These bits show the current state of the signal if the window 0 mode is enabled. If the window 0 function is not implemented, WSTATE0 always reads as zero.

Table 29-3. Window Mode Current State

Value	Name	Description
0x0	ABOVE	Signal is above window
0x1	INSIDE	Signal is inside window
0x2	BELOW	Signal is below window
0x3	–	Reserved

- **Bits 3:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 1:0 – STATEx: Comparator x Current State**
  This bit shows the current state of the output signal from COMPx. If comparator x is not implemented, STATEx always reads as zero. STATEx is only valid when STATUSB.READYx is one.

29.9.10 Window Control

Name: WINCTRL

Offset: 0x0C

Reset: 0x00

Property: Write-Synchronized, Write-Protected

Bit	7	6	5	4	3	2	1	0
						WINTSEL0[1:0]		**WEN0**
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 2:1 – WINTSEL0: Window 0 Interrupt Selection**
  These bits configure the interrupt mode for the comparator window 0 mode.

Table 29-4. Window 0 Interrupt Selection

Value	Name	Description
0x0	ABOVE	Interrupt on signal above window
0x1	INSIDE	Interrupt on signal inside window
0x2	BELOW	Interrupt on signal below window
0x3	OUTSIDE	Interrupt on signal outside window

- **Bit 0 – WEN0: Window 0 Mode Enable**
  0: Window mode is disabled for comparators 0 and 1.
  1: Window mode is enabled for comparators 0 and 1.

29.9.11 Scaler n

Name: SCALERn

Offset: 0x20+n*0x1 [n=0..1]

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
			VALUE[5:0]					
Access	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 5:0 – VALUE[5:0]: Scaler Value**
  These bits define the scaling factor for channel n of the V_{DD} voltage scaler. The output voltage, V_{SCALE}, is:

$$V_{SCALE} = \frac{V_{DD} \cdot (VALUE + 1)}{64}$$

29.9.12 Comparator Control n

The configuration of comparator n is protected while comparator n is enabled (COMPCTRLn.ENABLE = 1). Changes to the other bits in COMPCTRLn can only occur when COMPCTRLn.ENABLE is zero.

Name: COMPCTRLn

Offset: 0x10+n*0x4 [n=0..1]

Reset: 0x00000000

Property: Write-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
						FLEN[2:0]		
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
					HYST		OUT[1:0]	
Access	R	R	R	R	R/W	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SWAP		MUXPOS[1:0]			MUXNEG[2:0]		
Access	R/W	R	R/W	R/W	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		INTSEL[1:0]			SPEED[1:0]		SINGLE	ENABLE
Access	R	R/W	R/W	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 31:27 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bits 26:24 – FLEN[2:0]: Filter Length**
  These bits configure the filtering for comparator n. COMPCTRLn.FLEN can only be written while COMPCTRLn.ENABLE is zero.

  These bits are not synchronized.

Table 29-5. Filter Length

Value	Name	Description
0x0	OFF	No filtering
0x1	MAJ3	3-bit majority function (2 of 3)
0x2	MAJ5	5-bit majority function (3 of 5)
0x3-0x7	N/A	Reserved

- **Bits 23:20 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.
- **Bit 19 – HYST: Hysteresis Enable**
  This bit indicates the hysteresis mode of comparator n. Hysteresis is available only for continuous mode (COMPCTRLn.SINGLE=0). COMPCTRLn.HYST can be written only while COMPCTRLn.ENABLE is zero.
  0: Hysteresis is disabled.
  1: Hysteresis is enabled.
  These bits are not synchronized.
- **Bit 18 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.
- **Bits 17:16 – OUT[1:0]: Output**
  These bits configure the output selection for comparator n. COMPCTRLn.OUT can be written only while COMPCTRLn.ENABLE is zero.
  These bits are not synchronized.

Table 29-6. Output Selection

Value	Name	Description
0x0	OFF	The output of COMPn is not routed to the COMPn I/O port
0x1	ASYNC	The asynchronous output of COMPn is routed to the COMPn I/O port
0x2	SYNC	The synchronous output (including filtering) of COMPn is routed to the COMPn I/O port
0x3	N/A	Reserved

- **Bit 15 – SWAP: Swap Inputs and Invert**
  This bit swaps the positive and negative inputs to COMPn and inverts the output. This function can be used for offset cancellation. COMPCTRLn.SWAP can be written only while COMPCTRLn.ENABLE is zero.
  0: The output of MUXPOS connects to the positive input, and the output of MUXNEG connects to the negative input.
  1: The output of MUXNEG connects to the positive input, and the output of MUXPOS connects to the negative input.
  These bits are not synchronized.

- **Bit 14 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 13:12 – MUXPOS[1:0]: Positive Input Mux Selection**
  These bits select which input will be connected to the positive input of comparator n. COMPCTRLn.MUXPOS can be written only while COMPCTRLn.ENABLE is zero.
  These bits are not synchronized.

Table 29-7. Positive Input Mux Selection

Value	Name	Description
0x0	PIN0	I/O pin 0
0x1	PIN1	I/O pin 1
0x2	PIN2	I/O pin 2
0x3	PIN3	I/O pin 3

- **Bit 11 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 10:8 – MUXNEG[2:0]: Negative Input Mux Selection**
  These bits select which input will be connected to the negative input of comparator n. COMPCTRLn.MUXNEG can only be written while COMPCTRLn.ENABLE is zero.
  These bits are not synchronized.

Table 29-8. Negative Input Mux Selection

Value	Name	Description
0x0	PIN0	I/O pin 0
0x1	PIN1	I/O pin 1
0x2	PIN2	I/O pin 2
0x3	PIN3	I/O pin 3
0x4	GND	Ground
0x5	VSCALE	VDD scaler
0x6	BANDGAP	Internal bandgap voltage
0x7	DAC	DAC output

- **Bit 7 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 6:5 – INTSEL[1:0]: Interrupt Selection**
  These bits select the condition for comparator n to generate an interrupt or event. COMPCTRLn.INTSEL can be written only while COMPCTRLn.ENABLE is zero.
  These bits are not synchronized.

Table 29-9. Interrupt Selection

Value	Name	Description
0x0	TOGGLE	Interrupt on comparator output toggle
0x1	RISING	Interrupt on comparator output rising
0x2	FALLING	Interrupt on comparator output falling
0x3	EOC	Interrupt on end of comparison (single-shot mode only)

- **Bit 4 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written. This bit will always return zero when read.

- **Bits 3:2 – SPEED[1:0]: Speed Selection**
  This bit indicates the speed/propagation delay mode of comparator n. COMPCTRLn.SPEED can be written only while COMPCTRLn.ENABLE is zero.
  These bits are not synchronized.

Table 29-10. Speed Selection

Value	Name	Description
0x0	LOW	Low speed
0x1	HIGH	High speed
0x2-0x3	Reserved	

- **Bit 1 – SINGLE: Single-Shot Mode**
  This bit determines the operation of comparator n. COMPCTRLn.SINGLE can be written only while COMPCTRLn.ENABLE is zero.
  0: Comparator n operates in continuous measurement mode.
  1: Comparator n operates in single-shot mode.
  These bits are not synchronized.

- **Bit 0 – ENABLE: Enable**
  Writing a zero to this bit disables comparator n.
  Writing a one to this bit enables comparator n.
  After writing to this bit, the value read back will not change until the action initiated by the writing is complete. Due to synchronization, there is a latency of at least two GCLK_AC_DIG clock cycles from updating the register until the comparator is enabled/disabled. The bit will continue to read the previous state while the change is in progress.
  Writing a one to COMPCTRLn.ENABLE will prevent further changes to the other bits in COMPCTRLn. These bits remain protected until COMPCTRLn.ENABLE is written to zero and the write is synchronized.

30. DAC – Digital-to-Analog Converter

30.1 Overview

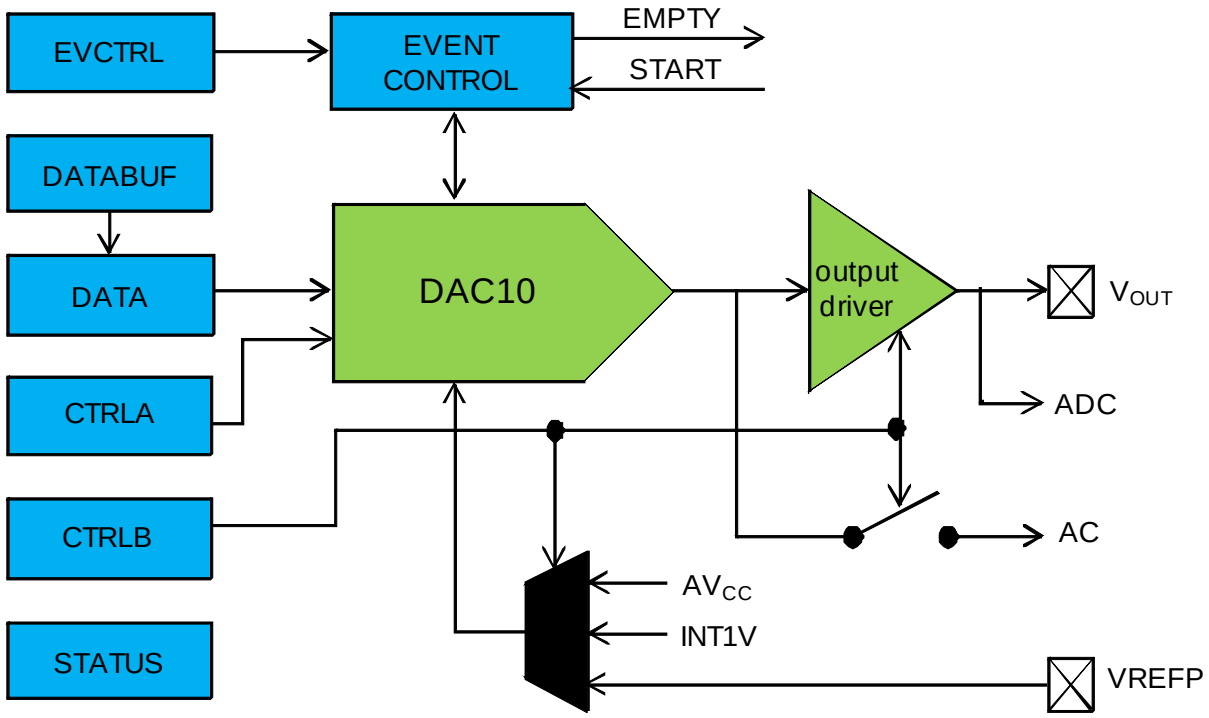
The Digital-to-Analog Converter (DAC) converts a digital value to a voltage. The DAC has one channel with 10-bit resolution, and it is capable of converting up to 350,000 samples per second (350ksps).

30.2 Features

- DAC with 10-bit resolution
- Up to 350ksps conversion rate
- Multiple trigger sources
- High-drive capabilities
- Output can be used as input to the Analog Comparator (AC)

30.3 Block Diagram

Figure 30-1. DAC Block Diagram



30.4 Signal Description

Signal Name	Type	Description
VOUT	Analog output	DAC output
VREFP	Analog input	External reference

Refer to "I/O Multiplexing and Considerations" on page 11 for the pin mapping of this peripheral. One signal can be mapped on several pins.

30.5 Product Dependencies

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

30.5.1 I/O Lines

Using the DAC's I/O lines requires the I/O pins to be configured using the port configuration (PORT).

Refer to "PORT" on page 276 for details.

30.5.2 Power Management

The DAC will continue to operate in any sleep mode where the selected source clock is running. The DAC interrupts can be used to wake up the device from sleep modes. The events can trigger other operations in the system without exiting sleep modes. Refer to "PM – Power Manager" on page 100 for details on the different sleep modes.

30.5.3 Clocks

The DAC bus clock (CLK_DAC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_DAC_APB can be found in the Peripheral Clock Masking section in "PM – Power Manager" on page 100.

A generic clock (GCLK_DAC) is required to clock the DAC. This clock must be configured and enabled in the Generic Clock Controller before using the DAC. Refer to "GCLK – Generic Clock Controller" on page 78 for details.

This generic clock is asynchronous to the bus clock (CLK_DAC). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains. Refer to "Synchronization" on page 541 for further details.

30.5.4 DMA

Not applicable.

30.5.5 Interrupts

The interrupt request line is connected to the Interrupt Controller. Using the DAC interrupts requires the Interrupt Controller to be configured first. Refer to "Nested Vector Interrupt Controller" on page 24 for details.

30.5.6 Events

The events are connected to the Event System. Refer to "EVSYS – Event System" on page 301 for details on how to configure the Event System.

30.5.7 Debug Operation

When the CPU is halted in debug mode the DAC continues normal operation. If the DAC 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.

30.5.8 Register Access Protection

All registers with write-access are optionally write-protected by the Peripheral Access Controller (PAC), except the following register:

- Interrupt Flag Status and Clear register (INTFLAG)

Write-protection is denoted by the Write-Protection property in the register description.

When the CPU is halted in debug mode, all write-protection is automatically disabled.

Write-protection does not apply for accesses through an external debugger. Refer to "PAC – Peripheral Access Controller" on page 27 for details.

30.5.9 Analog Connections

Not applicable.

30.6 Functional Description

30.6.1 Principle of Operation

The Digital-to-Analog Converter (DAC) converts the digital value written to the Data register (DATA) into an analog voltage on the DAC output. By default, a conversion is started when new data is written to DATA, and the corresponding voltage is available on the DAC output after the conversion time. It is also possible to enable events from the Event System to trigger the conversion.

30.6.2 Basic Operation

30.6.2.1 Initialization

Before enabling the DAC, it must be configured by selecting the voltage reference using the Reference Selection bits in the Control B register (CTRLB.REFSEL).

30.6.2.2 Enabling, Disabling and Resetting

The DAC is enabled by writing a one to the Enable bit in the Control A register (CTRLA.ENABLE). The DAC is disabled by writing a zero to CTRLA.ENABLE.

The DAC is reset by writing a one to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the DAC will be reset to their initial state, and the DAC will be disabled. Refer to the CTRLA register for details.

30.6.2.3 Enabling the Output Buffer

To enable the DAC output on the V_{OUT} pin, the output driver must be enabled by writing a one to the External Output Enable bit in the Control B register (CTRLB.EOEN).

The DAC output buffer provides a high-drive-strength output, and is capable of driving both resistive and capacitive loads. To minimize power consumption, the output buffer should be enabled only when external output is needed.

30.6.3 Additional Features

30.6.3.1 Conversion Range

The conversion range is between GND and the selected DAC voltage reference. The default voltage reference is the internal 1V (INT1V) reference voltage. The other voltage reference options are the 3.3V analog supply voltage (AV_{CC} = VDDANA) and the external voltage reference (VREFP). The voltage reference is selected by writing to the Reference Selection bits in the Control B register (CTRLB.REFSEL). The output voltage from the DAC can be calculated using the following formula:

$$V_{DAC} = \frac{DATA}{0x3FF} \cdot VREF$$

30.6.3.2 DAC as an Internal Reference

The DAC output can be internally enabled as input to the analog comparator. This is enabled by writing a one to the Internal Output Enable bit in the Control B register (CTRLB.IOEN). It is possible to have the internal and external output enabled simultaneously.

The DAC output can also be enabled as input to the Analog-to-Digital Converter. In this case, the output buffer must be enabled.

30.6.3.3 Data Buffer

The Data Buffer register (DATABUF) and the Data register (DATA) are linked together to form a two-stage FIFO. The DAC uses the Start Conversion event to load data from DATABUF into DATA and start a new conversion. The Start Conversion event is enabled by writing a one to the Start Event Input bit in the Event Control register (EVCTRL.STARTEI). If a Start Conversion event occurs when DATABUF is empty, an Underrun interrupt request is generated if the Underrun interrupt is enabled.

The DAC can generate a Data Buffer Empty event when DATABUF becomes empty and new data can be loaded to the buffer. The Data Buffer Empty event is enabled by writing a one to the Empty Event Output bit in the Event Control register (EVCTRL.EMPTYEO). A Data Buffer Empty interrupt request is generated if the Data Buffer Empty interrupt is enabled.

30.6.3.4 Voltage Pump

When the DAC is used at operating voltages lower than 2.5V, the voltage pump must be enabled. This enabling is done automatically, depending on operating voltage.

The voltage pump can be disabled by writing a one to the Voltage Pump Disable bit in the Control B register (CTRLB.VPD). This can be used to reduce power consumption when the operating voltage is above 2.5V.

The voltage pump uses the asynchronous GCLK_DAC clock, and requires that the clock frequency be at least four times higher than the sampling period.

30.6.3.5 Sampling Period

As there is no automatic indication that a conversion is done, the sampling period must be greater than or equal to the specified conversion time.

30.6.4 DMA Operation

Not applicable.

30.6.5 Interrupts

The DAC has the following interrupt sources:

- Data Buffer Empty
- Underrun
- Synchronization Ready

Each interrupt source has an interrupt flag associated with it. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a one to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing a one to the corresponding bit in the Interrupt Enable Clear register (INTENCLR). An interrupt request is generated when the interrupt flag is set and the corresponding interrupt is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled or the DAC is reset. See the register description for details on how to clear interrupt flags.

The DAC has one common interrupt request line for all the interrupt sources. The user must read the INTFLAG register to determine which interrupt condition is present.

Note that interrupts must be globally enabled for interrupt requests to be generated. Refer to “Nested Vector Interrupt Controller” on page 24 for details.

30.6.6 Events

The DAC can generate the following output events:

- Data Buffer Empty (EMPTY)

Writing a one to an Event Output bit in the Event Control register (EVCTRL.xxEO) enables the corresponding output event. Writing a zero to this bit disables the corresponding output event. Refer to “EVSYS – Event System” on page 301 for details on configuring the event system.

The DAC can take the following actions on an input event:

- Start Conversion (START)

Writing a one to an Event Input bit in the Event Control register (EVCTRL.xxEI) enables the corresponding action on an input event. Writing a zero to this bit disables the corresponding action on input event. Note that if several events are connected to the DAC, the enabled action will be taken on any of the incoming events. Refer to "EVSYS – Event System" on page 301 for details on configuring the event system.

30.6.7 Sleep Mode Operation

The generic clock for the DAC is running in idle sleep mode. If the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY) is one, the DAC output buffer will keep its value in standby sleep mode. If CTRLA.RUNSTDBY is zero, the DAC output buffer will be disabled in standby sleep mode.

30.6.8 Synchronization

Due to the asynchronicity between CLK_DAC_APB and GCLK_DAC, some registers must be synchronized when accessed. A register can require:

- Synchronization when written
- Synchronization when read
- Synchronization when written and read
- No synchronization

When executing an operation that requires synchronization, the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set immediately, and cleared when synchronization is complete. The Synchronization Ready interrupt can be used to signal when synchronization is complete.

If an operation that requires synchronization is executed while STATUS.SYNCBUSY is one, the bus will be stalled. All operations will complete successfully, but the CPU will be stalled and interrupts will be pending as long as the bus is stalled.

The following bits need synchronization when written:

- Software Reset bit in the Control A register (CTRLA.SWRST)
- Enable bit in the Control A register (CTRLA.ENABLE)
- All bits in the Data register (DATA)
- All bits in the Data Buffer register (DATABUF)

Synchronization is denoted by the Write-Synchronized property in the register description.

The following bits need synchronization when read:

- All bits in the Data register (DATA)

30.7 Register Summary

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0						RUNSTDBY	ENABLE	SWRST
0x01	CTRLB	7:0	REFSEL[1:0]				VPD	LEFTADJ	IOEN	EOEN
0x02	EVCTRL	7:0							EMPTYEO	STARTEI
0x03	Reserved									
0x04	INTENCLR	7:0						SYNCRDY	EMPTY	UNDERRUN
0x05	INTENSET	7:0						SYNCRDY	EMPTY	UNDERRUN
0x06	INTFLAG	7:0						SYNCRDY	EMPTY	UNDERRUN
0x07	STATUS	7:0	SYNCBUSY							
0x08	DATA	7:0	DATA[7:0]							
0x09		15:8	DATA[15:8]							
0x0A	Reserved									
0x0B	Reserved									
0x0C	DATABUF	7:0	DATABUF[7:0]							
0x0D		15:8	DATABUF[15:8]							
0x0E	Reserved									
0x0F	Reserved									

30.8 Register Description

Registers can be 8, 16 or 32 bits wide. Atomic 8-, 16- and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers are optionally write-protected by the Peripheral Access Controller (PAC). Write-protection is denoted by the Write-Protected property in each individual register description. Refer to “Register Access Protection” on page 538 for details.

Some registers require synchronization when read and/or written. Synchronization is denoted by the Synchronized property in each individual register description. Refer to “Synchronization” on page 541 for details.

30.8.1 Control A

Name: CTRLA

Offset: 0x0

Reset: 0x00

Property: Write-Protected, Write-Synchronized

Bit	7	6	5	4	3	2	1	0
						RUNSTDBY	ENABLE	SWRST
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – RUNSTDBY: Run in Standby**
  0: The DAC output buffer is disabled in standby sleep mode.
  1: The DAC output buffer can be enabled in standby sleep mode.
  This bit is not synchronized.

- **Bit 1 – ENABLE: Enable**
  0: The peripheral is disabled or being disabled.
  1: The peripheral is enabled or being enabled.
  Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRL.ENABLE will read back immediately and the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY) will be set. STATUS.SYNCBUSY is cleared when the operation is complete.

- **Bit 0 – SWRST: Software Reset**
  0: There is no reset operation ongoing.
  1: The reset operation is ongoing.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets the all registers in the DAC to their initial state, and the DAC will be disabled.
  Writing a one to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write operation will be discarded.
  Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and STATUS.SYNCBUSY will both be cleared when the reset is complete.

30.8.2 Control B

Name: CTRLB

Offset: 0x1

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
	REFSEL[1:0]			-	VPD	LEFTADJ	IOEN	EOEN
Access	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:6 – REFSEL[1:0]: Reference Selection**
  These bits select the reference voltage for the DAC according to Table 30-1.

Table 30-1. Reference Selection

REFSEL[1:0]	Reference Selection	Description
0x0	INT1V	Internal 1.0V reference
0x1	AVCC	AV_{CC}
0x2	VREFP	External reference
0x3		Reserved

- **Bits 5:4 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 3 – VPD: Voltage Pump Disable**
  This bit controls the behavior of the voltage pump.
  0: Voltage pump is turned on/off automatically.
  1: Voltage pump is disabled.

- **Bit 2 – LEFTADJ: Left-Adjusted Data**
  This bit controls how the 10-bit conversion data is adjusted in the Data and Data Buffer registers.
  0: DATA and DATABUF registers are right-adjusted.
  1: DATA and DATABUF registers are left-adjusted.

- **Bit 1 – IOEN: Internal Output Enable**
  0: Internal DAC output not enabled.
  1: Internal DAC output enabled to be used by the AC.

- **Bit 0 – EOEN: External Output Enable**
  0: The DAC output is turned off.
  1: The high-drive output buffer drives the DAC output to the V_{OUT} pin.

30.8.3 Event Control

Name: EVCTRL

Offset: 0x2

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
							EMPTYEO	STARTEI
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:2 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 1 – EMPTYEO: Data Buffer Empty Event Output**
  This bit indicates whether or not the Data Buffer Empty event is enabled and will be generated when the Data Buffer register is empty.
  0: Data Buffer Empty event is disabled and will not be generated.
  1: Data Buffer Empty event is enabled and will be generated.

- **Bit 0 – STARTEI: Start Conversion Event Input**
  This bit indicates whether or not the Start Conversion event is enabled and data are loaded from the Data Buffer register to the Data register upon event reception.
  0: A new conversion will not be triggered on an incoming event.
  1: A new conversion will be triggered on an incoming event.

30.8.4 Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Name: INTENCLR

Offset: 0x4

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						SYNCRDY	EMPTY	UNDERRUN
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready Interrupt Enable bit, which disables the Synchronization Ready interrupt.

- **Bit 1 – EMPTY: Data Buffer Empty Interrupt Enable**
  0: The Data Buffer Empty interrupt is disabled.
  1: The Data Buffer Empty interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Data Buffer Empty Interrupt Enable bit, which disables the Data Buffer Empty interrupt.

- **Bit 0 – UNDERRUN: Underrun Interrupt Enable**
  0: The Underrun interrupt is disabled.
  1: The Underrun interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Underrun Interrupt Enable bit, which disables the Underrun interrupt.

30.8.5 Interrupt Enable Set

This register allows the user to enable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

Name: INTENSET

Offset: 0x5

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						SYNCRDY	EMPTY	UNDERRUN
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – SYNCRDY: Synchronization Ready Interrupt Enable**
  0: The Synchronization Ready interrupt is disabled.
  1: The Synchronization Ready interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Synchronization Ready Interrupt Enable bit, which enables the Synchronization Ready interrupt.

- **Bit 1 – EMPTY: Data Buffer Empty Interrupt Enable**
  0: The Data Buffer Empty interrupt is disabled.
  1: The Data Buffer Empty interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Data Buffer Empty Interrupt Enable bit, which enables the Data Buffer Empty interrupt.

- **Bit 0 – UNDERRUN: Underrun Interrupt Enable**
  0: The Underrun interrupt is disabled.
  1: The Underrun interrupt is enabled.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will set the Underrun Interrupt Enable bit, which enables the Underrun interrupt.

30.8.6 Interrupt Flag Status and Clear

Name: INTFLAG

Offset: 0x6

Reset: 0x00

Property: Write-Protected

Bit	7	6	5	4	3	2	1	0
						SYNCRDY	EMPTY	UNDERRUN
Access	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 7:3 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

- **Bit 2 – SYNCRDY: Synchronization Ready**
  This flag is cleared by writing a one to the flag.
  This flag is set on a 1-to-0 transition of the Synchronization Busy bit in the Status register (STATUS.SYNCBUSY), except when the transition is caused by an enable or a software reset, and will generate an interrupt request if INTENCLR/SET.READY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Synchronization Ready interrupt flag.

- **Bit 1 – EMPTY: Data Buffer Empty**
  This flag is cleared by writing a one to the flag or by writing new data to DATABUF.
  This flag is set when data is transferred from DATABUF to DATA, and the DAC is ready to receive new data in DATABUF, and will generate an interrupt request if INTENCLR/SET.EMPTY is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Data Buffer Empty interrupt flag.

- **Bit 0 – UNDERRUN: Underrun**
  This flag is cleared by writing a one to the flag.
  This flag is set when a start conversion event occurs when DATABUF is empty, and will generate an interrupt request if INTENCLR/SET.UNDERRUN is one.
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the Underrun interrupt flag.

30.8.7 Status

Name: STATUS

Offset: 0x7

Reset: 0x00

Property: Read-Synchronized

Bit	7	6	5	4	3	2	1	0
	SYNCBUSY							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

- **Bit 7 – SYNCBUSY: Synchronization Busy Status**
  This bit is cleared when the synchronization of registers between the clock domains is complete.
  This bit is set when the synchronization of registers between clock domains is started.

- **Bits 6:0 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written. These bits will always return zero when read.

30.8.8 Data

Name: DATA

Offset: 0x8

Reset: 0x0000

Property: Write-Synchronized, Read-Synchronized, Write-Protected

Bit	15	14	13	12	11	10	9	8
	DATA[15:8]							
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – DATA: Data value to be converted**
  DATA register contains the 10-bit value that is converted to a voltage by the DAC. The adjustment of these 10 bits within the 16-bit register is controlled by CTRLB.LEFTADJ:
  - DATA[9:0] when CTRLB.LEFTADJ is zero.
  - DATA[15:6] when CTRLB.LEFTADJ is one.

30.8.9 Data Buffer

Name: DATABUF

Offset: 0xC

Reset: 0x0000

Property: Write-Synchronized, Write-Protected

Bit	15	14	13	12	11	10	9	8
	DATABUF[15:8]							
Access	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATABUF[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

- **Bits 15:0 – DATABUF: Data Buffer**
  DATABUF contains the value to be transferred into DATA register.

31. PTC - Peripheral Touch Controller

31.1 Overview

The purpose of PTC is to acquire signals to detect touch on capacitive sensors. The external capacitive touch sensor is typically formed on a PCB, and the sensor electrodes are connected to the analog front end of the PTC through the I/O pins in the device. The PTC supports both self- and mutual-capacitance sensors.

In mutual-capacitance mode, sensing is done using capacitive touch matrices in various X-Y configurations, including indium tin oxide (ITO) sensor grids. The PTC requires one pin per X-line and one pin per Y-line.

In self-capacitance mode, the PTC requires only one pin (Y-line) for each touch sensor.

31.2 Features

- Low-power, high-sensitivity, environmentally robust capacitive touch buttons, sliders, wheels and proximity sensing
  - Down to 8µA with 200ms scan rate
- Supports mutual capacitance and self-capacitance sensing
  - 6/10/16 buttons in self-capacitance mode, for 32-/48-/64- pins respectively
  - 60/120/256 buttons in mutual-capacitance mode, for 32-/48-/64- pins respectively
  - Mix-and-match mutual-and self-capacitance sensors
- One pin per electrode – no external components
- Load compensating charge sensing
  - Parasitic capacitance compensation and adjustable gain for superior sensitivity
- Zero drift over the temperature and V_{DD} range
  - Auto calibration and re-calibration of sensors
- Single-shot and free-running charge measurement
- Hardware noise filtering and noise signal de-synchronization for high conducted immunity
- Selectable channel change delay
  - Allows choosing the settling time on a new channel, as required
- Acquisition-start triggered by command or interrupt event
- Low CPU utilization through interrupt on acquisition-complete
  - 5% CPU utilization scanning 10 channels at 50ms scan rate
- Supported by the Atmel® QTouch® Composer development tool, which comprises QTouch Library project builder and QTouch analyzer

31.3 Block Diagram

Figure 31-1. PTC Block Diagram Mutual-capacitance

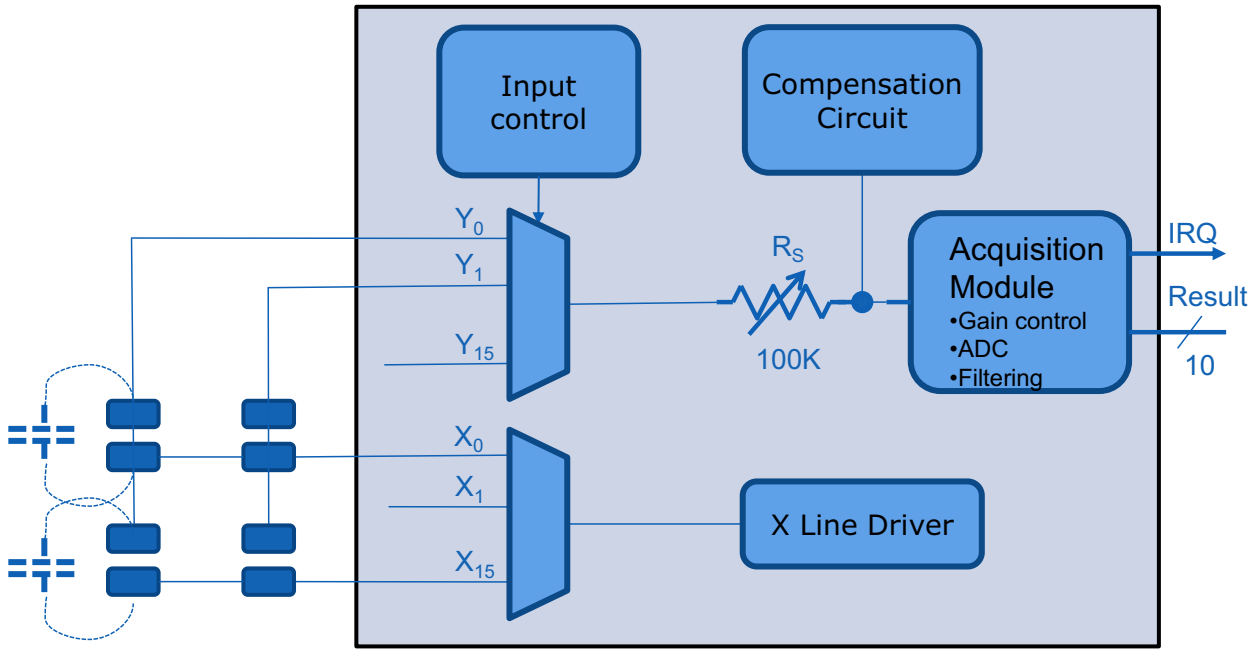
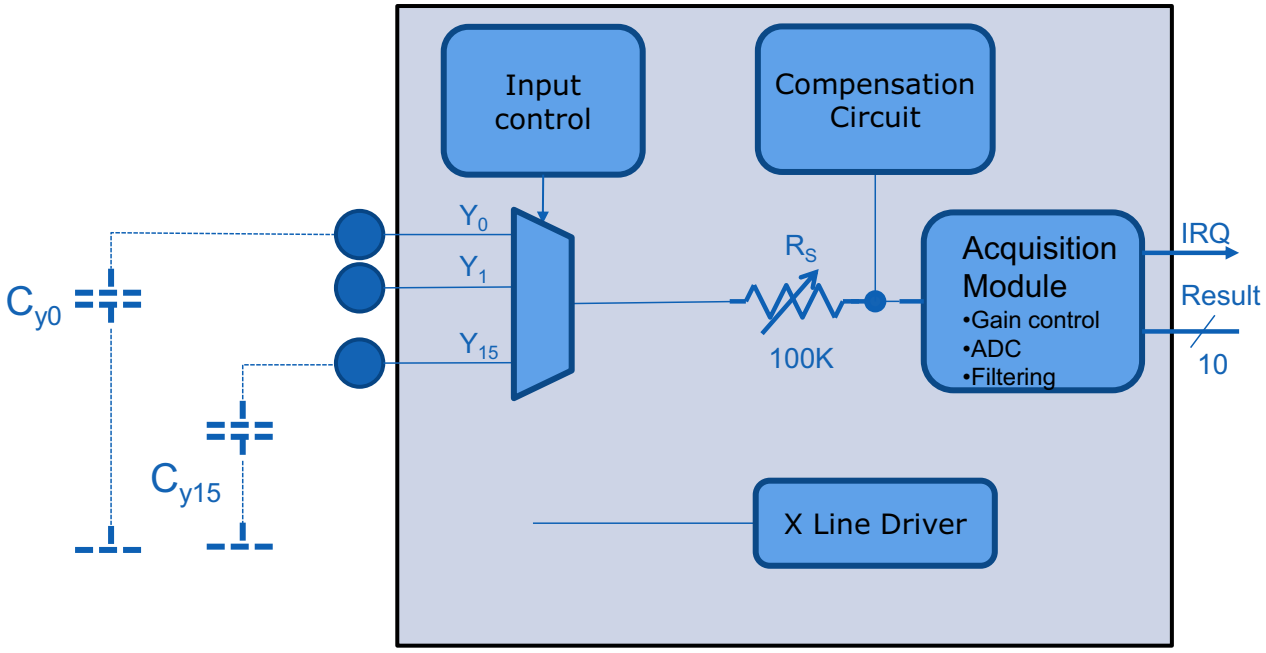


Figure 31-2. PTC Block Diagram Self-capacitance



31.4 Signal Description

Name	Type	Description
X[n:0]	Digital	X-line (Output)
Y[m:0]	Analog	Y-line (Input/Output)

Note: 1. The number of X and Y lines are device dependent. Refer to "Configuration Summary" on page 3 for details. Refer to "I/O Multiplexing and Considerations" on page 11 for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

31.5 Product Dependencies

In order to use this Peripheral, configure the other components of the system as described in the following sections.

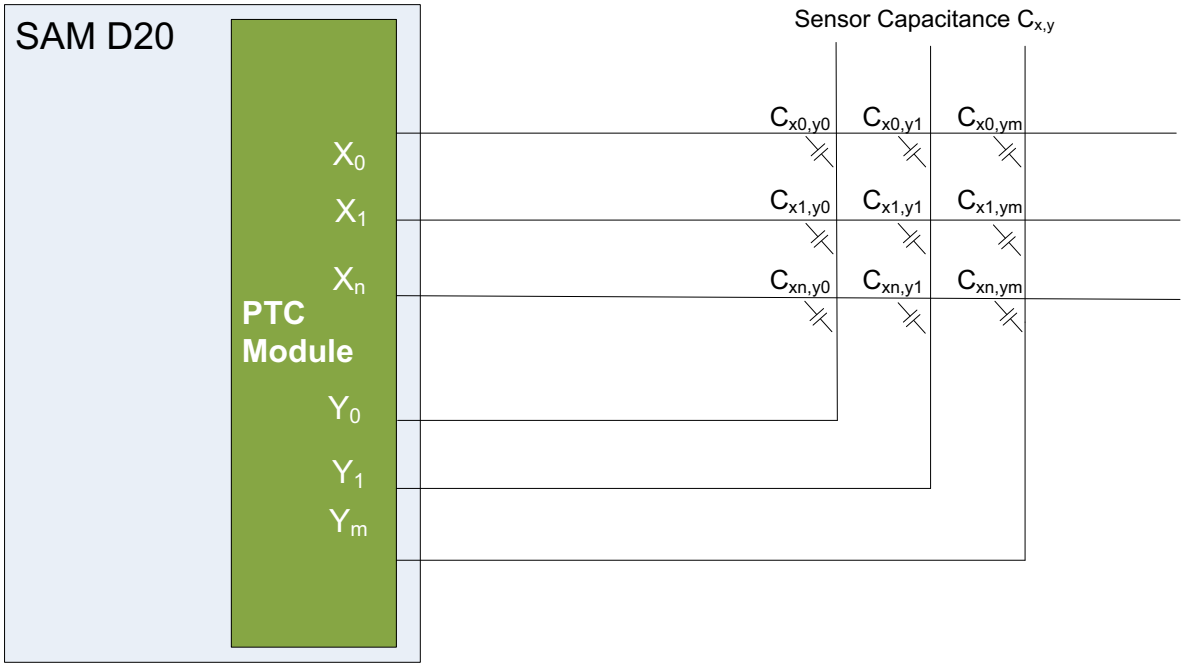
31.5.1 I/O Lines

The I/O lines used for analog X-lines and Y-lines must be connected to external capacitive touch sensor electrodes. External components are not required for normal operation. However, to improve the EMC performance, a series resistor of 1 KΩ can be used on X-lines and Y-lines.

Mutual-capacitance Sensor Arrangement

A mutual-capacitance sensor is formed between two I/O lines - an X electrode for transmitting and Y electrode for receiving. The mutual capacitance between the X and Y electrode is measured by the Peripheral Touch Controller.

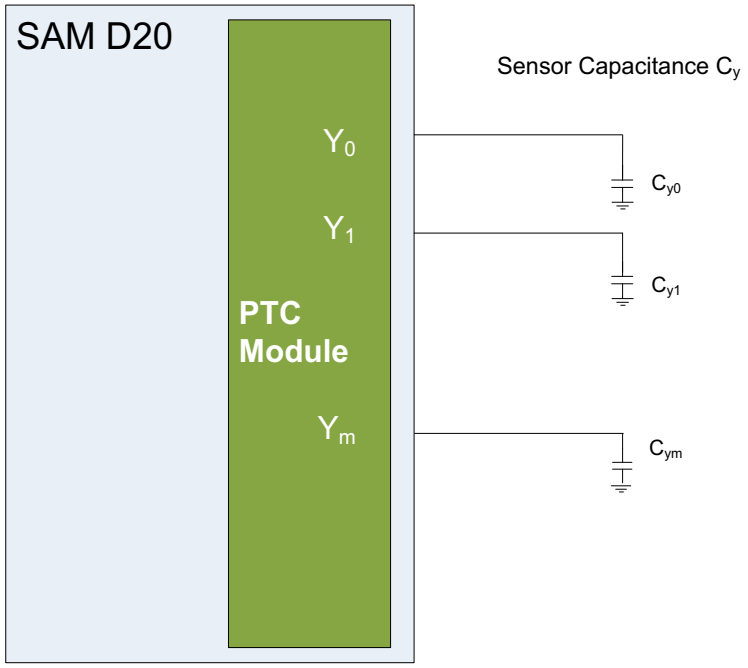
Figure 31-3. Mutual Capacitance Sensor Arrangement



Self-capacitance Sensor Arrangement

The self-capacitance sensor is connected to a single pin on the Peripheral Touch Controller through the Y electrode for receiving the signal. The sense electrode capacitance is measured by the Peripheral Touch Controller.

Figure 31-4. Self-capacitance Sensor Arrangement



For more information about designing the touch sensor, refer to Buttons, Sliders and Wheels Touch Sensor Design Guide on http://www.atmel.com.

31.5.2 Clocks

The PTC is clocked by the GCLK_PTC clock. The PTC operates from an asynchronous clock source and the operation is independent of the main system clock and its derivative clocks, such as the peripheral bus clock (CLK_APB). A number of clock sources can be selected as the source for the asynchronous GCLK_PTC. The clock source is selected by configuring the Generic Clock Selection ID in the Generic Clock Control register. For more information about selecting the clock sources, refer to “GCLK – Generic Clock Controller” on page 78.

The selected clock must be enabled in the Power Manager, before it can be used by the PTC. By default these clocks are disabled. The frequency range of GCLK_PTC is 400kHz to 4MHz.

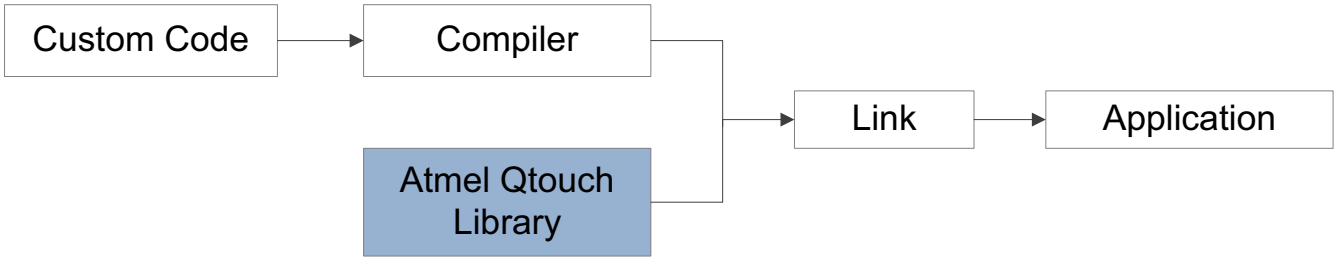
For more details, refer to “PM – Power Manager” on page 100.

31.6 Functional Description

In order to access the PTC, the user must use the QTouch Composer tool to configure and link the QTouch Library firmware with the application code. QTouch Library can be used to implement buttons, sliders, wheels and proximity sensor in a variety of combinations on a single interface.

For more information about QTouch library, refer to the Atmel QTouch Library Peripheral Touch Controller User Guide.

Figure 31-5. QTouch Library Usage



32. Electrical Characteristics

32.1 Disclaimer

All values in this chapter are preliminary and subject to change without further notice.

All typical values are measured at T = 25°C unless otherwise specified. All minimum and maximum values are valid across operating temperature and voltage unless otherwise specified.

32.2 Absolute Maximum Ratings

Stresses beyond those listed in Table 32-1 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.

Table 32-1. Absolute maximum ratings

Symbol	Parameter	Min.	Max.	Units
V_{DD}	Power supply voltage		3.63	V
I_{VDD}	Current into a V_{DD} pin		50	mA
I_{GND}	Current out of a GND pin		50	mA
V_{PIN}	Pin voltage with respect to GND and V_{DD}	GND-0.3V	V_{DD}+0.3V	V
I_{PIN}	I/O pin sink/source current			mA

32.3 General Operating Ratings

The device must operate within the ratings listed in Table 32-2 in order for all other electrical characteristics and typical characteristics of the device to be valid.

Table 32-2. General operating conditions(1)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{DD}	Power supply voltage		1.62(2)	3.3	3.63	V
V_{DDANA}	Analog supply voltage		1.62(2)	3.3	3.63	V
T_A	Temperature range		-40	25	85	°C
T_J	Junction temperature				100	°C

Notes: 1. These values are based on characterization. These values are not covered by test limits in production
2. With BOD33 disabled. If the BOD33 is enabled, check table “BOD33 Level Value”

32.4 Supply Characteristics

The following characteristics are applicable to the operating temperature range: T_A = -40°C to 85°C, unless otherwise specified and are valid for a junction temperature up to T_J = 100°C. Refer to "Power Supply and Start-Up Considerations" on page 16.

Table 32-3. Supply Characteristics

Symbol	Conditions	Voltage		
		Min.	Max.	Units
V_{DDIO} V_{DDIN} V_{DDANA}	Full Voltage Range	1.62	3.63	V

Table 32-4. Supply Rise Rates

Symbol	Parameter	Rise Rate	Units
		Max.	
V_{DDIO} V_{DDIN} V_{DDANA}	DC supply peripheral I/Os, internal regulator and analog supply voltage	0.1	V/µs

32.5 Maximum Clock Frequencies

Table 32-5. Maximum Clock Frequencies

Symbol	Parameter	Description	Max.	Units
f_{CPU}	CPU clock frequency		48	MHz
f_{AHB}	AHB clock frequency			
f_{APBA}	APBA clock frequency			
f_{APBB}	APBB clock frequency			
f_{APBC}	APBC clock frequency			
f_{GCLK0}	GCLK0 clock frequency	DFLL48M Reference		
f_{GCLK1}	GCLK1 clock frequency	WDT		
f_{GCLK2}	GCLK2 clock frequency	RTC		
f_{GCLK3}	GCLK3 clock frequency	EIC		
f_{GCLK4}	GCLK4 clock frequency	EVSYS_CHANNEL_0		
f_{GCLK5}	GCLK5 clock frequency	EVSYS_CHANNEL_1		
f_{GCLK6}	GCLK6 clock frequency	EVSYS_CHANNEL_2		
f_{GCLK7}	GCLK7 clock frequency	EVSYS_CHANNEL_3		
f_{GCLK8}	GCLK8 clock frequency	EVSYS_CHANNEL_4		
f_{GCLK9}	GCLK9 clock frequency	EVSYS_CHANNEL_5		

Table 32-5. Maximum Clock Frequencies (Continued)

Symbol	Parameter	Description	Max.	Units
f_{GCLK10}	GCLK10 clock frequency	EVSYS_CHANNEL_6	48	MHz
f_{GCLK11}	GCLK11 clock frequency	EVSYS_CHANNEL_7		
f_{GCLK12}	GCLK12 clock frequency	SERCOMx_SLOW		
f_{GCLK13}	GCLK13 clock frequency	SERCOM0_CORE		
f_{GCLK14}	GCLK14 clock frequency	SERCOM1_CORE		
f_{GCLK15}	GCLK15 clock frequency	SERCOM2_CORE		
f_{GCLK16}	GCLK16 clock frequency	SERCOM3_CORE		
f_{GCLK17}	GCLK17 clock frequency	SERCOM4_CORE		
f_{GCLK18}	GCLK18 clock frequency	SERCOM5_CORE		
f_{GCLK19}	GCLK19 clock frequency	TC0,TC1		
f_{GCLK20}	GCLK20 clock frequency	TC2,TC3		
f_{GCLK21}	GCLK21 clock frequency	TC4,TC5		
f_{GCLK22}	GCLK22 clock frequency	TC6,TC7		
f_{GCLK23}	GCLK23 clock frequency	ADC		
f_{GCLK24}	GCLK24 clock frequency	AC_DIG		
f_{GCLK25}	GCLK25 clock frequency	AC_ANA		
f_{GCLK26}	GCLK26 clock frequency	DAC		
f_{GCLK27}	GCLK27 clock frequency	PTC		

32.6 Power Consumption

The values in Table 32-6 are measured values of power consumption under the following conditions, except where noted:

- Operating conditions
  - V_{VDDIN} = 3.3V
- Wake up time from sleep mode is measured from the edge of the wakeup signal to the execution of the first instruction fetched in flash.
- Oscillators
  - XOSC (crystal oscillator) stopped
  - XOSC32K (32kHz crystal oscillator) running with external 32kHz crystal
  - DFLL48M using XOSC32K as reference and running at 48MHz
- Clocks
  - DFLL48M used as main clock source
  - CPU, AHB clocks undivided
  - APBA clock divided by 4
  - APBB and APBC bridges off
- The following AHB module clocks are running: NVMCTRL, APBA bridge
  - All other AHB clocks stopped
- The following peripheral clocks running: PM, SYSCTRL, RTC
  - All other peripheral clocks stopped
- I/Os are inactive with internal pull-up
- CPU is running on flash with 1 wait states
- Low power cache enabled
- BOD33 disabled

Table 32-6. Current Consumption

Mode	Conditions	T_A	Min.	Typ.	Max.	Units
ACTIVE	CPU running a While(1) algorithm	25°C		70		µA/MHz
		85°C		72		
	CPU running a While(1) algorithm V_{DDIN}=1.8V, CPU is running on Flash with 3 wait states	25°C		70		
		85°C		72		
	CPU running a Fibonacci algorithm	25°C		109		
		85°C		110		
	CPU running a Fibonacci algorithm V_{DDIN}=1.8V, CPU is running on flash with 3 wait states	25°C		109		
		85°C		110		
	CPU running a CoreMark algorithm	25°C		140		µA/MHz
		85°C		145		
	CPU running a CoreMark algorithm V_{DDIN}=1.8V, CPU is running on flash with 3 wait states	25°C		125		
		85°C		130		

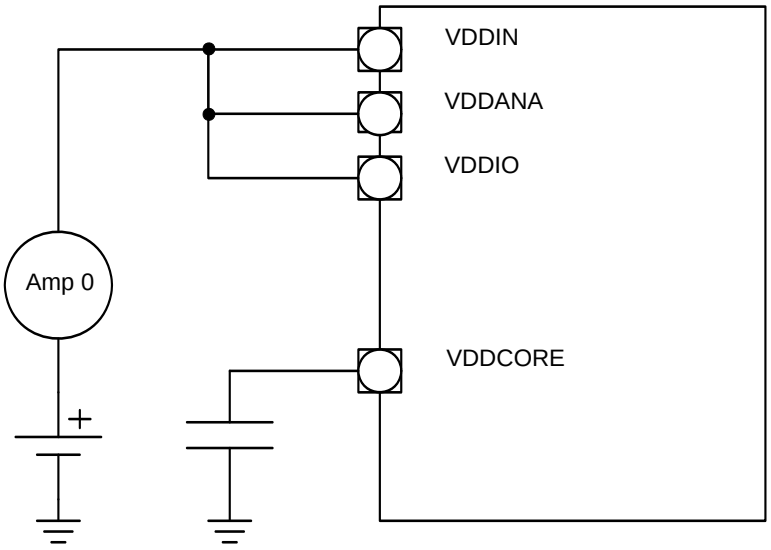
Table 32-6. Current Consumption (Continued)

Mode	Conditions	T_A	Min.	Typ.	Max.	Units
IDLE0		25°C		2.35		mA
		85°C		2.4		mA
IDLE1		25°C		1.9		mA
		85°C		2		mA
IDLE2		25°C		1.7		mA
		85°C		1.8		mA
STANDBY	XOSC32K running RTC running at 1 kHz	25°C		3.2		µA
		85°C		30		µA
	XOSC32K and RTC stopped	25°C		2		µA
		85°C		28		µA

Table 32-7. Wake-up Time

Mode	Conditions	T_A	Min.	Typ.	Max.	Units
IDLE0	OSC8M used as main clock source, low power cache disabled	25°C		3.5		µs
		85°C				µs
IDLE1	OSC8M used as main clock source, low power cache disabled	25°C		12		µs
		85°C				µs
IDLE2	OSC8M used as main clock source, low power cache disabled	25°C		12.5		µs
		85°C				µs
STANDBY	OSC8M used as main clock source, low power cache disabled	25°C		20		µs
		85°C				µs

Figure 32-1. Measurement Schematic



32.7 I/O Pin Characteristics

32.7.1 Normal I/O Pins

Table 32-8. Normal I/O Pins Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
R_{PULL}	Pull-up - Pull-down resistance		20	38	60	kΩ
V_{IL}	Input low-level voltage	V_{DD}=1.62V-2.7V	-	-	0.25*V_{DD}	V
		V_{DD}=2.7V-3.63V	-	-	0.3*V_{DD}	
V_{IH}	Input high-level voltage	V_{DD}=1.62V-2.7V	0.7*V_{DD}	-	-	
		V_{DD}=2.7V-3.63V	0.55*V_{DD}	-	-	
V_{OL}	Output low-level voltage	V_{DD}>1.6V, I_{OL} max	-	0.1*V_{DD}	0.2*V_{DD}	
V_{OH}	Output high-level voltage	V_{DD}>1.6V, I_{OH} max	0.8*V_{DD}	0.9*V_{DD}		
I_{OL}	Output low-level current	V_{DD}=1.6V-3V	-	-	8	mA
		V_{DD}=3V-3.63V	-	-	20	
I_{OH}	Output high-level current	V_{DD}=1.6V-3V	-	-	4.5	
		V_{DD}=3V-3.63V	-	-	10	
I_{LEAK}	Input leakage current	Pull-up resistors disabled	-1	+/-0.015	1	µA

32.7.2 I²C Pins

Refer to "I/O Multiplexing and Considerations" on page 11 to get the list of I²C pins.

Table 32-9. I²C Pins Characteristics in I²C configuration

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
R_{PULL}	Pull-up - Pull-down resistance		20	38	60	kΩ
V_{IL}	Input low-level voltage	V_{DD}=1.62V-2.7V	-	-	0.25*V_{DD}	V
		V_{DD}=2.7V-3.63V	-	-	0.3*V_{DD}	
V_{IH}	Input high-level voltage	V_{DD}=1.62V-2.7V	0.7*V_{DD}	-	-	
		V_{DD}=2.7V-3.63V	0.55*V_{DD}	-	-	
V_{HYS}	Hysteresis of Schmitt trigger inputs		0.8*V_{DD}	-	-	
V_{OL}	Output low-level voltage	V_{DD}> 2.0V I_{OL}=3mA	-	-	0.4	
		V_{DD}≤2.0V I_{OL}=2mA	-	-	0.2*V_{DD}	
I_{OL}	Output low-level current	V_{OL} =0.4V	3	-	-	mA
		V_{OL} =0.6V	6	-	-	
f_{SCL}	SCL clock frequency		-	-	400	kHz

I²C pins timing characteristics can be found in "SERCOM in I2C Mode Timing" on page 588.

Table 32-10. I²C Pins Characteristics in PORT Configuration

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
R_{PULL}	Pull-up - Pull-down resistance		20	38	60	kΩ
V_{IL}	Input low-level voltage	V_{DD}=1.62V-2.7V	-	-	0.25*V_{DD}	V
		V_{DD}=2.7V-3.63V	-	-	0.3*V_{DD}	
V_{IH}	Input high-level voltage	V_{DD}=1.62V-2.7V	0.7*V_{DD}	-	-	
		V_{DD}=2.7V-3.63V	0.55*V_{DD}	-	-	
V_{OL}	Output low-level voltage	V_{DD}>1.6V, I_{OL} max	-	0.1*V_{DD}	0.2*V_{DD}	
V_{OH}	Output high-level voltage	V_{DD}>1.6V, I_{OH} max	0.8*V_{DD}	0.9*V_{DD}		
I_{OL}	Output low-level current	V_{DD}=1.6V-3V	-	-	8	mA
		V_{DD}=3V-3.63V	-	-	20	
I_{OH}	Output high-level current	V_{DD}=1.6V-3V	-	-	4.5	
		V_{DD}=3V-3.63V	-	-	10	
I_{LEAK}	Input leakage current	Pull-up resistors disabled	-1	+/-0.015	1	µA

32.7.3 XOSC Pin

XOSC pins behave as normal pins when used as normal I/Os. Refer to Table 32-8.

32.7.4 XOSC32 Pin

XOSC32 pins behave as normal pins when used as normal I/Os. Refer to Table 32-8.

32.7.5 External Reset Pin

Reset pin has the same electrical characteristics as normal I/O pins. Refer to Table 32-8.

32.8 Analog Characteristics

32.8.1 Voltage Regulator Characteristics

Table 32-11. VREG Electrical Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
V_{DDIN}	Input voltage range		1.62	3.3	3.63	V
V_{DDCORE}	DC calibrated output voltage		1.1	1.23	1.30	

Note: Supplying any external components using V_{DDCORE} pin is not allowed to assure the integrity of the core supply voltage.

Table 32-12. Decoupling requirements

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
C_{IN}	Input regulator capacitor		-	1	-	µF
C_{OUT} (1)	Output regulator capacitor	I_{VDDIO}=10mA	-	1	-	µF

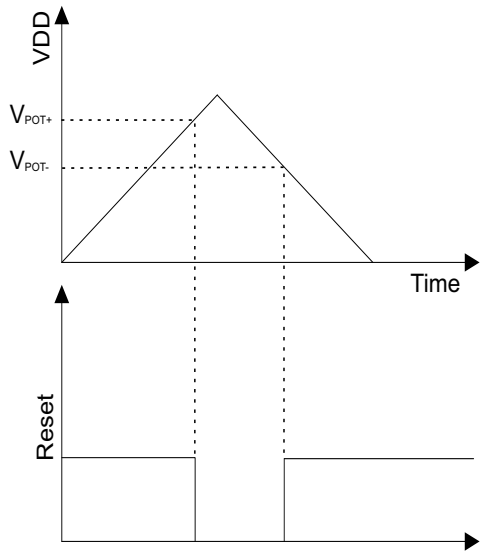
Note: 1. C_{OUT} closer to 1µF for I_{VDDIO} >= 10mA, C_{OUT} closer to 0.1µF for lower I_{VDDIO}.

32.8.2 Power-On Reset (POR) Characteristics

Table 32-13. POR Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
V_{POT+}	Voltage threshold on V_{DDIN} rising	V_{DD} falls at 1V/ms or slower	1.27	1.45	1.58	V
V_{POT-}	Voltage threshold on V_{DDIN} falling		0.72	0.99	1.02	V

Figure 32-2. POR Operating Principle



32.8.3 Brown-Out Detectors Characteristics

32.8.3.1 BOD33

Table 32-14. BOD33 LEVEL Value

BOD33.LEVEL	Conditions	Min.	Typ.	Max.	Units
6	Hysteresis on	-	1.715	1.745	V
7		-	1.750	1.779	
39		-	2.84	2.92	
48		-	3.2	3.3	
6	Hysteresis off	1.62	1.64	1.67	
7		1.64	1.675	1.71	
39		2.72	2.77	2.81	
48		3.0	3.07	3.2	

Table 32-15. BOD33 Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
	Step size, between adjacent values in BOD33.LEVEL		-	34	-	mV
V_{HYST}	Hysteresis		35		170	mV
t_{DET}	Detection time	Time with $V_{DDIN} < V_{TH}$ necessary to generate a reset signal		0.9(1)		µs
I_{BOD33}	Current consumption	Continuous mode		33(1)		µA
		Sampling mode				µA
$t_{STARTUP}$	Startup time			2.2(1)		µs

Note: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

32.8.4 Analog-to-Digital (ADC) characteristics

Table 32-16. Operating Conditions

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
RES	Resolution		8		12	bits
f_{CLK_ADC}	ADC Clock frequency		30		2100	kHz
	Sample rate(1)	Single shot	5		323	ksps
		Free running	5		350	ksps

Table 32-16. Operating Conditions (Continued)

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
	Sampling time[1]		0.5			cycles
	Conversion time[1]	1x Gain		6		cycles
V_{REF}	Voltage reference range		1.0		V_{DDANA}-0.6	V
$V_{REFINT1V}$	Internal 1V reference [2]			1.0		V
$V_{REFINTVCC0}$	Internal ratiometric reference 0[2]			V_{DDANA}/1.48		V
$V_{REFINTVCC1}$	Internal ratiometric reference 1[2]	V_{DDANA}>2.0V		V_{DDANA}/2		V
	Conversion range[1]	Differential mode	-V_{REF}/GAIN		+V_{REF}/GAIN	V
		Single-ended mode	0.0		+V_{REF}/GAIN	V
C_{SAMPLE}	Sampling capacitance[2]			3.5		pF
R_{SAMPLE}	Input channel source resistance[2]			2.8	2.8	kΩ
I_{DD}	DC supply current[1]	f_{CLK_ADC} = 2.1MHz[3]	0.77	1.15	1.54	mA

Notes:
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.
3. In this condition and for a sample rate of 350ksps, 1 Conversion at gain 1x takes 6 clock cycles of the ADC clock.

Table 32-17. Differential Mode

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
ENOB	Effective Number Of Bits	With gain compensation	-	10.5	11.1	bits
TUE	Total Unadjusted Error	1x Gain	1.5	4.3	15.0	LSB
INL	Integral Non Linearity	1x Gain	1.0	1.3	4.5	LSB
DNL	Differential Non Linearity	1x Gain	+/-0.3	+/-0.5	+/-0.95	LSB
	Gain Error	Ext. Ref 1x	-10.0	2.5	+10.0	mV
		V_{REF}=V_{DDANA}/1.48	-15.0	-1.5	+10.0	mV
		Bandgap	-15.0	-5.0	+5.0	mV
	Gain Accuracy	Ext. Ref. 0.5x	+/-0.1	+/-0.2	+/-0.45	%
		Ext. Ref. 2x to 16x	+/-0.05	+/-0.1	+/-0.11	%
	Offset Error	Ext. Ref. 1x	-5.0	-1.5	+5.0	mV
		V_{REF}=V_{DDANA}/1.48	-5.0	0.5	+5.0	mV
		Bandgap	-5.0	3.0	+5.0	mV

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
SFDR	Spurious Free Dynamic Range	1x Gain F_{CLK_ADC} = 2.1MHz F_{IN} = 40kHz A_{IN} = 95%FSR	62.7	70.0	75.0	dB
SINAD	Signal-to-Noise and Distortion		54.1	65.0	68.5	dB
SNR	Signal-to-Noise Ratio		54.5	65.5	68.6	dB
THD	Total Harmonic Distortion		-77.0	-64.0	-63.0	dB
	Noise RMS	T=25°C	0.6	1.0	1.6	mV

Notes:
1. Maximum numbers are based on characterization and not tested in production, and valid for 5% to 95% of the input voltage range.
2. Dynamic parameter numbers are based on characterization and not tested in production.
3. Respect the input common mode voltage through the equation: $0.2*V_{DDANA} – 0.1V < V_{CM\_IN} < 0.95*V_{DDANA} + V_{REF}/4 – 0.75V$ (where V_{CM_IN} is the Input channel common mode voltage)

Table 32-18. Single-Ended Mode

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
ENOB	Effective Number of Bits	With gain compensation	-	9.5	9.8	Bits
TUE	Total Unadjusted Error	1x gain	-	10.5	14.0	LSB
INL	Integral Non-Linearity	1x gain	1.0	1.6	3.5	LSB
DNL	Differential Non-Linearity	1x gain	+/-0.5	+/-0.6	+/-0.95	LSB
	Gain Error	Ext. Ref. 1x	-5.0	0.7	+5.0	mV
	Gain Accuracy	Ext. Ref. 0.5x	+/-0.2	+/-0.34	+/-0.4	%
		Ext. Ref. 2x to 16X	+/-0.01	+/-0.1	+/-0.2	%
	Offset Error	Ext. Ref. 1x	-5.0	1.5	+5.0	mV
SFDR	Spurious Free Dynamic Range	1x Gain F_{CLK_ADC} = 2.1MHz F_{IN} = 40kHz A_{IN} = 95%FSR	63.1	65.0	67.0	dB
SINAD	Signal-to-Noise and Distortion		47.5	59.5	61.0	dB
SNR	Signal-to-Noise Ratio		48.0	60.0	64.0	dB
THD	Total Harmonic Distortion		-65.4	-63.0	-62.1	dB
	Noise RMS	T = 25°C		1.0		mV

Notes:
1. Maximum numbers are based on characterization and not tested in production, and for 5% to 95% of the input voltage range.
2. Respect the input common mode voltage through the equation: $V_{REF}/4 - 0.3*V_{DDANA} – 0.1V < V_{CM\_IN} < 0.7*V_{DDANA} + V_{REF}/4 – 0.75V$ (where V_{CM_IN} is the Input channel common mode voltage)

32.8.4.1 Performance with the Averaging Digital Feature

Averaging is a feature which increases the sample accuracy. ADC automatically computes an average value of multiple consecutive conversions. The numbers of samples to be averaged is specified by the Number-of-Samples-to-be-collected bit group in the Average Control register (AVGCTRL.SAMPLENUM[3:0]) and the averaged output is available in the Result register (RESULT).

Table 32-19. Averaging feature

Average Number	Conditions	SNR (dB)	SINAD (dB)	SFDR (dB)	ENOB (bits)
1	In differential mode, 1x gain, V_{DDANA}=3.0V, V_{REF}=1.0V, 350kSps	66.0	65.0	72.8	9.75
8		67.6	65.8	75.1	10.62
32		69.7	67.1	75.3	10.85
128		70.4	67.5	75.5	10.91

32.8.4.2 Performance with the hardware offset and gain correction

Inherent gain and offset errors affect the absolute accuracy of the ADC. The offset error cancellation is handled by the Offset Correction register (OFFSETCORR) and the gain error cancellation, by the Gain Correction register (GAINCORR). The offset and gain correction value is subtracted from the converted data before writing the Result register (RESULT).

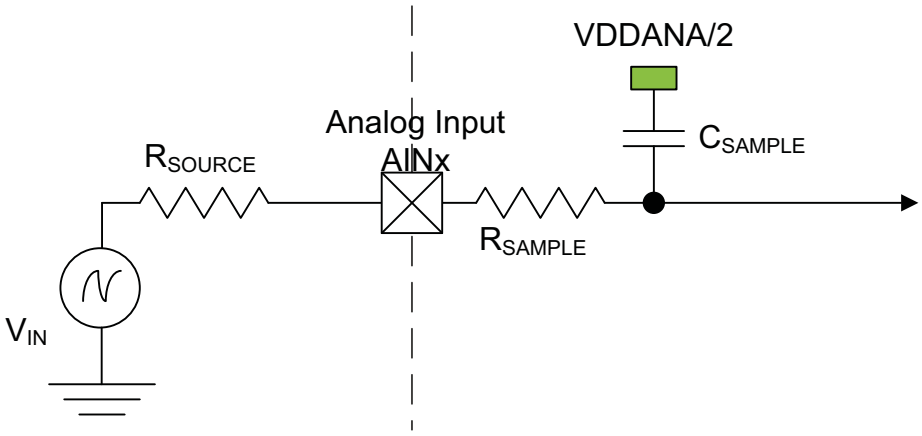
Table 32-20. Offset and Gain correction feature

Gain Factor	Conditions	Offset Error (mV)	Gain Error (mV)	Total Unadjusted Error (LSB)
0.5x	In differential mode, 1x gain, V_{DDANA}=3.0V, V_{REF}=1.0V, 350kSps	0.25	1.0	2.4
1x		0.20	0.10	1.5
2x		0.15	-0.15	2.7
8x		-0.05	0.05	3.2
16x		0.10	-0.05	6.1

32.8.4.3 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 32-3 shows the ADC input channel equivalent circuit.

Figure 32-3. ADC Input



To achieve *n* bits of accuracy, the C_{SAMPLE} capacitor must be charged at least to a voltage of

$$V_{CSAMPLE} \geq V_{IN} - 2^{n+1}$$

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

$$t_{SAMPLEHOLD} = \frac{1}{2 \times f_{ADC}}$$

32.8.5 Digital to Analog Converter (DAC) Characteristics

Table 32-21. Operating Conditions(1)

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
V_{DDANA}	Analog supply voltage		1.62	-	3.63	V
AV_{REF}	External reference voltage		1.0	-	V_{DDANA}-0.6	V
	Internal reference voltage 1		-	1	-	V
	Internal reference voltage 2		-	V_{DDANA}	-	V
	Linear output voltage range		0.05	-	V_{DDANA}-0.05	V
	Minimum resistive load		5	-	-	kΩ
	Maximum capacitance load		-	-	100	pF
i_{DD}	DC supply current(2)	V_{DDANA}=1.6V	-	145	190	µA
		V_{DDANA}=3.6V	-	180	460	µA

Notes: 1. These values are based on specifications otherwise noted.
2. These values are based on characterization. These values are not covered by test limits in production.

Table 32-22. Clock and Timing(1)

Symbol	Parameter	Conditions		Min.	Typ.	Max.	Units
f_{GCLK_DAC}	Conversion rate	C_{load}=100pF	Normal mode	-	-	350	ksps
	Startup time	V_{DDNA} > 2.6V		-	-	2.85	µs
		V_{DDNA} < 2.6V		-	-	10	µs

Note: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.

Table 32-23. Accuracy Characteristics

Symbol	Parameter	Conditions		Min.	Typ.	Max.	Units
RES	Input resolution[1]			-	-	10	Bits
INL	Integral non-linearity	V_{REF}= Ext 1.0V	V_{DD} = 1.6V	0.75	1.1	2.5	LSB
			V_{DD} = 3.6V	0.6	1.2	1.5	
		V_{REF} = V_{DDANA}	V_{DD} = 1.6V	1.4	2.2	2.5	
			V_{DD} = 3.6V	0.9	1.4	1.5	
		V_{REF}= INT1V	V_{DD} = 1.6V	0.75	1.3	1.5	
			V_{DD} = 3.6V	0.8	1.2	1.5	
DNL	Differential non-linearity	V_{REF}= Ext 1.0V	V_{DD} = 1.6V	+/-0.9	+/-1.2	+/-1.5	LSB
			V_{DD} = 3.6V	+/-0.9	+/-1.1	+/-1.2	
		V_{REF}= V_{DDANA}	V_{DD} = 1.6V	+/-1.1	+/-1.7	+/-1.5	
			V_{DD} = 3.6V	+/-1.0	+/-1.1	+/-1.2	
		V_{REF}= INT1V	V_{DD} = 1.6V	+/-1.1	+/-1.4	+/-1.5	
			V_{DD} = 3.6V	+/-1.0	+/-1.5	+/-1.6	
	Gain error	Ext. V_{REF}		+/-1.5	+/-5	+/-10	mV
	Offset error	Ext. V_{REF}		+/-2	+/-3	+/-6	mV

Note: 1. These values are according to specification. These values are not covered by test limits in production or characterization.

32.8.6 Analog Comparator Characteristics

Table 32-24. Electrical and Timing

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
	Positive input voltage range		0	-	V_{DDANA}	V
	Negative input voltage range		0	-	V_{DDANA}	
	Offset	Hysteresis = 0, Fast mode	-15	0.0	+15	mV
		Hysteresis = 0, Low power mode	-25	0.0	+25	mV
	Hysteresis	Hysteresis = 1, Fast mode	20	50	80	mV
		Hysteresis = 1, Low power mode	15	40	75	mV
	Propagation delay	Changes for V_{ACM}=V_{DDANA}/2 100mV overdrive, Fast mode	-	60	116	ns
		Changes for V_{ACM}=V_{DDANA}/2 100mV overdrive, Low power mode	-	225	370	ns

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
$t_{STARTUP}$	Startup time	Enable to ready delay Fast mode	-	1	2	µs
		Enable to ready delay Low power mode	-	12	19	µs
V_{SCALE}	INL		-	0.75	1.4	LSB
	DNL		-	0.25	0.9	LSB
	Offset Error		-27	12	55	mV
	Gain Error		-40	44	130	mV

32.8.7 Internal 1.1V Bandgap Reference Characteristics

Table 32-25. Bandgap and Internal 1.1V reference characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
INT1V	Internal 1.1V Bandgap reference	After calibration at T= 25°C, over [-40, +85]C	1.08	1.1	1.115	V
	Accuracy	After calibration at T= 25°C, over voltage and temperature.		1	1.5	%
		After calibration at T= 25°C, over voltage.	1.089	1.1	1.11	%

32.8.8 Temperature Sensor Characteristics

Table 32-26. Temperature Sensor characteristics(1)

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
	Temperature sensor output voltage	T= 25°C, V_{DDANA} = 3.3V	-	0.667	-	V
	Temperature sensor slope		2.3	2.4	2.5	mV/°C
	Variation over V_{DDANA} voltage	V_{DDANA}=1.62V to 3.6V	-1.7	1	3.7	mV/V

Note: 1. These values are based on characterization. These values are not covered by test limits in production.

32.9 NVM Characteristics

Table 32-27. Maximum Operating Frequency

V_{DD} range	NVM Wait States	Maximum Operating Frequency	Units
1.62V to 2.7V	0	14	MHz
	1	28	
	2	42	
	3	48	
2.7V to 3.63V	0	24	
	1	48	

Note that on this flash technology, a max number of 8 consecutive write is allowed per row. Once this number is reached, a row erase is mandatory.

Table 32-28. Flash Endurance and Data Retention

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
Ret_{NVM25k}	Retention after up to 25k	Average ambient 55°C	10	50		Years
$Ret_{NVM2.5k}$	Retention after up to 2.5k	Average ambient 55°C	20	100		Years
Ret_{NVM100}	Retention after up to 100	Average ambient 55°C	25	>100		Years
Cyc_{NVM}	Cycling Endurance(1)	-40°C < Ta < 85°C	25k	150k		Cycles

Note: 1. An endurance cycle is a write and an erase operation.

Table 32-29. Eeprom Emulation(1) Endurance and Data Retention

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
$Ret_{EEPROM100k}$	Retention after up to 100k	Average ambient 55°C	10	50		Years
$Ret_{EEPROM10k}$	Retention after up to 10k	Average ambient 55°C	20	100		Years
Cyc_{EEPROM}	Cycling Endurance(2)	-40°C < Ta < 85°C	100k	600k		Cycles

Notes: 1. The EEPROM emulation is a software emulation described in the App note AT03265.
2. An endurance cycle is a write and an erase operation.

Table 32-30. NVM Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
t_{FPP}	Page programming time				2.5	ms
t_{FRE}	Row erase time				6	ms
t_{FFP}	Fuse programming time					
t_{FEA}	Erase all page time (EA)					
t_{FCE}	DSU chip erase time (CHIP_ERASE)				240	ms

32.10 Oscillators Characteristics

32.10.1 Crystal Oscillator (XOSC) Characteristics

32.10.1.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN.

Table 32-31. Digital Clock Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{CPXIN}	XIN clock frequency		0.4		32	MHz

32.10.1.2 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 32-4. 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:

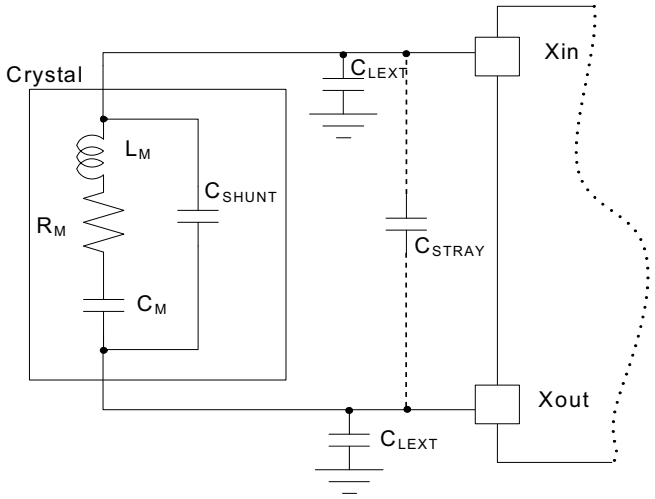
$$C_{LEXT} = 2(C_L - C_{STRAY} - C_{SHUNT})$$

where C_{STRAY} is the capacitance of the pins and PCB, C_{SHUNT} is the shunt capacitance of the crystal.

Table 32-32. Crystal Oscillator Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{OUT}	Crystal oscillator frequency		0.4		32	MHz
ESR	Crystal Equivalent Series Resistance Safety Factor = 3 The AGC doesn't have any noticeable impact on these measurements.	f = 0.455MHz, C_{LEXT} = 100pF XOSC.GAIN = 0	-	-	?	Ω
		f = 2MHz, C_{LEXT} = 20pF XOSC.GAIN = 0	-	-	4.8K	
		f = 4MHz, C_{LEXT} = 20pF XOSC.GAIN = 1	-	-	2.7K	
		f = 8MHz, C_{LEXT} = 20pF XOSC.GAIN = 2	-	-	0.34K	
		f = 16MHz, C_{LEXT} = 20pF XOSC.GAIN = 3	-	-	0.34K	
		f = 32MHz, C_{LEXT} = 18pF XOSC.GAIN = 4	-	-	0.28K	
C_{XIN}	Parasitic capacitor load			5.85		pF
C_{XOUT}	Parasitic capacitor load			3.11		pF
$t_{STARTUP}$	Startup time	f = 2MHz,XOSC.GAIN = 0	-	14K	48K	cycles
		f = 4MHz,XOSC.GAIN = 1	-	5600		
		f = 8MHz, XOSC.GAIN = 2	-	7300		
		f = 16MHz, XOSC.GAIN = 3	-	7500		
		f = 32MHz, XOSC.GAIN = 4	-	5K	10K	

Figure 32-4. Oscillator Connection



32.10.2 External 32 kHz Crystal Oscillator (XOSC32K) Characteristics

32.10.2.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN32 pin.

Table 32-33. Digital Clock Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
$f_{CPXIN32}$	XIN32 clock frequency			32.768		kHz
	XIN32 clock duty cycle			50		%

32.10.2.2 Crystal Oscillator Characteristics

Figure 32-4 and the equation in “Crystal Oscillator Characteristics” on page 576 also applies to the 32 kHz 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 be found in the crystal datasheet.

Table 32-34. 32 kHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{OUT}	Crystal oscillator frequency			32768		Hz
$t_{STARTUP}$	Startup time	R_m = 100kΩ, C_L = 12.5pF		28K	30K	cycles
C_L	Crystal load capacitance		-	-	12.5	pF
C_{SHUNT}	Crystal shunt capacitance			0.1		
C_{XIN32}	Parasitic capacitor load	TQFP64/48/32 packages		3.05		
C_{XOUT32}	Parasitic capacitor load			3.29		
$I_{XOSC32K}$	Current consumption		-	1.2	2.05	uA
ESR_{XTAL}	Crystal equivalent series resistance f=32.768kHz Safety Factor = 3	C_L=12.5pF (Amplitude control of gain = 63)			348	kΩ

32.10.3 Digital Frequency Locked Loop (DFLL) Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{OUT}	Output frequency	Open loop after calibration at room temperature	47	48	49	MHz
f_{REF}	Reference frequency			32.768		kHz
	Maximum fine step size(1)	Open loop			0.15	%
	Maximum coarse step size(1)	Open loop			2.5	
I_{DFLL}	Power consumption on V_{DDANA}(1)	Open loop, Coarse calibrated against 48MHz, FINE=128		140		µA
$t_{STARTUP}$	Startup time(1)	Open loop after calibration against 48MHz f_{OUT} within 90% of final value		6.1		µs
$t_{LCOARSE}$	Coarse lock time(1)	Quick lock enabled, Chill cycle disabled,CSTEP=3, f_{REF} = 32.768kHz		260		
t_{LFINE}	Fine lock time(1)	Quick lock disabled, Chill cycle disabled, CSTEP=3,FSTEP=1, f_{REF} = 32.768kHz		700		

Note: 1. These values are based on simulation. These values are not covered by test limits in production or characterization

32.10.4 32.768kHz Internal oscillator (OSC32K) Characteristics

Table 32-35. 32 kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{OUT}	Output frequency	Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V	28.508	32.768	34.734	kHz
		Calibrated against a 32.768kHz reference at 25°C, at V_{DD}=3.3V	32.276	32.768	33260	
		Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V	31.457	32.768	34.079	
I_{OSC32K}	Current consumption		-	-	-	µA
$t_{STARTUP}$	Startup time		-	1	2	cycle
Duty	Duty Cycle		-	50	-	%

32.10.5 Ultra Low Power Internal 32kHz RC Oscillator (OSCULP32K) Characteristics

Table 32-36. Ultra Low Power Internal 32 kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{OUT}	Output frequency	Calibrated against a 32.768kHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V	25.559	32.768	38.011	kHz
		Calibrated against a 32.768kHz reference at 25°C, at V_{DD}=3.3V	31.293	32.768	34.570	
		Calibrated against a 32.768kHz reference at 25°C, over [1.62, 3.63]V	31.293	32.768	34.570	
$i_{OSCULP32K}$ (1)(2)			-	-	125	nA
$t_{STARTUP}$	Startup time		-	10	-	cycles
Duty	Duty Cycle		-	50	-	%

Notes: 1. These values are based on simulation. These values are not covered by test limits in production or characterization.
2. This oscillator is always on.

32.10.6 8MHz RC Oscillator (OSC8M) Characteristics

Table 32-37. Internal 8MHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
f_{OUT}	Output frequency	Calibrated against a 8MHz reference at 25°C, over [-40, +85]C, over [1.62, 3.63]V	7.8	8	8.16	MHz
		Calibrated against a 8MHz reference at 25°C, at V_{DD}=3.3V	7.94	8	8.06	
		Calibrated against a 8MHz reference at 25°C, over [1.62, 3.63]V	7.92	8	8.08	
I_{OSC8M}	Current consumption		-	60	74	µA
$t_{STARTUP}$	Startup time		-	2.1	3	µs
Duty	Duty cycle		-	50	-	%

32.11 PTC Typical Characteristics

Figure 32-5. Power consumption [µA].

1 sensor, noise countermeasures disabled, f=48MHz, Vcc=3.3V

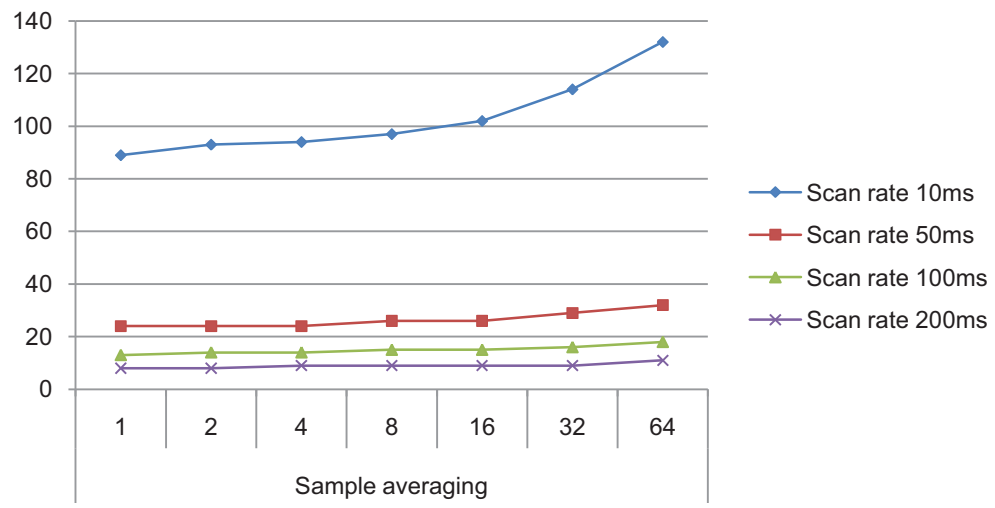


Figure 32-6. Power consumption [µA].

1 sensor, noise countermeasures Enabled, f=48MHz, Vcc=3.3V

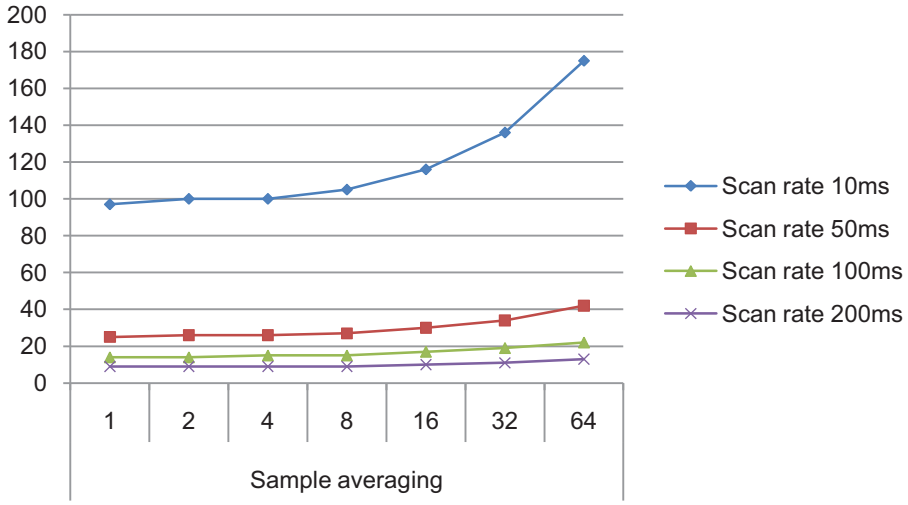


Figure 32-7. Power consumption [µA].

10 sensors, noise countermeasures disabled, f=48MHz, Vcc=3.3V

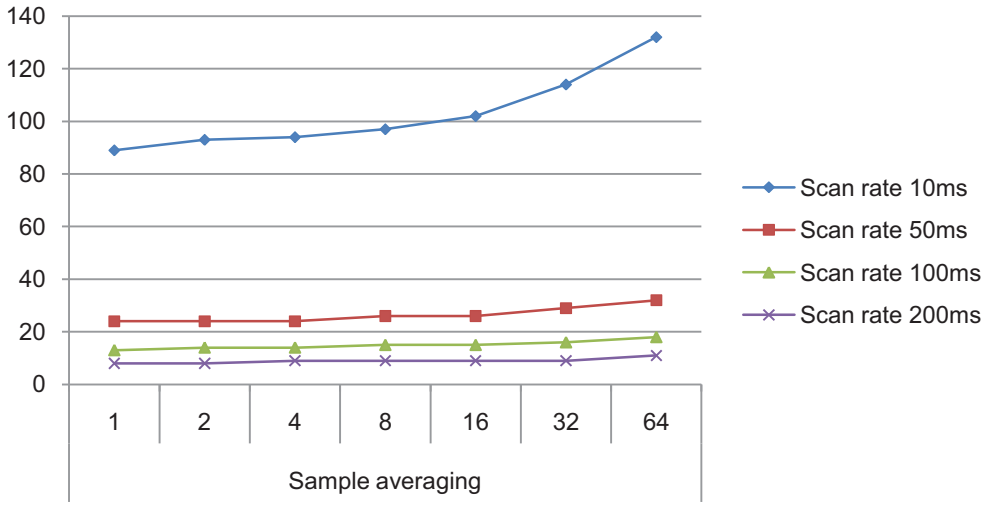


Figure 32-8. Power consumption [µA].

10 sensors, noise countermeasures Enabled, f=48MHz, Vcc=3.3V

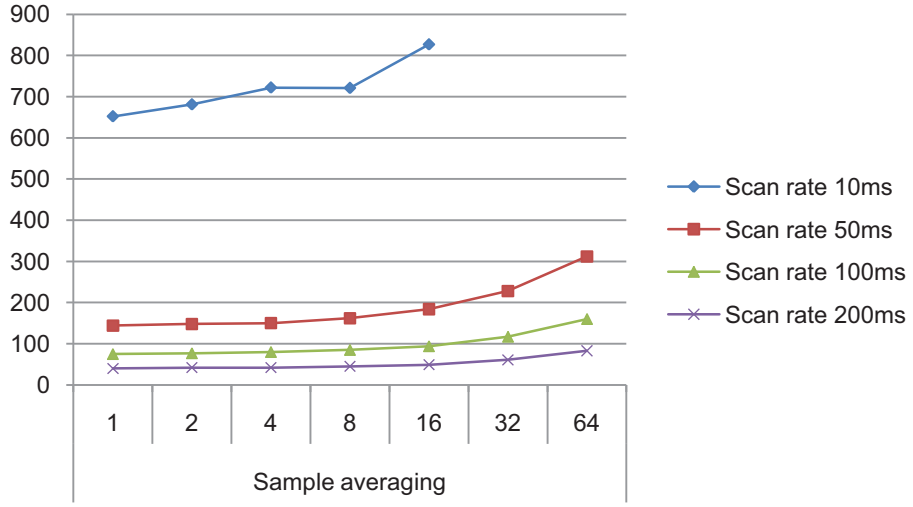


Figure 32-9. Power consumption [µA].

100 sensors, noise countermeasures disabled, f=48MHz, Vcc=3.3V

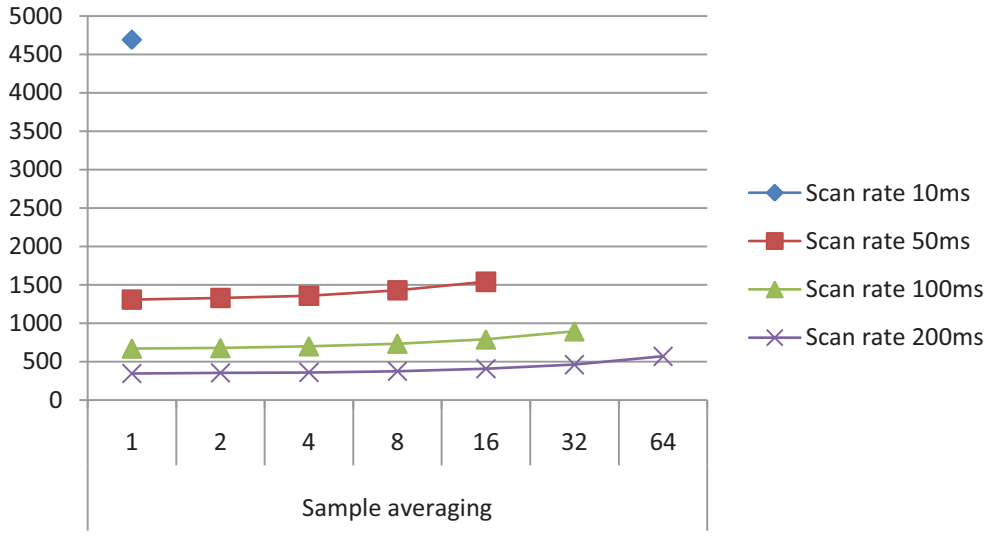


Figure 32-10.Power consumption [µA].

100 sensors, noise countermeasures Enabled, f=48MHz, Vcc=3.3V

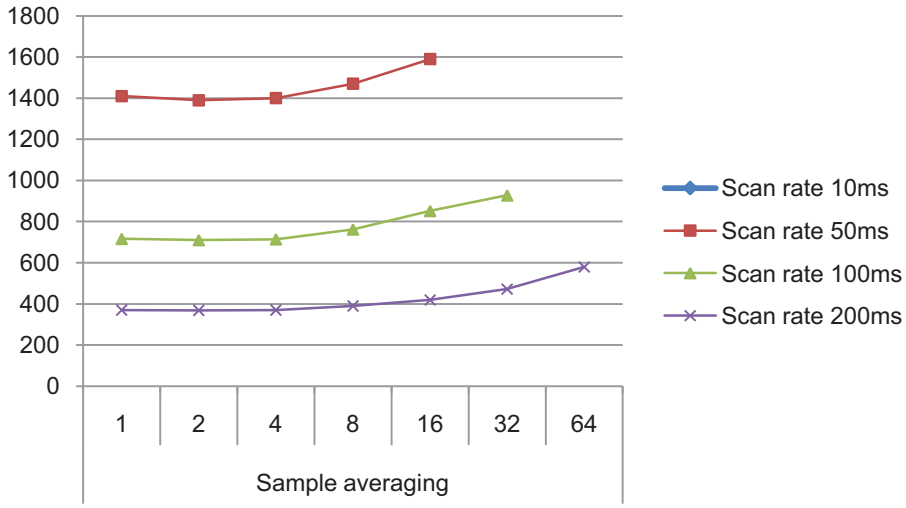
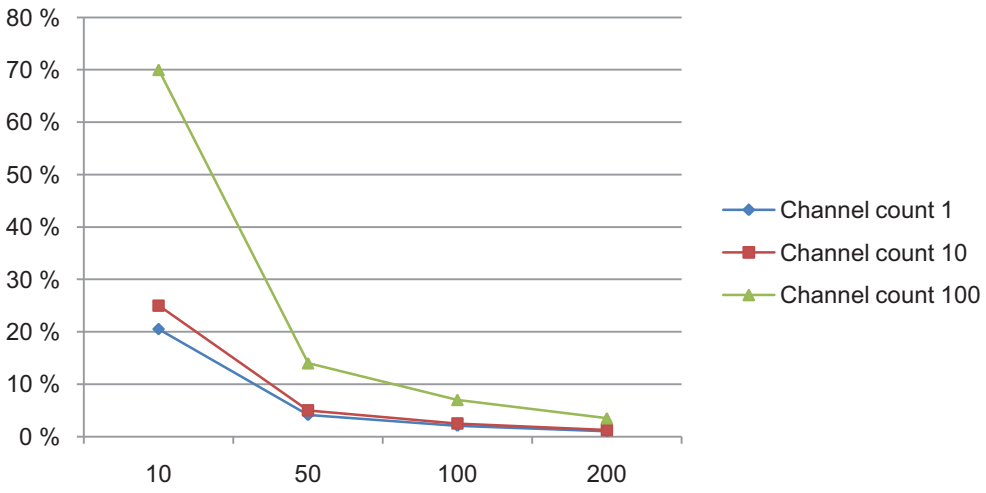


Figure 32-11.CPU utilization.



32.12 Timing Characteristics

32.12.1 External Reset

Table 32-38. External reset characteristics

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
t_{EXT}	Minimum reset pulse width		10			ns

32.12.2 SERCOM in SPI Mode Timing

Figure 32-12.SPI timing requirements in master mode

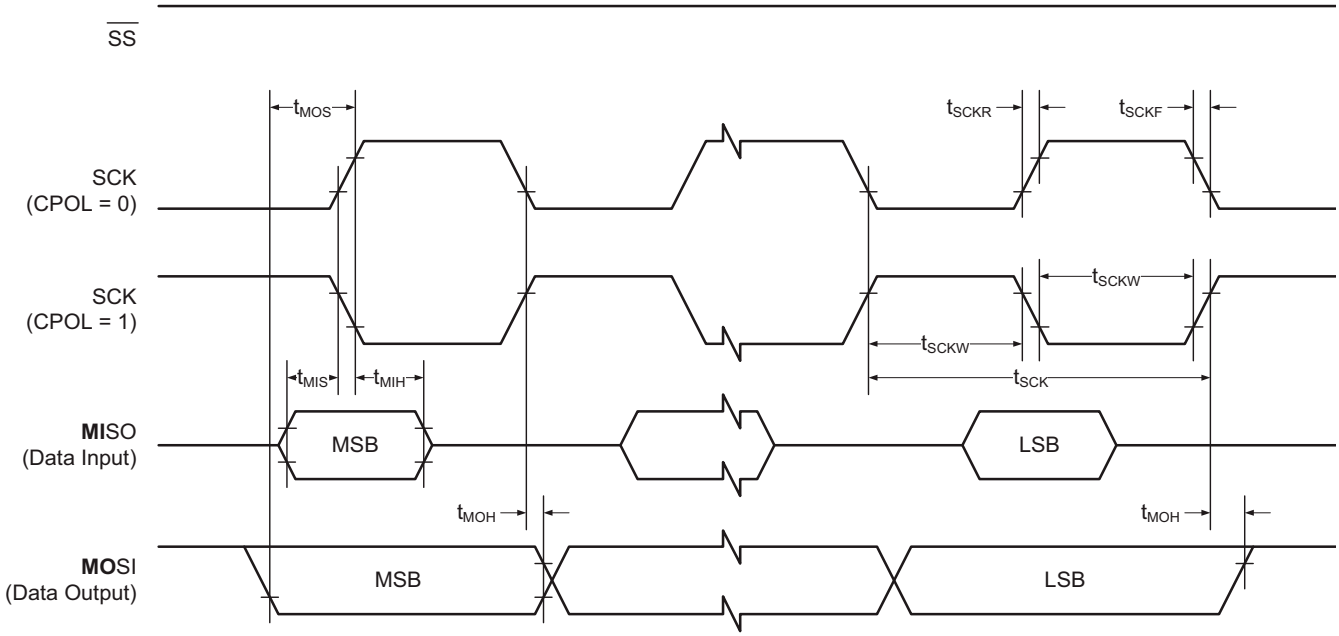


Figure 32-13.SPI timing requirements in slave mode

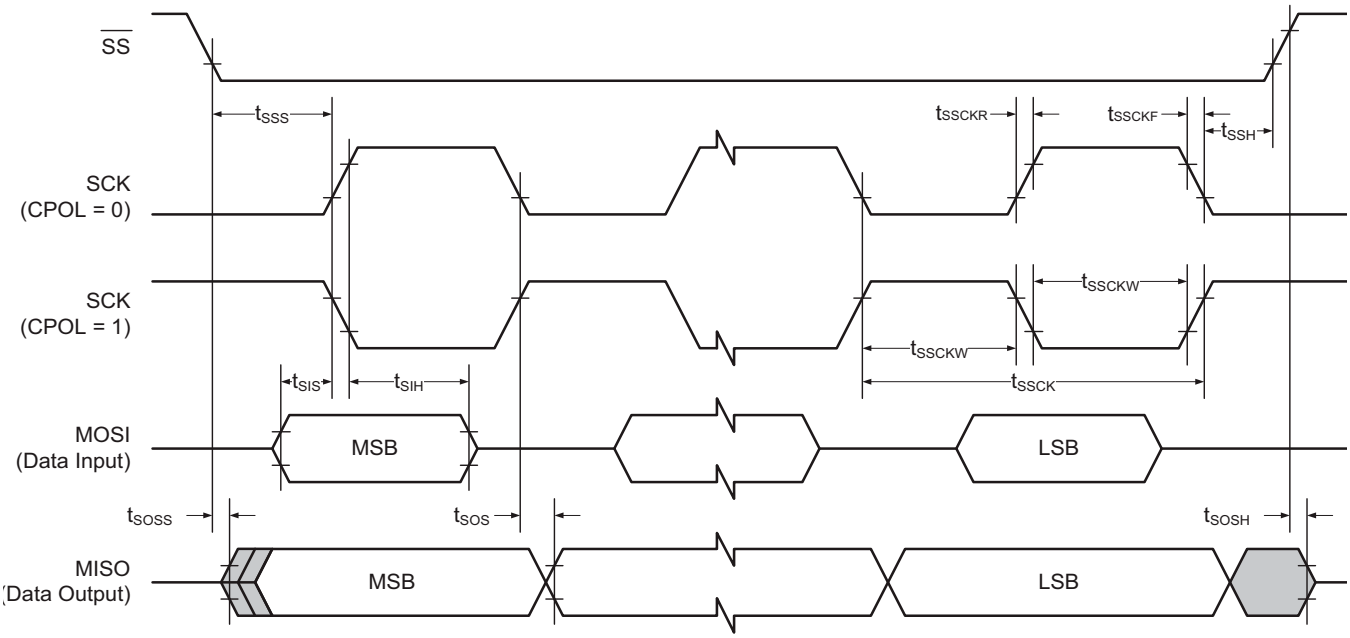


Table 32-39. SPI timing characteristics and requirements

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
t_{SCK}	SCK period	Master				ns
t_{SCKW}	SCK high/low width	Master				
t_{SCKR}	SCK rise time	Master				
t_{SCKF}	SCK fall time	Master				
t_{MIS}	MISO setup to SCK	Master				
t_{MIH}	MISO hold after SCK	Master				
t_{MOS}	MOSI setup SCK	Master				
t_{MOH}	MOSI hold after SCK	Master				
t_{SSCK}	Slave SCK Period	Slave				
t_{SSCKW}	SCK high/low width	Slave				
t_{SSCKR}	SCK rise time	Slave				
t_{SSCKF}	SCK fall time	Slave				
t_{SIS}	MOSI setup to SCK	Slave				
t_{SIH}	MOSI hold after SCK	Slave				
t_{SSS}	$\overline{SS}$ setup to SCK	Slave				
t_{SSH}	$\overline{SS}$ hold after SCK	Slave				
t_{SOS}	MISO setup SCK	Slave				
t_{SOH}	MISO hold after SCK	Slave				
t_{SOSS}	MISO setup after $\overline{SS}$ low	Slave				
t_{SOSH}	MISO hold after $\overline{SS}$ high	Slave				

32.12.3 SERCOM in I²C Mode Timing

Table 32-40 describes the requirements for devices connected to the I²C Interface Bus. Timing symbols refer to Figure 32-14.

Figure 32-14. I²C Interface Bus Timing

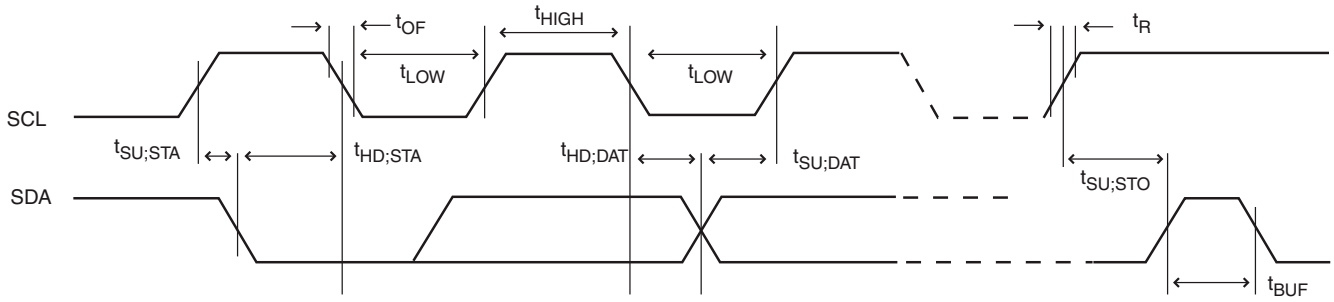


Table 32-40. I²C Interface Timing[1]

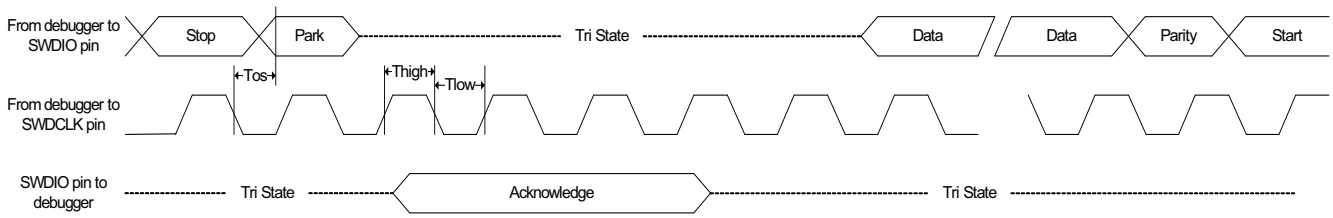
Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
t_R	Rise time for both SDA and SCL				300	ns
t_{OF}	Output fall time from V_{IHmin} to V_{ILmax}	10pF < C_b < 400pF	7.0	10.0	50.0	ns
$t_{HD;STA}$	Hold time (repeated) START condition	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
t_{LOW}	Low period of SCL Clock	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
t_{HIGH}	High period of SCL Clock	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
$t_{SU;STA}$	Setup time for a repeated START condition	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
$t_{HD;DAT}$	Data hold time	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
$t_{SU;DAT}$	Data setup time	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
$t_{SU;STO}$	Setup time for STOP condition	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				
t_{BUF}	Bus free time between a STOP and a START condition	$f_{SCL} \leq$ 100kHz				µs
		f_{SCL} > 100kHz				

Note: 1. These values are based on characterization. These values are not covered by test limits in production.

32.12.4 SWD Timing

Figure 32-15.SWD Interface Signals

Read Cycle



Write Cycle

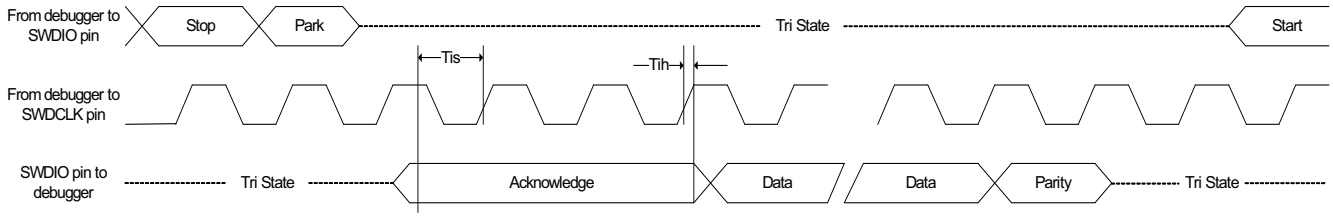


Table 32-41. SWD Timings

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Units
Thigh	SWDCLK High period	V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF				ns
Tlow	SWDCLK Low period					
Tos	SWDIO output skew to falling edge SWDCLK					
Tis	Input Setup time required between SWDIO					
Tih	Input Hold time required between SWDIO and rising edge SWDCLK					

33. Packaging Information

33.1 Thermal Considerations

33.1.1 Thermal Resistance Data

Table 33-1 summarizes the thermal resistance data depending on the package.

Table 33-1. Thermal Resistance Data

Package Type	θ_{JA}	θ_{JC}
32-pin TQFP	68°C/W	25.8°C/W
48-pin TQFP	78.8°C/W	12.3°C/W
64-pin TQFP	66.7°C/W	11.9°C/W
32-pin QFN	37.2°C/W	3.1°C/W
48-pin QFN	33°C/W	11.4°C/W
64-pin QFN	33.5°C/W	11.2°C/W

33.1.2 Junction Temperature

The average chip-junction temperature, T_J, in °C can be obtained from the following equations:

Equation 1

$$T_J = T_A + (P_D \times \theta_{JA})$$

Equation 2

$$T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$$

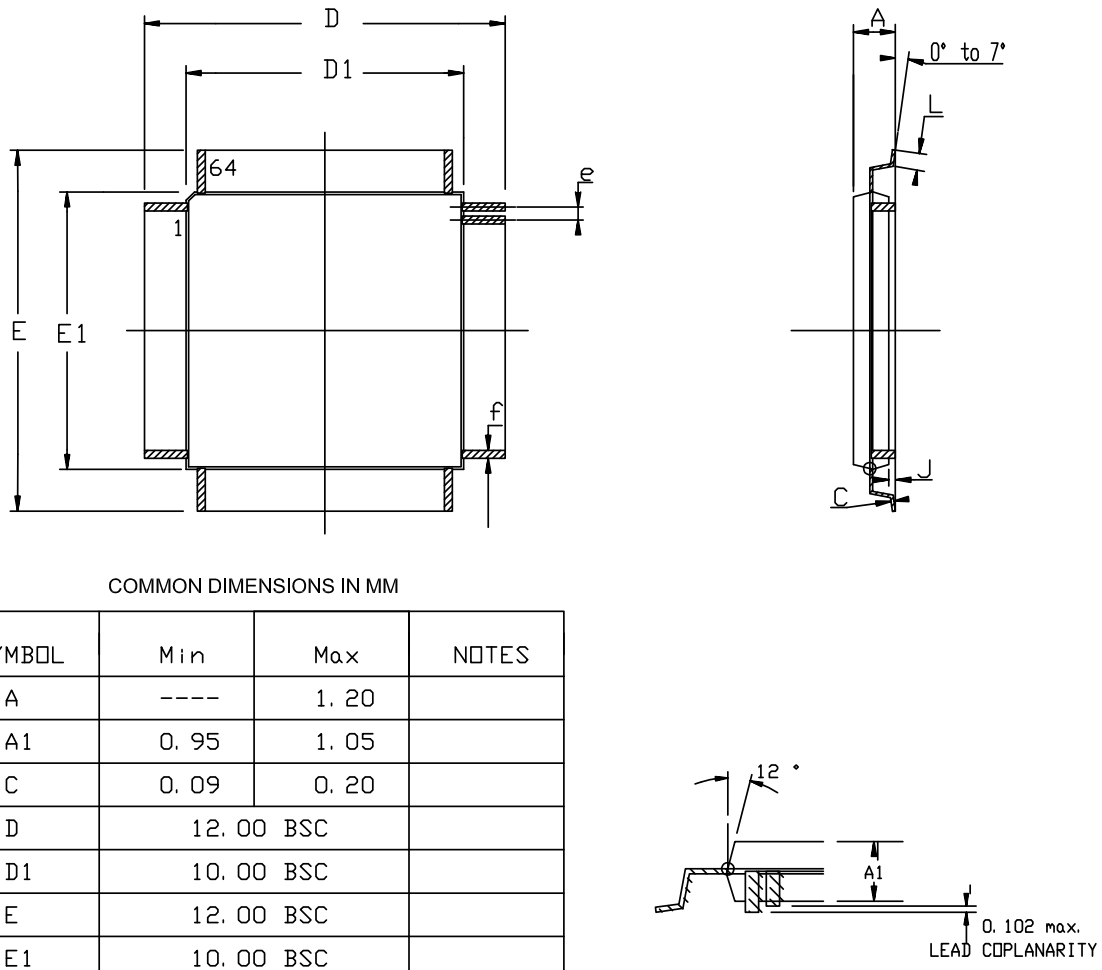
where:

- θ_{JA} = package thermal resistance, Junction-to-ambient (°C/W), provided in Table 33-1
- θ_{JC} = package thermal resistance, Junction-to-case thermal resistance (°C/W), provided in Table 33-1
- $\theta_{HEATSINK}$ = cooling device thermal resistance (°C/W), provided in the manufacturer datasheet
- P_D = device power consumption (W)
- T_A = ambient temperature (°C)

From “Equation 1” , 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, “Equation 2” should be used to compute the resulting average chip-junction temperature T_J in °C.

33.2 Package Drawings

33.2.1 64-pin TQFP



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 33-2. Device and Package Maximum Weight

300	mg

Table 33-3. Package Characteristics

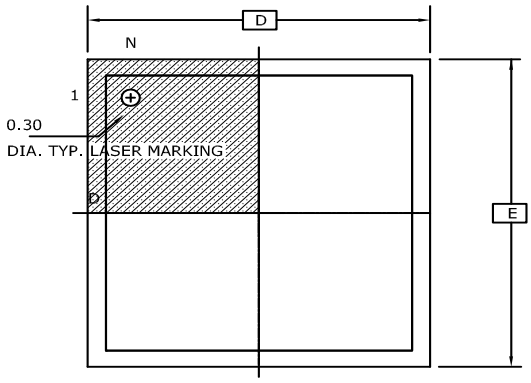
Moisture Sensitivity Level	MSL3

Table 33-4. Package Reference

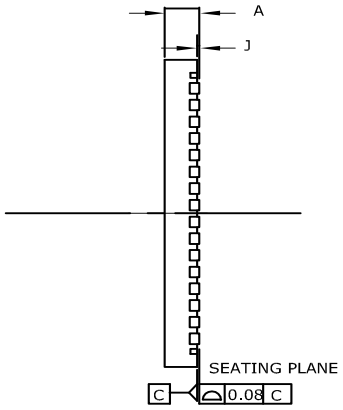
JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

33.2.2 64-pin QFN

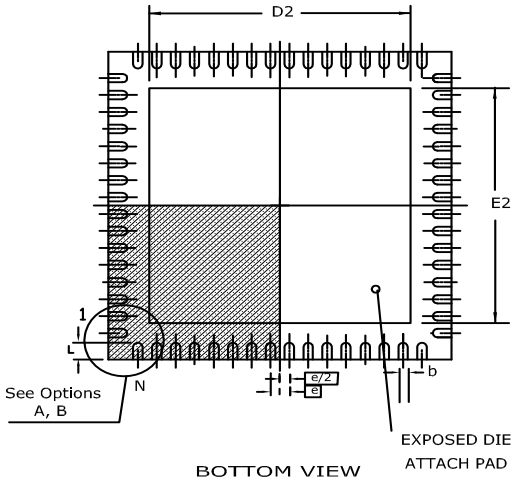
DRAWINGS NOT SCALED



TOP VIEW



SIDE VIEW

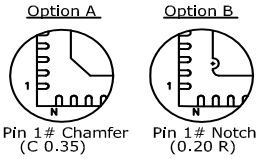


BOTTOM VIEW

COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	-----	1.00	
D/E		9.00 BSC		
D2/E2	4.60	4.70	4.80	
J	0.00	-----	0.05	
b	0.15	0.20	0.25	
e		0.50 BSC		
L	0.30	0.40	0.55	
N		64		



Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MO-220, Variation VMMD-4, for proper dimensions, tolerances, datums, etc.
2. Dimension b applies to metallized terminal and is measured between 0.15mm and 0.30mm from the terminal tip.
If the terminal has the optical radius on the other end of the terminal, the dimension should not be measured in that radius area.

Table 33-5. Device and Package Maximum Weight

200	mg

Table 33-6. Package Characteristics

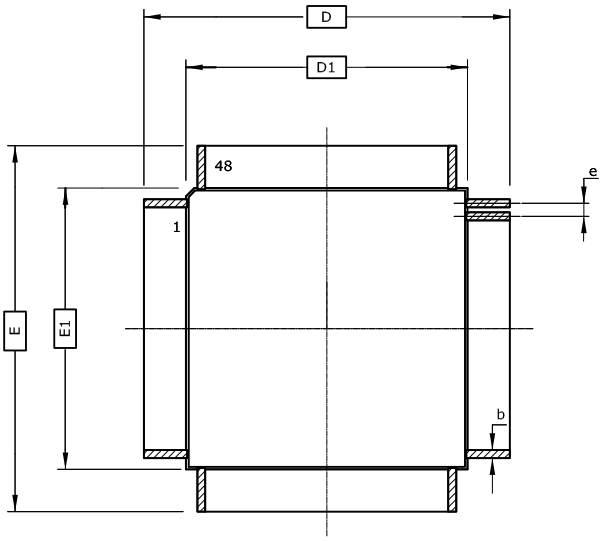
Moisture Sensitivity Level	MSL3

Table 33-7. Package Reference

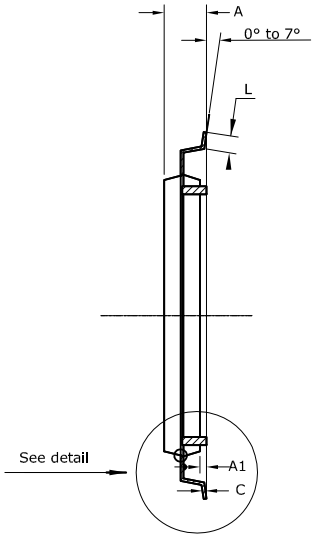
JEDEC Drawing Reference	MO-220
JESD97 Classification	E3

33.2.3 48-pin TQFP

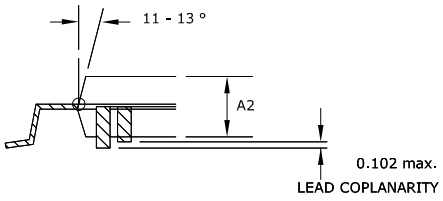
DRAWINGS NOT SCALED



TOP VIEW



SIDE VIEW



DETAIL VIEW

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			

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MS-026, Variation ABC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10mm maximum.

Table 33-8. Device and Package Maximum Weight

140	mg

Table 33-9. Package Characteristics

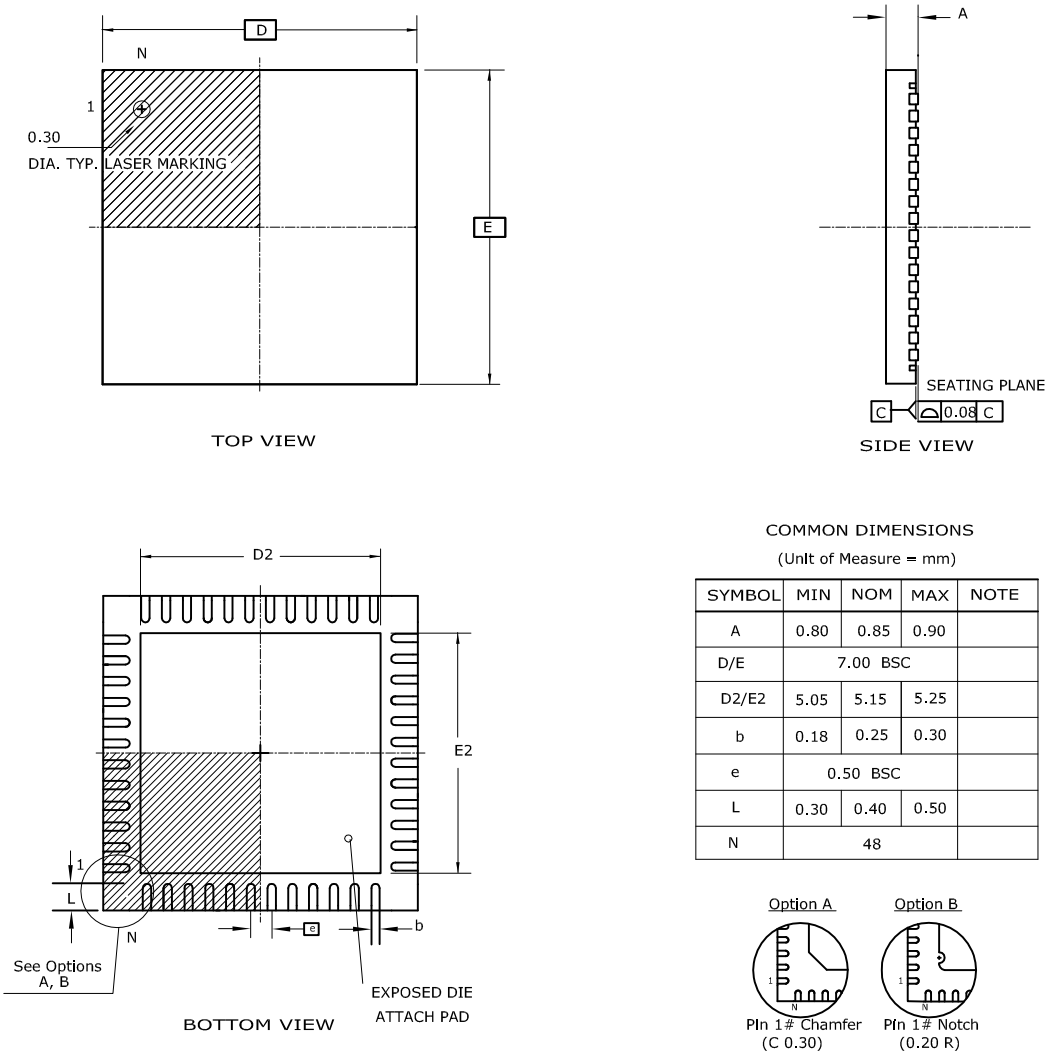
Moisture Sensitivity Level	MSL3

Table 33-10. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

33.2.4 48-pin QFN

DRAWINGS NOT SCALED



COMMON DIMENSIONS

(Unit of Measure = mm)

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		

Notes : 1. This drawing is for general information only. Refer to JEDEC Drawing MO-220, Variation VKKD-4, for proper dimensions, tolerances, datums, etc.
2. Dimension b applies to metallized terminal and is measured between 0.15mm and 0.30mm from the terminal tip.
If the terminal has the optical radius on the other end of the terminal, the dimension should not be measured in that radius area.

Table 33-11. Device and Package Maximum Weight

140	mg

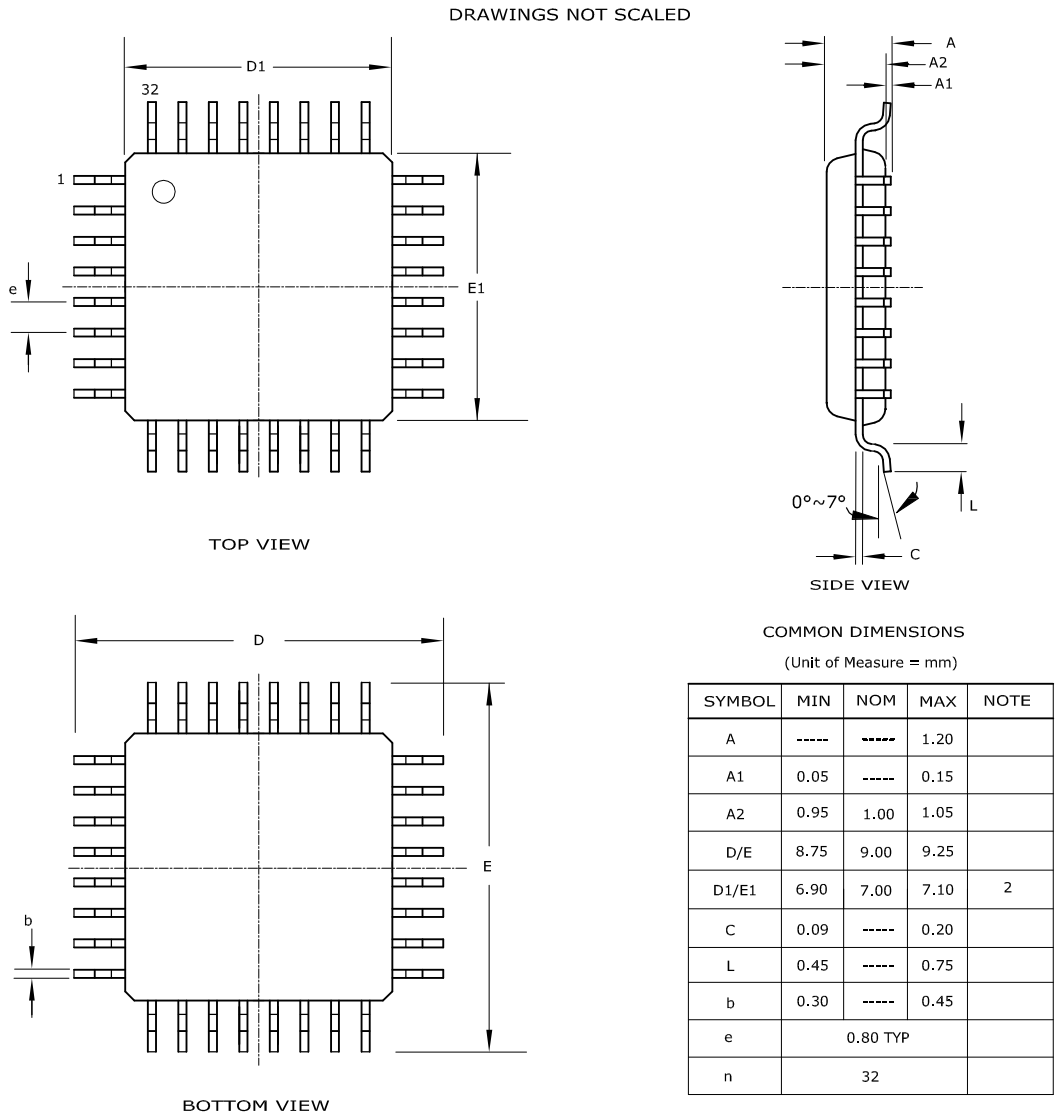
Table 33-12. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 33-13. Package Reference

JEDEC Drawing Reference	MO-220
JESD97 Classification	E3

33.2.5 32-pin TQFP



COMMON DIMENSIONS

(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	-----	-----	1.20	
A1	0.05	-----	0.15	
A2	0.95	1.00	1.05	
D/E	8.75	9.00	9.25	
D1/E1	6.90	7.00	7.10	2
C	0.09	-----	0.20	
L	0.45	-----	0.75	
b	0.30	-----	0.45	
e	0.80 TYP			
n	32			

Notes :
1. This drawing is for general information only. Refer to JEDEC Drawing MS-026, Variation ABA.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10mm maximum.

Table 33-14. Device and Package Maximum Weight

TBD	mg

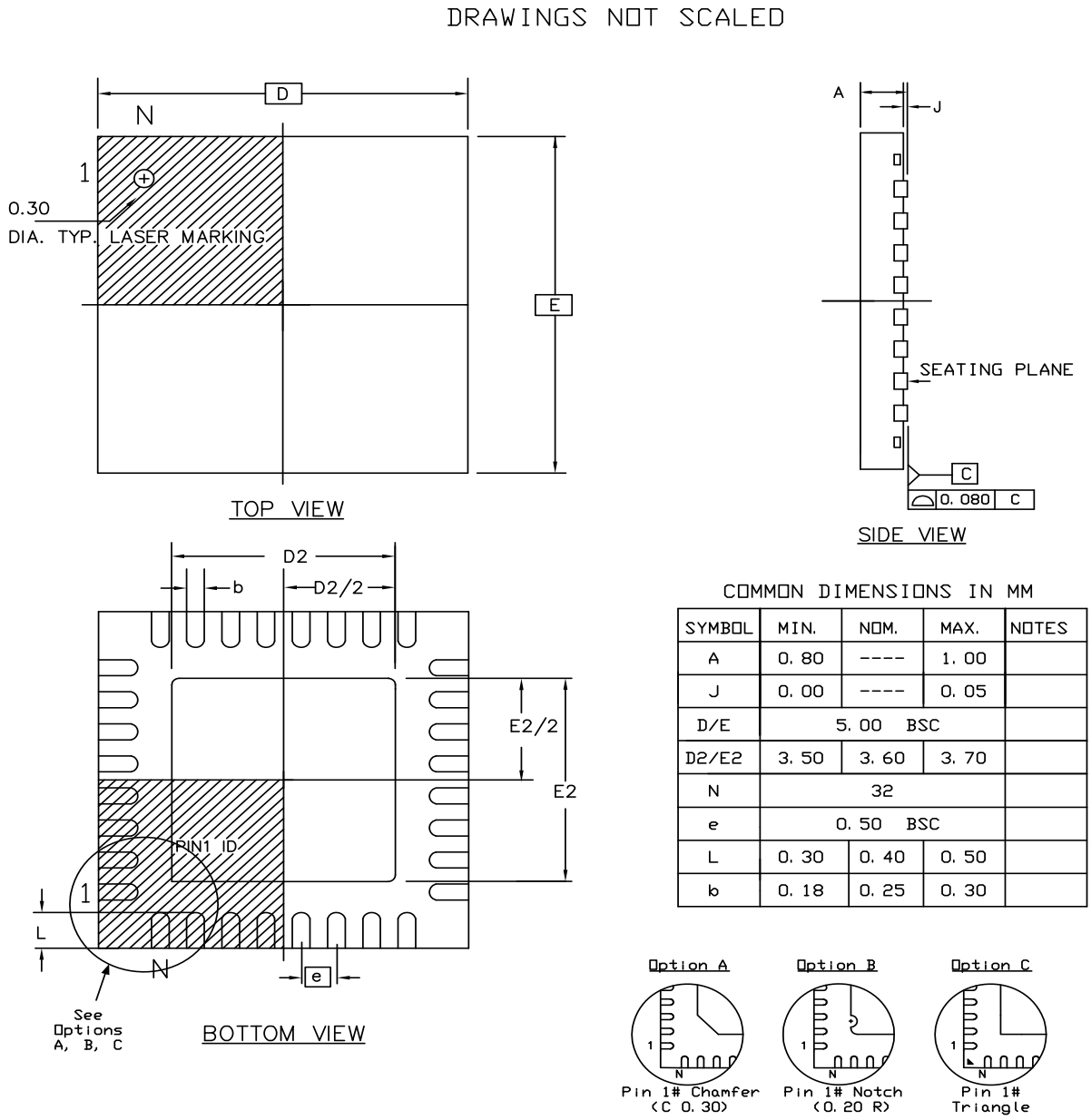
Table 33-15. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 33-16. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

33.2.6 32-pin QFN



SYMBOL	MIN.	NOM.	MAX.	NOTES
A	0.80	----	1.00	
J	0.00	----	0.05	
D/E		5.00 BSC		
D2/E2	3.50	3.60	3.70	
N		32		
e		0.50 BSC		
L	0.30	0.40	0.50	
b	0.18	0.25	0.30	

Table 33-17. Device and Package Maximum Weight

TBD	mg

Table 33-18. Package Characteristics

Moisture Sensitivity Level	MSL3

Table 33-19. Package Reference

JEDEC Drawing Reference	MO-220
JESD97 Classification	E3

33.3 Soldering Profile

Table Table 33-20 gives the recommended soldering profile from J-STD-20.

Table 33-20. 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	60-150s
Time within 5°C of Actual Peak Temperature	30s
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.

34. Schematic Checklist

34.1 Introduction

This chapter describes a common checklist which should be used when starting and reviewing the schematics for a SAM D20 design. This chapter illustrates a recommended power supply connection, how to connect external analog references, programmer, debugger, oscillator and crystal.

34.2 Power Supply

The SAM D20 supports a single power supply from 1.62 to 3.63V.

34.2.1 Power Supply Connections

Figure 34-1. Power Supply Schematic

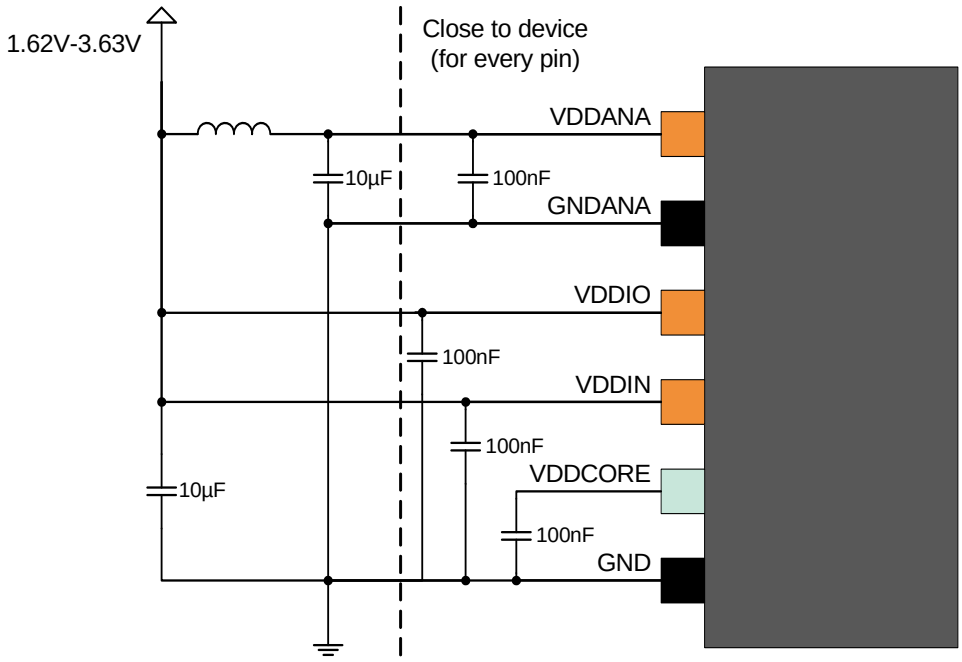


Table 34-1. Power Supply Connections, V_{DDCORE} From Internal Regulator

Signal Name	Recommended Pin Connection	Description
V_{DDIO}	1.6V to 3.6V Decoupling/filtering capacitors 100nF(1)(2) and 10µF(1) Decoupling/filtering inductor 10µH(1)(3)	Digital supply voltage
V_{DDANA}	1.6V to 3.6V Decoupling/filtering capacitors 100nF(1)(2) and 10µF(1) Ferrite bead(4) prevents the V_{DD} noise interfering the V_{DDANA}	Analog supply voltage

Table 34-1. Power Supply Connections, V_{DDCORE} From Internal Regulator (Continued)

Signal Name	Recommended Pin Connection	Description
V_{DDCORE}	1.6V to 1.8V Decoupling/filtering capacitor 100nF(1)(2)	Core supply voltage / external decoupling pin
GND		Ground
GND_{ANA}		Ground for the analog power domain

Notes:
1. These values are only given as typical examples.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group, low ESR caps should be used for better decoupling.
3. An inductor should be added between the external power and the V_{DD} for power filtering.
4. Ferrite bead has better filtering performance than the common inductor at high frequencies. It can be added between V_{DD} and V_{DDANA} for preventing digital noise from entering the analog power domain. The bead should provide enough impedance (e.g. 50Ω at 20MHz and 220Ω at 100MHz) for separating the digital power from the analog power domain. Make sure to select a ferrite bead designed for filtering applications with a low DC resistance to avoid a large voltage drop across the ferrite bead.

34.3 External Analog Reference Connections

The following schematic checklist is only necessary if the application is using one or more of the external analog references. If the internal references are used instead, the following circuits in Figure 34-2 and Figure 34-3 are not necessary.

Figure 34-2. External Analog Reference Schematic With Two References

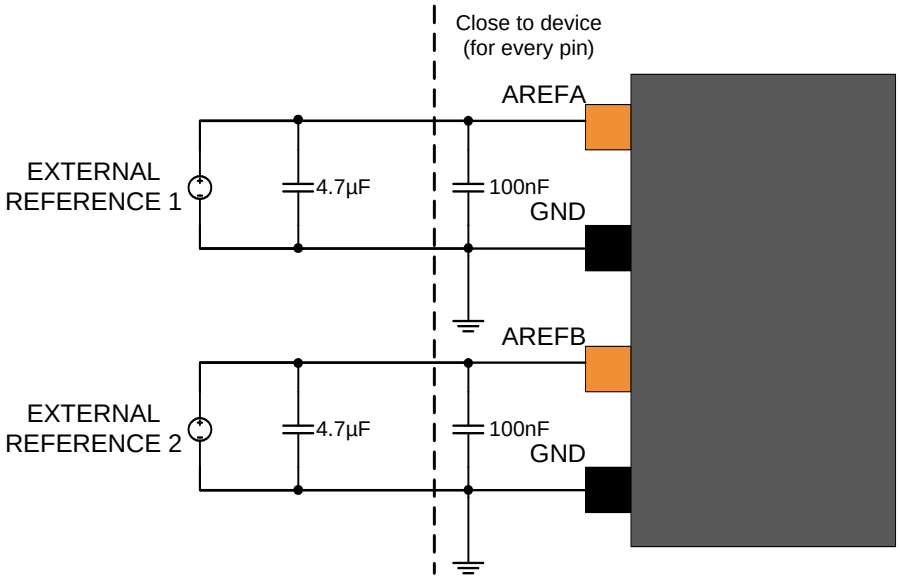


Figure 34-3. External Analog Reference Schematic With One Reference

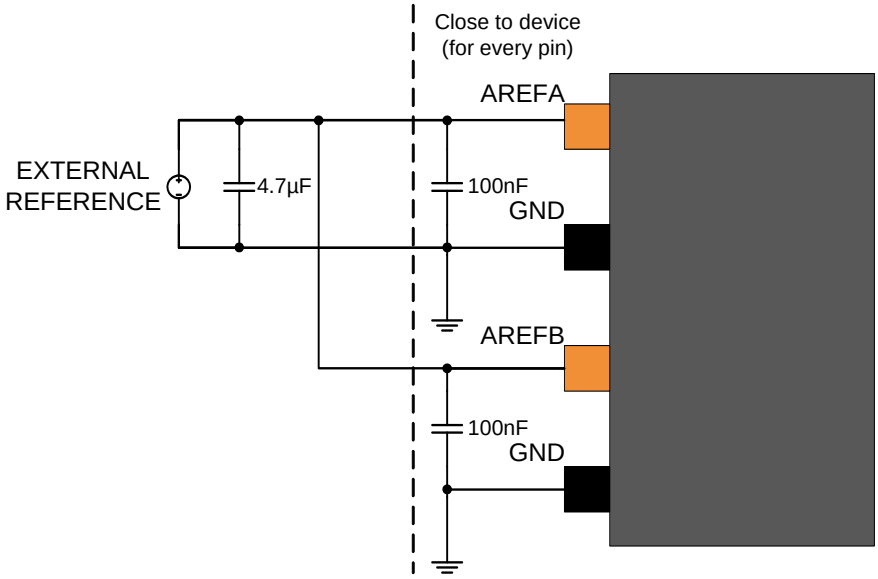


Table 34-2. External Analog Reference Connections

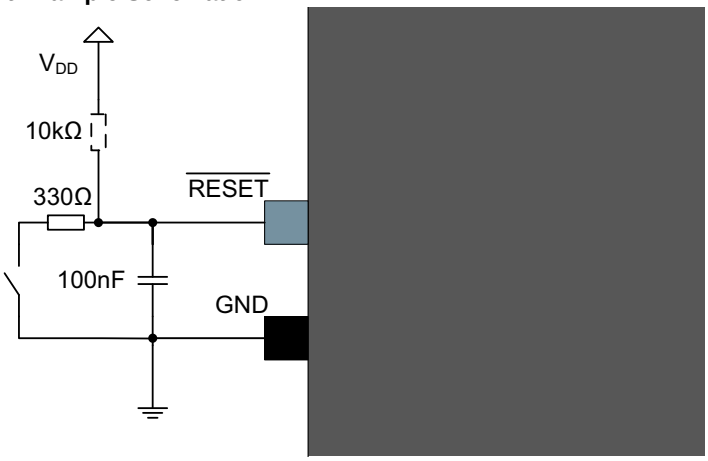
Signal Name	Recommended Pin Connection	Description
AREFx	1.0V to V_{DDANA} - 0.6V for ADC 1.0V to V_{DDANA} - 0.6V for DAC Decoupling/filtering capacitors 100nF[1][2] and 4.7µF[1]	External reference from AREFx pin on the analog port
GND		Ground

Notes: 1. These values are given as a typical example.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

34.4 External Reset Circuit

The external reset circuit is connected to the $\overline{\text{RESET}}$ pin when the external reset function is used. If the external reset function has been disabled, the circuit is not necessary. The reset switch can also be removed, if the manual reset is not necessary. The $\overline{\text{RESET}}$ pin itself has an internal pull-up resistor, hence it is optional to also add an external pull-up resistor.

Figure 34-4. External Reset Circuit Example Schematic



A pull-up resistor makes sure that the reset does not go low unintended causing a device reset. An additional resistor has been added in series with the switch to safely discharge the filtering capacitor, i.e. preventing a current surge when shorting the filtering capacitor which again causes a noise spike that can have a negative effect on the system.

Table 34-3. Reset Circuit Connections

Signal Name	Recommended Pin Connection	Description
$\overline{\text{RESET}}$	Reset low level threshold voltage V_{DDIO} = 1.6V - 2.0V: Below 0.33 * V_{DDIO} V_{DDIO} = 2.7V - 3.6V: Below 0.36 * V_{DDIO} Decoupling/filter capacitor 100nF(1) Pull-up resistor 10kΩ(1)(2) Resistor in series with the switch 330Ω(1)	Reset pin

Notes: 1. These values are given as a typical example.
2. The SAM D20 features an internal pull-up resistor on the $\overline{\text{RESET}}$ pin, hence an external pull-up is optional.

34.5 Unused or Unconnected Pins

Unused or unconnected pins (unless marked as NC where applicable) should not be left unconnected and floating. Floating pins will add to the overall power consumption of the device. To prevent this one should always draw the pin voltage towards a given level, either V_{DD} or GND, through a pull up/down resistor. External or internal pull up/down resistors can be used, e.g. the pins can be configured in pull-up or pull-down mode eliminating the need for external components, for more information see “PORT” on page 276 for details. There are no obvious benefit in choosing external vs. internal pull resistors.

34.6 Clocks and Crystal Oscillators

The SAM D20 can be run from internal or external clock sources, or a mix of internal and external sources. An example of usage will be to use the internal 8MHz oscillator as source for the system clock, and an external 32.768kHz watch crystal as clock source for the Real-Time counter (RTC).

34.6.1 External Clock Source

Figure 34-5. External Clock Source Example Schematic

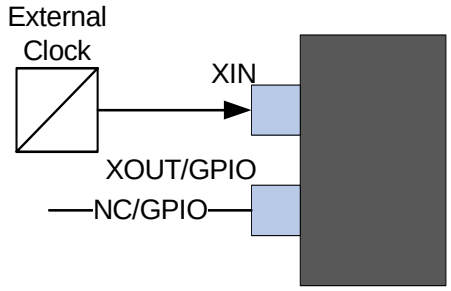
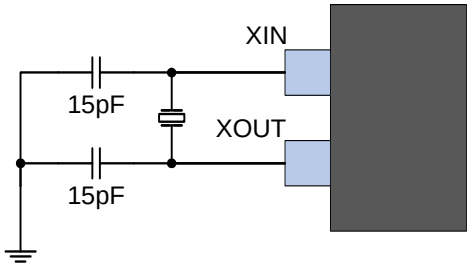


Table 34-4. External Clock Source Connections

Signal Name	Recommended Pin Connection	Description
XIN	XIN is used as input for an external clock signal	Input for inverting oscillator pin
XOUT/GPIO	Can be left unconnected or used as normal GPIO	

34.6.2 Crystal Oscillator

Figure 34-6. Crystal Oscillator Example Schematic



The crystal should be located as close to the device as possible. Long signal lines may cause too high load to operate the crystal, and cause crosstalk to other parts of the system.

Table 34-5. Crystal Oscillator Checklist

Signal Name	Recommended Pin Connection	Description
XIN	Load capacitor 15pF(1)(2)	External crystal between 0.4 to 30MHz
XOUT	Load capacitor 15pF(1)(2)	

Notes: 1. These values are given only as typical example.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

34.6.3 External Real Time Oscillator

The low frequency crystal oscillator is optimized for use with a 32.768kHz watch crystal. When selecting crystals, load capacitance and crystal's Equivalent Series Resistance (ESR) must be taken into consideration. Both values are specified by the crystal vendor.

SAM D20 oscillator is optimized for very low power consumption, hence close attention should be made when selecting crystals, see Table 34-6 for maximum ESR recommendations on 9pF and 12.5pF crystals.

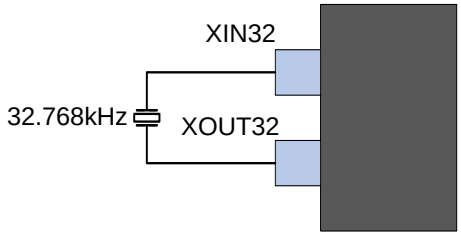
The Low-frequency Crystal Oscillator provides an internal load capacitance of typical 3.05pF and 3.29pF. Crystals with recommended 12.5pF load capacitance can be without external capacitors as shown in Figure 34-7.

Table 34-6. Maximum ESR Recommendation for 32.768kHz Crystal

Crystal C_L (pF)	Max ESR [kΩ]
12.5	313

Note: Maximum ESR is typical value based on characterization. These values are not covered by test limits in production.

Figure 34-7. External Real Time Oscillator without Load Capacitor



Crystals specifying load capacitance (CL) higher than 12.5pF, require external capacitors applied as described in Figure 34-8.

To find suitable load capacitance for a 32.768kHz crystal, consult the crystal datasheet.

Figure 34-8. External Real Time Oscillator with Load Capacitor

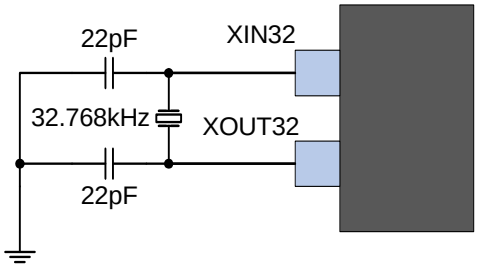


Table 34-7. External Real Time Oscillator Checklist

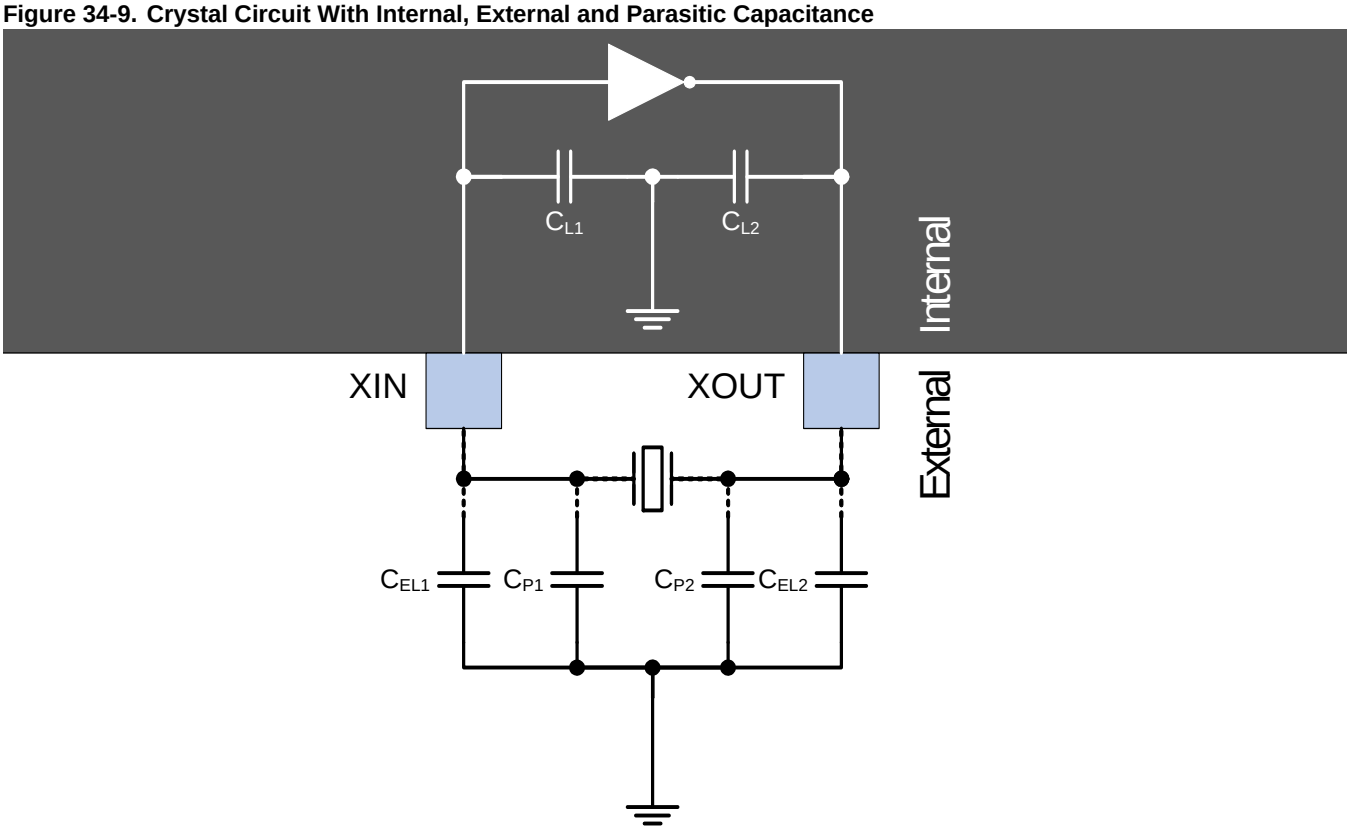
Signal Name	Recommended Pin Connection	Description
XIN32	Load capacitor 22pF(1)(2)	Timer oscillator input
XOUT32	Load capacitor 22pF(1)(2)	Timer oscillator output

Notes: 1. These values are given only as typical examples.
2. Decoupling capacitor should be placed close to the device for each supply pin pair in the signal group.

34.6.4 Calculating the Correct Crystal Decoupling Capacitor

In order to calculate correct load capacitor for a given crystal one can use the model shown in Figure 34-9 which includes internal capacitors C_{Ln}, external parasitic capacitance C_{ELn} and external load capacitance C_{Pn}.

Figure 34-9. Crystal Circuit With Internal, External and Parasitic Capacitance



Using this model the total capacitive load for the crystal can be calculated as shown in the equation below:

$$\sum C_{tot} = \frac{(C_{L1} + C_{P1} + C_{EL1})(C_{L2} + C_{P2} + C_{EL2})}{C_{L1} + C_{P1} + C_{EL1} + C_{L2} + C_{P2} + C_{EL2}}$$

where C_{tot} is the total load capacitance seen by the crystal, this value should be equal to the load capacitance value found in the crystal manufacturer datasheet.

The parasitic capacitance C_{ELn} can in most applications be disregarded as these are usually very small. If accounted for the value is dependent on the PCB material and PCB layout.

For some crystal the internal capacitive load provided by the device itself can be enough. To calculate the total load capacitance in this case. C_{ELn} and C_{Pn} are both zero, $C_{L1} = C_{L2} = C_L$, and the equation reduces to the following:

$$\sum C_{tot} = \frac{C_L}{2}$$

Table 34-8 shows the device equivalent internal pin capacitance.

Table 34-8. Equivalent Internal Pin Capacitance

Symbol	Value	Description
C_{XIN32}	3.05pF	Equivalent internal pin capacitance
C_{XOUT32}	3.29pF	Equivalent internal pin capacitance

34.7 Programming and Debug Ports

For programming and/or debugging the SAM D20 the device should be connected using the Serial Wire Debug, SWD, interface. Currently the SWD interface is supported by several Atmel and third party programmers and debuggers, like the SAM-ICE, JTAGICE3 or SAM D20 Xplained Pro (SAM D20 evaluation kit) Embedded Debugger.

Refer to the SAM-ICE, JTAGICE3 or SAM D20 Xplained Pro user guides for details on debugging and programming connections and options. For connecting to any other programming or debugging tool, refer to that specific programmer or debugger's user guide.

The SAM D20 Xplained Pro evaluation board for the SAM D20 supports programming and debugging through the onboard embedded debugger so no external programmer or debugger is needed.

34.7.1 Cortex Debug Connector (10-pin)

For debuggers and/or programmers that support the Cortex Debug Connector (10-pin) interface the signals should be connected as shown in Figure 34-10 with details described in Table 34-9.

Figure 34-10.Cortex Debug Connector (10-pin)

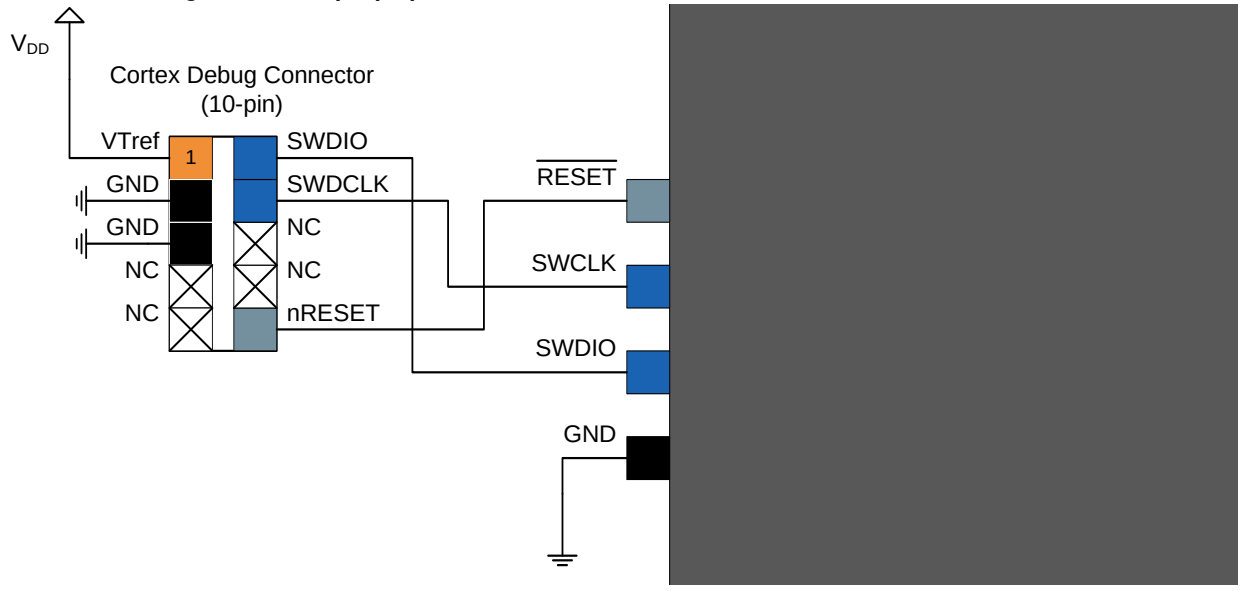


Table 34-9. Cortex Debug Connector (10-pin)

Header Signal Name	Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
$\overline{\text{RESET}}$	Target device reset pin, active low
VTref	Target voltage sense, should be connected to the device V_{DD}
GND	Ground

34.7.2 10-pin JTAGICE3 Compatible Serial Wire Debug Interface

The JTAGICE3 debugger and programmer does not support the Cortex Debug Connector (10-pin) directly, hence a special pinout is needed to directly connect the SAM D20 to the JTAGICE3, alternatively one can use the JTAGICE3 squid cable and manually match the signals between the JTAGICE3 and SAM D20. Figure 34-11 describes how to connect a 10-pin header that support connecting the JTAGICE3 directly to the SAM D20 without the need for a squid cable.

To connect the JTAGICE3 programmer and debugger to the SAM D20, one can either use the JTAGICE3 squid cable, or use a 10-pin connector as shown in Figure 34-11 with details given in Table 34-10 to connect to the target using the JTAGICE3 50 mil cable directly.

Figure 34-11.10-pin JTAGICE3 Compatible Serial Wire Debug Interface

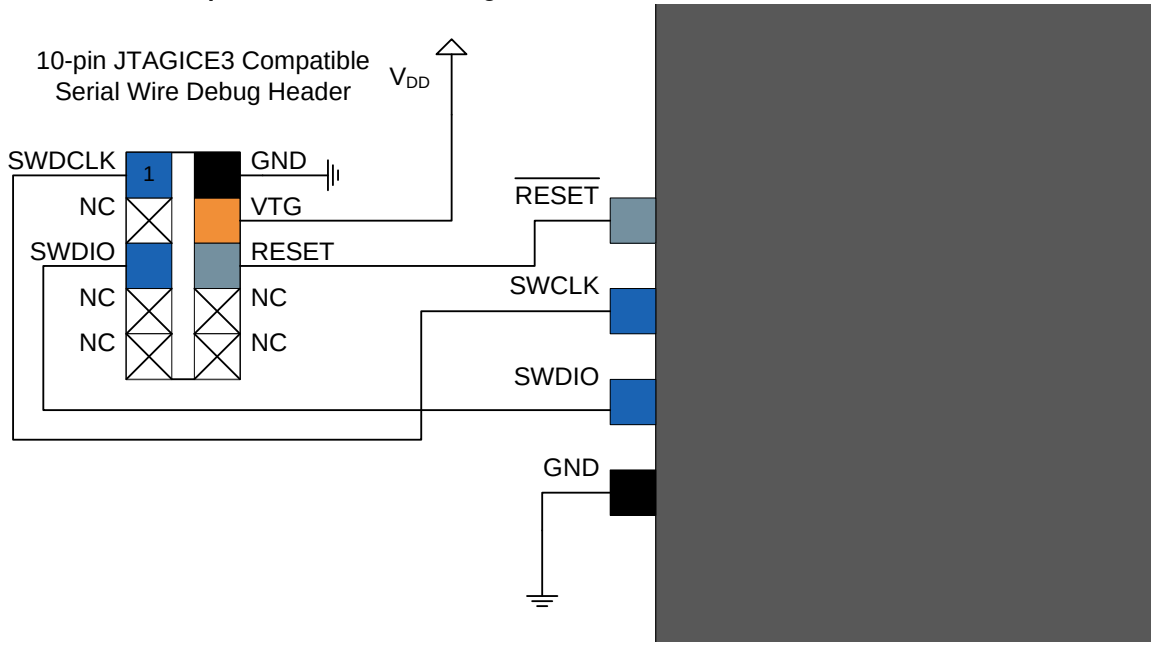


Table 34-10. 10-pin JTAGICE3 Compatible Serial Wire Debug Interface

Header Signal Name	Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
RESET	Target device reset pin, active low
VTG	Target voltage sense, should be connected to the device V_{DD}
GND	Ground

34.7.3 20-pin IDC JTAG Connector

For debuggers and/or programmers that support the 20-pin IDC JTAG Connector, e.g. the SAM-ICE, the signals should be connected as shown in Figure 34-12 with details described in Table 34-11.

Figure 34-12.20-pin IDC JTAG Connector

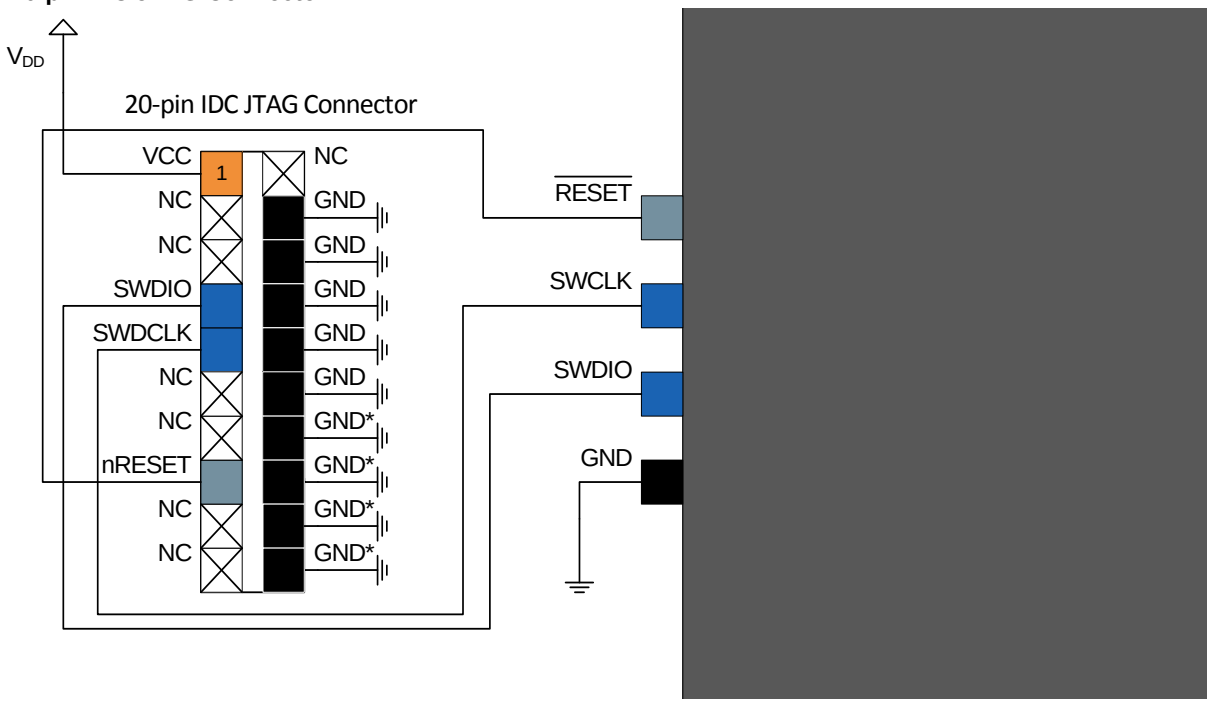


Table 34-11. 20-pin IDC JTAG Connector

Header Signal Name	Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
nRESET	Target device reset pin, active low
VCC	Target voltage sense, should be connected to the device V_{DD}
GND	Ground
GND*	These pins are reserved for firmware extension purposes. They can be left open or connected to GND in normal debug environment. They are not essential for SWD in general.

35. Errata

35.1 Revision C

35.1.1 Device

1 - If APB clock is stopped and GCLK clock is running, APB read access to read-synchronized registers will freeze the system. The CPU and the DAP AHB-AP are stalled, as a consequence debug operation is impossible. Errata reference: 10416

Fix/Workaround:

Do not make read access to read-synchronized registers when APB clock is stopped and GCLK is running. To recover from this situation, power cycle the device or reset the device using the RESETN pin.

2 - DFLLVAL.COARSE, DFLLVAL.FINE, DFLLMUL.CSTEP and DFLLMUL.FSTEP bit groups are not correctly located in the register map. Errata reference: 10988

DFLLVAL.COARSE is only 5 bits and located in DFLLVAL[12..8]. DFLLVAL.FINE is only 8 bits and located in DFLLVAL[7:0]. DFLLMUL.CSTEP is only 5 bits and located in DFLLMUL[28..24]. DFLLMUL.FSTEP is only 8 bits and located in DFLLMUL[23:16]

Fix/Workaround:

DFLLVAL.COARSE, DFLLVAL.FINE, DFLLMUL.CSTEP and DFLLMUL.FSTEP should not be used if code compatibility is required with future device revisions.

35.1.2 PM

1 - If a clock failure is detected (INTFLAG.CFD = 1) on the crystal oscillator connected to GCLKMAIN, and the INTFLAG.CFD bit and CTRL.BKUPCLK bit is cleared to switch the clock back to GCLKMAIN while the clock failure is still present, the INTFLAG.CFD bit will not be set again and the INTFLAG.XOSCRDY flag will not be cleared. Errata reference: 10858

Fix/Workaround:

If a clock failure is detected on the crystal oscillator connected to GCLKMAIN, the device should be reset after the failure condition has been resolved.

2 - The SysTick timer do not generate wake up signal to the Power Manager, and therefore cannot be used to wake up the CPU from sleep mode Errata reference: 11012

Fix/Workaround:

None.

35.1.3 GCLK

1 - When the GCLK generator is enabled (GENCTRL.GENEN = 1), set as output (GENCTRL.OE = 1) and use a division factor of one (GENDIV.DIV = 1 or 0 and GENCTRL.DIVSEL=0), the GCLK_IO might not be set to the configured GENCTRL.OOV value after disabling the GCLK generator (GENCTRL.GENEN=0). Errata reference: 10716

Fix/Workaround

Disable the OE request of the GCLK generator (GENCTRL.OE = 0) before disabling the GCLK generator (GENCTRL.GENEN = 0).

2 - The GCLK Generator clock is stuck when disabling the generator and changing the division factor from one to a different value while the GCLK generator is set as output. Errata reference: 10686

When the GCLK generator is enabled (GENCTRL.GENEN=1), set as output (GENCTRL.OE=1) and use a division factor of one (GENDIV.DIV=1 or 0 and GENCTRL.DIVSEL=0), if the division factor is written to a value different of one or zero after disabling the GCLK generator (GENCTRL.GENEN=0), the GCLK generator will be stuck.

Fix/Workaround

Disable the OE request of the GCLK generator (GENCTRL.OE=0) before disabling the GCLK generator (GENCTRL.GENEN=0).

3 - When a GCLK is locked and the generator used by the locked GCLK is not GCLK generator 1, issuing a GCLK software reset will lock up the GCLK with the SYNCBUSY flag always set. Errata reference: 10645

Fix/Workaround

Do not issue a GCLK SWRST or map GCLK generator 1 to ""locked"" GCLKs.

35.1.4 XOSC32K

1 - The automatic amplitude control of the XOSC32K do not work. Errata reference: 10933

Fix/Workaround:

Use the XOSC32K with Automatic Amplitude control disabled (XOSC32K.AAMPEN = 0)

35.1.5 DFLL48M

1 - If the firmware writes to the DFLLMUL.MUL register in the same cycle as the closed loop mode tries to update it, the fine calibration will first be reset to midpoint and then incremented/decremented by the closed loop mode. Then the coarse calibration will be performed with the updated fine value. If this happens before the dfll have got a lock, the new fine calibration value can be anything between 128-DFLLMUL.FSTEP and 128+DFLLMUL.FSTEP

which could give smaller calibration range for the fine calibration. Errata reference: 10634

Workaround:

Always wait until the DFLL48M has locked before writing the DFLLMUL.MUL register

2 - Changing the DFLLVAL.FINE calibration bits of the DFLL48M Digital Frequency Locked Loop might result in a short output frequency overshoot. This might occur both in open loop mode while writing DFLLVAL.FINE by software and closed loop mode when the DFLL automatically adjusts its output frequency. Errata reference: 10537

Fix/Workaround

- When using DFLL48M in open loop mode, be sure the DFLL48M is not used by any module while DFLLVAL.FINE is written.

- When using DFLL48M in closed loop mode, be sure that DFLLCTRL.STABLE is written to 1. The DFLL clock should not be used by any modules until the DFLL locks are set.

If the application requires on-the-fly DFLL calibration (temperature/VCC drift compensation), the firmware should perform, either periodically or when the DFLL48M frequency differ too much from target frequency (indicated by DFLLVAL.DIFF), the following:

o Switch system clock/module clocks to different clock than DFLL48M

o Re-initiate a DFLL48M closed loop lock sequence by disabling and re-enabling the DFLL48M

o Wait for fine lock (PCLKSR.DFLLLCKF set to 1)

o Switch back system clock/module clocks to the DFLL48M

Better accuracy is achieved using a high multiplier for the DFLL48M, using a scaled down or slow clock as reference. A multiplier of 6 will have a theoretical worst case frequency deviation from the reference clock of +/- 8.33%. A multiplier of 500 will have a theoretical worst case frequency deviation from the reference clock of +/- 0.1%.

3 - If the DFLL48M reaches the maximum or minimum COARSE or FINE calibration values during the locking sequence, an out of bounds interrupt will be generated. These interrupts will be generated even if the final calibration values at DFLL48M lock are not at maximum or minimum, and might therefore be false out of bounds interrupts. Errata reference: 10669

Fix/Workaround:

Check that the lockbits: DFLLLCKC and DFLLLCKF in the SYSCTRL Interrupt Flag Status and Clear register (INTFLAG) are both set before enabling the DFLLOOB interrupt.

35.1.6 EVSYS

1 - Using synchronous or resynchronized paths, some channels (0,3,6,7) detect an overrun on every event even if no overrun condition is present. Errata reference: 10895

Fix/Workaround:

Ignore overrun detection bit for channels 0,3,6,7.

Use channels 1,2,4,5 if overrun detection is required.

2 - Changing the selected generator of a channel can trigger a spurious interrupt/event. Errata reference: 10443

Fix/Workaround:

To change the generator of a channel, first write with EDGESEL written to zero, then perform a second write with EDGESEL written to its target value.

35.1.7 SERCOM

1 - The SERCOM SPI CTRLA register bit 17 (DOPO Bit 1) will always be zero, and cannot be changed. Therefore the SERCOM SPI cannot be switched between master and slave mode on the same DI and DO pins. Errata reference: 10812

Fix/Workaround:

Connect the alternate DI and DO pins externally and use the port MUX to switch between pin configurations for master and slave functionality.

2 - When the SERCOM is in slave SPI mode, the BUFOVF flag is not automatically cleared when CTRLB.RXEN is set to zero. Errata reference: 10563

Fix/Workaround:

The BUFOVF flag must be manually cleared by software.

3 - The sercom SPI BUFOVF status bit is not set until the next character is received after a buffer overflow, instead of directly after the overflow has occurred. Errata reference: 10551

Fix/Workaround:

None.

In the sercom SPI the CTRLA.IBON bit will always be zero, and cannot be changed.

Fix/Workaround:

None.

35.1.8 ADC

1 - The automatic right shift of the result when accumulating/averaging ADC samples does not work. Errata reference: 10530

Fix/Workaround:

To accumulate or average more than 16 samples, one must add the number of automatic right shifts to AVGCTRL.ADJRES to perform the correct number of right shifts. For example, for averaging 128 samples, AVGCTRL.ADJRES must be written to 7 instead of 4, as the automatic right shift of 3 is not done. For oversampling to 16 bits resolution, AVGCTRL.ADJRES must be written to 4 instead of 0 as the automatic right shift of 4 is not done.

The maximum number of right shifts that can be done using ADJRES is 7. This means that when averaging more than 128 samples, the result will be more than 12 bits, and the additional right shifts to get the result down to 12 bits must be done by firmware.

35.1.9 Flash

1 - When cache read mode is set to deterministic (READMODE=2), setting CACHEDIS=1 does not lead to 0 wait states on Flash access. Errata reference: 10830

Fix/Workaround:

When disabling the cache (CTRLB.CACHEDIS=1), the user must also set READMODE to 0 (CTRLB.READMODE=0).

2 - When NVMCTRL issues either erase or write commands and the NVMCTRL cache is not in LOW_POWER mode, CPU hardfault exception may occur. Errata reference: 10804

Fix/Workaround

Either:

- turn off cache before issuing flash commands by setting the NVMCTRL CTRLB.CACHEDIS bit to one.

- Configure the cache in LOW_POWER mode by writing 0x1 into the NVMCTRL CTRLB.READMODE bits.

35.2 Revision B

35.2.1 Device

1 - If APB clock is stopped and GCLK clock is running, APB read access to read-synchronized registers will freeze the system. The CPU and the DAP AHB-AP are stalled, as a consequence debug operation is impossible. Errata reference: 10416

Fix/Workaround:

Do not make read access to read-synchronized registers when APB clock is stopped and GCLK is running. To recover from this situation, power cycle the device or reset the device using the RESETN pin.

2 - DFLLVAL.COARSE, DFLLVAL.FINE, DFLLMUL.CSTEP and DFLLMUL.FSTEP bit groups are not correctly located in the register map. Errata reference: 10988

DFLLVAL.COARSE is only 5 bits and located in DFLLVAL[12..8]. DFLLVAL.FINE is only 8 bits and located in DFLLVAL[7:0]. DFLLMUL.CSTEP is only 5 bits and located in DFLLMUL[28..24]. DFLLMUL.FSTEP is only 8 bits and located in DFLLMUL[23:16]

Fix/Workaround:

DFLLVAL.COARSE, DFLLVAL.FINE, DFLLMUL.CSTEP and DFLLMUL.FSTEP should not be used if code compatibility is required with future device revisions.

35.2.2 PM

1 - If a clock failure is detected (INTFLAG.CFD = 1) on the crystal oscillator connected to GCLKMAIN, and the INTFLAG.CFD bit and CTRL.BKUPCLK bit is cleared to switch the clock back to GCLKMAIN while the clock failure is still present, the INTFLAG.CFD bit will not be set again and the INTFLAG.XOSCRDY flag will not be cleared. Errata reference: 10858

Fix/Workaround:

If a clock failure is detected on the crystal oscillator connected to GCLKMAIN, the device should be reset after the failure condition has been resolved.

2 - The SysTick timer do not generate wake up signal to the Power Manager, and therefore cannot be used to wake up the CPU from sleep mode Errata reference: 11012

Fix/Workaround:

None.

35.2.3 GCLK

1 - When the GCLK generator is enabled (GENCTRL.GENEN = 1), set as output (GENCTRL.OE = 1) and use a division factor of one (GENDIV.DIV = 1 or 0 and GENCTRL.DIVSEL=0), the GCLK_IO might not be set to the configured GENCTRL.OOV value after disabling the GCLK generator (GENCTRL.GENEN=0). Errata reference: 10716

Fix/Workaround

Disable the OE request of the GCLK generator (GENCTRL.OE = 0) before disabling the GCLK generator (GENCTRL.GENEN = 0).

2 - The GCLK Generator clock is stuck when disabling the generator and changing the division factor from one to a different value while the GCLK generator is set as output. Errata reference: 10686

When the GCLK generator is enabled (GENCTRL.GENEN=1), set as output (GENCTRL.OE=1) and use a division factor of one (GENDIV.DIV=1 or 0 and GENCTRL.DIVSEL=0), if the division factor is written to a value different of one or zero after disabling the GCLK generator (GENCTRL.GENEN=0), the GCLK generator will be stuck.

Fix/Workaround

Disable the OE request of the GCLK generator (GENCTRL.OE=0) before disabling the GCLK generator (GENCTRL.GENEN=0).

3 - When a GCLK is locked and the generator used by the locked GCLK is not GCLK generator 1, issuing a GCLK software reset will lock up the GCLK with the SYNCBUSY flag always set. Errata reference: 10645

Fix/Workaround

Do not issue a GCLK SWRST or map GCLK generator 1 to ""locked"" GCLKs.

35.2.4 XOSC32K

1 - The automatic amplitude control of the XOSC32K do not work. Errata reference: 10933

Fix/Workaround:

Use the XOSC32K with Automatic Amplitude control disabled (XOSC32K.AAMPEN = 0)

35.2.5 DFLL48M

1 - If the firmware writes to the DFLLMUL.MUL register in the same cycle as the closed loop mode tries to update it, the fine calibration will first be reset to midpoint and then incremented/decremented by the closed loop mode. Then the coarse calibration will be performed with the updated fine value. If this happens before the dfll have got a lock, the new fine calibration value can be anything between 128-DFLLMUL.FSTEP and 128+DFLLMUL.FSTEP which could give smaller calibration range for the fine calibration. Errata reference: 10634

Workaround:

Always wait until the DFLL48M has locked before writing the DFLLMUL.MUL register

2 - Changing the DFLLVAL.FINE calibration bits of the DFLL48M Digital Frequency Locked Loop might result in a short output frequency overshoot. This might occur both in open loop mode while writing DFLLVAL.FINE by

software and closed loop mode when the DFLL automatically adjusts its output frequency. Errata reference: 10537

Fix/Workaround

- When using DFLL48M in open loop mode, be sure the DFLL48M is not used by any module while DFLLVAL.FINE is written.

- When using DFLL48M in closed loop mode, be sure that DFLLCTRL.STABLE is written to 1. The DFLL clock should not be used by any modules until the DFLL locks are set.

If the application requires on-the-fly DFLL calibration (temperature/VCC drift compensation), the firmware should perform, either periodically or when the DFLL48M frequency differ too much from target frequency (indicated by DFLLVAL.DIFF), the following:

o Switch system clock/module clocks to different clock than DFLL48M

o Re-initiate a DFLL48M closed loop lock sequence by disabling and re-enabling the DFLL48M

o Wait for fine lock (PCLKSR.DFLLLCKF set to 1)

o Switch back system clock/module clocks to the DFLL48M

Better accuracy is achieved using a high multiplier for the DFLL48M, using a scaled down or slow clock as reference. A multiplier of 6 will have a theoretical worst case frequency deviation from the reference clock of +/- 8.33%. A multiplier of 500 will have a theoretical worst case frequency deviation from the reference clock of +/- 0.1%.

3 - If the DFLL48M reaches the maximum or minimum COARSE or FINE calibration values during the locking sequence, an out of bounds interrupt will be generated. These interrupts will be generated even if the final calibration values at DFLL48M lock are not at maximum or minimum, and might therefore be false out of bounds interrupts. Errata reference: 10669

Fix/Workaround:

Check that the lockbits: DFLLLCKC and DFLLLCKF in the SYSCTRL Interrupt Flag Status and Clear register (INTFLAG) are both set before enabling the DFLLOOB interrupt.

35.2.6 EVSYS

1 - Using synchronous or resynchronized paths, some channels (0,3,6,7) detect an overrun on every event even if no overrun condition is present. Errata reference: 10895

Fix/Workaround:

Ignore overrun detection bit for channels 0,3,6,7.

Use channels 1,2,4,5 if overrun detection is required.

2 - Changing the selected generator of a channel can trigger a spurious interrupt/event. Errata reference: 10443

Fix/Workaround:

To change the generator of a channel, first write with EDGESEL written to zero, then perform a second write with EDGESEL written to its target value.

35.2.7 SERCOM

1 - The SERCOM SPI CTRLA register bit 17 (DOPO Bit 1) will always be zero, and cannot be changed. Therefore the SERCOM SPI cannot be switched between master and slave mode on the same DI and DO pins. Errata reference: 10812

Fix/Workaround:

Connect the alternate DI and DO pins externally and use the port MUX to switch between pin configurations for master and slave functionality.

2 - When the SERCOM is in slave SPI mode, the BUFOVF flag is not automatically cleared when CTRLB.RXEN is set to zero. Errata reference: 10563

Fix/Workaround:

The BUFOVF flag must be manually cleared by software.

3 - The sercom SPI BUFOVF status bit is not set until the next character is received after a buffer overflow, instead of directly after the overflow has occurred. Errata reference: 10551

Fix/Workaround:

None.

In the sercom SPI the CTRLA.IBON bit will always be zero, and cannot be changed.

Fix/Workaround:

None.

35.2.8 ADC

1 - The automatic right shift of the result when accumulating/averaging ADC samples does not work. Errata reference: 10530

Fix/Workaround:

To accumulate or average more than 16 samples, one must add the number of automatic right shifts to AVGCTRL.ADJRES to perform the correct number of right shifts. For example, for averaging 128 samples, AVGCTRL.ADJRES must be written to 7 instead of 4, as the automatic right shift of 3 is not done. For oversampling to 16 bits resolution, AVGCTRL.ADJRES must be written to 4 instead of 0 as the automatic right shift of 4 is not done.

The maximum number of right shifts that can be done using ADJRES is 7. This means that when averaging more than 128 samples, the result will be more than 12 bits, and the additional right shifts to get the result down to 12 bits must be done by firmware.

35.2.9 PTC

1 - PTC in self-capacitance mode Errata reference: 10684

The two lowest gain settings are not selectable and an attempt by the QTouch Library to set enable of these may result in a higher sensitivity than optimal for the sensor. The PTC will not detect all touches.

This errata does not affect mutual-capacitance mode which operates as specified.

Fix/Workaround:

Use SAM D20 revision C or later for self-capacitance touch sensing.

35.2.10 Flash

1 - When cache read mode is set to deterministic (READMODE=2), setting CACHEDIS=1 does not lead to 0 wait states on Flash access. Errata reference: 10830

Fix/Workaround:

When disabling the cache (CTRLB.CACHEDIS=1), the user must also set READMODE to 0 (CTRLB.READMODE=0).

2 - When NVMCTRL issues either erase or write commands and the NVMCTRL cache is not in LOW_POWER mode, CPU hardfault exception may occur. Errata reference: 10804

Fix/Workaround

Either:

- turn off cache before issuing flash commands by setting the NVMCTRL CTRLB.CACHEDIS bit to one.

- Configure the cache in LOW_POWER mode by writing 0x1 into the NVMCTRL CTRLB.READMODE bits.

35.3 Revision A

Not Sampled

36. Datasheet Revision History

Please 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.

36.1 Rev. H – 10/2013

Features	Updated the feature list to reflect only the SAM D20 device features
Configuration Summary	Added 256KB Flash and 32KB SRAM to the SAM D20E
Ordering Information	Added SAMD20E18 to the "SAM D20E" on page 4
SYSCTRL	Added note to INTFLAG
NVMCTRL	Updated links to the Table 20-3
SERCOM USART	Updated Table 24-6. SERCOM PAD[X] renamed PAD[X]
ADC	• Updated the "Features" on page 469. Up to 350,000 samples per second (350ksps) • Updated the broken link for INPUTCTRL • Removed "Additional Features"
Schematic Checklist	• Added information about JTAGICE3 compatible SWD connector • Updated connector names to match the names used by ARM • Added information about general debugging and programming to "Programming and Debug Ports" on page 605

Electrical Characteristics	• Updated Table 32-2. Added table note related to BOD33
	• Updated "I/O Pin Characteristics" on page 564. All V_{DDANA} ranges are 1.62V - 2.7V and 2.7V - 3.63V
	• Updated "Digital Frequency Locked Loop (DFLL) Characteristics" on page 578
	• Updated Table 32-4
	  ◦ Replaced the Maximum value that was based on simulation by the actual measurement value
	  ◦ Removed the unused columns
	• Updated the typical values of while(1) at 1.8V in the Table 32-6
	• Updated the Table 32-12
	  ◦ Used measurement values
	  ◦ Removed the min and max values
	  ◦ Added values for C_{IN} and C_{OUT}
	  ◦ Added a table note
	• Updated the Table 32-16
	  ◦ Added min and max values for I_{DD}
	  ◦ Updated typical values
	• Updated the Table 32-18
	  ◦ Added min and max values
	  ◦ Added characteristics for ENOB, SFDR, SINAD and THD

Electrical Characteristics	• Updated the Table 32-21. Added min and max values for I_{DD}
	• Replaced V_{DD} by V_{DDANA} in the Table 32-22
	• Updated the Table 32-24
	  ◦ Added min and max values
	  ◦ Added characterization data for V_{SCALE}
	• Updated the Table 32-17, Table 32-23 and Table 32-26
	  ◦ Added min and max values
	• Updated Table 32-28 and Table 32-29. Added table note related to the cycling endurance
	• Added output frequency characteristics data to the Table 32-35, Table 32-36 and Table 32-37
	• Updated all tables in "I/O Pin Characteristics" on page 564. Added new characterization data
	• Added V_{DDCORE} characteristics to the Table 32-11
	• Updated the table Table 32-13, Table 32-25, Table 32-14 and Table 32-15. Added new characterization data
	• Updated all tables in "Oscillators Characteristics" on page 576. Added new characterization data

ERRATA	Added Errata rev C

36.2 Rev. G – 10/2013

Features	Added the Power Consumption
GCLK	Updated Table 14-10 • Generic clock generator 0 has 8 division factor bits - DIV[7:0]
NVMCTRL	Updated the Table 20-3

Electrical characteristics	Updated the "Electrical Characteristics" on page 558
	• Updated the Table 32-6. Added values of CPU running a "While(1)" algorithm
	• Moved the PTC typical figures from the Typical characteristics into the "Electrical Characteristics" on page 558
	• Updated the "Analog Characteristics" on page 567 Cout max value in the Table 32-12 is 1000nF instead of 200nF
	• Updated the "NVM Characteristics" on page 575:
	  ◦ Added note about the max number of consecutive write in a row before an erase becomes mandatory
	  ◦ Removed all "based on simulation" notes
	  ◦ Updated the Table 32-27, Table 32-28 and Table 32-29

36.3 Rev. F – 10/2013

I/O Multiplexing and Considerations	Updated the Table 5-1 • PA16 and PA17 are I^2C pins in SERCOM1
Memories	Updated the Table 9-4 • Bit Positions [14:3] and [26:15] are “Reserved”
ADC	Updated the “Calibration” on page 472 according to the update done in the “NVM Software Calibration Row Mapping” on page 22
PTC	The “PTC - Peripheral Touch Controller” on page 553 was completely updated
Typical Characteristics	Added the PTC in the Typical Characteristics
ERRATA	Added PTC in Errata rev B

36.4 Rev. E – 09/2013

Ordering Information	Updated the figure of the “Ordering Information” on page 4 • Removed “H = -40 - 85C NiPdAu Plating” from the Package Grade • Renamed “Product Variant” to “Device variant” in the figure of the “Ordering Information” on page 4
DSU	Updated the DID register DID • Renamed SUBFAMILY [7:0] bits to SERIES [7:0] bits • The whole description of the DID bit registers updated • Added Device (all products in the SAM D20 family) column in Device Selection table (DEVSEL)
SYSCTRL	Added ENABLE bits for the BODs and oscillators
Electrical characteristics	Updated “Supply Characteristics” on page 559 • Moved the max values to the min columns into the Table 32-4 • Updated the max values by correct data in the Table 32-4 Updated the “I/O Pin Characteristics” on page 564 • Fixed typos in Table 32-8, Table 32-9 and Table 32-10: “Vdd” was missing in some cells of the tables.

36.5 Rev. D – 08/2013

Description	Updated the whole Atmel SAM D20 “Description” on page 1
General	Fixed different typos throughout the datasheet and applied correctly the template
Block Diagram	Updated the “Block Diagram” on page 7 • Added 2KB RAM and 16KB FLASH

DSU	Updated the Figure 12-1 • Removed HRAM from the block diagram
Clock System	Updated “Clock System” on page 72 • The description of the Basic Read Request has been updated • Updated the Figure 13-3
SYSCTRL	• Updated the writing of the interrupts sources in “Interrupts” on page 136 • Added the reference to INTFLAG
NVMCTRL	Updated the Figure 20-2 • Removed the blue mark from the figure
PORT	• “CPU Local Bus” on page 278: IOBUS address 0x60000000 added • Removed RWM from the description
EVSYS	Channel register (CHANNEL): • Bits 25:24: CHANNEL:PATH description updated.
Schematic Checklist	Updated the “Schematic Checklist” on page 598 • Updated “Introduction” on page 598 • Replaced all TDB by their respective values • Wrote correctly the Ohm symbol in “External Reset Circuit” on page 601
Electrical Characteristics	• Updated “Electrical Characteristics” on page 558 • Removed the colors from “Electrical Characteristics” on page 558 • Added footnote in the Figure 32-16. $f_{ADC} = 6 * CLK_{ADC}$
Table Of Contents	• Applied correctly the template for the TOC

36.6 Rev. C – 07/2013

Description	Updated the front page: • Removed the “Embedded Flash” from the title and from the description on the page 1 • Replaced “speeds” by “frequencies” on the page 1 • Added a sub-bullet on PTC in feature list (256-Channel capacitive touch and proximity sensing) on the page 2 • Replaced IO lines by IO pins on the page 2
Configuration Summary	Updated the table • The RTC • I/O lines changed to I/O pins • Changed 32.768kHz high-accuracy oscillator to 32.768kHz oscillator • Changed 32.768kHz ultra-low power internal oscillator to 32kHz ULP oscillator • Changed 8MHz internal oscillator to 8MHz high-accuracy internal oscillator • Updated SW Debug Interface • Updated the WDT
Ordering information	• Replaced “base line” by “general purpose” • Centered the tables except the ordering code table.
About the Document	• Renamed the chapter to Appendix A and Appendix B • Moved the two Appendixes at the end of the datasheet • Changed the tag of the tables to the tag of appendix tables

Pinout	• Updated the description of “Multiplexing Signals” • Replaced “PORT controller” by “PORT” • Set the Table 5-1 as a continuing table • Updated the table notes of the Table 5-1 • Replaced I/O lines by I/O pins on the page 17
Signal Description	• Removed the column “Comment” from the table
Power Supply	• Removed “nominal” from power supplies • Updated the description of vector regulator • Added link to the “Schematic Checklist”
Clock System	Added the link in the description of “Write-Synchronization” on page 74
Power Manager	Updated the Table 15-4: • The column “Clock Sources” has been updated with new commands • The table note 2 replaced by a reference to “On-demand, Clock Requests” on page 76
System Controller	• Removed “ENABLE bit” from “VREG register”
ADC	Interrupt Flag Status and Clear register (INTFLAG): • Bit 2: INTFLAG.WINMON description updated • Bit 1: INTFLAG.OVERRUN description updated • Bit 0: INTFLAG.RESRDY description updated
DAC	• “Register Summary” on page 542: DATA and DATABUF bit fields updated. • DATA register: Bit fields and description updated. • DATABUF register: Bit fields and description updated.
Electrical Chara	Added “Electrical Characteristics” on page 558
Package Information	Replaced the 64 pins QFN drawing by the correct one

36.7 Rev. B – 07/2013

Block Diagram	“Block Diagram” on page 7: Added output from Analog Comparator block
Signal Description	“Signal Descriptions List” on page 14: Signal Description table cleaned up
Memories	“NVM Software Calibration Row Mapping” on page 22: Added OSC32K Calibration to Table 9-4
DSU	Die Identification register (DID): • Bit 15:12: Added DIE[3:0] bit group • Bit 11:8: Added REVISION[3:0] bit group
EVSYS	“Features” on page 301: Number of event generators updated from 59 to 58
SERCOM SPI	Control A register (CTRLA): • Bit 16: CTRLA.DOPO updated to Bit17:16: CTRLA.DOPO[1:0] • Bit 17:16 - DOPO[1:0] description updated Status register (STATUS): • Bit 2 - STATUS.BUFOVF description updated

ADC	“Accumulation” on page 476: Section added “Averaging” on page 476: Section updated “Oversampling and Decimation” on page 477: Section updated
AC	“Starting a Comparison” on page 511: Heading updated from Basic Operation. “Synchronization” on page 518: Updated with list of write-synchronized bits and registers Register property updated to “Write-Synchronized”: • CTRLA, Comparator Control n

SYSCTRL	• Removed VDDMON and ENABLE bits from registers.
	• Updated start-up time tables for XOSC32K and OSC32K:
	  ◦ XOSC register: Table 16-2
	  ◦ XOSC32K register: Table 16-4
	  ◦ OSC32K register: Table 16-5

Errata Rev. B	“Revision B” on page 612 updates: • “Device” on page 612: Two erratas added (10988 and 10537) • “PM” on page 613: Two erratas added (10858 and 11012) • “XOSC32K” on page 614: One errata added (10933) • “DFLL48M” on page 614: Two erratas added (10634, 10537), one errata updated (10669) • “EVSYS” on page 615: One errata added (10895) • “SERCOM” on page 616: Two erratas added (10812 and 10563), one errata removed (10563) • “ADC” on page 616: One errata updated (10530) • “Flash” on page 617: One errata updated (10804)
Errata Rev. A	Status changed to “Not Sampled”

36.8 Rev. A – 06/2013

1.	Initial revision

Appendix A. Conventions

A.1 Numerical Notation

Table A-1. Numerical Notation

Symbol	Description
165	Decimal number
0b0101	Binary number (example 0b0101 = 5 decimal)
0101	Binary numbers are given without suffix if unambiguous
0x3B24	Hexadecimal number
X	Represents an unknown or don't care value
Z	Represents a high-impedance (floating) state for either a signal or a bus

A.2 Memory Size and Type

Table A-2. Memory Size and Bit Rate

Symbol	Description
KB (kbyte)	kilobyte (2^{10} = 1024)
MB (Mbyte)	megabyte (2^{20} = 1024*1024)
GB (Gbyte)	gigabyte (2^{30} = 1024*1024*1024)
b	bit (binary 0 or 1)
B	byte (8 bits)
1kbit/s	1,000 bit/s rate (not 1,024 bit/s)
1Mbit/s	1,000,000 bit/s rate
1Gbit/s	1,000,000,000 bit/s rate
word	32 bit
half-word	16 bit

A.3 Frequency and Time

Table A-3. Frequency and Time

Symbol	Description
kHz	1kHz = 10^3Hz = 1,000Hz
MHz	10^6 = 1,000,000Hz
GHz	10^9 = 1,000,000,000Hz
s	second

Table A-3. Frequency and Time (Continued)

Symbol	Description
ms	millisecond
µs	microsecond
ns	nanosecond

A.4 Registers and Bits

Table A-4. Register and Bit Mnemonics

Symbol	Description
R/W	Read/Write accessible register bit. The user can read from and write to this bit.
R	Read-only accessible register bit. The user can only read this bit. Writes will be ignored.
W	Write-only accessible register bit. The user can only write this bit. Reading this bit will return an undefined value.
BIT	Bit names are shown in uppercase. (Example ENABLE)
FIELD[n:m]	A set of bits from bit n down to m. (Example: PINA[3:0] = {PINA3, PINA2, PINA1, PINA0}
Reserved	Reserved bits are unused and reserved for future use. For compatibility with future devices, always write reserved bits to zero when the register is written. Reserved bits will always return zero when read.
PERIPHERAL*i*	If several instances of a peripheral exist, the peripheral name is followed by a number to indicate the number of the instance in the range 0-n. PERIPHERAL0 denotes one specific instance.
Reset	Value of a register after a power reset. This is also the value of registers in a peripheral after performing a software reset of the peripheral, except for the Debug Control registers.
SET/CLR	Registers with SET/CLR suffix allows the user to clear and set bits in a register without doing a read-modify-write operation. These registers always come in pairs. Writing a one to a bit in the CLR register will clear the corresponding bit in both registers, while writing a one to a bit in the SET register will set the corresponding bit in both registers. Both registers will return the same value when read. If both registers are written simultaneously, the write to the CLR register will take precedence.

Appendix B. Acronyms and Abbreviations

Table B-1 contains acronyms and abbreviations used in this document.

Table B-1. Acronyms and Abbreviations

Abbreviation	Description
AC	Analog Comparator
ADC	Analog-to-Digital Converter
ADDR	Address
AHB	AMBA Advanced High-performance Bus
AMBA®	Advance Microcontroller Bus Architecture
APB	AMBA Advanced Peripheral Bus
AREF	Analog reference voltage
BLB	Boot Lock Bit
BOD	Brown-out detector
CAL	Calibration
CC	Compare/Capture
CLK	Clock
CRC	Cyclic Redundancy Check
CTRL	Control
DAC	Digital-to-Analog Converter
DAP	Debug Access Port
DFLL	Digital Frequency Locked Loop
DSU	Device Service Unit
EEPROM	Electrically Erasable Programmable Read-Only Memory
EIC	External Interrupt Controller
EVSYS	Event System
GCLK	Generic Clock Controller
GND	Ground
GPIO	General Purpose Input/Output
I^2C	Inter-Integrated Circuit
IF	Interrupt flag
INT	Interrupt
MBIST	Memory built-in self-test
MEM-AP	Memory Access Port
NMI	Non-maskable interrupt

Table B-1. Acronyms and Abbreviations (Continued)

Abbreviation	Description
NVIC	Nested Vector Interrupt Controller
NVM	Non-Volatile Memory
NVMCTRL	Non-Volatile Memory Controller
OSC	Oscillator
PAC	Peripheral Access Controller
PC	Program Counter
PER	Period
PM	Power Manager
POR	Power-on reset
PTC	Peripheral Touch Controller
PWM	Pulse Width Modulation
RAM	Random-Access Memory
REF	Reference
RTC	Real-Time Counter
RX	Receiver/Receive
SERCOM	Serial Communication Interface
SMBus™	System Management Bus
SP	Stack Pointer
SPI	Serial Peripheral Interface
SRAM	Static Random-Access Memory
SYSCTRL	System Controller
SWD	Serial Wire Debug
TC	Timer/Counter
TX	Transmitter/Transmit
ULP	Ultra-low power
USART	Universal Synchronous and Asynchronous Serial Receiver and Transmitter
V_{DD}	Common voltage to be applied to VDDIO, VDDIN and VDDANA
V_{DDIN}	Digital supply voltage
V_{DDIO}	Digital supply voltage
V_{DDANA}	Analog supply voltage
VREF	Voltage reference
WDT	Watchdog Timer
XOSC	Crystal Oscillator

Table of Contents

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
USA
Tel: (+1) (408) 441-0311
Fax: (+1) (408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Roa
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 G.K.
16F Shin-Osaki Kangyo Bldg
1-6-4 Osaki, Shinagawa-ku
Tokyo 141-0032
JAPAN
Tel: (+81) (3) 6417-0300
Fax: (+81) (3) 6417-0370