PIC32CM6408PL10028/032/048/064 Data Sheet

PIC32CM6408PL10028/032/048/064

Product Page Links

5 Volt, 64 KB Flash, 8 KB SRAM with Advanced Analog

Operating Conditions

- 1.8–5.5V, -40°C to +85°C, DC to 24 MHz
- 1.8–5.5V, -40°C to +125°C, DC to 24 MHz

Qualification

- AEC-Q100 Grade 1 (-40°C to +125°C)

Core: Arm® Cortex®-M0+ CPU Running at up to 24 MHz

- Single-cycle hardware multiplier

Memories

- 64 KB in-system self-programmable Flash
- 8 KB SRAM

System

- Power-On Reset (POR) and Brown-Out Detection (BOD)
- Internal and external clock options
- External Interrupt Controller (EIC)
- One non-maskable interrupt

Low Power

- Idle and Standby sleep modes
- SleepWalking peripherals

Input/Output (I/O)

- Up to 55 programmable I/O pins
- Up to 16 external interrupts
- Multi-Voltage I/O (MVIO)

Clock System

- Main Clock Controller (MCLK)
- Generic Clock Controller (GCLK)
- Clock sources:
  - Internal 1-24 MHz High Frequency Oscillator (OSCHF)
  - Internal 32.768 kHz Oscillator (OSC32K)
  - External 32.768 kHz Oscillator input (XOSC32K)

Debugger Development Support

- In-circuit and in-application programming
- Two-Wire Serial Wire Debug Port Interface
- Four hardware breakpoints and one data watchpoint
- Micro Trace Buffer (MTB) for instruction trace in SRAM
- Programming and Debugging Interface Disable (PDID) functionality

Advanced Analog

- One 12-bit, 800 ksps Analog-to-Digital Converter (ADC)
  - Differential and single-ended input
  - Automatic offset and gain error compensation
  - Oversampling and decimation in hardware to support up to 16-bit resolution
- Two Analog Comparators (AC) with window compare function
- Peripheral Touch Controller (PTC) with up to 29 self-capacitance touch channels
- Internal and external voltage reference options

Peripherals

- 2-channel Direct Memory Access Controller (DMAC) with CRC Generator
- 4-channel Event System
- Four 16-bit Timer/Counters:
  - Three 16-bit basic timers (TC0-2)
  - One 16-bit Timer/Counter for Control with four PWM channels (TCC0)
- 32-bit Real Time Counter (RTC) with clock/calendar function
- Watchdog Timer (WDT)
- Two Serial Communication Interfaces (SERCOM), configurable to operate as:

  - USART with full-duplex and single-wire half-duplex configuration
  - ISO7816 UART
  - I^2C up to 1 MHz (SERCOM0)
  - SPI
  - LIN Host/Client
  - RS-485
- Configurable Custom Logic (CCL) with four Look-Up Tables (LUT)
- Integrated Temperature Sensor

Available Packages

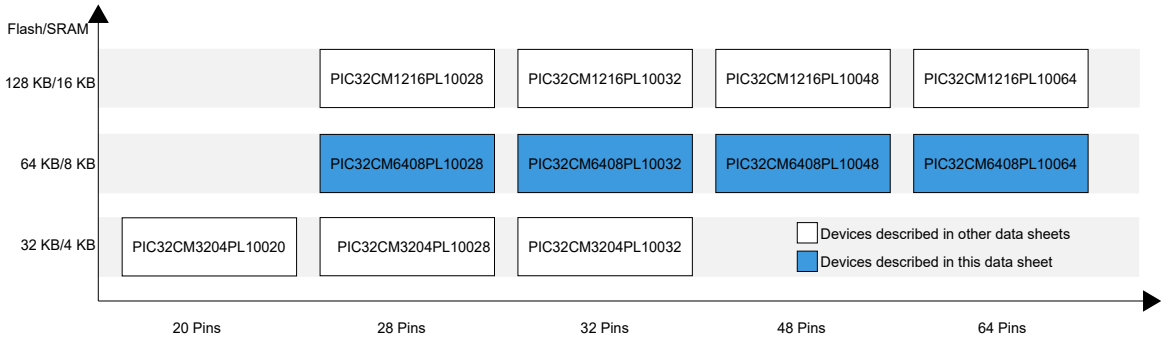
- 28-pin VQFN with Wettable Flanks (WF), SSOP, and SPDIP
- 32-pin TQFP and VQFN WF
- 48-pin TQFP and VQFN WF
- 64-pin TQFP and VQFN WF

Family Overview

The figure below shows the PIC32CM PL10 family devices, illustrating pin count variants and memory sizes:

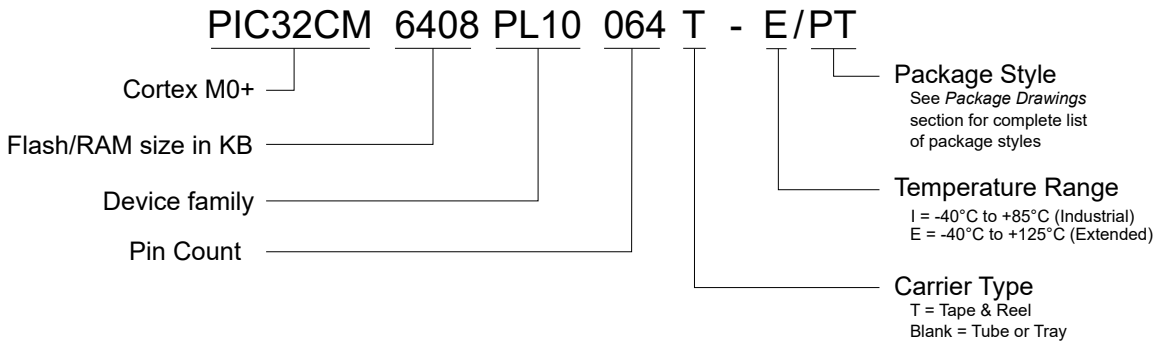
- Vertical migration is possible without code modification, as these devices are fully pin- and feature-compatible
- Horizontal migration to the left reduces the pin count and, therefore, the available features

Figure 1. PIC32CM PL10 family Overview



The name of a device in the PIC32CM PL10 family is decoded as follows:

Figure 2. PIC32CM PL10 family Family Device Designations



Memory Overview

Table 1. Memory Overview

Devices	CM3204-020 CM3204-028 CM3204-032	CM6408-028 CM6408-032 CM6408-048 CM6408-064	CM1216-028 CM1216-032 CM1216-048 CM1216-064
Flash memory	32 KB	64 KB	128 KB
SRAM	4 KB	8 KB	16 KB
Boot ROM (BOOTROM)	8 KB	8 KB	8 KB
User row (USERROW)	512B	512B	512B

Peripheral Overview

Table 2. Peripheral Overview

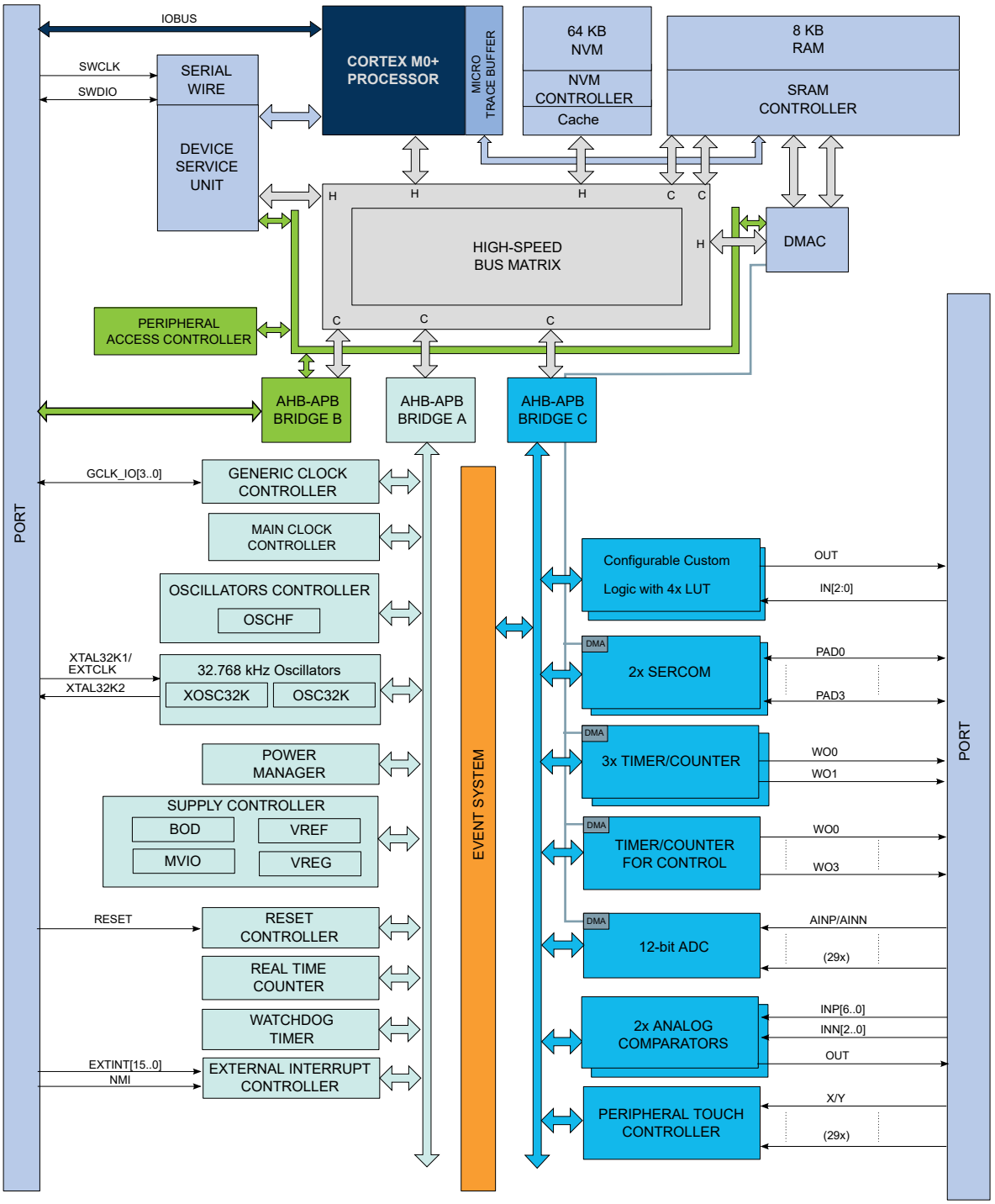
Feature	CM3204-020	CM3204-028 CM6408-028 CM1216-028	CM3204-032 CM6408-032 CM1216-032	CM6408-048 CM1216-048	CM6408-064 CM1216-064
Pins	20	28	32	48	64
Max. frequency (MHz)	24	24	24	24	24
Generic Clock Controller (GCLK)	4	4	4	4	4
External 32.768 kHz Oscillator	1	1	1	1	1
Timer/Counter (TC)	3	3	3	3	3
Timer/Counter for Control Applications (TCC)	1	1	1	1	1
Real-Time Counter (RTC)	1	1	1	1	1
Serial Communication Interface (SERCOM)	2	2	2	2	2
ADC (channels)	1 (11)	1 (16)	1 (20)	1 (25)	1 (29)
Analog Comparator (AC)	2	2	2	2	2
Peripheral Touch Controller (PTC) and (channels)	1 (11)	1 (16)	1 (20)	1 (25)	1 (29)
Configurable Custom Logic Look-up Table (CCL LUT)	4	4	4	4	4
Watchdog Timer (WDT)	1	1	1	1	1
Direct Memory Access Channels (DMA)	2	2	2	2	2
Event System channels (EVSYS)	4	4	4	4	4
General Purpose I/O(1)	17/16	23/22	27/26	42/41	56/55
Multi-Voltage I/O (MVIO)	3	4	4	8	8
External Interrupts	11	13	15	16	16

Note:

1. The PA30/$\overline{\text{RESET}}$ pin is input only.

Table of Contents

1. Block Diagram



2. Pinout

Each pin is controlled by the I/O Pin Controller (PORT) as a general purpose I/O and can alternatively be assigned to one of the peripheral functions A-P.

The following table describes the peripheral signals multiplexed to the PORT I/O pins for each package.

Refer to the *Schematic Checklist* and the *Electrical Characteristics* chapters for the hardware requirements and other considerations for the pin connections.

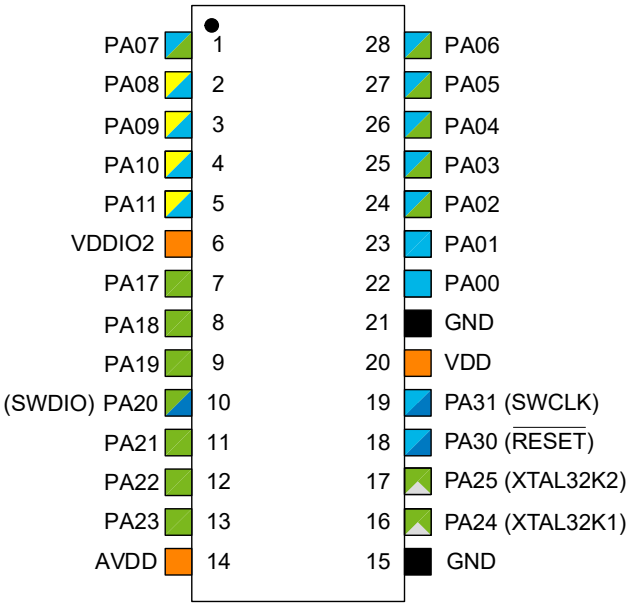
2.1. Configuration Summary

Table 2-1. Configurations and Peripherals of PIC32CM6408PL10

PIC32CM6408PL10				
Peripheral	**64-Pin Package**	**48-Pin Package**	**32-Pin Package**	**28-Pin Package**
Maximum CPU frequency	24 MHz			
SysTick timer	1			
Serial Wire Debug Interface	Yes			
Oscillators	32.768 kHz crystal oscillator (XOSC32K)			
	32.768 kHz internal oscillator (OSC32K)			
	24 MHz high-accuracy internal oscillator (OSCHF)			
Generic Clock - GCLK	4			
DMA channels	2			
Event System channels	4			
I/O Pins	55	41	26	22
External Interrupt lines	16 and one non-maskable interrupt (NMI)			
Serial Communication Interfaces - SERCOM/of which I^2C is supported	2 / 1	2 / 1	2 / 1	2 / 1
Timer Counter - TC	3			
Waveform outputs/Capture inputs channels per TC instance	2			
TC Maximum and Minimum Capture	Yes			
Timer Counter for Control - TCC (number of waveform output channels)	1(4)			
Configurable Custom Logic - CCL (number of LUTs)	1(4)			
Analog-to-Digital Converter - ADC (channels)	1(29)	1(25)	1(20)	1(16)
Analog Comparators - AC	2			
Real-Time Counter - RTC	1			
RTC alarms	1			
RTC compare values	One 32-bit value or two 16-bit values			
Watchdog Timer - WDT	Yes			
Memories CRC32 computation in DMAC	Yes (SRAM, Flash)			
Brown-Out Detection - BOD	VDD, VDDCORE			
Peripheral Touch Controller - PTC (channels)	1(29)	1(25)	1(20)	1(16)

2.2. Pinout Diagrams

2.2.1. 28-pin SSOP and SPDIP

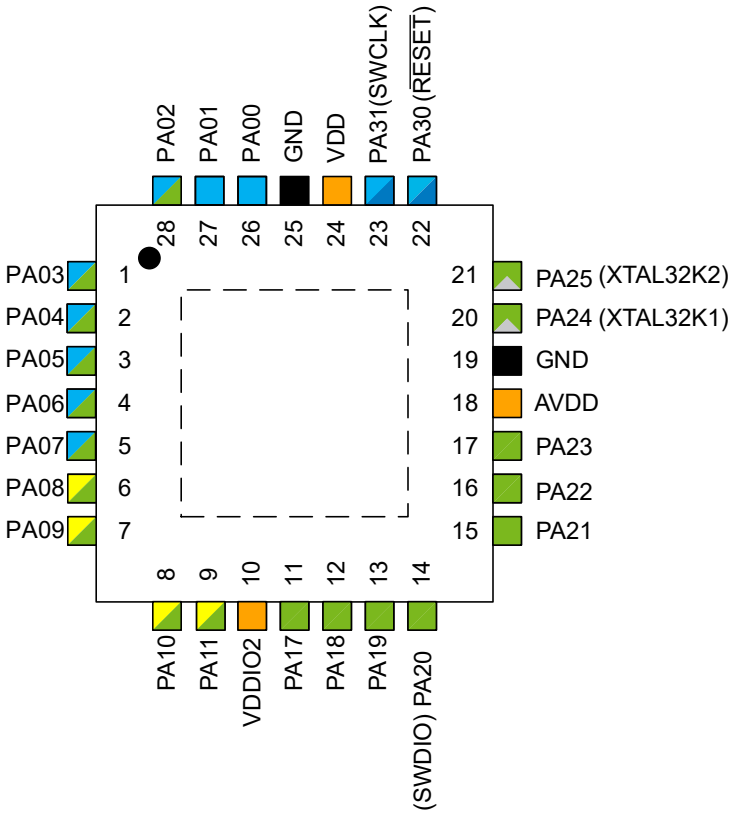


Note: AVDD is internally connected to VDD (not separate power domains).

Power
- Power Supply
- Ground
- Pin on VDD Power Domain
- Pin on AVDD Power Domain
- Pin on VDDIO2 Power Domain

Functionality
- Programming/Debug
- Clock/Crystal
- Digital Function Only
- Analog Function

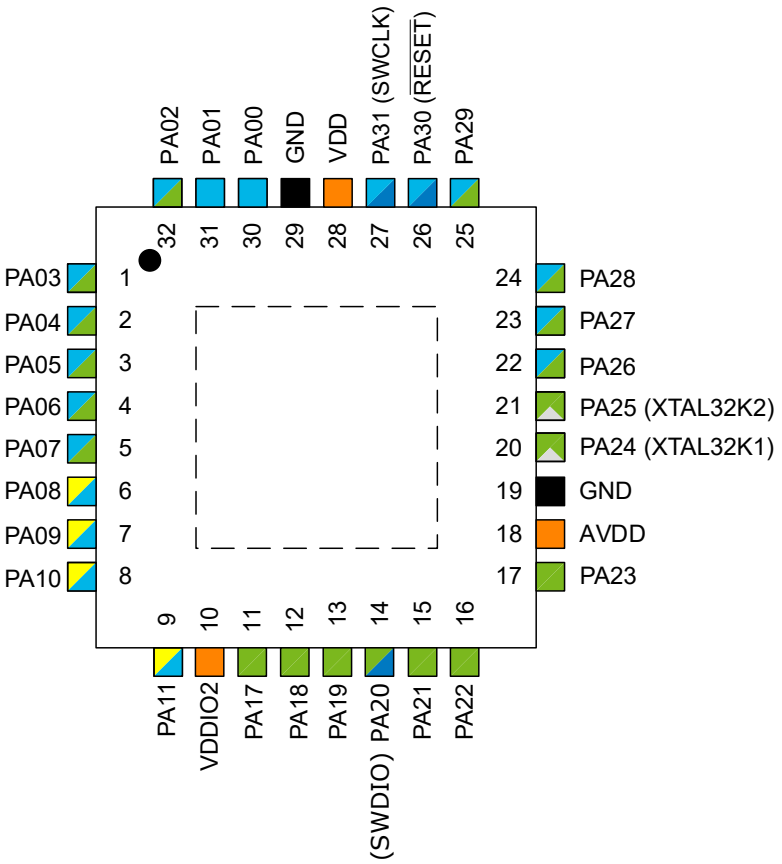
2.2.2. 28-pin VQFN



Note: AVDD is internally connected to VDD (not separate power domains).

Power	Functionality
Power Supply	Programming/Debug
Ground	Clock/Crystal
Pin on VDD Power Domain	Digital Function Only
Pin on AVDD Power Domain	Analog Function
Pin on VDDIO2 Power Domain	

2.2.3. 32-pin VQFN and TQFP

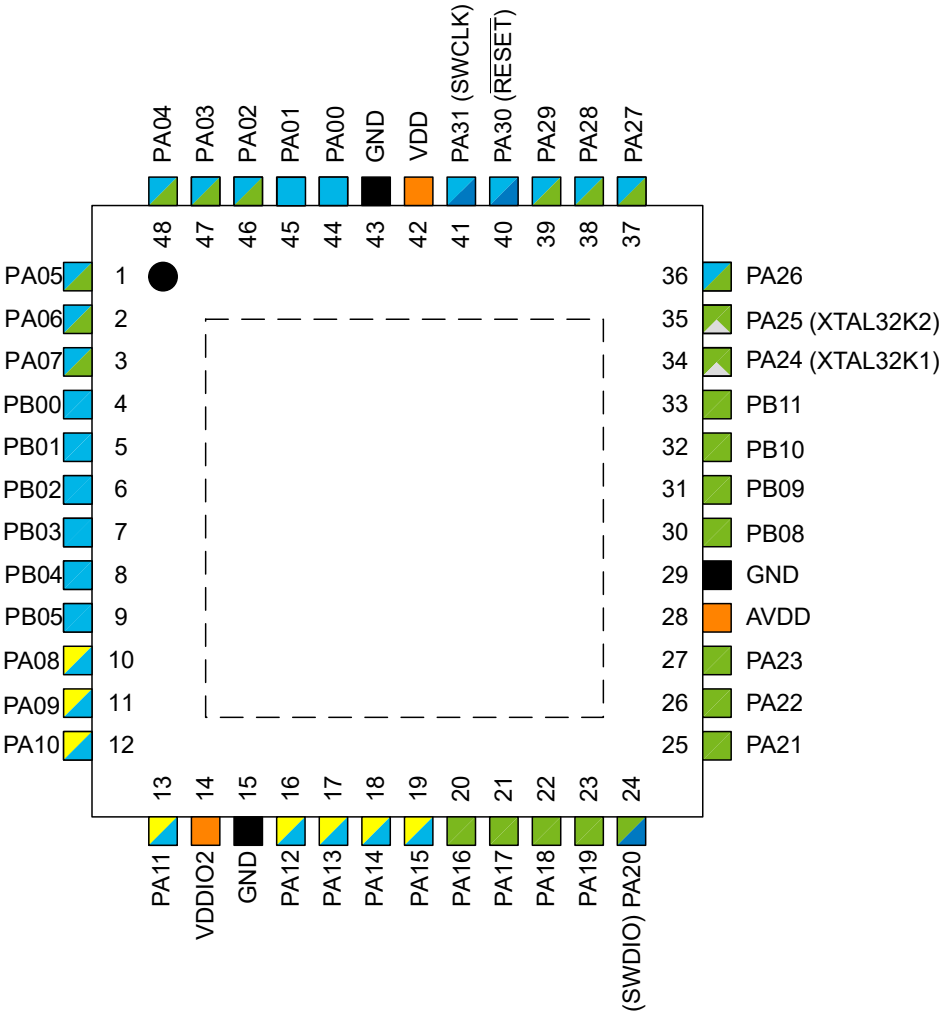


Note: AVDD is internally connected to VDD (not separate power domains).

Power
- Power Supply
- Ground
- Pin on VDD Power Domain
- Pin on AVDD Power Domain
- Pin on VDDIO2 Power Domain

Functionality
- Programming/Debug
- Clock/Crystal
- Digital Function Only
- Analog Function

2.2.4. 48-pin VQFN and TQFP

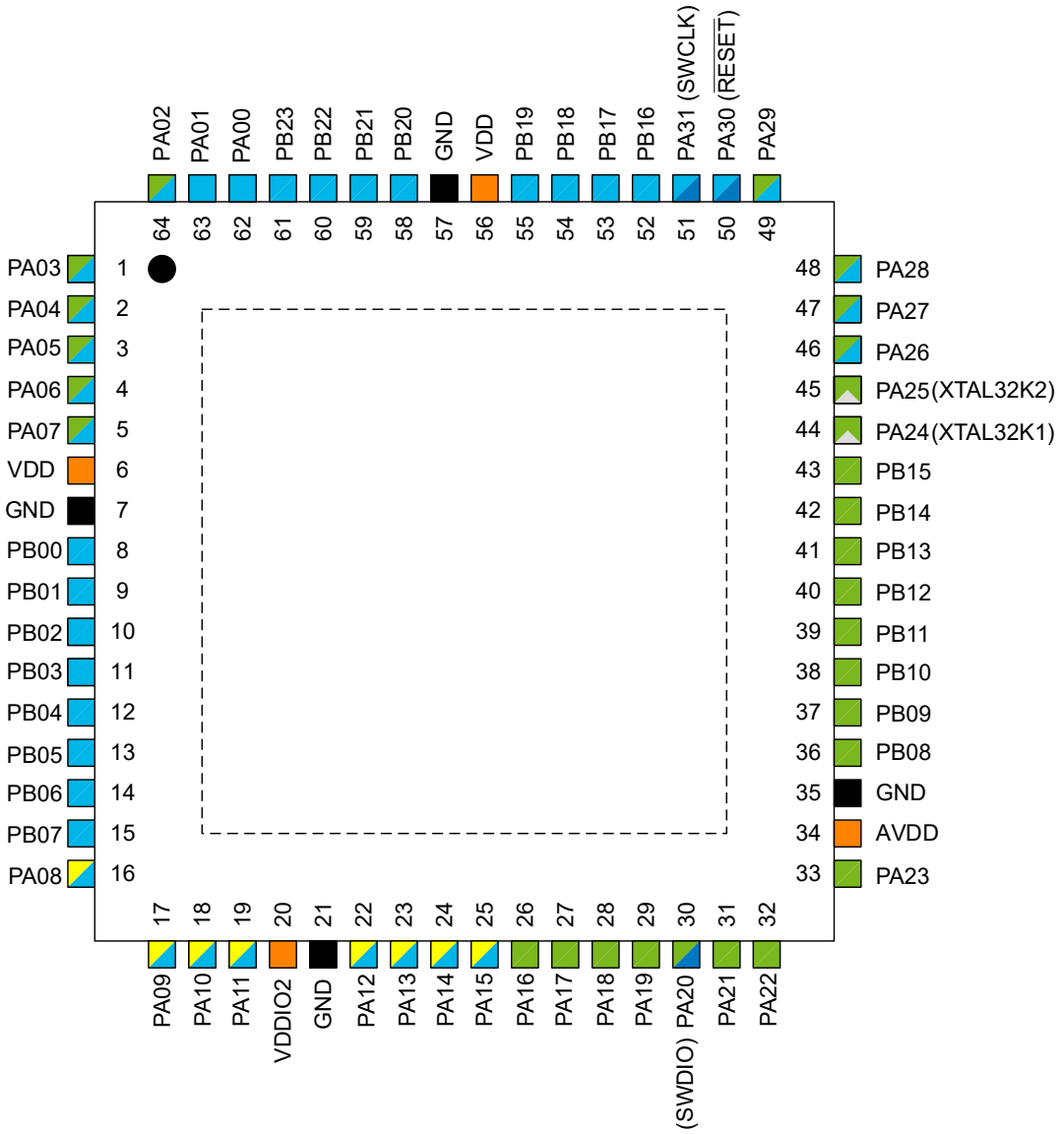


Note: AVDD is internally connected to VDD (not separate power domains).

Power
- Power Supply
- Ground
- Pin on VDD Power Domain
- Pin on AVDD Power Domain
- Pin on VDDIO2 Power Domain

Functionality
- Programming/Debug
- Clock/Crystal
- Digital Function Only
- Analog Function

2.2.5. 64-pin VQFN and TQFP



Note: AVDD is internally connected to VDD (not separate power domains).

Power	Functionality
Power Supply	Programming/Debug
Ground	Clock/Crystal
Pin on VDD Power Domain	Digital Function Only
Pin on AVDD Power Domain	Analog Function
Pin on VDDIO2 Power Domain	

2.3. Pinout and Multiplexing

Package/Pin Number					Pin Name	Special	A	B				C	D	E	F	G	H	I	P
VQFN TQFP			VQFN	SSOP SPDIP			EXTINT	ADC0		PTC	ACn	SERCOMn		TCn	TCC0	SWCLK	CCLn, ACn OUT	GCLKn	PTC
64-pin	48-pin	32-pin	28-pin	28-pin				Pos	Neg	EXTCINT									X/Y
62	44	30	26	22	PA00		EXTINT[0]					0,PAD[0] (2)	1,PAD[0]	0,WO0	WO0		0,IN0	1,IO	
63	45	31	27	23	PA01		NMI					0,PAD[1] (2)	1,PAD[1]	0,WO1	WO1		0,IN1	2,IO	
64	46	32	28	24	PA02		EXTINT[2]	AINP2	AINN2			0,PAD[2]	1,PAD[2]	1,WO0	WO2		0,IN2	3,IO	X2/Y2
1	47	1	1	25	PA03		EXTINT[3]	AINP3	AINN3			0,PAD[3]	1,PAD[3]	1,WO1	WO3		0,OUT	0,IO	X3/Y3
2	48	2	2	26	PA04		EXTINT[4]	AINP4	AINN4			0,PAD[0] (2)						1,IO	X4/Y4
3	1	3	3	27	PA05		EXTINT[5]	AINP5	AINN5			0,PAD[1] (2)					AC1(4)	2,IO	X5/Y5
4	2	4	4	28	PA06		EXTINT[6]	AINP6	AINN6			0,PAD[2]					0,OUT	3, IO	X6/Y6
5	3	5	5	1	PA07		NMI	AINP7	AINN7			0,PAD[3]					AC0(4)	0,IO	X7/Y7
6					VDD														
7					GND														
8	4				PB00		EXTINT[0]						1,PAD[0]		WO0				
9	5				PB01		EXTINT[1]						1,PAD[1]		WO1				
10	6				PB02		EXTINT[2]						1,PAD[2]		WO2				
11	7				PB03		EXTINT[3]						1,PAD[3]		WO3				
12	8				PB04		EXTINT[4]												
13	9				PB05		EXTINT[5]												
14					PB06		EXTINT[6]												
15					PB07		EXTINT[7]												
16	10	6	6	2	PA08	VDDIO2(3)	EXTINT[8]					1,PAD[2]	0,PAD[2]	1,WO0	WO0		1,IN0	1.IO	
17	11	7	7	3	PA09	VDDIO2(3)	EXTINT[9]					1,PAD[3]	0,PAD[3]	1,WO1	WO1		1,IN1	2,IO	
18	12	8	8	4	PA10	VDDIO2(3)	EXTINT[10]					1,PAD[0]	0,PAD[0] (2)	0,WO0	WO2		1,IN2	3,IO	
19	13	9	9	5	PA11	VDDIO2(3)	EXTINT[11]					1,PAD[1]	0,PAD[1] (2)	0,WO1	WO3		1,OUT	1,IO	
20	14	10	10	6	VDDIO2														
21	15				GND														
22	16				PA12	VDDIO2(3)	EXTINT[12]					1,PAD[2]							
23	17				PA13	VDDIO2(3)	EXTINT[13]					1,PAD[3]					AC1(4)		
24	18				PA14	VDDIO2(3)	EXTINT[14]					1,PAD[0]					1,OUT		
25	19				PA15	VDDIO2(3)	EXTINT[15]					1,PAD[1]					AC0(4)		
26	20				PA16		EXTINT[0]	AINP16	AINN16		0,N1 1,N1				WO0		2,IN0		X16/Y16
27	21	11	11	7	PA17		EXTINT[1]	AINP17	AINN17						WO1		2,IN1		X17/Y17
28	22	12	12	8	PA18		EXTINT[2]	AINP18	AINN18		0,P0 1,P0				WO2		2,IN2		X18/Y18
29	23	13	13	9	PA19		EXTINT[3]	AINP19	AINN19		0,N0 1,P1				WO3		2,OUT		X19/Y19
30	24	14	14	10	PA20	SWDIO	EXTINT[4]	AINP20	AINN20		0,P5 1,P2	1,PAD[0]		2,WO0				2,IO	X20/Y20
31	25	15	15	11	PA21		EXTINT[5]	AINP21	AINN21	EXTCINT0	0,P6 1,N0/P5	1,PAD[1]		2,WO1				3,IO	X21/Y21
32	26	16	16	12	PA22		EXTINT[6]	AINP22	AINN22	EXTCINT1	0,P3 1,P3	1,PAD[2]					2,OUT	1, IO	X22/Y22
33	27	17	17	13	PA23	VREFA	EXTINT[7]	AINP23	AINN23		0,N2 1,N2	1,PAD[3]						0,IO	X23/Y23
34	28	18	18	14	AVDD														
35	29	19	19	15	AGND														
36	30				PB08		EXTINT[8]	AINP40	AINN40		0,P1				WO0				X40/Y40
37	31				PB09		EXTINT[9]	AINP41	AINN41						WO1				X41/Y41
38	32				PB10		EXTINT[10]	AINP42	AINN42		0,P2				WO2				X42/Y42
39	33				PB11		EXTINT[11]	AINP43	AINN43						WO3				X43/Y43
40					PB12		EXTINT[12]	AINP44	AINN44										X44/Y44
41					PB13		EXTINT[13]	AINP45	AINN45										X45/Y45
42					PB14		EXTINT[14]	AINP46	AINN46										X46/Y46
43					PB15		EXTINT[15]	AINP47	AINN47										X47/Y47
44	34	20	20	16	PA24	XTAL32K 1	EXTINT[8]	AINP24	AINN24			1,PAD[2]			WO0		3,IN0	1,IO	X24/Y24
45	35	21	21	17	PA25	XTAL32K 2	EXTINT[9]	AINP25	AINN25			1,PAD[3]			WO1		3,IN1	2,IO	X25/Y25
46	36	22			PA26		EXTINT[10]	AINP26	AINN26			1,PAD[0]			WO2		3,IN2	3,IO	X26/Y26
47	37	23			PA27		EXTINT[11]	AINP27	AINN27			1,PAD[1]			WO3		3,OUT		X27/Y27
48	38	24			PA28		EXTINT[12]	AINP28	AINN28										X28/Y28
49	39	25			PA29		EXTINT[13]	AINP29	AINN29									1,IO	X29/Y29
50	40	26	22	18	PA30	RESET	NMI												
51	41	27	23	19	PA31	SWCLK	EXTINT[15]	AINP31	AINN31			1,PAD[0]	0,PAD[2]			SWCLK		0,IO	X31/Y31
52					PB16		EXTINT[0]								WO0				
53					PB17		EXTINT[1]								WO1				
54					PB18		EXTINT[2]								WO2				
55					PB19		EXTINT[3]								WO3				
56	42	28	24	20	VDD														
57	43	29	25	21	GND														
58					PB20		EXTINT[4]							2,WO0					
59					PB21		EXTINT[5]							2,WO1					
60					PB22		EXTINT[6]												
61					PB23		EXTINT[7]												

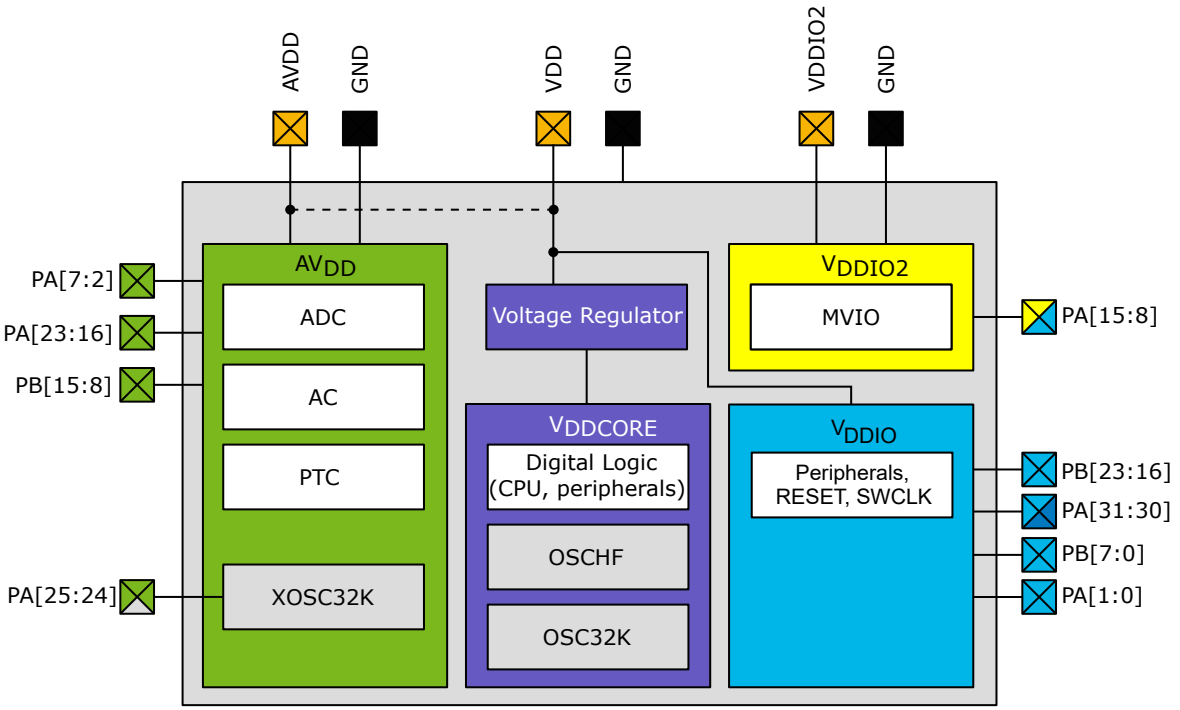
1. To select pin positions, refer to the *PORT* chapter.
2. This pin supports I^2C with SERCOM0.

3. This pin is on the VDDIO2 power domain. For usage and properties, refer to the MVIO description in the *SUPC - Supply Controller* and the *Electrical Characteristics* sections.
4. The output of the Analog Comparator to the pin can be used as an input to the CCL. Refer also to the *CCL - Configurable Custom Logic* section.

3. Power Supply and Start-Up Considerations

3.1. Power Domain Overview

Figure 3-1. Power Domain Overview



3.2. Power Supply Considerations

3.2.1. Power Supplies

The PIC32CM6408PL10 has the following power supply pins:

- VDD: Powers the I/O lines (VDDIO) and the VREG
- AVDD: Powers the analog peripherals: AC, ADC, and PTC
- VDDIO2: Powers the VDDIO2 I/O lines

The same voltage must be applied to both the VDD and AVDD pins. This common voltage is referred to as V_{DD} in the data sheet.

The ground pins, GND, are common to all supplies.

For decoupling recommendations for the different power supplies, refer to the *Schematic Checklist* chapter.

3.2.2. Voltage Regulator

The PIC32CM6408PL10 voltage regulators ensure correct supply voltage to the CPU, digital peripherals, internal oscillators, and other digital logic. They have the following modes:

- Normal mode: The CPU and peripherals are running
- Low Power (LP) mode: This mode is used when the device is in Standby sleep mode

3.2.3. Power-Up Sequence

If VDDIO2 is configured in single supply mode, VDD and VDDIO2 must have the same supply sequence. It is recommended to connect them together outside of the device. See also the *SUPC - Supply Controller* chapter. Observe the minimum and maximum rise rates for V_{DD}, as described in the *Electrical Characteristics* chapter.

3.3. Start-Up

After a successful power-up, the device behavior is controlled by the Power Manager (PM) and the Reset Controller (RSTC).

3.3.1. 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 Boot ROM will initialize the device and select the internal OSCHF divided to a 4 MHz clock signal as the default clock for the device. This is also used as the clock source for Generic Clock Generator 0 (GCLK0). In turn, GCLK0 provides the main clock, GCLK_MAIN, which is used by the Main Clock (MCLK) peripheral. Any further configuration of the clock system is done by the application.

Some synchronous system clocks are active after startup, allowing software execution.

Refer to the Clock Mask Register in the *MCLK - Main Clock* section for the list of default peripheral clocks that are running.

3.3.2. I/O Pins

After power-up, the I/O pins are tri-stated except for RESETn, SWDIO and SWCLK, which are pull-up enabled and configured as input.

3.3.3. Fetching of Initial Instructions

After the device is released from the initial reset, the CPU starts fetching the Program Counter (PC) and Stack Pointer (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 the Arm Architecture Reference Manual for additional information on CPU startup (http://www.arm.com).

3.4. Power-On Reset (POR) and Brown-Out Detectors (BOD)

The following features are used to monitor, warn, and reset the device:

- POR: Power-On Reset on V_{DD} and V_{DDIO2}, handled by the Reset Controller (RSTC) peripheral. The POR is always activated and monitors the voltages at startup and during all sleep modes:
  - If V_{DD} goes below the threshold voltage, the entire device is reset
  - If V_{DDIO2} goes below the threshold voltage, all I/Os supplied by V_{DDIO2} are reset
- BODVDD: Brown-Out Detector on V_{DD}, controlled by the Supply Controller (SUPC) peripheral
- BODVDDIO2: Brown-Out Detector on V_{DDIO2}, controlled by the SUPC peripheral

4. CPU and Architecture

4.1. Cortex M0+ Processor

The PIC32CM6408PL10 implements the Arm Cortex-M0+ processor, based on the ARMv6 Architecture and Thumb®-2 ISA. The Cortex M0+ is fully instruction set compatible with its predecessor, the Cortex-M0 core, and upward compatible with the Cortex-M3 and M4 cores. The implemented Arm Cortex-M0+ is revision r0p1. For more information refer to www.arm.com.

4.1.1. Cortex M0+ Configuration

Table 4-1. Cortex M0+ Configuration

Features	PIC32CM6408PL10 Configuration
Interrupt lines	16
Data endianness	Little-endian
SysTick timer	Present
Number of watchpoint comparators	1
Number of breakpoint comparators	4
Halting debug support	Present
Multiplier	Fast (single cycle)
Single-cycle I/O port	Present
Wake-up interrupt controller	Not supported
Vector Table Offset Register	Present
Unprivileged/Privileged support	All accesses are privileged
Memory Protection Unit	None
Reset all registers	Absent
Instruction fetch width	32-bit

The Arm Cortex-M0+ core has the following bus interfaces:

- A single 32-bit AMBA-3 AHB-Lite system interface provides connections to peripherals and all system memory, including Flash and RAM
- A single 32-bit I/O port bus interfaces to the PORT with 1-cycle loads and stores

4.1.2. 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).
- Nested Vectored Interrupt Controller (NVIC)
  - External interrupt signals connect to the NVIC, which 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 handling and efficient processing of late-arriving interrupts. Refer to NVIC – Nested Vectored Interrupt Controller and *Cortex-M0+ Technical Reference Manual* for details (www.arm.com).
- System Timer (SysTick)
  - The System Timer is a 24-bit timer clocked by CLK_CPU that extends the functionality of both the processor and the NVIC. Refer to *Cortex-M0+ Technical Reference Manual* for details (www.arm.com). When the SysTick Overflow Interrupt is enabled, the RAM Back Bias Control must be disabled (PM->STDBYCFG.bit.BBIASHS = 0) before entering Standby sleep mode.
- System Control Block (SCB)

  - The System Control Block provides system implementation information and system control. This includes configuration, control, and reporting of system exceptions. Refer to *Cortex-M0+ Devices Generic User Guide* for details (www.arm.com).

- Micro Trace Buffer (MTB)
  - The CoreSight MTB-M0+ (MTB) provides a simple execution trace capability to the Cortex-M0+ processor. Refer to the section Micro Trace Buffer and the *CoreSight MTB-M0+ Technical Reference Manual* for details (www.arm.com).

4.1.3. Cortex-M0+ Address Map

Table 4-2. 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)
0x41008000	Micro Trace Buffer (MTB)

4.1.4. I/O Interface

The AMBA® AHB-Lite™ interface offers direct access from the CPU to the PORT registers.

Accesses to the AHB-Lite™ and the single cycle I/O interfaces can be made concurrently: The Cortex-M0+ processor can fetch the next instructions while accessing the I/Os, sustaining single cycle I/O accesses for as long as needed.

4.2. NVIC – Nested Vectored Interrupt Controller

4.2.1. Overview

The Nested Vectored Interrupt Controller (NVIC) in the PIC32CM6408PL10 supports 16 interrupt lines with four different priority levels. For more details, refer to the Cortex-M0+ Technical Reference Manual (www.arm.com).

4.2.2. Interrupt Line Mapping

Each of the interrupt lines is connected to one peripheral instance, as shown in the table below. Each peripheral can have one or more interrupt flags, which are 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 by the peripheral when the interrupt flag is set and the corresponding interrupt is enabled.

The interrupt requests for one peripheral are ORed together at the 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.

Table 4-3. Interrupt Line Mapping

Vector Number	Interrupt Vector	Interrupt Source	Description
NMI	NMI	NMI	Non-Maskable Interrupt from the EIC
0	SYSTEM	CKRDY	Clock Ready interrupt from MCLK
		OSCHFRDY	OSCHF is ready interrupt from OSCCTRL
		XOSC32KRDY	XOSC32K ready interrupt from OSCCTRL
		CLKFAIL	XOSC32K has failed interrupt from OSCCTRL
		OSC32KRDY	OSC32K ready interrupt from OSCCTRL
		VDDIO2LPMPOR	V_{DDIO2} Low-Power Mode POR interrupt from SUPC
		VDDIO2OK	V_{DDIO2} OK interrupt from SUPC
		VLM	Voltage Level Monitor interrupt from SUPC
		BODVDDRDY	V_{DD} Brown-Out interrupt Detector Ready interrupt from SUPC
		FLASH	Access Error is detected by client Flash
		HSRAMCM0P	Access Error is detected by client SRAMCM0P
		HSRAMDSU	Access Error is detected by client SRAMDSU
		APBB	Access Error is detected by client APBB
		APBA	Access Error is detected by client APBA
		APBC	Access Error is detected by client APBC
		SRAMDMAC	Access Error is detected by client SRAMDMAC
		BROM	Access Error is detected by client BROM
		PAC	Peripheral Access Error occurs while accessing PAC
		PM	Peripheral Access Error occurs while accessing PM
		MCLK	Peripheral Access Error occurs while accessing MCLK
		RSTC	Peripheral Access Error occurs while accessing RSTC
		OSCCTRL	Peripheral Access Error occurs while accessing OSCCTRL
		SUPC	Peripheral Access Error occurs while accessing SUPC
		GCLK	Peripheral Access Error occurs while accessing GCLK
		WDT	Peripheral Access Error occurs while accessing WDT
		RTC	Peripheral Access Error occurs while accessing RTC
		EIC	Peripheral Access Error occurs while accessing EIC
		PORT	Peripheral Access Error occurs while accessing PORT
		DSU	Peripheral Access Error occurs while accessing DSU
		NVMCTRL	Peripheral Access Error occurs while accessing NVMCTRL
		DMAC	Peripheral Access Error occurs while accessing DMAC
		MTB	Peripheral Access Error occurs while accessing MTB
		HMATRIXHS	Peripheral Access Error occurs while accessing HMATRIXHS
		EVSYS	Peripheral Access Error occurs while accessing EVSYS
		SERCOMn	Peripheral Access Error occurs while accessing SERCOMn
		TCn	Peripheral Access Error occurs while accessing TCn
		TCCn	Peripheral Access Error occurs while accessing TCCn
		ADC0	Peripheral Access Error occurs while accessing ADC0
		AC	Peripheral Access Error occurs while accessing AC
		CCL	Peripheral Access Error occurs while accessing CCL
		PTC	Peripheral Access Error occurs while accessing PTC
		SYSCTRL	Peripheral Access Error occurs while accessing SYSCTRL
1	WDT	EW	Early Warning interrupt from WDT

Table 4-3. Interrupt Line Mapping (continued)

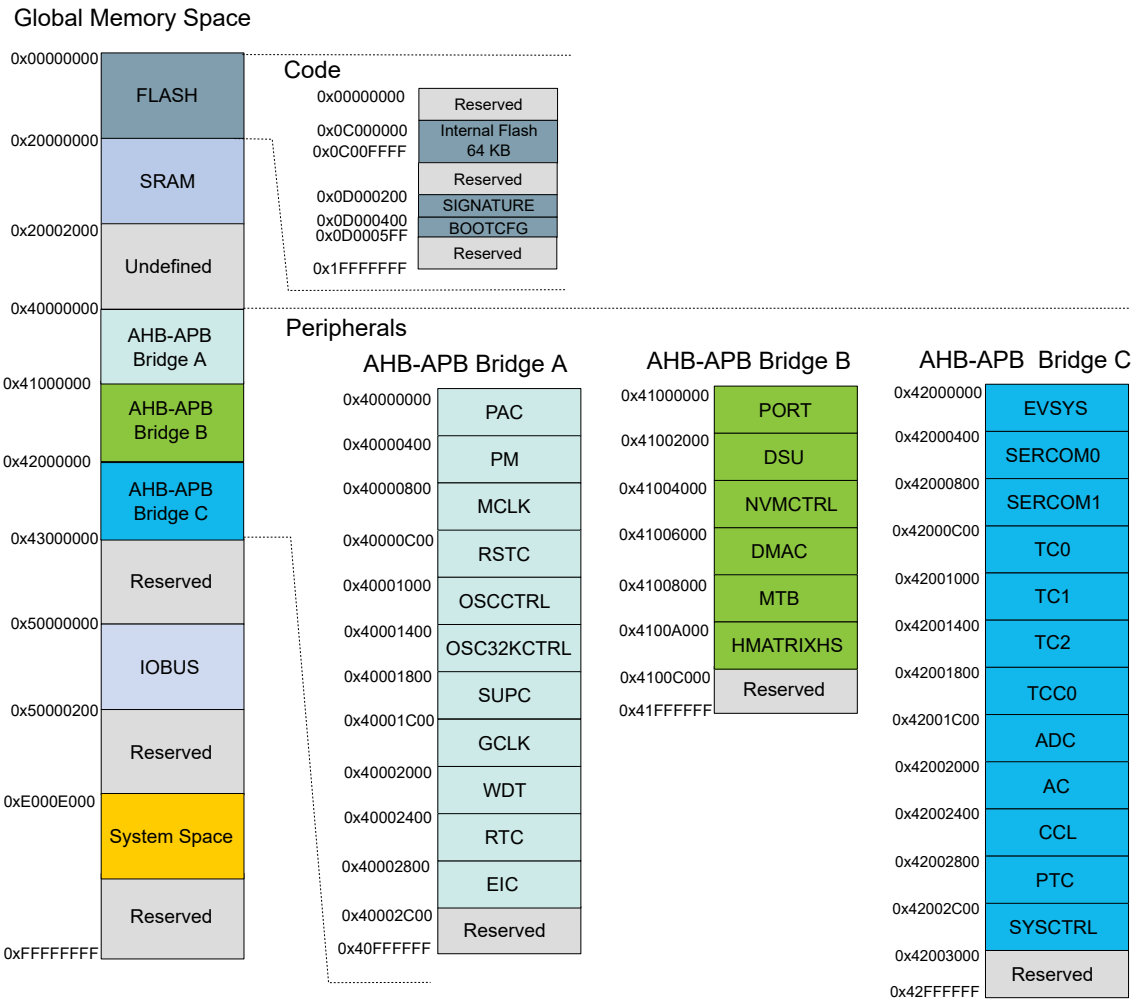
Vector Number	Interrupt Vector	Interrupt Source	Description
2	RTC	OVF	Overflow interrupt from RTC
		CMPn/ALARMn	Compare n interrupt (Mode 0 and 1)/Alarm n interrupt (Mode 2) from RTC
		PERn	Periodic Interval n interrupt from RTC
3	EIC	EXTINTn	External Interrupt n from the EIC
4	NVMCTRL	ERROR	Error interrupt from NVMCTRL
		READY	Flash ready interrupt from NVMCTRL
5	DMAC	TERR	Transfer error interrupt from DMAC
		TCMPL	Transfer complete interrupt from DMAC
		SUSP	Suspend interrupt from DMAC
6	EVSYS	OVRn	Overrun Channel n interrupt from EVSYS
		EVDn	Event Detected Channel n interrupt from EVSYS
7	SERCOM0	DRE	Data Register Empty from SERCOM0
		RXC	Receive Complete from SERCOM0
		TXC	Transmit Complete from SERCOM0
		RXS	Receive Start from SERCOM0
		CTSIC	Clear to Send Input Change from SERCOM0
		RXBRK	Received Break from SERCOM0
		SSL	SPI Select Low from SERCOM0
		DRDY	Data Ready from SERCOM0
		AMATCH	Address Match from SERCOM0
		PREC	Stop Received from SERCOM0
		ERROR	Error from SERCOM0
8	SERCOM1	DRE	Data Register Empty from SERCOM1
		RXC	Receive Complete from SERCOM1
		TXC	Transmit Complete from SERCOM1
		RXS	Receive Start from SERCOM1
		CTSIC	Clear to Send Input Change from SERCOM1
		RXBRK	Received Break from SERCOM1
		SSL	SPI Select Low from SERCOM1
		DRDY	Data Ready from SERCOM1
		AMATCH	Address Match from SERCOM1
		PREC	Stop Received from SERCOM1
		ERROR	Error from SERCOM1
9	TC0	OVF	Overflow interrupt from TC0
		ERR	Error interrupt from TC0
		MCn	Match or Capture Channel n=(0,1) interrupt from TC0
10	TC1	OVF	Overflow interrupt from TC1
		ERR	Error interrupt from TC1
		MCn	Match or Capture Channel n=(0,1) interrupt from TC1
11	TC2	OVF	Overflow interrupt from TC2
		ERR	Error interrupt from TC2
		MCn	Match or Capture Channel n=(0,1) interrupt from TC2

Table 4-3. Interrupt Line Mapping (continued)

Vector Number	Interrupt Vector	Interrupt Source	Description
12	TCC0	OVF	Overflow interrupt from TCC0
		TRG	Retrigger interrupt from TCC0
		CNT	Counter interrupt from TCC0
		ERR	Error interrupt from TCC0
		UFS	Non-Recoverable Update Fault interrupt from TCC0
		DFS	Non-Recoverable Debug Fault interrupt from TCC0
		FAULTx	Recoverable Fault x interrupt from TCC0
		FAULTn	Non-Recoverable Fault n interrupt from TCC0
		MCn	Match or Capture Channel n interrupt from TCC0
13	ADC0	TRIGOVR	Trigger Overrun interrupt from ADC0
		SAMPOVR	Sample Overwrite interrupt from ADC0
		RESOVR	Result Overwrite interrupt from ADC0
		RESRDY	Result Ready interrupt from ADC0
		WCMP	Window Comparator interrupt from ADC0
		SAMPRDY	Sample Ready interrupt from ADC0
14	AC	COMPn	Comparator n interrupt from AC
		WINn	Window n interrupt from AC
15-31	-	-	Reserved

4.3. Product Mapping

Figure 4-1. PIC32CM6408PL10 Product Mapping



The section labelled *System Space* maps the core peripherals of the Cortex-M0+ CPU. Refer to the *Cortex-M0+ Address Map* table in the *CPU and Architecture* chapter.

4.4. HMATRIXHS - High-Speed Bus System

4.4.1. Features

The High-Speed Bus System (HMATRIXHS) has the following features:

- Symmetric crossbar bus switch implementation
- Allows concurrent accesses from different hosts to different clients
- 32-bit data bus
- Operation at a 1-to-1 clock frequency with the bus hosts

4.4.2. Configuration

Figure 4-2. Host/Client Relation High-Speed Bus Matrix

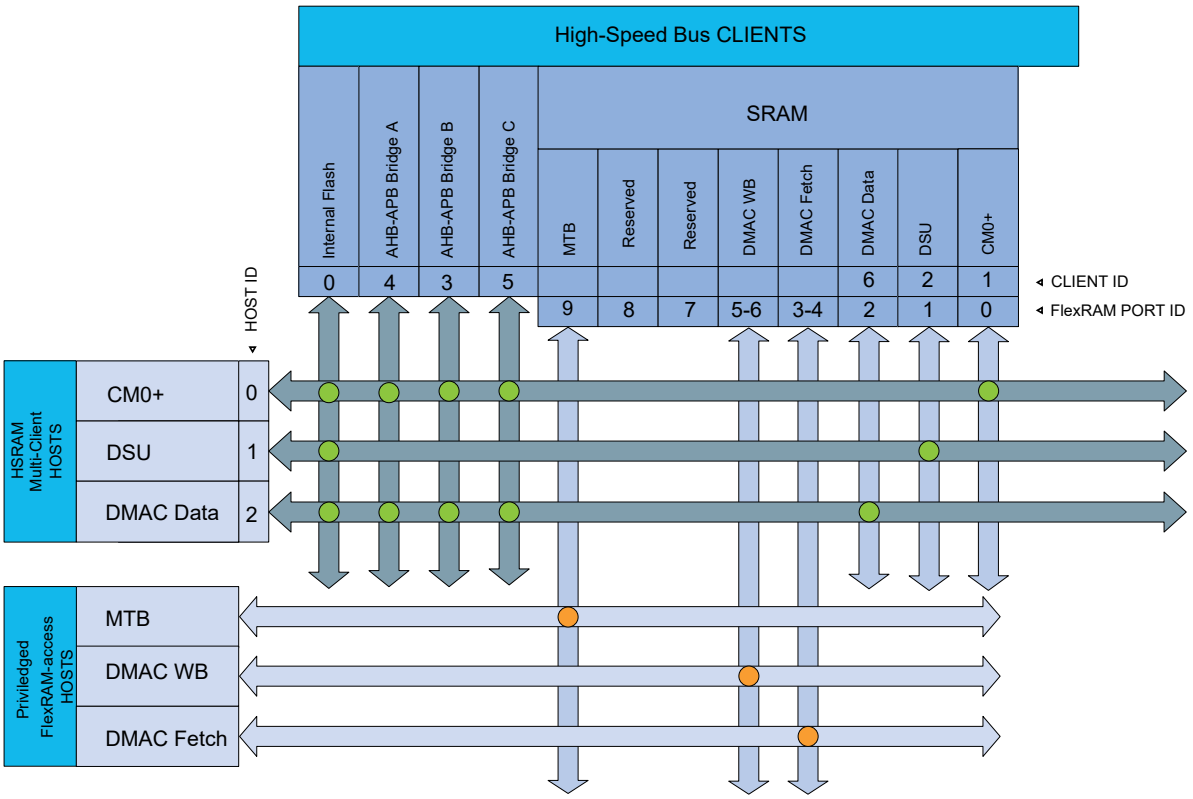


Table 4-4. Bus Matrix Hosts

Hosts (HSRAM)
Cortex M0+ Processor (CM0+)
Device Service Unit (DSU)
Direct Memory Access Controller/Data Access (DMAC)

Table 4-5. Bus Matrix Clients

Bus Matrix Clients
Internal Flash Memory
SRAM - CM0+ Access
SRAM - DSU Access
AHB-APB Bridge A
AHB-APB Bridge B
AHB-APB Bridge C
SRAM - DMAC Data Access

Table 4-6. SRAM Port Connections

SRAM Port Connection	Port ID	Connection Type
Cortex M0+ (CM0+) Processor	0	Bus Matrix
Device Service Unit (DSU)	1	Bus Matrix
Direct Memory Access Controller (DMAC) - Data Access	2	Bus Matrix
Direct Memory Access Controller (DMAC) - Fetch Access 0	3	Direct
Direct Memory Access Controller (DMAC) - Fetch Access 1	4	Direct

Table 4-6. SRAM Port Connections (continued)

SRAM Port Connection	Port ID	Connection Type
Direct Memory Access Controller (DMAC) - Write-Back Access 0	5	Direct
Direct Memory Access Controller (DMAC) - Write-Back Access 1	6	Direct
Reserved	7	
Reserved	8	
Micro Trace Buffer (MTB)	9	Direct

4.4.3. SRAM Quality of Service

To ensure that hosts with latency requirements receive sufficient priority when accessing SRAM, the different hosts can be configured to have a given priority for different types of access.

The Quality of Service (QoS) level is independently selected for each host accessing the SRAM. For any access to the SRAM, the SRAM also receives the QoS level. The QoS levels and their corresponding bit values for the QoS level configuration is shown in the following table.

Table 4-7. Quality of Service Level Configuration

Value	Name	Description
0x0	DISABLE	Background (no sensitive operation)
0x1	LOW	Sensitive Bandwidth
0x2	MEDIUM	Sensitive Latency
0x3	HIGH	Critical Latency

If a host is configured with QoS level DISABLE (0x0) or LOW (0x1), there will be a minimum latency of one cycle for the SRAM access.

The priority order for concurrent accesses is determined by two factors: First, the QoS level for the host, and second, a static priority given by the SRAM Port ID, as defined in SRAM Port Connections. The lowest port ID has the highest static priority.

The MTB has a fixed QoS level of HIGH (0x3).

The CPU QoS level can be written to or read from address 0x4100A110, bits [1:0]. Its reset value is 0x0.

Refer to the different host registers for configuring their QoS.

4.5. Micro Trace Buffer

4.5.1. Features

- Program flow tracing for the Cortex-M0+ processor
- MTB SRAM can be used for both trace and general purpose storage by the processor
- The position and size of the trace buffer in SRAM are configurable by software
- CoreSight compliant

4.5.2. Overview

When enabled, the MTB records changes in the program flow, reported by the Cortex-M0+ processor over the execution trace interface shared between the Cortex-M0+ processor and the CoreSight MTB-M0+. This information is stored as trace packets in the SRAM by the MTB. An off-chip debugger can extract the trace information using the Debug Access Port to read the trace information from the SRAM. The debugger can then reconstruct the program flow from this information.

The MTB simultaneously stores trace information in the SRAM and provides the processor with access to the SRAM. The MTB ensures that trace write accesses have priority over processor accesses.

The execution trace packet consists of a pair of 32-bit words that the MTB generates when it detects that the processor PC value changes non-sequentially. A non-sequential PC change can occur during branch instructions or exception entry. Refer to the *CoreSight MTB-M0+ Technical Reference Manual* for details on the MTB execution trace packet format.

Tracing is enabled when the MASTER.EN bit in the Host Trace Control Register is 1. There are various ways to set the bit to '1' to start tracing, or to '0' to stop tracing. Refer to the *CoreSight Cortex-M0+ Technical Reference Manual* for details on the Trace start and stop, and for a detailed description of the MTB's MASTER register. The MTB can be programmed to stop tracing automatically when the memory fills to a specified watermark level, or to start or stop tracing by writing directly to the MASTER.EN bit. If the watermark mechanism is not used and the trace buffer overflows, then the buffer wraps around, overwriting previous trace packets.

The base address of the MTB registers is 0x41008000, and this address is also listed in the CoreSight ROM table. The offset of each register from the base address is fixed, as defined by the *CoreSight MTB-M0+ Technical Reference Manual*. The MTB has four programmable registers to control the behavior of the trace features:

- **POSITION:** Contains the trace write pointer and the wrap bit
- **MASTER:** Contains the main trace enable bit and other trace control fields
- **FLOW:** Contains the WATERMARK address and the AUTOSTOP and AUTOHALT control bits
- **BASE:** Indicates where the SRAM is located in the processor memory map. This register is provided to enable auto-discovery of the MTB SRAM location, by a debug agent.

Refer to the *CoreSight MTB-M0+ Technical Reference Manual* for a detailed description of these registers.

4.6. Reset Sources

Generally, PIC32CM devices can distinguish between these types of reset:

1. Power Resets
   - POR: Power On Reset
   - BOR: Brownout Detector Reset
2. User Resets
   - WDT: Watchdog Timer
   - $\overline{\text{RESET}}$ pin
   - CPU - both `sysreset` request and core lockup

Important: On devices of the PIC32CM PL10 family, all User Reset sources are treated equally like Power Reset sources.

5. Memories

5.1. Embedded Memories

- Internal high-speed Flash
- Internal high-speed SRAM with single-cycle access at full speed

5.2. Physical Memory

Table 5-1. Memory Properties

Memory		StartAddress	PIC32CM6408PL10
Embedded Flash	Size	0x00000000	64 KB
	Page number		128
	Page size		512B
Embedded high-speed SRAM		0x20000000	8 KB

The high-speed bus is implemented as a bus matrix. All high-speed bus addresses are fixed and are never remapped in any way, even during boot. See the Product Mapping chapter for the logical organization and addresses.

5.3. Register Properties

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

Registers can have one or multiple properties, as indicated in the register description:

PAC Write-Protection

Some registers can be write-protected by the Peripheral Access Controller (PAC).

PAC write protection is indicated by the “PAC Write-Protection” property in the register description. For more details, refer to the *PAC - Peripheral Access Controller* chapter.
Note: PAC Write-Protection is optional.

Local Write-Protection

Many peripherals offer a local, key-based write-protection mechanism for registers with write access. These peripherals have a Write Protection Control (WPCTRL) register, and the registers that can be protected bear the property “Local Write-Protection”.

Follow these steps to configure the optional local write-protection:

- When writing to the WPCTRL register, the Write Protection Key (WPKEY) bit field must contain the specific KEY value.
- The local write-protection is enabled by writing a `'1'` to the Write Protection Enable (WPEN) bit in the WPCTRL register.
- The WPCTRL register itself can be protected by writing a `'1'` to the Write Protection Lock (WPLCK) bit. This bit can be cleared by a reset, but not by the application.

Note: The optional local write-protection mechanism is not preventing debugger access.

Enable-Protected

Some registers or bit fields can only be written when the peripheral is disabled, denoted by the “Enable-Protected” property in the register description.

Note: Enable-Protection is not optional.

Read-Synchronized, Write-Synchronized

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers or bit fields need to be synchronized when being written or read. Required write-synchronization is indicated by the “Write-Synchronized” property in the register description. For more details, refer to the *Register Synchronization* section in the *CS - Clock System* chapter.

5.4. SIGNATURE

Each device has a unique 128-bit serial number, which is a concatenation of four 32-bit words from the Serial Number fuses SERNUM[3:0]. The uniqueness of the serial number is ensured only when all 128 bits are used.

For general device selection purposes, such as by a debugger, the pin/memory configuration of the device is also stored in the Device ID register of the Device Service Unit (DSU).

5.4.1. Register Summary - SIGNATURE

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x1F	Reserved									
0x20	SERNUM0	7:0	SERNUM0[7:0]							
		15:8	SERNUM0[15:8]							
		23:16	SERNUM0[23:16]							
		31:24	SERNUM0[31:24]							
0x24 ... 0x27	Reserved									
0x28	SERNUM1	7:0	SERNUM1[7:0]							
		15:8	SERNUM1[15:8]							
		23:16	SERNUM1[23:16]							
		31:24	SERNUM1[31:24]							
0x2C ... 0x2F	Reserved									
0x30	SERNUM2	7:0	SERNUM2[7:0]							
		15:8	SERNUM2[15:8]							
		23:16	SERNUM2[23:16]							
		31:24	SERNUM2[31:24]							
0x34 ... 0x37	Reserved									
0x38	SERNUM3	7:0	SERNUM3[7:0]							
		15:8	SERNUM3[15:8]							
		23:16	SERNUM3[23:16]							
		31:24	SERNUM3[31:24]							

5.4.1.1. Serial Number 0

Name: SERNUM0
Offset: 0x20
Reset: 0xXXXXXXXXXXXXXXXX
Property: R

Each device has a unique 128-bit serial number, which is a concatenation of four 32-bit words from the Serial Number fuses SERNUM[3:0]. The uniqueness of the serial number is ensured only when all 128 bits are used.

Bit	31	30	29	28	27	26	25	24
	SERNUM0[31:24]							
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
	SERNUM0[23:16]							
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
	SERNUM0[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
	SERNUM0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 31:0 – SERNUM0[31:0] Serial Number 0

5.4.1.2. Serial Number 1

Name: SERNUM1
Offset: 0x28
Reset: 0xXXXXXXXXXXXXXXXX
Property: R

Each device has a unique 128-bit serial number, which is a concatenation of four 32-bit words from the Serial Number fuses SERNUM[3:0]. The uniqueness of the serial number is ensured only when all 128 bits are used.

Bit	31	30	29	28	27	26	25	24
	SERNUM1[31:24]							
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
	SERNUM1[23:16]							
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
	SERNUM1[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
	SERNUM1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 31:0 – SERNUM1[31:0] Serial Number 1

5.4.1.3. Serial Number 2

Name: SERNUM2
Offset: 0x30
Reset: 0xXXXXXXXXXXXXXXXX
Property: R

Each device has a unique 128-bit serial number, which is a concatenation of four 32-bit words from the Serial Number fuses SERNUM[3:0]. The uniqueness of the serial number is ensured only when all 128 bits are used.

Bit	31	30	29	28	27	26	25	24
	SERNUM2[31:24]							
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
	SERNUM2[23:16]							
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
	SERNUM2[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
	SERNUM2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 31:0 – SERNUM2[31:0] Serial Number 2

5.4.1.4. Serial Number 3

Name: SERNUM3
Offset: 0x38
Reset: 0xXXXXXXXXXXXXXXXX
Property: R

Each device has a unique 128-bit serial number, which is a concatenation of four 32-bit words from the Serial Number fuses SERNUM[3:0]. The uniqueness of the serial number is ensured only when all 128 bits are used.

Bit	31	30	29	28	27	26	25	24
	SERNUM3[31:24]							
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
	SERNUM3[23:16]							
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
	SERNUM3[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
	SERNUM3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 31:0 – SERNUM3[31:0] Serial Number 3

5.5. BOOTCFG - Boot Configuration Fuses

The BOOTCFG fuses contain configuration details for the device and essential peripherals, such as BOD or WDT, that are applied after a device reset and during start-up.

When the Sequence Number (SEQNUM) fuse is not 0xFFFFFFFF at start-up and after a Reset, the contents of the other BOOTCFG fuses are copied to the respective peripheral registers and bit fields, as described in the following fuse descriptions. Follow these steps to alter the device configuration:

1. Write the desired values to the fuses.
2. Reset the device. The values are copied from the fuses to the peripheral registers.
3. (Optional) To read the actual configuration, read the respective registers of the SYSCTRL, WDT, and SUPC.

Attention: When the SEQNUM fuse is not 0xFFFFFFFF, *all* of the BOOTCFG fuse values are copied to the peripheral registers. For this reason, all BOOTCFG fuses must be written to desired values—the factory values of these fuses have no practical meaning.
When the value of the Sequence Number (SEQNUM) fuse is equal to 0xFFFFFFFF, *none* of the BOOTCFG fuse values are copied to any of the peripheral registers at startup/reset. Instead, the respective reset values given in the peripheral registers' descriptions are used.

Note: The entire BOOTCFG section is unaffected by the Chip Erase command. For this reason, the residual space of the BOOTCFG section (after the Fuses described below) can be used to store persistent application data, such as serial numbers or application keys.

5.5.1. Register Summary - BOOTCFG

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	SEQNUM	7:0	SEQNUM[7:0]							
		15:8	SEQNUM[15:8]							
		23:16	SEQNUM[23:16]							
		31:24	SEQNUM[31:24]							
0x04 ... 0x07	Reserved									
0x08	BOOTPROT	7:0						BOOTPROT[2:0]		
		15:8								
		23:16								
		31:24								
0x0C ... 0x0F	Reserved									
0x10	WDTCFG	7:0					ALWAYSON	WEN	ENABLE	
		15:8	WINDOW[3:0]				PER[3:0]			
		23:16					EWOFFSET[3:0]			
		31:24								
0x14 ... 0x17	Reserved									
0x18	BODCFG	7:0		RUNSTDBY	STDBYCFG				ENABLE	
		15:8				SAMPFREQ				ACTCFG
		23:16							LEVEL[1:0]	
		31:24	WRTLOCK				VLMCFG[1:0]		VLMLVL[1:0]	
0x1C ... 0x1F	Reserved									
0x20	USERCFG	7:0	CRCBOOT	CRCSEL						MVIOMODE
		15:8						SUT[2:0]		
		23:16								
		31:24								
0x24 ... 0x27	Reserved									
0x28	BOOT_GPIOSEL	7:0	ENABLE			GPIOPINSEL[4:0]				
		15:8					GPIOPORTSEL[3:0]			
		23:16				SLEWLIM	ODRAIN		POL	
		31:24								

5.5.1.1. Sequence Number Fuse

Name: SEQNUM
Offset: 0x0000
Default: 0xFFFFFFFF
Property: R/W

The default value given in this fuse description is the factory-programmed value and must not be mistaken for the Reset value.

Bit	31	30	29	28	27	26	25	24
	SEQNUM[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	SEQNUM[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	SEQNUM[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1

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

Bits 31:0 – SEQNUM[31:0] Sequence Number
When the value of this bit field is equal to `0xFFFFFFFF` (factory default) at start-up, none of the BOOTCFG fuse values is copied to any of the peripheral registers at startup or reset. Instead, the respective reset values given in the peripheral registers' descriptions are used.
When this bit field is **not** `0xFFFFFFFF` at start-up, the user-defined Boot Configuration (BOOTCFG) fuse values are applied.
The value of this fuse can be used by the bootloader or application for version tracking or similar purposes, as long as it is not `0xFFFFFFFF`.

5.5.1.2. Boot Protection Fuse

Name: BOOTPROT
Offset: 0x0008
Default: 0xFFFFFFFF
Property: R/W

The bit group values of this fuse are written to the corresponding bit groups of the NVM Parameters (PARAM) register in the Non-Volatile Memory Controller peripheral (NVMCTRL) at start-up. The default value given in this fuse description is the factory-programmed value and must not be mistaken for the Reset value.

Bit	31	30	29	28	27	26	25	24
Access								
Default								

Bit	23	22	21	20	19	18	17	16
Access								
Default								

Bit	15	14	13	12	11	10	9	8
Access								
Default								

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

Bits 2:0 – BOOTPROT[2:0] Boot Protection
This bit field value is loaded into the Boot Loader Size (BOOTPROT) bit field of the NVMCTRL.PARAM register during Reset. Refer to the *NVMCTRL - Non-Volatile Memory Controller* section for further details.

Value	Name	Description
0x0	SIZE_32768BYTES	32768 bytes are protected
0x1	SIZE_16384BYTES	16384 bytes are protected
0x2	SIZE_8192BYTES	8192 bytes are protected
0x3	SIZE_4096BYTES	4096 bytes are protected
0x4	SIZE_2048BYTES	2048 bytes are protected
0x5	SIZE_1024BYTES	1024 bytes are protected
0x6	SIZE_512BYTES	512 bytes are protected
0x7	SIZE_0BYTES	0 bytes are protected (default)

5.5.1.3. Watchdog Timer Configuration Fuse

Name: WDTCFG
Offset: 0x0010
Default: 0xFFFFFFFF
Property: R/W

The bit group values of this fuse are written to the corresponding bit groups of the Control A (CTRLA), Configuration (CONFIG), and Early Warning Control (EWCTRL) registers in the Watchdog Timer peripheral (WDT) at start-up.

The default value given in this fuse description is the factory-programmed value and must not be mistaken for the Reset value.

Bit	31	30	29	28	27	26	25	24
Access								
Default								

Bit	23	22	21	20	19	18	17	16
					EWOFFSET[3:0]			
Access					R/W	R/W	R/W	R/W
Default					1	1	1	1

Bit	15	14	13	12	11	10	9	8
	WINDOW[3:0]				PER[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
					ALWAYSON	WEN	ENABLE	
Access					R/W	R/W	R/W	
Default					1	1	1	

Bits 19:16 – EWOFFSET[3:0] Early Warning Interrupt Time Offset Fuse
This bit field value is loaded into the Early Warning Interrupt Time Offset (EWOFFSET) bit field of the WDT.EWCTRL register during Reset. Refer to the *WDT - Watchdog Timer* section for further details.

Value	Name	Description
0x0	CYC8	8 clock cycles
0x1	CYC16	16 clock cycles
0x2	CYC32	32 clock cycles
0x3	CYC64	64 clock cycles
0x4	CYC128	128 clock cycles
0x5	CYC256	256 clock cycles
0x6	CYC512	512 clock cycles
0x7	CYC1024	1024 clock cycles
0x8	CYC2048	2048 clock cycles
0x9	CYC4096	4096 clock cycles
0xA	CYC8192	8192 clock cycles
0xB - 0xF	—	Reserved

Bits 15:12 – WINDOW[3:0] Window
This bit field value is loaded into the WINDOW bit field of the WDT.CONFIG register during Reset. Refer to the *WDT - Watchdog Timer* section for further details.

Value	Name	Description
0x0	CYC8	8 clock cycles
0x1	CYC16	16 clock cycles
0x2	CYC32	32 clock cycles
0x3	CYC64	64 clock cycles
0x4	CYC128	128 clock cycles
0x5	CYC256	256 clock cycles
0x6	CYC512	512 clock cycles
0x7	CYC1024	1024 clock cycles
0x8	CYC2048	2048 clock cycles
0x9	CYC4096	4096 clock cycles
0xA	CYC8192	8192 clock cycles
0xB	CYC16384	16384 clock cycles
Other	—	Reserved

Bits 11:8 – PER[3:0] Time-Out Period

This bit field value is loaded into the PER bit field of the Watchdog Configuration (WDT.CONFIG) register during Reset. Refer to the *WDT - Watchdog Timer* section for further details.

Value	Name	Description
0x0	CYC8	8 clock cycles
0x1	CYC16	16 clock cycles
0x2	CYC32	32 clock cycles
0x3	CYC64	64 clock cycles
0x4	CYC128	128 clock cycles
0x5	CYC256	256 clock cycles
0x6	CYC512	512 clock cycles
0x7	CYC1024	1024 clock cycles
0x8	CYC2048	2048 clock cycles
0x9	CYC4096	4096 clock cycles
0xA	CYC8192	8192 clock cycles
0xB	CYC16384	16384 clock cycles
Other	—	Reserved

Bit 3 – ALWAYSON Always On

This bit value is loaded into the ALWAYSON bit field of the WDT.CTRLA register during Reset. Refer to the *WDT - Watchdog Timer* section for further details.

Value	Description
0	The WDT can be enabled and disabled through the ENABLE bit
1	The WDT is enabled and can only be disabled by a Power-on Reset (POR)

Bit 2 – WEN Window Mode Enable

This bit value is loaded into the WEN bit field of the WDT.CTRLA register during Reset. Refer to the *WDT - Watchdog Timer* section for further details.

Value	Description
0	Window mode is disabled
1	Window mode is enabled

Bit 1 – ENABLE Enable

This bit value is loaded into the ENABLE bit in the WDT.CTRLA register during Reset. Refer to the *WDT - Watchdog Timer* section for further details.

Value	Description
0	The WDT is disabled
1	The WDT is enabled

5.5.1.4. BOD Configuration Fuse

Name: BODCFG
Offset: 0x018
Default: 0xFFFFFFFF
Property: R/W

The bit group values of this fuse are written to the corresponding bit groups of the BOD VDD Control (BODVDD) register in the Supply Controller (SUPC) at start-up.

The default value given in this fuse description is the factory-programmed value and must not be mistaken for the Reset value.

Bit	31	30	29	28	27	26	25	24
	WRTLOCK				VLMCFG[1:0]		VLMLVL[1:0]	
Access	R/W				R/W	R/W	R/W	R/W
Default	1				1	1	1	1

Bit	23	22	21	20	19	18	17	16
							LEVEL[1:0]	
Access							R/W	R/W
Default							1	1

Bit	15	14	13	12	11	10	9	8
				SAMPFREQ				ACTCFG
Access				R/W				R/W
Default				1				1

Bit	7	6	5	4	3	2	1	0
		RUNSTDBY	STDBYCFG				ENABLE	
Access		R/W	R/W				R/W	
Default		1	1				1	

Bit 31 – WRTLOCK Write Lock
This bit value is loaded into the WRTLOCK bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Description
0x0	The BODVDD configuration is not locked
0x1	The BODVDD configuration is locked

Bits 27:26 – VLMCFG[1:0] Voltage Level Monitor Interrupt Configuration
This bit value is loaded into the VLMCFG bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Name	Description
0x0	FALLING	VDD falls below VLM threshold
0x1	RISING	VDD rises above VLM threshold
0x2	BOTH	VDD crosses VLM threshold

Bits 25:24 – VLMLVL[1:0] Voltage Level Monitor Level
This bit field value is loaded into the VLMLVL bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Name	Description
0x0	OFF	VLM disabled
0x1	5ABOVE	VLM threshold is 5% above BOD threshold

Value	Name	Description
0x2	15ABOVE	VLM threshold is 15% above BOD threshold
0x3	25ABOVE	VLM threshold is 25% above BOD threshold

Bits 17:16 – LEVEL[1:0] BOD Level

This bit field value is loaded into the LEVEL bit field in the SUPC.BODVDD register during Reset.

Notes:

1. Refer to the *BOD and POR Characteristics* in the *Electrical Characteristics* section for further details.
2. Values in the description are typical values.

Value	Name	Description
0x0	BODLEVEL0	1.90V
0x1	BODLEVEL1	2.45V
0x2	BODLEVEL2	2.70V
0x3	BODLEVEL3	2.85V

Bit 12 – SAMPFREQ BOD Sampling Frequency

This bit value is loaded into the SAMPFREQ bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Name	Description
0	128HZ	The sample frequency is 128 Hz
1	32HZ	The sample frequency is 32 Hz

Bit 8 – ACTCFG BOD Operation Mode in Active Mode

This bit value is loaded into the ACTCFG bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Name	Description
0	ENABLE	In active mode, the BOD operates in continuous mode
1	SAMPLE	In active mode, the BOD operates in sampling mode

Bit 6 – RUNSTDBY Run in Standby

This bit value is loaded into the RUNSTDBY bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Description
0x0	The BOD is not running in Standby sleep mode
0x1	The BOD is running in Standby sleep mode if enabled by the ENABLE bit

Bit 5 – STDBYCFG BOD Configuration in Standby Sleep Mode

This bit value is loaded into the STDBYCFG bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Name	Description
0x0	CONT	In Standby sleep mode, the BOD is configured in continuous mode
0x1	SAMP	In Standby sleep mode, the BOD is configured in sampling mode

Bit 1 – ENABLE BOD Enable

This bit value is loaded into the ENABLE bit in the SUPC.BODVDD register during Reset. Refer to the *SUPC - Supply Controller* chapter for further details.

Value	Name	Description
0	DISABLE	BOD is disabled
1	ENABLE	BOD is enabled

5.5.1.5. User Configuration Fuse

Name: USERCFG
Offset: 0x0020
Default: 0xFFFFFFFF
Property: R/W

The bit field values of this fuse are written to the corresponding bit fields of the User Configuration register (SYSCTRL.USERCFG) in the System Controller and the Multivoltage I/O register (SUPC.MVIO) in the Supply Controller at start-up. The default value given in this fuse description is the factory-programmed value and must not be mistaken for the Reset value.

Bit	31	30	29	28	27	26	25	24
Access								
Default								

Bit	23	22	21	20	19	18	17	16
Access								
Default								

Bit	15	14	13	12	11	10	9	8
						SUT[2:0]		
Access						R/W	R/W	R/W
Default						1	1	1

Bit	7	6	5	4	3	2	1	0
	CRCBOOT	CRCSEL						MVIOMODE
Access	R/W	R/W						R/W
Default	1	1						1

Bits 10:8 – SUT[2:0] Start-up Time
This bit field value is loaded into the SUT bit field in the USERCFG register of the System Controller (SYSCTRL.USERCFG) during Reset. Refer to the *SYSCTRL - System Controller* section for further details.

Value	Name	Description
0x0	0MS	0 ms
0x1	1MS	1 ms
0x2	2MS	2 ms
0x3	4MS	4 ms
0x4	8MS	8 ms
0x5	16MS	16 ms
0x6	32MS	32 ms
0x7	64MS	64 ms

Bit 7 – CRCBOOT CRC Boot
This bit value is loaded into the CRCBOOT bit in the USERCFG register of the System Controller (SYSCTRL.USERCFG) during Reset. Refer to the *SYSCTRL - System Controller* section for further details.

Value	Name	Description
0	DISABLE	No CRC
1	ENABLE	CRC of the Boot section

Bit 6 – CRCSEL CRC Polynomial Selection

This bit value is loaded into the CRCSEL bit in the USERCFG register of the System Controller (SYSCTRL.USERCFG) during Reset. Refer to the *SYSCTRL - System Controller* section for further details.

Value	Name	Description
0	CRC16	CRC-16-CCITT
1	CRC32	CRC-32 (IEEE 802.3)

Bit 0 – MVIOMODE MVIO Operation Mode

This bit value is loaded into the MODE bit in the MVIO register of the Supply Controller (SUPC.MVIO) during Reset. Refer to the *SUPC - Supply Controller* section for further details.

Value	Name	Description
0	DUAL	MVIO operating in dual supply mode
1	SINGLE	MVIO operating in single supply mode

5.5.1.6. Boot External Notification I/O Pin Fuse

Name: BOOT_GPIOSEL
Offset: 0x0028
Default: 0xFFFFFFFF
Property: R/W

The bit fields in this fuse select the pin and its signal properties for the Boot External Notification signal at start-up.

The default value given in this fuse description is the factory-programmed value and must not be mistaken for the Reset value.

Bit	31	30	29	28	27	26	25	24
Access								
Default								

Bit	23	22	21	20	19	18	17	16
				SLEWLIM	ODRAIN		POL	
Access				R/W	R/W		R/W	
Default				1	1		1	

Bit	15	14	13	12	11	10	9	8
					GPIOPORTSEL[3:0]			
Access					R/W	R/W	R/W	R/W
Default					1	1	1	1

Bit	7	6	5	4	3	2	1	0
	ENABLE			GPIOPINSEL[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Default	1			1	1	1	1	1

Bit 20 – SLEWLIM Slew Rate Control for Boot External Notification Signal
This bit controls whether the slew rate is limited.

Value	Name	Description
0	FALSE	The slew rate is not limited (fast rise/fall time operation)
1	TRUE	The slew rate is limited.

Bit 19 – ODRAIN Open Drain Control for Boot External Notification Signal
This bit controls whether the fault source is of normal drive type or open-drain drive type. Together with the Pin Output Level for Boot External Notification Signal (POL) bit, the resulting behavior of the pin is as follows:

	ODRAIN = 0	ODRAIN = 1
POL = 0	Tristate	Drive high
POL = 1	Drive low	Drive low

Value	Name	Description
0	FALSE	The selected fault source is open drain drive
1	TRUE	The selected fault source is normal drive

Bit 17 – POL Select Polarity in Normal Drive Mode for Boot External Notification Signal
This bit selects whether the Boot External Notification signal is active-low or active-high. Refer also to the Open Drain Control for Boot External Notification Signal (ODRAIN) bit description, because ODRAIN and POL are interdependent properties for the pin.

Value	Name	Description
0	FALSE	The selected fault source is active-high
1	TRUE	The selected fault source is active-low

Bits 11:8 – GPIOPORTSEL[3:0] GPIO Port Select for Boot External Notification Signal
This bit field selects the I/O pin port used for the Boot External Notification signal.
Note: Select only a port that exists on the device.

Value	Name	Description
0x0	PORTA	Port A
0x1	PORTB	Port B
0x2	PORTC	Port C
0x3	PORTD	Port D
Other	-	No Boot External Notification signal used

Bit 7 – ENABLE Enable Boot External Notification Signal
This bit enables the Boot External Notification signal.

Value	Name	Description
0	DISABLE	The Boot External Notification signal is disabled
1	ENABLE	The Boot External Notification signal is enabled

Bits 4:0 – GPIOPINSEL[4:0] GPIO Pin Select for Boot External Notification Signal
This bit field selects the I/O pin used for the Boot External Notification signal.
Note: Select only a pin that exists on the device.

Value	Name	Description
0x00	PIN00	Pin 0
0x01	PIN01	Pin 1
...	...	...
0x1E	PIN30	Pin 30
0x1F	PIN31	Pin 31

6. BootROM - Boot ROM

6.1. Features

- Device Integrity Check to Verify Authenticity and Configuration
- Device Lock to Prevent any External Read/Write Access (Programming and Debugging Interface Disable - PDID)
- Boot Failure Indication via I/O Pin
- Boot Interactive Mode (IMODE) for Programming and Maintenance Operation With a Connected Debugger
- Deterministic Boot Process: ROM Code Executes Without Debugger Access Using a Dedicated Clock Source

6.2. Overview

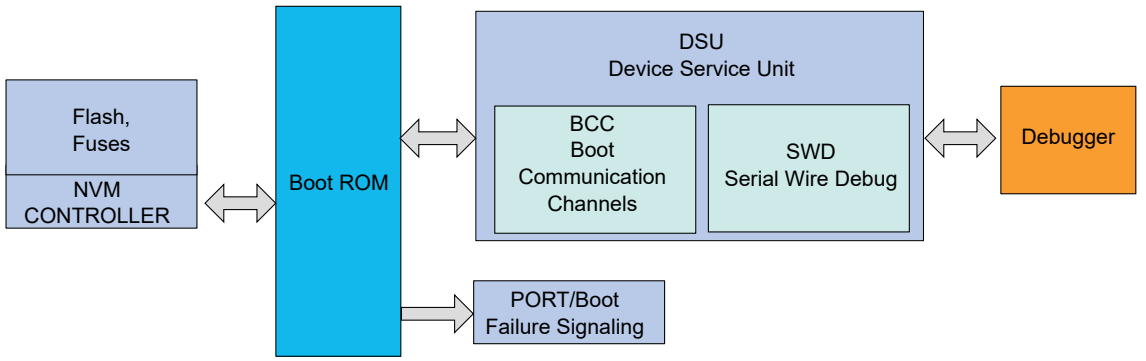
The Boot ROM is the entry point of the device following any type of reset. It performs a series of vital checks, including image validation and configuration verification. If these checks pass, the CPU loads the initial Program Counter (PC) and Stack Pointer (SP) values from Flash memory and begins executing the user code.

The Boot ROM also provides the Interactive mode, allowing both hot-plugging and cold-plugging of a debugger and programming.

Note:
Throughout this document, *Debugger* shall be interpreted as SWD-Host (and vice-versa) that is supporting this device.

6.3. Block Diagram

Figure 6-1. Boot ROM Block Diagram



6.4. Functional Description

6.4.1. Boot ROM Operation at Startup

The Boot ROM operation at startup can be logically divided into the following functional stages:

1. **Integrity Check of the Program Flash Memory (Optional)**
   If the CRCBOOT bit in the User Configuration (BOOTCFG.USERCFG) fuse is set, the Boot ROM will perform a CRC-32 scan on the Program Flash Memory. If any integrity violation is detected, the Boot ROM halts further execution, reports the error in the DSU.BCC registers(1), and places the device into a known safe state called Interactive mode (IMODE)(2). Otherwise, the Boot ROM proceeds to the next stage.

2. **Debugger Probe Detection**
   The Boot ROM is designed to asynchronously accept a cold-plug-in request from the debugger (i.e. SWD host) at any time during the device's operating life cycle. The Boot ROM automatically detects debugger attachement to the device as soon as the debugger follows the cold-plug-in sequence. Refer to the *DSU - Device Service Unit* chapter and the *Programming Specifications* section in this chapter for details on debugger operation.
3. **Application of User Settings and Finalization**
   Applies user-selected fuse configurations to the respective peripherals. Refer to the *BOOTCFG - Boot Configuration Fuses* chapter for more details.
   At the end of the boot process, the OSCHF at 4 MHz is selected as the default clock for GCLK0 and, consequently, MCLK. See the chapters on the *Clock System, GCLK, MCLK* for details.
4. **Transition to First Mutable Application Executable**
   Transfers control to the first mutable application executable by loading the PC and SP values only if the boot operation is successful. Refer to the *Product Mapping* and *Memories* chapters for details.

Notes:

1. If the Boot External Notification I/O Pin (BOOT_GPIOSEL) fuse in the BOOTCFG memory section is written, a fault signal is asserted on the configured pin.
2. When no debugger probe is detected after entering IMODE, the device will reset.

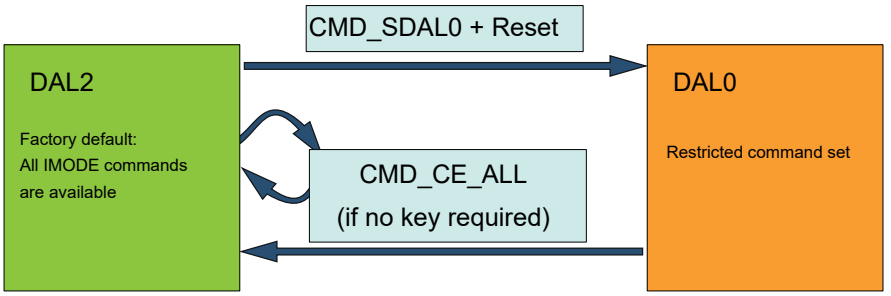
6.4.2. Debugger Access Level (DAL)

The Boot ROM supports two levels of debugger access, known as **Device Access Levels (DAL)**. These levels define the access a debugger has to the device during and after boot:

- **DAL2**: Full access. The device is completely open, allowing all operations supported by the device. This level is typically used during development and debugging.
- **DAL0**: Locked state. The device is highly restricted. Debug access is limited and defined by the user prior to setting DAL to 0.

The active DAL is determined by the value programmed into an internal memory location. Once configured, the DAL governs the debugger's ability to interact with the device, including access to memory, peripherals, and Boot ROM features:

Figure 6-2. Changing Between The Debugger Access Levels



1. The device is factory-programmed with **DAL2** to allow the user to configure the device according to end application requirements. All commands of the Interactive mode IMODE are accessible, allowing full debugging and programming access.
2. Once application development is finished, the device can be configured to **DAL0** by issuing the Select DAL 0 command `CMD_SDAL0` in IMODE and resetting the device. Now, only a restricted set of commands is available in IMODE.
   Note: Refer to the sections *Programming and Debug Interface Disable (PDID)* and *Programming Specification* for more information on how to restrict commands and access.

3. Issuing the Chip Erase All command `CMD_CE_ALL` in IMODE will clear all Flash and volatile memories (but not the BOOTCFG and calibration fuses) and, eventually, set the DAL back to DAL2.

> **CAUTION** It is possible to lock the `CMD_CE_ALL` command. When the device is set to DAL0 and the `CMD_CE_ALL` command is locked, the device may become **permanently unrecoverable** if the user does not retain access to the Key Value fuse required to unlock `CMD_CE_ALL`. Refer to the *Programming Specification* section for more information.

> **WARNING** Carefully plan and validate the fuse configurations before transitioning the device to DAL0, especially in production environments where recovery options are limited.

6.4.3. Operating Modes

The Boot ROM operation is logically divided into the following four modes, each serving a distinct purpose during the device startup and debug process:

1. **Boot Mode**
   This mode is entered immediately after any type of Reset when no debugger probe is present. In this mode, the Boot ROM performs system initialization, fuse integrity checks, and attempts to boot the device using the user-configured settings.
   Under normal operation, the device will then transition to *Mission mode*.
   When a boot error occurs, a boot failure signal can be transmitted (see the *Boot Failure Signal* section), and the device resets.
   When a boot error occurs and a debugger is connected, *Interactive mode* is invoked.
2. **Interactive Mode (IMODE)**
   The Boot ROM enters IMODE under either of the following conditions:
   - A boot failure occurs and a debugger is connected
   - A debugger explicitly requests entry into IMODE to perform advanced operations

   The IMODE allows to perform tasks such as:
   - Full chip erase
   - Device locking (setting DAL = 0)
   - Configuration Flash Memory (CFM) integrity checks

   Some tasks may be prohibited due to deliberate locking or protection measures.
   IMODE is always exited by a Reset. The cause of the Reset can be an IMODE command, a hard fault, the absence of a connected debugger, or any regular Reset source. The device will then enter *Boot mode*.
3. **Mission Mode**
   This mode allows the device to begin execution from the first mutable application executable location in Flash.
   The device remains in *Mission mode* until it is reset for any reason. After a Reset, the device will enter *Boot mode*.
4. **Park Mode**

This mode is invoked under certain conditions after a cold-plugging sequence. *Park mode* is specifically designed to enable the debugger to offer additional debug features. Refer to the *Programming Specifications* section for information on invoking and exiting this mode.

6.4.4. Boot Failure Signal

The Boot ROM provides a mechanism to indicate boot failure through a user-configurable Boot Failure Signal, enabled via the Boot External Notification I/O Pin (BOOT_GPIOSEL) fuse located in the Boot Configuration (BOOTCFG) fuse.

Users can select a specific I/O pin to serve as the fault indicator for their application. If a boot failure occurs (e.g., due to a fuse integrity check failure or invalid configuration), the Boot ROM asserts the selected I/O pin to signal the fault condition.

Note: By default, the Boot Failure Signal is **disabled** (BOOT_GPIOSEL = NONE). To enable this feature, users must program the BOOT_GPIOSEL fuse with a valid I/O selection. Refer to the *BOOTCFG - Boot Configuration Fuse* and *Pinout* chapters for details on available I/O options.

Important: Once the Boot ROM transfers control to the first mutable application code, it no longer manages or controls the fault signal pin. The application is responsible for any further de-assertion of the signal and configuration of the pin.

6.4.5. Sequence Number Fuse Check

The Boot ROM reads the Sequence Number (SEQNUM) fuse in the Boot Configuration (BOOTCFG) memory section automatically upon any type of reset to determine the preferred boot configuration:

1. When BOOTCFG.SEQNUM is *not* 0xFFFFFFFF, the Boot ROM copies the contents of the BOOTCFG fuses to the respective peripherals registers.
2. When BOOTCFG.SEQNUM reads 0xFFFFFFFF, the device is booted using the factory defaults of the peripheral registers, and no pin is asserted for the Boot Failure Signal.

When the value of this fuse is not all-F, it can be used by the bootloader or application for version tracking, as a time stamp, or for similar purposes.

6.4.6. Debugger Usage

The Interactive mode (IMODE) serves for debugger interaction. In IMODE, the debugger can send commands and receive status messages and error codes.

Involuntary Entry to IMODE

IMODE is only entered involuntarily when a debugger is connected and the Boot ROM experiences any error and/or Fault during the boot operation that prevents the device from executing the first mutable application code. The device may encounter unsafe operating conditions due to various reasons, including but not limited to:

- Invalid or inconsistent device configuration
- Memory Faults
- Unintended or undefined device states

In such scenarios, the Boot ROM proactively halts the boot process and transitions the device into IMODE as a known safe state. See also the *Programming Specifications* section and the *DSU - Device Service Unit* chapter for details.
Note: When the Boot ROM encounters unsafe operating conditions and no debugger is connected, it will signal a Fault on a pin (if configured; see the *Boot Failure Signal* section) and wait for 100 ms before resetting the system.

Voluntary Entry to IMODE

This denotes when users have connected a debugger to the device and the device is ready to accept one of the supported Interactive mode commands. Voluntary entry into IMODE is typically done during product development or when debugging issues with a released product. Often, an IDE such as MPLAB® X serves as the user interface for issuing commands.

Debugger Interaction

Both entry methods require a cold-plugging sequence. See the *DSU - Device Service Unit* chapter for details. The debugger or programmer used must support IMODE and be compatible with this device. Compatible debuggers can be Microchip-designed, such as PICkit™ 5, ICE 4, ICD 5, or from a third party.

All compatible debuggers support commands to enter and exit IMODE, to manipulate the Device Access Level (DAL), and perform a full device erase.

The IMODE commands to read out configuration fuses and to run CRC on specific memory regions are considered advanced debugging features. As such, support for these commands is optional in debugger implementations, since they are not not intended for post-production use.

Refer to the *Programming Specifications* section for further details on the commands.

⚠ CAUTION It is possible to lock commands, i.e., prohibit their execution on the device. The device may become **permanently unrecoverable** if the user does not retain access to recover paths. Refer to the *Programming Specification* section for more information.

6.4.7. Program and Debug interface Disable (PDID)

The Program and Debug Interface Disable (PDID) comprises of a number of measures and dependencies that prevent access to the device's reprogrammable Flash memory via programmer or debugger. After activating the PDID measures, a programmer or debugger is prevented from making any changes to the device through the DSU, but can still read out a restricted set of device information.

Follow these steps to inhibit Flash manipulation via a debugger or programmer:

1. Enter the Interactive Mode (IMODE) of the Boot ROM.
2. Disable the IMODE commands by writing their respective key value in the ROM Configuration (CFM::ROMCFG) to all-zeroes. This change will take effect only after the next device reset.
3. Issue the command `CMD_SDAL0` to set the Debugger Access Level to DAL0. This change will take effect only after the next device reset.
4. Reset the device so the previous measures take effect.

Now the following rules apply:

- A debugger can only access DSU registers mapped in the DSU external space (at offsets between 0x0100–0x01FF) and the DSU CoreSight™ ROM table. This allows device identification but inhibits read/write access to the PFM sections.
- The Chip Erase command (`CMD_CE_ALL`) is disabled, so it cannot force the factory default for neither Debugger Access Level (factory default: DAL2) nor command key values (factory default: all valid).
- Read or write operations on the application code can only be performed by code located in the Boot Code section (bootloader).

Note:
The bootloader software must be able to receive new data and to program the Application Code section. The bootloader cannot alter code stored in the Boot Code section, and the access to the Boot Code section by the DSU is restricted by the DAL0 setting.

The application authors must ensure that the bootloader implementation fulfills the security requirements.

⚠ CAUTION The device may become **permanently unrecoverable** if the user does not retain access to recover paths. Devices with PDID invoked will have extremely limited failure analysis capabilities.

6.4.8. Status and Error Codes

During the boot process or while interacting with a debugger, the device may emit a 32-bit code indicating either a status code (starting with `0x0000nnnn`) or an error code (`0xEEEEnnnn`). These codes are primarily intended to assist the host debugger in interpreting the device's state, especially during Interactive mode, and to confirm successful boot operation to the application.

6.4.8.1. Status Codes

Status codes report the current state or progress of the boot process. Most status codes are internal and/or used by the debugger to support IMODE commands.

Under normal operation, the code STATUS_BOOTOK (`0x00000004`) indicates that the entire boot sequence has completed successfully. If applicable, the Boot ROM writes the STATUS_BOOTOK code to the Boot ROM Channel (BCC) register in the DSU. The DSU.BCC register can be read by the first mutable application executable code to confirm that the device has booted correctly.

Note: If the boot operation fails, the device will not transfer control to the first mutable executable application code, and the STATUS_BOOTOK code will not be written. The device may remain in a recovery or Interactive mode.

6.4.8.2. Error Codes

During the boot process, the device may encounter error conditions that prevent it from completing initialization. In such cases, the Boot ROM issues a 32-bit error code, which is written to the DSU.BCC register. The DSU.BCC register retains the *last value written* by the device, allowing the user or debugger to inspect the cause of the boot failure.

Note: Error code values start with `0xEEEEnnnnn`. The presence of an error code indicates that the boot operation was interrupted and the first mutable application executable code was not executed.

6.5. Programming Specifications

The following sections provide the necessary information for programming and debugging the device without using the software and hardware tools provided by Microchip.

6.5.1. Reference Documents

Aside from the data sheet, refer also to the following documents:

- ARM®v6-M Architecture Reference Manual - ARM DDI 0419E
- AMBA 3 AHB-Lite Protocol Specification v1.0 - ARM IHI 0033A
- Debug Interface v5.2 Architecture Specification - IHI0031C
- ARM® CoreSight™ Architecture Specification v3.0 - ARM IHI 0029E
- CoreSight™ MTB-M0+ Technical Reference Manual - ARM DDI 0486B

6.5.2. Memory Map

Refer to the *Memory Map* chapter or use the `CMSIS *.SVD` files provided for each part when implementing your tools. In addition, these registers from the ROM Configuration (ROMCFG) fuse are relevant for programming and debugging the device:

Table 6-1. ROMCFG Key Values

Address	Name	Disables/Allows	Bit Field Values	Value after Chip Erase
`0x0D00_0018`	KEYVAL_CRC	`CMD_CRC` (check memory integrity)	`KEYVAL[31:0]`	`0xFFFF_FFFF`
`0x0D00_0020`	KEYVAL_CE_ALL	`CMD_CE_ALL` (Chip Erase)	`KEYVAL[31:0]`	`0xFFFF_FFFF`
`0x0D00_0028`	KEYVAL_SDAL0	`CMD_SDAL0` (Set Debugger Access Level to 0)	`KEYVAL[31:0]`	`0xFFFF_FFFF`

Notes: Decoding of `KEYVAL` bit field values:

- All bytes are 0xFF: Command is allowed
- All bytes are 0x00: Command is diabled

6.5.3. Device ID

The device is identified by reading the Device Identification (DID) register in the Device Service Unit (DSU):

Table 6-2. Device ID

Part Number	DSU.DID[31:0]		
	DSU.DID[31:28] (Silicon Revision ID)	DSU.DID[27:12] (Part Number)	DSU.DID[11:0] (Fixed)
CM6408PL10028	0x0	0xBA00	0x053
CM6408PL10032	0x0	0xBA01	0x053
CM6408PL10048	0x0	0xBA02	0x053
CM6408PL10064	0x0	0xBA03	0x053

Note: Refer to the register description in the *DSU - Device Service Unit* chapter for more details.

6.5.4. Device Reset

The device only supports Reset sources defined by the Arm® implementation. No additional vendor-specific Reset controllers are implemented.

6.5.4.1. System Reset

Writing `0x05FA_0004` to the AIRCR core register triggers SYSRESETREQ, which resets the entire chip, including all peripherals.

6.5.4.2. Vector Catch

The vector catch feature allows the CPU to halt after a Reset or other exceptions. Note that, since the PIC32CM PL10 family has a Boot ROM, the Reset vector catch cannot be used to prevent the CPU from executing the first instruction in the First Mutable Executable (FME), as the vector catch would halt the CPU on the first instruction in the Boot ROM while the device is locked (DAL = `0`). Instead, Park mode is used to halt the CPU and prevent it from executing the FME.

Refer to the *ARM®v6-M Architecture Reference Manual* for more details.

6.5.4.3. External Reset

A debugger can pull the external $\overline{\text{RESET}}$ pin low to reset the device. The application software can reconfigure the $\overline{\text{RESET}}$ pin to the General Purpose Input/Output (GPIO) function, in which case the Reset function is lost. This reconfiguration is performed by the application code, i.e., after the Boot ROM has completed, so Interactive or Park modes can be entered to prevent such $\overline{\text{RESET}}$ pin reconfiguration.

6.5.4.4. CPU Reset Extension

The DSU peripheral has CPU Reset and Boot ROM extension mechanisms that allow a debugger to be connected while preventing the CPU from starting execution of the customer application. This is also known as debugger cold-plugging.

This feature requires the debugger to control the SWCLK and $\overline{\text{RESET}}$ pins. The debugger forces a CPU Reset extension by driving three pulses on SWCLK when the $\overline{\text{RESET}}$ pin is low.

The main purpose of the CPU Reset extension is to avoid executing potentially corrupted code at the boot and to provide a safe way to connect to a device without making any assumptions about its current state. Combined with the Boot ROM Reset extension, the debugger controls the behavior of the Boot ROM Interactive and Park modes. This allows reading and writing of the memory and peripherals without CPU interference.

After the initial connection sequence, the debugger must control the CPU Reset and Boot ROM extension to maneuver the Boot ROM into Interactive or Park modes. Typically, Park mode is used to program the device using the ARM DAP AHB-AP port.

The CPU Reset Extension (bit 8) and Boot ROM Extension (bit 16) are located in the DSU Status A (DSU.STATUSA) register @0x4100_2104. Each of these bits can be cleared by writing a '1' to them. To release the CPU, write a '1' to the CRSTEXT0 bit in the STAUSA (STATUSA.CRSTEXT0) register. The BREXT0 bit in the STATUSA (STATUSA.BREXT0) register is typically written or left untouched within the same write operation, depending on whether entry into Interactive mode or Park mode is required. Writing a '0' to these bits has no effect.

6.5.4.5. CPU Boot

After leaving the Boot ROM, the Cortex-M0+ processor boots from a vector table at address offset '0' in the Program Flash Memory (PFM). The processor reads the Main Stack Pointer value from the address offset 0x0 and the code entry point (Reset vector) from address offset 0x4.

6.5.5. SW-DP – Serial Wire Debug Port

This section describes how to access the Serial Wire Debug Port (SW-DP).

6.5.5.1. Overview

The Serial Wire Debug (SWD) protocol is described in Chapter 5 of *Arm® Debug Interface v5 Architecture Specification (ARM IHI 0031 Revision x*).

The exact revision of the document is not specified by Arm.

6.5.5.2. Operation Sequences

To initialize or uninitialize the SWD, follow the Debug Interface specification (ARM® IHI 0031).

Refer also to the *Programming* and *In-Circuit Debugging* sections for more details.

6.5.5.3. Debugger Features

The Device Service Unit (DSU) supports debugger features required by the Arm CoreSight specification, namely a CoreSight ROM table, as previously described. It also extends the debugger functionality by supporting:

- The debugging and testing of a protected or protectable device
- Debugger interface pin allocation
- Other debug and test features that add value to the device

The debugger interfaces with the device through the Debug Access Port (DAP). In addition to its use in Arm's CoreSight debug architecture, the Serial Wire Interface can also be used for production testing, validation and internal debugging.

6.5.5.3.1. Debugger Routing

The debugger explicitly selects the target AHB-AP in its request. AHB-AP0 is connected to the CPU's (the Cortex M0+) AHB bus.

All debugger transactions addressed to a Private Peripheral Bus (PPB) peripheral remain within the CPU. The PPB is where the ARM embedded debug components reside. All debugger transactions are routed to the CPU's main bus, where they can be forwarded to other peripherals of the system via the AHB bus matrix.

6.5.5.4. Debugger Security

The DSU implements security features to prevent regular access to a protected device through the debug interface, thereby protecting sensitive data from being read and user code from reverse engineering.

6.5.5.4.1. DAL – Debug Access Level

The current Debug Access Level (DAL) is indicated in the CPU0 bit field of the DSU.DAL register.

The DAL setting controls debugger access to everything in the system. The lower the DAL value, the more restricted the debugger access is.

Table 6-3. DAL Settings

DAL[1:0]	Access
0x3	Reserved, do not use
0x2	The debugger can access all addresses accessible from CPU0
0x1	Reserved, do not use
0x0	The debugger can only access the DSU addresses in the external address range, i.e., the DSU External Space Address Registers (offset 0x0100–0x01FF) and the DSU CoreSight™ ROM table (offset 0x1000–0x1FFF)

It is possible to set DAL < 0x2 and also prevent this from being changed by disabling the interactive mode command CMD_CE_ALL. See also the *Program and Debug interface Disable (PDID)* section of the Boot ROM chapter.

6.5.5.4.2. DSU Register Access

The DSU registers can be accessed through its APB bus interface.

When DAL < 0x2, only DSU registers mapped in the DSU external space (at offsets between 0x100 and 0xFFF, and the DSU ROM table) are accessible.

6.5.5.4.3. Debugger Detection

The DSU detects the presence of the debugger and supports both hot plugging and cold plugging:

- Cold plugging detection occurs if at least three SWCLK pulses are detected before the $\overline{\text{RESET}}$ pin is de-asserted. The device must be out of Power-on Reset (POR) before these pulses can be detected.
- Hot plugging detection occurs on a falling edge of SWCLK when the part is not in Reset. An internal pull-up on SWCLK prevents false detection if SWCLK is unconnected. Hot plugging is not available when DAL = 0x0 or if the pin that SWCLK is on has been switched to a different function.

Refer to the *CPU Reset Extension* section for more information on hot- and cold-plugging. Once the debugger is detected, the SWDIO function takes control of the pin to which it is assigned and can only be released back for other usage by a Power-on Reset (POR) in the case of cold plugging, or by either a POR or external reset in the case of hot plugging.

6.5.5.5. DSU Register Write Protection

Several DSU registers with write access, except for DCC0/1 and BCC0/1, are write-protected by the WPCTRL mechanism.

6.5.5.6. Serial Wire Interface

Table 6-4. Serial Wire Signals

DIR	Signal	Description
I	SWCLK	Clock input
I/O	SWDIO	Bidirectional serial data

The devices in the PIC32CM PL10 family support Arm's CoreSight Serial Wire Interface. This interface consists of two pins: SWCLK, a clock signal that is always an input to the device, and SWDIO, a bidirectional data pin. This interface provides access to all of Arm's customer-visible debug features.

This section summarizes the critical details of this interface for production test and validation.

Refer to the *Arm® Debug Interface Architecture Specification ADIv5.0 to ADIv5.2 (IHI 0031)* for more details.

6.5.5.6.1. Architecture

As defined by Arm, the Serial Wire Interface connects from external pins to a Debug Port (DP). The Debug Port handles communication and synchronization between the external host, debugger, or tester and the target/device. The Debug Port then connects to one or more Access Ports (AP) to communicate with other resources within the device.

6.5.5.6.2. Debug Port

The SW-DP handles communication and synchronization between the external host, debugger, or tester and the target/device using the Serial Wire Interface. Each transaction can access either a DP or AP register. The devices in the PIC32CM PL10 family implement a DPv2-compliant DP.

6.5.5.6.3. Memory Access Port

Essentially, a MEM-AP is a bus host on a bus interface. It is accessed and controlled by reading and writing to its register interface using the DP. Any memory-mapped resource that is accessible from the bus driven by this MEM-AP can be written or read from, following the normal access permissions of that resource.

The devices in the PIC32CM PL10 family contain one MEM-AP, which is an AHB-AP.

The AHB-AP connects through the CPU's AP and to the AHB bus matrix. The MEM-AP can access all subsystem memory-mapped resources in the design. The only exception to this is when the device is protected (DAL < 0x2); in this case, the DSU performs security filtering as described in the *Debugger Security* section.

6.5.6. Boot ROM for Programming and Debugging

The Boot ROM plays an essential part during device programming and debugging, as illustrated below.

Figure 6-3. Boot Flow Overview

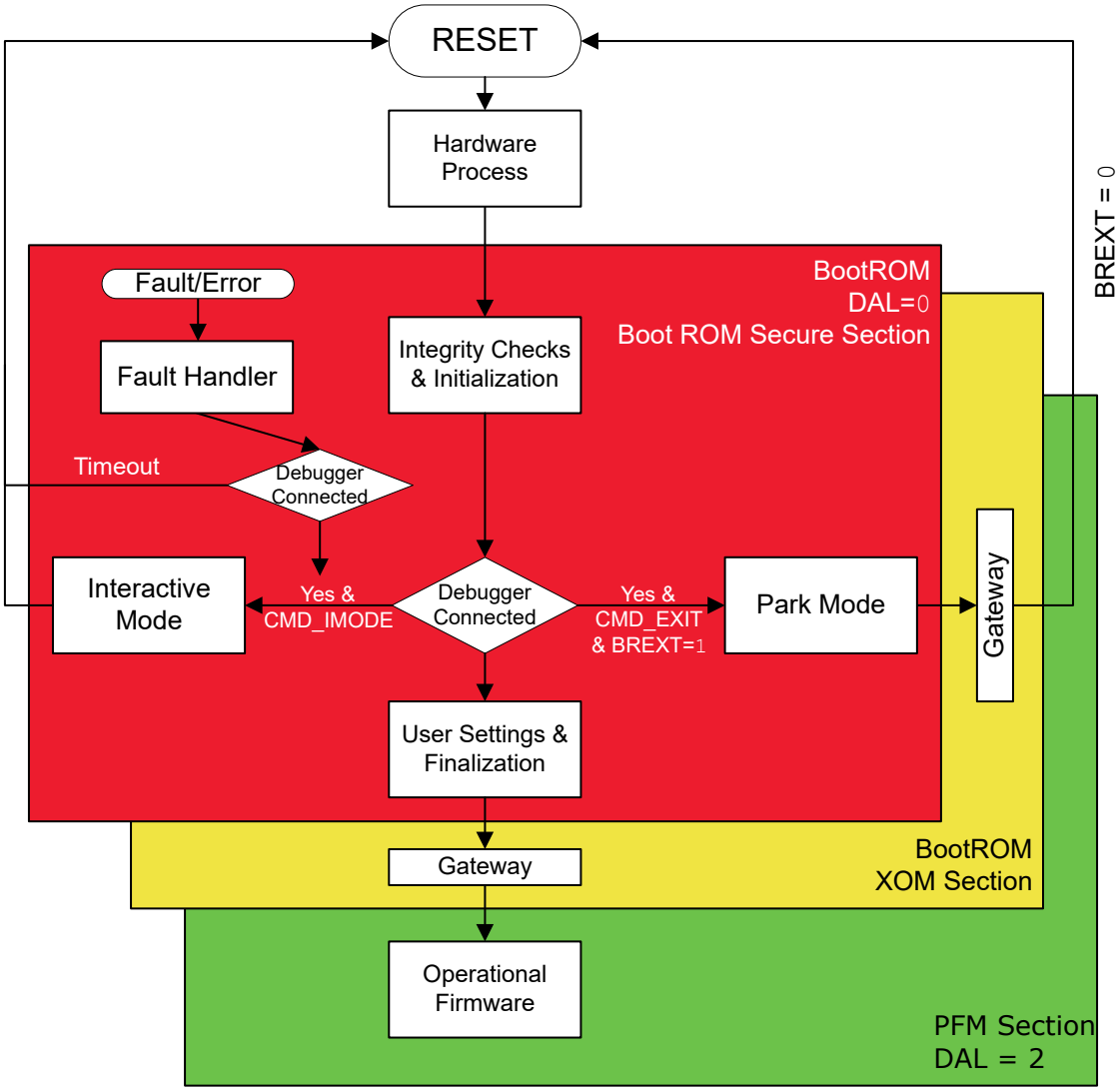
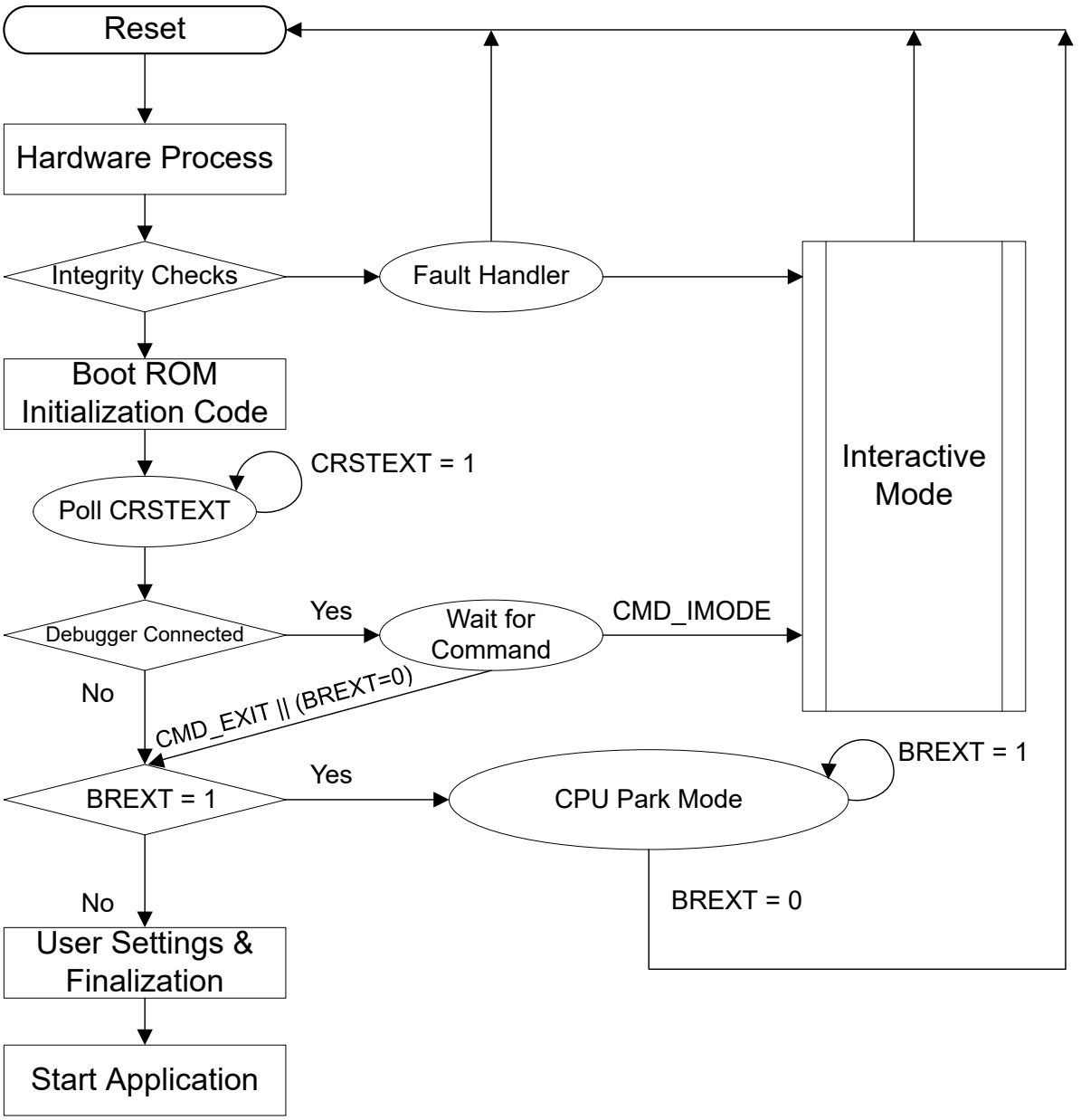


Figure 6-4. Boot ROM Detail



6.5.6.1. Integrity Checks and Initialization

The Boot ROM initializes the device, applies calibration data, and starts clocks.

Detected errors are handled as noted in the *Error Handling* section.

6.5.6.1.1. Hardware CRC of PFM on Boot

The CRC table is listed in Table 6-13. This table is located at `ENDADDR(CFM::BOOTCFG.BOOTPROT.BOOTPROT)-16`. If `BOOTPROT==7` (0 bytes), then the CRC table is located at `ENDADDR(PFM)-16`.

6.5.6.1.2. Error Handling

When the Boot ROM detects an error in the boot flow or a hardware fault, it writes a code as indicated in Table 6-10 to BCC1 and jumps to a fault handler.

As part of Power-On Self Test (POST), the Boot ROM checks for faults in the hardware and memories it uses. If a fault is detected, the Boot ROM enters the fault handler.

- If a debugger is not connected and the user has selected a GPIO configuration in CFM::BOOTCFG.BOOT_GPIOSEL, the Boot ROM attempts to apply those settings. It then waits 100 ms (so the error status can be observed) before resetting the device to check if the error is transient.
- If a debugger is connected, the Boot ROM does not apply GPIO settings. It waits indefinitely for the debugger to read status and send commands:
  - If the command is CMD_IMODE, the Boot ROM enters Interactive mode
  - If the command is CMD_EXIT, the Boot ROM resets the device
- If the Boot ROM is already in Interactive mode when it detects an error, it puts the status in BCC1 and returns to the Wait-for-Command state. The GPIO selections are not applied in this case.
- If POST finds no faults, the Boot ROM continues normal operation (Boot, Interactive, and Park modes). Therefore, a hardware or memory error occurring during normal operation is a failure condition and causes a hard fault. A hard fault is an unrecoverable error that triggers a device Reset, regardless of whether a debugger is connected. Interactive mode is not available in this situation.

6.5.6.2. Debugger Communication Channels

The DSU provides two pairs of registers for debugger and device communication as detailed in the following subsections.

6.5.6.2.1. Debug Communication Channels

The DSU provides two registers: Debug Communication Channel x (DCC0 and DCC1) for communication and synchronization between the debugger and the CPU. Each register has a corresponding dirty bit, DCCD0 and DCCD1, respectively. These two dirty bits are set on a write and cleared on a read by hardware, and they reside in DSU.STATUSB register.

Customers can use these registers to build their own communication protocol.

6.5.6.2.2. Boot Communication Channels

The DSU also provides two additional registers, BCC0 and BCC1, for communication between the debugger and CPU while the Boot ROM is executing. These registers map to the DCC0/1 registers but have their own dirty bits. These dirty bits also reside in the DSU.STATUSB register and are called BCCD0 and BCCD1. These bits were added to ensure that the usage of the Boot ROM does not interfere with DCCx communication.

6.5.6.3. Boot ROM Operation During Programming and Debugging

The Boot ROM supports two modes. Refer to Figure 6-4:

1. **CPU Park mode:** In this mode, the core is halted in sleep mode, so it does not access any peripheral registers or memories. Therefore, the debugger can safely program the device.
2. **Interactive mode:** In this mode, the Boot ROM waits for commands and executes them.

The core never needs to be halted through the standard Cortex-Mx (DHCSR, DEMCR, AIRCR) registers. Park mode is used instead.

To run the Boot ROM in Interactive mode, a special cold-plug Reset sequence is used (refer to the *CPU Reset Extension* section). This process pauses the Boot ROM execution after the Boot ROM has initialized the device. To end the Reset extension, the debugger must clear the CRSTEXT bit in the DSU.STATUS register. The bit is cleared by writing a '1' to it.

After some time (5 ms is recommended), the debugger must check if the Boot ROM has flagged any errors by reading BCC1. At this point, the debugger may exit the application by clearing the BREXT bit and sending the CMD_EXIT command. If CMD_EXIT is issued while the BREXT bit is set, the Boot ROM will enter Park mode and the response code will be set to STATUS_BOOTOK.

Alternatively, by leaving the BREXT bit set and waiting for the Boot ROM to set STATUS_INITCHECK_OK, the debugger may issue `CMD_IMODE` to enter the Interactive mode. Once the Boot ROM reports `STATUS_OK`, the device is in Interactive mode. In this mode, the Boot ROM is ready to accept other commands.

6.5.6.4. Interactive Mode

The Boot ROM provides interactive debugging capabilities to facilitate chip erase operations on protected devices (DAL < `0x2`). This functionality is also available on unprotected devices.

It also allows the debugger to request a Reset. During interactive debugging, the host, debugger, and the target/device utilize the Boot Communication Channels to send a limited and predetermined set of commands and responses between each other. Refer to Boot Communication Channels for more information.

By convention, BCC0 is always used to send commands from the host to the target while BCC1 is always used to send responses from the target to the host.

6.5.6.4.1. Entering Interactive Mode

The procedure in Table 6-5 must be performed to enter Interactive debug mode unless an integrity check fails. In that case, the device will automatically enter Interactive debug mode. If there is a cold-plug detection of the debugger, the procedure in Table 6-5 must be followed, regardless of whether the user wants to enter Interactive debug mode.

Table 6-5. Debugger Cold Plug Start-Up Procedure

Step	Host/Debugger	Target/Device
1	Assert $\overline{\text{RESET}}$ pin	Device is held in Reset
2	Drive at least three pulses on the SWCLK pin; de-assert $\overline{\text{RESET}}$ pin	Cold-plugging: The debugger is detected, and the Boot ROM starts the system start-up sequence
3	Wait enough time to guarantee that system initialization has completed	The CPU performs mandatory integrity checks and, if any fail, carries out error handling as described in the *Error Handling* section
4	SW: Set DSU.STATUSA.CRSTEXT to `1'b1` to clear the CPU Reset Extension	The CPU continues executing from the Boot ROM
5	Wait for DSU.STATUSB.BCCD1 = `1'b1`	
6	SW: Read DSU.BCC1 for the response code to check if any errors were logged	CPU polls DSU.STATUSB.BCCD0 = `1'b1`

After this start-up procedure, the device waits for a command from the host. If no errors are reported, there are only two valid procedures to continue the process.. The host can follow the procedure indicated in Table 6-6 to enter Interactive debug mode or follow the procedure in Table 6-7 to skip it. If any other command is written to DSU.BCC0, the Boot ROM will simply ignore it and continue waiting for one of these commands.

Table 6-6. Interactive Debug Mode Entry Procedure

Step	Host/Debugger/Tester	Target/Device
7	SW: Write DSU.BCC0 with `CMD_IMODE`	The CPU polls DSU.STATUSB.BCCD0 = `1'b1` (continued from Step 6 in Table 6-5)
8	Wait for DSU.STATUSB.BCCD1 = `1'b1`	The CPU reads DSU.BCC0 for the command code and acknowledges `CMD_IMODE` by writing STATUS_CMD_VALID to DSU.BCC1
9	SW: Read DSU.BCC1 for the response code to confirm entry into Interactive debug mode	Device enters Interactive debug mode

Table 6-7. Interactive Debug Mode Skip Procedure

Step	Host/Debugger/Tester	Target/Device
7	SW: Write DSU.BCC0 with `CMD_EXIT`	The CPU polls DSU.STATUSB.BCCD0 for the command code (continued from Step 6 in Table 6-5)
8	Wait for DSU.STATUSB.BCCD1 = `1'b1`	The CPU reads DSU.BCC0 for the command code and acknowledges `CMD_EXIT` by writing STATUS_BOOTOK to DSU.BCC1
9	SW: Read DSU.BCC1 for the response code to confirm Boot ROM exit	The CPU completes Boot ROM execution and enters Park mode

If Interactive debug mode is skipped, the normal operation of the Boot ROM continues until it enters Park mode. Refer to the *CPU Reset Extension* section for details on Boot Entry.

Once in Interactive debug mode, an external Reset or the `CMD_EXIT` command can be used to exit this mode.

6.5.6.4.2. Interactive Mode Command Access Controls

The Command Access controls for non-secure devices do not provide secure, authenticated use of the command, even though they use the same protocol as secure devices. The protocol uses a sequence to support command lock selection based on a KeyVal field associated with each command. Refer to the *Memory Map* section in this chapter for available KeyVal registers.

6.5.6.4.3. Challenge-Response

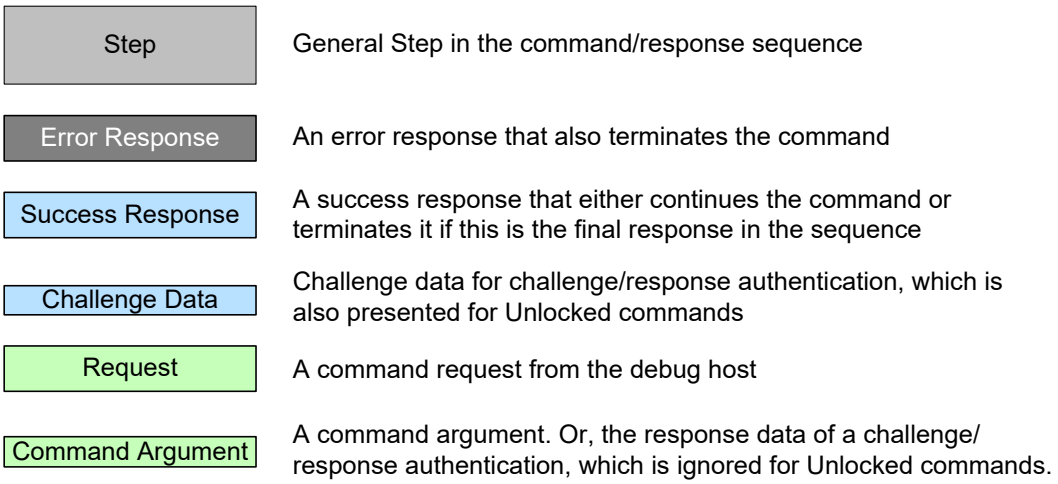
The standard protocol for the PIC32CM PL10 devices uses a challenge-and-response mechanism that supports both unlocked and authenticated commands. In this case, the host sends a command, and the device responds with STATUS_CHALLENGE. The device then sends 128 bits of challenge data and waits for the 256 bits of response data. The host then performs an HMAC to generate the response.

⚠ CAUTION This challenge-response protocol (i.e., KeyVal = Authenticated) is not supported by the PIC32CM PL10 family: These devices do not check the response and always return a status of STATUS_CMD_VALID. They will send any challenge, and the host can therefore send any response.

When the command is set to Locked, the device responds with STATUS_ARG_INVALID. The device does not process the command any further, and it waits for another command.

6.5.6.4.4. Command Diagram and Definition Legend

Figure 6-5. Command Message Sequence Legend



Command Definition tables use the following column headers:

- Type: The type of information that flows during a part of the command sequence
- Input: Information from the debugger. This precedes the output, except in the case of challenge and response.
- Size: The size of non-status information passed between the debugger and the device
- Range: The valid range of the input information
- Output: Either STATUS or the information requested by the COMMAND

6.5.6.4.5. Interactive Mode Commands

The PIC32CM PL10 devices support the debug commands listed in Table 6-8. All commands, except for CMD_IMODE and CMD_EXIT, can only be used once the device is in Interactive debug mode. The CMD_IMODE and CMD_EXIT commands are used to enter or exit the Interactive debug mode.

All the commands listed in Table 6-8, except for the CMD_IMODE and CMD_EXIT, initiate an Interactive debug operation. The descriptions and procedures for these commands are provided below. These procedures assume that the device has already entered Interactive debug mode. Upon completion of any of these commands, the Boot ROM will wait for the next command. If, for any reason, the Boot ROM does not properly recognize a command, it will return STATUS_CMD_INVALID instead of STATUS_CMD_VALID.

Table 6-8. Interactive Mode Commands

Name	Description	Value
CMD_IMODE	Enter Interactive mode	0x444247 55
CMD_EXIT	Jump to exit section	0x444247 AA
CMD_READCFM	Read a configuration memory word	0x444247 4C
CMD_CRC	Check the integrity of memory using the Cyclic Redundancy Check (CRC) table	0x444247 C0
CMD_CE_ALL	Erase Flash memory, wipe volatile memories, and set the device to DAL2	0x444247 E3
CMD_SDAL0	Set the device to DAL0	0x444247 10

Table 6-9. Interactive Mode Completion Status

Name	Definition	Code
STATUS_CRC_FAIL	The CRC of memory area doesn't match the expected value in the table	0x00000001
STATUS_CRC_OK	The CRC of memory area matches the expected value in the table	0x00000002
STATUS_INITCHECK_OK	Boot initialization complete, Interactive mode may be called	0x00000003
STATUS_BOOTOK	Entire boot sequence complete, DAL is set to the final value	0x00000004
STATUS_CMD_VALID	The previously transmitted command is valid and command flow continues	0x00000005
STATUS_ARG_INVALID	Unexpected or incoherent argument submitted to command or a locked command	0x00000006
STATUS_DATA_VALID	Boot ROM has valid data to output to debugger	0x00000008
STATUS_OK	Operation or command sub-sequence succeeded	0x00000009
STATUS_CHALLENGE	Prepare to accept an authentication challenge	0x0000000B

Table 6-10. Interactive Mode Error Status

Name	Definition	Interactive Mode Available	Code
STATUS_ERR_HARDFAULT	Hard fault occurred	No	0xEEEE0001
STATUS_ERR_CALOTP	Cal-OTP page integrity check failed	Yes	0xEEEE0002
STATUS_ERR_BOOTCFG	BOOTCFG page integrity check failed	Yes	0xEEEE0004
STATUS_ERR_EXEC	Boot ROM execution error, program flow diverted	No	0xEEEE0006
STATUS_ERR_PROG	Flash programming error	Yes	0xEEEE0007
STATUS_ERR_ECC	Double error detected on Flash while running the command	Yes	0xEEEE0008
STATUS_ERR_BUS	Bus error occurred on access to a system address	Yes	0xEEEE000C
STATUS_ERR_CRCTABLE	Incorrect CRC table format/values	Yes	0xEEEE000D
STATUS_ERR_ERASE	Erasing of Flash memory failed	Yes	0xEEEE000F
STATUS_ERR_FUSEUP	Hardware fuse update compare check failed	Yes	0xEEEE0011
STATUS_ERR_ROMCFG	ROMCFG page integrity check failed (Only `CMD_READCFM` is guaranteed to be available. Other commands may be disabled or their KEYVAL is in error.)	Yes	0xEEEE0019
STATUS_ERR_PFMCRC	Program Flash Memory integrity check failed	Yes	0xEEEE001A

Command IMODE

This command is used to enter Interactive mode, as noted above.

Type	Input	Size	Range	Output	Comment
Command	`CMD_IMODE`	32-bits		STATUS_CMD_VALID	Refer to Table 6-9

Command EXIT

This command is used to proceed to the Exit section, where the selection of BREXT determines entry into either Mission Debug mode or Park mode.

Type	Input	Size	Range	Output	Comment
Command	`CMD_EXIT`	32-bits		STATUS_CMD_VALID	Refer to Table 6-9

Command READCFM

This command reads an address in the `NVMCTRL::CFM` space and returns the data. If the address is read-protected or out of range, the device returns an invalid argument status and waits for another command. If the address is within range, the command succeeds with data valid status, followed by the data.

This procedure allows one 32-bit value to be read at a time, but continues accepting target addresses until an address outside the allowable range is requested. By convention, write a target address of `32'h0000_0000` when you intentionally want to exit this procedure.

The Table 6-11 defines the command structure, and Figure 6-6 shows the communication sequence.

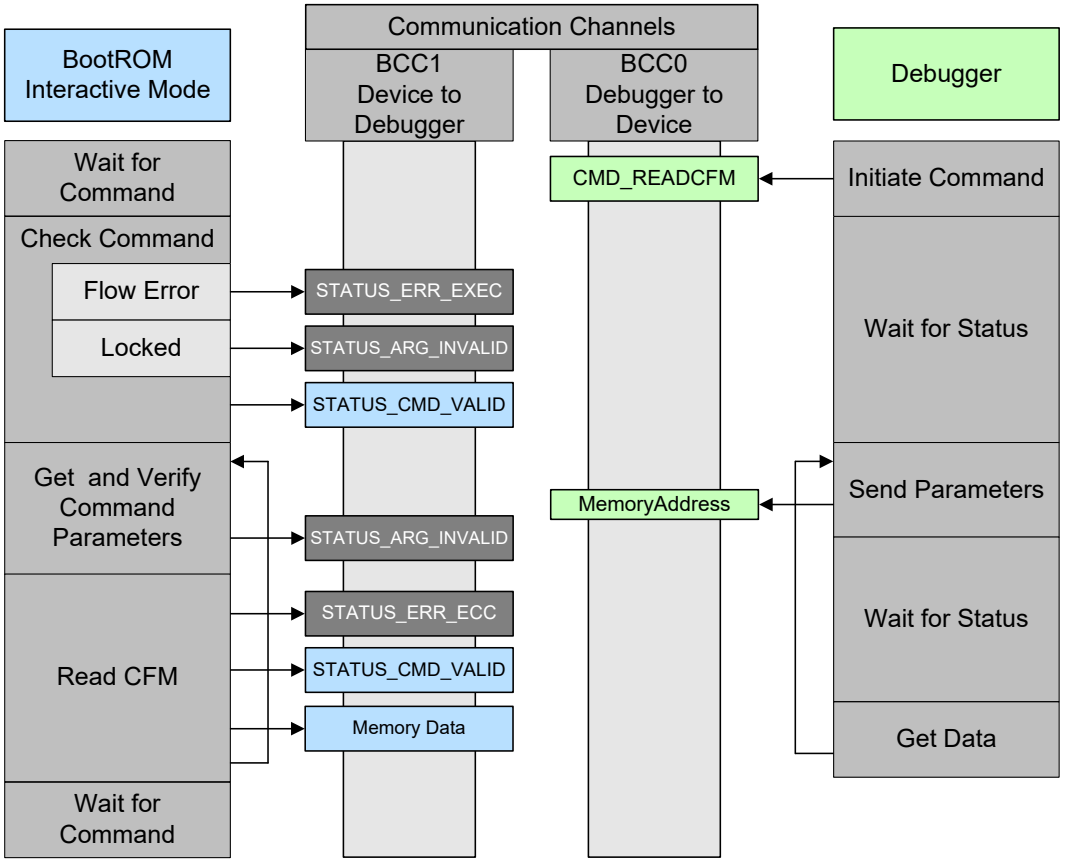
Table 6-11. CMD_READCFM Definition

Type	Input	Size	Range	Output	Comment
Command	`CMD_READCFM`			STATUS_ERR_EXEC STATUS_ARG_INVALID STATUS_CMD_VALID	Refer to Table 6-9 and Table 6-10
Argument1	Address	32 bits	Refer to Table 6-12	STATUS_ARG_INVALID	The command is aborted on an address error. This is how the command is intentionally ended. Refer to Table 6-9
Result				STATUS_ERR_ECC STATUS_DATA_VALID	Refer to Table 6-9 and Table 6-10
Returns		32 bits		Data	The command awaits a new Address after a valid read

Table 6-12. READ_CFM Valid Addresses

Region	Description
`CFM::SIGNATURE`	Firmware metadata
`CFM::BOOTCFG`	User configuration

Figure 6-6. CMD_READCFM Message Sequence



Command CRC

This command performs a CRC32 on a range of Flash memory as specified by a CRC table that exists in that region. Applicable Flash regions are those listed in the device's "System Address Map" that start with `NVMC::`. The CRC tables are user-programmable, and multiple tables can exist in a region. Tables must be aligned to a 16-byte address boundary.

These restrictions apply as security measures:

- The CRC command will not return the result of the CRC; it will only indicate whether the value matched or mismatched the expected value
- The region of memory to check and its expected CRC must be stored internally in a table. Therefore, the regions that may be tested must be planned in advance, and their expected CRCs must be calculated beforehand The debugger does not provide the expected CRC value; instead, it provides the starting location of one of these tables. As a result, the user must know where these tables are stored. Refer to Table 6-13 for more information on the CRC tables.
- There is a 128-bit fuse value (KeyVal*) used as a key to unlock the CRC command. If KeyVal is set to all '1's, no key is required; if it is set to all '0's, this functionality is disabled.

Table 6-13. General CRC Table Format

Word	Offset	Name	Value/Format	Description
0	0x0	HDR	0x43524349	Constant, denotes the CRC header
1	0x4	ADDR	0x0C000000	Start Address, must be word-aligned (four bytes)
2	0x8	SIZE	0x100	Size in bytes, but must be a multiple of four. The maximum size is the largest Flash region.

Table 6-13. General CRC Table Format (continued)

Word	Offset	Name	Value/Format	Description
3	0xC	REFVAL	0xAABBCCDD	Expected CRC32 result

CRC Table Structure

```
typedef struct {
 unsigned int hdr;
 unsigned int addr;
 unsigned int size;
 unsigned int refval;
} crcTable;
```

The calculation used is CRC-32-IEEE 802.3 with the following settings:

- Polynomial Length (PLEN) = 32 bits
- Polynomial (POLY) = `0x04C11DB7` (Poly)
- Initial Value = `0xFFFFFFFF` (Init)
- Reflected Input (RIN) = Input data is reflected—Least Significant bit (LSb) first
- Reflected Output (ROUT) = Output data is reflected
- No XOR (FXOR) is applied to the output CRC Accumulator (ACC)
  - This is equivalent to an FXOR of `0x00000000`
- The accumulator result is compared to REFVAL to determine pass or fail

The CRC command returns its availability before accepting two arguments: MEMMAP and TableAddress. After processing, the command returns the status of the operation, which can indicate a CRC table error, pass or fail. The Accumulator and Checksum are never returned.

Table 6-15 defines the command structure, and Figure 6-7 shows the communication sequence.

CRC Tables

If Microchip is performing Quick Turn Programming (QTP) for a customer, the following CRC tables must also be programmed for these regions. Otherwise, if the customer is performing their own Flash programming, they are strongly encouraged to create and program these CRC tables so the contents can be verified even after the device is secured.

Table 6-14. Suggested Application Region CRC Tables

Region
BOOT
Main

All information in the SIGNATURE and BOOTCFG sectors can always be read directly using the READCFM capability of the Boot ROM, Therefore, CRC tables are not required for these sectors.

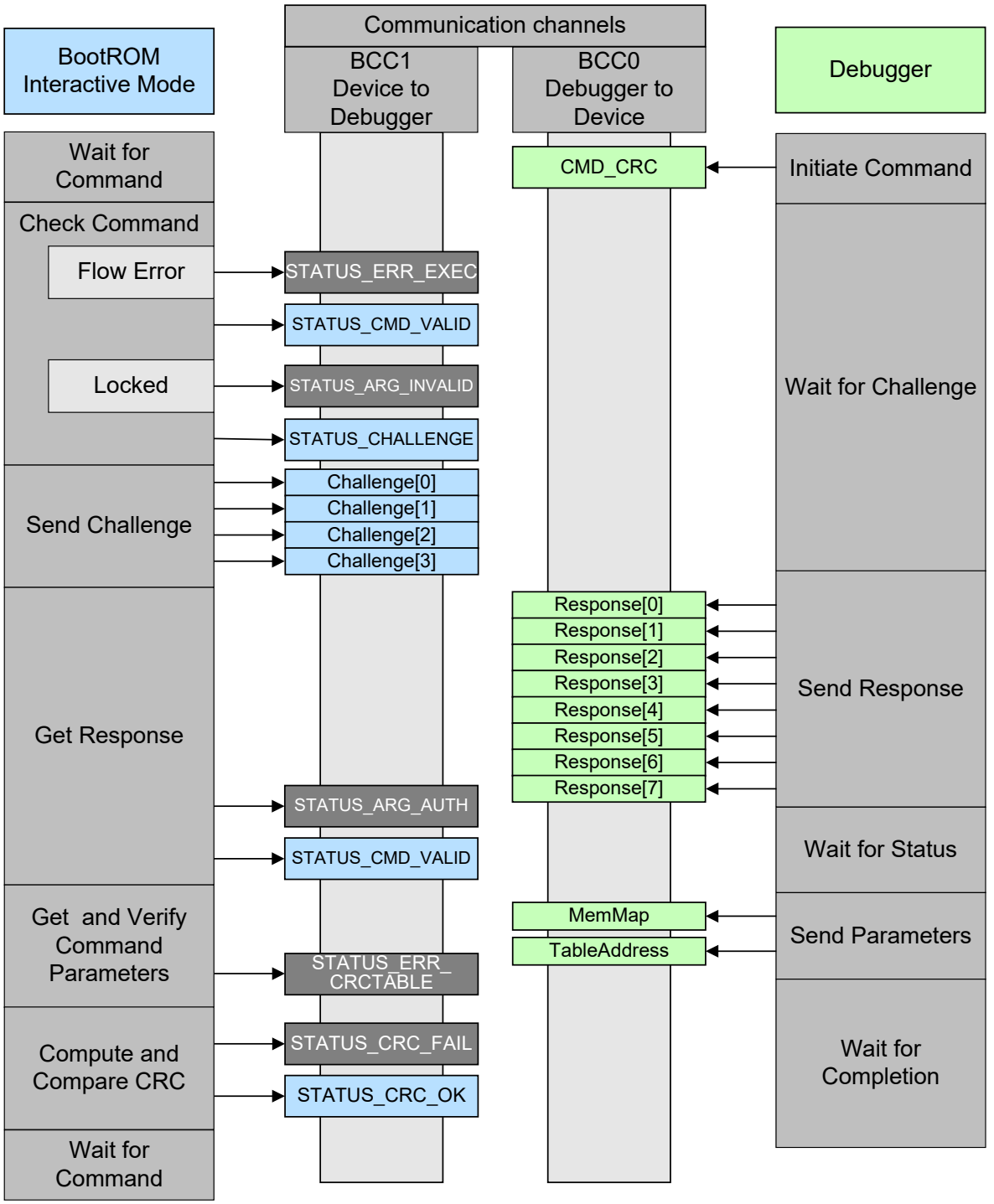
Table 6-15. `CMD_CRC` Definition

Type	Input	Size	Range	Output	Comment
Command	`CMD_CRC`	32-bits		STATUS_ERR_EXEC STATUS_ARG_INVALID STATUS_CMD_VALID STATUS_CHALLENGE	Refer to Table 6-9 and Table 6-10
Authenticate		128-bits		Challenge[]	LS Word first
Authenticate	Response[]	256-bits		STATUS_ARG_AUTH STATUS_CMD_VALID	LS Word first
Argument1	MEMMAP	32-bits	0x0		Must be 0x0

Table 6-15. CMD_CRC Definition (continued)

Type	Input	Size	Range	Output	Comment
Argument2	Table Address	32-bits	Flash Address	STATUS_ERR_CRCTABLE	Must be a 16-byte aligned address. Refer to Table 6-9
Result		32-bits		STATUS_CRC_FAIL STATUS_CRC_OK	Refer to Table 6-9 and Table 6-10

Figure 6-7. CMD_CRC Message Sequence



Command CE_ALL

This command initiates a chip erase operation, which removes all data from the PFM and CFM.ROMCFG memory regions and clears all volatile memory. CFM.SIGNATURE, CFM.BOOTCFG, pages are not erased. The Boot ROM performs `CMD_CE_ALL` in stages, erasing ROMCFG after all other memories have been wiped. The final step is to program the device to DAL2. A Reset is required after the command completes to access the device at DAL2.

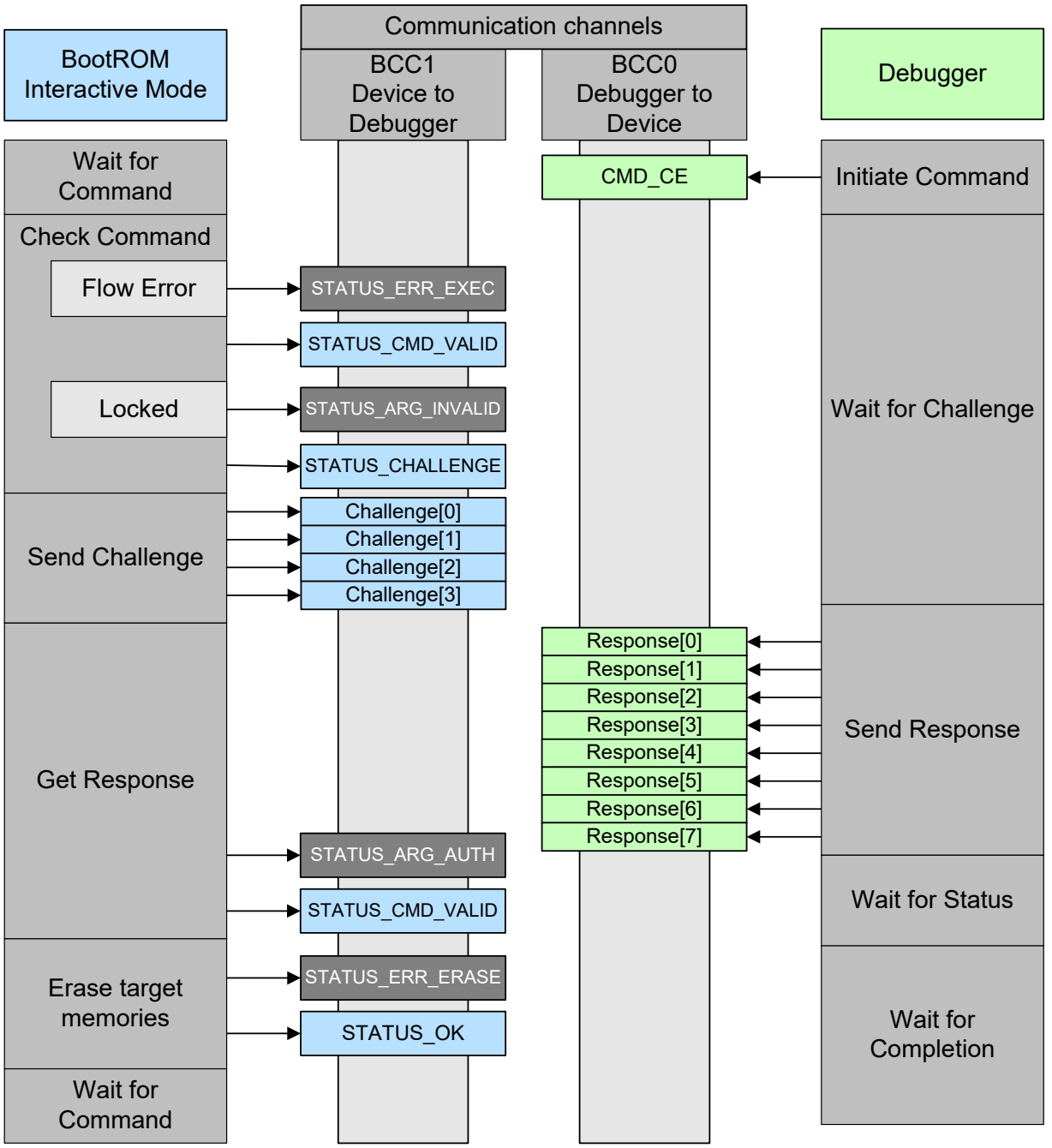
The Boot ROM accepts the command access controls selected in ROMCFG for all Debug Access Levels. Therefore, if the device is at DAL2 but Chip Erase is Locked (via the ROMCFG setting), the device cannot be erased via CMD_CE_ALL from Interactive mode. In this case, Park mode must be used to perform a chip erase by issuing a CHER command to the NVMCTRL.

The Table 6-16 defines the command structure, and Figure 6-8 shows the communication sequence.

Table 6-16. CMD_CE_ALL Definition

Type	Input	Size	Range	Output	Comment
Command	CMD_CE_ALL	32 bits		STATUS_ERR_EXEC STATUS_ARG_INVALID STATUS_CMD_VALID STATUS_CHALLENGE	Refer to Table 6-9 and Table 6-10
Authenticate		128 bits		Challenge[]	LS Word first
Authenticate	Response[]	256 bits		STATUS_ARG_AUTH STATUS_CMD_VALID	LS Word first
Result		32 bits		STATUS_ERR_ERASE STATUS_OK	Refer to Table 6-9 and Table 6-10

Figure 6-8. CMD_CE_All Message Sequence



Command SDAL0

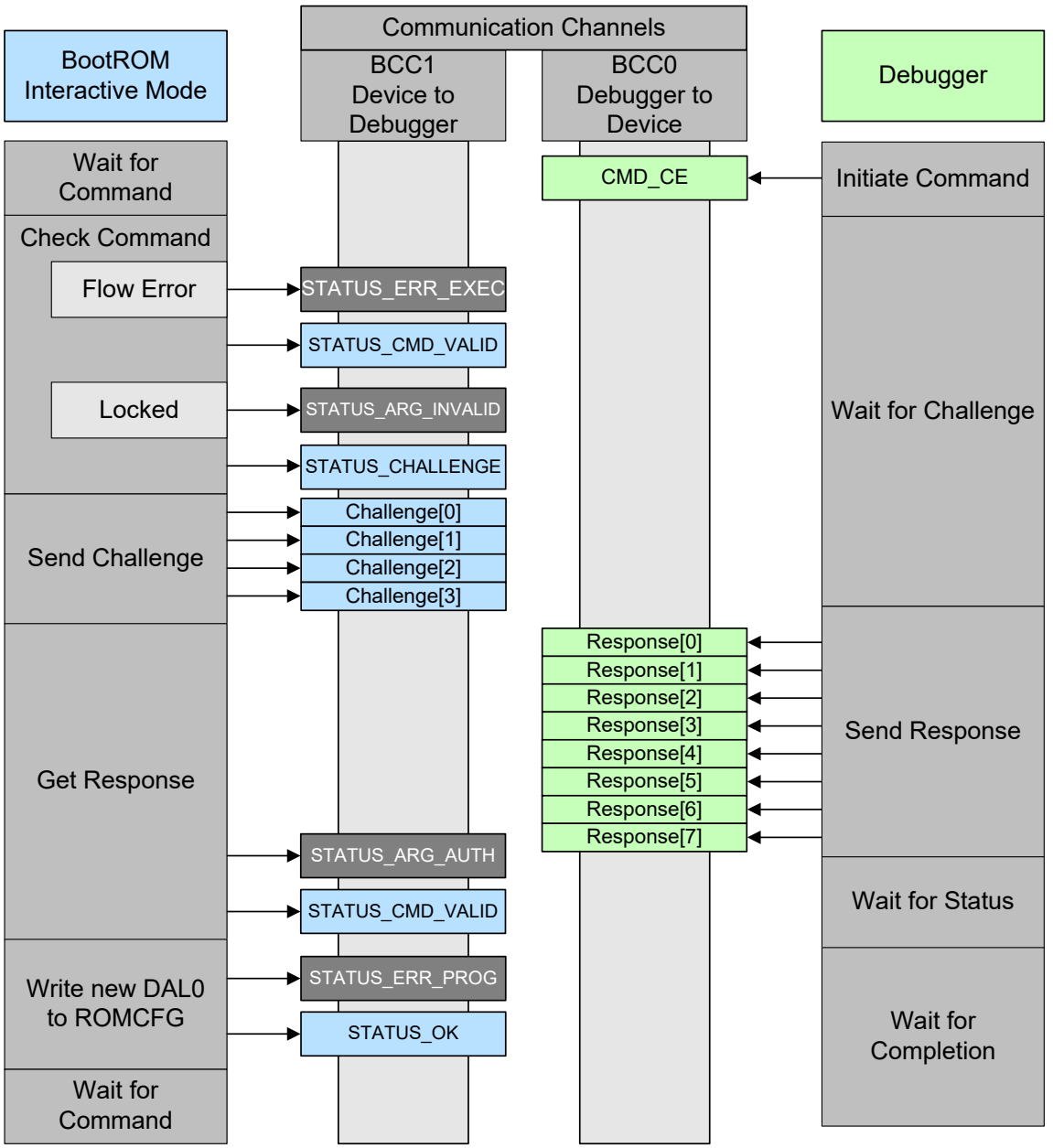
This command sets the Debug Access Level in ROMCFG to DAL0. The device must be reset for this value to take effect.

The Table 6-17 defines the command structure, and Figure 6-9 shows the communication sequence.

Table 6-17. `CMD_SDAL0` Definition

Type	Input	Size	Range	Output	Comment
Command	`CMD_SDAL0`	32 bits		STATUS_ERR_EXEC STATUS_ARG_INVALID STATUS_CMD_VALID STATUS_CHALLENGE	Refer to Table 6-9 and Table 6-10
Authenticate		128 bits		Challenge[]	LS Word first
Authenticate	Response[]	256 bits		STATUS_ARG_AUTH STATUS_CMD_VALID	LS Word first
Result		32 bits		STATUS_ERR_PROG STATUS_OK	Refer to Table 6-9 and Table 6-10

Figure 6-9. CMD_SDAL0 Message Sequence



6.5.7. Programming

6.5.7.1. General

This section describes the rddi_dap sequences required to:

- Get the Device ID
- Read memory (for each area of memory, programming operations are required to access it)
- Write memory (for each area of memory, programming operations are required to access it)
- Disable AHB exceptions

- Read the Debug Halting Control and Status Register (DHCSR)

6.5.7.1.1. Get Device ID

The code is equivalent to reading a word at address `0x4100_2120` (DSU_DID register).

Table 6-18. Get Device ID Sequence

Steps	rddi_dap
Select BANK0	`WriteDP(DP_SELECT, MEMAP_BANK_0)`
Access size: 32-bit	`WriteAP(MEM_AP_CSW, SIZE32)`
Set to read the Device Identification (DSU.DID) register	`WriteAP(MEM_AP_TAR, DSU_DID)`
	`ReadAP(MEM_AP_DRW, value)`

6.5.7.1.2. Read Memory

Table 6-19. Read Memory Sequence

Steps	rddi_dap
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`
Select BANK0	`WriteDP(DP_SELECT, MEMAP_BANK_0)`
Access size: 32-bit (or any other)	`WriteAP(MEM_AP_CSW, SIZE32)`
<begin loop>	
	`WriteAP (MEM_AP_TAR, Address)`
	`ReadAP(MEM_AP_DRW, value)`
<end loop>	
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`

6.5.7.1.3. Write SRAM

Table 6-20. Write SRAM Sequence

Steps	rddi_dap
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`
Select BANK0	`WriteDP(DP_SELECT, MEMAP_BANK_0)`
Set access	`WriteAP(MEM_AP_CSW, (SIZE32 | SADDRINC | HPROT[3] | RESERVED | MSTRDBG))` • Access size: 32-bit • Single Address Increment • User/Privilege Control (read-only) • Master type: Debug
<outer loop>	
Write Address to the Transfer Address Register (TAR)	`WriteAP(MEM_AP_TAR, Address)`
Write data to the device	`WriteAP(MEM_AP_DRW, Value)`
<end outer loop>	

6.5.7.1.4. DHCSR – Read Debug Halting Control and Status Register

Table 6-21. Read DHCSR Register Sequence

Steps	rddi_dap
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`
Select BANK0	`WriteDP(DP_SELECT, MEMAP_BANK_0)`
	`WriteAP(MEM_AP_CSW, (MSTRDBG | RESERVED | HPROT | SIZE32))`
Load the Debug Halting Control and Status Register (DHCRS) into the Transfer Address Register (TAR)	`WriteAP(MEM_AP_TAR, DHCSR)`
Read the Debug Halting Control and Status Register (DHCRS)	`ReadAP(MEM_AP_DRW, Value)`
Read the Read Buffer	`ReadAP(DP_RDBUFF, Value)`

6.5.7.1.5. Write Word

All Flash programming routines are described using word reads and writes. It is much easier to operate at that level.

This is referred to as `dap_write_word`.

Table 6-22. Write Word Sequence

Steps	rddi_dap
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`
Select BANK0	`WriteDP(DP_SELECT, MEMAP_BANK_0)`
Access size: 32-bit	`WriteAP(MEM_AP_CSW, SIZE32)`
	`WriteAP(MEM_AP_TAR, Address)`
	`WriteAP(MEM_AP_DRW, value)`
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`

6.5.7.1.6. Read Word

This is referred to as `dap_read_word`.

Table 6-23. Read Word Sequence

Steps	rddi_dap
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`
Select BANK0	`WriteDP(DP_SELECT, MEMAP_BANK_0)`
Access size: 32-bit	`WriteAP(MEM_AP_CSW, SIZE32)`
	`WriteAP (MEM_AP_TAR, Address)`
	`ReadAP(MEM_AP_DRW, value)`
Clear sticky bits	`WriteDP(DP_ABORT, (ORUNERRCLR | WDERRCLR | STKERRCLR | STKCMPCLR))`

6.5.7.1.7. Erase Memory

Erasing SRAM is equivalent to writing a block of data with the desired value using the Write SRAM procedure.

Erasing the entire Flash is performed through a Boot ROM Interactive mode command. Refer to the `CMD_CE_ALL` command in the *Command CE_ALL* section.

Erasing individual pages is performed through an NVM controller command.

Table 6-24. Erase Memory Sequence

Steps	Commands
Enable the Erase Command	`dap_write_word (NVMCTRL_CTRLB, NVMCTRL_CMD_FLPER);`
Set the Page Address	`dap_write_word (NVMCTRL_ADDR, page_addr);`
Wait for Erase to complete	`while (0 == (dap_read_word (NVMCTRL_INTFLAG) & 1));`
Disable the Erase Command	`dap_write_word (NVMCTRL_CTRLB, NVMCTRL_CMD_NOCMD);`

This sequence assumes that the page is not locked by LOCK fuses or BOOTPROT.

Relevant registers are:

- NVMCTRL_CTRLB `0x4100_4004`
- NVMCTRL _INTFLAG `0x4100_4014`
- NVMCTRL _ADDR `0x4100_4020`

6.5.7.1.8. Programming a Flash Page

Table 6-25. Programming a Flash Page Sequence

Steps	Commands
Enable automatic writes	`dap_write_word(NVMCTRL_CTRLB, NVMCTRL_CMD_FLWR);`
Wait for Unlock to complete	`while (0 == (dap_read_word(NVMCTRL_INTFLAG) & 1));`
<begin loop>	
Write Data	`dap_write_word(addr, data);`
Wait for Write to complete	`while (0 == (dap_read_word(NVMCTRL_INTFLAG) & 1));`
<end loop>	
Disable the Write Command	`dap_write_word(NVMCTRL_CTRLB, NVMCTRL_CMD_NOCMD);`

This sequence assumes that the page is not locked by LOCK fuses or BOOTPROT.

6.5.7.1.9. Check If The Device Is Locked

Table 6-26. Check If The Device Is Locked Sequence

Steps	Commands
Read status register	`Status = dap_read_word(DSU_DAL);`
Check the value of bit 1:0	`Locked = !(Status & 0x00000003);`

If bits 1:0 are '0', then the device is locked and requires a Chip Erase before any other access can occur.

6.5.8. In-Circuit Debugging

6.5.8.1. Debug and Trace Support

The PIC32CM PL10 family implements only the SWD interface, no JTAG interface is available.

The Data Watchpoint and Trace (DWT) unit provides one Data Watch point comparator and execution monitoring.

The Flash Patch Breakpoint (FPB) unit provides hardware breakpointing with four instruction address comparators.

The Arm Micro Trace Buffer (MTB) is implemented. The *CoreSight MTB-M0+ Technical Reference Manual (ARM DDI 0486B)* provides a detailed description of the MTB operation.

6.5.8.2. Debugger Access Support

6.5.8.2.1. Debug and Access Ports (SW-DP And AHB-AP Subblocks)

Refer to the *SW-DP – Serial Wire Debug Port* section.

6.5.8.2.2. Breakpoint, Watchpoint and Trace Support

All devices implement one watchpoint comparator and four breakpoint comparators.

6.5.8.3. Debug Reset

The debugger needs to perform the following actions:

After Reset, wait until the DALUN bit is cleared before accessing the core registers. Then, read DHCSR.S_RESET_ST to check if a Reset has occurred, because until DALUN is cleared, the debugger has no access to any internal address space except for the DSU's external address space (i.e., offset `0x100-0x1FF`).

6.5.8.4. Operation Sequences

This section details the operation sequences for:

- Enter Debug mode
- Exit Debug mode
- Register reads and writes
- Run
- Halt
- Get halt status
- Get core registers
- Get Program Status Register (PSR)
- Get and set Program Counter
- Get and set Stack Pointer
- Single step
- Set core registers

6.5.8.4.1. Enter Debug Mode

Since the PIC32CM PL10 devices have a Boot ROM, the Reset vector catch cannot be used to prevent the CPU from executing the first instruction in the First Mutable Executable (FME), because the vector catch would halt the CPU on the first instruction in the Boot ROM while the device is locked (DAL = `0`).

To force the processor to enter the Debug state as soon as possible, and before executing the FME, the debugger must maneuver the Boot ROM into Park mode. Thereafter, the debugger sets DHCSR.C_DEBUGEN to '1' to enable halting debug, and DHCSR.C_HALT to '`1`' to enter Debug state.

Reference: ARM DDI 0403E.b, *C1.4.1 Entering Debug state on leaving reset state*

Table 6-27. Enter Debug Mode Sequence

Steps	rddi_dap
Enable Debug	`WriteD32(DHCSR, (DBGKEY | C_DEBUGEN))`
Issue Halt	`WriteD32(DHCSR, (DBGKEY | C_HALT))`

6.5.8.4.2. Exit Debug Mode

The processor exits the Debug state:

- When the debugger writes '`0`' to DHCSR.C_HALT
- Upon receipt of an external restart request

If software clears DHCSR.C_HALT to '`0`' while the processor is in Debug state, a subsequent read of the DHCSR that returns '`1`' for both C_HALT and S_HALT indicates that the processor has re-entered the Debug state because it has detected a new debug event.

Reference: ARM DDI 0403E.b, *C1.5.4 Exiting Debug state*

Table 6-28. Exit Debug Mode Sequence

Steps	rddi_dap
Exit Debug mode	`WriteD32(DHCSR, (DBGKEY | C_HALT))`

6.5.8.4.3. Register Reads and Writes

To read and write device registers:

Table 6-29. Register Reads and Writes Sequence

Steps	rddi_dap
Read register	`ReadD32(Register, Value)`
Write register	`WriteD32(Register, Value)`

6.5.8.4.4. Run

The processor exits the Debug state:

- When the debugger writes '0' to DHCSR.C_HALT
- When it receives an external restart request

Reference: ARM DDI 0403E.b, *C1.5.4 Exiting Debug state*

Table 6-30. Run Sequence

Steps	rddi_dap
	`WriteD32(DHCSR, (DBGKEY | C_MASKINTS | C_HALT | C_DEBUGEN))`
	`WriteD32(DHCSR, (DBGKEY | C_MASKINTS | C_STEP | C_DEBUGEN))`
	`WriteD32(DHCSR, (DBGKEY | C_HALT | C_DEBUGEN))`
	`WriteD32(DHCSR, (DBGKEY | C_DEBUGEN))`

6.5.8.4.5. Halt

A debugger can use halting debug stepping to exit the Debug state, execute a single instruction, and then re-enter the Debug state. Halting debug stepping is active when all of the following apply:

- DHCSR.C_DEBUGEN is set to '1' (halting debug enabled)
- DHCSR.C_STEP is set to '1' (halting stepping enabled)
- The processor is in the Non-Debug state

Reference: ARM DDI 0403E.b, *C1.5.1 Debug stepping - Halting debug stepping*

Table 6-31. Halt Sequence

Steps	rddi_dap
Debug halt	`WriteD32(DHCSR, (DBGKEY | DHCSR.C_DEBUGEN | DHCSR.C_MASKINTS | DHCSR.C_HALT))`

6.5.8.4.6. Get Halt Status

Check the state of the Halt flag in the Debug Halting Control and Status Register (DHCSR).

Table 6-32. Get Halt Status Sequence

Steps	rddi_dap
Read the DHCSR	`ReadD32(DHCSR, Value)`
Poll for Halt Status bit is set	`S_HALT = 1`

6.5.8.4.7. Get Core Registers

To transfer a data word from an Arm core register, special-purpose register, or Floating-point extension register, a debugger:

- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '0' to indicate a read access. This write clears the DHCSR.S_REGRDY bit to '0'.
- Polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the value of the selected register to DCRDR.
- Reads the required value from DCRDR

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-33. Get Core Registers Sequence

Steps	rddi_dap
<begin loop 14 times>	
	`WriteD32(DCRSR, COREREG)`
Poll DHCSR for S_REGRDY	
	`ReadD32(DCRDR, value)`
Increment COREREG	
<end loop>	

6.5.8.4.8. Get Program Status Register (PSR)

To transfer a data word from an Arm core register, special-purpose register, or Floating-point extension register, a debugger:

- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '0' to indicate a read access. This write clears the DHCSR.S_REGRDY bit to '0'.
- Polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the value of the selected register to DCRDR.
- Reads the required value from DCRDR

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-34. Get PSR Sequence

Steps	rddi_dap
	`WriteD32(DCRSR, COREREG.R16)`
Poll DHCSR for S_REGRDY	
	`ReadD32(DCRDR, value)`

6.5.8.4.9. Get Program Counter

To transfer a data word from an Arm core register, special-purpose register, or Floating-point extension register, a debugger:

- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '0' to indicate a read access. This write clears the DHCSR.S_REGRDY bit to '0'.
- Polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the value of the selected register to DCRDR.
- Reads the required value from DCRDR

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-35. Get Program Counter Sequence

Steps	rddi_dap
	`WriteD32(DCRSR, COREREG.R15)`
Poll DHCSR for S_REGRDY	
	`ReadD32(DCRDR, value)`

6.5.8.4.10. Set Program Counter

To transfer a data word to an Arm core register, special-purpose register, or Floating-point extension register, a debugger:

- Writes the required word (PC value) to DCRDR
- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '1' to indicate a write access. This write clears the DHCSR S_REGRDY bit to '0'.
- If required, polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the DCRDR value to the selected register.

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-36. Set Program Counter Sequence

Steps	rddi_dap
	`WriteD32(DCRDR, Program Counter Value)`
	`WriteD32(DCRSR, (Address | REGWnR = 1))`
Poll DHCSR for S_REGRDY	

6.5.8.4.11. Get Stack Pointer

To transfer a data word from an Arm core register, special-purpose register, or Floating-point extension register, a debugger:

- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '0' to indicate a read access. This write clears the DHCSR.S_REGRDY bit to '0'.
- Polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the value of the selected register to DCRDR.
- Reads the required value from DCRDR

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-37. Get Stack Pointer Sequence

Steps	rddi_dap
	`WriteD32(DCRSR, COREREG.R13)`
Poll DHCSR for S_REGRDY	
	`ReadD32(DCRDR, value)`

Note: This sequence is similar to the Get Program Counter sequence, but it involves accessing a different register.

6.5.8.4.12. Set Stack Pointer

To transfer a data word to an Arm core register, special-purpose register, or Floating-point extension register, a debugger:

- Writes the required word (PC value) to DCRDR
- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '1' to indicate a write access. This write clears the DHCSR S_REGRDY bit to '0'.
- If required, polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the DCRDR value to the selected register.

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-38. Set Stack Pointer Sequence

Steps	rddi_dap
	`WriteD32(DCRDR, Stack Pointer Value)`

Table 6-38. Set Stack Pointer Sequence (continued)	
Steps	**rddi_dap**
	`WriteD32(DCRSR, (Address | REGWnR = 1))`
Poll DHCSR for S_REGRDY	

Note: This sequence is similar to the Set Program Counter sequence, but it involves accessing a different register.

6.5.8.4.13. Single Step

A debugger can use debug monitor stepping to return from the Debug Monitor exception handler, execute a single instruction, and then re-enter the Debug Monitor exception handler. Debug monitor stepping is active when all of the following conditions are met:

- DHCSR.C_DEBUGEN is set to '0' (halting debug disabled)
- DEMCR.MON_EN is set to '1' (monitor debug enabled)
- DEMCR.MON_STEP is set to '1' (monitor stepping enabled)
- The execution priority is below the priority of the Debug Monitor exception

Reference: ARM DDI 0403E.b, *C1.5.1 Debug stepping*

Table 6-39. Single Step Sequence

Steps	rddi_dap
	`WriteD32(DHCSR, (DBGKEY | C_MASKINTS | C_HALT | C_DEBUGEN))`
	`WriteD32(DHCSR, (DBGKEY | C_MASKINTS | C_STEP | C_DEBUGEN))`
	`WriteD32(DHCSR, (DBGKEY | C_HALT | C_DEBUGEN))`

6.5.8.4.14. Set Core Registers

To transfer a data word to an Arm core register, special-purpose register or Floating-point extension register, a debugger:

- Writes the required word to DCRDR
- Writes to the DCRSR, with the REGSEL value indicating the required register and the REGWnR bit set to '1' to indicate a write access. This write clears the DHCSR S_REGRDY bit to '0'.
- If required, polls DHCSR until DHCSR.S_REGRDY reads as '1'. This indicates that the processor has transferred the DCRDR value to the selected register.

Reference: ARM DDI 0403E.b, *C1.6.4 Debug Core Register Data Register, DCRDR*

Table 6-40. Set Core Registers

Steps	rddi_dap
	`WriteD32(DCRDR, Register Value)`
	`WriteD32(DCRSR, (Register | REGWnR = 1))`
Poll DHCSR for S_REGRDY = `1`	`ReadD32(DHCSR, S_REGRDY)`

6.6. Dependencies

The Boot ROM uses the user-configured boot settings defined in the Boot Configuration (BOOTCFG) fuses. No external hardware connections are required for Boot ROM operation, provided the device operates within the *Electrical Specifications*.

Note: Users must ensure that the pin selected for configuring the BOOTCFG.BOOT_GPIOSEL fuse is configured in a way that does not interfere with the overall hardware functionality of the application. Refer also to the *BOOTCFG - Boot Configuration Fuses* chapter.

Clocks

When the Boot ROM transitions the device to the first mutable application executable, the clock system is configured to use OSCHF at 4 MHz as the source for GCLK0 and, consequently, MCLK. This ensures consistent and deterministic behavior after the boot phase. Any clock configuration changes are performed by the user application *after* control is transferred from the Boot ROM.

Configuration Flash Memory (CFM)

The Boot ROM reads the Boot Configuration (BOOTCFG) fuses and writes their configuration information, such as Boot Loader Size or Watchdog Timer parameters, to the respective peripheral registers. These user-defined fuse values must be correctly programmed using the supported device programmer to ensure proper device operation. Refer to the *BOOTCFG - Boot Configuration Fuses* section for additional information.

Note: Incorrect fuse settings may prevent the device from booting successfully. In extreme cases, invalid fuse configurations may render the device permanently non-functional.

Memories

The Boot ROM may use the SRAM during start-up. Before transferring control to the user application, all locations of the SRAM are written to 0x0000. Do not rely on any pre-initialized data in this region when developing the first mutable application code.

Debug Operation

The device supports both hot-plugging and cold-plugging. The interaction between the external debugger and the device is controlled by the Debugger Access Level (DAL0 or DAL2). See the *Functional Description* for more details.

7. SYSCTRL - System Controller

The Boot Control (BOOTCTRL) register configures certain options of the Boot ROM. Refer to the *BootROM - Boot ROM* chapter and the *ARM®v6-M Architecture Reference Manual* for more details.

The other registers of the SYSCTRL provide a read interface for the User Configuration (USERCFG) fuse and the Temperature Sensor Calibration (TEMPSENSE) data.

7.1. Register Summary - SYSCTRL

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x7F	Reserved									
0x80	BOOTCTRL	7:0						FORCEPARK	BREXTEN	SYSINIT
		15:8								
		23:16								
		31:24								
0x84	USERCFG	7:0	CRCBOOT	CRCSEL						
		15:8						SUT[2:0]		
		23:16								
		31:24								
0x88	TEMPSENSE	7:0	TEMPSENSE0[7:0]							
		15:8	TEMPSENSE0[15:8]							
		23:16	TEMPSENSE1[7:0]							
		31:24	TEMPSENSE1[15:8]							

7.1.1. Boot Control

Name: BOOTCTRL
Offset: 0x0080
Default: 0x00000000
Property: R

Bit	31	30	29	28	27	26	25	24
Access								
Default								

Bit	23	22	21	20	19	18	17	16
Access								
Default								

Bit	15	14	13	12	11	10	9	8
Access								
Default								

Bit	7	6	5	4	3	2	1	0
						FORCEPARK	BREXTEN	SYSINIT
Access						R/W	R/W	R
Default						0	0	0

Bit 2 – FORCEPARK Force Park Mode Entry

Bit 1 – BREXTEN BREXT Enable

Bit 0 – SYSINIT System Initialization Ongoing

7.1.2. User Configuration

Name: USERCFG
Offset: 0x0084
Default: 0x00000000
Property: R

At start-up and after a Reset, the bit field values of this register are determined either by the Default value given here or by the corresponding BOOTCFG fuse. Refer to the *BOOTCFG - Boot Configuration Fuses* section for details.

Bit	31	30	29	28	27	26	25	24
Access								
Default								

Bit	23	22	21	20	19	18	17	16
Access								
Default								

Bit	15	14	13	12	11	10	9	8
						SUT[2:0]		
Access						R	R	R
Default						0	0	0

Bit	7	6	5	4	3	2	1	0
	CRCBOOT	CRCSEL						
Access	R	R						
Default	0	0						

Bits 10:8 – SUT[2:0] Start-up Time
These bits select the start-up time between power-on and code execution.

Value	Name	Description
0x0	0MS	0 ms
0x1	1MS	1 ms
0x2	2MS	2 ms
0x3	4MS	4 ms
0x4	8MS	8 ms
0x5	16MS	16 ms
0x6	32MS	32 ms
0x7	64MS	64 ms

Bit 7 – CRCBOOT CRC Boot
This bit enables a Cyclic Redundancy Check (CRC) of the BOOT section at start-up.

Value	Name	Description
0	DISABLE	No CRC
1	ENABLE	CRC of the Boot section

Bit 6 – CRCSEL CRC Polynomial Selection
This bit is used to select which polynomial type for Cyclic Redundancy Checks.

Value	Name	Description
0	CRC16	CRC-16-CCITT

Value	Name	Description
1	CRC32	CRC-32 (IEEE 802.3)

7.1.3. Temperature Sensor Calibration

Name: TEMPSENSE
Offset: 0x0088
Default: factory calibration value
Property: R

These values are determined and written to each device individually during production. See the Temperature Sensor chapter for information on how to use them.

Bit	31	30	29	28	27	26	25	24
	TEMPSENSE1[15:8]							
Access	R	R	R	R	R	R	R	R
Default								

Bit	23	22	21	20	19	18	17	16
	TEMPSENSE1[7:0]							
Access	R	R	R	R	R	R	R	R
Default								

Bit	15	14	13	12	11	10	9	8
	TEMPSENSE0[15:8]							
Access	R	R	R	R	R	R	R	R
Default								

Bit	7	6	5	4	3	2	1	0
	TEMPSENSE0[7:0]							
Access	R	R	R	R	R	R	R	R
Default								

Bits 31:16 – TEMPSENSE1[15:0] Temperature Sensor Calibration Offset
These bits provide the offset calibration factor for the temperature sensor. See the Temperature Sensor chapter for information on how to use them.

Bits 15:0 – TEMPSENSE0[15:0] Temperature Sensor Calibration Slope
These bits provide the slope calibration factor for the temperature sensor. See the Temperature Sensor chapter for information on how to use them.

8. Peripherals Configuration Summary

Table 8-1. Peripherals Configuration Summary for PIC32CM6408PL10

Peripheral	Base Address	IRQ	AHB CLK	APB CLK	Generic CLK	PAC		Events		DMA	Sleep Walking
			Enabled at reset	Enabled at reset	Index	Index	Prot at reset	User	Generator	Index	
AHB-APB-Bridge A	0x40000000	-	Y	-	-	-	-	-	-	-	N/A
PAC	0x40000000	INT 0	Y	Y	-	0	N	-	60:ACCERR	-	N/A
PM	0x40000400	INT 0	-	Y	-	1	N	-	-	-	N/A
MCLK	0x40000800	INT 0	-	Y	-	2	N	-	-	-	Y
RSTC	0x40000C00	-	-	Y	-	3	N	-	-	-	N/A
OSCCTRL	0x40001000	INT 0	-	Y		4	N	-		-	Y
OSC32KCTRL	0x40001400	INT 0	-	Y	-	5	N	-	1: XOSCC32K_CFDO	-	Y
SUPC	0x40001800	LVDET 0, LVDRDY 0	-	Y	-	6	N	-	2: BODDVDDIO2DT, 3: VLM	-	N/A
GCLK	0x40001C00	-	-	Y	-	7	N	-	-	-	N/A
WDT	0x40002000	EW 1	-	Y	-	8	N	-	-	-	Y
RTC	0x40002400	INT 2	-	Y	-	9	N	-	4: CMP0 5: CMP1 6: OVF 7:14: PER0-7	-	Y
EIC	0x40002800	NMI; EXTINT[15:0] 3	-	Y	EIC0	10	N	-	15-30:EXTINT0-15	-	Y
AHB-APB-Bridge B	0x41000000	-	Y	-	-	-	-	-	-	-	N/A
PORT	0x41000000	-		Y	-	0	N	1-4: EV0-3	-	-	Y
DSU	0x41002000	-	Y	Y	-	1	Y	-	61: SWCCSTAT, 62: COMP	21: DCC0, 22: DCC1 -	-
NVMCTRL	0x41004000	INT 4	Y	Y	-	2	N	-	-	-	Y
DMAC	0x41006000	TCMPL0, SUSP0, TERR0, TCMPL1, SUSP1, TERR1 5	Y	N/A	-	3	N	5-6: CH0-1	31-32: CH0-1	-	Y
MTB	0x41008000	-	-	N/A	-	4	N	21: START 22: STOP	-	-	N/A
HMATRIXHS	0x4100BFFF					5					
AHB-APB-Bridge C	0x42000000	-	Y	-	-	-	-	-	-	-	N/A
EVSYS	0x42000000	INT 6	-	N	1-4: one per Channel 0-3	0	N	-	-	-	Y
SERCOM0	0x42000400	SERCOM0 7	-	N	5: SLOW 6: CORE	1	N	-	-	1: RX 2: TX	Y
SERCOM1	0x42000800	SERCOM1 8	-	N	7: SLOW 8: CORE	2	N	-	-	3: RX 4: TX	Y
TC0	0x42000C00	INT 9	-	N	9: TC	3	N	7: EVU	33:OVF, 34: MC0, 35: MC1	5:OVF, 6: MC0, 7: MC1	Y
TC1	0x42001000	INT 10	-	N	9: TC	4	N	8: EVU	36:OVF, 37: MC0, 38: MC1	8:OVF, 9: MC0, 10: MC1	Y
TC2	0x42001400	INT 11			10: TC	5		9: EVU	39:OVF, 40: MC0, 41: MC1	11:OVF, 12: MC0, 13: MC1	

Table 8-1. Peripherals Configuration Summary for PIC32CM6408PL10 (continued)

Peripheral	Base Address	IRQ	AHB CLK	APB CLK	Generic CLK	PAC		Events		DMA	Sleep Walking
			Enabled at reset	Enabled at reset	Index	Index	Prot at reset	User	Generator	Index	
TCC0	0x42001800	INT 12	-	N	11: TCC	6	N	10-11: EV0-1	42: OVF	14: OVF	Y
									43: TRG		
								12-15: MC0-3	44: CNT	15-18: MC0-3	
									45-48: MC0-3		
ADC	0x42001C00	ADC 13	-	N	-	7	N	16: START	49: RESRDY	19: RESRDY	Y
									50: SAMPRDY	20: SAMPRDY	
									51: WCMP	-	
AC	0x42002000	INT 14	-	N	-	8	N	-	52-53: COMP0-1	-	Y
									54: WIN0		
CCL	0x42002400	-	-	N	12: ccl	9	N	17-20: LUTIN0-3	55-58: LUTOUT0-3	-	Y
PTC	0x42002800	-	Y	-	-	10	-	-	-	-	N/A
SYSCTRL	0x42002C00	-	Y	-	-	11	-	-	-	-	N/A

9. Clock System

This chapter contains a summary of the clock distribution and terminology in the PIC32CM6408PL10 device. For a more detailed description, refer to the peripheral chapters and the *GCLK - Generic Clock Controller* chapter.

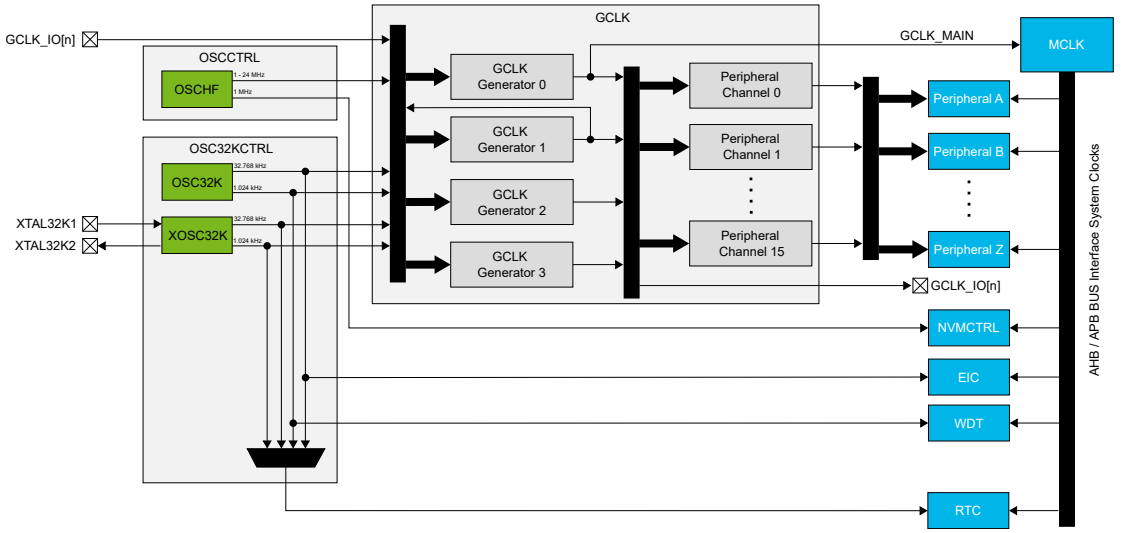
9.1. Clock Distribution

The clock system of the PIC32CM6408PL10 device consists of several clock sources, the Generic Clock Generator module (GCLK), and the Main Clock module (MCLK), each containing its own control logic.

A clock source provides a time base that is used by the peripherals in the device. The GCLK module clocks the majority of the device's peripherals, with a few exceptions. The MCLK module clocks the CPU. Some peripherals are clocked directly from the clock sources.

The figure below provides an overview of the clock system.

Figure 9-1. Clock Distribution



9.2. Clock System Features

The PIC32CM6408PL10 clock distribution system features are as follows:

- **Clock sources (oscillators)** controlled by OSCCTRL and OSC32KCTRL:
  - A clock source provides a time base that is used by other modules, such as the Generic Clock Generators. Examples of clock sources include the internal high-frequency oscillator (OSCHF) and the external 32.768 kHz oscillator (XOSC32K).
- **Generic Clock Controller (GCLK)**, which generates, controls, and distributes asynchronous clocks, consisting of:
  - *Generic Clock Generators*: These contain programmable prescalers that can use any of the system clock sources. See the *GCLK - Generic Clock Controller* chapter for the number of available GCLK Generators. Any of the GCLK Generators (GCLK_GENn) can be sent to peripherals as needed. Generic Clock Generator 0 (GCLK_GEN0) generates the clock signal GCLK_MAIN, which is the only clock source for the Main Clock module.
  - *Generic Clocks*: These are the clock signals generated by the Generic Clock Generators and are typically the clock inputs for the peripherals of the system, also known as Peripheral Clocks (GCLK_PERIPH[n]). The Generic Clocks, through the Generic Clock Multiplexer, can use

any of the Generic Clock Generators as their clock source. Multiple instances of a peripheral will typically have a separate Generic Clock (GCLK_PERIPH[n]) for each instance.

- **Main Clock (MCLK)**:
  - The MCLK module generates and controls the synchronous clocks for the system. This includes the CPU, the bus clocks Advanced Peripheral Bus (APB) and Advanced High-Speed Bus (AHB), as well as the registers of the peripherals. For the PIC32CM6408PL10, all system clocks are derived from GCLK Generator 0 (GCLK_GEN0/GCLK_MAIN).
  - The system clocks can be prescaled using the CPU Clock Division register (MCLK.CPUDIV). All system clocks will be prescaled with the same prescaler.
  - When a peripheral that interfaces with the system clock (synchronous clock) is disabled, it automatically gates off the system clock.
  - The user can mask the system clock from a peripheral using the MCLK masking registers (MCLK.xxxMASK). See the *MCLK - Main Clock* chapter for details.

All clock sources are routed to the Generic Clock Controller (GCLK). The GCLK distributes clock signals from the clock sources to the Main Clock Controller and most peripherals. Some peripherals take clock signals directly from the clock sources. Refer to the *GCLK – Generic Clock Controller* section for details on the number of available GCLK instances. Any GCLK instance can be used to clock any peripheral, while MCLK always uses GCLK0 as its clock source (GCLK_GEN0/GCLK_MAIN).

9.3. Clocks After Reset

After a Device Reset, the Boot ROM configures the OSCHF oscillator to 4 MHz (OSCHF divided by 6) and assigns it as the clock source for GCLK Generator 0 (GCLK_GEN0), which is also the clock source for MCLK (GCLK_MAIN). All other GCLK generators and 32K clock sources are disabled by a Device Reset.

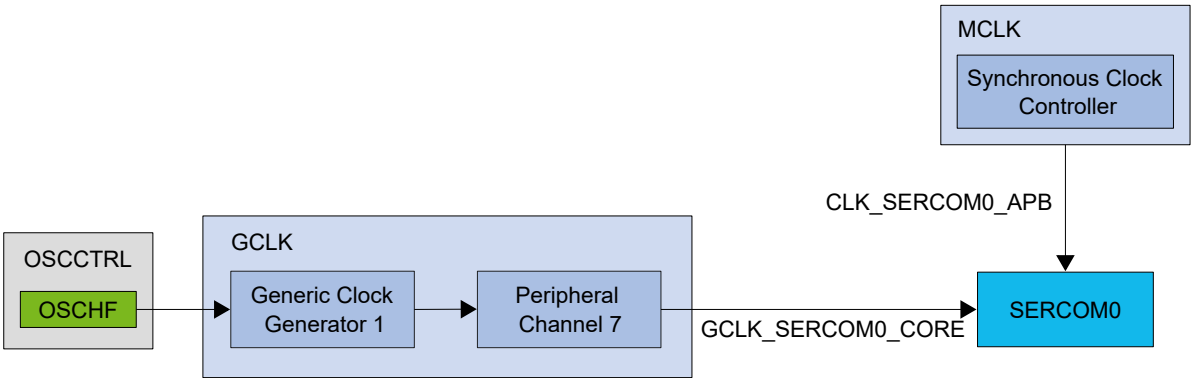
9.4. SERCOM Clocking Example

The figure below shows an example where SERCOM0 is clocked by OSCHF. OSCHF is enabled, Generic Clock Generator 1 uses OSCHF as its clock source, and feeds into Peripheral Channel 7. Generic Clock Peripheral Channel 7 connects to SERCOM0. The SERCOM0 interface, clocked by CLK_SERCOM0_APB, has been unmasked in the corresponding clock mask register in the MCLK module.

To customize the clock distribution, refer to the following registers and bit fields:

- The source oscillator for a Generic Clock Generator n (GCLK_GEN[n]) is selected by writing to the Source bit field in the Generator Control n register (GENCTRLn.SRC) in the GCLK peripheral
- A Peripheral Channel m can be configured to use a specific Generic Clock Generator by writing to the Generic Clock Generator bit field in the respective Peripheral Channel m register (PCHCTRLm.GEN) in the GCLK peripheral
- The Peripheral Channel number m is fixed for a given peripheral. See the Mapping table in the description of the PCHCTRLm register in the GCLK chapter.
- The APB and AHB clocks are enabled or disabled by writing to the respective bits in the clock masking registers in the Main Clock peripheral (MCLK.xxxMASK). See the *MCLK - Main Clock* chapter for details.

Figure 9-2. Example of SERCOM Clock



9.5. Synchronous and Asynchronous Clocks

As the CPU and peripherals can be clocked from different clock domains, peripheral accesses by the CPU need to be synchronized. In these cases, the peripheral includes a Synchronization Busy (SYNCBUSY) register that can be used to check if a synchronization operation is complete.

In the data sheet, references to Synchronous Clocks refer to the CPU and bus clocks (MCLK), while asynchronous clocks are those generated by the Generic Clock Controller (GCLK).

9.6. Register Synchronization

9.6.1. Overview

Most peripherals are connected to two clock domains:

- The peripheral digital bus interface, which is connected to the APB or AHB bus and is clocked by the corresponding synchronous clock in the Main Clock domain
- The peripheral core clock, which is connected to the peripheral Generic Clock (GCLK_PERIPH[n])

Communication between these clock domains must be synchronized. This mechanism is implemented in hardware, so the synchronization process takes place even if the peripheral Generic Clock is running from the same clock source and at the same frequency as the synchronous bus clock.

All registers in the bus interface are accessible without synchronization.

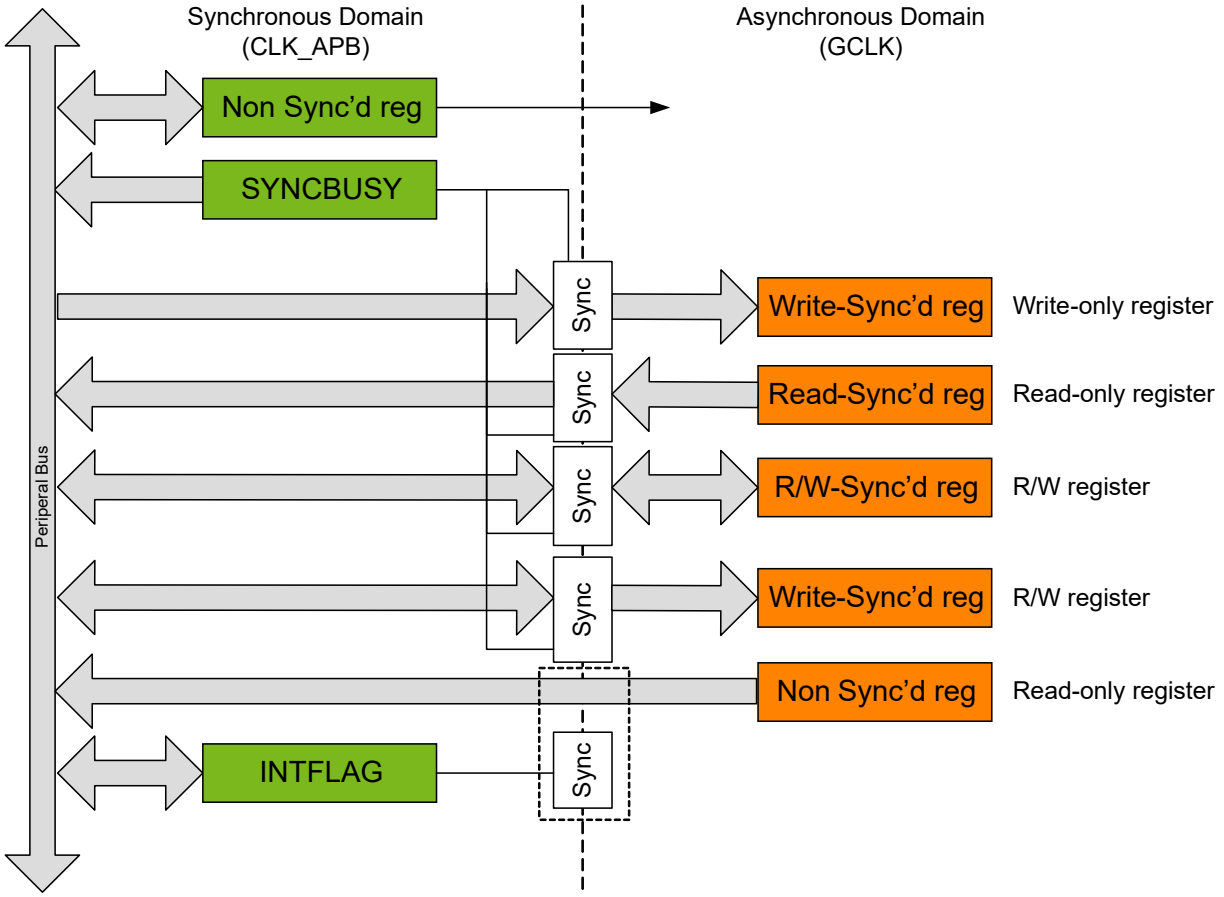
All registers in a peripheral are synchronized when written. Some registers are also synchronized when read.

Register descriptions will include the properties “Read-Synchronized” and/or “Write-Synchronized” if a register is synchronized.

As shown in the figure below, each register that requires synchronization has its own synchronization mechanism and its individual synchronization status bit in the peripheral Synchronization Busy (SYNCBUSY) register.

Note: For registers that require both read and write synchronization, the corresponding SYNCBUSY bit is shared.

Figure 9-3. Register Synchronization Overview



9.6.2. General Write Synchronization

Write Synchronization is triggered by writing to a register in the Peripheral Clock domain. The respective bit in the Synchronization Busy (SYNCBUSY) register will be set when the write synchronization starts and cleared when the write synchronization is complete. Refer to the *Synchronization Delay* section for details on the synchronization delay.

As long as write synchronization is ongoing for a register, any write attempts to that register will be discarded, and an error will be reported.

The following is an example of how this synchronization works. REGA and REGB are 8-bit core registers, and REGC is a 16-bit core register.

Example 9-1. Write-synchronization example

Offset	Register
0x00	REGA
0x01	REGB
0x02	REGC
0x03	

Synchronization is performed separately for each register, so multiple registers can be synchronized in parallel. Users can write to REGA (8-bit access) and then immediately write to REGB (8-bit access) without error.

Users can write to REGC (16-bit access) without affecting REGA or REGB. However, if a user writes to to REGC in two consecutive 8-bit accesses without waiting for synchronization, the second write attempt will be discarded and an error will be generated.

A 32-bit write access to offset 0x00 will write to all three registers. Note that REGA, REGB and REGC can be updated at different times because of independent write synchronization.

9.6.3. General Read-Synchronization

Read-Synchronized registers are synchronized each time the register value is updated, but the corresponding Synchronization Busy (SYNCBUSY) register bits are not set. Reading a read-synchronized register does not start a new synchronization; it returns the last synchronized value.

Note: The corresponding bits in SYNCBUSY will automatically be set when the device wakes up from a sleep mode, because read-synchronized registers must be synchronized. For this reason, reading a read-synchronized register before its corresponding SYNCBUSY bit is cleared will return the last synchronized value from before entering a sleep mode.

However, if a register is also write-synchronized, any write access while the SYNCBUSY bit is set will be executed successfully.

9.6.4. Completion of Synchronization

To check if the synchronization is complete, the user can either poll the relevant bits in SYNCBUSY or use the Synchronization Ready interrupt (if available). The synchronization Ready interrupt flag will be set when all ongoing synchronizations are complete, i.e., when all bits in SYNCBUSY are '0'.

9.6.5. Write-Synchronization for CTRLA.ENABLE

Writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE) of a peripheral will trigger write-synchronization and set the Enable Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE).

The updated value of the CTRLA.ENABLE bit will be available immediately after it is written.

The SYNCBUSY.ENABLE bit will be cleared by hardware when the operation is complete.

The Synchronization Ready interrupt (if available) cannot be used to enable write synchronization.

9.6.6. Software Reset Write Synchronization

Writing a '1' to the Software Reset (SWRST) bit in the CTRLA register will trigger write synchronization and set the Software Reset Synchronization Busy (SWRST) bit of the Synchronization Busy (SYNCBUSY) register.

When writing a '1' to the CTRLA.SWRST bit, it will immediately read as '1'. Both CTRLA.SWRST and SYNCBUSY.SWRST will be cleared by hardware when the peripheral has been reset. Writing a '0' to the CTRLA.SWRST bit has no effect. The Synchronization Ready interrupt (if available) cannot be used for Software Reset write synchronization.

Note: Not all peripherals have the SWRST bit in their respective CTRLA register.

9.6.7. Synchronization Delay

Synchronization will delay the duration of read and write access duration by a delay D, as shown in the equation below:

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

Where:

- P_{GCLK} is the period of the generic clock
- P_{APB} is the period of the peripheral bus clock

A normal peripheral bus register access duration is $2 \times P_{APB}$.

9.7. Enabling a Peripheral

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

- A running clock source selected from the OSCCTRL or OSC32KCTRL peripheral
- A Generic Clock Generator must be configured to use one of the running clock sources, and the generator must be enabled
- The Peripheral Channel that provides the Generic Clock to the peripheral must be configured to use the enabled and running Generic Clock Generator
- The user interface of the peripheral needs to be unmasked in the Main Clock Masking registers (MCLK.xxxMASK). If this is not done, all registers of the peripheral will read as 0x00, and any writes to the peripheral will be discarded.

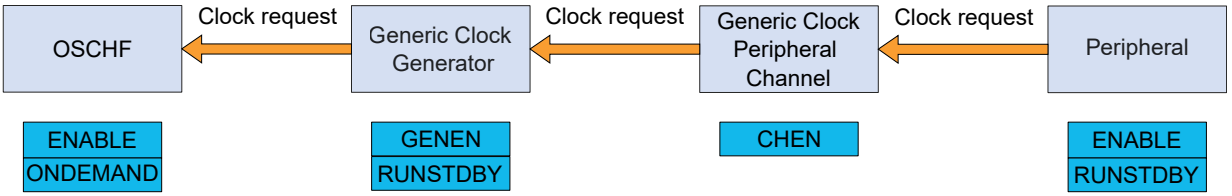
9.8. On-Demand Clock Requests

All clock sources in the system can operate in On-Demand mode, where the clock source is stopped when no peripherals are requesting it. Clock requests propagate from the peripheral, through the GCLK, to the clock source. If one or more peripherals are using a clock source, the clock source will be started or kept running. As soon as the clock source is no longer needed and no peripherals have an active request, the clock source will be stopped until it is requested again.

The clock request can reach the clock source only if the peripheral, the generic clock peripheral channel, and the Generic Clock Generator and its clock are enabled. The time taken from when a clock request is asserted to when the clock source being ready is dependent on the clock source start-up time, clock source frequency, and the divider used in the Generic Clock Generator.

The clock source routing sequence is shown in the figure below:

Figure 9-4. Clock Request Routing



The total start-up time, T_{START} , from when a clock request is made to when the clock is available for the peripheral, is between:

$T_{START\_MIN} = T_{SOURCE\_STARTUP} + T_{SOURCE\_PERIOD} + T_{SOURCE\_PERIOD\_DIVIDED}$

$T_{START\_MAX} = T_{SOURCE\_STARTUP} + (2 \times T_{SOURCE\_PERIOD}) + (2 \times T_{SOURCE\_PERIOD\_DIVIDED})$

Where:

- $T_{SOURCE\_STARTUP}$ is the clock source start-up time
- T_{SOURCE_PERIOD} is the clock source period
- $T_{SOURCE\_PERIOD\_DIVIDED}$ is the divided clock source period

The time between when the last active clock request stops and when the clock is shut down T_{STOP} is between:

$T_{STOP\_MIN} = T_{SOURCE\_PERIOD\_DIVIDED} + T_{SOURCE\_PERIOD}$

$T_{STOP\_MAX} = (2 \times T_{SOURCE\_PERIOD\_DIVIDED}) + (2 \times T_{SOURCE\_PERIOD})$

The On-Demand feature can be disabled individually for each clock source by writing a '0' to the On-Demand bit (ONDEMAND) in each clock source controller register. This will force the clock to always run, even if there are no active clock requests. While this eliminates the clock source start-up time, it will increase the total power consumption of the device.

9.9. Power Consumption Versus Speed

When considering power consumption versus execution speed of the PIC32CM6408PL10, some factors must be taken into account.

Due to the nature of asynchronous clocking for the peripherals, clocking a peripheral at a very low frequency will reduce the active power consumption of the peripheral.

At the same time, synchronization with the synchronous clock domain (MCLK) depends on the peripheral clock speed. With a slower peripheral clock, the response time will be longer and more time will be spent waiting for synchronization to complete.

10. GCLK – Generic Clock Controller

10.1. Features

- Provides a Device-Defined, Configurable Number of Peripheral Channel Clocks
- Wide Frequency Range:
  - Various clock sources
  - Embedded dividers

10.2. Overview

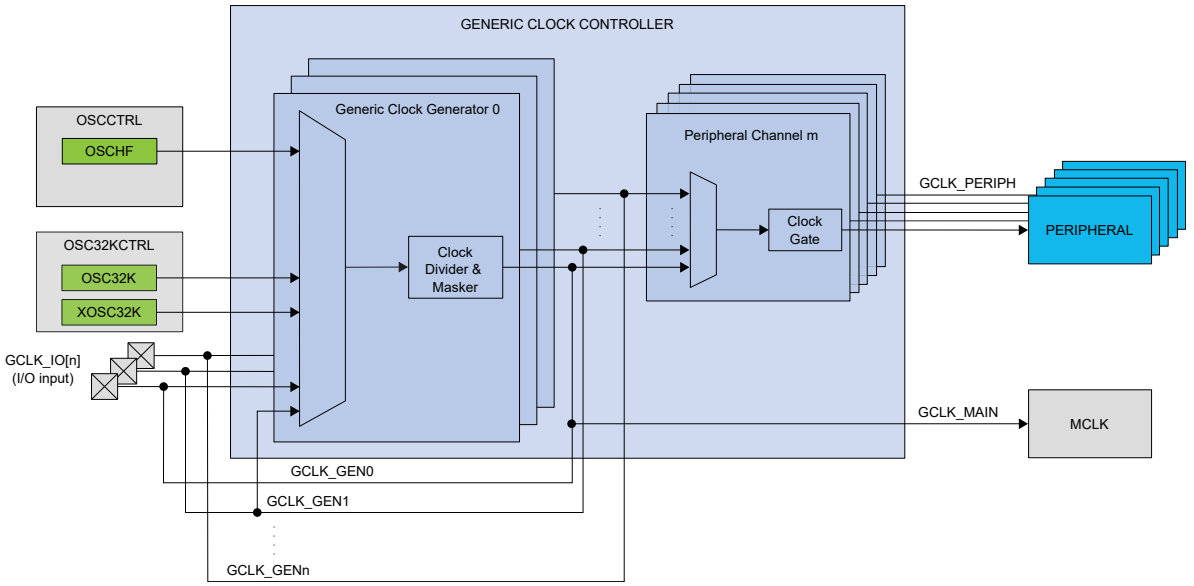
The Generic Clock Controller (GCLK) features four Generic Clock Generators that can provide a wide range of clock frequencies.

The generators can use six different external and internal oscillators as clock sources. The clock source of each generator can be divided down.

The outputs from the generators are used as generic clocks for the peripheral channels. There are 13 peripheral channels. Generator 0 is also the clock source for the main clock, as shown in the *GCLK Block Diagram*.

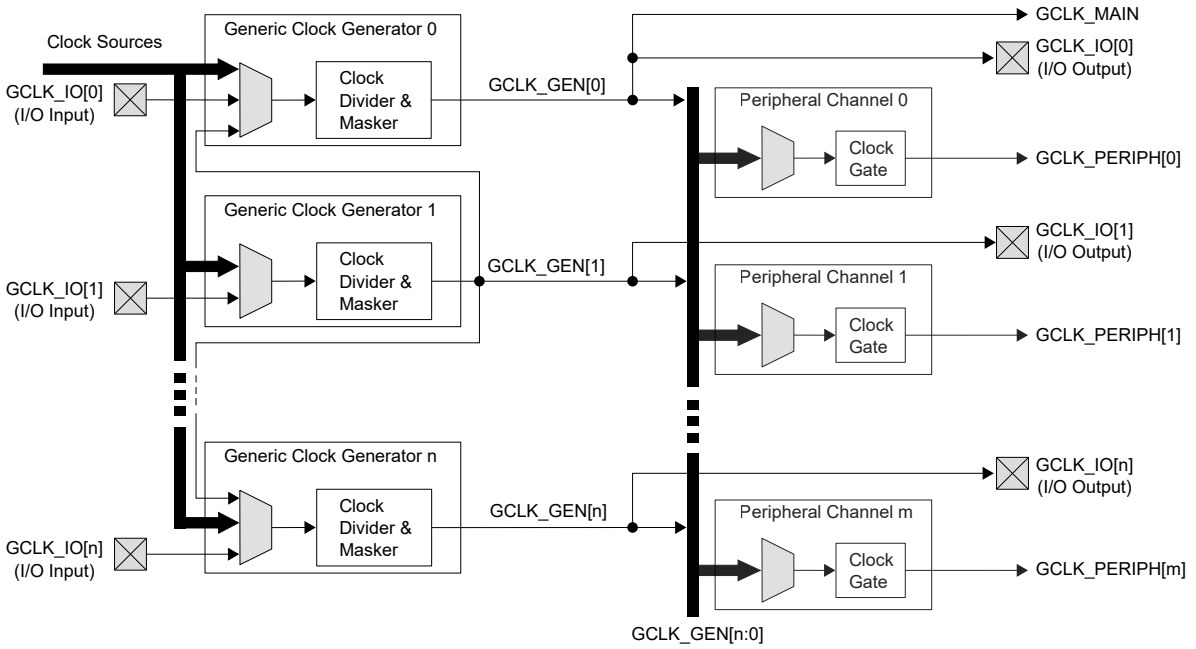
10.3. Block Diagram

Figure 10-1. Device Clocking Diagram



The generation of the Peripheral Clock signals (GCLK_PERIPH) and the Main Clock (GCLK_MAIN) can be seen in the following figure:

Figure 10-2. Device Clocking Diagram



Note: Generic Clock Generator 0 (GCLK_GEN0) is always the source of the GCLK_MAIN signal, and the default clock source is OSCHF divided by six, i.e. 4 MHz (after successful device initialization).

10.3.1. Signal Description

Table 10-1. GCLK External Signal Description

Signal Name	Type	Description
GCLK_IO[3:0](1)(2)	Digital I/O	• Clock source for Generators when input • Generic Clock signal when output

Notes:

1. One signal can be mapped to several pins.
2. The number of I/O signals connected to GCLK for this device is four.

10.4. Functional Description

10.4.1. Initialization

Follow these steps to configure and enable a generator:

1. Select the clock source by writing the corresponding value into the Generator Clock Source Select bit field of the Generator Control n register (GENCTRL[n].SRC).
2. Set the Division Factor bit field (GENCTRL[n].DIV) and the Divide Selection bit (GENCTRL[n].DIVSEL) according to the desired division of the clock source frequency.
3. Configure the Peripheral Channel Control n register. The generator used as a clock source for a peripheral must be selected by writing the desired value to the Generator Selection bit field (PCHCTRL[n].GEN), and the peripheral channel must be enabled by writing '1' to the Channel Enable bit (PCHCTRL[n].CHEN).
4. Enable the Generator by writing '1' to the generator enable bit of the GENCTRL[n] register (GENCTRL[n].GENEN).

Note: Each Generator n is configured by one dedicated register, GENCTRL[n].

Note: Each Peripheral Channel n is configured by one dedicated register, PCHCTRL[n].

10.4.2. Operation

The GCLK peripheral consists of four Generic Clock Generators that can provide a generic clock for the 13 Peripheral Channels and the Main Clock (GCLK_MAIN).

A clock source selected as input to a generator can either be used directly or prescaled in the generator. A generator output is used by one or more peripheral channels to provide a Peripheral Generic Clock Signal (GCLK_PERIPH) to the peripherals.

Note: Only Generator 0 (GCLK_GEN0) can provide a clock signal to the Main Clock (GCLK_MAIN).

10.4.2.1. Enabling, Disabling and Resetting

The GCLK peripheral has no enable/disable bit that enables or disables the complete peripheral.

All registers in the GCLK peripheral will be reset to their initial state.

A peripheral channel and its associated generator may be locked, causing any future writes to that peripheral channel or generator to be discarded. Refer to the *Configuration Lock* section for details.

10.4.2.2. Generic Clock Generators

Each Generic Clock Generator (GCLK_GEN[n]) can be set to run from one of six different clock sources, except for GCLK_GEN[1], which can be set to run from one of five clock sources. GCLK_GEN[1] is the only Generator that can be selected as a clock source for other Generators.

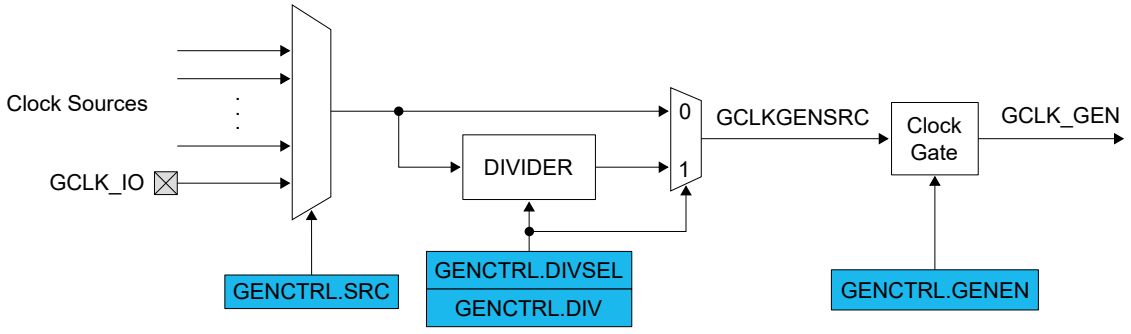
Each generator GCLK_GEN[n] can be connected to its corresponding GCLK_IO[n] pin. The GCLK_IO[n] pins can be configured either to act as a source for GCLK_GEN[n] or to output the clock signal generated by GCLK_GEN[n].

The selected source can be divided, and each Generator can be enabled or disabled independently.

Each GCLK_GEN[n] clock signal can be used as a clock source for Peripheral Channels. Each Generator output can be allocated to one or more Peripherals.

GCLK_GEN[0] is used as the clock source (GCLK_MAIN) for the synchronous clock controller inside the Main Clock. Refer to the *MCLK - Main Clock Controller* description for details on the synchronous clock generation. The default clock source for GCLK_GEN[0] is OSCHF divided by six, i.e. 4 MHz (after successful device initialization).

Figure 10-3. Generic Clock Generator



10.4.2.2.1. Enabling and Disabling a Generator

A Generator is enabled by writing the Generator Enable bit of the Generator Control register (GENCTRL[n].GENEN) to '1'.

A Generator is disabled by writing '0' to the generator enable bit of the Generator Control register (GENCTRL[n].GENEN). When the GENEN bit is '0', the Generator clock (GCLK_GENn) is disabled and gated.

10.4.2.2.2. Selecting a Clock Source for the Generator

Each Generator can individually select a clock source by writing to the Source Select bit field in the Generator Control register (GENCTRL[n].SRC).

Changing from one clock source (A) to another (B) can be done on the fly. If clock source B is not ready, the Generator will continue using clock source A. As soon as source B is ready, the Generator will switch to it.

During the switching operation, the Generator maintains clock requests to both clock sources A and B, and will release source A as soon as the switch is complete. The Generator Control n bit in the Synchronization Busy register (SYNCBUSY.GENCTRLn) will remain at '1' until the switch operation is completed.

The available clock sources are shown in the Clock System block diagram. Only Generic Clock Generator 1 can be used as a common source for all other Generators.

Note: Before switching Generic Clock Generator 0 (GCLK_GEN0) from clock source A to another clock source B, enable the On-Demand Operation feature (xxxCTRL.ONDEMAND) of clock source A to ensure a proper transition.

10.4.2.2.3. Changing the Clock Frequency and Duty Cycle

The selected clock source for a Generator can be divided by writing to the Division Factor bit field of the Generator Control register (GENCTRL[n].DIV). The actual division factor is determined by the divide selection bit of the GENCTRL[n] register (GENCTRL[n].DIVSEL).

When dividing a clock with an odd division factor, the duty-cycle will not be 50/50. Writing '1' to the improve duty cycle bit of the GENCTRL[n] register (GENCTRL[n].IDC) will result in a 50/50 duty cycle.

The resulting clock frequency f_{GCLK} is calculated as follows:

- If GENCTRL[n].DIVSEL = ' 0' and GENCTRL[n].DIV is either '0x0' or '0x1', the output clock will not be divided
- If GENCTRL[n].DIVSEL is '0' then the resulting frequency of the GCLK is calculated as follows:

  $$f_{GCLK} = \frac{GENCTRLn.SRC}{DIV}$$

  - If GENCTRL[n].DIV is an odd number, the Improve Duty Cycle bit of the GENCTRL[n] register (GENCTRL[n].IDC) must be set to '1'
  - If GENCTRL[n].DIV is an even number, the GENCTRL[n].IDC bit must be '0'
- If GENCTRL[n].DIVSEL is '1' then the resulting frequency of the GCLK is calculated as follows:

  $$f_{GCLK} = \frac{GENCTRLn.SRC}{2^{(DIV + 1)}}$$

  - GENCTRL[n].IDC must always be '0'

The following table shows the number of divider bits associated with each Generator:

Table 10-2. Generator Selection Overview

Value	Description	Enabled at Reset?	Divider bits (GENCTRLn.DIV)
0x0	Generator 0	Yes	8
0x1	Generator 1	No	16
0x2	Generator 2	No	8
0x3	Generator 3	No	8
0x4 – 0xf	Reserved	-	-

Note: The size of the DIV bit field may vary from Generator to Generator.

10.4.2.2.4. External Clock

The output clock of a Generator (GCLK_GENn) can be sent to an I/O pin (GCLK_IO[n]).

If the Output Enable bit in the Generator Control register (GENCTRL[n].OE) is '1' and the Generator is enabled (GENCTRL[n].GENEN = '1'), the Generator requests its clock source and the GCLK_GENn clock is output to an I/O pin.
Note: The I/O pin must be configured as a GCLK_GENn output by writing to the corresponding PORT registers before enabling the clock output on GLKC_IO[n]. See the *PORT* chapter for more information.

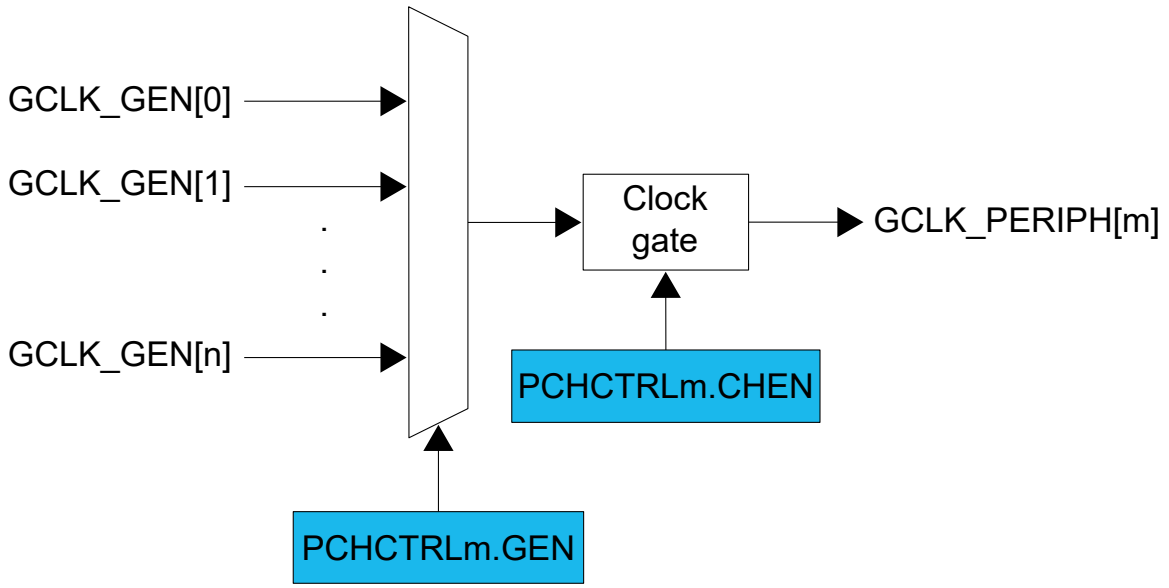
If GENCTRL[n].OE is '0', the corresponding I/O pin is set to a value selected by the Output Off Value bit field of the GENCTRL[n] register (GENCTRL[n].OOV). If GENCTRL[n].OOV is '0', the output clock will be low. If this bit is '1', the output clock will be high.

If the device enters Standby sleep mode while GENCTRL[n].OE is '1', the clock output on the I/O pin depends on the Run In Standby bit in the GENCTRL[n] register (GENCTRL[n].RUNSTDBY):

- If GENCTRL[n].RUNSTDBY is '0', the GCLK_IO[n] pin is held at the OOV value
- If GENCTRL[n].RUNSTDBY is '1', the GCLK_IO[n] pin outputs the GCLK_GENn clock

10.4.2.3. Peripheral Clocks

Figure 10-4. Peripheral Clocks



10.4.2.3.1. Enabling a Peripheral Clock

Before enabling a peripheral clock, one of the Generators must be enabled (GENCTRL[n].GENEN = '1') and selected as a source for the peripheral channel by setting the Generator Selection (GEN) bit field of the Peripheral Channel Control (PCHCTRL[n]) register. Any available Generator can be selected as a clock source for any peripheral channel. Refer to the following mapping table for mapping of peripherals to index n.

When a Generator has been selected, the peripheral clock is enabled by writing the Channel Enable bit in the PCHCTRL[n] register (PCHCTRL[n].CHEN) to '1'. The CHEN bit must be synchronized to the generic clock domain. The CHEN bit will continue to read as its previous state until the synchronization is complete.

The index values of the PCHCTRLm registers are shown in the following table. Use this index to configure the desired generator with the relevant peripheral channel:

Table 10-3. PCHCTRLm Mapping

Index [m]	Name	Description
0	GCLK_EIC	EIC
1	GCLK_EVSYS_CHANNEL0	EVSYS CHANNEL 0
2	GCLK_EVSYS_CHANNEL1	EVSYS CHANNEL 1
3	GCLK_EVSYS_CHANNEL2	EVSYS CHANNEL 2
4	GCLK_EVSYS_CHANNEL3	EVSYS CHANNEL 3
5	GCLK_SERCOM0_SLOW	SERCOM0 SLOW
6	GCLK_SERCOM0_CORE	SERCOM0 CORE
7	GCLK_SERCOM1_SLOW	SERCOM1 SLOW
8	GCLK_SERCOM1_CORE	SERCOM1 CORE
9	GCLK_TC0, GCLK_TC1	TC0, TC1
10	GCLK_TC2	TC2
11	GCLK_TCC0	TCC0
12	GCLK_CCL	CCL

10.4.2.3.2. Disabling a Peripheral Clock

A peripheral clock is disabled when the Channel Enable bit of the Peripheral Channel Control register (PCHCTRLm.CHEN) is '0'. The CHEN bit must be synchronized to the Generic Clock domain. The CHEN bit will remain in its previous state until synchronization is complete. The peripheral clock is gated when disabled.

10.4.2.3.3. Selecting the Clock Source for a Peripheral

When changing a peripheral clock source by writing to the Generator Selection bit field of the Peripheral Channel Control register (PCHCTRLm.GEN), the peripheral clock must be disabled before being re-enabled with the new clock source setting. The following actions prevent glitches during the transition:

1. Disable the peripheral channel by setting the Channel Enable bit of the PCHCTRL[n] register (PCHCTRL[n].CHEN) to '0'.
2. Verify that PCHCTRLm.CHEN reads '0'.
3. Change the source of the peripheral channel by writing the corresponding value to PCHCTRLm.GEN.
4. Re-enable the peripheral channel by writing PCHCTRLm.CHEN to '1'.
5. Verify that PCHCTRLm.CHEN reads '1'.

10.4.2.3.4. Configuration Lock

The peripheral clock configuration can be locked against further write accesses by setting the Write Lock bit in the Peripheral Channel Control register (PCHCTRL[n].WRTLOCK) to '1'. After this, all writes to the PCHCTRL[n] register will be ignored. The PCHCTRL[n] register can only be unlocked by a Device Reset.

The Generator source of a locked peripheral channel will also be locked. The corresponding Generator Control (GENCTRL[n]) register is locked and can only be unlocked by a Device Reset.

Note: Generator 0 (GCLK_GEN0) is an exception to the configuration lock. Since Generator 0 is the only clock source for GCLK_MAIN, it can not be locked.

10.4.2.4. Synchronization

Since the Main Clock (MCLK) domain and the Peripheral Clock (GCLK_PERIPH) domains are not synchronous, some registers must be synchronized when accessed.

An exception is the Channel Enable bit in the Peripheral Channel Control registers (PCHCTRL[n].CHEN). When changing this bit, the bit value must be read back to ensure that the synchronization is complete and to ensure glitch-free internal operation.

Note: Changing the CHEN bit value during ongoing synchronization will *not* generate an error.

The following registers are synchronized when written:

- The Generic Clock Generator Control register (GENCTRL[n])
- The Control A register (CTRLA)

Required write synchronization is denoted by the "Write Synchronized" property in the register description.

10.4.2.5. Generic Clock After Reset

GCLK will provide a default clock after a Device Reset. This means that both a clock source and a generator need to be configured and enabled automatically as the device starts up again.

The following table shows the Generator Control register (GENCTRL[n]) value after a Device Reset:

Table 10-4. GENCTRL[n] Reset Value after a Device Reset

GCLK Generator	Reset Value after a Device Reset
Generator 0	0x00000105
All other Generators	0x00000000

Refer to the GENCTRL[n] register for details on the values after reset and which bits and bit fields are affected.

The following table shows the Peripheral Channel Control registers (PCHCTRL[n]) and the function of the Write Lock bit (WRTLOCK) after a Device Reset:

Table 10-6. Reset Values After a User Reset or a Power Reset

Reset	WRTLOCK (before reset)	PCHCTRLm.GEN	PCHCTRLm.CHEN	WRTLOCK (after reset)
Power Reset	0	0x0	0x0	0x0
	1	0x0	0x0	0x0
User Reset	0	0x0	0x0	No change
	1	No change	No change	No change

Refer to the PCHCTRL[n] register for details on the values after reset and which bits and bit fields are affected.

10.4.3. Sleep Mode Operation

If the system is in a sleep mode where the Generic Clocks are stopped, a peripheral that needs its clock in order to execute a process must first request it from the Generic Clock Controller.

The Generic Clock Controller receives this request, determines which Generic Clock Generator is involved, and which clock source needs to be awakened. It then wakes up the respective clock source, enables the Generator and Peripheral Channel stages successively, and delivers the clock to the peripheral.

The Run In Standby bit in the Generator Control register (GENCTRL[n].RUNSTDBY) controls clock output to the pin during Standby sleep mode. If the bit is cleared, the Generator output is not available on the pin. When set, GCLK can continuously output the generator signal to the GCLK_IO[n] pin. Refer to the *External Clock* section for details.

The following table identifies when a Clock Generator is off in Standby mode, minimizing the power consumption:

Table 10-7. Clock Generator n Activity in Standby Mode

Request for Clock n present	GENCTRL[n].RUNSTDBY	GENCTRL[n].OE	Clock Generator n
Yes	-	-	Active

Table 10-7. Clock Generator n Activity in Standby Mode (continued)

Request for Clock n present	GENCTRL[n].RUNSTDBY	GENCTRL[n].OE	Clock Generator n
No	0	0	Off
No	0	1	Off
No	1	0	Off
No	1	1	Active

A delay may occur when the device is put into Standby sleep mode before the power is turned off. This delay is caused by running clock Generators: if the RUNSTDBY bit in the GENCTRL[n] register is '0', GCLK must verify that the clock is turned off. The duration of this verification depends on the clock frequency.

For additional information, refer to the *PM - Power Manager* section.

10.4.4. Debug Operations

When the CPU is halted in debug mode, the GCLK continues normal operation. If the GCLK is configured in a way that requires periodic servicing by the CPU through interrupts or similar mechanisms, improper operation or data loss may occur during debugging.

10.5. Dependencies

10.5.1. I/O Lines

Using the GCLK's I/O lines requires the I/O pins to be configured. Refer to the *Pinout and Multiplexing* section in the *PORT – I/O Pin Controller* chapter for details.

10.5.2. Power Management

The GCLK can operate in all sleep modes if required. Refer to the *PM - Power Manager* chapter for details on the different sleep modes.

10.5.3. Clocks

The GCLK bus clock (CLK_GCLK_APB) can be enabled and disabled in the Main Clock (MCLK) peripheral, and the default state of CLK_GCLK_APB can be found in the *Peripheral Clock Masking* section in the *MCLK - Main Clock* chapter.

10.5.4. DMA

Not applicable.

10.5.5. Interrupts

Not applicable.

10.5.6. Events

Not applicable.

10.5.7. Analog Connections

Not applicable.

10.6. Register Summary - GCLK

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	SYNCBUSY	7:0			GENCTRL3	GENCTRL2	GENCTRL1	GENCTRL0		
		15:8								
		23:16								
		31:24								
0x08 ... 0x1F	Reserved									
0x20	GENCTRL[0]	7:0				SRC[4:0]				
		15:8			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
		23:16	DIV[7:0]							
		31:24	DIV[15:8]							
0x24	GENCTRL[1]	7:0				SRC[4:0]				
		15:8			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
		23:16	DIV[7:0]							
		31:24	DIV[15:8]							
0x28	GENCTRL[2]	7:0				SRC[4:0]				
		15:8			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
		23:16	DIV[7:0]							
		31:24	DIV[15:8]							
0x2C	GENCTRL[3]	7:0				SRC[4:0]				
		15:8			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
		23:16	DIV[7:0]							
		31:24	DIV[15:8]							
0x30 ... 0x7F	Reserved									
0x80	PCHCTRL[0]	7:0	WRTLOCK	CHEN			GEN[3:0]			
		15:8								
		23:16								
		31:24								
...										
0xB0	PCHCTRL[12]	7:0	WRTLOCK	CHEN			GEN[3:0]			
		15:8								
		23:16								
		31:24								

10.6.1. Synchronization Busy

Name: SYNCBUSY
Offset: 0x04
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			GENCTRL3	GENCTRL2	GENCTRL1	GENCTRL0		
Access			R	R	R	R		
Reset			0	0	0	0		

Bits 2, 3, 4, 5 – GENCTRLn Generator Control n Synchronization Busy
This bit is cleared when synchronization of the Generator Control n (GENCTRL[n]) register between clock domains is complete, or when a clock switching operation is complete.
This bit is set when the GENCTRL[n] register synchronization between clock domains is started.

10.6.2. Generator Control

Name: GENCTRL[n]
Offset: 0x20 + n*0x04 [n=0..3]
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

The GENCTRL[n] register controls the settings of Generator n.

A Device Reset will reset all GENCTRL[n] registers. The Reset values of the GENCTRL[n] registers are shown in the *Generic Clock After Reset* section and are listed below:

- Generator 0: `0x00000105`
- All other Generators: `0x00000000`

Bit	31	30	29	28	27	26	25	24
	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	23	22	21	20	19	18	17	16
	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	15	14	13	12	11	10	9	8
			RUNSTDBY	DIVSEL	OE	OOV	IDC	GENEN
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

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

Bits 31:16 – DIV[15:0] Division Factor
This bit field holds the division value for the corresponding Generator. The actual division factor used is dependent on the state of the Divide Selection bit (GENCTRL[n].DIVSEL). The number of relevant DIV bits for each Generator are shown in the table below. Bit values outside the specified range will be ignored.

Table 10-8. Division Factor Bits

Generic Clock Generator	Division Factor Bits
Generator 0	8 division factor bits - DIV[7:0]
Generator 1	16 division factor bits - DIV[15:0]
Generator 2	8 division factor bits - DIV[7:0]
Generator 3	8 division factor bits - DIV[7:0]

Bit 13 – RUNSTDBY Run in Standby
This bit is used to keep the Generator running in Standby sleep mode as long as it is configured to output to a dedicated GCLK_IO pin. If the Output Enable bit (GENCTRL[n].OE) is '0', this bit has no effect, and the generator will only run if a peripheral requires the clock.

Value	Description
0	The Generator is stopped in Standby sleep mode, and the GCLK_IO pin state will depend on the value of the Output Off Value bit (GENCTRL[n].OOV)
1	The Generator is kept running, and the output is routed to its dedicated GCLK_IO pin when the device is in Standby sleep mode

Bit 12 – DIVSEL Divide Selection

This bit determines how the division factor of the clock source of the Generator will be calculated from DIV.
If the clock source should not be divided, DIVSEL must be set to '0' and the GENCTRL[n].DIV value must be either '0x0' or '0x1'.

Value	Name	Description
0	DIV1	The Generator clock frequency equals the clock source frequency divided by GENCTRL[n].DIV
1	DIV2	The Generator clock frequency equals the clock source frequency divided by $2^{(GENCTRL[n].DIV + 1)}$

Bit 11 – OE Output Enable

This bit is used to output the generator clock to the corresponding pin (GCLK_IO[n]), as long as GCLK_IO is not selected as the generator source in the Generator Clock Source Selection bit field (GENCTRL[n].SRC).

Value	Description
0	No Generator clock signal on the GCLK_IO[n] pin
1	The Generator clock signal is output on the corresponding GCLK_IO[n] pin, unless GCLK_IO is selected as a generator source in the GENCTRL[n].SRC bit field

Bit 10 – OOV Output Off Value

This bit is used to control the clock output value on pin (GCLK_IO[n]) pin when the Generator is turned off or the OE bit is zero, as long as GCLK_IO is not defined as the Generator source in the GENCTRL[n].SRC bit field.

Value	Description
0	The GCLK_IO[n] pin will be LOW when the generator is turned off or when the OE bit is '0'
1	The GCLK_IO[n] pin will be HIGH when the generator is turned off or when the OE bit is '0'

Bit 9 – IDC Improve Duty Cycle

This bit is used to improve the duty cycle of the Generator output to 50/50 for odd division factors.
Notes:

- If DIVSEL = '1', then IDC must always be set to '0'
- If DIVSEL = '0', then:
  - If DIV is an odd number, IDC must be set to '1'
  - If DIV is an even number, IDC must be set to '0'

Value	Description
0	The generator output clock duty cycle is not balanced to 50/50 for odd division factors
1	The generator output clock duty cycle is 50/50

Bit 8 – GENEN Generator Enable

This bit is used to enable and disable Generator n.

Value	Description
0	Generator n is disabled
1	Generator n is enabled

Note: For Generator 0, the reset value of this bit is '1' since it is automatically enabled after a Device Reset. For all other generators, the reset value of this bit is '0'.

Bits 4:0 – SRC[4:0] Generator Clock Source Selection
These bits select the Generator clock source.

Value	Name	Description
0x00	OSCHF	OSCHF oscillator output
0x01	GCLKIN	Generator input pin (GCLK_IO)
0x02	GCLKGEN1	Generic Clock Generator 1 output
0x03	OSC32K	OSC32K oscillator output
0x04	XOSC32K	XOSC32K oscillator output
0x05-0x1F	Reserved	-

10.6.3. Peripheral Channel Control n

Name: PCHCTRL[n]
Offset: 0x80 + n*0x04 [n=0..12]
Reset: 0x00000000
Property: PAC Write-Protection

PCHTRL[n] controls the settings of Peripheral Channel n.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 7 – WRTLOCK Write Lock
Writing a '0' to this bit will set the Peripheral Channel register (PCHCTRL[n]) and the associated Generator register (GENCTRL[n]) to an unlocked state.
Writing a '1' to this bit will cause future writes to the PCHCTRL[n] register to be discarded. The control register of the corresponding Generator m (GENCTRL[m]), as assigned by the Generator Select (GEN) bit field of the Peripheral Control (PCHCTRL[n]) register, will also be locked. It can only be unlocked by a Device Reset.
Note: Generator 0 (GENCTRL[0]) cannot be locked.

Value	Description
0	The Peripheral Channel register and the associated Generator register are not locked
1	The Peripheral Channel register and the associated Generator register are locked

Bit 6 – CHEN Channel Enable
This bit is used to enable and disable a Peripheral Channel.

Value	Description
0	The Peripheral Channel is disabled
1	The Peripheral Channel is enabled

Note: This bit requires synchronization. When writing to this bit, its value must be polled (read back and checked against ' expected value) to ensurethat synchronization is complete.

Bits 3:0 – GEN[3:0] Generator Selection
This bit field selects the Generator to be used as clock source of a peripheral.
A Device Reset will reset all the PCHCTRL[n] registers.

Value	Name	Description
0x0	GCLK0	Generic Clock Generator 0
0x1	GCLK1	Generic Clock Generator 1
0x2	GCLK2	Generic Clock Generator 2
0x3	GCLK3	Generic Clock Generator 3

Note: Refer to the *Peripheral Clocks* section for details on the peripheral channels and their corresponding peripheral instances available on the device (PCHCTRL[n] mapping).

11. MCLK – Main Clock

11.1. Features

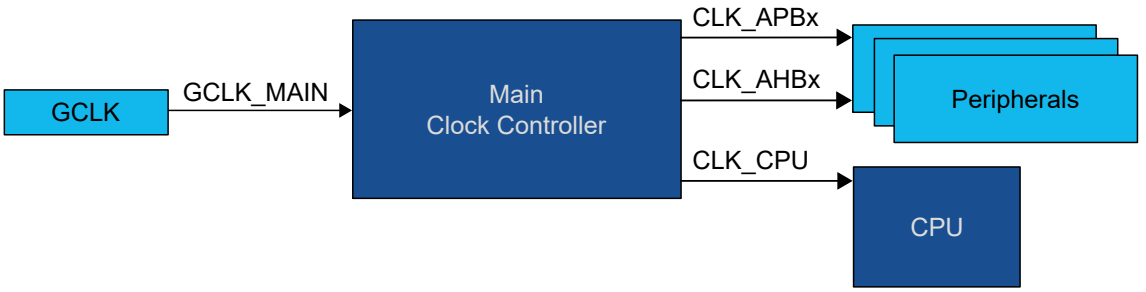
- Generates CPU, Advanced High-Performance Bus (AHB), and Advanced Peripheral Bus (APB) System Clocks
  - Clock source and division factor from GCLK
  - Clock prescaler with 1x to 128x division
- Safe Run-Time Clock Switching From GCLK
- Module-Level Clock Gating Through Maskable Peripheral Clocks

11.2. Overview

The Main Clock (MCLK) controls the generation of synchronous clocks for the device. It receives its clock signal (GCLK_MAIN) from the Generic Clock Module (GCLK) and provides synchronous system clocks to the CPU and all the peripherals connected to the AHBx and APBx buses.

11.3. Block Diagram

Figure 11-1. MCLK Block Diagram



11.3.1. Signal Description

Not applicable.

11.4. Functional Description

11.4.1. Initialization

After a Reset, the default clock source for 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 prescaler division.

11.4.2. Operation

The GCLK_MAIN clock signal from the GCLK peripheral is the source for the main clock, which is the common root for the synchronous clocks for the CPU, APBx and AHBx peripherals. The GCLK_MAIN can be divided by an 8-bit prescaler. Each of the derived clocks can operate from any divided or undivided main clock, ensuring synchronous clock sources for each clock domain. The CPU clock domain can be changed on the fly to respond to varying application loads. The clocks for each peripheral within a clock domain can be individually masked to reduce power consumption in inactive peripherals. Depending on the sleep mode, some clock domains can be turned off.

11.4.2.1. Enabling, Disabling and Resetting

The MCLK peripheral is always enabled and cannot be reset.

11.4.2.2. Selecting the Main Clock Source

Refer to the Generic Clock Controller (GCLK) section for details on how to configure the clock source of the GCLK_MAIN clock.

11.4.2.3. Selecting the Synchronous Clock Division Ratio

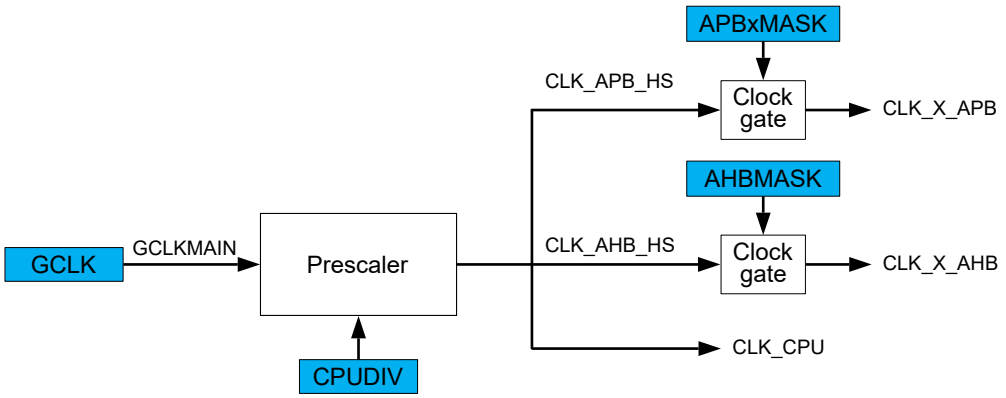
The main clock, GCLK_MAIN, feeds an 8-bit prescaler, which can be used to generate the synchronous clocks. By default, the synchronous clocks run on the undivided main clock. The user can select a prescaler division for the CPU clock domain by writing to the CPU Clock Division (CPUDIV) register, resulting in a CPU clock domain frequency determined by the following equation:

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

If the application attempts to write forbidden values in the CPUDIV register, the register is written, but these invalid values are not used. Additionally, a violation is reported to the Peripheral Access Controller (PAC) peripheral.

The CPUDIV register can be written without halting or disabling peripherals. Writing to CPUDIV allows a new clock setting to be applied simultaneously to all synchronous clocks within the corresponding clock domain.

Figure 11-2. Synchronous Clock Selection and Prescaler



11.4.2.4. Clock Ready Flag

There is a slight delay after writing to CPUDIV before the new clock settings become effective. During this interval, the Clock Ready flag in the Interrupt Flag Status and Clear register (INTFLAG.CKRDY) is '0'. If the Clock Ready Interrupt Enable bits in the Interrupt Enable Clear and Interrupt Enable Set registers (INTENCLR/SET.CKRDY) are '1', the Clock Ready interrupt will be triggered when the new clock setting is effective. The clock settings must not be rewritten while INTFLAG.CKRDY is '0'. The system may become unstable or hang, and a violation is reported to the PAC peripheral.

11.4.2.5. Peripheral Clock Masking

It is possible to disable or enable the AHB or APB clock for a peripheral by writing the corresponding bit in the Clock Mask registers (AHBMASK and APBxMASK) to '0' and '1', respectively. The default state of the peripheral clocks is given by the peripheral bit Reset value in the AHBMASK and APBxMASK registers.

When the APB clock is not supplied to a peripheral, its registers cannot be read or written. The peripheral can be re-enabled by setting the corresponding mask bit to '1'.

Some peripherals may be connected to multiple clock domains (for instance, AHB and APB), in which case it will have several mask bits.

Clocks must be switched off only if it is certain that the peripheral 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 MCLK peripheral (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.

11.4.3. Sleep Mode Operation

The MCLK will operate in all sleep modes if a synchronous clock is required in these modes.

11.4.4. Debug Operations

When the CPU is halted in debug mode, the MCLK continues normal operation. If the MCLK is configured to require periodic servicing by the CPU through interrupts or similar mechanisms, improper operation or data loss may occur during debugging.

11.5. Dependencies

11.5.1. I/O Lines

Not applicable.

11.5.2. Power Management

The MCLK will continue to operate in all sleep modes if a synchronous clock is required during these modes.

11.5.3. Clocks

A generic clock (GCLK_MAIN) is required to clock the MCLK. This clock must be configured in the generic clock controller before using the MCLK. Refer to the *GCLK – Generic Clock Controller* chapter for details.

11.5.4. DMA

Not applicable.

11.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use MCLK interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 11-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
MCLK	CKRDY	The synchronous CPU, AHBx and APBx clocks have frequencies as set in the CPU Clock Division (CPUDIV) register	The CPU Clock Division Factor bit field in the CPU Clock Division register (CPUDIV.CPUDIV)

11.5.6. Events

Not applicable.

11.5.7. Analog Connections

Not applicable.

11.6. Register Summary - MCLK

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	Reserved									
0x01	INTENCLR	7:0								CKRDY
0x02	INTENSET	7:0								CKRDY
0x03	INTFLAG	7:0								CKRDY
0x04	CPUDIV	7:0	CPUDIV[7:0]							
0x05 ... 0x0F	Reserved									
0x10	AHBMASK	7:0	DMAC	HSRAM	NVMCTRL	HMATRIXHS	DSU	APBC	APBB	APBA
		15:8							BROM	PAC
		23:16								
		31:24								
0x14	APBAMASK	7:0	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
		15:8						EIC	RTC	WDT
		23:16								
		31:24								
0x18	APBBMASK	7:0			HMATRIXHS	MTB	DMAC	NVMCTRL	DSU	PORT
		15:8								
		23:16								
		31:24								
0x1C	APBCMASK	7:0	ADC0	TCC0	TC2	TC1	TC0	SERCOM1	SERCOM0	EVSYS
		15:8					SYSCTRL	PTC	CCL	AC
		23:16								
		31:24								

11.6.1. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x01
Reset: 0x00
Property: PAC Write-Protection

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

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

Bit 0 – CKRDY Clock Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Clock Ready Interrupt Enable bit, which disables the Clock Ready interrupt.

Value	Description
0	The Clock Ready interrupt is disabled
1	The Clock Ready interrupt is enabled

11.6.2. Interrupt Enable Set

Name: INTENSET
Offset: 0x02
Reset: 0x00
Property: PAC Write-Protection

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

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

Bit 0 – CKRDY Clock Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Clock Ready Interrupt Enable (CKRDY) bit, thereby enabling the Clock Ready interrupt.

Value	Description
0	The Clock Ready interrupt is disabled
1	The Clock Ready interrupt is enabled

11.6.3. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x03
Reset: 0x01
Property: –

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

Bit 0 – CKRDY Clock Ready Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set when the synchronous CPU, AHBx and APBx clocks have frequencies as set in the CPU Clock Division (CPUDIV) register. If INTENCLR.CKRDY or INTENSET.CKRDY is set to '1', an interrupt will be generated.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Clock Ready interrupt flag.

11.6.4. CPU Clock Division

Name: CPUDIV
Offset: 0x04
Reset: 0x01
Property: PAC Write-Protection

Bit	7	6	5	4	3	2	1	0
	CPUDIV[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	1

Bits 7:0 – CPUDIV[7:0] CPU Clock Division Factor
This bit field defines the division ratio of the main clock prescaler for the CPU clock domain. The resulting frequencies must never exceed the specified maximum frequency for each clock domain.

Value	Name	Description
0x01	DIV1	Divide by 1
0x02	DIV2	Divide by 2
0x04	DIV4	Divide by 4
0x08	DIV8	Divide by 8
0x10	DIV16	Divide by 16
0x20	DIV32	Divide by 32
0x40	DIV64	Divide by 64
0x80	DIV128	Divide by 128
Other	—	Reserved

11.6.5. AHB Bridge Mask

Name: AHBMASK
Offset: 0x10
Reset: 0x000003FF
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							BROM	PAC
Access							R/W	R/W
Reset							1	1

Bit	7	6	5	4	3	2	1	0
	DMAC	HSRAM	NVMCTRL	HMATRIXHS	DSU	APBC	APBB	APBA
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

Bit 9 – BROM Boot ROM AHB Clock Enable

Value	Description
0	The AHB clock for the Boot ROM is disabled
1	The AHB clock for the Boot ROM is enabled

Bit 8 – PAC PAC AHB Clock Enable

Value	Description
0	The AHB clock for the Peripheral Access Controller peripheral (CLK_PAC_AHB) is disabled
1	CLK_PAC_AHB is enabled

Bit 7 – DMAC DMAC AHB Clock Enable

Value	Description
0	The AHB clock for the Direct Memory Access Controller peripheral (CLK_DMAC_AHB) is disabled
1	CLK_DMAC_AHB is enabled

Bit 6 – HSRAM HSRAM AHB Clock Enable
Refer to the *HMATRIXHS - High-Speed Bus System* section for more information about High Speed RAM (HSRAM).

Value	Description
0	The AHB clock for the High speed RAM (HSRAM) is disabled
1	The AHB clock for the HSRAM is enabled

Bit 5 – NVMCTRL NVMCTRL AHB Clock Enable

Value	Description
0	The AHB clock for the Non-Volatile Memory Controller peripheral (CLK_NVMCTRL_AHB) is disabled
1	CLK_NVMCTRL_AHB is enabled

Bit 4 – HMATRIXHS HMATRIXHS AHB Clock Enable

Value	Description
0	The AHB clock for the High-Speed Bus System (HMATRIXHS) is disabled
1	The AHB clock for the HMATRIXHS is enabled

Bit 3 – DSU DSU AHB Clock Enable

Value	Description
0	The AHB clock for the Device Service Unit peripheral (CLK_DSU_AHB) is disabled
1	CLK_DSU_AHB is enabled

Bit 2 – APBC APB Bridge C AHB Clock Enable

Value	Description
0	The AHB clock for the APB Bridge C is disabled
1	The AHB clock for the APB Bridge C is enabled

Bit 1 – APBB APB Bridge B AHB Clock Enable

Value	Description
0	The AHB clock for the APB Bridge B is disabled
1	The AHB clock for the APB Bridge B is enabled

Bit 0 – APBA APB Bridge A AHB Clock Enable

Value	Description
0	The AHB clock for the APB Bridge A is disabled
1	The AHB clock for the APB Bridge A is enabled

11.6.6. APB Bridge A Mask

Name: APBAMASK
Offset: 0x14
Reset: 0x000007FF
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						EIC	RTC	WDT
Access						R/W	R/W	R/W
Reset						1	1	1

Bit	7	6	5	4	3	2	1	0
	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
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

Bit 10 – EIC EIC APB Clock Enable

Value	Description
0	The APB clock for the External Interrupt Controller peripheral (CLK_EIC_APB) is disabled
1	CLK_EIC_APB is enabled

Bit 9 – RTC RTC APB Clock Enable

Value	Description
0	The APB clock for the Real-Time Clock peripheral (CLK_RTC_APB) is disabled
1	CLK_RTC_APB is enabled

Bit 8 – WDT WDT APB Clock Enable

Value	Description
0	The APB clock for the Watchdog Timer peripheral (CLK_WDT_APB) is disabled
1	CLK_WDT_APB is enabled

Bit 7 – GCLK GCLK APB Clock Enable

Value	Description
0	The APB clock for the Generic Clock Controller peripheral (CLK_GCLK_APB) is disabled
1	CLK_GCLK_APB is enabled

Bit 6 – SUPC SUPC APB Clock Enable

Value	Description
0	The APB clock for the Supply Controller peripheral (CLK_SUPC_APB) is disabled

Value	Description
1	CLK_SUPC_APB is enabled

Bit 5 – OSC32KCTRL OSC32KCTRL APB Clock Enable

Value	Description
0	The APB clock for the 32.768 kHz Oscillators Controller peripheral (CLK_OSC32KCTRL_APB) is disabled
1	CLK_OSC32KCTRL_APB is enabled

Bit 4 – OSCCTRL OSCCTRL APB Clock Enable

Value	Description
0	The APB clock for the Oscillators Controller peripheral (CLK_OSCCTRL_APB) is disabled
1	CLK_OSCCTRL_APB is enabled

Bit 3 – RSTC RSTC APB Clock Enable

Value	Description
0	The APB clock for the Reset Controller peripheral (CLK_RSTC_APB) is disabled
1	CLK_RSTC_APB is enabled

Bit 2 – MCLK MCLK APB Clock Enable

Value	Description
0	The APB clock for the Main Clock peripheral (CLK_MCLK_APB) is disabled
1	CLK_MCLK_APB is enabled

Bit 1 – PM PM APB Clock Enable

Value	Description
0	The APB clock for the Power Management peripheral (CLK_PM_APB) is disabled
1	CLK_PM_APB is enabled

Bit 0 – PAC PAC APB Clock Enable

Value	Description
0	The APB clock for the Peripheral Access Controller peripheral (CLK_PAC_APB) is disabled
1	CLK_PAC_APB is enabled

11.6.7. APB Bridge B Mask

Name: APBBMASK
Offset: 0x18
Reset: 0x00000027
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 5 – HMATRIXHS HMATRIXHS APB Clock Enable

Value	Description
0	The APB clock for the High-Speed Bus System (HMATRIXHS) is disabled
1	The APB clock for the HMATRIXHS is enabled

Bit 4 – MTB MTB APB Clock Enable

Value	Description
0	The APB clock for the Micro Trace Buffer (MTB) is disabled
1	The APB clock for the MTB is enabled

Bit 3 – DMAC DMAC APB Clock Enable

Value	Description
0	The APB clock for the Direct Memory Access Controller peripheral (CLK_DMAC_APB) is disabled
1	CLK_DMAC_APB is enabled

Bit 2 – NVMCTRL NVMCTRL APB Clock Enable

Value	Description
0	The APB clock for the Non-Volatile Memory Controller peripheral (CLK_NVMCTRL_APB) is disabled
1	CLK_NVMCTRL_APB is enabled

Bit 1 – DSU DSU APB Clock Enable

Value	Description
0	The APB clock for the Device Service Unit peripheral (CLK_DSU_APB) is disabled
1	CLK_DSU_APB is enabled

Bit 0 – PORT PORT APB Clock Enable

Value	Description
0	The APB clock for the I/O Pin Controller peripheral (CLK_PORT_APB) is disabled
1	CLK_PORT_APB is enabled

11.6.8. APB Bridge C Mask

Name: APBCMASK
Offset: 0x1C
Reset: 0x00000800
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					SYSCTRL	PTC	CCL	AC
Access					R/W	R/W	R/W	R/W
Reset					1	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADC0	TCC0	TC2	TC1	TC0	SERCOM1	SERCOM0	EVSYS
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 11 – SYSCTRL SYSCTRL APB Clock Enable

Value	Description
0	The APB clock for the System Controller (SYSCTRL) is disabled
1	The APB clock for the SYSCTRL is enabled

Bit 10 – PTC PTC APB Clock Enable

Value	Description
0	The APB clock for the Peripheral Touch Controller peripheral (CLK_PTC_APB) is disabled
1	CLK_PTC_APB is enabled

Bit 9 – CCL CCL APB Clock Enable

Value	Description
0	The APB clock for the Configurable Custom Logic peripheral (CLK_CCL_APB) is disabled
1	CLK_CCL_APB is enabled

Bit 8 – AC AC APB Clock Enable

Value	Description
0	The APB clock for the Analog Comparator peripheral (CLK_AC_APB) is disabled
1	CLK_AC_APB is enabled

Bit 7 – ADC0 ADC0 APB Clock Enable

Value	Description
0	The APB clock for the ADC0 instance of the Analog-to-Digital Converter peripheral (CLK_ADC0_APB) is disabled

Value	Description
1	CLK_ADC0_APB is enabled

Bit 6 – TCC0 TCC0 APB Clock Enable

Value	Description
0	The APB clock for the TCC0 instance of the Timer/Counter for Control Applications peripheral (CLK_TCC0_APB) is disabled
1	CLK_TCC0_APB is enabled

Bit 5 – TC2 TC2 APB Clock Enable

Value	Description
0	The APB clock for the TC2 instance of the Timer/Counter peripheral (CLK_TC2_APB) is disabled
1	CLK_TC2_APB is enabled

Bit 4 – TC1 TC1 APB Clock Enable

Value	Description
0	The APB clock for the TC1 instance of the Timer/Counter peripheral (CLK_TC1_APB) is disabled
1	CLK_TC1_APB is enabled

Bit 3 – TC0 TC0 APB Clock Enable

Value	Description
0	The APB clock for the TC0 instance of the Timer/Counter peripheral (CLK_TC0_APB) is disabled
1	CLK_TC0_APB is enabled

Bit 2 – SERCOM1 SERCOM1 APB Clock Enable

Value	Description
0	The APB clock for the SERCOM1 instance of the Serial Communication Interface peripheral (CLK_SERCOM1_APB) is disabled
1	CLK_SERCOM1_APB is enabled

Bit 1 – SERCOM0 SERCOM0 APB Clock Enable

Value	Description
0	The APB clock for the SERCOM0 instance of the Serial Communication Interface peripheral (CLK_SERCOM0_APB) is disabled
1	CLK_SERCOM0_APB is enabled

Bit 0 – EVSYS EVSYS APB Clock Enable

Value	Description
0	The APB clock for the Event System peripheral (CLK_EVSYS_APB) is disabled
1	CLK_EVSYS_APB is enabled

12. OSCCTRL – Oscillators Controller

12.1. Features

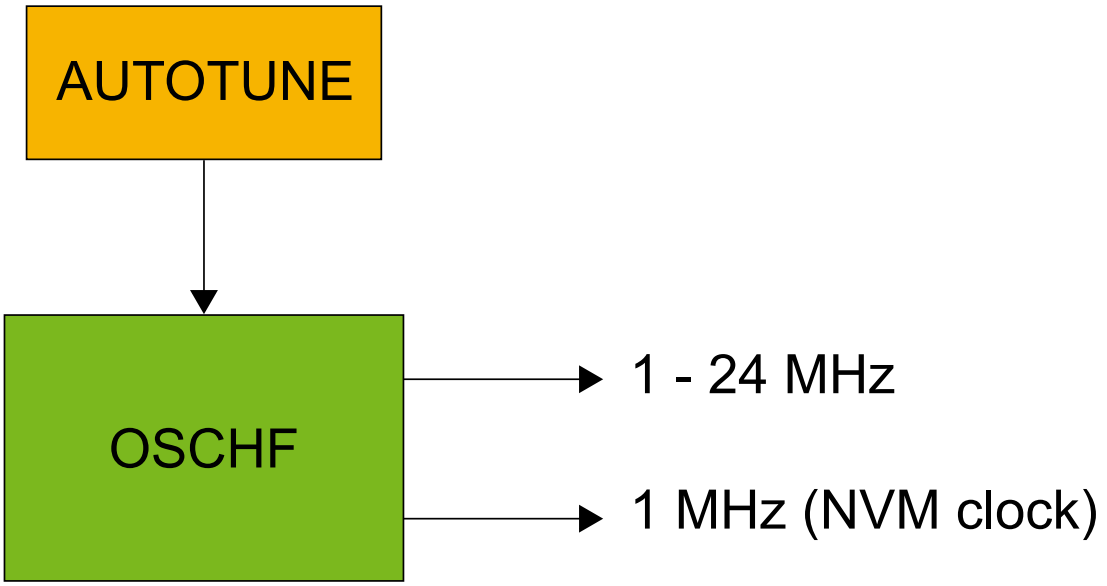
- Controls High-Speed Oscillator
  - Internal High-Frequency Oscillator (OSCHF) with 1–24 MHz clock output and 1 MHz output to NVMCTRL
  - Optional autonomous frequency tuning of OSCHF against a 32.768 kHz Crystal Oscillator (XOSC32K)

12.2. Overview

The Oscillators Controller (OSCCTRL) provides a user interface to the High Frequency Oscillator (OSCHF). Through this peripheral, the oscillator can be configured and tuned.

12.3. Block Diagram

Figure 12-1. OSCCTRL Block Diagram



12.3.1. Signal Description

Not applicable.

12.4. Functional Description

12.4.1. Initialization

Before enabling peripherals within the system, the system clock frequency must be configured by performing the following steps:

1. Write the value of the desired frequency to the Frequency Select bit field in the OSCHF Control register (OSCHF.FRQSEL).
2. Wait for the OSCHF is Ready bit (OSCHFRDY) in the Interrupt Flag Status and Clear register (INTFLAG) to be set, then clear the bit.

12.4.2. Operation

The Oscillators Controller (OSCCTRL) provides a user interface to the OSCHF. Through this peripheral, the oscillator can be configured and tuned. The Status register collects status signals

from the oscillator controlled by the OSCCTRL. These status signals can be used to generate a system interrupt, and in some cases, wake the system from sleep mode, provided the corresponding interrupt is enabled.

12.4.2.1. Internal High Frequency Oscillator (OSCHF) Operation

The OSCHF is an internal oscillator generating a frequency up to 24 MHz. The OSCHF frequency is selected by writing to the Frequency Select bit field in the OSCHFCTRL register (OSCHFCTRL.FRQSEL). The OSCHF is by default enabled and can only be disabled if the On-Demand Operation bit in the OSCHF Control register (OSCHFCTRL.ONDEMAND) is set and no peripheral is requesting the OSCHF clock.

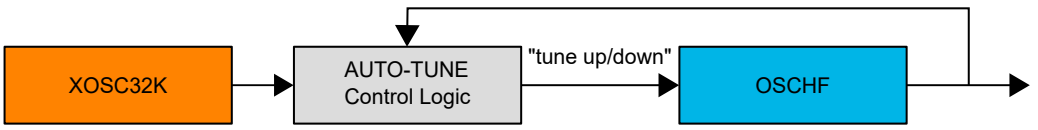
The OSCHF clock is output as soon as the oscillator is ready (STATUS.OSCHFRDY = 1). After reset, OSCHF serves as the default clock source at 4 MHz. The OSCHF can take as input an external 32.768 kHz oscillator to tune against to reduce drift.

12.4.3. Additional Features

12.4.3.1. Auto-Tune Operation

The auto-tune feature is used to improve the accuracy of the internal oscillator. It compares its internal 1 MHz clock and a 1.024 kHz reference from the external 32.768 kHz crystal oscillator. If an error is detected, it adjusts the frequency calibrations accordingly. There is an hysteresis built into the hardware that avoids the auto-tuning jumping between two tune settings. The auto-tune system tunes the oscillator to a maximum error that is comparable to the calibration step size of the oscillator. For the step size, refer to the *Electrical Characteristics* section.

Figure 12-2. OSCHFTUNE Auto-Tune Block Diagram



The auto-tune algorithm operates as follows:

1. Perform an auto-tune every millisecond. The algorithm will determine the change to OSCHFTUNE required to perfectly tune the oscillator, up to a maximum delta of 10.
2. If a change greater than 10 is needed to tune the oscillator, several auto-tune attempts will be needed.
3. When two auto-tune attempts have been made that resulted in no change in tuning, pause for 64 milliseconds before restarting from 1).

12.4.4. Sleep Mode Operations

Internal High Frequency Oscillator (OSCHF)

The OSCHF will behave differently in different sleep modes, based on the settings of OSCHFCTRL.ONDEMAND:

- The oscillator will always run when ONDEMAND is disabled
- The oscillator will only run if requested by a peripheral when ONDEMAND is enabled

12.4.5. Debug Operations

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

12.5. Dependencies

12.5.1. I/O Lines

Not applicable.

12.5.2. Power Management

The OSCCTRL will continue to operate in any sleep mode where the selected source clock is running. Refer to the Power Manager chapter for details on the different sleep modes.

12.5.3. Clocks

The OSCCTRL bus clock (CLK_OSCCTRL_APB) can be enabled and disabled in the Main Clock Controller and the default state can be found in the Peripheral Clock Masking section of the *MCLK - Main Clock Controller* chapter.

12.5.4. DMA

Not applicable.

12.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use OSCCTRL interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 12-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
SYSTEM	OSCHFRDY	OSCHF is ready	

12.5.6. Events

Not applicable.

12.5.7. Analog Connections

Not applicable.

12.6. Register Summary - OSCCTRL

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	INTENCLR	7:0				OSCHFRDY				
		15:8								
		23:16								
		31:24								
0x08	INTENSET	7:0				OSCHFRDY				
		15:8								
		23:16								
		31:24								
0x0C	INTFLAG	7:0				OSCHFRDY				
		15:8								
		23:16								
		31:24								
0x10	INTFLAGSET	7:0				OSCHFRDY				
		15:8								
		23:16								
		31:24								
0x14	STATUS	7:0				OSCHFRDY				
		15:8								
		23:16								
		31:24								
0x18 ... 0x23	Reserved									
0x24	OSCHFCTRL	7:0	ONDEMAND							
		15:8			FRQSEL[3:0]					AUTOTUNE
		23:16								
		31:24								
0x28	OSCHFTUNE	7:0	TUNE[7:0]							
		15:8								
		23:16								
		31:24								
0x2C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

12.6.1. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 4 – OSCHFRDY OSCHF Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the OSCHF Ready Interrupt Enable bit, thereby disabling the OSCHF is Ready interrupt.

Value	Description
0	The OSCHF Ready interrupt is disabled
1	The OSCHF Ready interrupt is enabled

12.6.2. Interrupt Enable Set

Name: INTENSET
Offset: 0x08
Reset: 0x00000000
Property: Local Write-Protection

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 (INTENCLR) register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 4 – OSCHFRDY OSCHF Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the OSCHF Ready Interrupt Enable bit, thereby enabling the OSCHF Ready interrupt.

Value	Description
0	The OSCHF Ready interrupt is disabled
1	The OSCHF Ready interrupt is enabled

12.6.3. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x0C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 4 – OSCHFRDY OSCHF is Ready
This flag is cleared by writing a '1' to it.
This flag is set on 0-to-1 transition of the corresponding flag in the Status register (STATUS.OSCHFRDY) and will generate an interrupt request if the OSCHFRDY bit in the Interrupt Enable Set/Clear register (INTENSET/CLR.OSCHFRDY) is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the OSCHFRDY interrupt flag.

12.6.4. Interrupt Flag Software Set

Name: INTFLAGSET
Offset: 0x10
Reset: 0x00000000
Property: Local Write-Protection

The Interrupt Flag Software Set (INTFLAGSET) register offers a method to set the INTFLAG register bits individually by software such that an associated Interrupt Software Routine (ISR) software routine can be tested. This feature is useful for safety applications.

Read value reflects the current state of the Interrupt Flag Status and Clear (INTFLAG) register.

Write any bit to '1' to set the corresponding interrupt flag in the INTFLAG register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 4 – OSCHFRDY OSCHF is Ready
This flag is cleared by writing a '1' to INTFLAG.OSCHFRDY.
This flag is set on 0-to-1 transition of the corresponding flag in the Status (STATUS) register and will generate an interrupt request if INTENSET/CLR.OSCHFRDY is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the OSCHFRDY bit in the INTFLAG register.

12.6.5. Status

Name: STATUS
Offset: 0x14
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 4 – OSCHFRDY OSCHF is Ready
This flag is cleared when the OSCHF oscillator is not ready.
This flag is set when the OSCHF oscillator is ready to be used as a clock source.

12.6.6. OSCHF Control

Name: OSCHFCTRL
Offset: 0x24
Reset: 0x00000C80
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			FRQSEL[3:0]					AUTOTUNE
Access			R/W	R/W	R/W	R/W		R/W
Reset			0	0	1	1		0

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

Bits 13:10 – FRQSEL[3:0] Frequency Select
This bit field controls the OSCHF frequency.

Value	Name	Description
0x00	1M	1 MHz output clock
0x01	2M	2 MHz output clock
0x02	3M	3 MHz output clock
0x03	4M	4 MHz output clock (default)
0x04	—	Reserved
0x05	8M	8 MHz output clock
0x06	12M	12 MHz output clock
0x07	16M	16 MHz output clock
0x08	20M	20 MHz output clock
0x09	24M	24 MHz output clock
0x0A - 0x0F	—	Reserved

Bit 8 – AUTOTUNE Automatic oscillator tune
Enable automatic tuning using 32.768 kHz from XOSC32K.

Value	Name	Description
0	OFF	Automatic oscillator frequency tune disabled
1	ON	Automatic oscillator frequency tune enabled

Bit 7 – ONDEMAND On-Demand Operation
The On-Demand Operation (ONDEMAND) bit allows the OSCHF to be enabled or disabled depending on peripheral clock requests. If ONDEMAND is set, the OSCHF will only be running when requested

by a peripheral. If there is no peripheral requesting the OSCHF clock source, the OSCHF will not be running.
For details, refer to the *Sleep Mode Operations* section.

Value	Name	Description
0	DISABLE	The OSCHF is always running
1	ENABLE	The OSCHF On-Demand Operation is enabled and the oscillator will be running when requested

12.6.7. OSCHF Oscillator Frequency Tune

Name: OSCHFTUNE
Offset: 0x28
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	TUNE[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 – TUNE[7:0] Oscillator Tune

Allows tuning the output frequency from the target frequency or towards the target due to the oscillator drift. The number is in 2's complement format. The nominal value is '0', indicating no correction from the factory-calibrated tuning.

When auto-tune is enabled, the OSCHFTUNE register is locked and will be updated with the latest auto-tune result only after auto-tune is disabled. If the user tries to write to the OSCHFTUNE register while auto-tune is enabled, a Peripheral Access Error is asserted (see the *PAC - Peripheral Access Controller* chapter for details). Writes to this register are allowed only if auto-tune is disabled (OSCHFCTRL.AUTOTUNE = '0'). Note that the OSCHFCTRL register is write protected (Local Write-Protection).

12.6.8. Write Protection Control

Name: WPCTRL
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write operation to be successful.

Value	Name	Description
0x4F5343	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	OSCCTRL register write protection is disabled
1	Write protection is enabled on OSCCTRL registers with the Local Write-Protection property. Non-debugger writes to OSCCTRL registers with Local Write-Protection are ignored and will generate a bus error.

13. OSC32KCTRL – 32.768 kHz Oscillators Controller

13.1. Features

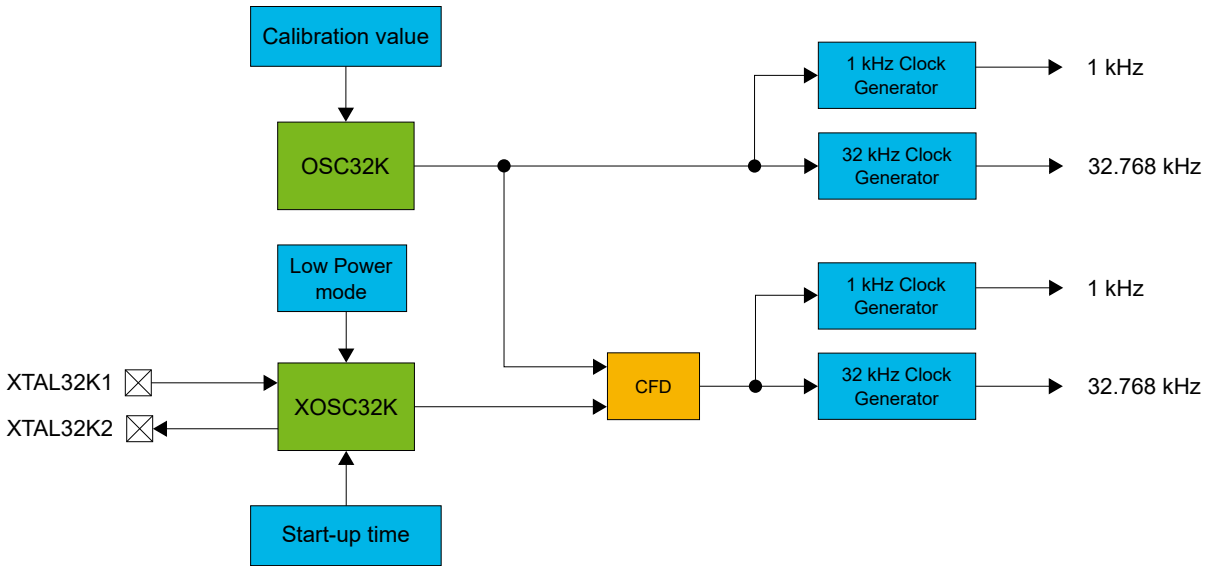
- Controls 32.768 kHz Oscillators
  - Internal 32.768 kHz oscillator (OSC32K)
  - External 32.768 kHz crystal oscillator (XOSC32K)
- Internal 32.768 kHz Oscillator (OSC32K)
  - Calibration values are loaded at reset
- External 32.768 kHz Crystal Oscillator (XOSC32K)
  - Selectable start-up time to optimize for various crystals
  - Clock source selectable from crystal or external clock
  - Optional clock failure detector
  - Optional autonomous frequency tuning of OSCHF against the 32.768 kHz crystal oscillator (XOSC32K)
  - Programmable drive strength to control power consumption and noise immunity

13.2. Overview

The 32.768 kHz Oscillators Controller (OSC32KCTRL) provides a user interface to the two 32.768 kHz oscillators, OSC32K (internal) and XOSC32K (external).

13.3. Block Diagram

Figure 13-1. OSC32KCTRL Block Diagram



13.3.1. Signal Description

Signal	Description	Type
XTAL32K1	Analog Input	32.768 kHz Crystal Oscillator input (XTAL mode)
	Digital Input	External Clock Generator input (EXTCLK mode)
XTAL32K2	Analog Output	32.768 kHz Crystal Oscillator output

Notes:

- The I/O lines are automatically selected when XOSC32K is enabled
- When a pin is configured to be used as a clock or oscillator, the pull-up on that pin will be overridden and set to the disabled state. This affects XTAL32K1 and XTAL32K2.
- The signal from the external crystal oscillator may affect the jitter of neighboring pads

13.4. Functional Description

13.4.1. Initialization

13.4.1.1. OSC32K Initialization

After a reset, OSC32K is configured by default to run if requested by any peripheral. If no peripheral is requesting OSC32K, the oscillator will be off.

OSC32K requires a start-up time to stabilize on the correct frequency. During the start-up time, the oscillator output is masked to ensure that no unstable clock propagates to the digital logic.

The oscillator can be forced to always run by writing the On-Demand Operation bit in the OSC32K Control register (OSC32KCTRL.ONDEMAND) to '0'.

A reset will revert this behavior so that the oscillator will only run if requested by a peripheral.

13.4.1.2. XOSC32K Initialization

After a reset, the XOSC32K is disabled.

Before any peripherals can request the XOSC32K oscillator, the oscillator must be started with the following steps:

1. Select the external source by writing the appropriate value to the External Source Select bit in the XOSC32K Control register (XOSC32KCTRL.XTALEN).
2. **Optional:** If the external source is set to External Crystal (XTAL), configure the appropriate start-up time by writing to the Start-Up Time bit field in the XOSC32KCTRL register (XOSC32KCTRL.CSUT) according to the external crystal specifications.
3. Enable the XOSC32K oscillator by writing the Enable bit in the XOSC32KCTRL register (XOSC32KCTRL.ENABLE) to '1'.

A crystal oscillator typically requires a long start-up time to stabilize on the correct frequency, depending on the external crystal specification. This start-up time is configured by changing the Oscillator Start-Up Time bit field in the XOSC32KCTRL register (XOSC32KCTRL.CSUT). During the start-up time, the oscillator output is masked to ensure that no unstable clock propagates to the digital logic.

The oscillator can be forced to always run by writing the ONDEMAND bit in the XOSC32KCTRL register (XOSC32KCTRL.ONDEMAND) to '0'. This will cause the oscillator to run regardless of whether it is requested by any peripheral.

13.4.2. Operation

The OSC32KCTRL oscillator(s) can be enabled, disabled, calibrated, and monitored through interface registers.

All oscillator statuses are collected in the Status register (STATUS). They can additionally trigger interrupts upon status changes through the Interrupt Enable Set (INTENSET), Interrupt Enable Clear (INTENCLR), and Interrupt Flag Status and Clear (INTFLAG) registers.

13.4.2.1. OSC32K - Internal 32.768 kHz Oscillator Operation

The OSC32K provides a tunable, low-speed, and low-power clock source. OSC32K runs whenever it is requested by any peripheral.

Once OSC32K is stable and ready to be used as a clock source, the OSC32K Ready bit in the STATUS register (STATUS.OSC32KRDY) is set to '1'. An interrupt is generated on the OSC32RDY '0' to '1' transition if the OSC32K Ready bit in the Interrupt Enable Set register (INTENSET.OSC32KRDY) is set to '1'.

It is possible to lock the OSC32K configuration by setting the Write Lock bit in the OSC32KCTRL register (OSC32KCTRL.WRTLOCK) to '1'. The OSC32K configuration is now locked until a Device Reset is detected.

OSC32K can be used as a source for the Real-Time Counter (RTC). A clock request for the OSC32K is automatically generated. Before enabling the RTC peripheral, the corresponding oscillator output must be selected in the RTC Clock Source Selection bit field in the RTC Control register (RTCCTRL.RTCSEL). Also, the OSC32K oscillator must be configured in the OSC32KCTRL register in order to ensure proper operation. For details on RTC clock configuration, refer to the *Real-Time Counter Clock Selection* section.

OSC32K can also be used as a source for Generic Clock Generators (GCLK), the Watchdog Timer (WDT), or for other peripherals through a generic clock request interface implemented in hardware. Requests over this interface will start the oscillator if it is not already running. Before enabling the peripheral that will generate a clock request, the oscillator must be configured in the OSC32KCTRL register to ensure proper operation.

13.4.2.2. XOSC32K - External 32.768 kHz Crystal Oscillator Operation

The XOSC32K operates in two different modes:

- External clock mode, with an external clock signal connected to the XTAL32K1 pin
- Crystal oscillator mode, with an external 32.768 kHz crystal connected between the XTAL32K1 and XTAL32K2 pins

XOSC32K is enabled by setting the ENABLE bit in the XOSC32KCTRL register (XOSC32KCTRL.ENABLE) to '1'. XOSC32K is disabled by clearing the Enable bit. When disabling XOSC32K, it is important to write the Enable bit to '0' before additional changes are made to the XOSC32K register.

To enable XOSC32K as an external crystal oscillator, the XTALEN bit in the XOSC32KCTRL register (XOSC32KCTRL.XTALEN) must be set to '1'. If XTALEN is '0', XOSC32K is configured for external clock mode and the external clock input signal on XTAL32K1 will be enabled.

The XOSC32K is disabled after any reset.

It is possible to lock the XOSC32K configuration by setting the Write Lock bit in the OSC32KCTRL register (OSC32KCTRL.WRTLOCK) to '1'. The XOSC32K configuration will remain locked until a Device Reset occurs.

The XTAL32K1 and XTAL32K2 pins can be used as General Purpose Input/Output (GPIO) pins or by other peripherals in the system. When XOSC32K is enabled, the operating mode determines the pin configuration. In External Crystal (XTAL) mode, the XTAL23K1 and XTAL32K2 pins are controlled by OSC32KCTRL, and the GPIO functions are overridden on both pins. In External Clock (EXTCLK) mode, only the XTAL32K1 pin is overridden and controlled by OSC32KCTRL, while the XTAL32K2 pin may be used as a GPIO pin.

XOSC32K can be used as a source for the RTC. A clock request for XOSC32K is automatically generated when the RTC peripheral is enabled. Before enabling the RTC peripheral, the corresponding oscillator output must be selected in the RTC Clock Source Selection bit field in the RTC Control register (RTCCTRL.RTCSEL). Also, the XOSC32K oscillator must be configured in the XOSC32KCTRL register in order to ensure proper operation. For details on RTC clock configuration, refer to the *Real-Time Counter Clock Selection* section.

XOSC32K can be used as a source for the GCLK or for other modules through a generic clock request interface implemented in hardware. Requests on this interface will start the oscillator if it

is not already running. Before enabling the module that will generate a clock request, the oscillator must be configured and enabled (XOSC32KCTRL) in order to ensure proper operation.

13.4.2.3. Watchdog Timer Clock Selection

The WDT uses the internal 1.024 kHz OSC32K output clock. This clock is automatically requested by the WDT peripheral.

13.4.2.4. Real-Time Counter (RTC) Clock Selection

Before enabling the RTC peripheral, the RTC clock must be selected by writing the RTCSEL bit field in the RTCCTRL register.

To ensure proper operation, disable the RTC peripheral before changing the RTC clock source. See the *RTC - Real-Time Counter* chapter for additional information.

13.4.2.5. Supply Controller (SUPC) Sampled Brown-Out Detector (BOD) Clock Selection

The Supply Controller (SUPC) uses the internal 32.768 kHz OSC32K output clock for sampled Brown-Out Detector (BOD) operation. This clock is automatically requested by the SUPC peripheral.

The BOD is integrated into the SUPC peripheral. See the *SUPC - Supply Controller* chapter for additional information.

13.4.2.6. External Interrupt Controller (EIC) Clock Selection

The External Interrupt Controller (EIC) can be configured to use the OSC32K output clock. This clock is automatically requested by the EIC peripheral.

13.4.3. Additional Features

13.4.3.1. CFD - Clock Failure Detector

The Clock Failure Detector (CFD) allows the user to monitor the external clock or crystal oscillator signal provided by the XOSC32K. The CFD detects failing operation of the XOSC32K clock with reduced latency, and supports switching to a safe clock source in case of the clock failure. The user can also switch from the safe clock back to XOSC32K in case of recovery. The safe clock is derived from the OSC32K oscillator with a configurable prescaler. This allows configuring the safe clock in order to fulfill the operative conditions of the microcontroller.

In sleep modes, the CFD operation is automatically disabled when the external oscillator is not requested to run by a peripheral. Refer to the *Sleep Mode Operation* section for additional information.

The user interface registers allow the CFD to be enabled, disabled, and configured. The STATUS register provides status flags for failure and clock switch conditions. The CFD can optionally trigger an interrupt or an event when a failure is detected.

13.4.3.1.1. Clock Failure Detection

The CFD is disabled at reset. The CFD does not monitor the XOSC32K clock when:

1. The XOSC32K oscillator is disabled (the ENABLE bit in the XOSC32KCTRL register is '0').
2. The XOSC32K oscillator is not running (the XOSC32KRDY bit in the STATUS register is '0'), that is, when not in sleep and the ONDEMAND bit in the XOSC32KCTRL register (XOSC32KCTRL.ONDEMAND) is '1' and no peripherals are requesting the XOSC32K oscillator.

Enabling the CFD will automatically request the safe (monitor) clock, such as OSC32K.

CFD operation is started by setting the CFD Enable bit in the CFD Control register (CFDCTRL.CFDEN) to '1'. After starting or restarting XOSC32K, the CFD does not detect failure until the start-up time has elapsed. The start-up time is configured by the CSUT bit field in the XOSC32KCTRL register. Once the XOSC32K start-up time is elapsed, the XOSC32K clock is constantly monitored.

Note: The ready timer for XOSC32K uses a CSUT-controlled count of XOSC32K cycles before the XOSC32KRDY signal is asserted. Since the CFD does not detect a failure until the start-up time has elapsed, a total XOSC32K failure (i.e., the oscillator never starts) will not be detected by the CFD. The application must handle this case by waiting for the XOSC32KRDY after enabling XOSC32K to know that the oscillator has indeed started. Enabling the CFD detector alone, without waiting for the XOSC32KRDY, is not sufficient.

During a period of four safe clocks (monitor period), the CFD watches for a clock activity from the XOSC32K. There must be at least one rising and one falling XOSC32K clock edge during four safe clock periods to meet non-failure conditions. If no or insufficient activity is detected, the failure status is asserted: the Clock Failure Detector status bit (CLKFAIL) in the STATUS register and the Clock Failure Detector interrupt flag bit (CLKFAIL) in the INTFLAG register are both set to '1'. If the XOSC32K Clock Failure Detected bit (CLKFAIL) in the INTENSET register is set, an interrupt is generated. If the Event Output enable bit (CFDEO) in the Event Control register (EVCTRL) is set, an output event is generated.

After a clock failure is detected, monitoring of the XOSC32K clock continues, and the Clock Failure Detector status bit (CLKFAIL) in the STATUS register reflects the current XOSC32K activity. If the CFD is disabled (CFDEN bit in the CFDCTRL register is '0'), XOSC32K must be disabled (ENABLE bit in the XOSC32KCTRL register is '0') before re-enabling the CFD. Not following this guideline can lead to a false clock failure detection. When the XOSC32K Clock Failure Detector is enabled (CFDEN bit in the CFDCTRL register is '1') and a failure is detected (CLKFAIL bit in the STATUS register is '1'), the XOSC32KRDY bit in the STATUS register is not cleared. When checking the XOSC32K ready status, the state of the XOSC32K clock failure status (CLKFAIL bit in the STATUS register) must be checked before the XOSC32KRDY bit, and the XOSC32KRDY status bit should be disregarded if the XOSC32K clock failure status is set (CLKFAIL bit in the STATUS register is '1').

13.4.3.1.2. Clock Switch

When a clock failure is detected, the XOSC32K clock is replaced by the safe clock in order to maintain an active clock during the XOSC32K clock failure. The safe clock source is the OSC32K oscillator clock. Both 32.768 kHz and 1.024 kHz outputs of the XOSC32K are replaced by the respective 32.768 kHz and 1.024 kHz OSC32K outputs. The safe clock source can be scaled down by a configurable prescaler to ensure that the safe clock frequency does not exceed the operating conditions selected by the application. When the XOSC32K clock is switched to the safe clock, the XOSC32K Clock Switch bit (CLKSW) in the STATUS register is set by hardware.

When the CFD has switched to the safe clock, XOSC32K is not disabled. If desired, the application must take the necessary actions to disable the oscillator. The application must also take the necessary actions to configure the system clocks to continue normal operations. If the application can recover XOSC32K, it can switch back to the XOSC32K clock by setting the Switch Back Enable bit (SWBACK) in the CFDCTRL register to '1' . Once the XOSC32K clock is switched back, the SWBACK bit in the CFDCTRL register is cleared by hardware.

13.4.3.1.3. Prescaler

The CFD has a prescaler to generate the safe clock from the OSC32K oscillator. The prescaler setting is controlled by the Clock Failure Detector Prescaler bit (CFDPRESC) in the CFDCTRL register. The prescaler allows the OSC32K oscillator to be scaled down to ensure that the safe clock frequency is not higher than the XOSC32K clock frequency monitored by the CFD.

The prescaler settings are either no prescaling (OSC32K) or divided by 2 (OSC32K/2). The prescaler is applied to both outputs (32.768 kHz and 1.024 kHz) of the safe clock.

Example 13-1. CFD Prescaler Example

For an external crystal oscillator at 32.768 kHz and an OSC32K frequency of 32.768 kHz, the CFDPRESC bit in the CFDCTRL register should be set to '0' to provide a safe clock of equal frequency.

13.4.3.1.4. Events

If the CFDEO bit in the EVCTRL register is written to '1', the Clock Failure Detector event output will be generated if the CFD detects a clock failure.

If a clock failure is detected when the CFD is already switched to the safe clock, a CFD event will not be generated.

13.4.3.1.5. Sleep Mode

The CFD is halted depending on configuration of the XOSC32K and the peripheral clock requests. The XOSC32K CLKFAIL interrupt can be used to wake up the device from sleep modes.

Refer to the *Sleep Mode Operation* section for additional information.

13.4.4. Sleep Mode Operation

OSC32K Internal 32.768 kHz Oscillator

The OSC32K oscillator's sleep mode operation is determined by the setting of the ONDEMAND bit in the OSC32KCTRL register.

After a reset, the ONDEMAND bit in the OSC32KCTRL register is automatically set and the OSC32K oscillator is only running if requested by a peripheral. The oscillator will run in any sleep mode as long as it is requested by a peripheral.

If the ONDEMAND bit is written to '0', the oscillator will always run in any sleep mode.

Table 13-1. OSC32K Sleep Behavior

Sleep Mode	OSC32KCTRL.ONDEMAND	Sleep Behavior
Active, Idle, Standby	0	Always running
	1	Running if requested by a peripheral

XOSC32K External 32.768 kHz Oscillator

The XOSC32K oscillator's sleep mode operation is determined by the setting of the ONDEMAND bit in the XOSC32KCTRL register.

After a reset, the XOSC32K oscillator is not enabled and the ONDEMAND bit is set. In order for the oscillator to run, it must first be configured and enabled, and a peripheral must request the oscillator. In this case, the oscillator will run in any sleep mode as long as any peripheral is requesting it.

If the ONDEMAND bit is written to '0', the oscillator will always run as long as it is enabled.

The CFD is halted depending on the configuration of the XOSC32K and the peripheral clock requests. The XOSC32K CLKFAIL interrupt can be used to wake up the device from a sleep mode.

Table 13-2. XOSC32K Sleep Behavior

Sleep Mode	XOSC32KCTRL. ONDEMAND	XOSC32K and CFD Sleep Behavior
Active, Idle, Standby	0	Always running
	1	Running if requested by a peripheral

Note:
If the ENABLE bit in the XOSC32KCTRL register is '0', the XOSC32K oscillator is unavailable.

If the ENABLE bit in the XOSC32KCTRL register is '1', this table is valid.

13.4.5. Debug Operations

When the CPU is halted in Debug mode, OSC32KCTRL continues normal operation. If OSC32KCTRL is configured in a way that requires it periodic servicing by the CPU through interrupts or similar mechanisms, improper operation or data loss may occur during debugging.

13.5. Dependencies

13.5.1. I/O Lines

The XTAL32K1 and XTAL32K2 pins are automatically configured when the XOSC32K oscillator is enabled. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

13.5.2. Power Management

The OSC32KCTRL will continue to operate in any sleep mode if requested by any peripheral. The OSC32KCTRL'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 the *PM - Power Manager* chapter for additional information on the different sleep modes.

13.5.3. Clocks

The OSC32KCTRL bus clock (CLK_OSC32KCTRL_APB) can be enabled and disabled in the Main Clock Controller (MCLK), and its default state can be found in the *Peripheral Clock Masking* section of the *MCLK - Main Clock Controller* chapter.

13.5.4. DMA

Not applicable.

13.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use OSC32KCTRL interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 13-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
SYSTEM	XOSC32KRDY	XOSC32K is ready	
SYSTEM	CLKFAIL	XOSC32K has failed	
SYSTEM	OSC32KRDY	OSC32K is ready	

13.5.6. Events

Event Generators

The OSC32KCTRL can generate the following events:

Table 13-4. OSC32KCTRL Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
OSC32KCTRL	CLKFAIL	XOSC32K Failure	Level	OSC32K	As long as flag is set

The Clock Failure Detector (CLKFAIL) Event is generated when the Clock Failure Detector (CFD) status bit is set in the Status (STATUS) register. The CFD event is not generated when the Clock Switch (CLKSW) bit in the STATUS register is set (CFD is switched to the safe clock).

Writing a '1' to an Event Output bit in the Event Control Register (EVCTRL.xxEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

Refer to the *EVSYS - Event System* chapter for additional information on configuring the event system.

Event Users

This peripheral has no event user(s).

13.5.7. Analog Connections

The external 32.768 kHz crystal must be connected between the XTAL32K1 and XTAL32K2 pins, along with any required load capacitors.

13.6. Register Summary - OSC32KCTRL

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	INTENCLR	7:0						CLKFAIL	OSC32KRDY	XOSC32KRDY
		15:8								
		23:16								
		31:24								
0x08	INTENSET	7:0						CLKFAIL	OSC32KRDY	XOSC32KRDY
		15:8								
		23:16								
		31:24								
0x0C	INTFLAG	7:0						CLKFAIL	OSC32KRDY	XOSC32KRDY
		15:8								
		23:16								
		31:24								
0x10	INTFLAGSET	7:0						CLKFAIL	OSC32KRDY	XOSC32KRDY
		15:8								
		23:16								
		31:24								
0x14	STATUS	7:0					CLKSW	CLKFAIL	OSC32KRDY	XOSC32KRDY
		15:8								
		23:16								
		31:24								
0x18	RTCCTRL	7:0						RTCSEL[2:0]		
		15:8								
		23:16								
		31:24								
0x1C	XOSC32KCTRL	7:0	ONDEMAND					XTALEN	ENABLE	
		15:8			LPMODE				CSUT[1:0]	
		23:16								
		31:24	WRTLOCK							
0x20	CFDCTRL	7:0						CFDPRESC	SWBACK	CFDEN
		15:8								
		23:16								
		31:24								
0x24	EVCTRL	7:0								CFDEO
		15:8								
		23:16								
		31:24								
0x28	OSC32KCTRL	7:0	ONDEMAND							
		15:8								
		23:16								
		31:24	WRTLOCK							
0x2C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

13.6.1. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection

This register allows the user to disable an interrupt without doing a Read-Modify-Write (RMW) operation.

Changes in this register will also be reflected in the INTENSET register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						CLKFAIL	OSC32KRDY	XOSC32KRDY
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – CLKFAIL XOSC32K Clock Failure Detection Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the XOSC32K Clock Failure Detection Interrupt Enable bit, which disables the XOSC32K Clock Failure Detected interrupt.

Bit 1 – OSC32KRDY OSC32K Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the OSC32K Ready Interrupt Enable bit, which disables the OSC32K Ready interrupt.

Bit 0 – XOSC32KRDY XOSC32K Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the XOSC32K Ready Interrupt Enable bit, which disables the XOSC32K Ready interrupt.

13.6.2. Interrupt Enable Set

Name: INTENSET
Offset: 0x08
Reset: 0x00000000
Property: Local Write-Protection

This register allows the user to disable an interrupt without performing a Read-Modify-Write (RMW) operation. Changes in this register will also be reflected in the INTENCLR register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						CLKFAIL	OSC32KRDY	XOSC32KRDY
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – CLKFAIL XOSC32K Clock Failure Detection Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the XOSC32K Clock Failure Detection Interrupt Enable bit, which enables the XOSC32K Clock Failure Detected interrupt.

Value	Description
0	The XOSC32K Clock Failure Detection interrupt is disabled
1	The XOSC32K Clock Failure Detection interrupt is enabled

Bit 1 – OSC32KRDY OSC32K Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the OSC32K Ready Interrupt Enable bit, which enables the OSC32K Ready interrupt.

Value	Description
0	The OSC32K Ready interrupt is disabled
1	The OSC32K Ready interrupt is enabled

Bit 0 – XOSC32KRDY XOSC32K Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the XOSC32K Ready Interrupt Enable bit, which enables the XOSC32K Ready interrupt.

Value	Description
0	The XOSC32K Ready interrupt is disabled

Value	Description
1	The XOSC32K Ready interrupt is enabled

13.6.3. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x0C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						CLKFAIL	OSC32KRDY	XOSC32KRDY
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – CLKFAIL XOSC32K Clock Failure Detected
This flag is cleared by writing a '1' to it.
This flag is set on a '0'-to-'1' transition of the corresponding flag in the STATUS register and will generate an interrupt request if the INTENSET/CLR.CLKFAIL bit is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the XOSC32K Clock Failure Detected interrupt flag.

Bit 1 – OSC32KRDY OSC32K Ready
This flag is cleared by writing a '1' to it.
This flag is set on a '0'-to-'1' transition of the corresponding flag in the STATUS register and will generate an interrupt request if the INTENSET/CLR.OSC32KRDY bit is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the OSC32K Ready interrupt flag.

Bit 0 – XOSC32KRDY XOSC32K Ready
This flag is cleared by writing a '1' to it.
This flag is set on a '0'-to-'1' transition of the corresponding flag in the STATUS register and will generate an interrupt request if the INTENSET/CLR.XOSC32KRDY bit is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the XOSC32K Ready interrupt flag.

13.6.4. Interrupt Flag Software Set

Name: INTFLAGSET
Offset: 0x10
Reset: 0x00000000
Property: Local Write-Protection

The Interrupt Flag Software Set register (INTFLAGSET) offers a way to set the INTFLAG register bits individually by software. This allows for testing of any associated Interrupt Software Routine (ISR). This feature may be useful for safety applications.

The read value reflects the current state of the INTFLAG register.

Write any bit to '1' to set the corresponding interrupt flag in the INTFLAG register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						CLKFAIL	OSC32KRDY	XOSC32KRDY
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – CLKFAIL XOSC32K Clock Failure Detected
This flag is cleared by writing a '1' to INTFLAG.CLKFAIL.
This flag is set on a '0'-to-'1' transition of the corresponding flag in the STATUS register and will generate an interrupt request if the INTENSET/INTENCLR.CLKFAIL bit is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to set the corresponding bit in the INTFLAG register.

Bit 1 – OSC32KRDY OSC32K Ready
This flag is cleared by writing a '1' to INTFLAG.OSC32KRDY.
This flag is set on a '0'-to-'1' transition of the corresponding flag in the STATUS register and will generate an interrupt request if the INTENSET/INTENCLR.OSC32KRDY bit is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to set the corresponding bit in the INTFLAG register.

Bit 0 – XOSC32KRDY XOSC32K Ready
This flag is cleared by writing a '1' to INTFLAG.XOSC32KRDY.
This flag is set on a '0'-to-'1' transition of the corresponding flag in the STATUS register and will generate an interrupt request if the INTENSET/INTENCLR.XOSC32KRDY bit is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to set the corresponding bit in the INTFLAG register.

13.6.5. Status

Name: STATUS
Offset: 0x14
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CLKSW	CLKFAIL	OSC32KRDY	XOSC32KRDY
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – CLKSW XOSC32K Clock Switch
This flag is cleared when XOSC32K is not switched and provides the external clock or crystal oscillator clock.
This flag is set when an oscillator failure is detected and XOSC32K is switched to provide the safe clock.

Bit 2 – CLKFAIL XOSC32K Clock Failure Detected
This flag is cleared when no failure of the XOSC32K external clock or crystal oscillator has been detected.
This flag is set when a failure of the XOSC32K external clock or crystal oscillator has been detected.

Bit 1 – OSC32KRDY OSC32K Ready
This flag is cleared when the OSC32K oscillator is not ready.
This flag is set when the OSC32K oscillator is ready to be used as a clock source.

Bit 0 – XOSC32KRDY XOSC32K Ready
This flag is cleared when the XOSC32K external clock or crystal oscillator is not stable and not ready to use.
This flag is set when the XOSC32K external clock or crystal oscillator is stable and ready to use.

13.6.6. RTC Control

Name: RTCCTRL
Offset: 0x18
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bits 2:0 – RTCSEL[2:0] RTC Clock Source Selection

Value	Name	Description
0x0	—	Reserved
0x1	—	Reserved
0x2	OSC1K	1.024 kHz from the internal 32.768 kHz oscillator
0x3	OSC32K	32.768 kHz from the internal 32.768 kHz oscillator
0x4	XOSC1K	1.024 kHz from the external 32.768 kHz crystal oscillator
0x5	XOSC32K	32.768 kHz from the external 32.768 kHz crystal oscillator
0x6	—	Reserved
0x7	—	Reserved

13.6.7. XOSC32K Control

Name: XOSC32KCTRL
Offset: 0x1C
Reset: 0x00000080
Property: Local Write-Protection

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

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			LPMODE				CSUT[1:0]	
Access			R/W				R/W	R/W
Reset			0				0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND					XTALEN	ENABLE	
Access	R/W					R/W	R/W	
Reset	1					0	0	

Bit 31 – WRTLOCK Write Lock
This bit controls whether the XOSC32K Control register (XOSC32KCTRL) is locked for future writes.
When the Write Lock (XOSC32KCTRL.WRTLOCK) bit is '1', the configuration for the XOSC32K oscillator is locked and can only be unlocked by a Device Reset.
Writes to this register, when Local Write-Protection is enabled, will be discarded and return a bus error.
A write access from a debugger will disregard the status of WRTLOCK. Even if the XOSC32KCTRL.WRTLOCK bit is '1', this will not prevent a debugger write to the XOSC32KCTRL register.
A Device Reset will clear this bit to the unlocked state.

Value	Description
0	The XOSC32K configuration is not locked
1	The XOSC32K configuration is locked

Bit 13 – LPMODE Low Power Mode
This bit selects whether the 32.768 kHz crystal oscillator is set to Low Power mode.

Value	Description
0	Low Power off
1	Low Power on

Bits 9:8 – CSUT[1:0] Crystal Oscillator Start-Up Time
This bit field selects the start-up time for the crystal oscillator.
If an external clock is selected in the External Source Select bit (XTALEN = '0'), the start-up time will not be applied.

Value	Name	Description
0x0	1K	1K XOSC32K cycles
0x1	16K	16K XOSC32K cycles
0x2	32K	32K XOSC32K cycles
0x3	64K	64K XOSC32K cycles

Bit 7 – ONDEMAND On-Demand Operation

Writing a '0' to this bit will cause the XOSC32K oscillator to always run when the Oscillator Enable bit (XOSC32KCTRL.ENABLE) is '1'.
Writing a '1' to this bit will cause the XOSC32K oscillator to run only if the XOSC32KCTRL.ENABLE bit is '1' and it is requested by a peripheral. If there no peripherals are requesting the XOSC32K clock source, the XOSC32K oscillator will be in a disabled state.
Refer to the *Sleep Mode Operation* section for additional information.

Value	Name	Description
0	DISABLE	The XOSC32K On-Demand Operation is disabled
1	ENABLE	The XOSC32K On-Demand Operation is enabled

Bit 2 – XTALEN External Source Select

This bit selects the external clock source type.

Value	Name	Description
0	EXTCLK	External Clock on the XTAL32K1 pin. XTAL32K2 is available for other functions. The CSUT timer is disregarded, and no start-up time is used.
1	XTAL	External Crystal connected to the XTAL32K1 and XTAL32K2 pins

Bit 1 – ENABLE Oscillator Enable

Writing a '0' to this bit disables the XOSC32K oscillator, and the respective input pins XTAL32K1 and XTAL32K2 may be configured as I/O pins.
Note: Due to synchronization, to safely disable an enabled XOSC32K oscillator, write the ENABLE bit (XOSC32KCTRL.ENABLE) to '0' without altering any other bits. Then, wait until the ENABLE bit has been synchronized by polling the XOSC32K Ready bit in the STATUS register (STATUS.XOSC32KRDY) until it is '0'. After the synchronization is completed, all bits in this register can be freely written to any value.

Writing a '1' to this bit overrides the configuration of the respective input pins to XTAL32K1 and, depending on the XTALEN configuration, potentially XTAL32K2. If the XOSC32KCTRL.ONDEMAND bit is '1', the oscillator is not started unless it is requested as a clock source. If the XOSC32KCTRL.ONDEMAND bit is '0', the oscillator is always running.

Value	Name	Description
0	DISABLED	XOSC32K oscillator is disabled
1	ENABLED	XOSC32K oscillator is enabled

13.6.8. Clock Failure Detector Control

Name: CFDCTRL
Offset: 0x20
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 2 – CFDPRESC Clock Failure Detector Prescaler
This bit selects the prescaler for the clock failure detector.

Value	Description
0	The CFD safe clock frequency is equal to the OSC32K frequency
1	The CFD safe clock frequency is equal to the OSC32K frequency divided by 2

Bit 1 – SWBACK Clock Switch Back Enable
This bit controls the XOSC32K output switch back to the external clock or crystal oscillator in case of clock recovery. This bit cannot be set to '1' until the Clock Failure Detector Enable bit (CFDCTRL.CFDEN) is set to '1'.
This bit is reset once the XOSC32K clock is switched back to the external clock or crystal oscillator.

Value	Name	Description
0	DISABLE	The clock switch back is disabled
1	ENABLE	The clock switch back is enabled

Bit 0 – CFDEN Clock Failure Detector Enable
This bit controls the XOSC32K clock failure detector.
Refer to the *Clock Failure Detection Operation* section for additional information.

Value	Name	Description
0	DISABLE	Clock failure detection is disabled
1	ENABLE	Clock failure detection is enabled

13.6.9. Event Control

Name: EVCTRL
Offset: 0x24
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 0 – CFDEO Clock Failure Detector Event Output Enable
This bit configures whether the CFD event output is enabled. An output event will be generated when the CFD detects a clock failure.

Value	Description
0	CFD event output is disabled, and no event will be generated
1	CFD event output is enabled, and an event will be generated

13.6.10. 32.768 kHz Internal Oscillator (OSC32K) Control

Name: OSC32KCTRL
Offset: 0x28
Reset: 0x00000080
Property: Local Write-Protection

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

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 31 – WRTLOCK Write Lock

This bit controls whether the OSC32K Control register (OSC32KCTRL) is locked for future writes.
When the Write Lock bit (OSC32KCTRL.WRTLOCK) is '1', the configuration for the OSC32K oscillator is locked and can only be unlocked by a Device Reset.
Writes to this register, when Local Write-Protection is enabled, will be discarded and return a bus error.
A write access from a debugger will disregard the status of WRTLOCK. Even if the OSC32KCTRL.WRTLOCK bit is '1', this will not prevent a debugger write to the OSC32KCTRL register.
A Device Reset reset will clear this bit to the unlocked state.

Value	Description
0	The OSC32K configuration is not locked
1	The OSC32K configuration is locked

Bit 7 – ONDEMAND On-Demand Operation

Writing a '0' to this bit will cause the OSC32K oscillator to always run when the Oscillator Enable bit (OSC32KCTRL.ENABLE) is '1'.
Writing a '1' to this bit will cause the OSC32K oscillator run only if the OSC32KCTRL.ENABLE bit is '1' and it is requested by a peripheral. If no peripherals are requesting the OSC32K clock source, the OSC32K oscillator will be in a disabled state.
Refer to the *Sleep Mode Operation* section for additional information.

Value	Name	Description
0	DISABLE	The OSC32K On-Demand Operation is disabled
1	ENABLE	The OSC32K On-Demand Operation is enabled

13.6.11. Write Protection Control

Name: WPCTRL
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write to be successful.

Value	Name	Description
0x4F5343	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	Write protection is disabled for the OSC32KCTRL registers
1	Write protection is enabled for the OSC32KCTRL registers with the Local Write-Protection property. Non-debugger writes to the OSC32KCTRL registers with Local Write-Protection are ignored and generate a bus error.

14. PM – Power Manager

14.1. Features

- Power Management for Adjusting Power Consumption and Functions
- Controls Sleep Modes
  - Idle sleep mode
  - Standby sleep mode

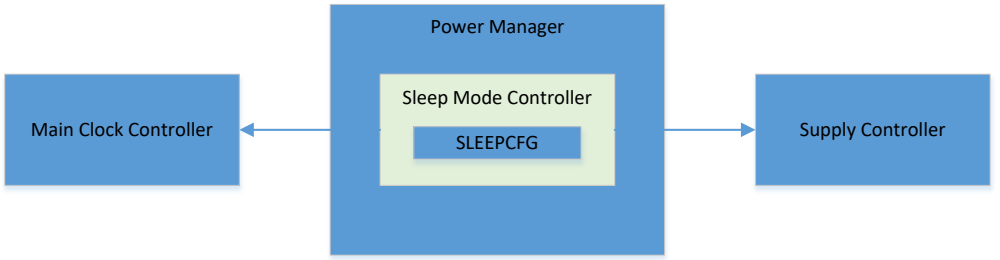
14.2. Overview

The Power Manager (PM) manages the device's sleep modes.

Various sleep modes are available to meet different power consumption needs. In Active mode, the CPU runs the application code. When the device enters a sleep mode, program execution stops, and the PM automatically disables certain peripherals and clock domains depending on the sleep mode. The application code determines which sleep mode to enter and when. Interrupts from enabled peripherals or reset sources can wake the device from sleep mode and return it to Active mode.

14.3. Block Diagram

Figure 14-1. PM Block Diagram



14.3.1. Signal Description

Not applicable.

14.4. Functional Description

14.4.1. Initialization

After a Power-On Reset (POR), the PM is enabled and the device starts in Active mode.

14.4.2. Operation

In Active mode, all clock domains and power domains are active, allowing both software execution and peripheral operation. The PM saves power by controlling different sleep modes based on application requirements. Refer to the Sleep Mode Operation section for more details on the Idle and Standby sleep modes.

In both Active mode or any sleep mode, if an AHB/APB clock is masked in MCLK.AHBMASK or MCLK.APBxMASK registers, the clock is gated at the output of the Main Clock (MCLK) and not provided to the corresponding peripheral, regardless of whether the peripheral requests it.

The Supply Controller (SUPC) allows power consumption in the Standby sleep mode to be further reduced.

14.4.2.1. Enabling, Disabling and Resetting

The PM is always enabled and cannot be reset.

14.4.2.2. Sleep Modes

The device can enter a sleep mode, during which the CPU is stopped. Peripherals may remain active or become idle, depending on the depth of the selected sleep mode.

- Idle sleep mode: The CPU is stopped. Synchronous clocks are halted except when requested. The state of the logic and the contents of the SRAM are retained.
- Standby sleep mode: The CPU and peripherals are stopped. The logic state and the contents of the SRAM are retained.

Refer to Sleep Mode Operation for more details on Idle and Standby sleep modes.

The PM is always active.

14.4.3. Additional Features

14.4.3.1. Regulator Automatic Low-Power Mode

In Standby sleep mode, the PM selects either the Low Voltage Regulator (LDO) or the Ultra-Low Power (ULP) voltage regulator to supply the VDDCORE. By default, the ULP voltage regulator is used.

If a sleepwalking peripheral requests a clock source other than a slow clock (e.g., 32 kHz), the LDO regulator is automatically enabled during the sleepwalking operation. This behavior can be changed by writing the SUPC.VREG.RUNSTDBY bit. Refer to the following table for details.

Table 14-1. Regulator State in Sleep Mode

Sleep Mode	SUPC.VREG.RUNSTDBY	SleepWalking	Regulator Used for VDDCORE
Active	-	-	ULP/LDO(1)
Idle	-	-	ULP/LDO(1)
Standby	0x0: Auto	No	ULP
		Yes	ULP/LDO(2)
	0x1: Performance	-	LDO

Notes:

1. If the CPU or a peripheral requests a “fast” clock, the LDO is automatically enabled. In Active mode, the LDO is also used if SUPC.VREG.RUNSTDBY is set to Performance. Otherwise, the ULP is used.
2. During sleepwalk, if a peripheral requests a “fast” clock, the LDO is automatically enabled for the duration of the sleepwalk. Otherwise, the ULP is used.

14.4.4. Sleep Mode Operation

A sleep mode is entered by executing the Wait For Interrupt (WFI) instruction. The Sleep Mode bit field in the Sleep Configuration (SLEEPCFG.SLEEPMODE) register selects the level of the sleep mode.

Note: There is a small latency between the store instruction and the actual writing of the SLEEPCFG register due to bus bridges. Software must verify that the SLEEPCFG register contains the desired value before executing the WFI instruction.

Table 14-2. Sleep Mode Entry and Exit

Mode	Mode Entry	Wake-Up Sources
IDLE	SLEEPCFG.SLEEPMODE = IDLE	Synchronous(2), Asynchronous(1)
STANDBY	SLEEPCFG.SLEEPMODE = STANDBY	Synchronous(3), Asynchronous(1)

Notes:

1. Asynchronous: Interrupt is generated on GCLK generic clock, external clock, or an external event.
2. Synchronous: Interrupt is generated on the APB clock.
3. Synchronous interrupt: Only for peripherals configured to run in Standby sleep mode.

Note: The type of wake-up source (synchronous or asynchronous) is specified in the interrupt section of each peripheral.

The sleep modes (Idle, Standby) and their effect on clock activity and the regulator are described in the table and the sections below.

Table 14-3. Sleep Mode Overview

Mode	CPU Clock	AHB/APB Clocks	Main Clock	GCLK0 Clock	GCLK1 -->n Clocks	Clock Sources		Regulator
						ONDEMAND = 0	ONDEMAND = 1	
IDLE	Stop	Run(1)	Run	Run	Run/Stop(2)	Run	Run/Stop(3)	Main(8)
STANDBY	Stop	Stop(4)	Stop(4)	Stop if RUNSTDBY=0		Stop if RUNSTDBY=0		ULP(7)
				Stop/Run if RUNSTDBY=1(5)		Run if RUNSTDBY=1	Stop/Run if RUNSTDBY=1(6)	

Notes:

1. The AHB and APB clocks operate up to the MCLK, and are supplied only to the peripherals requesting them. The other peripherals, not requesting the clocks, are gated at MCLK output.
2. Each GCLK (GCLK1 to GCLKn) operates if at least one peripheral requests the associated generated clock. If no peripheral requests the clock, it is stopped.
3. The clock source remains active if at least one GCLK Generator requests it. If no GCLK Generator is requesting the clock, the source is stopped. It will automatically restart when a peripheral requests a clock from a GCLK that is supplied by this clock source.
4. The AHB and APB clocks are stopped unless requested by at least one peripheral, and in this case, only provided to this/these peripherals through GCLK0 and MCLK.
5. Each GCLK generator is stopped unless the clock it generates is requested by at least one peripheral.
6. Each Clock Source is stopped unless the clock it generates is requested by at least one GCLK Generator.
7. The regulator state can be programmed using the SUPC.VREG.RUNSTDBY bit.
8. If running on a slow clock (e.g., 32 kHz), the ULP regulator is used.

14.4.4.1. Idle Sleep Mode

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

The CPU is stopped.

The clock source feeding the GCLK generator 0, the GCLK generator 0, and the MCLK are kept active. The AHB/APB clocks are gated at the MCLK output, unless requested by a peripheral.

Other clock sources and GCLK generators may be running or stopped depending on the ONDEMAND bit of each clock source and whether any peripherals are requesting these clocks.

- **Entering Idle sleep mode**: The Idle sleep mode is entered by setting SLEEPCFG.SLEEPMODE to IDLE and by executing the Wait For Interrupt (WFI) instruction. Additionally, if the SLEEPONEXIT bit in the ARM Cortex System Control (SCR) register is set, the Idle sleep mode will also be entered when the CPU exits the lowest priority interrupt service routine (ISR). This feature is useful for applications that only require the processor to run when an interrupt occurs. Before entering Idle sleep mode, the Sleep Configuration (SLEEPCFG) register must be properly configured.

- **Exiting Idle sleep mode**: The processor wakes the system when it detects any non-masked interrupt with sufficient priority to cause exception entry. The system returns to the Active mode. The CPU and affected peripherals are restarted.

14.4.4.2. Standby Sleep Mode

The Standby sleep mode is the minimum power configuration, preserving the state of the logic and the contents of the SRAM.

This mode depends on:

- The peripherals running in standby and requesting their asynchronous GCLK clock or their synchronous AHB/APB clock
- The RUNSTDBY bit of the GCLK generators
- The RUNSTDBY/ONDEMAND bit combination of the clock sources

Each clock source and GCLK generator can be:

- Stopped for the entire duration of standby
- Running for the entire duration of standby
- Clock sources and GCLK generators can be automatically woken up and switched off depending on the clocks requested by peripherals during standby (SleepWalking). For example, a peripheral can run during standby and request its GCLK asynchronous clock, which will wake up the related GCLK and clock source. Another peripheral may request its APB clock, which will wake up the MCLK, GCLK generator 0 and the related clock source running. In this case, other AHB/APB clocks remain gated at the MCLK output.

As described above, depending on the configuration, the device's current consumption in Standby sleep mode may vary slightly.

All features that don't require CPU intervention are supported in Standby sleep mode. Here are some examples:

- Autonomous peripherals features
- Features that use the event system for autonomous communication between peripherals
- Features that rely on on-demand clock
- DMA transfers

Entering Standby sleep mode: This mode is entered by setting SLEEPCFG.SLEEPMODE to STANDBY and executing the WFI instruction. The SLEEPONEXIT feature is also available in Standby sleep mode, as in Idle mode.

Exiting Standby sleep mode: Any peripheral capable of generating an asynchronous interrupt can wake up the system. For example, a peripheral running on a GCLK clock can trigger an interrupt. When an enabled asynchronous wake-up event occurs and the system is awakened, the device will either execute the interrupt service routine or resume the normal program execution, depending on the configuration of the Priority Mask (PRIMASK) register in the CPU. The system can also be awakened by a synchronous interrupt from any peripheral configured to run in the Standby sleep mode.

14.4.5. Debug Operation

When the CPU is halted in debug mode the PM continues normal operation.

14.5. Dependencies

14.5.1. I/O Lines

Not applicable.

14.5.2. Power Management

The PM peripheral will continue to operate in any sleep mode where the selected source clock is running. Refer to Functional Description for more details on resets, sleep modes and wake-up.

14.5.3. Clocks

The PM bus clock (CLK_PM_APB) can be enabled or disabled in the Main Clock (MCLK), and the default state of CLK_PM_APB can be found in the *Peripheral Clock Masking* section of the *MCLK - Main Clock* chapter.

14.5.4. DMA

Not applicable.

14.5.5. Interrupts

Not applicable.

14.5.6. Events

Not applicable.

14.5.7. Analog Connections

Not applicable.

14.6. Register Summary - PM

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	SLEEPCFG	7:0						SLEEPMODE[2:0]		
		15:8								
		23:16								
		31:24								
0x08 ... 0x0B	Reserved									
0x0C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

14.6.1. Sleep Configuration

Name: SLEEPCFG
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bits 2:0 – SLEEPMODE[2:0] Sleep Mode

Note: There is a small latency between the store instruction and actual writing of the SLEEPCFG register due to bridges. Software must verify that the SLEEPCFG register contains the intended value before executing the Wait For Interrupt (WFI) instruction.

Value	Name	Description
0x0	—	Reserved
0x1	—	Reserved
0x2	IDLE	Idle sleep mode enabled
0x3	—	Reserved
0x4	STANDBY	Standby sleep mode enabled
Other	—	Reserved

14.6.2. Write Protection Control

Name: WPCTRL
Offset: 0x0C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write operation to be successful.

Value	Name	Description
0x505752	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	PM register write protection is disabled
1	Write protection is enabled on PM registers that have the Local Write-Protection property. Non-debugger write attempts to these PM registers are ignored and will result in a bus error.

15. SUPC – Supply Controller

15.1. Features

- Voltage Regulator System
  - Main voltage regulator: Low Dropout (LDO) in Active mode
  - Ultra Low-Power (ULP) voltage regulator in Standby sleep mode
- Brown-out Detection Monitors the Power Supply to Prevent Operation Below a Programmable Level
  - Three modes:
    - Enabled (Continuous)
    - Sampled
    - Disabled
  - Separate selection of mode for active and sleep modes
  - Voltage Level Monitor (VLM) with interrupt
  - Programmable VLM level relative to the BOD level
- MVIO System
  - Two mode configurations:
    - Dual Supply mode, with independent supply target voltage, ramp and removal on V_{DD} and V_{DDIO2}
    - Single Power Supply mode, with MVIO supply monitors are turned off to minimize current consumption
  - Mode selection fuse
  - Interrupt request for V_{DDIO2} power loss
  - CPU-readable V_{DDIO2} Power Status bit (domain-monitor status)
  - Pin access and port override function the same way as with V_{DD}

15.2. Overview

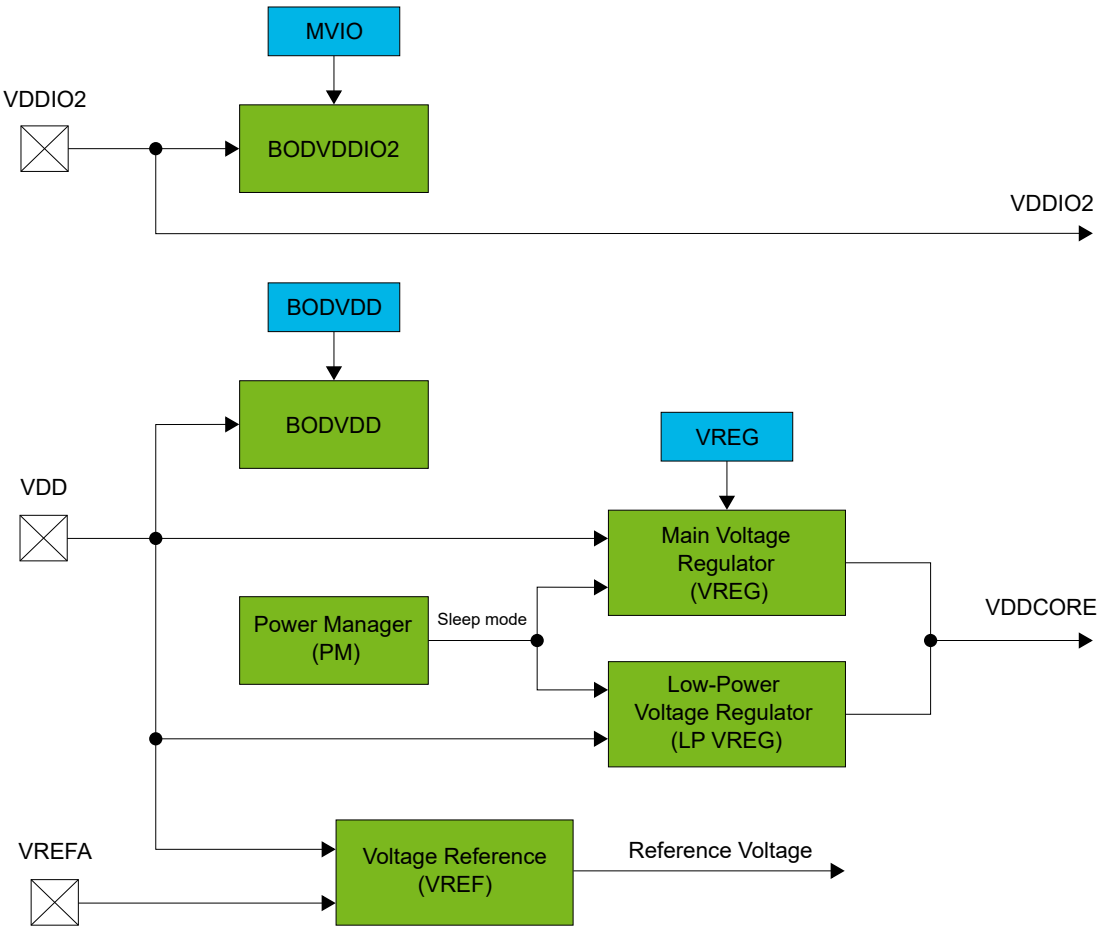
The Supply Controller (SUPC) manages the voltage reference and power supply of the device. The SUPC controls the voltage regulators for the core (VDDCORE) domain and adjusts the voltage regulators according to the sleep modes or user configuration.

The SUPC includes two brown-out detectors: BODVDD and BODVDDIO2. BODVDD monitors the voltage applied to the device (VDD), and BODVDDIO2 monitors the internal voltage of the V_{DDIO2} domain. BODVDD can monitor the supply voltage either continuously (Continuous mode) or periodically (Sampling mode).

The output supply level of the main voltage regulator is automatically determined by the sleep mode selected in the Power Manager.

15.3. Block Diagram

Figure 15-1. SUPC Block Diagram



15.3.1. Signal Description

Table 15-1. SUPC External Signal Description

Signal Name	Type	Description
VDD	Supply	Voltage supply V_{DD}
VDDIO2	Supply	Voltage supply V_{DDIO2}
VREFA	Analog Input	External voltage reference

15.4. Functional Description

15.4.1. Initialization

Before the brown-out detector (BODVDD) is enabled, the following must be configured:

- Set the BODVDD threshold level in the VDD Brown-Out Detector Control register (BODVDD.LEVEL)
- Set the configuration for Active and Standby modes (BODVDD.ACTCFG, BODVDD.STDBYCFG)
- Set the sampling frequency if the BODVDD runs in Sampling mode (BODVDD.SAMPFREQ)
- Set the Write Lock (BODVDD.WRTLOCK) bit to prevent future writes to the register

15.4.2. Brown-out Detectors

15.4.2.1. BODVDD Basic Operation

The brown-out detector (BODVDD) monitors the power supply and compares the voltage with two programmable brown-out threshold levels. The brown-out threshold level defines when to generate a Reset. A Voltage Level Monitor (VLM) analyzes the power supply and compares it to a threshold higher than the BODVDD threshold. The VLM can then generate an interrupt request as an early warning when the supply voltage is about to drop below the VLM threshold. The VLM threshold level is expressed as a percentage above the BODVDD threshold level.

The BODVDD is mainly controlled by fuses. The mode used in Standby sleep mode and Power-Down sleep mode can be altered during normal program execution. The VLM part of the BODVDD is also controlled by the I/O registers. When activated, the BODVDD can operate in Continuous mode, where it is continuously active, or in Sampling mode, where it is activated periodically to check the supply voltage level.

15.4.2.2. VLM Basic Operation

The Voltage Level Monitor function can be enabled by writing a '1' to the VLM Interrupt Enable (VLM) bit in the Interrupt Enable Set (SUPC.INTENSET) register. The Voltage Level Monitor cannot be enabled separately from the brown-out detector. However, when the BOD is enabled, the Voltage Level Monitor can be configured by VLMLVL to be either OFF or to trigger at a certain percentage above the BOD level.

The VLM interrupt is configured by writing the VLM Configuration (VLMCFG) bits in SUPC.BODVDD. An interrupt is requested when the supply voltage falls below, rises above, or crosses the VLM threshold.

The VLM functionality will follow the BODVDD mode. If BODVDD is disabled (BODVDD.ENABLE=0), the VLM will not be enabled, even if the BODVDD.VLMLVL is not equal to 0. If BODVDD is using Sampled mode, the VLM will also be sampled.

When enabling VLM, the Interrupt flag will always be set if VLMCFG = 0x2, and may be set if VLMCFG is configured to FALLING (0x0) or RISING (0x1).

VLM Interrupts are generated on events that occur after VLMCFG is configured. E.g., if the VLM is already detecting a voltage that is too low, changing setting to FALLING will not result in an interrupt. Therefore, it is recommended to read the STATUS.VLM bit after configuring VLMCFG to determine the current state.

In addition, the Interrupt flag must be cleared, as it may contain old information.

The VLM threshold is defined by writing the VLM Level (VLMLVL) bits in SUPC.BODVDD.

15.4.2.3. Enabling, Disabling, and Resetting

After power or user reset, the BODVDD register values are loaded from the BOOTCFG fuse.

The BODVDD is enabled by writing a '1' to the Enable bit in the BODVDD control (BODVDD.ENABLE) register.

The BODVDD is disabled by writing a '0' to BODVDD.ENABLE.

15.4.2.4. Brown-Out Reset Operation

The brown-out reset (BODVDD) monitors the V_{DD} supply and compares the voltage with the brown-out threshold level set in the VDD Brown-Out Detector Control (BODVDD) register using the LEVEL bitfield (BODVDD.LEVEL). When V_{DD} drops below the brown-out threshold level, the BODVDD generates a Reset.

At start-up or at Power-on Reset (POR), the BODVDD register values are loaded from the BOOTCFG fuse.

15.4.2.5. Continuous Mode

Continuous mode is the default mode for BODVDD.

The BODVDD continuously monitors the V_{DD} supply voltage if it is enabled in VDD Brown-Out Detector Control BODVDD register (BODVDD.ENABLE = 1) .

The Continuous mode is enabled in Active mode for BODVDD by writing a '0' to the ACTCFG bit (BODVDD.ACTCFG = 0).

The Continuous mode is enabled in Standby mode by writing a '0' to the STDBYCFG bit (BODVDD.STDBYCFG = 0).

15.4.2.6. Sampling Mode

The Sampling mode is a low-power mode in which BODVDD is repeatedly enabled each time the sampling clock ticks, provided BODVDD is enabled in the VDD Brown-Out Detector Control BODVDD register (BODVDD.ENABLE = 1). The BODVDD will monitor the supply voltage for a short period of time and then enter a Low-Power Disabled state until the next sampling clock tick.

The Sampling mode is enabled in Active mode for BODVDD by writing a '1' to the ACTCFG bit (BODVDD.ACTCFG = 1).

The Sampling mode is enabled in Standby mode by writing a '1' to the STDBYCFG bit (BODVDD.STDBYCFG = 1).

15.4.2.7. Sleep Mode Operation

15.4.2.7.1. Standby Mode

The BODVDD can be used in Standby mode if the BODVDD is enabled and the corresponding Run in Standby (BODVDD.RUNSTDBY) bit is written to '1'.

The BODVDD can be configured to operate in either Continuous or Sampling mode by writing a '1' to the configuration bit for Standby sleep mode (BODVDD.STDBYCFG) bit.

15.4.3. Voltage Regulator System Operation

15.4.3.1. VREG Basic Operation

The Low Dropout regulator (LDO) allows the device to be powered over the full specified operating range while limiting internal voltage to a level suitable for low-voltage transistors.

The secondary regulator (ULP) is a multi-mode, ultra-low power regulator used for the lowest power consumption for both code execution and Sleep.

15.4.3.2. Initialization

After a Reset, the LDO voltage regulator supplying VDDCORE is enabled.

15.4.3.3. Enabling, Disabling, and Resetting

The LDO main voltage regulator is enabled after any reset. The output supply level of the main voltage regulator is automatically determined by the sleep mode selected in the *Power Manager* peripheral.

15.4.3.4. Sleep Mode Operation

In Standby sleep mode, the Ultra-Low-Power Voltage Regulator (ULPVREG) is used to supply VDDCORE.

15.4.3.5. Debug Mode

The LDO is always enabled and in High-Power mode when a debugger is attached. When the CPU is halted in Debug mode, the PM continues its normal operation. If Standby Sleep mode is requested by the system while in Debug mode, the LDO is kept on. As a consequence, the power consumption may be higher than normal when a debugger is attached.

15.4.4. Multi-Voltage I/O

15.4.4.1. Initialization

Initialize the MVIO in Dual Supply configuration by following these steps:

1. Program the MVIO Operation Mode (MVIOMODE) bit in the User Configuration Fuse (USERCFG) register from BOOTCFG - Boot Configuration Fuses to the Dual Supply configuration (BOOTCFG.USERCGF.MVIOMODE = '0').
2. Optional: Write the VDDIO2 OK Interrupt Enable bit to '1' in the Interrupt Enable Set (INTENSET) register (INTENSET.VDDIO2OK=1).
3. Read the VDDIO2 OK (STATUS.VDDIO2OK) bit to check if the V_{DDIO2} voltage is within the acceptable range for operation.
4. Configure and use the PORT pins powered by V_{DDIO2}.

If the MVIOMODE bit in the Boot Configuration Fuse is programmed to the Single Supply configuration (BOOTCFG.USECFG.MVIOMODE = '1'), the MVIO peripheral is not configurable, and the STATUS register will not be updated.

15.4.4.2. Low Power Operation

When operating in Dual Supply mode, the MVIO peripheral can be configured in one of three ways, selected by VDDIO2 Configuration bit field in MVIO Control register (MVIO.VDDIO2CFG):

- Normal mode (NORMAL), in which the V_{DDIO2} voltage domain is active and voltage detectors are enabled
- Low Power mode (LOWPOWER), in which the V_{DDIO2} voltage domain is active, but the voltage detectors are turned off to save power
- Off mode (VDDIO2OFF), in which the MVIO-capable I/O pins are disabled. This mode can be used, for example, during sleep mode to save power when the V_{DDIO2} domain is not needed.

In Low-Power mode, if the voltage on the VDDIO2 pin(s) drops below the POR voltage and is then restored, the VDDIO2 Low-Power mode POR (STATUS.VDDIO2LPMPOR) bit will be set. The power failure will not be detected until power-on VDDIO2 is restored.

15.4.4.3. Power Sequencing

The system supports the following power ramp scenarios for MVIO when configured in Dual Supply mode:

- Supply ramp of V_{DD} (V_{DDIO}) before V_{DDIO2}
- Supply ramp of V_{DDIO2} before V_{DD} (V_{DDIO})
- V_{DD} (V_{DDIO}) loses and regains power
- V_{DDIO2} loses and regains power

When either voltage domain loses power, the MVIO I/O pins are tri-stated. The pins will reload the current configuration of the PORT registers if V_{DDIO2} regains power. If V_{DD} (V_{DDIO}) loses power, the device will reset, and all PORTs must be reinitialized. Refer to the *Electrical Characteristics* section for V_{DD} and V_{DDIO2} power supply thresholds.

If configured, an interrupt can be triggered either when the voltage drops below the threshold or when the voltage is restored, but never on both. The interrupt is triggered only in Dual Power mode and with V_{DDIO2} configured in Low-Power mode, meaning the voltage monitors are disabled.

Note:
When a peripheral is connected to the MVIO pins, it will continue to operate when the pins are tri-stated. Monitor the VDDIO2 OK (STATUS.VDDIO2OK) flag to ensure the correct operation of the peripheral and I/O pins.

15.4.4.4. Sleep Mode Operation

When the module is enabled by fuses, it will be ON in all sleep modes. If the power drops in V_{DDIO2} when in Sleep, an interrupt can be generated.

15.5. Dependencies

15.5.1. I/O Lines

Using the SUPC's I/O lines requires the I/O pins to be configured.

All external I/O pins are powered by V_{DDIO}. Level translators are used to interface between the externally supplied V_{DDIO} and the internally regulated V_{DD_SC}.

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

15.5.2. Power Management

The SUPC will continue to operate in any sleep mode where the selected source clock is running. The SUPC'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 the *Power Manager* chapter for details on the different sleep modes.

15.5.3. Clocks

The SUPC bus clock (CLK_SUPC_APB) can be enabled and disabled by the Main Clock Controller, and its default state can be found in the *Peripheral Clock Masking* section.

15.5.4. DMA

Not applicable.

15.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use SUPC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 15-2. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
SUPC	VDDIO2LPMPOR	V_{DDIO2} was lost and restored while the MVIO system was in Low Power mode	
	VDDIO2OK	The voltage level on V_{DDIO2} is insufficient for communication, or V_{DDIO2} has been restored	
	VLM	The Voltage Level Monitor has been triggered	
	BODVDDRDY	BODVDD is ready	

15.5.6. Events

The SUPC can generate the events described in the table below.

Table 15-3. Event Generators in SUPC

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
SUPC	MVIO	The voltage level on VDDIO2 is insufficient for communication.	Level	CLK_SUPC_APB	As long as INTFLAG.VDDIO2OK is set
	VLM	VLM is triggered	Level	CLK_SUPC_APB	As long as INTFLAG.VLM is set

Note: The MVIO event cannot be used when running in unprotected mode.

Writing a '1' to an Event Output Enable bit in the Event Control Register (EVCTRL.xxEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event

Refer to the *EVSYS – Event System* chapter for details on how to configure the Event System.

15.5.7. Analog Connections

Not applicable.

15.6. Register Summary - SUPC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	INTENCLR	7:0			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
		15:8								
		23:16								
		31:24								
0x08	INTENSET	7:0			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
		15:8								
		23:16								
		31:24								
0x0C	INTFLAG	7:0			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
		15:8								
		23:16								
		31:24								
0x10	INTFLAGSET	7:0			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
		15:8								
		23:16								
		31:24								
0x14	STATUS	7:0			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
		15:8								
		23:16								
		31:24								
0x18 ... 0x1B	Reserved									
0x1C	BODVDD	7:0		RUNSTDBY	STDBYCFG				ENABLE	
		15:8				SAMPFREQ				ACTCFG
		23:16							LEVEL[1:0]	
		31:24	WRTLOCK				VLMCFG[1:0]		VLMLVL[1:0]	
0x20	VREG	7:0		RUNSTDBY						
		15:8								
		23:16								
		31:24								
0x24	MVIO	7:0					VDDIO2CFG[1:0]			MODE
		15:8								
		23:16								
		31:24								
0x28	EVCTRL	7:0							VLMEO	MVIOEO
		15:8								
		23:16								
		31:24								
0x2C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

15.6.1. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
Access			R/W	R/W	R/W			R/W
Reset			0	0	0			0

Bit 5 – VDDIO2LPMPOR VDDIO2 Low-Power Mode POR Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the VDDIO2LPMPOR interrupt enable.
This bit will read as the current value of the VDDIO2LPMPOR interrupt enable.

Bit 4 – VDDIO2OK VDDIO2OK Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the VDDIO2OK interrupt enable.
This bit will read as the current value of the VDDIO2OK interrupt enable.

Bit 3 – VLM Voltage Level Monitor Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Voltage Level Monitor interrupt enable.
This bit will read as the current value of the Voltage Level Monitor interrupt enable.

Bit 0 – BODVDDRDY BODVDD Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the BODVDD Ready interrupt enable.
This bit will read as the current value of the BODVDD Ready interrupt enable.

15.6.2. Interrupt Enable Set

Name: INTENSET
Offset: 0x08
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
Access			R/W	R/W	R/W			R/W
Reset			0	0	0			0

Bit 5 – VDDIO2LPMPOR VDDIO2 Low Power Mode POR Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the VDDIO2LPMPOR interrupt enable.
This bit will read as the current value of the VDDIO2LPMPOR interrupt enable.

Bit 4 – VDDIO2OK VDDIO2 OK Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the VDDIO2OK interrupt enable.
This bit will read as the current value of the VDDIO2OK interrupt enable.

Bit 3 – VLM Voltage Level Monitor Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Voltage Level Monitor interrupt enable.

Bit 0 – BODVDDRDY BODVDD Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the BODVDD Ready interrupt enable.
This bit will read as the current value of the BODVDD Ready interrupt enable.

15.6.3. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x0C
Reset: 0x000000XX
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
Access			R/W	R/W	R/W			R/W
Reset			x	x	x			x

Bit 5 – VDDIO2LPMPOR VDDIO2 Low Power Mode POR
This flag is set on a '0'-to-'1' transition of the corresponding flag in the Status register.
It can be cleared by writing a '1' to it.

Bit 4 – VDDIO2OK VDDIO2 OK
This flag is set on either a '0'-to-'1' or '1'-to-'0' transition of the corresponding flag in the Status register.
It can be cleared by writing a '1' to it.

Bit 3 – VLM Voltage Level Monitor
This flag is set on a '0'-to-'1' and/or an '1'-to-'0' transition of the corresponding flag in the Status register, according to the configuration of the Voltage Level Monitor Interrupt Configuration (VLMCFG) bit field in the VDD Brown-Out Detector Control (BODVDD) register.
It can be cleared by writing a '1' to it.

Bit 0 – BODVDDRDY BODVDD Ready
This flag is set on '0'-to-'1' transition of the corresponding flag in the Status register.
It can be cleared by writing a '1' to it.

15.6.4. Interrupt Flag Software Set

Name: INTFLAGSET
Offset: 0x10
Reset: 0x00000000
Property: Local Write-Protection

The Interrupt Flag Software Set register provides a way to set the INTFLAG bits individually by software. This allows the associated INTFLAG bit to be set so that the ISR can be tested. This feature may be useful for safety applications.

The read value reflects the state of the INTFLAG.

Write to '1' to set the corresponding bit in INTFLAG.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
Access			R/W	R/W	R/W			R/W
Reset			0	0	0			0

Bit 5 – VDDIO2LPMPOR VDDIO2 Low Power Mode POR
Writing a '0' to this flag has no effect.
Writing a '1' to this bit sets the corresponding bit in INTFLAG.

Bit 4 – VDDIO2OK VDDIO2 OK
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the corresponding bit in INTFLAG.

Bit 3 – VLM Voltage Level Monitor
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the corresponding bit in INTFLAG.

Bit 0 – BODVDDRDY BODVDD Ready
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the corresponding bit in INTFLAG.

15.6.5. Status

Name: STATUS
Offset: 0x14
Reset: 0x000000XX
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			VDDIO2LPMPOR	VDDIO2OK	VLM			BODVDDRDY
Access			R	R	R			R
Reset			x	x	x			x

Bit 5 – VDDIO2LPMPOR VDDIO2 Low Power Mode POR
The VDDIO2LPMPOR status flag is set if the selected MVIO configuration, as indicated in MVIOCTRL, does not match the actual state of the MVIO system.
This occurs when the VDDIO2 Configuration (VDDIO2CFG) bit field of the MVIO Control (MVIO) register is set to LOWPOWER and the power source for VDDIO2 drops low enough to trigger VDDIO2POR.
The MVIO will then return to operating in Normal mode, even though MVIO.VDDIO2CFG indicates Low-Power mode.
This bit is cleared by hardware when MVIO.VDDIO2CFG is written to NORMAL.

Value	Description
0x0	VDDIO2 operates according to the configuration setting in MVIOCTRL
0x1	VDDIO2 is configured to Low-Power mode, and power-on VDDIO2 has failed and subsequently been restored

Bit 4 – VDDIO2OK VDDIO2 OK
The VDDIO2 voltage is within acceptable range for operation.

Value	Description
0x0	VDDIO2 is disabled, not ready, or the voltage is outside the acceptable range
0x1	The VDDIO2 monitor is enabled and ready, and voltage is within the acceptable range, or the low-power mode is enabled

Bit 3 – VLM Voltage Level Monitor
This flag shows the output level from the comparator of the VLM.

Value	Description
0x0	The voltage is above the VLM threshold

Value	Description
0x1	The voltage is below the VLM threshold

Bit 0 – BODVDDRDY BODVDD Ready

The BODVDD can be enabled at start-up from the BOOTCFG fuse.
The state of this bit is only applicable in BOD Continuous mode. In Sampling mode, this bit is never set.

Value	Description
0x0	BODVDD is not ready
0x1	BODVDD is ready

15.6.6. VDD Brown-Out Detector Control

Name: BODVDD
Offset: 0x1C
Reset: 0x00000000
Property: Local Write-Protection

After a Reset, the bit field values of this register are determined by the corresponding BOOTCFG fuse which is BOD Configuration Fuse (BODCFG) . Refer to the BOOTCFG - Boot Configuration for details.

Fuses section for details

Bit	31	30	29	28	27	26	25	24
	WRTLOCK				VLMCFG[1:0]		VLMLVL[1:0]	
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit	23	22	21	20	19	18	17	16
							LEVEL[1:0]	
Access							R/W	R/W
Reset							0	0

Bit	15	14	13	12	11	10	9	8
				SAMPFREQ				ACTCFG
Access				R/W				R/W
Reset				0				0

Bit	7	6	5	4	3	2	1	0
		RUNSTDBY	STDBYCFG				ENABLE	
Access		R/W	R/W				R/W	
Reset		0	0				0	

Bit 31 – WRTLOCK Write Lock
This bit locks the BODVDD register for future writes, effectively freezing the BODVDD configuration. Any write to this BODVDD register will be discarded and return a bus error if WRTLOCK = '1'. The Reset value is loaded from the WRTLOCK bit in BOOTCFG.BODCFG.

Value	Description
0x0	The BODVDD configuration is not locked
0x1	The BODVDD configuration is locked

Bits 27:26 – VLMCFG[1:0] Voltage Level Monitor Interrupt Configuration
This bit field selects which incident will trigger a VLM interrupt. The Reset value is loaded from the VLMCFG bit in BOOTCFG.BODCFG.

Value	Name	Description
0x0	FALLING	VDD falls below the VLM threshold
0x1	RISING	VDD rises above the VLM threshold
0x2	BOTH	VDD crosses the VLM threshold

Bits 25:24 – VLMLVL[1:0] Voltage Level Monitor Level
The Reset value is loaded from the VLMLVL bit in BOOTCFG.BODCFG.

Value	Name	Description
0x0	OFF	VLM is disabled

Value	Name	Description
0x1	5ABOVE	The VLM threshold is 5% above the BOD threshold
0x2	15ABOVE	The VLM threshold is 15% above the BOD threshold
0x3	25ABOVE	The VLM threshold is 25% above the BOD threshold

Bits 17:16 – LEVEL[1:0] BOD Level

This bit field controls the BOD threshold level. The Reset value is loaded from the LEVEL bits in BOOTCFG.BODCFG.

Notes:

- The given values are typical values. For further details, refer to the *Electrical Characteristics* section
- The BOD Level must only be changed or written when the BOD is disabled
- To change the BOD level when the BOD is enabled: first, write a '0' to the ENABLE bit without updating the BOD level. Then, perform another write with LEVEL set to the desired value and ENABLE=1

Value	Name	Description
0x0	BODLEVEL0	1.90V
0x1	BODLEVEL1	2.45V
0x2	BODLEVEL2	2.70V
0x3	BODLEVEL3	2.85V

Bit 12 – SAMPFREQ Sampling Frequency

This bit selects the BOD sample frequency. The Reset value is loaded from the SAMPFREQ bit in BOOTCFG.BODCFG.

Value	Name	Description
0	128HZ	The sample frequency is 128 Hz
1	32HZ	The sample frequency is 32 Hz

Bit 8 – ACTCFG BOD Configuration in Active Mode

The Reset value is loaded from the ACTCFG bit in BOOTCFG.BODCFG.

Value	Name	Description
0	ENABLE	In active mode, the BOD operates in continuous mode
1	SAMPLE	In active mode, the BOD operates in sampling mode

Bit 6 – RUNSTDBY Run in Standby

The Reset value is loaded from the RUNSTDBY bit in BOOTCFG.BODCFG.

Value	Description
0x0	The BOD is not running in Standby sleep mode
0x1	The BOD is running in Standby sleep mode if enabled by the ENABLE bit

Bit 5 – STDBYCFG BOD Configuration in Standby Sleep Mode

If the RUNSTDBY bit is set to '1', the STDBYCFG bit determines the BOD configuration in Standby sleep mode.

The Reset value is loaded from the STDBYCFG bit in BOOTCFG.BODCFG.

Value	Description
0x0	In Standby sleep mode, the BOD is configured in Continuous mode
0x1	In Standby sleep mode, the BOD is configured in Sampling mode

Bit 1 – ENABLE Enable

The Reset value is loaded from the ENABLE bit in BOOTCFG.BODCFG.

Value	Name	Description
0	DISABLE	BOD is disabled
1	ENABLE	BOD is enabled

15.6.7. Voltage Regulator Control

Name: VREG
Offset: 0x20
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 6 – RUNSTDBY Run in Standby
For more details, refer to the *Effect Of RUNSTDBY On Regulator Usage* table.

Value	Description
0x0	Automatically switches from the LDO to the ULP regulator in sleep and when running on the 32.768 kHz oscillator
0x1	The main LDO voltage regulator is ON and powers the device in Standby sleep mode

15.6.8. MVIO Control

Name: MVIO
Offset: 0x24
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					VDDIO2CFG[1:0]			MODE
Access					R/W	R/W		R
Reset					0	0		0

Bits 3:2 – VDDIO2CFG[1:0] VDDIO2 Configuration
This bit field is used to configure VDDIO2 when operating in Dual Mode. When operating in Single Mode, writes to this bit field will be ignored. Switching between LOWPOWER and VDDIO2OFF modes must go via NORMAL mode.
If the MVIO system is in NORMAL mode with STATUS.VDDIO2OK=0, writes to this bit field will be ignored.

Value	Name	Description
0x0	NORMAL	VDDIO2 with voltage protection enabled
0x1	LOWPOWER	VDDIO2 with voltage protection disabled
0x2	VDDIO2OFF	VDDIO2 domain is disabled. I/O pins are not available.
Other	—	Reserved

Bit 0 – MODE Operation Mode
This bit indicates the MVIO operation mode. The configuration is loaded from the BOOTCFG - Boot Configuration Fuse (USERCFG.MVIOMODE) and cannot be changed by the user. The content of the MODE is reset on POR and BOR, and is re-loaded from fuse during the system initialization after all resets.

Value	Name	Description
0	DUAL	MVIO operating in dual supply mode
1	SINGLE	MVIO operating in single supply mode

15.6.9. Event Control

Name: EVCTRL
Offset: 0x28
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – VLMEO VLM Event Output Enable
This bit indicates whether the VLM event output is enabled.

Value	Description
0x0	The VLM event output is disabled and no event will be generated
0x1	The VLM event output is enabled and an event will be generated

Bit 0 – MVIOEO MVIO Event Output Enable
This bit indicates whether the MVIO event output is enabled.

Value	Description
0x0	The MVIO event output is disabled and no event will be generated
0x1	The MVIO event output is enabled and an event will be generated

15.6.10. Write Protection Control

Name: WPCTRL
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write to be successful.

Value	Name	Description
0x535550	KEY	Allows writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	SUPC register write protection is disabled
1	Write protection is enabled on SUPC registers with the Local Write-Protection property. Non-debugger writes to SUPC registers with Local Write-Protection are ignored and generate a bus error.

16. RSTC – Reset Controller

16.1. Features

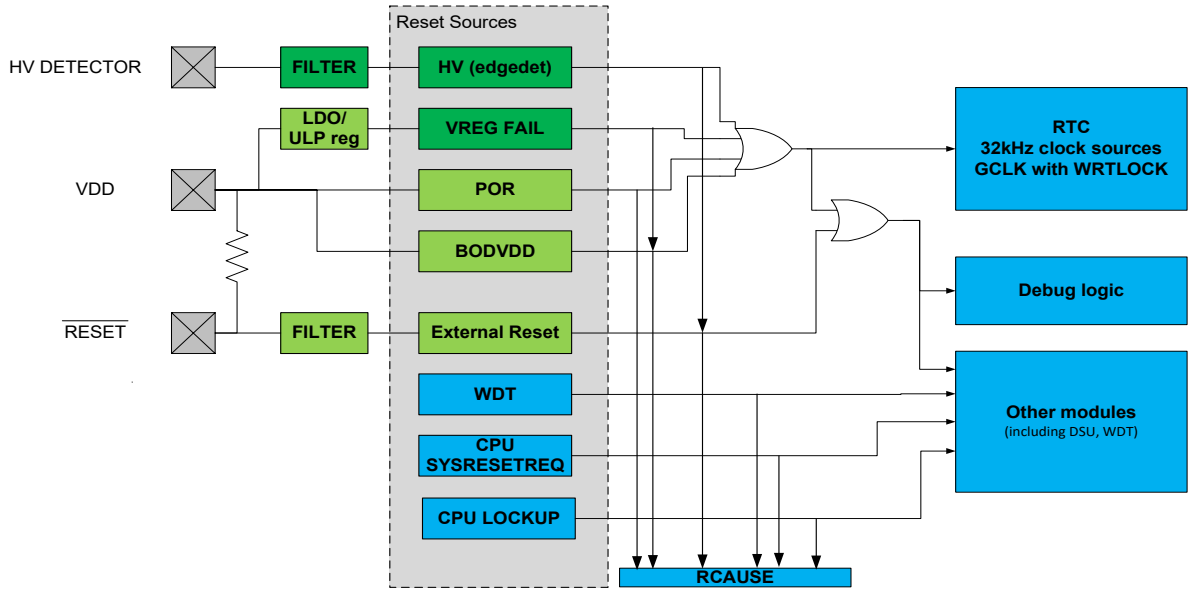
- Reset the Microcontroller (MCU) and set it to an initial state according to the reset source
- Reset Cause Register to read the reset source from the application code
- Multiple Reset Sources
  - Power supply Reset sources: POR, BODVDD,
  - User Reset sources: External Reset ($\overline{\text{RESET}}$), Watchdog Reset, System Reset Request and CPU Lockup Reset

16.2. Overview

The Reset Controller (RSTC) manages the Reset of the microcontroller. It issues a microcontroller Reset, sets the device to its initial state and allows the Reset source to be identified by software.

16.3. Block Diagram

Figure 16-1. Reset System



16.3.1. Signal Description

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

16.4. Functional Description

16.4.1. Initialization

After a device reset, the RSTC is enabled and the Reset Cause (RCAUSE) register indicates the POR source.

16.4.2. Operation

16.4.2.1. Enabling, Disabling, and Resetting

The RSTC module is always enabled.

16.4.2.2. Reset Causes and Effects

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

These are the types of Reset sources:

- Power supply Reset: Resets caused by an electrical issue. It covers POR and BOD Resets.
- User Reset: Resets caused by the application. It covers external Resets, system Reset requests, and Watchdog Resets (WDR).

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

Table 16-1. Effects of the Different Reset Causes

	Power Supply Reset	User Reset	
	POR, BODVDD	External Reset	WDT Reset, System Reset Request
RTC, OSC32KCTRL, RSTC	Reset	-	-
GCLK with WRTLOCK	Reset	-	-
Debug logic	Reset	Reset	-
Others	Reset	Reset	Reset

An external Reset is generated when the $\overline{\text{RESET}}$ pin is pulled low.

The POR and BODVDD Reset sources are generated by their corresponding modules in the Supply Controller Interface (SUPC). Refer to the *Supply Controller* section for more information.

The Watchdog Timer (WDT) Reset is generated by the Watchdog Timer. Refer to the *Watchdog Timer* section for more information.

The System Reset Request is a Reset generated by the CPU when the SYSRESETREQ bit ,located in the Reset Control register of the CPU, is asserted. Refer to the Arm® Cortex® Technical Reference Manual on http://www.arm.com for more information.

The CPU Lockup Reset Request is a Reset generated when the CPU enters the architecturally defined LOCKUP state. Refer to the Arm® Cortex® Technical Reference Manual on http://www.arm.com for more information.

Note: Refer to the *TRST Specification* in the *Power Supply* section of the *Electrical Characteristics* section for more information.

16.4.3. Sleep Mode Operation

The RSTC module is active in all sleep modes.

16.4.4. Debug Operation

When the CPU is halted in debug mode, the RSTC continues normal operation. If the DBGCTRL.LCKUPDIS bit is set, the CPU LOCKUP Reset source is disabled in debug mode. Refer to the DBGCTRL register for details.

16.5. Dependencies

16.5.1. I/O Lines

The $\overline{\text{RESET}}$ pin is enabled by default. It can be disabled and used as a general GPIO pin. Refer to the Control A register (CTRLA) for detailed information.

Using the peripheral I/O lines requires the I/O pin to be configured. Refer to the *Pinout and Multiplexing* section in the *PORT – I/O Pin Controller* chapter for additional information.

16.5.2. Power Management

The RSTC is always running.

16.5.3. Clocks

Not applicable.

16.5.4. DMA

Not applicable.

16.5.5. Interrupts

Not applicable.

16.5.6. Events

Not applicable.

16.5.7. Analog Connections

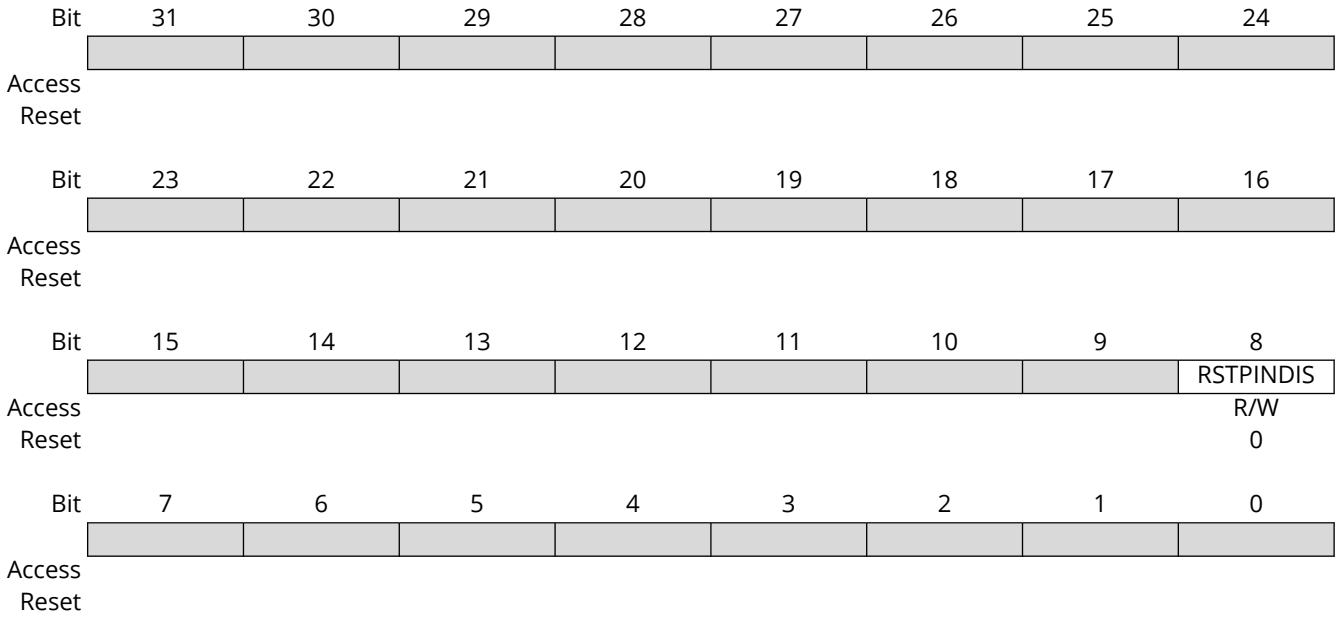
Not applicable.

16.6. Register Summary - RSTC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0								
		15:8								RSTPINDIS
		23:16								
		31:24								
0x04	RCAUSE	7:0		LOCKUP	SYST	WDT	EXT		BORVDD	POR
		15:8								
		23:16								
		31:24								
0x08	DBGCTRL	7:0								LCKUPDIS
		15:8								
		23:16								
		31:24								
0x0C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

16.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: Local Write-Protection



Bit 8 – RSTPINDIS Reset Pin Disable

Value	Description
0	The $\overline{\text{RESET}}$ pin function is enabled. Driving the $\overline{\text{RESET}}$ pin low resets the device.
1	The $\overline{\text{RESET}}$ pin function is disabled. The $\overline{\text{RESET}}$ pin now works as a regular GPIO pin and is controlled by the PORT peripheral.

16.6.2. Reset Cause

Name: RCAUSE
Offset: 0x04
Reset: 0x00000XXX
Property: –

When a Reset occurs, the bit corresponding to the Reset source is set to '1' and all other bits are written to '0'.

If multiple Resets are requested at the same time, RCAUSE will indicate all of the requests that were active simultaneously, meaning that multiple flags may be set.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		LOCKUP	SYST	WDT	EXT		BORVDD	POR
Access		R	R	R	R		R	R
Reset		x	x	x	x		x	x

Bit 6 – LOCKUP Software Reset Flag
This bit is set if software generates a system Reset.

Bit 5 – SYST System Reset Flag
This bit is set if a System Reset flag has occurred.

Bit 4 – WDT Watchdog Reset Flag
This bit is set if the Watchdog generates a system Reset.

Bit 3 – EXT External Reset Flag
This bit is set if the External pin generates a system Reset.

Bit 1 – BORVDD VDD Brownout Reset Flag
This bit is set if a VDD Brownout Reset occurs.

Bit 0 – POR Power-On Reset Flag
This bit is set if a Power-On Reset occurs.

16.6.3. Debug Control

Name: DBGCTRL
Offset: 0x08
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 0 – LCKUPDIS Lockup Disable

Value	Description
0	LOCKUP Reset is enabled in Debug mode
1	LOCKUP Reset is disabled in Debug mode

16.6.4. Write Protection Control

Name: WPCTRL
Offset: 0x0C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write to be successful.

Value	Name	Description
0x525354	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	RSTC register write protection is disabled
1	Write protection is enabled on the RSTC registers with the Local Write-Protection property. Non-debugger writes to the RSTC registers with Local Write-Protection are ignored and generate a bus error.

17. PAC - Peripheral Access Controller

17.1. Features

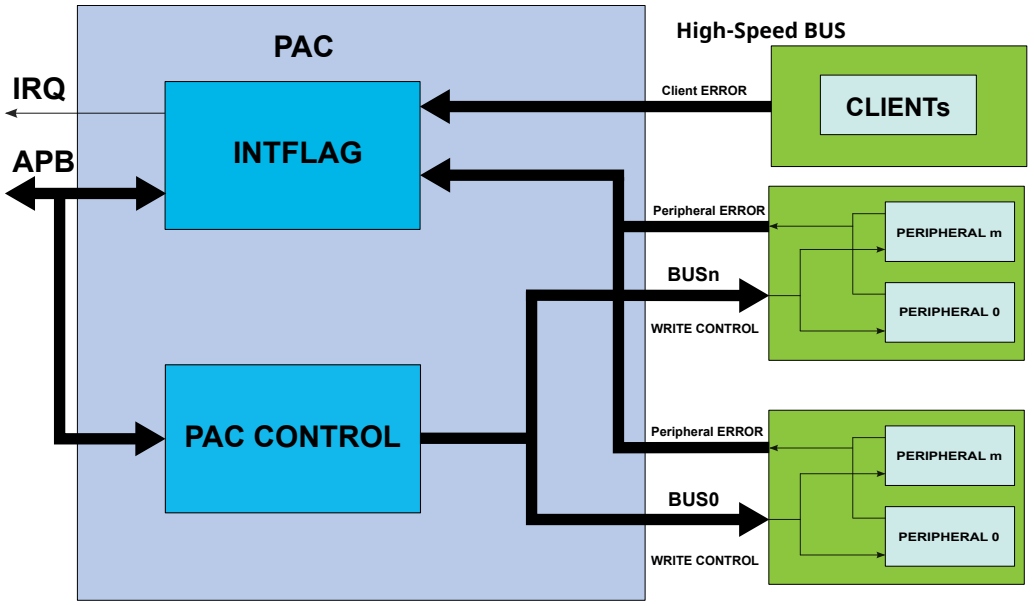
- Manages the write protection access
- Reports access errors for the peripheral modules or bridges

17.2. Overview

The Peripheral Access Controller (PAC) provides an interface for the locking and unlocking of peripheral registers within the device. It reports all violations that could happen when accessing a peripheral: Write-protected access, illegal access, enable protected access, access when clock synchronization or software reset is ongoing. These errors are reported in a unique interrupt flag for a peripheral. The PAC module also reports errors occurring at the client bus level, when an access to a non-existing address is detected.

17.3. Block Diagram

Figure 17-1. PAC Block Diagram



17.3.1. Signal Description

Not applicable.

17.4. Functional Description

17.4.1. Initialization

The PAC is always enabled after reset.

Only a hardware reset will reset the PAC module.

17.4.2. Operation

The PAC module allows the user to set, clear or lock the write protection status of all peripherals on all Peripheral Bridges.

If a peripheral register violation occurs, the Peripheral Interrupt Flag Status and Clear n (INTFLAGn) registers are updated to inform the user on the status of the violation in the peripherals connected

to the Peripheral Bridge n. The corresponding Peripheral Write Protection Status n (STATUSn) register gives the state of the write protection for all peripherals connected to the corresponding Peripheral Bridge n. Refer to the *Peripheral Access Errors* section for details.

The PAC module also reports the errors occurring at client bus level when an access to the reserved area is detected. The AHB Client Bus Interrupt Flag Status and Clear (INTFLAGAHB) register informs the user on the status of the violation in the corresponding client. Refer to the *AHB Client Bus Errors* section for details.

Write Control (WRCTRL) register in PAC module has no effect for the peripherals which are write-protected by Local Write-Protection set in the peripheral's Write Protection Control (WPCTRL) register.

17.4.2.1. Peripheral Access Errors

The following events will generate a Peripheral Access Error:

- Protected write: To avoid unexpected writes to a peripheral's registers, each peripheral can be write-protected. Only the registers denoted as "PAC Write-Protection" or Local Write-Protection in the module's data sheet can be protected. If a peripheral is not write-protected, write data accesses are performed normally. If a peripheral is write-protected and if a write access is attempted, data will not be written and the peripheral returns an access error. The corresponding interrupt flag bit in the INTFLAGn register will be set.
- Illegal access: Access to an unimplemented register within the peripheral except PORT, MTB, EIC, HMATRIX
- Synchronized write error: For write-synchronized registers, an error will be reported if the register is written while a synchronization is ongoing

When any of the Peripheral Interrupt Flag Status and Clear n (INTFLAGn) registers bit are set, an interrupt will be requested if the Peripheral Access Error Interrupt Enable (INTENCLR.ERR) or Peripheral Access Error Interrupt Disable (INTENSET.ERR) bit is set.

17.4.2.2. Write Access Protection Management

Peripheral access control can be enabled or disabled by writing to the Write Control (WRCTRL) register.

The data written to the WRCTRL register is composed of two fields: WRCTRL.PERID and WRCTRL.KEY. The WRCTRL.PERID is an unique identifier corresponding to a peripheral. The WRCTRL.KEY is a key value that defines the operation to be done on the control access bit. These operations can be "clear protection", "set protection", and "set and lock protection bit".

The "clear protection" operation will remove the write access protection for the peripheral selected by WRCTRL.PERID. Write accesses are allowed for the registers in this peripheral.

The "set protection" operation will set the write access protection for the peripheral selected by WRCTRL.PERID. Write accesses are not allowed for the registers with write protection property in this peripheral.

The "set and lock protection" operation will set the write access protection for the peripheral selected by WRCTRL.PERID and locks the access rights of the selected peripheral registers. The write access protection will only be cleared by a hardware reset.

The peripheral access control status can be read from the corresponding Peripheral Write Protection Status (STATUSn) register.

17.4.2.3. Write Access Protection Management Errors

Only word-wise writes to the Write Control (WRCTRL) register will effectively change the access protection. Other type of accesses will have no effect and will cause a PAC write access error. This error is reported in the Peripheral Interrupt Flag Status and Clear (INTFLAGA.PAC) bit.

PAC also offers an additional safety feature for correct program execution with an interrupt generated on double write clear protection or double write set protection. If a peripheral is write-

protected and a subsequent set protection operation is detected, then the PAC returns an error, and similarly for a double clear protection operation.

In addition, an error is generated when writing a “set and lock” protection to a write-protected peripheral or when a write access is done to a locked set protection. This can be used to ensure that the application follows the intended program flow by always following a write-protect with an unprotect and the other way around. However, in applications where a write-protected peripheral is used in several contexts, e.g. interrupt, care must be taken so that either the interrupt can not happen while the main application or other interrupt levels manipulates the write protection status or when the interrupt handler needs to unprotect the peripheral based on the current protection status by reading the Peripheral Write Protection Status (STATUSn) register.

The errors generated while accessing the PAC module registers (for example, key error, double protect error, and so on) will set the Peripheral Interrupt Flag Status and Clear (INTFLAGA.PAC) bit.

17.4.2.4. AHB Client Bus Errors

The PAC module reports errors occurring at the AHB Client bus level. These errors are generated when an access is performed at an address where no client (bridge or peripheral) is mapped. These errors are reported in the corresponding bits of the Peripheral Interrupt Flag Status and Clear B (INTFLAGAHB) register.

17.4.3. Sleep Mode Operation

In sleep mode, the PAC is kept enabled if an available bus host (CPU, DMA) is running. The PAC will continue to catch access errors from the module and generate interrupts or events.

17.4.4. Debug Operation

When the CPU is halted in Debug mode, write protection of all peripherals is disabled and the PAC continues to operate normally.

17.5. Dependencies

17.5.1. I/O Lines

Not applicable.

17.5.2. Power Management

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

17.5.3. Clocks

The PAC bus clock (CLK_PAC_APB) can be enabled and disabled in the Main Clock module. The default state of CLK_PAC_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

17.5.4. DMA

Not applicable.

17.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use PAC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear registers listed below:

- AHB Client Bus Interrupt Flag Status and Clear (INTFLAGAHB) register
- Peripheral Interrupt Flag Status and Clear A (INTFLAGA) register
- Peripheral Interrupt Flag Status and Clear B(INTFLAGB) register
- Peripheral Interrupt Flag Status and Clear C(INTFLAGC) register

The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAGAHB, INTFLAGA, INTFLAGB and INTFLAGC register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 17-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
PAC	FLASH	Access Error is detected by client Flash	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	HSRAMCM0P	Access Error is detected by client SRAMCM0P	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	HSRAMDSU	Access Error is detected by client SRAMDSU	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	APBB	Access Error is detected by client APBB	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	APBA	Access Error is detected by client APBA	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	APBC	Access Error is detected by client APBC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	SRAMDMAC	Access Error is detected by client SRAMDMAC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	BROM	Access Error is detected by client BROM	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	PAC	Peripheral Access Error occurs while accessing PAC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	PM	Peripheral Access Error occurs while accessing PM	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	MCLK	Peripheral Access Error occurs while accessing MCLK	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	RSTC	Peripheral Access Error occurs while accessing RSTC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	OSCCTRL	Peripheral Access Error occurs while accessing OSCCTRL	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	SUPC	Peripheral Access Error occurs while accessing SUPC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	GCLK	Peripheral Access Error occurs while accessing GCLK	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	WDT	Peripheral Access Error occurs while accessing WDT	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	RTC	Peripheral Access Error occurs while accessing RTC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set

Table 17-1. Available Interrupt Vectors and Sources (continued)

Vector Name	Source Name	Condition	Dependency
	EIC	Peripheral Access Error occurs while accessing EIC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	PORT	Peripheral Access Error occurs while accessing PORT	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	DSU	Peripheral Access Error occurs while accessing DSU	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	NVMCTRL	Peripheral Access Error occurs while accessing NVMCTRL	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	DMAC	Peripheral Access Error occurs while accessing DMAC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	MTB	Peripheral Access Error occurs while accessing MTB	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	HMATRIXHS	Peripheral Access Error occurs while accessing HMATRIXHS	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	EVSYS	Peripheral Access Error occurs while accessing EVSYS	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	SERCOMn	Peripheral Access Error occurs while accessing SERCOMn	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	TCn	Peripheral Access Error occurs while accessing TCn	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	TCCn	Peripheral Access Error occurs while accessing TCCn	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	ADC0	Peripheral Access Error occurs while accessing ADC0	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	AC	Peripheral Access Error occurs while accessing AC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	CCL	Peripheral Access Error occurs while accessing CCL	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	PTC	Peripheral Access Error occurs while accessing PTC	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	SYSCTRL	Peripheral Access Error occurs while accessing SYSCTRL	The Peripheral Access Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set

17.5.6. Events

Table 17-2. Generated Events

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
PAC	ACCERR	Access Error	Level	CLK_PAC_APB	As long as a interrupt flag is set

Event is generated when one of the interrupt flag register's bit is set. Writing a '1' to an Event Output bit in the Event Control Register (EVCTRL.ERREO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

17.5.7. Analog Connections

Not applicable.

17.6. Register Summary - PAC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	WRCTRL	7:0	PERID[7:0]							
		15:8	PERID[15:8]							
		23:16	KEY[7:0]							
		31:24								
0x04	EVCTRL	7:0								ERREO
0x05 ... 0x07	Reserved									
0x08	INTENCLR	7:0								ERR
0x09	INTENSET	7:0								ERR
0x0A ... 0x0F	Reserved									
0x10	INTFLAGAHB	7:0	BROM	SRAMDMAC	APBC	APBA	APBB	HSRAMDSU	HSRAMCM0P	FLASH
		15:8								
		23:16								
		31:24								
0x14	INTFLAGA	7:0	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
		15:8						EIC	RTC	WDT
		23:16								
		31:24								
0x18	INTFLAGB	7:0			HMATRIXHS	MTB	DMAC	NVMCTRL	DSU	PORT
		15:8								
		23:16								
		31:24								
0x1C	INTFLAGC	7:0	ADC0	TCC0	TC2	TC1	TC0	SERCOM1	SERCOM0	EVSYS
		15:8					SYSCTRL	PTC	CCL	AC
		23:16								
		31:24								
0x20 ... 0x33	Reserved									
0x34	STATUSA	7:0	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
		15:8						EIC	RTC	WDT
		23:16								
		31:24								
0x38	STATUSB	7:0			HMATRIXHS	MTB	DMAC	NVMCTRL	DSU	PORT
		15:8								
		23:16								
		31:24								
0x3C	STATUSC	7:0	ADC0	TCC0	TC2	TC1	TC0	SERCOM1	SERCOM0	EVSYS
		15:8					SYSCTRL	PTC	CCL	AC
		23:16								
		31:24								

17.6.1. Write Control

Name: WRCTRL
Offset: 0x00
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	KEY[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
	PERID[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
	PERID[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 23:16 – KEY[7:0] Peripheral Access Control Key
This field defines the peripheral access control key.

Value	Name	Description
0x0	OFF	No action
0x1	CLR	Clear the peripheral write-control
0x2	SET	Set the peripheral write-control
0x3	SETLCK	Set and lock the peripheral write-control until the next hardware reset

Bits 15:0 – PERID[15:0] Peripheral Identifier
This field represents the peripheral whose access control is changed using WRCTRL.KEY.

Value	Name	Description
0x00	PAC	APBA peripheral
0x01	PM	APBA peripheral
0x02	MCLK	APBA peripheral
0x03	RSTC	APBA peripheral
0x04	OSCCTRL	APBA peripheral
0x05	OSC32KCTRL	APBA peripheral
0x06	SUPC	APBA peripheral
0x07	GCLK	APBA peripheral
0x08	WDT	APBA peripheral
0x09	RTC	APBA peripheral
0x0A	EIC	APBA peripheral

Value	Name	Description
0x20	PORT	APBB peripheral **Notes:** 1. IOBUS writes are not prevented to PAC write-protected registers when the PORT module is PAC protected. 2. PORT read/write attempts on non-implemented registers, including addresses beyond the last implemented register group do not generate a PAC protection error.
0x21	DSU	APBB peripheral
0x22	NVMCTRL	APBB peripheral
0x23	DMAC	APBB peripheral
0x24	MTB	APBB peripheral
0x25	HMATRIXHS	APBB peripheral
0x40	EVSYS	APBC peripheral
0x41	SERCOM0	APBC peripheral
0x42	SERCOM1	APBC peripheral
0x43	TC0	APBC peripheral
0x44	TC1	APBC peripheral
0x45	TC2	APBC peripheral
0x46	TCC0	APBC peripheral
0x47	ADC0	APBC peripheral
0x48	AC	APBC peripheral
0x49	CCL	APBC peripheral
0x4A	PTC	APBC peripheral
0x4B	SYSCTRL	APBC peripheral
Other	—	Reserved

17.6.2. Event Control

Name: EVCTRL
Offset: 0x04
Reset: 0x00
Property: –

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

Bit 0 – ERREO Peripheral Access Error Event Output
This bit indicates if the Peripheral Access Error Event Output is enabled or not. When enabled, an event will be generated when one of the Interrupt Flag register (INTFLAGAHB, INTFLAGx) bits is set.

Value	Description
0	The Peripheral Access Error Event Output is disabled
1	The Peripheral Access Error Event Output is enabled

17.6.3. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x08
Reset: 0x00
Property: PAC Write-Protection

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.

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

Bit 0 – ERR Peripheral Access Error Interrupt Disable
This bit indicates that the Peripheral Access Error Interrupt is enabled and an interrupt request will be generated when one of the Interrupt Flag register (INTFLAGAHB, INTFLAGx) bits is set.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Peripheral Access Error interrupt Enable bit, which disables the Peripheral Access Error interrupt.

Value	Description
0	The Peripheral Access Error interrupt is enabled
1	The Peripheral Access Error interrupt is disabled

17.6.4. Interrupt Enable Set

Name: INTENSET
Offset: 0x09
Reset: 0x00
Property: PAC Write-Protection

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.

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

Bit 0 – ERR Peripheral Access Error Interrupt Enable
This bit indicates that the Peripheral Access Error Interrupt is enabled and an interrupt request will be generated when one of the Interrupt Flag register (INTFLAGAHB, INTFLAGx) bits is set.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Peripheral Access Error interrupt Enable bit, which enables the Peripheral Access Error interrupt.

Value	Description
0	The Peripheral Access Error interrupt is disabled
1	The Peripheral Access Error interrupt is enabled

17.6.5. AHB Client Bus Interrupt Flag Status and Clear

Name: INTFLAGAHB
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	BROM	SRAMDMAC	APBC	APBA	APBB	HSRAMDSU	HSRAMCM0P	FLASH
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit 7 – BROM CLIENT Boot ROM Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 6 – SRAMDMAC CLIENT SRAMDMAC Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 5 – APBC CLIENT APBC Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 4 – APBA CLIENT APBA Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.

Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 3 – APBB CLIENT APBB Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 2 – HSRAMDSU CLIENT SRAMDSU Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 1 – HSRAMCM0P CLIENT SRAMCM0P Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 0 – FLASH CLIENT FLASH Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when an access error is detected by the client n, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

17.6.6. Peripheral Interrupt Flag Status and Clear A

Name: INTFLAGA
Offset: 0x14
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						EIC	RTC	WDT
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
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 10 – EIC EIC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 9 – RTC RTC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 8 – WDT WDT Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 7 – GCLK GCLK Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.

Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 6 – SUPC SUPC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 5 – OSC32KCTRL OSC32KCTRL Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 4 – OSCCTRL OSCCTRL Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 3 – RSTC RSTC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 2 – MCLK MCLK Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 1 – PM PM Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 0 – PAC PAC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGA bit, and will generate an interrupt request if INTENCLR/SET.ERR is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

17.6.7. Peripheral Interrupt Flag Status and Clear B

Name: INTFLAGB
Offset: 0x18
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 5 – HMATRIXHS HMATRIXHS Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGB bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 4 – MTB MTB Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGB bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 3 – DMAC DMAC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGB bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 2 – NVMCTRL NVMCTRL Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.

A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGB bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 1 – DSU DSU Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGB bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 0 – PORT PORT Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGB bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

17.6.8. Peripheral Interrupt Flag Status and Clear C

Name: INTFLAGC
Offset: 0x1C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					SYSCTRL	PTC	CCL	AC
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADC0	TCC0	TC2	TC1	TC0	SERCOM1	SERCOM0	EVSYS
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 11 – SYSCTRL SYSCTRL Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 10 – PTC PTC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 9 – CCL CCL Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 8 – AC AC Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.

A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 7 – ADC0 ADC0 Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 6 – TCCn TCCn Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bits 3, 4, 5 – TCn TCn Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bits 1, 2 – SERCOMn SERCOMn Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

Bit 0 – EVSYS EVSYS Peripheral Interrupt Flag
A flag is cleared by writing a '1' to it.
A flag is set when a Peripheral Access Error occurs while accessing the peripheral associated with the respective INTFLAGC bit, and will generate an interrupt request if Interrupt Enable Set (INTENSET.ERR) bit or Interrupt Enable Clear (INTENCLR.ERR) bit is '1'.
Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding interrupt flag.

17.6.9. Peripheral Write Protection Status A

Name: STATUSA
Offset: 0x34
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

Bit	7	6	5	4	3	2	1	0
	GCLK	SUPC	OSC32KCTRL	OSCCTRL	RSTC	MCLK	PM	PAC
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 10 – EIC EIC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 9 – RTC RTC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 8 – WDT WDT Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 7 – GCLK GCLK Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 6 – SUPC SUPC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 5 – OSC32KCTRL OSC32KCTRL Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 4 – OSCCTRL OSCCTRL Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 3 – RSTC RSTC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 2 – MCLK MCLK Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 1 – PM PM Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

Bit 0 – PAC PAC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	The peripheral is not write-protected
1	The peripheral is write-protected

17.6.10. Peripheral Write Protection Status B

Name: STATUSB
Offset: 0x38
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			HMATRIXHS	MTB	DMAC	NVMCTRL	DSU	PORT
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – HMATRIXHS HMATRIXHS Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected.
1	Peripheral is write-protected.

Bit 4 – MTB MTB Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected.
1	Peripheral is write-protected.

Bit 3 – DMAC DMAC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected.
1	Peripheral is write-protected.

Bit 2 – NVMCTRL NVMCTRL Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected.
1	Peripheral is write-protected.

Bit 1 – DSU DSU Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected.
1	Peripheral is write-protected.

Bit 0 – PORT PORT Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected.
1	Peripheral is write-protected.

17.6.11. Peripheral Write Protection Status C

Name: STATUSC
Offset: 0x3C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					SYSCTRL	PTC	CCL	AC
Access					R	R	R	R
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADC0	TCC0	TC2	TC1	TC0	SERCOM1	SERCOM0	EVSYS
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 11 – SYSCTRL SYSCTRL Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bit 10 – PTC PTC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bit 9 – CCL CCL Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bit 8 – AC AC Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bit 7 – ADC0 ADC0 Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bit 6 – TCCn TCCn Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bits 3, 4, 5 – TCn TCn Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bits 1, 2 – SERCOMn SERCOMn Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

Bit 0 – EVSYS EVSYS Peripheral Write Protection Status
A bit indicates if the corresponding peripheral is write-protected or not.

Value	Description
0	Peripheral is not write-protected
1	Peripheral is write-protected

18. DSU – Device Service Unit

18.1. Features

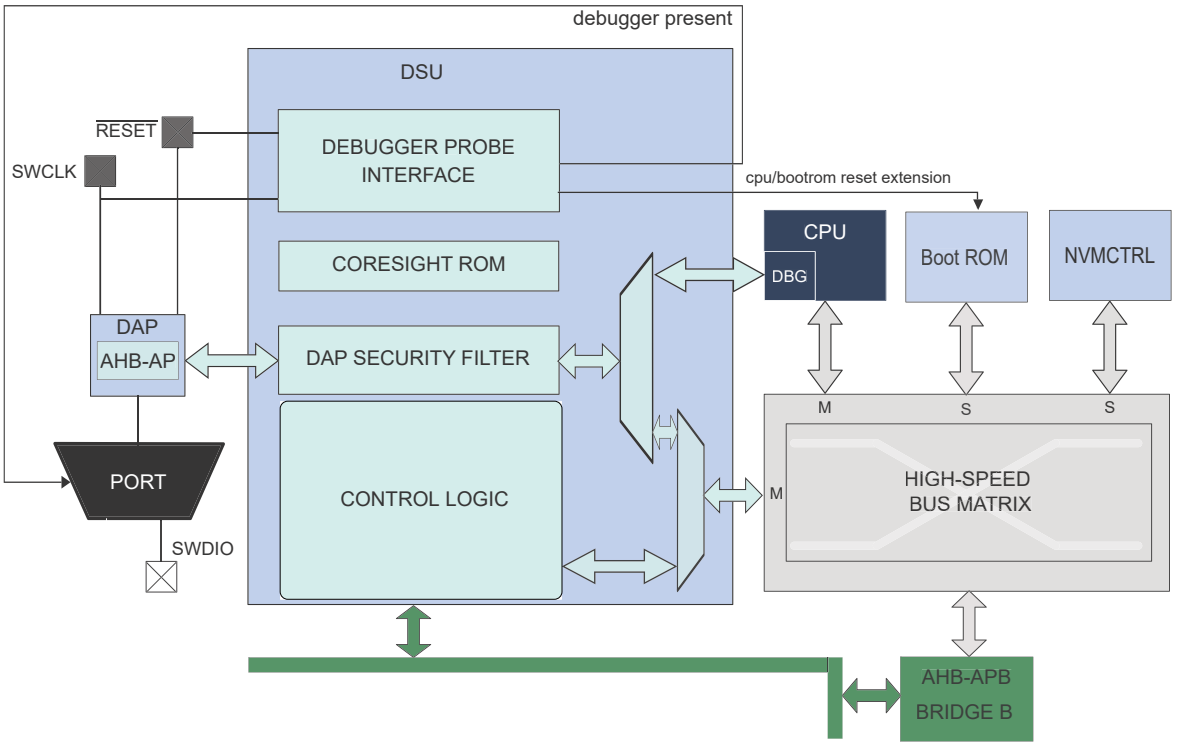
- Arm® CoreSight™ Compliant Device Identification
- Microchip Device Identification
- CPU Reset Extension
- Debugger Probe Detection (Cold- and Hot-plugging)
- Two Debug Communication Channels with DMA Support
- Two Boot Communication Channels
- Support for Debug Authentication

18.2. Overview

The Device Service Unit (DSU) provides multiple functions, including debugger access restrictions and Microchip device identification. The DSU is responsible for enabling, disabling, or restricting plugging detection and debugger access to the device. It also provides the Arm® CoreSight™ Debug ROM table and a communication mailbox for ROM services.

18.3. Block Diagram

Figure 18-1. DSU Block Diagram



18.3.1. Signal Description

The DSU uses the following three signals to function.

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

18.4. Functional Description

18.4.1. Initialization

The module is enabled by enabling its clock signals.

18.4.2. Operation

18.4.2.1. Operation From a Debug Adapter

Debug adapters access the DSU registers in the external address range `0x100 – 0x1FFF`. If the device is protected, accessing the first `0x100` bytes causes the system to return an error. Refer to the *Intellectual Property Protection* section.

18.4.2.2. Operation From the CPU

There are no restrictions when accessing DSU registers from the CPU. However, to avoid external security restrictions, the user may access DSU registers in the internal address range (0x0 – 0x100). Refer to the *Intellectual Property Protection* section.

18.4.2.3. Programming

The programming procedure for the Flash and SRAM memories is as follows:

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). The system remains in this static state until the internally regulated supplies have reached a safe operating level.
2. The Power Manager (PM) starts, and the 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. $\overline{\text{RESET}}$ is released, resulting in a debugger cold-plugging procedure.
4. The debugger generates a clock signal on the SWCLK pin, and 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 before programming.
7. Programming is available through the AHB-AP.
8. After the operation is completed, the chip can be restarted either by asserting $\overline{\text{RESET}}$ or toggling power. Ensure that the SWCLK pin is high when releasing $\overline{\text{RESET}}$ to prevent extending the CPU reset.

18.4.3. Additional Features

18.4.3.1. Debug Communication Channels

The Debug Communication Channels (DCCO and DCC1) consist of a pair of registers with associated handshake logic, accessible by both the CPU and the debugger with no security restrictions. The registers can exchange data between the CPU and the debugger during run time and 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. However, it is impossible to connect a debugger while the CPU is running (STATUSA.CRSTEXT0 is not writable, and the CPU is held in Reset) when the device is protected.

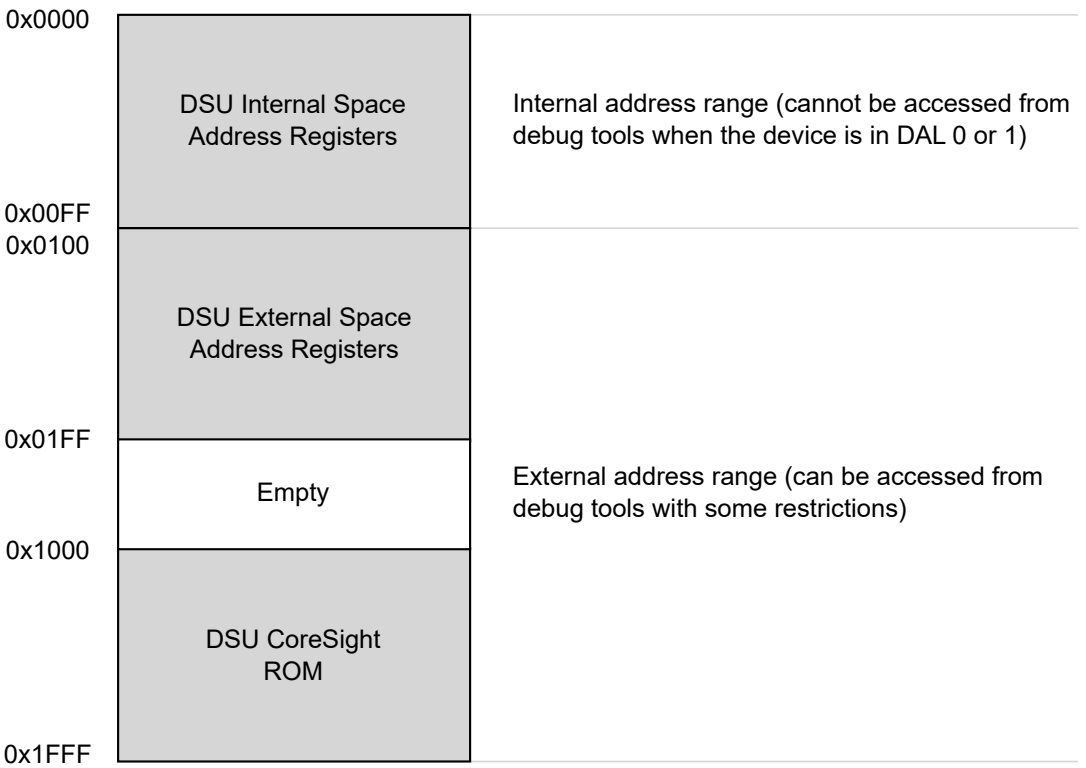
Two Debug Communication Channel status bits in the Status B register (STATUSB.DCCDn) indicate whether a new value has been written in DCC0 or DCC1. These bits, DCCD0 and DCCD1, are located in the STATUSB register. They are automatically set on write and cleared on read.

18.4.3.2. Intellectual Property Protection

The DSU provides the Debug Access Level (DAL) mechanism, which disables or limits debugger access to the device. Refer to the *Debugger Access Level (DAL)* section in the *BootROM - Boot ROM* chapter for more information about how the DAL value is determined.

If the debugger attempts to access a device region that is not permitted by the current DAL setting, the DAP Security filter will block the access and return an error to the Arm AHB-AP hardware, resulting in the assertion of the ARM DP sticky error bit. The DAP Security filter will also block access during system initialization, so a debugger cannot perform unauthorized access until the boot ROM has had time to read and apply the DAL setting.

Figure 18-2. APB Memory Mapping



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

Table 18-1. Feature Availability Under Protection

Features	Availability when DAL is not 2
CPU Reset Extension	Yes
Clear CPU Reset Extension	Yes
Debugger Cold-Plugging	Yes

Table 18-1. Feature Availability Under Protection (continued)	
Features	**Availability when DAL is not 2**
Debugger Hot-Plugging	No

18.4.3.3. Device Identification

The DSU provides access to the Arm CoreSight Identification registers and the Microchip Device ID register, both of which are accessible via the debugger and APB interfaces.

18.4.3.3.1. 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 18-3. Conceptual 64-bit Peripheral ID

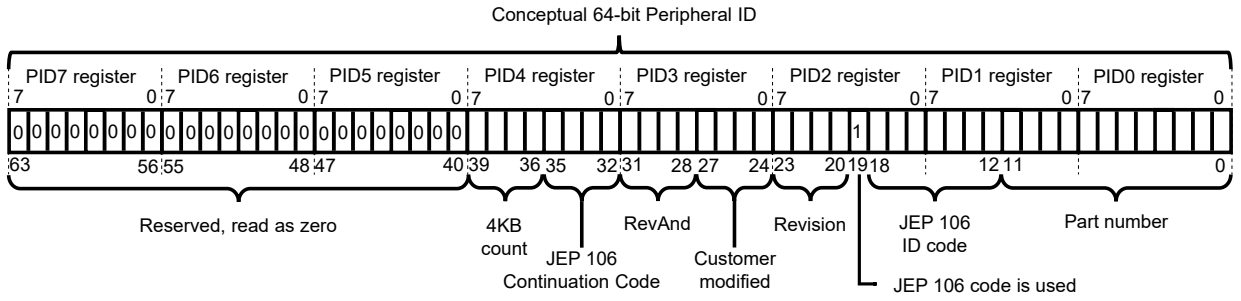


Table 18-2. Conceptual 64-Bit Peripheral ID Bit Descriptions

Field	Size	Description	Location
JEP-106 CC code	4	Microchip Continuation code: 0x0	PID4
JEP-106 ID code	7	Microchip Device ID: 0x29	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 which starts at 0x0 and increments by 1 for each version of the DSU	PID2

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

18.4.3.3.2. Chip Identification Method

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

- Processor identification
- Product family identification
- Product series identification
- Device select

18.4.4. Sleep Mode Operation

The DSU will continue to operate in any sleep mode in which its bus clocks are running.

Refer to the *PM - Power Manager* chapter for more information about sleep mode operation.

18.4.5. Debug Operation

18.4.5.1. Principle of Operation

The DSU monitors the SWCLK and $\overline{\text{RESET}}$ pins on the device to determine whether a debugger is connected and whether a debug session can be initiated. The DSU supports debugger detection during Reset (cold-plugging) and when the device is out of Reset (hot-plugging).

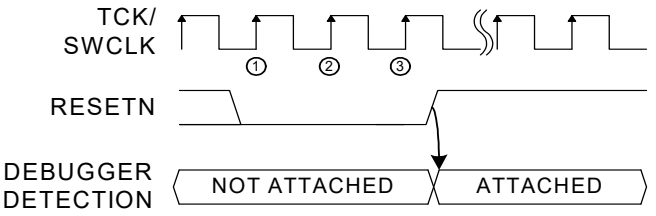
18.4.5.2. Debugger Probe Detection

18.4.5.2.1. Cold-Plugging

Cold-plugging is the detection of a debugger when the system is reset. Cold-plugging detection occurs when the external Reset is asserted for three consecutive SWCLK rising edges, as shown in the figure.

As described below, a CPU Reset extension is requested when cold-plugging is detected.

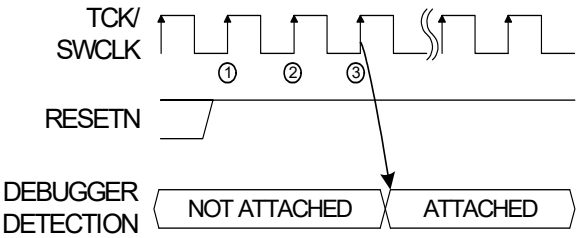
Figure 18-4. Cold-Plugging Detection Timing Diagram



18.4.5.2.2. Hot-Plugging

Hot-plugging is the detection of a debugger probe when the system is not reset. Hot-plugging detection occurs when the external Reset pin is not asserted and at least three SWCLK rising edges are detected, as shown in the figure. A hot-plugging debug session terminates on a Brown-out Reset, including an assertion of the external Reset. Hot-plugging is only possible if the device PORTMUX selects the SWCLK function. If the SWCLK function is changed, the hot-plugging feature is disabled until a power Reset or external Reset occurs. The availability of the hot-plugging feature can be read from the Hot-Plugging Status bit of the Status B (STATUSB.HPS) register.

Figure 18-5. Hot-Plugging Detection Timing Diagram



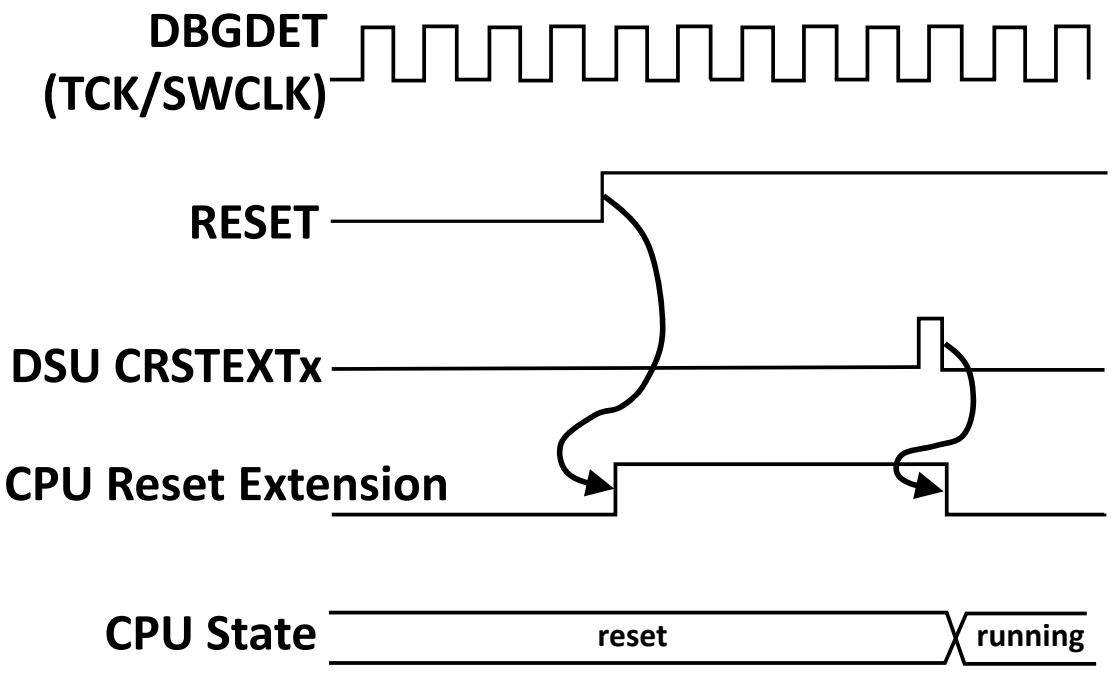
The presence of a debugger probe is detected when either hot-plugging or cold-plugging is detected. Once detected, the Debugger Present bit of the Status B (STATUSB.DBGPRES) register is set.

18.4.5.2.3. CPU Reset Extension

On cold-plugging attachment, the DSU sets the STATUSA.CRSTEXT0 bit, which forces the CPU into a Reset state until the bit is cleared. This ensures that the CPU is not executing code and is in a known state when a debugger connects.

To release the CPU from reset, the debugger writes a '1' to clear STATUSA.CRSTEXT0. On Reset extension release, CPU0 starts executing instructions.

Figure 18-6. Typical CPU Reset Extension Set and Clear Timing Diagram



18.5. Dependencies

18.5.1. I/O Lines

Using the DSU's I/O lines requires the I/O pins to be configured. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

18.5.2. Power Management

The DSU will continue to operate in any sleep mode in which its bus clocks are running. Refer to the *PM – Power Manager* chapter for details on the different sleep modes.

18.5.3. Clocks

The DSU bus clocks (CLK_DSU_AHB and CLK_DSU_APB) can be enabled and disabled in the *MCLK - Main Clock controller*. By default, these clocks are enabled, so the DSU will be operational.

18.5.4. DMA

The DSU generates the following DMA request:

- Debug Communication Channel n (DCCn): The request is set when the DCCn register is read. The request is cleared when the DCCn register is written. This behavior can be inverted by setting the DMA Trigger Level bit in the Control B (CTRLB.DCCDMALVLn) register.

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the DSU's DMA requests.

Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

18.5.5. Interrupts

Not applicable.

18.5.6. Events

Not applicable.

18.5.7. Analog Connections

Not applicable.

18.6. Register Summary - DSU

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	CTRLB	7:0					QOS[1:0]		DCCDMALVL1	DCCDMALVL0
		15:8								
		23:16								
		31:24								
0x08	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x0C ... 0xFF	Reserved									
0x0100	CTRLC	7:0								DBGRW0
		15:8								
		23:16								
		31:24								
0x0104	STATUSA	7:0								DONE
		15:8								CRSTEXT0
		23:16								BREXT0
		31:24								
0x0108	STATUSB	7:0					DCCD1	DCCD0	BCCD1	BCCD0
		15:8						HPS		DBGPRES
		23:16								
		31:24								
0x010C ... 0x010F	Reserved									
0x0110	BCC[0]	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0x0114	BCC[1]	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0x0118	DCC[0]	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0x011C	DCC[1]	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0x0120	DID	7:0	MANID[6:0]							FV
		15:8	PNDID[3:0]				MANID[10:7]			
		23:16	PNMID[3:0]				PNDID[7:4]			
		31:24	VER[3:0]				PNMID[7:4]			
0x0124	DAL	7:0							CPU0[1:0]	
		15:8								
		23:16								
		31:24								
0x0128 ... 0x0FFF	Reserved									

DSU (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x1000	ENTRY0	7:0							FMT	EPRES
		15:8	ADDOFF[3:0]							
		23:16	ADDOFF[11:4]							
		31:24	ADDOFF[19:12]							
0x1004	ENTRY1	7:0							FMT	EPRES
		15:8	ADDOFF[3:0]							
		23:16	ADDOFF[11:4]							
		31:24	ADDOFF[19:12]							
0x1008	ENTRY2	7:0							FMT	EPRES
		15:8	ADDOFF[3:0]							
		23:16	ADDOFF[11:4]							
		31:24	ADDOFF[19:12]							
0x100C	ENTRY3	7:0							FMT	EPRES
		15:8	ADDOFF[3:0]							
		23:16	ADDOFF[11:4]							
		31:24	ADDOFF[19:12]							
0x1010 ... 0x1FCB	Reserved									
0x1FCC	MEMTYPE	7:0								SYSMEM
		15:8								
		23:16								
		31:24								
0x1FD0	PID4	7:0	SIZE[3:0]				DES2[3:0]			
		15:8								
		23:16								
		31:24								
0x1FD4	PID5	7:0	RES0[7:0]							
		15:8								
		23:16								
		31:24								
0x1FD8	PID6	7:0	RES0[7:0]							
		15:8								
		23:16								
		31:24								
0x1FDC	PID7	7:0	RES0[7:0]							
		15:8								
		23:16								
		31:24								
0x1FE0	PID0	7:0	PART0[7:0]							
		15:8								
		23:16								
		31:24								
0x1FE4	PID1	7:0	DES0[3:0]				PART1[3:0]			
		15:8								
		23:16								
		31:24								
0x1FE8	PID2	7:0	REVISION[3:0]				JEDEC	DES1[2:0]		
		15:8								
		23:16								
		31:24								
0x1FEC	PID3	7:0	REVAND[3:0]				CMOD[3:0]			
		15:8								
		23:16								
		31:24								
0x1FF0	CID0	7:0	PRMBL0[7:0]							
		15:8								
		23:16								
		31:24								

DSU (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x1FF4	CID1	7:0	CCLASS[3:0]				PRMBL1[3:0]			
		15:8								
		23:16								
		31:24								
0x1FF8	CID2	7:0	PRMBL2[7:0]							
		15:8								
		23:16								
		31:24								
0x1FFC	CID3	7:0	PRMBL3[7:0]							
		15:8								
		23:16								
		31:24								

18.6.1. Control B

Name: CTRLB
Offset: 0x0004
Reset: 0x00000004
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					QOS[1:0]		DCCDMALVL1	DCCDMALVL0
Access					R/W	R/W	R/W	R/W
Reset					0	1	0	0

Bits 3:2 – QOS[1:0] DSU QoS Level

Value	Name	Description
0x0	DISABLE	Background (no sensitive operation)
0x1	LOW	Sensitive bandwidth
0x2	MEDIUM	Sensitive latency
0x3	HIGH	Critical latency

Bits 0, 1 – DCCDMALVLn DMA Trigger n Level

Value	Name	Description
0	EMPTY	Trigger n rises when DCCn is read and falls when it is written
1	FULL	Trigger n rises when DCCn is written and falls when it is read

18.6.2. Write Protection Control

Name: WPCTRL
Offset: 0x0008
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write to be successful.

Value	Name	Description
0x445355	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

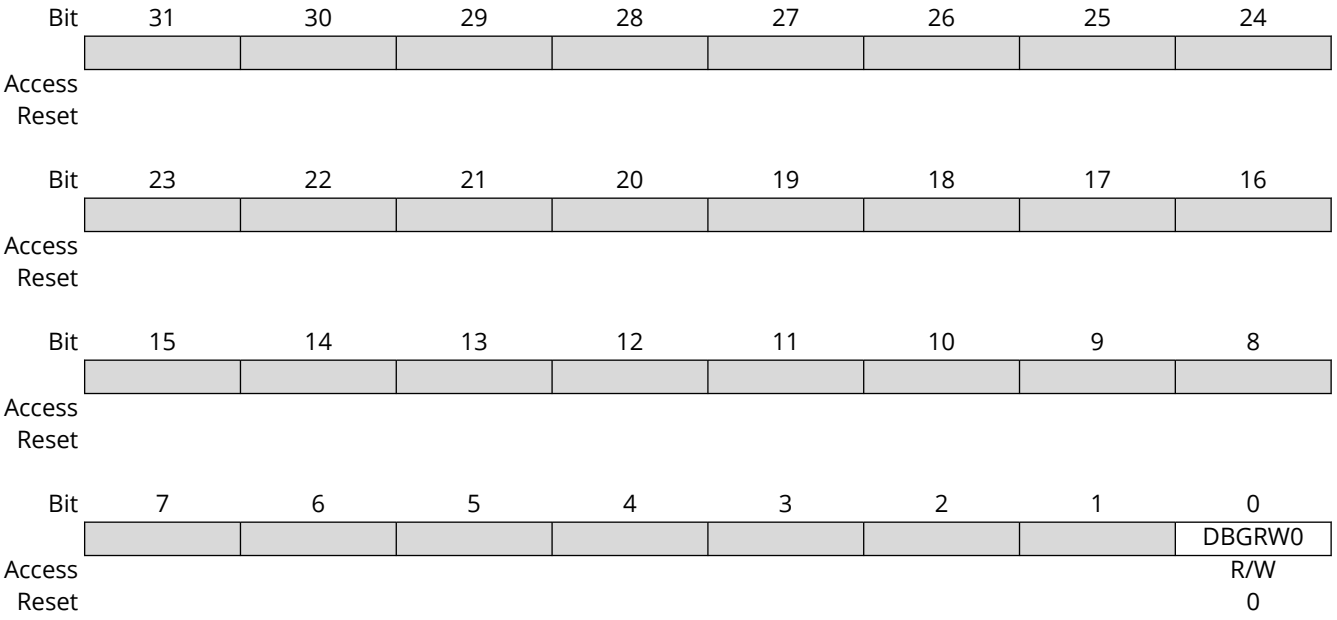
Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	DSU register write protection is disabled
1	Write protection is enabled on DSU registers with the Local Write-Protection property. Non-debugger writes to DSU registers with Local Write-Protection are ignored and generate a bus error.

18.6.3. Control C

Name: CTRLC
Offset: 0x0100
Reset: 0x00000000
Property: Local Write-Protection



Bit 0 – DBGRW0 CPU0 Debug R/W

Value	Description
0	Allow debugger accesses through CPU0 to carry the debug attribute (i.e., debugger reads are non-destructive)
1	Debugger accesses through CPU0 do not carry the debug attribute (i.e., debugger reads are destructive)

18.6.4. Status A

Name: STATUSA
Offset: 0x0104
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
								BREXT0
Access								R
Reset								0

Bit	15	14	13	12	11	10	9	8
								CRSTEXT0
Access								R
Reset								0

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

Bit 16 – BREXT0 Boot ROM 0 Phase Extension Status

Value	Description
0	Boot ROM Phase Extension released for CPU0
1	Boot ROM Phase Extension for CPU0. Set on cold-plugging detection.

Bit 8 – CRSTEXT0 CPU0 Reset Extension Status

Value	Description
0	CPU0 released from reset extension
1	CPU0 held in reset extension. Set on cold-plugging detection.

Bit 0 – DONE Done Status

Value	Description
0	DSU is executing a command or idle
1	Command completed

18.6.5. Status B

Name: STATUSB
Offset: 0x0108
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

Bit	7	6	5	4	3	2	1	0
					DCCD1	DCCD0	BCCD1	BCCD0
Access					R	R	R	R
Reset					0	0	0	0

Bit 10 – HPS Hot-plugging Status

Value	Description
0	Hot-plugging inactive
1	Hot-plugging is available

Bit 8 – DBGPRES Debugger Present Status

Value	Description
0	No debugger detected
1	Debugger probe detected

Bits 2, 3 – DCCDn Debug Communication Channel n Dirty

Value	Description
0	DCCDn.DATA register was read from
1	DCCDn.DATA register was written to

Bits 0, 1 – BCCDn Boot ROM Communication Channel n Dirty

Value	Description
0	BCCDn.DATA register was read from
1	BCCDn.DATA register was written to

18.6.6. Boot ROM Channel n

Name: BCC[n]
Offset: 0x0110 + n*0x04 [n=0..1]
Reset: 0xXXXXXXXX
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	x	x	x	x	x	x	x	x

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	x	x	x	x	x	x	x	x

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	x	x	x	x	x	x	x	x

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	x	x	x	x	x	x	x	x

Bits 31:0 – DATA[31:0] Data
This is a Data register used for ROM commands and responses. Writing to the Data register sets `STATUSB.BCCDn`. Reading from the Data register clears `STATUSB.BCCDn`.

18.6.7. Debug Communication Channel n

Name: DCC[n]
Offset: 0x0118 + n*0x04 [n=0..1]
Reset: 0xXXXXXXXX
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	x	x	x	x	x	x	x	x

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	x	x	x	x	x	x	x	x

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	x	x	x	x	x	x	x	x

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	x	x	x	x	x	x	x	x

Bits 31:0 – DATA[31:0] Data
Data register used for debug communication. Writing to the register sets `STATUSB.DCCDn`; reading from the register clears `STATUSB.DCCDn`.

18.6.8. Device Identification

Name: DID
Offset: 0x0120
Reset: 0x0BAXX053
Property: –

Bit	31	30	29	28	27	26	25	24
	VER[3:0]				PNMID[7:4]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	0	1	1

Bit	23	22	21	20	19	18	17	16
	PNMID[3:0]				PNDID[7:4]			
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	0	x	x	x	x

Bit	15	14	13	12	11	10	9	8
	PNDID[3:0]				MANID[10:7]			
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
	MANID[6:0]							FV
Access	R	R	R	R	R	R	R	R
Reset	0	1	0	1	0	0	1	1

Bits 31:28 – VER[3:0] Version Code
Provides the version code where a value of '0' represents A.0.

Bits 27:20 – PNMID[7:0] Part Number Mask ID
Provides bit field [15:8] of the Device Part Number, where the entire Part Number is given by PN[15:0] = DID[27:12].

Bits 19:12 – PNDID[7:0] Part Number Device ID
Provides bit field [7:0] of the Device Part Number, where the entire Part Number is given by PN[15:0] = DID[27:12].

Bits 11:1 – MANID[10:0] JEDEC Manufacture ID
This field contains the value defined in JEDEC Publication JEP106Q for Microchip Technology Inc., which is 0x29.

Bit 0 – FV Fixed Value of 1

18.6.9. Debugger Access Level

Name: DAL
Offset: 0x0124
Reset: 0xXXXXXXXX
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							CPU0[1:0]	
Access							R	R
Reset							x	x

Bits 1:0 – CPU0[1:0] CPU 0 Debugger Access Level

Value	Name	Description
0x0	SECURED	The Debugger targeting the CPU0 domain can only access the DSU external address space; otherwise, debugger access is disabled.
0x1	—	Reserved
0x2	FULL_DEBUG	The Debugger has unrestricted access
0x3	—	Reserved

18.6.10. CoreSight ROM Table Entry 0

Name: ENTRY0
Offset: 0x1000
Reset: 0x9F0FC003
Property: –

Bit	31	30	29	28	27	26	25	24
	ADDOFF[19:12]							
Access	R	R	R	R	R	R	R	R
Reset	1	0	0	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	ADDOFF[11:4]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	ADDOFF[3:0]							
Access	R	R	R	R				
Reset	1	1	0	0				

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

Bits 31:12 – ADDOFF[19:0] CoreSight ROM Table Address Offset
The base address of the component is relative to the base address of this ROM table.

Bit 1 – FMT CoreSight Rom Table Format

Value	Description
0	8-bit format
1	32-bit format

Bit 0 – EPRES CoreSight Entry Present
This bit is set to '0' by hardware under the following condition:

- DAL.CPU0 == DAL0, indicating that the device is locked from debug features

Value	Description
0	Entry not present
1	Entry present

18.6.11. CoreSight ROM Table Entry 1

Name: ENTRY1
Offset: 0x1004
Reset: 0x00005003
Property: –

Bit	31	30	29	28	27	26	25	24
	ADDOFF[19:12]							
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
	ADDOFF[11:4]							
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
	ADDOFF[3:0]							
Access	R	R	R	R				
Reset	0	1	0	1				

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

Bits 31:12 – ADDOFF[19:0] CoreSight ROM Table Address Offset
The base address of the component is relative to the base address of this ROM table.

Bit 1 – FMT CoreSight Rom Table Format

Value	Description
0	8-bit format
1	32-bit format

Bit 0 – EPRES CoreSight Entry Present
This bit is set to '0' by hardware under the following condition:

- DAL.CPU0 == DAL0, meaning the device is locked from debug features

Value	Description
0	Entry not present
1	Entry present

18.6.12. CoreSight ROM Table Entry 2

Name: ENTRY2
Offset: 0x1008
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	ADDOFF[19:12]							
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
	ADDOFF[11:4]							
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
	ADDOFF[3:0]							
Access	R	R	R	R				
Reset	0	0	0	0				

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

Bits 31:12 – ADDOFF[19:0] CoreSight ROM Table Address Offset
The base address of the component is specified relative to the base address of this ROM table.

Bit 1 – FMT CoreSight Rom Table Format

Value	Description
0	8-bit format
1	32-bit format

Bit 0 – EPRES CoreSight Entry Present
This bit is set to '0' by hardware under the following condition:

- DAL.CPU0 == DAL0, indicating that the device is locked from debug features

Value	Description
0	Entry not present
1	Entry present

18.6.13. CoreSight ROM Table Entry 3

Name: ENTRY3
Offset: 0x100C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	ADDOFF[19:12]							
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
	ADDOFF[11:4]							
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
	ADDOFF[3:0]							
Access	R	R	R	R				
Reset	0	0	0	0				

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

Bits 31:12 – ADDOFF[19:0] CoreSight ROM Table Address Offset
The base address of the component is relative to the base address of this ROM table.

Bit 1 – FMT CoreSight Rom Table Format

Value	Description
0	8-bit format
1	32-bit format

Bit 0 – EPRES CoreSight Entry Present
This bit is set to '0' by hardware under the following condition:

- DAL.CPU0 == DAL0, meaning the device is locked from debug features

Value	Description
0	Entry not present
1	Entry present

18.6.14. CoreSight ROM Table Memory Type

Name: MEMTYPE
Offset: 0x1FCC
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 0 – SYSMEM CoreSight System Memory Present
This bit's value is set to '1' when CPU0 effective DAL is DAL1 or DAL2; otherwise, it is set to '0'.

Value	Description
0	The system memory is not present on the bus, as this is a dedicated debug bus
1	The system memory is also present on this bus

18.6.15. CoreSight Peripheral Identification 4

Name: PID4
Offset: 0x1FD0
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	SIZE[3:0]				DES2[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:4 – SIZE[3:0] CoreSight Size
Always returns 0x0, indicating that this component is a ROM.

Bits 3:0 – DES2[3:0] CoreSight JEP-106 Continuation Code
Always returns 0x0.

18.6.16. CoreSight Peripheral Identification 5

Name: PID5
Offset: 0x1FD4
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RES0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – RES0[7:0] CoreSight Reserved Field
Always returns `0x00`.

18.6.17. CoreSight Peripheral Identification 6

Name: PID6
Offset: 0x1FD8
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RES0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – RES0[7:0] CoreSight Reserved Field
Always returns `0x00`.

18.6.18. CoreSight Peripheral Identification 7

Name: PID7
Offset: 0x1FDC
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RES0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – RES0[7:0] CoreSight Reserved Field
Always returns `0x00`.

18.6.19. CoreSight Peripheral Identification 0

Name: PID0
Offset: 0x1FE0
Reset: 0x000000D0
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	PART0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	0	1	0	0	0	0

Bits 7:0 – PART0[7:0] CoreSight Part Number Bits
Always returns `0xD0` (the Microchip DSU part number is `0xCD0`).

18.6.20. CoreSight Peripheral Identification 1

Name: PID1
Offset: 0x1FE4
Reset: 0x0000009C
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	DES0[3:0]				PART1[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	1	0	0	1	1	1	0	0

Bits 7:4 – DES0[3:0] CoreSight JEP106 Identification Code Bits
Always returns `0x9` for Microchip devices (Microchip JEP106 Identity Code = `0x29`).

Bits 3:0 – PART1[3:0] CoreSight Part Number Bits
Always returns `0xC` (the Microchip DSU part number is `0xCD0`).

18.6.21. CoreSight Peripheral Identification 2

Name: PID2
Offset: 0x1FE8
Reset: 0x0000003A
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	REVISION[3:0]				JEDEC	DES1[2:0]		
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	1	1	0	1	0

Bits 7:4 – REVISION[3:0] CoreSight Revision Number
Provides the REVISION value to identify this particular DSU.

Bit 3 – JEDEC CoreSight JEDEC Assignment
Always returns `0x1`, which indicates that a JEDEC value is used.

Bits 2:0 – DES1[2:0] CoreSight JEP106 Identification Code Bits
Always returns `0x2`, indicating a Microchip device (Microchip JEP106 Identity Code = `0x29`).

18.6.22. CoreSight Peripheral Identification 3

Name: PID3
Offset: 0x1FEC
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bits 7:4 – REVAND[3:0] CoreSight REVAND
Always returns `0x0` when read.

Bits 3:0 – CMOD[3:0] CoreSight Custom Modifier 0
Always returns `0x0` when read.

18.6.23. CoreSight Component Identification 0

Name: CID0
Offset: 0x1FF0
Reset: 0x0000000D
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	PRMBL0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	1	0	1

Bits 7:0 – PRMBL0[7:0] CoreSight Preamble 0
Always returns `0x0D`.

18.6.24. CoreSight Component Identification 1

Name: CID1
Offset: 0x1FF4
Reset: 0x00000010
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	CCLASS[3:0]				PRMBL1[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	0	0

Bits 7:4 – CCLASS[3:0] CoreSight Component Class
Always returns `0x1`, which identifies this component as Arm CoreSight ROM table.

Bits 3:0 – PRMBL1[3:0] CoreSight Preamble 1
Always returns `0x0` when read.

18.6.25. CoreSight Component Identification 2

Name: CID2
Offset: 0x1FF8
Reset: 0x00000005
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	PRMBL2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	1

Bits 7:0 – PRMBL2[7:0] CoreSight Preamble 2
Always returns `0x05`.

18.6.26. CoreSight Component Identification 3

Name: CID3
Offset: 0x1FFC
Reset: 0x000000B1
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	PRMBL3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	1	0	0	0	1

Bits 7:0 – PRMBL3[7:0] CoreSight Preamble 3
Always returns `0xB1`.

19. DMAC – Direct Memory Access Controller

19.1. Features

- Data Transfer From:
  - Peripheral-to-peripheral
  - Peripheral-to-memory
  - Memory-to-peripheral
  - Memory-to-memory
- Transfer Trigger Sources:
  - Software
  - Events from the Event System
  - Dedicated requests from peripherals
- SRAM-Based Transfer Descriptors:
  - Single transfer using one descriptor
  - Multi-buffer or circular buffer modes by linking multiple descriptors
- 2 Channels:
  - Enable up to 2 independent transfers
  - Automatic descriptor fetch for each channel
  - Suspend/resume operation support for each channel
- Flexible Arbitration Scheme:
  - 4 configurable priority levels for each channel
  - Fixed or round-robin priority scheme within each priority level
- Data Transfer from 1 KB to 256 KB in a Single Block Transfer
- Multiple Addressing Modes:
  - Static
  - Configurable increment scheme
- Optional Interrupt Generation:
  - On block transfer complete
  - On error detection
  - On channel suspend
- 2 Event Inputs:
  - One event input for each of the 2 lowest-numbered DMA channels
  - Can be configured to trigger normal, periodic, or conditional transfers
  - Can be configured to suspend or resume channel operation
- 2 Event Outputs:
  - One event output for each of the 2 lowest-numbered DMA channels
  - Can be configured to generate event output on beat, block, or transaction transfer completion
- Error Management Supported by Write-Back Function:
  - Dedicated write-back memory section for each channel to store ongoing descriptor transfer
- CRC Polynomial Options:

- CRC-16 (CRC-CCITT)
- CRC-32 (IEEE® 802.3)

19.2. Overview

The Direct Memory Access Controller (DMAC) contains both a Direct Memory Access engine and a Cyclic Redundancy Check (CRC) engine. The DMAC can transfer data between memories and peripherals, thereby offloading these tasks from the CPU. It enables high data transfer rates with minimal CPU intervention, and frees up valueable CPU time. With access to all peripherals, the DMAC can automatically handle data transfers between communication peripherals.

The DMA engine of the DMAC has several channels that can receive different types of transfer triggers and generate transfer requests from the DMA channels to the arbiter. The arbiter selects one DMA channel at a time to act as the active channel. When an active channel is selected, the fetch engine of the DMA fetches a transfer descriptor from the SRAM and stores it in the internal memory of the active channel, which then executes the data transaction. Refer to the *Block Diagram* section for a graphical overview of the DMAC.

An ongoing data transfer on an active channel can be interrupted by a higher priority DMA channel. The DMAC will write back the updated transfer descriptor from the internal memory of the active channel to SRAM, and grant the higher prioritized channel permission to start its transfer as the new active channel. Once a DMA channel completes its transfer, interrupts and events can optionally be generated.

The DMAC has four bus interfaces:

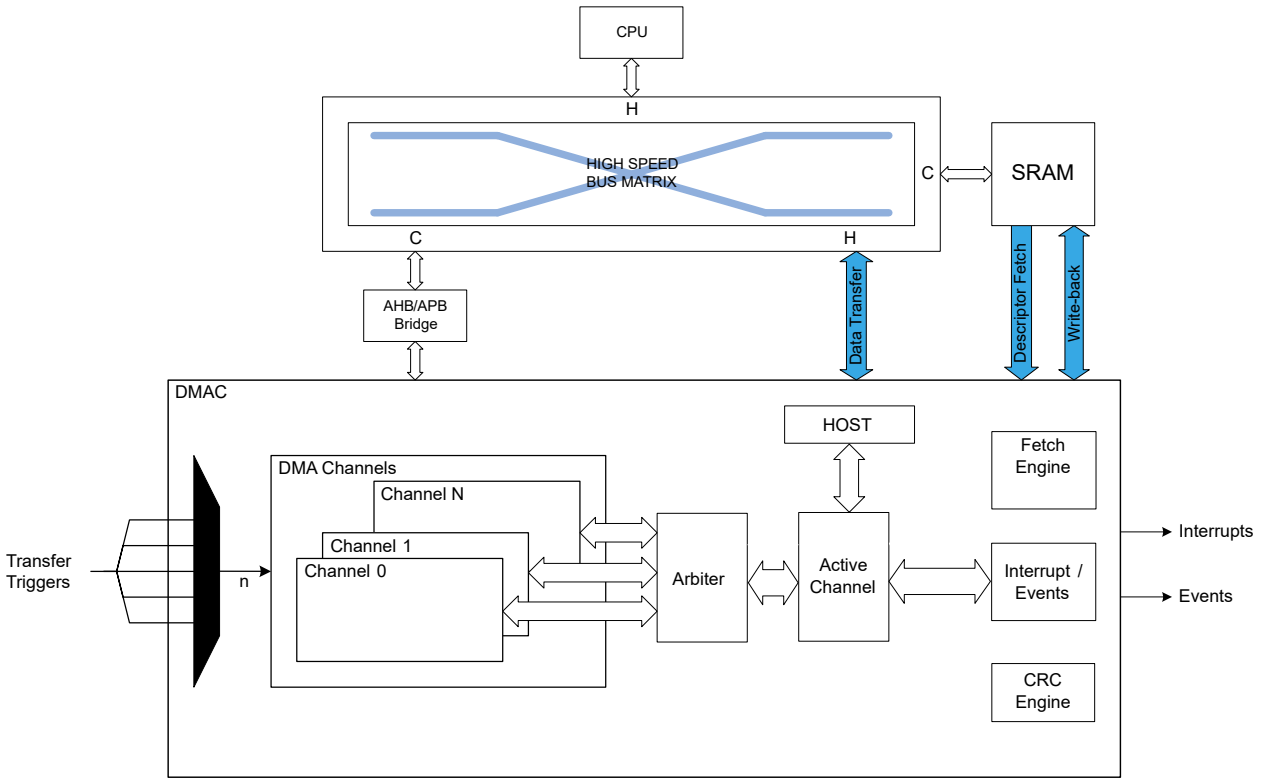
- Data Transfer Bus—Used for performing the actual DMA transfers
- AHB/APB Bridge Bus—Used for writing to and reading from the DMAC's I/O registers
- Descriptor Fetch Bus—Used by the fetch engine to retrieve transfer descriptors before a data transfer can be started or continued
- Write-back Bus—Used to write the transfer descriptor back to SRAM

All buses are Advanced High-Performance Bus (AHB) host interfaces, except for the AHB/APB Bridge bus, which is an Advanced Peripheral Bus (APB) client interface.

The CRC engine can be used by software to detect accidental errors in transferred data and to take corrective action, such as requesting the data to be sent again or simply discarding the incorrect data.

19.3. Block Diagram

Figure 19-1. DMAC Block Diagram



Note: This device has 2 DMA channels.

19.3.1. Signal Description

Not applicable.

19.4. Functional Description

19.4.1. Initialization

The following DMAC registers and bits/bit fields are enable-protected, meaning they can only be written to when the DMAC is disabled (CTRL.DMAENABLE = 0):

- Descriptor Base Memory Address (BASEADDR) register
- Write-Back Memory Base Address (WRBADDR) register

The following DMAC bit is enable-protected, meaning it can only be written when both the DMAC and CRC are disabled (CTRL.DMAENABLE = 0 and CTRL.CRCENABLE = 0):

- Software Reset bit in the Control register (CTRL.SWRST)

The following DMA channel register and bit are enable-protected, meaning it can only be written to when the corresponding DMA channel is disabled (CHCTRLA.ENABLE = 0):

- Channel Control B (CHCTRLB) register, except for the Command (CHCTRLB.CMD) bit and the Channel Arbitration Level (CHCTRLB.LVL) bit
- Channel Software Reset bit in Channel Control A (CHCTRLA.SWRST) register

The following CRC registers are enable-protected, meaning they can only be written to when the CRC is disabled (CTRL.CRCENABLE = 0):

- CRC Control (CRCCTRL) register
- CRC Checksum (CRCCHKSUM) register

Enable-protection is indicated by the *Enable-Protected* property in the register description.

Before the DMA engine is enabled, it must be configured according to the following steps:

- The SRAM address of the descriptor memory section must be written to the Description Base Address (BASEADDR) register
- The SRAM address of the write-back memory section must be written to the Write-Back Memory Base Address (WRBADDR) register
- Priority level n of the arbiter must be enabled by writing a '1' the Priority Level n Enable bit in the Control register (CTRL.LVLENn) for the arbiter to handle channels with priority level n
- Once the DMA engine is configured it can be enabled by setting the DMA Enable bit in the Control (CTRL.DMAENABLE) register

Before a DMA channel is enabled, both the channel itself and the corresponding first transfer descriptor must be configured according to the following steps:

- DMA channel configurations
  - The channel number of the DMA channel to be configured must be selected by writing to the Channel ID (CHID) register
  - Trigger action must be selected by writing to the Trigger Action bit field in the Channel Control B (CHCTRLB.TRIGACT) register
  - Trigger source must be selected by writing to the Trigger Source bit field in the Channel Control B (CHCTRLB.TRIGSRC) register
- Transfer Descriptor configurations
  - The size of each access on the data transfer bus must be selected by writing the Beat Size bit field in the Block Transfer Control (BTCTRL.BEATSIZE) register
  - The transfer descriptor must be made valid by writing a '1' to the Valid bit in the Block Transfer Control (BTCTRL.VALID) register
  - The number of beats in the block transfer must be set by writing to the Block Transfer Count (BTCNT) register
  - The source address for the block transfer must be set by writing to the Block Transfer Source Address (SRCADDR) register
  - The destination address for the block transfer must be set by writing to the Block Transfer Destination Address (DSTADDR) register
- Once the selected DMA channel is configured it can be enabled by setting the Channel Enable bit in the Channel Control A register (CHCTRLA.ENABLE)

If CRC calculation is needed, the CRC engine must be configured before it is enabled, as outlined by the following steps:

- The CRC input source must selected by writing to the CRC Input Source bit field in the CRC Control (CRCCTRL.CRCSRC) register
- The type of CRC calculation must be selected by writing to the CRC Polynomial Type bit field in the CRC Control (CRCCTRL.CRCPOLY) register
- If I/O is selected as the input source, the beat size must be selected by writing to the CRC Beat Size bit field in the CRC Control (CRCCTRL.CRCBEATSIZE) register
- Once the CRC engine is configured it can be enabled by setting the CRC Enable bit in the Control (CTRL.CRCENABLE) register

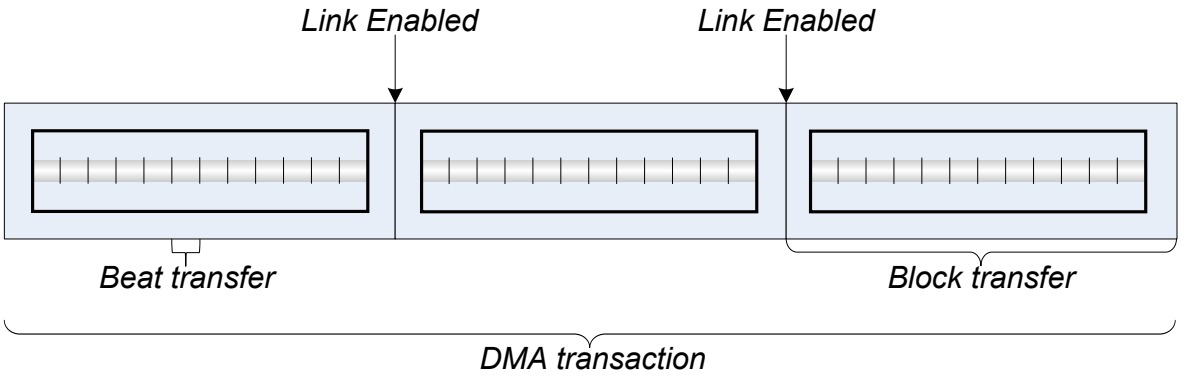
19.4.2. Operation

The DMAC consists of a DMA module and a CRC module.

19.4.2.1. DMA

The DMAC transfers data between memories and peripherals without CPU intervention. The data transferred by the DMAC are called transactions, and these transactions can be divided into smaller data transfers. The following figure illustrates the relationship between the different transfer sizes:

Figure 19-2. DMA Transfer Sizes



- Beat transfer—The size of a single data transfer bus access; the size is selected by writing to the Beat Size bit field in the Block Transfer Control (BTCTRL.BEATSIZE) register
- Block transfer—The amount of data one transfer descriptor can transfer, ranging from 1 to 64k beats. A block transfer can be interrupted by a different channel with a higher priority level.
- Transaction—The DMAC can link several transfer descriptors by having each descriptor point to the next, as shown in the figure above. A DMA transaction is the complete transfer of all blocks within a linked list.

A transfer descriptor specifies the parameters for a block transfer to be executed by the DMAC and must be stored in SRAM. For further details on the transfer descriptor, refer to the *Transfer Descriptors* section.

The figure above shows several block transfers linked together, which are called linked descriptors. For further information about linked descriptors, refer to the *Linked Descriptors* section.

A DMA transaction is initiated by an incoming transfer trigger on one of the DMA channels. This trigger can be configured as a software trigger, an event trigger, or a dedicated peripheral trigger. The transfer trigger results in a DMA transfer request from the specific channel to the arbiter. If multiple DMA channels have pending transfer requests, the arbiter selects which channel is granted access to become the active channel. The active channel will carry out the transaction as configured in the transfer descriptor. An ongoing transaction can be interrupted by a higher-prioritiy channel, however, the transaction will resume the block transfer when the lower-prioritiy channel of the DMA channels is granted access as the active channel.

For each beat transfer, an optional event output can be generated. For each block transfer, optional interrupts and an optional event output can be generated. When a transaction is completed, depending on the configuration, the DMA channel will either be suspended or disabled.

19.4.2.2. CRC

The internal CRC engine supports two commonly used CRC polynomials: CRC-16 (CRC-CCITT) and CRC-32 (IEEE 802.3). It can be used on a selectable DMA channel or on the I/O interface. Refer to the *CRC Operation* section for more details.

19.4.2.3. Enabling, Disabling and Resetting

The DMAC is enabled by writing a '1' to the DMA Enable bit in the Control (CTRL.DMAENABLE) register. The DMAC is disabled by writing a '0' to CTRL.DMAENABLE.

A DMA channel is enabled by writing a '1' to the Enable bit in the Channel Control A (CHCTRLA.ENABLE) register, after writing the corresponding channel ID to the Channel ID bit field in the Channel ID (CHID.ID) register. A DMA channel is disabled by writing a '0' to CHCTRLA.ENABLE.

The CRC is enabled by writing a '1' to the CRC Enable bit in the Control (CTRL.CRCENABLE) register. The CRC is disabled by writing a '0' to CTRL.CRCENABLE.

The DMAC is reset by writing a '1' to the Software Reset bit in the Control (CTRL.SWRST) register while both the DMAC and CRC are disabled. All registers in the DMAC, except DBGCTRL, will be reset to their initial state.

A DMA channel is reset by writing a '1' to the Software Reset bit in the Channel Control A (CHCTRLA.SWRST) register, after writing the corresponding channel ID to the Channel ID bit field in the Channel ID (CHID.ID) register. The channel registers will be reset to their initial state. The corresponding DMA channel must be disabled for the reset to take effect.

19.4.2.4. Transfer Descriptors

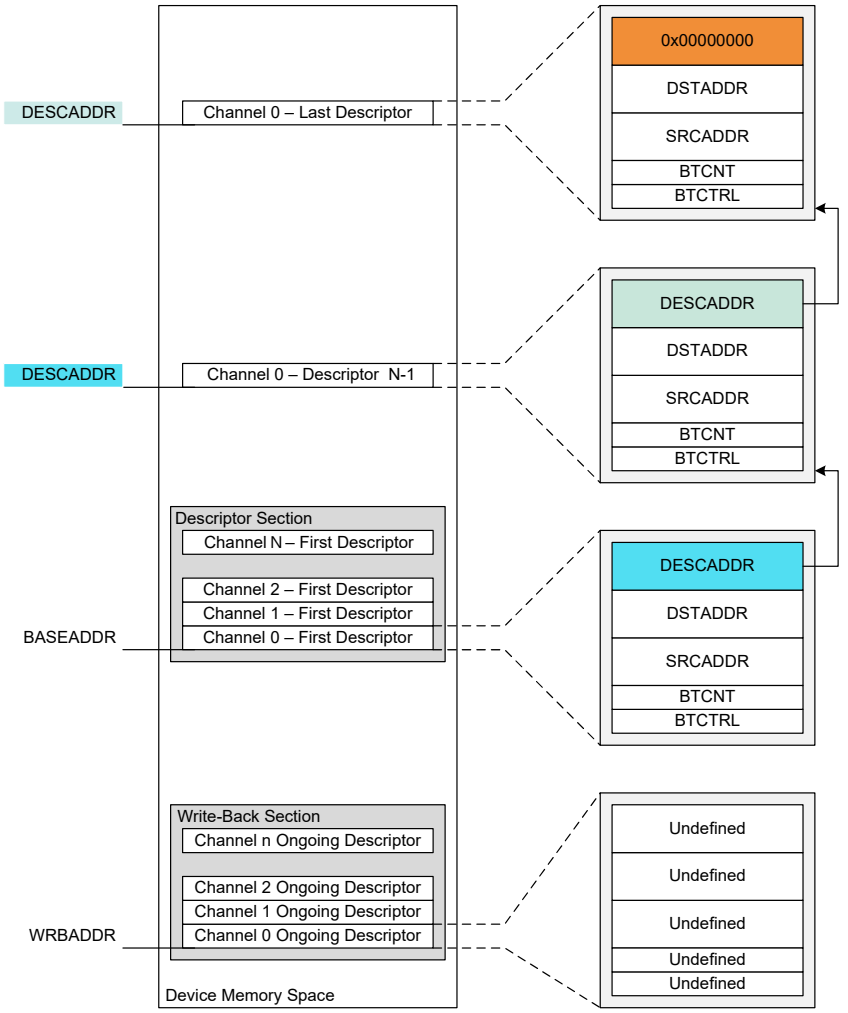
Together with the channel configurations, the transfer descriptors determine how a block transfer will be executed. Before a DMA channel is enabled (CHCTRLA.ENABLE is set to '1'), and receives a transfer trigger, its first transfer descriptor must be initialized and marked as valid (BTCTRL.VALID). The first transfer descriptor defines the first block transfer of a transaction.

All transfer descriptors must reside in SRAM. The addresses stored in the Descriptor Memory Section Base Address (BASEADDR) and Write-Back Memory Section Base Address (WRBADDR) registers indicate to the DMAC where to find the descriptor memory section and the write-back memory section.

The descriptor memory section is where the DMAC expects to find the first transfer descriptors for all DMA channels. Since BASEADDR only points to the first transfer descriptor of channel 0 (see Figure 19-3), all first transfer descriptors must be stored in a contiguous memory section, ordered according to their channel number. Refer to the *Linked Descriptors* section for more details.

The write-back memory section is where the DMAC stores the transfer descriptors for ongoing block transfers. WRBADDR points to the ongoing transfer descriptor of channel 0. All ongoing transfer descriptors are stored in a contiguous memory section, ordered according to their channel number. The figure below shows an example of linked descriptors on DMA channel 0. Refer to the *Linked Descriptors* section for further details.

Figure 19-3. Memory Sections



Note: This device has 2 DMA channels.

The size of the descriptor and write-back memory sections depends on the highest-numbered enabled DMA channel, *N*, as shown below:

$Size = 128\,\text{bits} \cdot (N + 1)$

For memory optimization, it is recommended to use the lowest-numbered DMA channels when not all channels are required.

The descriptor and write-back memory sections can either be two separate memory sections, or share the same memory section (BASEADDR = WRBADDR). The benefit of having them in two separate sections is that the same transaction for a channel can be repeated without modifying the first transfer descriptor. The benefit of having descriptor memory and write-back memory in the same section is that it requires less SRAM. Additionally, the latency from fetching the first descriptor of a transaction to the first executed beat transfer is reduced.

19.4.2.5. Arbitration

If a DMA channel is enabled and not suspended when it receives a transfer trigger, it will send a transfer request to the arbiter. When the arbiter receives the transfer request, it will include the DMA channel in the queue of channels with pending transfers, and the corresponding Pending Channel n bit in the Pending Channels (PENDCH.PENDCHn) registers will be set. Depending on the arbitration scheme, the arbiter will choose which DMA channel will be the next active channel.

The active channel is the DMA channel granted access to perform its next beat transfer. When the arbiter has granted a DMA channel access to the DMAC, the corresponding PENDCH.PENDCHn bit will be cleared. See also Figure 19-4.

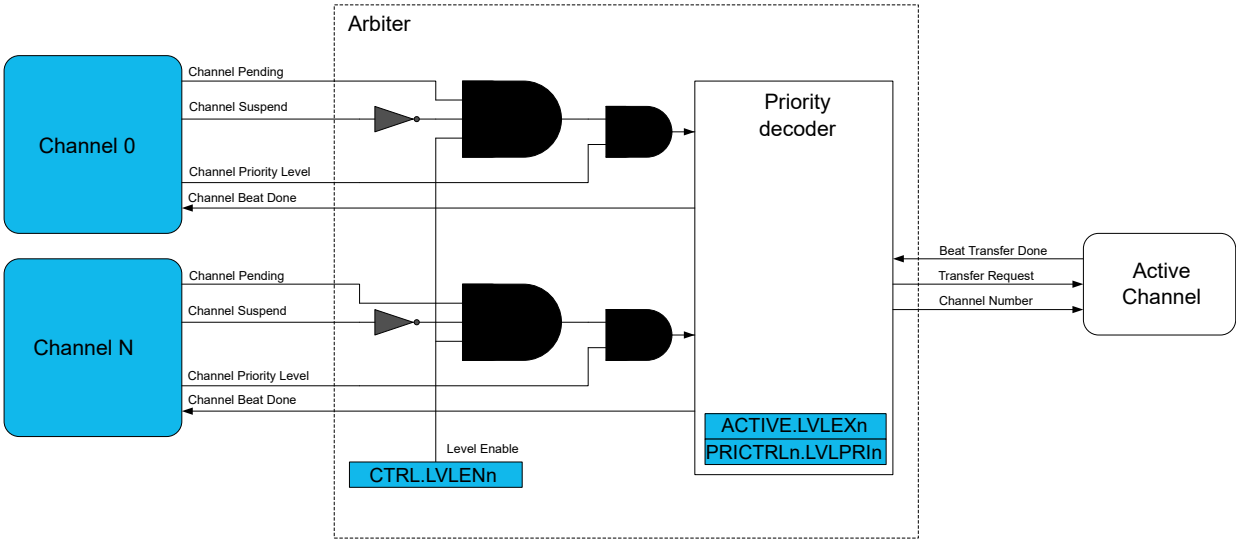
If the upcoming beat transfer is the first for the transfer request, the corresponding Busy Channel n bit in the Busy Channels register will be set (BUSYCH.BUSYCHn = 1), and it will remain '1' for the subsequent granted beat transfers.

When the channel has performed its granted beat transfer(s), it will either be place back into the queue of channels with pending transfers, set to wait for a new transfer trigger, suspended, or disabled. This behavior depends on the channel and block transfer configuration. If the DMA channel is placed back into the queue of channels with pending transfers, the corresponding BUSYCH.BUSYCHn bit will remain '1'. If the DMA channel is set to wait for a new transfer trigger, suspended, or disabled, the corresponding BUSYCH.BUSYCHn bitwill be cleared.

If a DMA channel is suspended while it has a pending transfer, it will be removed from the queue of pending channels, but the corresponding PENDCH.PENDCHn bitwill remain set. When the same DMA channel is resumed, it will be added to the queue of pending channels again.

If a DMA channel is disabled (CHCTRLA.ENABLE = 0) while it has a pending transfer, it will be removed from the queue of pending channels, and the corresponding PENDCH.PENDCHn will be cleared.

Figure 19-4. Arbiter Overview



Note: This device has 2 DMA channels.

Priority Levels

When a channel level is pending or the channel is transferring data, the corresponding Level Executing bit is set in the Active Channel and Levels register (ACTIVE.LVLEXn).

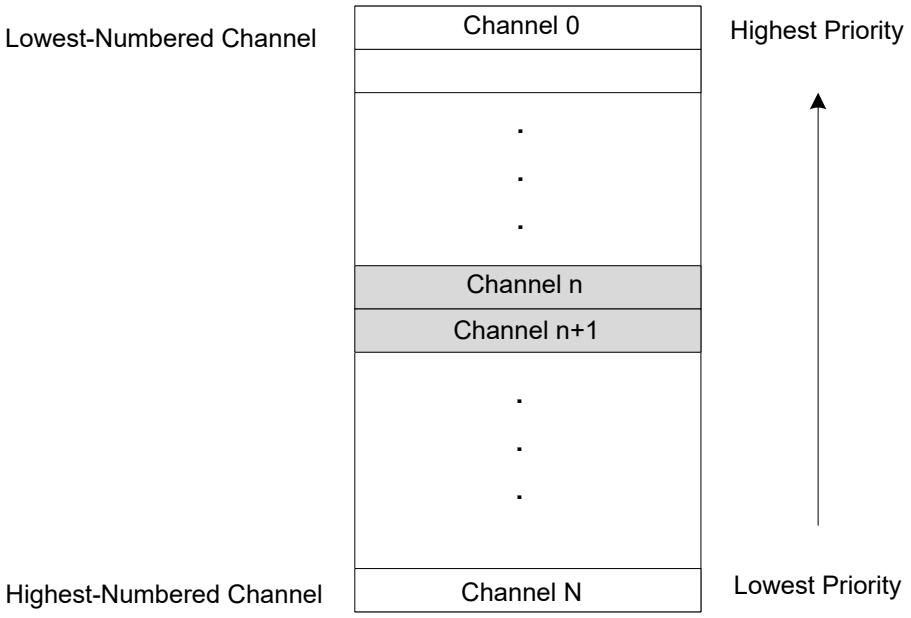
Each DMA channel supports a 4-level priority scheme. The priority level for a channel is configured by writing to the Channel Arbitration Level bit field in the Channel Control B register (CHCTRLB.LVL). As long as all priority levels are enabled, a channel with a higher priority level number will have priority over a channel with a lower priority level number. Each priority level n is enabled by setting the corresponding Priority Level n Enable bit in the Control register (CTRL.LVLENn = 1).

Within each priority level, the DMAC's arbiter can be configured to prioritize channels either statically or dynamically:

Static arbitration within a priority level is selected by writing a '0' to the Level n Round-Robin Scheduling Enable bit in the Priority Control 0 (PRICTRL0.RRLVLENn) register.

When static arbitration is selected, the arbiter will prioritize lower channel numbers over higher channel numbers as shown Figure 19-5. When using the static arbitration there is a risk that higher channel numbers may never be granted access as the active channel. This riskcan be avoided by using a dynamic arbitration scheme.

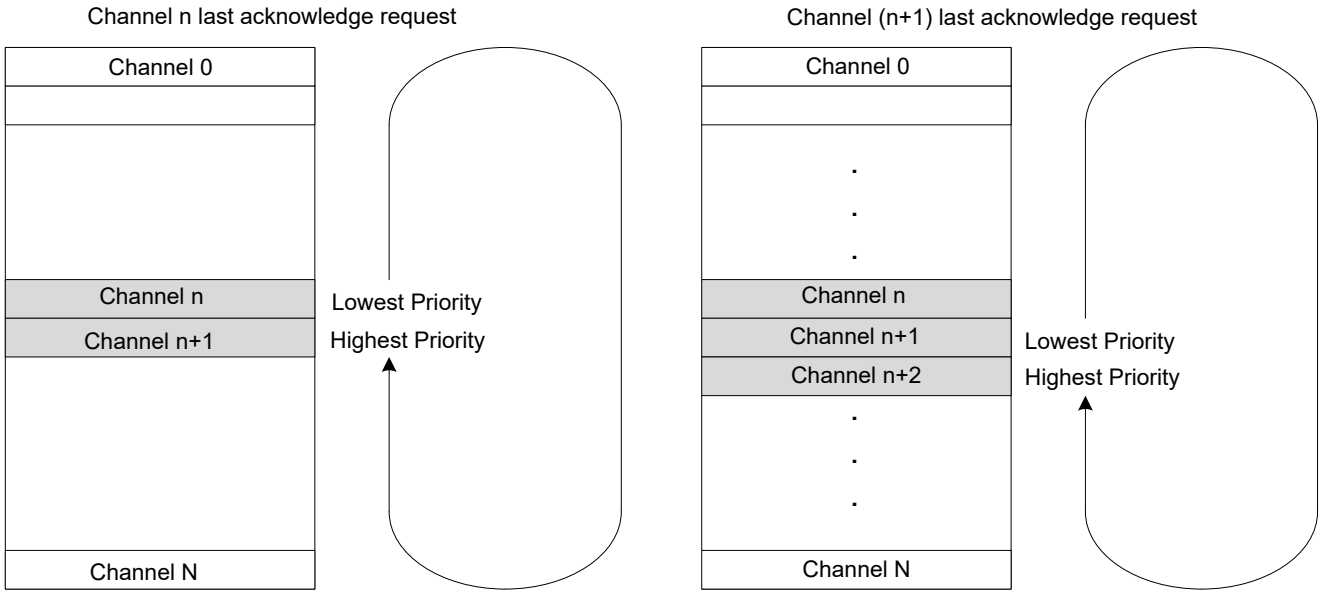
Figure 19-5. Static Priority Scheduling



Dynamic arbitration within a priority level is selected by writing a '1' to PRICTRL0.RRLVLENn.

The dynamic arbitration scheme in the DMAC is round-robin. With the round-robin scheme, the channel number of the last channel being granted access will have the lowest priority the next time the arbiter must grant access to a channel within the same priority level, as shown Figure 19-6. The channel number of the last channel granted access as the active channel is stored in the Level n Channel Priority Number bit field in the Priority Control 0 (PRICTRL0.LVLPRIn) register for the corresponding priority level.

Figure 19-6. Dynamic (Round-Robin) Priority Scheduling



19.4.2.6. Data Transaction

Before the DMAC can perform a data transaction, a DMA channel must be configured and enabled, its corresponding transfer descriptor must be initialized, and the arbiter must grant the DMA channel access as the active channel.

Once the arbiter has granted a DMA channel access as the active channel (refer to the *DMA Block Diagram* section), the transfer descriptor for the DMA channel will be fetched from SRAM using the fetch bus and stored in the internal memory for the active channel. When the transfer starts, the corresponding Channel Busy status flag is set in Channel Status register (CHSTATUS.BUSY).

For a new block transfer, the transfer descriptor will be fetched from the descriptor memory section. Refer to the *Descriptor Memory Section Base Address (BASEADDR)* register section for more information.

For an ongoing block transfer, the descriptor will be fetched from the write-back memory section. Refer to the *Write-Back Memory Section Base Address (WRBADDR)* register section for more information.

Using the data transfer bus, the DMAC reads data from the current source address and writes it to the current destination address. Refer to the *Addressing* section for further details on how the current source and destination addresses are calculated.

The arbitration procedure is performed after each beat transfer. If the current DMA channel is granted access again, the Block Transfer Counter (BTCNT) of the internal transfer descriptor will be decremented by '1', an event output may optionally be generated, and the active channel will perform a new beat transfer. If a different DMA channel is granted access, the block transfer counter value will be written to the write-back section before the transfer descriptor of the newly granted DMA channel is fetched into the internal memory of the active channel.

When a block transfer is complete (BTCNT is '0'), the Valid bit in the Block Transfer Control register will be cleared (BTCTRL.VALID = 0) before the entire transfer descriptor is written to the write-back memory. Optional interrupts, Channel Transfer Complete and Channel Suspend, and the optional Block event output, will be generated if configured and enabled. After the last block transfer in a transaction, the Next Descriptor Address (DESCADDR) register will hold the value 0x00000000, and the DMA channel will either be suspended or disabled, depending on the configuration in the

Block Action bit field in the Block Transfer Control (BTCNT.BLOCKACT) register. If the transaction has additional block transfers pending, DESCADDR will hold the SRAM address of the next transfer descriptor to be fetched. The DMAC will fetch the next descriptor into the internal memory of the active channel and write its contents to the write-back section for the channel, before the arbiter selects the next active channel.

19.4.2.7. Transfer Triggers and Actions

A DMA transfer through a DMA channel can be started only when a DMA transfer request is detected and the DMA channel has been granted access to the DMA. A transfer request can be triggered by software, a peripheral or an event.

A peripheral trigger source for each DMA channel can be selected using the Trigger Source bit field in the Channel Control B (CHCTRLB.TRIGSRC) register.

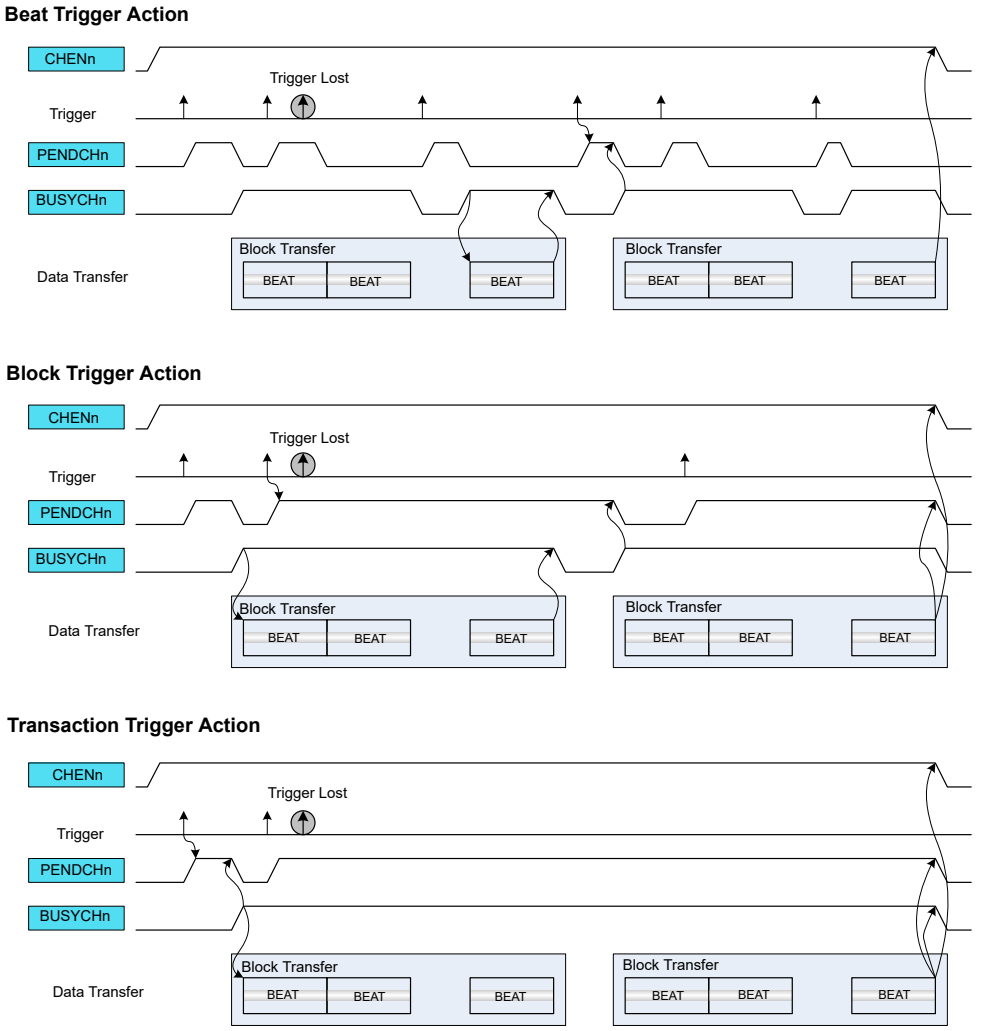
A DMA software trigger can be generated by setting the Channel n Software Trigger bit in the Software Trigger Control (SWTRIGCTRL.SWTRIGn) register. Software triggers can be generated regardless of the CHCTRLB.TRIGSRC configuration.

Refer to the *Event Input Actions* section for details on configuring DMA transfers triggered by events.

The trigger actions are available in the Trigger Action bit field in the Channel Control B (CHCTRLB.TRIGACT) register. By default, a trigger generates a request for a block transfer operation. If a single descriptor is defined for a channel, the channel is automatically disabled when a block transfer is completed. If a list of linked descriptors is defined for a channel, the channel is automatically disabled when the last descriptor in the list is executed. If the list still contains descriptors to execute, the channel will wait for the next block transfer trigger. When enabled again, the channel will continue to wait for the next block transfer trigger. The trigger actions can also be configured to generate a request for a beat transfer (CHCTRLB.TRIGACT = `0x2`) or transaction transfer (CHCTRLB.TRIGACT = `0x3`), instead of a block transfer (CHCTRLB.TRIGACT = `0x0`).

The following figure shows an example of triggers used with two linked block descriptors.

Figure 19-7. Trigger Action and Transfers



If the trigger source generates a transfer request for a channel during an ongoing transfer, the new transfer request will be kept pending (CHSTATUS.PEND = 1), and the new transfer can start after the ongoing one is complete. Only one pending transfer can be kept per channel. If the trigger source generates additional transfer requests while one is already pending, those additional requests will be lost. All channels pending status flags are also available in the Pending Channels (PENDCH) register.

When the transfer starts, the corresponding Channel Busy status flag is set in Channel Status (CHSTATUS.BUSY) register. When the trigger action is complete, the Channel Busy status flag is cleared. All channel busy status flags are also available in the Busy Channels (BUSYCH) register in theDMAC.

19.4.2.8. Addressing

Each block transfer must have both a source address and a destination address defined. The source address is set by writing to the Transfer Source Address (SRCADDR) register, and the destination address is set by writing the Transfer Destination Address (DSTADDR) register.

The addressing of the DMAC can be static or incremental for either source or destination of a block transfer, or for both.

Incrementation for the source address of a block transfer is enabled by writing a '1' to the Source Address Incrementation Enable bit in the Block Transfer Control (BTCTRL.SRCINC) register. The step

size for incrementation is configurable and can be changed by writing a '1' to the Step Selection bit in the Block Transfer Control (BTCTRL.STEPSEL) register and selecting the desired step size in the Address Increment Step Size bit field in the Block Transfer Control (BTCTRL.STEPSIZE) register. If BTCTRL.STEPSEL is '0', the step size for source incrementation will be the size of one beat.

When source address incrementation is enabled (BTCTRL.SRCINC = 1), SRCADDR must be set to the ending address and is calculated as follows:

If BTCTRL.STEPSEL = 1:

$$\text{SRCADDR} = \text{SRCADDR}_{\text{START}} \cdot \text{BTCNT} + \text{BiB} \cdot 2^{\text{STEPSIZE}}$$

If BTCTRL.STEPSEL = 0:

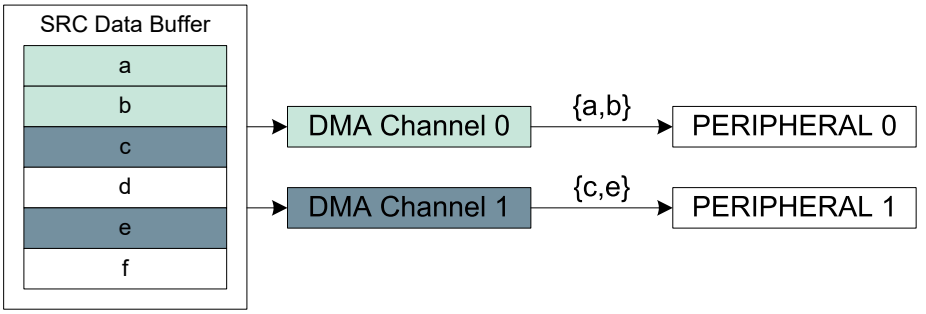
$$\text{SRCADDR} = \text{SRCADDR}_{\text{START}} \cdot \text{BTCNT} + \text{BiB}$$

Where,

- $\text{SRCADDR}_{\text{START}}$ is the source address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BiB is the configured number of bytes in a beat, as controlled by the BEATSIZE bit field.
- STEPSIZE is the configured number of beats for each incrementation

Figure 19-8 shows an example where DMA channel 0 is configured to increment the source address by one beat after each beat transfer (BTCTRL.SRCINC = 1), and DMA channel 1 is configured to increment the source address by two beats (BTCTRL.SRCINC = 1, BTCTRL.STEPSEL = 1 and, BTCTRL.STEPSIZE = 0x1). Since the destination address for both channels is a peripheral, destination incrementation is disabled (BTCTRL.DSTINC = 0).

Figure 19-8. Source Address Increment



Incrementation for the destination address of a block transfer is enabled by writing a '1' to the Destination Address Incrementation Enable bit in the Block Transfer Control (BTCTRL.DSTINC) register. The step size for incrementation is configurable by clearing BTCTRL.STEPSEL = 0 and setting BTCTRL.STEPSIZE to the desired step size. If BTCTRL.STEPSEL is set to '1' , the step size for the destination incrementation will be the size of one beat.

When destination address incrementation is enabled (BTCTRL.DSTINC = 1), DSTADDR must be set to the ending address and calculated as follows:

If BTCTRL.STEPSEL = 1:

$$\text{DSTADDR} = \text{DSTADDR}_{\text{START}} \cdot \text{BTCNT} + \text{BiB}$$

If BTCTRL.STEPSEL = 0:

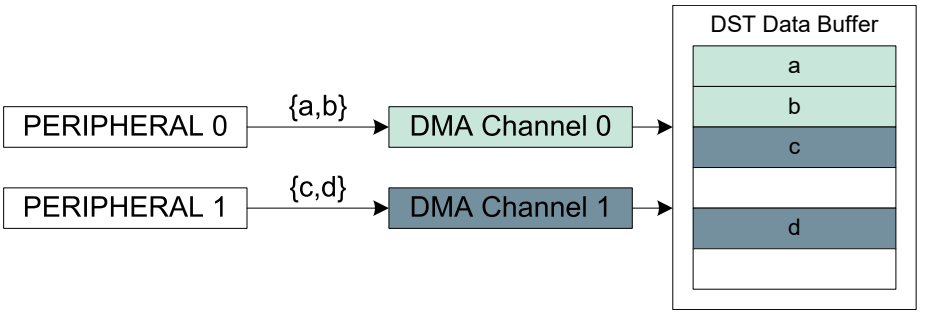
$$\text{DSTADDR} = \text{DSTADDR}_{\text{START}} \cdot \text{BTCNT} + \text{BiB} \cdot 2^{\text{STEPSIZE}}$$

Where,

- $DSTADDR_{START}$ is the destination address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BiB is the configured number of bytes in a beat, as controlled by the BEATSIZE bit field.
- STEPSIZE is the configured number of beats for each incrementation

Figure 19-9 shows an example where DMA channel 0 is configured to increment the destination address by one beat (BTCTRL.DSTINC = 1), and DMA channel 1 is configured to increment the destination address by two beats (BTCTRL.DSTINC = 1, BTCTRL.STEPSEL = 0, and BTCTRL.STEPSIZE = 0x1). Since the source address for both channels is a peripheral, source incrementation is disabled (BTCTRL.SRCINC = 0).

Figure 19-9. Destination Address Increment



19.4.2.9. Error Handling

If a bus error is received from an AHB client during a DMA data transfer, the corresponding active channel is disabled and the corresponding Channel Transfer Error Interrupt flag in the Channel Interrupt Status and Clear (CHINTFLAG.TERR) register is set. If enabled, the optional transfer error interrupt request is generated. The transfer counter will not be decremented, and its current value is written back to the write-back memory section before the channel is disabled.

When the DMAC fetches an invalid descriptor (BTCTRL.VALID = 0), or when the channel is resumed and the DMA fetches the next descriptor with null address (DESCADDR = 0x00000000), the corresponding channel operation is suspended. The Channel Suspend Interrupt flag in the Channel Interrupt Flag Status and Clear (CHINTFLAG.SUSP) register is set, and the Channel Fetch Error bit in the Channel Status (CHSTATUS.FERR) register is set. If enabled, the optional suspend interrupt is generated.

19.4.3. Additional Features

19.4.3.1. Linked Descriptors

A transaction can consist of either a single block transfer or of several block transfers. When a transaction consists of multiple block transfers, this is accomplished using linked descriptors.

Figure 19-3 illustrates how linked descriptors work. When the first block transfer is completed on DMA channel 0, the DMAC fetches the next transfer descriptor, which is pointed to by the value stored in the Next Descriptor Address (DESCADDR) register of the first transfer descriptor is reached. This process of fetching the next transfer descriptor continues until the last transfer descriptor. When the block transfer for the last transfer descriptor is executed and DESCADDR = 0x00000000, the transaction is terminated. For further details on how the next descriptor is fetched from SRAM, refer to the *Data Transaction* section.

19.4.3.1.1. Adding a Descriptor to the End of a List

To add a new descriptor at the end of the descriptor list, create the descriptor in SRAM with the Next Descriptor Address (DESCADDR) register = 0x00000000, indicating that it is the new last descriptor in the list. Then modify the DESCADDR value of the current last descriptor to point to the address of the newly created descriptor.

19.4.3.1.2. Adding a Descriptor to a List

To add descriptors to a linked list, perform the following actions:

1. Enable the Suspend Interrupt for the DMA channel.
2. Enable the DMA channel.
3. Reserve memory space in SRAM to configure a new descriptor.
4. Configure the new descriptor:
   - Set the Next Descriptor Address (DESCADDR) register
   - Set the Block Transfer Destination Address (DSTADDR) register
   - Set the Block Transfer Source Address (SRCADDR) register
   - Configure the Block Transfer Control (BTCTRL) register and set the Descriptor Valid (VALID) bit
5. Clear the VALID bit for the existing list and for the descriptor that needs to be updated.
6. Read DESCADDR from the Write-Back memory.
   - If the DMA has not already fetched the descriptor that requires changes (i.e., DESCADDR is incorrect):
     - Update the DESCADDR location of the descriptor in the list
     - Clear the Channel Suspend flag (CHINTFLAG.SUSP) if the Suspend Block Action was enabled (BTCTRL.BLOCKACT = `0x2`)
     - Set the Descriptor Valid bit
     - Issue a Resume software command (CHCTRLB.CMD = `0x02`) if the channel was suspended
   - If the DMA is executing the same descriptor as the one which requires changes:
     - Issue the SUSPEND software command (CHCTRLB.CMD = `0x1`) and wait for the Channel Suspend (CHINTFLAG.SUSP) interrupt flag to be set
     - Update the next descriptor address DESCRADDR in the Write-Back memory
     - Clear CHINTFLAG.SUSP and issue the Resume software command (CHCTRLB.CMD = `0x02`)
     - Update the DESCADDR location of the descriptor in the list
     - Optionally clear the Suspend Block Action
     - Set the Descriptor Valid (VALID) bit
7. Repeat from step three if more descriptors are needed.

19.4.3.1.3. Adding a Descriptor Between Existing Descriptors

To insert a new descriptor 'C' between two existing descriptors ('A' and 'B'), the descriptor currently being executed by the DMA must be identified.

1. If DMA is executing descriptor B, descriptor C cannot be inserted.
2. If DMA has not started executing descriptor A, follow the steps:
   a. Clear the Descriptor Valid (BTCTRL.VALID) bit of Descriptor A.
   b. Configure the Next Descriptor Address (DESCADDR) register of descriptor A to point to descriptor C instead of descriptor B.
   c. Configure the Next Descriptor Address (DESCADDR) register of descriptor C to point to descriptor B.
   d. Set the Descriptor Valid (BTCTRL.VALID) bit of descriptor A.
3. If DMA is executing descriptor A:

a. Issue a Suspend software command (CHCTRLB.CMD = 0x01) to the channel.
b. Perform steps 2.a through 2.d.
c. Issue a Resume software command (CHCTRLB.CMD = 0x02) to the channel.

19.4.3.2. Channel Suspend

The channel operation can be suspended at any time by software by writing to the Suspend command in the Command bit field of Channel Control B register (CHCTRLB.CMD = 0x01). After the ongoing beat transfer is completed, the channel operation is suspended and CHCTRLB.CMD is automatically cleared.

When suspended, the Channel Suspend Interrupt flag in the Channel Interrupt Status and Clear register is set (CHINTFLAG.SUSP = 1), and the optional suspend interrupt is generated.

By configuring the block action to suspend by writing to the Block Action bit field in the Block Transfer Control register (BTCTRL.BLOCKACT is 0x2 or 0x3), the DMA channel will be suspended after it has completed a block transfer. The DMA channel will remain enabled and able to receive transfer triggers, but it will be removed from the arbitration scheme.

If an invalid transfer descriptor (BTCTRL.VALID = 0) is fetched from SRAM, the DMA channel will be suspended, and the Channel Fetch Error bit in the Channel Status (CHASTATUS.FERR) register will be set.

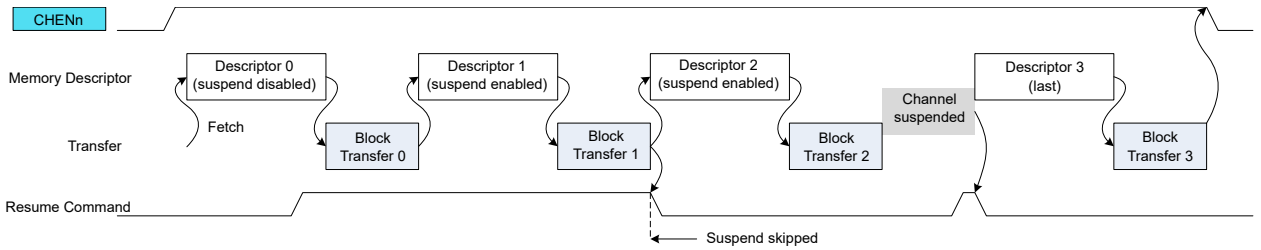
Note: Only enabled DMA channels can be suspended. If a channel is disabled when a suspend command is issued, the internal suspend command will be ignored.

Refer to the *Transfer Descriptors* section for more details on transfer descriptors.

19.4.3.3. Channel Resume and Next Suspend Skip

A channel operation can be resumed by software by setting the Resume command in the Command bit field of the Channel Control B (CHCTRLB.CMD) register. If the channel is already suspended, the channel operation resumes from where it previously stopped when the Resume command is detected. If the Resume command is issued before the channel is suspended, the next suspend action is skipped and the channel continues the normal operation.

Figure 19-10. Channel Suspend/Resume Operation



19.4.3.4. Aborting Transfers

Transfers on any channel can be aborted by software by disabling the corresponding DMA channel. It is also possible to abort all ongoing or pending transfers by disabling the DMAC.

When a DMA channel disable request or DMAC disable request is detected:

- Ongoing transfers of the active channel will be disabled after the ongoing beat transfer is completed and the write-back memory section is updated. This prevents transfer corruption before the channel is disabled.
- All other enabled channels will be disabled in the next clock cycle

The corresponding Channel Enable bit in the Channel Control A register is cleared (CHCTRLA.ENABLE = 0) when the channel is disabled.

The corresponding DMAC Enable bit in the Control register is cleared (CTRL.DMAENABLE = 0) when the entire DMAC module is disabled.

19.4.3.5. CRC Operation

A Cyclic Redundancy Check (CRC) is an error detection technique used to identify errors in data. It is commonly used to determine whether data transmitted or stored in data and program memories has been corrupted. A CRC takes a data stream or a block of data as input and generates a 16- or 32-bit output that can be appended to the data and used as a checksum.

When the data is received, the device or application repeats the CRC calculation. If the new CRC result does not match the one calculated earlier, the block contains a data error. The application will then detect this and may take a corrective action, such as requesting the data to be sent again or simply discarding the incorrect data.

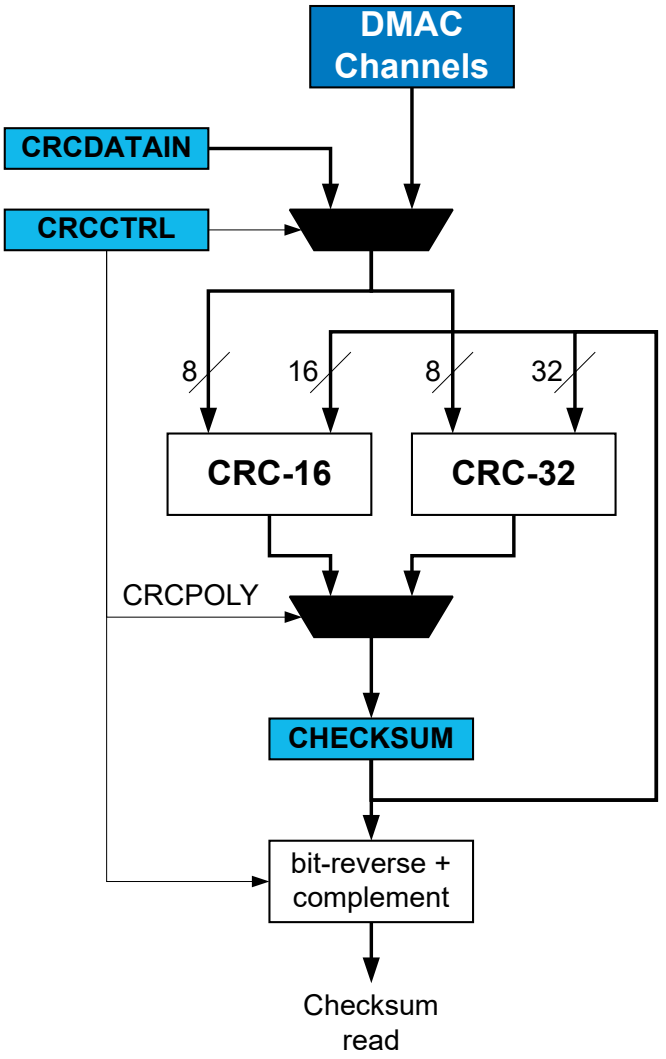
The CRC engine in the DMAC supports two commonly used CRC polynomials: CRC-16 (CRC-CCITT) and CRC-32 (IEEE 802.3).

- CRC-16:
  - Polynomial: $x^{16} + x^{12} + x^5 + 1$
  - Hex value: 0x1021
- CRC-32:
  - Polynomial: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
  - Hex value: 0x04C11DB7

The data source for the CRC engine can be either one of the DMA channels or the APB bus interface, and must be selected by writing to the CRC Input Source bits in the CRC Control (CRCCTRL.CRCSRC) register. The CRC engine then takes data input from the selected source and generates a checksum based on these data. The checksum is available in the CRC Checksum (CRCCHKSUM) register. When CRC-32 polynomial is used, the final checksum read is bit-reversed and complemented, as shown in the following figure.

The CRC polynomial is selected by writing to the CRC Polynomial Type bit in the CRC Control (CRCCTRL.CRCPOLY) register, the default is CRC-16. The CRC engine operates on byte only. When the DMA is used as the data source for the CRC engine, the DMA channel beat size setting will be used. When the APB bus interface is used, the application must configure the CRC Beat Size bit field of the CRC Control (CRCCTRL.CRCBEATSIZE) register. 8-, 16-, or 32-bit bus transfer access types are supported. The corresponding number of bytes will be written in the CRCDATAIN, register and the CRC engine will operate on the input data in a byte-by-byte manner.

Figure 19-11. CRC Generator Block Diagram



CRC on DMA data	CRC-16 or CRC-32 calculations can be performed on data passing through any DMA channel. Once a DMA channel is selected as the source, the CRC engine will continuously generate the CRC on the data passing through that channel. The checksum is available for readout once the DMA transaction is completed or aborted. A CRC can also be generated on SRAM, Flash or I/O memory by passing these data through a DMA channel. In this case, the destination register for the DMA data can be the data input CRC Data Input (CRCDATAIN) register in the CRC engine.
CRC using the I/O interface	Before using the CRC engine with the I/O interface, the application must set the CRC Beat Size bit field in the CRC Control (CRCCTRL.CRCBEATSIZE) register. 8/16/32-bit bus transfer type can be selected.

CRC can be performed on any data by loading it into the CRC engine using the CPU and writing the data to the CRCDATAIN register. Using this method, an arbitrary number of bytes can be written to the register by the CPU, and the CRC is calculated continuously for each byte, starting with the lowest byte. When the CRC engine is loaded by word or half-word, the CRC calculation also starts with the lowest byte of each word.

The completion of the CRC checksum calculation is signaled by the DMAC setting the CRC Busy bit in the CRC Status (CRCSTATUS.CRCBUSY) register. Before reading the checksum, CRCSTATUS.CRCBUSY must be cleared, as the 'bit-reverse + complement' operation is not performed while this bit is set.

CRCSTATUS.CRCBUSY will be set by the hardware when the first byte of data is loaded. Once all the data has been loaded, CRCSTATUS.CRCBUSY must be cleared in software for the 'bit-reverse + complement' operation to be performed, ensuring that the CRCCHKSUM register is ready to be read.

Manual writes to the CRC must be performed while CRCSTATUS.CRCBUSY is set.

19.4.4. Sleep Mode Operation

Each DMA channel can be configured to operate in any sleep mode. To enable operation in standby mode, the RUNSTDBY bit in Channel Control A (CHCTRLA.RUNSTDBY) register must be set. The DMAC can wake up the device using interrupts from any sleep mode or perform actions through the Event System.

For channels where CHCTRLA.RUNSTDBY is '0', it is up to the software to stop DMA transfers on these channels and wait for completion before going into Standby sleep mode using the following sequence:

1. Suspend the DMAC channels for which CHCTRLA.RUNSTDBY is '0'.
2. Check the SYNCBUSY bits of the registers accessed by the DMAC channels being suspended.
3. Enter sleep mode.
4. When the device wakes up, resume the suspended channels.

Note:

1. In Standby sleep mode, the DMAC can only access RAM when it is not back biased. Refer to the *PM – Power Manager* chapter for information on back biasing.

19.4.5. Debug Operation

When the CPU is halted in Debug mode, the DMAC will halt normal operation. However, the DMAC can be forced to continue operation during debugging. Refer to the *DBGCTRL* register section for details.

19.5. Dependencies

19.5.1. I/O Lines

Not applicable.

19.5.2. Power Management

The DMAC will continue to operate in any sleep mode where the selected source clock is running. The DMAC's interrupts can be used to wake up the device from sleep modes. Refer to the *PM – Power Manager* chapter for details on the different sleep modes.

19.5.3. Clocks

The DMAC bus clocks (CLK_DMAC_AHB and CLK_DMAC_APB) can be enabled and disabled in the Main Clock. The default state of bus clocks can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

19.5.4. DMA

Not applicable.

19.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use DMAC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Channel Interrupt Flag Status and Clear (CHINTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be

individually enabled by writing a '1' to the corresponding bit in the Channel Interrupt Enable Set (CHINTENSET) register, and disabled by writing a '1' to the corresponding bit in the Channel Interrupt Enable Clear (CHINTENCLR) register.

CHINTFLAG, CHINTENSET and CHINTENCLR reflects the interrupt status and configuration of the channel currently selected in the Channel ID (CHID) register. The Interrupt Status (INTSTATUS) register or Interrupt Pending (INTPEND) register must be used to determine which value should be written to CHID to handle a set interrupt flag. INTSTATUS indicates all channels with a pending interrupt, while INTPEND will indicate the ID and interrupt flag status of the lowest-numbered DMA channel that has a pending interrupt.

An interrupt request is generated when the interrupt flag is set and the corresponding interrupt source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the CHINTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the CHINTFLAG register must be read to determine what the interrupt condition is.

Table 19-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
DMAC	TERR	A bus error has occurred during a beat transfer or, an invalid descriptor has been fetched on Channel n	
DMAC	TCMPL	A block transfer has completed on Channel n	
DMAC	SUSP	Channel n has been suspended	

19.5.6. Events

The DMAC can generate the following events:

Table 19-2. DMAC Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
DMAC	CH_n	Transfer on channel n is complete	Pulse	CLK_DMAC_AHB	One CLK_DMAC_AHB period

Writing a '1' to the Channel Event Output Enable bit in the Channel Control B (CHCTRLB.EVOE) register enables the corresponding event output. Writing a '0' to this bit disables the corresponding event output.

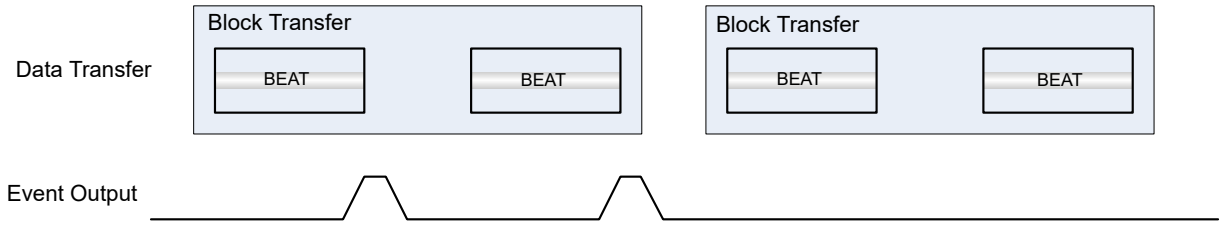
The event output cause is selected by writing to the Event Output Selection bits in the Block Transfer Control (BTCTRL.EVOSEL) register. It is possible to generate events after each block transfer (BTCTRL.EVOSEL = 0x1) or after each beat transfer (BTCTRL.EVOSEL = 0x3). To enable the generation of an event when a transaction is complete, the block event selection must be set only in the last transfer descriptor.

Note: Only the 2 lowest-numbered DMA channels can be configured as event output generators.

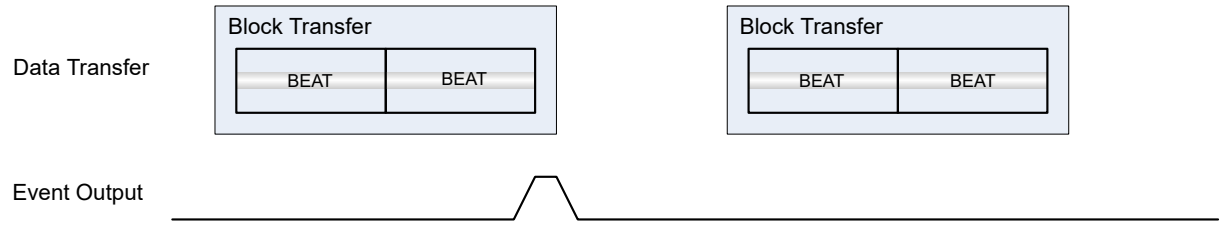
Figure 19-12 shows an example where the event output generation is enabled for the first block transfer and disabled for the second block.

Figure 19-12. Event Output Generation

Beat Event Output



Block Event Output



The DMAC can connect to the following events:

Table 19-3. Event Users

User Name		Description	Input Detection	Channel Path Type
Peripheral	Input			
DMAC	CH_n	Channel n	Level	Asynchronous Synchronous Resynchronized

Writing a '1' to a Channel Event Input Enable bit in the Channel Control B register (CHCTRLB.EVIE) enables the corresponding action on input event. Writing a '0' to this bit disables the corresponding action on input event.

Note: Only the 2 lowest-numbered DMA channels can be configured as event users.

The event users can trigger the following actions:

Table 19-4. Event Actions

Event Input	Event Action	Description
CH_n	TRIG	Normal transfer or conditional transfer on strobe triggered by an event
	CTRIG	Conditional transfer triggered by an event
	CBLOCK	Conditional block transfer triggered by an event
	SUSPEND	Suspends a channel's operation on event
	RESUME	Resumes a suspended channel's operation on event
	SSKIP	Skips the next block suspend action

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

19.5.6.1. Event Input Actions

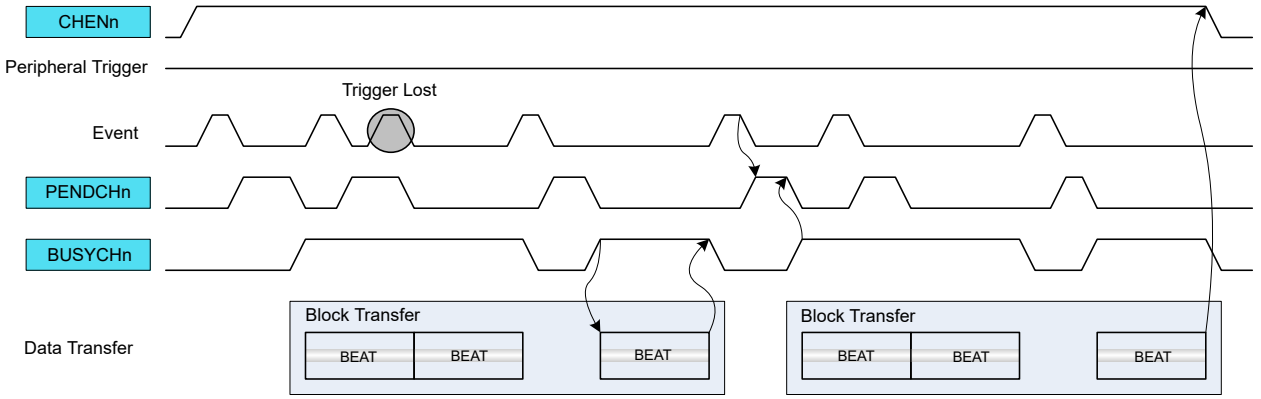
Normal Transfer

The event input is used to trigger a beat transfer on peripherals.

The event is acknowledged as soon as it is received. When received, both the Channel Pending status bit in the Channel Status (CHSTATUS.PEND) register and the corresponding Channel n bit in the Pending Channels (PENDCH.PENDCHn) register are set. If the event is received while the channel is already pending, the event trigger is lost.

Figure 19-13 shows an example where beat transfers are enabled by internal events.

Figure 19-13. Beat Event Trigger Action



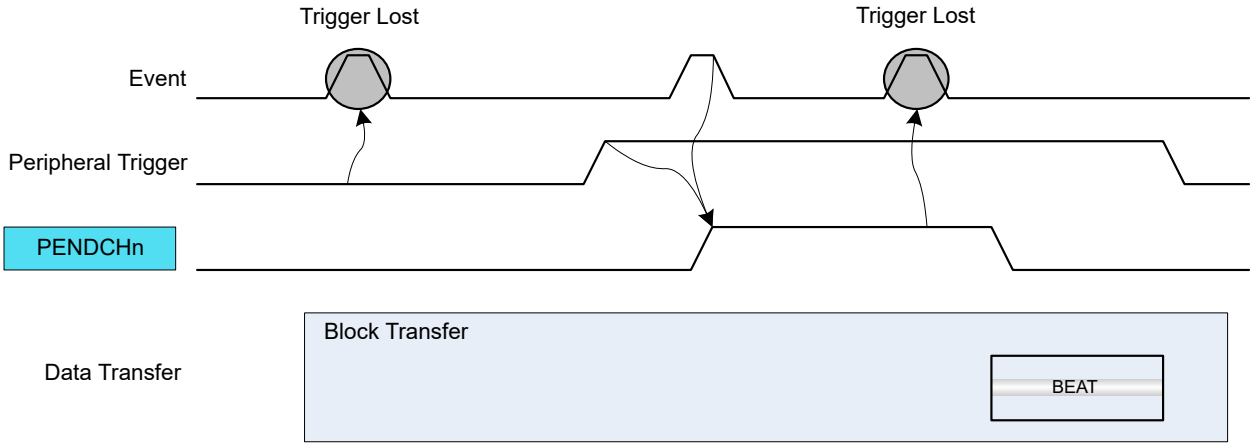
Conditional Transfer on Strobe

The event input is used to trigger a transfer on peripherals with pending transfer requests. This event action is intended to be used with peripheral triggers, such as those for timed communication protocols or periodic transfers between peripherals: the transfer is issued only when the peripheral trigger coincides with the occurrence of a (possibly cyclic) event.

The event is acknowledged as soon as it is received. The peripheral trigger request is stored internally when the previous trigger action is completed (i.e., the channel is not pending) and when an active event is received. If the peripheral trigger is active, the DMA will wait for an event before the peripheral trigger is internally registered. When both the event and the peripheral transfer trigger are active, both CHSTATUS.PEND and PENDCH.PENDCHn are set. A software trigger will now initiate a transfer.

Figure 19-14 shows an example where the peripheral beat transfer is started by a conditional strobe event action.

Figure 19-14. Periodic Event With Beat Peripheral Triggers



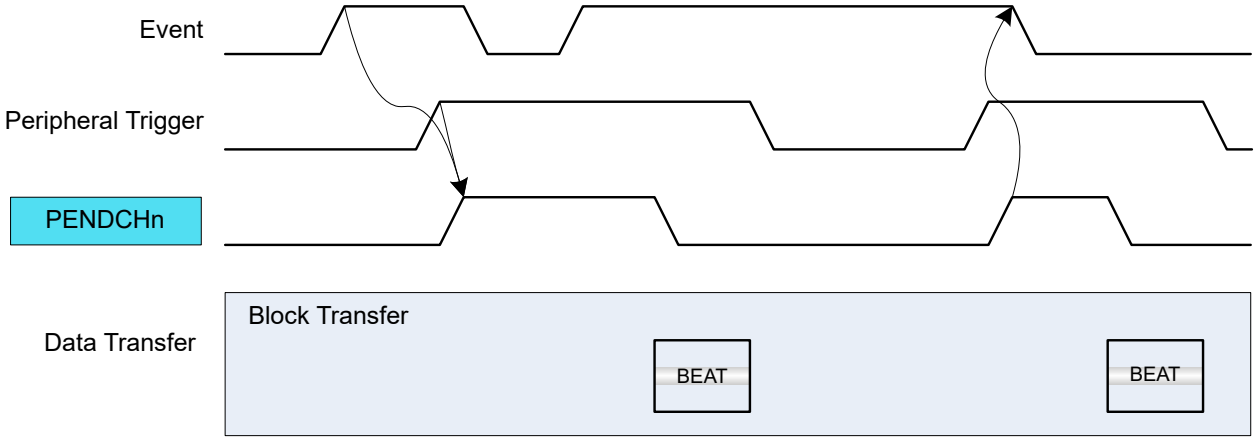
Conditional Transfer

The event input is used to trigger a conditional transfer on peripherals with pending transfer requests. For example, this type of event can be used for peripheral-to-peripheral transfers, where one peripheral is the source of the event and the second peripheral is the source of the trigger.

Each peripheral trigger is stored internally when the event is received. When the peripheral trigger is stored internally, the Channel Pending Status (CHSTATUS.PEND) bit is set, the respective Pending Channel n Bit in the Pending Channels (PENDCH.PENDCHn) register is set, and the event is acknowledged. A software trigger will now initiate a transfer.

Figure 19-15 shows an example where a conditional event is enabled with peripheral beat trigger requests.

Figure 19-15. Conditional Event With Beat Peripheral Triggers



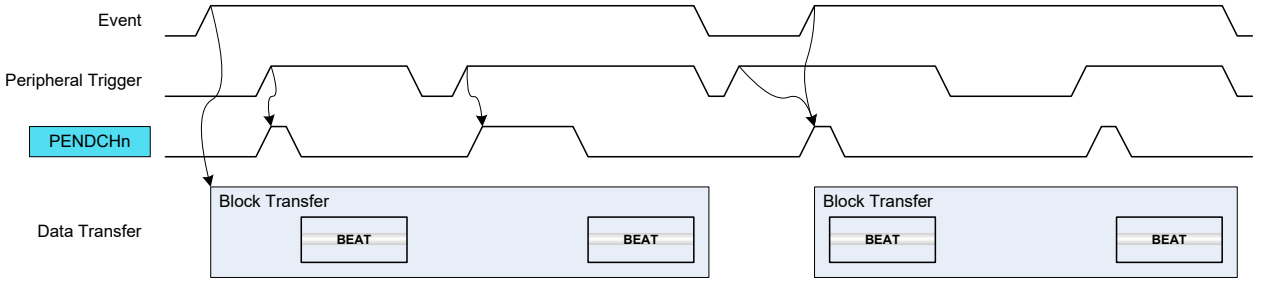
Conditional Block Transfer

The event input is used to trigger a conditional block transfer on peripherals.

Before starting transfers within a block, an event must be received. The event is acknowledged when the block transfer is completed. A software trigger will initiate a transfer.

Figure 19-16 shows an example where a conditional event block transfer is started with peripheral beat trigger requests.

Figure 19-16. Conditional Block Transfer With Beat Peripheral Triggers



Channel Suspend

The event input is used to suspend an ongoing channel operation. The event is acknowledged when the current AHB access is completed. Refer to the *Channel Suspend* section for further details.

Channel Resume

The event input is used to resume a suspended channel operation. The event is acknowledged as soon as it is received, and the Channel Suspend Interrupt (CHINTFLAG.SUSP) flag is cleared. Refer to the *Channel Suspend* section for further details.

Skip Next Block Suspend

This event can be used to skip the next block suspend action. If the channel is suspended before the event occurs, the channel operation is resumed and the event is acknowledged. If the event occurs before a suspend block action is detected, the event is held until the next block suspend detection. When the block transfer is completed, the channel continues operation (not suspended) and the event is acknowledged.

19.5.7. Analog Connections

Not applicable.

19.6. Register Summary - DMAC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRL	7:0						CRCENABLE	DMAENABLE	SWRST
		15:8					LVLEN3	LVLEN2	LVLEN1	LVLEN0
0x02	CRCCTRL	7:0					CRCPOLY[1:0]		CRCBEATSIZE[1:0]	
		15:8			CRCSRC[5:0]					
0x04	CRCDATAIN	7:0	CRCDATAIN[7:0]							
		15:8	CRCDATAIN[15:8]							
		23:16	CRCDATAIN[23:16]							
		31:24	CRCDATAIN[31:24]							
0x08	CRCCHKSUM	7:0	CRCCHKSUM[7:0]							
		15:8	CRCCHKSUM[15:8]							
		23:16	CRCCHKSUM[23:16]							
		31:24	CRCCHKSUM[31:24]							
0x0C	CRCSTATUS	7:0							CRCZERO	CRCBUSY
0x0D	DBGCTRL	7:0								DBGRUN
0x0E	QOSCTRL	7:0			DQOS[1:0]		FQOS[1:0]		WRBQOS[1:0]	
0x0F	Reserved									
0x10	SWTRIGCTRL	7:0							SWTRIG1	SWTRIG0
		15:8								
		23:16								
		31:24								
0x14	PRICTRL0	7:0	RRLVLEN0			LVLPRI0[4:0]				
		15:8	RRLVLEN1			LVLPRI1[4:0]				
		23:16	RRLVLEN2			LVLPRI2[4:0]				
		31:24	RRLVLEN3			LVLPRI3[4:0]				
0x18 ... 0x1F	Reserved									
0x20	INTPEND	7:0				ID[4:0]				
		15:8	PEND	BUSY	FERR			SUSP	TCMPL	TERR
0x22 ... 0x23	Reserved									
0x24	INTSTATUS	7:0							CHINT1	CHINT0
		15:8								
		23:16								
		31:24								
0x28	BUSYCH	7:0							BUSYCH1	BUSYCH0
		15:8								
		23:16								
		31:24								
0x2C	PENDCH	7:0							PENDCH1	PENDCH0
		15:8								
		23:16								
		31:24								
0x30	ACTIVE	7:0					LVLEX3	LVLEX2	LVLEX1	LVLEX0
		15:8	ABUSY			ID[4:0]				
		23:16	BTCNT[7:0]							
		31:24	BTCNT[15:8]							
0x34	BASEADDR	7:0	BASEADDR[7:0]							
		15:8	BASEADDR[15:8]							
		23:16	BASEADDR[23:16]							
		31:24	BASEADDR[31:24]							
0x38	WRBADDR	7:0	WRBADDR[7:0]							
		15:8	WRBADDR[15:8]							
		23:16	WRBADDR[23:16]							
		31:24	WRBADDR[31:24]							
0x3C ... 0x3E	Reserved									

DMAC (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x3F	CHID	7:0				ID[4:0]				
0x40	CHCTRLA	7:0		RUNSTDBY					ENABLE	SWRST
0x41 ... 0x43	Reserved									
0x44	CHCTRLB	7:0	LVL[2:0]			EVOE	EVIE	EVACT[2:0]		
		15:8	TRIGSRC[7:0]							
		23:16	TRIGACT[1:0]							
		31:24							CMD[1:0]	
0x48 ... 0x4B	Reserved									
0x4C	CHINTENCLR	7:0						SUSP	TCMPL	TERR
0x4D	CHINTENSET	7:0						SUSP	TCMPL	TERR
0x4E	CHINTFLAG	7:0						SUSP	TCMPL	TERR
0x4F	CHSTATUS	7:0						FERR	BUSY	PEND

19.6.1. Control

Name: CTRL
Offset: 0x00
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

Bit	15	14	13	12	11	10	9	8
					LVLEN3	LVLEN2	LVLEN1	LVLEN0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

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

Bits 8, 9, 10, 11 – LVLENn Priority Level n Enable
This bit controls whether requests with the corresponding level will be fed into the arbiter block. Refer to the *Arbitration* section for more details.
Note: This bit is not enable-protected.

Value	Description
0	Transfer requests for Priority level n will not be handled
1	Transfer requests for Priority level n will be handled

Bit 2 – CRCENABLE CRC Enable
This bit controls whether the CRC is enabled. Writing a '0' to this bit will disable the CRC calculation when the CRC Status Busy (CRCSTATUS. CRCBUSY) flag is cleared. The bit is '0' when the CRC is disabled.
Writing a '1' to this bit will enable CRC calculation.
Note: This bit is not enable-protected.

Value	Description
0	The CRC calculation is disabled
1	The CRC calculation is enabled

Bit 1 – DMAENABLE DMA Enable
This bit controls whether the DMA is enabled. Writing a '0' to this bit will disable the DMA module. If a '0' is written during an ongoing transfer, the bit will not be cleared until the internal data transfer buffer is empty and the DMA transfer is aborted. The internal data transfer buffer will be empty once the ongoing beat transfer is completed.
Note: This bit is not enable-protected.

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will always take precedence, meaning that all other writes in the same write-operation will be discarded.
Writing a '1' to this bit when both the DMAC and the CRC module are disabled (DMAENABLE and CRCENABLE are '0') resets all registers in the DMAC (except DBGCTRL) to their initial state. If either

the DMAC or CRC module is enabled, the Reset request will be ignored and the DMAC will return an access error.

Value	Description
0	There is no Reset operation ongoing
1	A Reset operation is ongoing

19.6.2. CRC Control

Name: CRCCTRL
Offset: 0x02
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

Bit	15	14	13	12	11	10	9	8
			CRCSRC[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

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

Bits 13:8 – CRCSRC[5:0] CRC Input Source
This bit field selects the input source for generating the CRC, as shown in the table below. The selected source is locked until either the CRC generation is completed or the CRC module is disabled. This means the CRCSRC cannot be modified while the CRC operation is ongoing. The lock is signaled by the CRCBUSY status bit. CRC generation complete is generated and signaled from the selected source when used with the DMA channel.

Value	Name	Description
0x00	NOACT	No action
0x01	IO	I/O interface
0x02-0x1F	—	Reserved
0x20	CHN0	DMA Channel 0
0x21	CHN1	DMA Channel 1
Other	—	Reserved

Bits 3:2 – CRCPOLY[1:0] CRC Polynomial Type
This bit field controls the polynomial type used to calculate the CRC checksum.

Value	Name	Description
0x0	CRC16	CRC-16 (CRC-CCITT)
0x1	CRC32	CRC32 (IEEE® 802.3)
Other	—	Reserved

Bits 1:0 – CRCBEATSIZE[1:0] CRC Beat Size
This bit field controls the size of the data transfer for each bus access when the CRC is used with the I/O interface.

Value	Name	Description
0x0	BYTE	8-bit bus transfer
0x1	HWORD	16-bit bus transfer
0x2	WORD	32-bit bus transfer
Other	—	Reserved

19.6.3. CRC Data Input

Name: CRCDATAIN
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
	CRCDATAIN[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
	CRCDATAIN[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
	CRCDATAIN[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
	CRCDATAIN[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 – CRCDATAIN[31:0] CRC Data Input
This bit field stores the data for which the CRC checksum is computed. A new CRC checksum becomes available in the CRC Checksum (CRCCHKSUM) register one clock cycle after writing to the CRCDATAIN register.

19.6.4. CRC Checksum

Name: CRCCHKSUM
Offset: 0x08
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

The CRCCHKSUM represents the 16- or 32-bit checksum value and the generated CRC. The register is reset to '0' by default, but it is possible to reset all bits to '1' by writing to the CRCCHKSUM register directly. It is possible to write this register only when the CRC module is disabled. If CRC-32 is selected and the CRC Status Busy flag is cleared (i.e., CRC generation is completed or aborted), the bit reversed (bit 31 is swapped with bit 0, bit 30 with bit 1, etc.) and complemented result will be read from CRCCHKSUM. If CRC-16 is selected or the CRC Status Busy flag is set (i.e., CRC generation is ongoing), CRCCHKSUM will contain the actual content.

Bit	31	30	29	28	27	26	25	24
	CRCCHKSUM[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
	CRCCHKSUM[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
	CRCCHKSUM[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
	CRCCHKSUM[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 – CRCCHKSUM[31:0] CRC Checksum
This bit field stores the generated CRC result. The 16 MSbs are always read zero when CRC-16 is enabled.

19.6.5. CRC Status

Name: CRCSTATUS
Offset: 0x0C
Reset: 0x00
Property: PAC Write-Protection

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

Bit 1 – CRCZERO CRC Zero
This bit is cleared when a new CRC source is selected.
This bit is set when the CRC generation is complete and the CRC Checksum is '0'.
When running CRC-32 and appending the checksum at the end of the packet (as little endian), the final checksum will be `0x2144df1c`. However, if the checksum is complemented before it is appended (as little endian) to the data, the final result in the checksum register will be zero. See the description of CRCCHKSUM to read out different versions of the checksum.

Bit 0 – CRCBUSY CRC Module Busy
This bit is cleared by writing a '0' to it when used with the I/O interface. When used with a DMA channel, the bit is set when the corresponding DMA channel is enabled, and cleared when the corresponding DMA channel is disabled. This bit cannot be written to while the CRC is used with a DMA channel.
This bit is set when a source configuration is selected in CRCSRC and as long as the source is using the CRC engine.

19.6.6. Debug Control

Name: DBGCTRL
Offset: 0x0D
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Debug Run
This bit controls the functionality when the CPU is halted by an external debugger.

Value	Description
0	The DMAC operation is suspended when the CPU is halted by an external debugger
1	The DMAC continues normal operation when the CPU is halted by an external debugger

19.6.7. Quality of Service Control

Name: QOSCTRL
Offset: 0x0E
Reset: 0x2A
Property: PAC Write-Protection

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

Bits 5:4 – DQOS[1:0] Data Transfer Quality of Service
This bit field controls the SRAM priority access during the data transfer operation. Refer to the *SRAM Quality of Service* section in the *CPU and Architecture* chapter for details.

Value	Name	Description
0x0	DISABLE	Background (no sensitive operation)
0x1	LOW	Sensitive Bandwidth
0x2	MEDIUM	Sensitive Latency
0x3	HIGH	Critical Latency

Bits 3:2 – FQOS[1:0] Fetch Quality of Service
This bit field controls the SRAM priority access during the fetch operation. Refer to the *SRAM Quality of Service* section in the *CPU and Architecture* chapter for details.

Value	Name	Description
0x0	DISABLE	Background (no sensitive operation)
0x1	LOW	Sensitive Bandwidth
0x2	MEDIUM	Sensitive Latency
0x3	HIGH	Critical Latency

Bits 1:0 – WRBQOS[1:0] Write-Back Quality of Service
This bit field controls the SRAM priority access during the write-back operation. Refer to the *SRAM Quality of Service* section in the *CPU and Architecture* chapter for details.

Value	Name	Description
0x0	DISABLE	Background (no sensitive operation)
0x1	LOW	Sensitive Bandwidth
0x2	MEDIUM	Sensitive Latency
0x3	HIGH	Critical Latency

19.6.8. Software Trigger Control

Name: SWTRIGCTRL
Offset: 0x10
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							SWTRIG1	SWTRIG0
Access							R/W	R/W
Reset							0	0

Bits 0, 1 – SWTRIGn Channel n Software Trigger
This bit is cleared when the Channel Pending bit in the Channel Status (CHSTATUS.PEND) register for the corresponding channel is either set, or by writing a '1' to it.
This bit is set if CHSTATUS.PEND is already '1' when writing a '1' to it.
Writing a '0' to this bit will clear the bit.
Writing a '1' to this bit will generate a DMA software trigger on channel n, if CHSTATUS.PEND = 0 for channel n. CHSTATUS.PEND will be set and SWTRIGn will remain cleared.

19.6.9. Priority Control 0

Name: PRICTRL0
Offset: 0x14
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
	RRLVLEN3			LVLPRI3[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RRLVLEN2			LVLPRI2[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RRLVLEN1			LVLPRI1[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RRLVLEN0			LVLPRI0[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bits 7, 15, 23, 31 – RRLVLENn Level n Round-Robin Arbitration Enable
This bit controls which arbitration scheme is selected for DMA channels with priority level n. Refer to the *Arbitration* section for details.

Value	Description
0	Static arbitration scheme for channels with level n priority
1	Round-robin arbitration scheme for channels with level n priority

Bits 0:4, 8:12, 16:20, 24:28 – LVLPRIn Level n Channel Priority Number
When round-robin arbitration is enabled (PRICTRL0.RRLVLENn = 1) for priority level n, this register holds the channel number of the last DMA channel being granted access as the active channel with priority level n.
When static arbitration is enabled (PRICTRL0.RRLVLENn = 0) for priority level n, and the value of this bit field is non-zero, it will not affect the static priority scheme.
This bit field is not reset when round-robin arbitration gets disabled (PRICTRL0.RRLVLENn written to '0').

Value	Name	Description
0x00	CHN0	DMA Channel 0
0x01	CHN1	DMA Channel 1
Other	—	Reserved

19.6.10. Interrupt Pending

Name: INTPEND
Offset: 0x20
Reset: 0x0000
Property: –

This register allows the user to identify the lowest DMA channel with pending interrupt.

Bit	15	14	13	12	11	10	9	8
	PEND	BUSY	FERR			SUSP	TCMPL	TERR
Access	R	R	R			R/W	R/W	R/W
Reset	0	0	0			0	0	0

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

Bit 15 – PEND Pending
This bit is set when the channel selected by Channel ID field (ID) is pending.
This bit is otherwise cleared.

Bit 14 – BUSY Busy
This bit is set when the channel selected by Channel ID field (ID) is busy.
This bit is otherwise cleared.

Bit 13 – FERR Fetch Error
This bit is set when the channel selected by Channel ID field (ID) fetched an invalid descriptor.
This bit is otherwise cleared.

Bit 10 – SUSP Channel Suspend
This bit is cleared by writing a '1' to it.
This bit is set when the channel selected by Channel ID field (ID) has pending Suspend interrupt.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel ID (ID) Suspend interrupt flag.

Bit 9 – TCMPL Transfer Complete
This bit is cleared by writing a '1' to it.
This bit is set when the channel selected by Channel ID field (ID) has a pending Transfer Complete interrupt.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel ID (ID) Transfer Complete interrupt flag.

Bit 8 – TERR Transfer Error
This bit is cleared by writing a '1' to it.
This bit is set when the channel selected by Channel ID field (ID) has a pending Transfer Error interrupt.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel ID (ID) Transfer Error interrupt flag.

Bits 4:0 – ID[4:0] Channel ID
This bit field stores the lowest channel number with pending interrupts. The number is valid if the Suspend (SUSP), Transfer Complete (TCMPL) or Transfer Error (TERR) bits are set. The Channel ID

field is refreshed when a new channel (with a channel number lower than the current one) with pending interrupts is detected, or when the application clears the corresponding channel interrupt sources. When no pending channels interrupts are available, this bit field will always return a zero value when read.
When this bit field is written, indirect access to the corresponding Channel Interrupt Flag Status and Clear (CHINTFLAG) register is enabled.

Value	Name	Description
0x00	CHN0	DMA Channel 0
0x01	CHN1	DMA Channel 1
Other	—	Reserved

19.6.11. Interrupt Status

Name: INTSTATUS
Offset: 0x24
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							CHINT1	CHINT0
Access							R	R
Reset							0	0

Bits 0, 1 – CHINTn Channel n Pending Interrupt
This bit is set when Channel n has a pending interrupt/the interrupt request is received.
This bit is cleared when the corresponding Channel n interrupts are disabled or the interrupts sources are cleared.

19.6.12. Busy Channels

Name: BUSYCH
Offset: 0x28
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							BUSYCH1	BUSYCH0
Access							R	R
Reset							0	0

Bits 0, 1 – BUSYCHn Busy Channel n
This bit is cleared when the channel trigger action for DMA channel n is complete, when a bus error for DMA channel n is detected, or when DMA channel n is disabled.
This bit is set when DMA channel n starts a DMA transfer.

19.6.13. Pending Channels

Name: PENDCH
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							PENDCH1	PENDCH0
Access							R	R
Reset							0	0

Bits 0, 1 – PENDCHn Pending Channel n
This bit is cleared when trigger execution, as defined by the channel trigger action settings for DMA channel n, is initiated when a bus error for DMA channel n is detected or when DMA channel n is disabled. For details on trigger action settings, refer to CHCTRLB.TRIGACT.
This bit is set when a transfer is pending on DMA channel n.

19.6.14. Active Channel and Levels

Name: ACTIVE
Offset: 0x30
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	BTCNT[15:8]							
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
	BTCNT[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
	ABUSY			ID[4:0]				
Access	R			R	R	R	R	R
Reset	0			0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
					LVLEX3	LVLEX2	LVLEX1	LVLEX0
Access					R	R	R	R
Reset					0	0	0	0

Bits 31:16 – BTCNT[15:0] Active Channel Block Transfer Count
This bit field holds the 16-bit block transfer count of the ongoing transfer. This value is stored in the active channel and written back in the corresponding Write-Back channel memory location when the arbiter grants a new channel access. The value is valid only when the active channel Active Busy (ABUSY) flag is set.

Bit 15 – ABUSY Active Channel Busy
This bit is cleared when the active transfer count is written back in the write-back memory section.
This bit is set when the next descriptor transfer count is read from the write-back memory section.

Bits 12:8 – ID[4:0] Active Channel ID
This bit field holds the channel index currently stored in the active channel registers. The value is updated each time the arbiter grants a new channel transfer access request.

Value	Name	Description
0x00	CHN0	DMA Channel 0
0x01	CHN1	DMA Channel 1
Other	—	Reserved

Bits 0, 1, 2, 3 – LVLEXn Level n Channel Trigger Request Executing
This bit is set when a level n channel trigger request is executing or pending.
This bit is cleared when no request is pending or being executed.

19.6.15. Descriptor Memory Section Base Address

Name: BASEADDR
Offset: 0x34
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
	BASEADDR[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
	BASEADDR[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
	BASEADDR[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
	BASEADDR[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 – BASEADDR[31:0] Descriptor Memory Base Address
This bit field stores the Descriptor memory section base address. The value must be 128-bit aligned.

19.6.16. Write-Back Memory Section Base Address

Name: WRBADDR
Offset: 0x38
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
	WRBADDR[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
	WRBADDR[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
	WRBADDR[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
	WRBADDR[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 – WRBADDR[31:0] Write-Back Memory Base Address
This bit field stores the Write-Back memory base address. The value must be 128-bit aligned.

19.6.17. Channel ID

Name: CHID
Offset: 0x3F
Reset: 0x00
Property: –

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

Bits 4:0 – ID[4:0] Channel ID
This bit field controls the channel number that will be affected by the channel registers (CH*). Before reading or writing a channel register, the Channel ID bit field must be written first.

Value	Name	Description
0x00	CHN0	DMA Channel 0
0x01	CHN1	DMA Channel 1
Other	—	Reserved

19.6.18. Channel Control A

Name: CHCTRLA
Offset: 0x40
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

This register affects the DMA channel selected in the Channel ID (CHID.ID) register.

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

Bit 6 – RUNSTDBY Channel run in standby
This bit is used to keep the DMAC channel running in Standby sleep mode.
Note: This bit is not enable-protected.

Value	Description
0	The DMAC channel is halted in Standby sleep mode
1	The DMAC channel continues to run in Standby sleep mode

Bit 1 – ENABLE Channel Enable
When writing a '0' to this bit during an ongoing transfer, the bit will not be cleared until the internal data transfer buffer is empty and the DMA transfer is aborted. The internal data transfer buffer will be empty once the ongoing beat transfer is completed.
Writing a '1' to this bit will enable the DMA channel.
Note: This bit is not enable-protected.

Value	Description
0	DMA channel is disabled
1	DMA channel is enabled

Bit 0 – SWRST Channel Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets the channel registers to their initial state. The bit can be set when the channel is disabled (ENABLE = 0). Writing a '1' to this bit will be ignored as long as ENABLE = 1. This bit is automatically cleared when the reset is completed.

Value	Description
0	There is no reset operation ongoing
1	The reset operation is ongoing

19.6.19. Channel Control B

Name: CHCTRLB
Offset: 0x44
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

This register affects the DMA channel selected in the Channel ID (CHID.ID) register.

Bit	31	30	29	28	27	26	25	24
							CMD[1:0]	
Access							R/W	R/W
Reset							0	0

Bit	23	22	21	20	19	18	17	16
	TRIGACT[1:0]							
Access	R/W	R/W						
Reset	0	0						

Bit	15	14	13	12	11	10	9	8
	TRIGSRC[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
	LVL[2:0]			EVOE	EVIE	EVACT[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 25:24 – CMD[1:0] Software Command
This bit field controls the software commands. Refer to Channel Suspend and Channel Resume and Next Suspend Skip.
Note: This bit field is not enable-protected.

Value	Name	Description
0x0	NOACT	No action
0x1	SUSPEND	Channel suspend operation
0x2	RESUME	Channel resume operation
0x3	—	Reserved

Bits 23:22 – TRIGACT[1:0] Trigger Action
This bit field controls the trigger action used for a transfer.
Note: This bit field is enable-protected.

Value	Name	Description
0x0	BLOCK	One trigger required for each block transfer
0x1	—	Reserved
0x2	BEAT	One trigger required for each beat transfer
0x3	TRANSACTION	One trigger required for each transaction

Bits 15:8 – TRIGSRC[7:0] Trigger Source
This bit field controls the peripheral trigger which is source of the transfer. Refer to the *Transfer Triggers and Actions* section and Trigger Action (TRIGACT) bit field in the Channel Control B (CHCTRLB) register for details on trigger selection and trigger modes.

Value	Name	Description
0x00	DISABLE	Only software/event triggers
0x01	SERCOM0_RX	SERCOM0 Receive
0x02	SERCOM0_TX	SERCOM0 Transmit
0x03	SERCOM1_RX	SERCOM1 Receive
0x04	SERCOM1_TX	SERCOM1 Transmit
0x05	TC0_OVF	TC0 Overflow
0x06	TC0_MC0	TC0 Match or Capture Channel 0
0x07	TC0_MC1	TC0 Match or Capture Channel 1
0x08	TC1_OVF	TC1 Overflow
0x09	TC1_MC0	TC1 Match or Capture Channel 0
0x0A	TC1_MC1	TC1 Match or Capture Channel 1
0x0B	TC2_OVF	TC2 Overflow
0x0C	TC2_MC0	TC2 Match or Capture Channel 0
0x0D	TC2_MC1	TC2 Match or Capture Channel 1
0x0E	TCC0_OVF	TCC0 Overflow
0x0F	TCC0_MC0	TCC0 Match or Capture Channel 0
0x10	TCC0_MC1	TCC0 Match or Capture Channel 1
0x11	TCC0_MC2	TCC0 Match or Capture Channel 2
0x12	TCC0_MC3	TCC0 Match or Capture Channel 3
0x13	ADC_RESRDY	ADC Result Ready
0x14	ADC_SAMPRDY	ADC Sample Ready
0x15	DSU_DCC0	DSU Debug Communication Channel 0
0x16	DSU_DCC1	DSU Debug Communication Channel 1
0x17-0xFF	-	Reserved

Bits 7:5 – LVL[2:0] Channel Arbitration Level

This bit field controls the arbitration level used for the DMA channel, where a high level has priority over a low level. Refer to the *Arbitration* section for further details on arbitration schemes.

Note: This bit field is not enable-protected.

Value	Name	Description
0x0	LVL0	Channel priority level 0
0x1	LVL1	Channel priority level 1
0x2	LVL2	Channel priority level 2
0x3	LVL3	Channel priority level 3
Other	—	Reserved

Bit 4 – EVOE Channel Event Output Enable

This bit indicates if the Channel event generation is enabled. The event will be generated for every condition defined in the descriptor Event Output Selection (EVOSEL) bit field in the Block Transfer Control (BTCTRL) register.

This bit is available only for the lower-numbered DMA channels. Refer to the *EVSYS – Event System* chapter for details on event generators.

Value	Description
0	Channel event generation is disabled
1	Channel event generation is enabled

Bit 3 – EVIE Channel Event Input Enable

This bit is available only for the lower-numbered DMA channels. Refer to the *EVSYS – Event System* chapter for details on event users.

Value	Description
0	Channel event action will not be executed on any incoming event
1	Channel event action will be executed on any incoming event

Bits 2:0 – EVACT[2:0] Event Input Action

This bit field controls the event input action, as shown below. The action is executed only if the corresponding EVIE bit in the CHCTRLB register of the channel is set.
This bit is available only for the lower-numbered DMA channels. Refer to the *EVSYS – Event System* chapter for details on event generators.

Value	Name	Description
0x0	NOACT	No action
0x1	TRIG	Normal transfer and conditional transfer on strobe trigger
0x2	CTRIG	Conditional transfer trigger
0x3	CBLOCK	Conditional block transfer
0x4	SUSPEND	Channel suspend operation
0x5	RESUME	Channel resume operation
0x6	SSKIP	Skip next block suspend action
0x7	—	Reserved

19.6.20. Channel Interrupt Enable Clear

Name: CHINTENCLR
Offset: 0x4C
Reset: 0x00
Property: PAC Write-Protection

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 Channel Interrupt Enable Set (CHINTENSET) register.
This register affects the DMA channel selected in the Channel ID (CHID.ID) register.

Bit	7	6	5	4	3	2	1	0
						SUSP	TCMPL	TERR
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – SUSP Suspend Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel Suspend Interrupt Enable bit, which disables the Channel Suspend interrupt.

Value	Description
0	The Channel Suspend interrupt is disabled
1	The Channel Suspend interrupt is enabled

Bit 1 – TCMPL Transfer Complete Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel Transfer Complete Interrupt Enable bit, which disables the Channel Transfer Complete interrupt.

Value	Description
0	The Channel Transfer Complete interrupt is disabled. When block action is set to none, the TCMPL flag will not be set when a block transfer is completed.
1	The Channel Transfer Complete interrupt is enabled

Bit 0 – TERR Transfer Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel Transfer Error Interrupt Enable bit, which disables the Channel Transfer Error interrupt.

Value	Description
0	The Channel Transfer Error interrupt is disabled
1	The Channel Transfer Error interrupt is enabled

19.6.21. Channel Interrupt Enable Set

Name: CHINTENSET
Offset: 0x4D
Reset: 0x00
Property: PAC Write-Protection

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 Channel Interrupt Enable Clear (CHINTENCLR) register.
This register affects the DMA channel selected in the Channel ID (CHID.ID) register.

Bit	7	6	5	4	3	2	1	0
						SUSP	TCMPL	TERR
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – SUSP Suspend Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Channel Suspend Interrupt Enable bit, which enables the Channel Suspend interrupt.

Value	Description
0	The Channel Suspend interrupt is disabled
1	The Channel Suspend interrupt is enabled

Bit 1 – TCMPL Transfer Complete Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Channel Transfer Complete Interrupt Enable bit, which enables the Channel Transfer Complete interrupt.

Value	Description
0	The Channel Transfer Complete interrupt is disabled
1	The Channel Transfer Complete interrupt is enabled

Bit 0 – TERR Transfer Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Channel Transfer Error Interrupt Enable bit, which enables the Channel Transfer Error interrupt.

Value	Description
0	The Channel Transfer Error interrupt is disabled
1	The Channel Transfer Error interrupt is enabled

19.6.22. Channel Interrupt Flag Status and Clear

Name: CHINTFLAG
Offset: 0x4E
Reset: 0x00
Property: –

This register affects the DMA channel selected in the Channel ID (CHID.ID) register.

Bit	7	6	5	4	3	2	1	0
						SUSP	TCMPL	TERR
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – SUSP Suspend
This flag is cleared by writing a '1' to it.
This flag is set when a block transfer with suspend block action is completed, when a software Suspend command is executed, when a suspend event is received or when an invalid descriptor is fetched by the DMA.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Channel Suspend interrupt flag for the corresponding channel.
For details on available software commands, refer to CHCTRLB.CMD.
For details on available event input actions, refer to CHCTRLB.EVACT.
For details on available block actions, refer to BTCTRL.BLOCKACT.

Bit 1 – TCMPL Transfer Complete
This flag is cleared by writing a '1' to it.
This flag is set when a block transfer is completed and the corresponding interrupt block action is enabled.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Transfer Complete interrupt flag for the corresponding channel.

Bit 0 – TERR Transfer Error
This flag is cleared by writing a '1' to it.
This flag is set when a bus error is detected during a beat transfer or when the DMAC fetches an invalid descriptor.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Transfer Error interrupt flag for the corresponding channel.

19.6.23. Channel Status

Name: CHSTATUS
Offset: 0x4F
Reset: 0x00
Property: –

This register affects the DMA channel selected in the Channel ID (CHID.ID) register.

Bit	7	6	5	4	3	2	1	0
						FERR	BUSY	PEND
Access						R	R	R
Reset						0	0	0

Bit 2 – FERR Fetch Error
This bit is cleared when a Resume software command is executed by writing to the Command bit field in the Channel Control B (CHCTRLB.CMD) register.
This bit is set when an invalid descriptor is fetched.

Bit 1 – BUSY Busy
This bit is cleared when the channel trigger action is completed, when a bus error is detected or when the channel is disabled.
This bit is set when the DMA channel starts a DMA transfer.

Bit 0 – PEND Pending
This bit is cleared when the channel trigger action is started, when a bus error is detected or when the channel is disabled. For details on trigger action settings, refer to CHCTRLB.TRIGACT.
This bit is set when a transfer is pending on the DMA channel, as soon as the transfer request is received.

19.7. Register Summary - DMAC_SRAM

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	BTCTRL	7:0				BLOCKACT[1:0]		EVOSEL[1:0]		VALID
		15:8	STEPSIZE[2:0]			STEPSEL	DSTINC	SRCINC	BEATSIZE[1:0]	
0x02	BTCNT	7:0	BTCNT[7:0]							
		15:8	BTCNT[15:8]							
0x04	SRCADDR	7:0	SRCADDR[7:0]							
		15:8	SRCADDR[15:8]							
		23:16	SRCADDR[23:16]							
		31:24	SRCADDR[31:24]							
0x08	DSTADDR	7:0	DSTADDR[7:0]							
		15:8	DSTADDR[15:8]							
		23:16	DSTADDR[23:16]							
		31:24	DSTADDR[31:24]							
0x0C	DESCADDR	7:0	DESCADDR[7:0]							
		15:8	DESCADDR[15:8]							
		23:16	DESCADDR[23:16]							
		31:24	DESCADDR[31:24]							

19.7.1. Block Transfer Control

Name: BTCTRL
Offset: 0x00
Reset: 0x0000
Property: –

The BTCTRL register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10.

Bit	15	14	13	12	11	10	9	8
	STEPSIZE[2:0]			STEPSEL	DSTINC	SRCINC	BEATSIZE[1:0]	
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
				BLOCKACT[1:0]		EVOSEL[1:0]		VALID
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 15:13 – STEPSIZE[2:0] Address Increment Step Size
This bit field selects the step size of the address increment. The setting applies to the to source or destination address, depending on the STEPSEL setting.

Value	Name	Description
0x0	X1	Next ADDR = ADDR + (Beat size in byte) * 1
0x1	X2	Next ADDR = ADDR + (Beat size in byte) * 2
0x2	X4	Next ADDR = ADDR + (Beat size in byte) * 4
0x3	X8	Next ADDR = ADDR + (Beat size in byte) * 8
0x4	X16	Next ADDR = ADDR + (Beat size in byte) * 16
0x5	X32	Next ADDR = ADDR + (Beat size in byte) * 32
0x6	X64	Next ADDR = ADDR + (Beat size in byte) * 64
0x7	X128	Next ADDR = ADDR + (Beat size in byte) * 128

Bit 12 – STEPSEL Step Selection
This bit controls if the source or destination addresses are using the step size settings.

Value	Name	Description
0x0	DST	Step size settings apply to the destination address
0x1	SRC	Step size settings apply to the source address

Bit 11 – DSTINC Destination Address Increment Enable
This bit controls whether the destination address increments between beat transfers or not. When this bit is set, the destination address is by default incremented by one beat size in byte(s). If the STEPSEL bit is '0', the STEPSIZE register will control the size of the increments.

Value	Description
0	The destination address increment is disabled
1	The destination address increment is enabled

Bit 10 – SRCINC Source Address Increment Enable
This bit controls whether the source address increments between beat transfers or not. When this bit is set, the source address is by default incremented by one beat size in byte(s). If the STEPSEL bit is '1', the STEPSIZE register will control the size of the increments.

Value	Description
0	The source address increment is disabled

Value	Description
1	The source address increment is enabled

Bits 9:8 – BEATSIZE[1:0] Beat Size
This bit field controls the size of one beat. A beat is the size of one data transfer bus access, and the setting applies to both read and write accesses.

Value	Name	Description
0x0	BYTE	8-bit
0x1	HWORD	16-bit
0x2	WORD	32-bit
Other	—	Reserved

Bits 4:3 – BLOCKACT[1:0] Block Action
This bit field controls what actions the DMAC will take after a block transfer has completed.

Value	Name	Description
0x0	NOACT	Channel will be disabled if it is the last block transfer in the transaction
0x1	INT	Channel will be disabled if it is the last block transfer in the transaction and block interrupt
0x2	SUSPEND	Channel suspend operation is completed
0x3	BOTH	Both channel suspend operation and block interrupt

Bits 2:1 – EVOSEL[1:0] Event Output Selection
This bit field controls the event output selection.

Value	Name	Description
0x0	DISABLE	Event generation disabled
0x1	BLOCK	Event strobe when block transfer complete
0x2	—	Reserved
0x3	BEAT	Event strobe when beat transfer complete

Bit 0 – VALID Descriptor Valid
Writing a '0' to this bit in the Descriptor or Write-Back memory will suspend the DMA channel operation when fetching the corresponding descriptor.
The bit is automatically cleared in the Write-Back memory section when the channel is aborted, when an error is detected during the block transfer, or when the block transfer is completed.

Value	Description
0	The descriptor is not valid
1	The descriptor is valid

19.7.2. Block Transfer Count

Name: BTCNT
Offset: 0x02
Reset: 0x0000
Property: –

The BTCNT register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10

Bit	15	14	13	12	11	10	9	8
	BTCNT[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
	BTCNT[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 – BTCNT[15:0] Block Transfer Count
This bit field holds the 16-bit block transfer count.
During a transfer, the internal counter value is decremented by one after each beat transfer. The internal counter is written to the corresponding Write-Back memory section for the DMA channel when the DMA channel loses priority, is suspended or gets disabled. The DMA channel can be disabled by a complete transfer, a transfer error or by software.

19.7.3. Block Transfer Source Address

Name: SRCADDR
Offset: 0x04
Reset: 0x00000000
Property: –

The SRCADDR register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10.

Bit	31	30	29	28	27	26	25	24
	SRCADDR[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
	SRCADDR[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
	SRCADDR[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
	SRCADDR[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 – SRCADDR[31:0] Transfer Source Address

This bit field holds the block transfer source address.
When source address incrementation is disabled (BTCTRL.SRCINC = 0), SRCADDR corresponds to the last beat transfer address in the block transfer.
When source address incrementation is enabled (BTCTRL.SRCINC = 1), SRCADDR must be set to the ending address and is calculated as follows:
If BTCTRL.STEPSEL = 1:

$$\mathrm{SRCADDR} = \mathrm{SRCADDR}_{\mathrm{START}} \cdot \mathrm{BTCNT} + \mathrm{BiB} \cdot 2^{\mathrm{STEPSIZE}}$$

If BTCTRL.STEPSEL = 0:

$$\mathrm{SRCADDR} = \mathrm{SRCADDR}_{\mathrm{START}} \cdot \mathrm{BTCNT} + \mathrm{BiB}$$

Where,

- $\mathrm{SRCADDR}_{\mathrm{START}}$ is the source address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BiB is the configured number of bytes in a beat, as controlled by the BEATSIZE bit field.
- STEPSIZE is the configured number of beats for each incrementation

19.7.4. Block Transfer Destination Address

Name: DSTADDR
Offset: 0x08
Reset: 0x00000000
Property: –

The DSTADDR register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10.

Bit	31	30	29	28	27	26	25	24
	DSTADDR[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
	DSTADDR[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
	DSTADDR[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
	DSTADDR[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 – DSTADDR[31:0] Transfer Destination Address
This bit field holds the block transfer destination address.
When destination address incrementation is disabled (BTCTRL.DSTINC = 0), DSTADDR corresponds to the last beat transfer address in the block transfer.
When destination address incrementation is enabled (BTCTRL.DSTINC = 1), DSTADDR must be set to the ending address and calculated as follows:
If BTCTRL.STEPSEL = 1:

$$DSTADDR = DSTADDR_{START} \cdot BTCNT + BiB$$

If BTCTRL.STEPSEL = 0:

$$DSTADDR = DSTADDR_{START} \cdot BTCNT + BiB \cdot 2^{STEPSIZE}$$

Where,

- $DSTADDR_{START}$ is the destination address of the first beat transfer in the block transfer
- BTCNT is the initial number of beats remaining in the block transfer
- BiB is the configured number of bytes in a beat, as controlled by the BEATSIZE bit field.
- STEPSIZE is the configured number of beats for each incrementation

19.7.5. Next Descriptor Address

Name: DESCADDR
Offset: 0x0C
Reset: 0x00000000
Property: –

The DESCADDR register offset is relative to (BASEADDR or WRBADDR) + Channel Number * 0x10.

Bit	31	30	29	28	27	26	25	24
	DESCADDR[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
	DESCADDR[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
	DESCADDR[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
	DESCADDR[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 – DESCADDR[31:0] Next Descriptor Address
This bit field holds the SRAM address of the next descriptor. The value must be 128-bit aligned. If the value of this SRAM register is 0x00000000, the transaction will be terminated when the DMAC tries to load the next transfer descriptor.

20. WDT – Watchdog Timer

20.1. Features

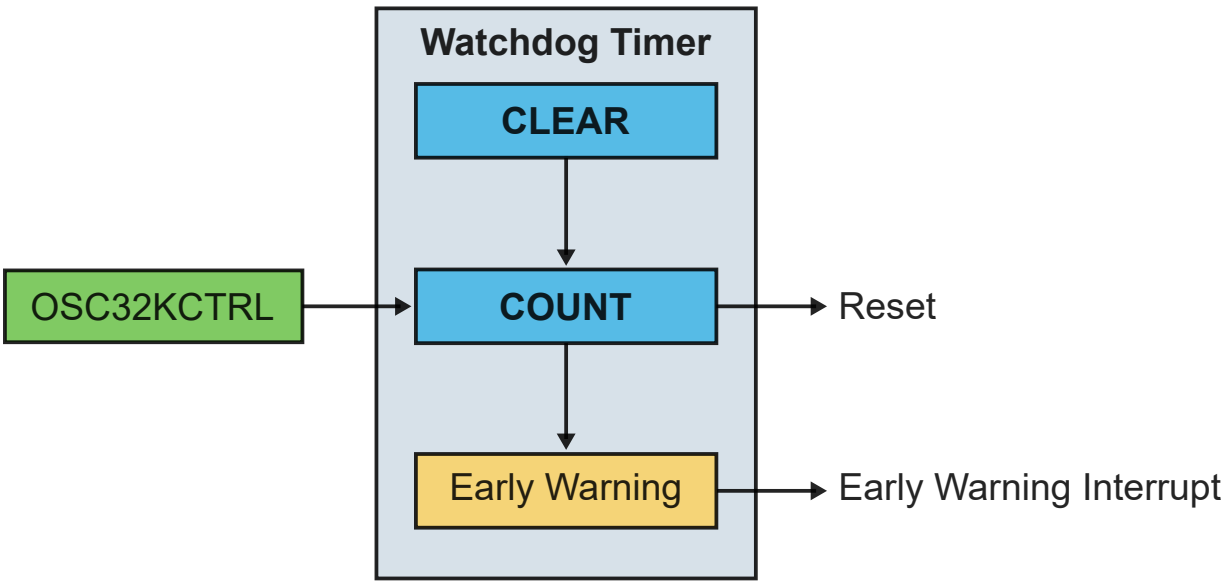
- Issues a System Reset If the Watchdog Timer Is Not Cleared Before Its Time-Out Period
- Early Warning Interrupt Generation
- Asynchronous Operation From Dedicated Oscillator
- Two Types of Operation:
  - Normal mode
  - Window mode
- Selectable Time-Out Periods
  - From eight cycles to 16,384 cycles in Normal mode
  - From 16 cycles to 32,768 cycles in Window mode
- Always-On Capability

20.2. 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 by having the WDT issue a system Reset request.

20.3. Block Diagram

Figure 20-1. WDT Block Diagram



20.3.1. Signal Description

Not applicable.

20.4. Functional Description

20.4.1. Initialization

Normal Mode

Before the WDT is enabled in Normal mode, it must be configured by following these steps:

1. Select the time-out period by writing the Time-Out Period bit field in the Configuration register (CONFIG.PER).
2. Enable the WDT by writing the Enable bit in the Control A register (CTRLA.ENABLE).

Window Mode

Before the WDT is enabled in Window mode, it must be configured by following these steps:

1. Select the time-out period by writing the Time-Out Period bit field in the Configuration register (CONFIG.PER).
2. Select the closed window period by writing the Window Mode Time-Out Period bit field in the Configuration register (CONFIG.WINDOW).
3. Enable the Window mode by writing a '1' to the Window Enable bit in the Control A register (CTRLA.WEN).
4. Enable the WDT by writing the Enable bit in the Control A register (CTRLA.ENABLE).

20.4.2. Operation

The WDT is a continuously running counter configured to a predefined time-out period. Before the end of the time-out period, the WDT must be cleared, otherwise, a system Reset request is issued.

The WDT can be configured in Normal mode or Window mode, both offering the option of early warning interrupt generation. The settings in the Control A (CTRLA) register and the Interrupt Enable register (handled by INTENCLR/INTENSET) determine the mode of operation. The description for each of the modes is given in the following sections.

Table 20-1. WDT Operating Modes

Mode	CTRLA.ENABLE	CTRLA.WEN	Interrupt Enable
Stopped	0	x	x
Normal mode	1	0	0
Normal mode with early warning interrupt	1	0	1
Window mode	1	1	0
Window mode with early warning interrupt	1	1	1

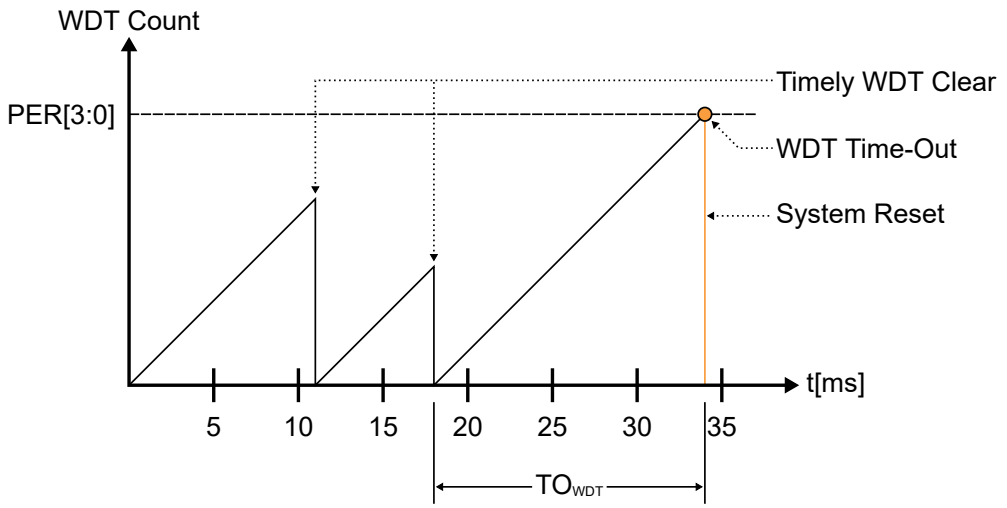
When enabled, the WDT will run in active mode and all sleep modes. It is asynchronous and runs from a CPU-independent clock source. The WDT will continue operation and issue a system Reset request or interrupt even if the main clocks fail.

20.4.2.1. Normal Mode

In Normal mode operation, the length of the time-out period is configured in the Time-Out Period bit field in the Configuration register (CONFIG.PER). There are 12 possible WDT time-out (TO_{WDT}) periods, selectable from 8 ms to 16s. Once enabled, the WDT will issue a system Reset request if a time-out occurs. This can be prevented by clearing the WDT at any time during the time-out period.

The WDT is cleared, and a new WDT time-out period is started by writing `0xA5` to the Clear (CLEAR) register. Writing any value other than `0xA5` to CLEAR will issue an immediate system Reset request.

Figure 20-2. Normal Mode Operation



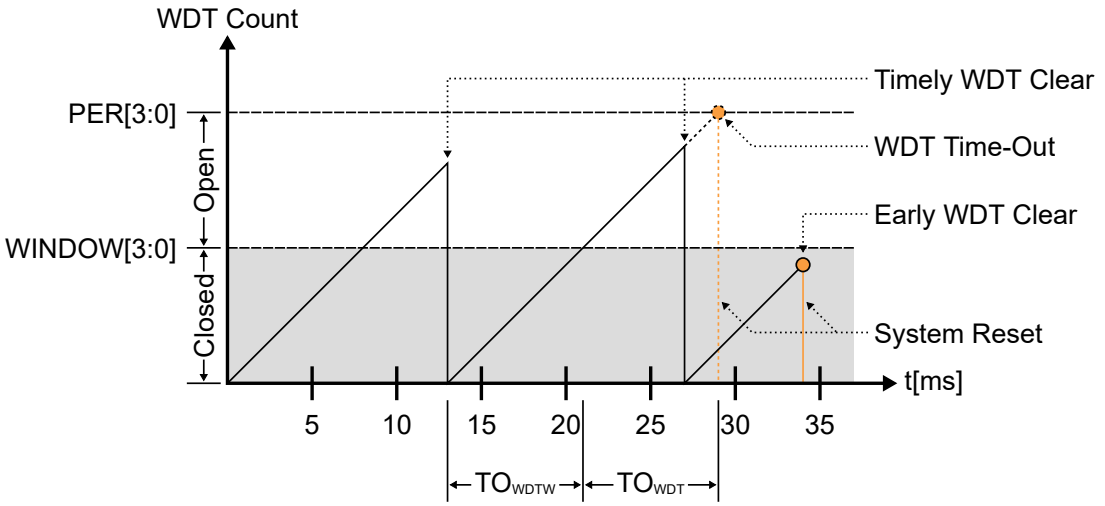
20.4.2.2. Window Mode

In Window mode operation, the WDT uses two different time specifications: The WDT can only be cleared by writing `0xA5` to the CLEAR register *after* the closed window time-out period (TO_{WDTW}), during the subsequent Normal time-out period (TO_{WDT}). If the WDT is cleared before the time window opens (before TO_{WDTW} is over), the WDT will issue a system Reset request.

The TO_{WDTW} and TO_{WDT} parameters are periods that range from 8 ms to 16s. The total duration of the WDT time-out period is the sum of the two parameters.

The closed window period is defined by the Window Mode Time-Out Period bit field in the Configuration register (CONFIG.WINDOW), and the open window period is defined by the Time-Out Period bit field in the Configuration register (CONFIG.PER).

Figure 20-3. Window Mode Operation



20.4.2.3. Enabling, Disabling, and Resetting

The WDT is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The WDT is disabled by writing a '0' to CTRLA.ENABLE.

The WDT can be disabled only if the Always-On bit in the Control A register (CTRLA.ALWAYSON) is '0'.

20.4.2.4. Configurable Reset Values

After a Power-On Reset (POR), some registers will be loaded with the initial values from the *BOOTCFG - Boot Configuration Fuses* section. This includes the following bits and bit fields:

- Enable bit in the Control A register (CTRLA.ENABLE)
- Always-On bit in the Control A register (CTRLA.ALWAYSON)
- Watchdog Timer Window Mode Enable bit in the Control A register (CTRLA.WEN)
- Watchdog Timer Window Mode Time-Out Period bits in the Configuration register (CONFIG.WINDOW)
- Time-Out Period bits in the Configuration register (CONFIG.PER)
- Early Warning Interrupt Time Offset bits in the Early Warning Interrupt Control register (EWCTRL.EWOFFSET)

20.4.3. Additional Features

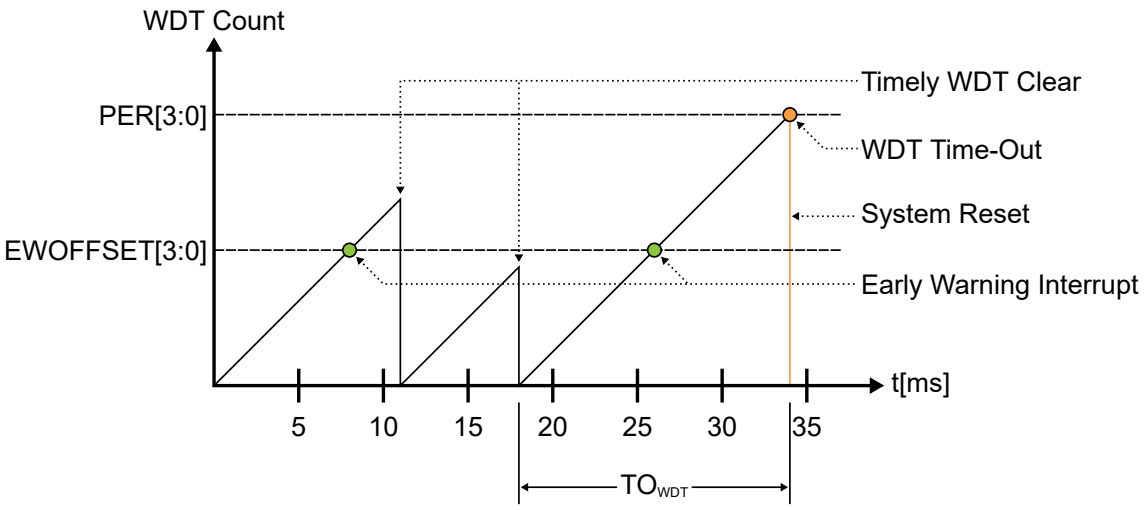
20.4.3.1. Early Warning

The early warning feature notifies with an interrupt that the WDT is approaching its time-out condition and behaves differently in Normal mode and in Window mode.

In Normal mode, the early warning interrupt generation is defined by the Early Warning Offset bits in the Early Warning Control register (EWCTRL.EWOFFSET). The Early Warning Offset bits define the number of CLK_WDT_OSC clocks before the interrupt is generated, relative to the start of the WDT time-out period.

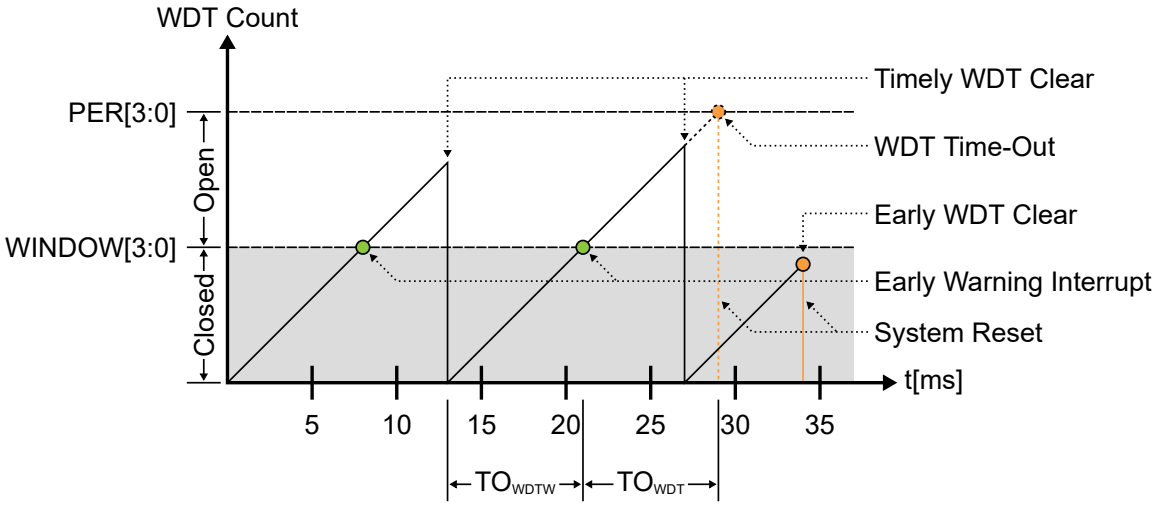
The user must take caution when programming the Early Warning Offset bits. If these bits define an early warning interrupt generation time greater than the WDT time-out period, the WDT time-out system Reset is generated prior to the early warning interrupt. Consequently, the early warning interrupt will never be generated.

Figure 20-4. Normal Mode With Early Warning Interrupt



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, the early warning interrupt can be used to wake up and clear the WDT, after which the system can perform other tasks or return to the sleep mode.

Figure 20-5. Window Mode With Early Warning Interrupt



Example:

If the WDT is operating in Normal mode with CONFIG.PER = `0x2` and EWCTRL.EWOFFSET = `0x1`, the early warning interrupt is generated 16 CLK_WDT_OSC clock cycles after the start of the time-out period. The time-out system Reset is generated 32 CLK_WDT_OSC clock cycles after the start of the WDT time-out period.

20.4.3.2. Always-On Mode

The Always-On mode is enabled by setting the Always-On bit in the Control A register (CTRLA.ALWAYSON = `1`). When the Always-On mode is enabled, the WDT runs continuously, regardless of the state of CTRLA.ENABLE. Once written, the Always-On bit can only be cleared by a Reset. The Configuration (CONFIG) and Early Warning Control (EWCTRL) registers are read-only while the CTRLA.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 (CTRLA.WEN) is allowed while in 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 Always-On mode, but note that EWCTRL.EWOFFSET cannot be changed.

Table 20-2 shows the operation of the WDT when CTRLA.ALWAYSON = `1`.

Table 20-2. 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

20.4.4. Sleep Mode Operation

Normal mode operation:

In sleep modes, an Early Warning interrupt can wake up the device if the corresponding bit in the Interrupt Enable Set (INTENSET) register is written to '1', and the Early Warning Interrupt Time Offset is configured in the Early Warning Control (EWCTRL.EWOFFSET) register.

Window mode operation:

In sleep modes, an Early Warning interrupt can wake up the device if the corresponding bit in the Interrupt Enable Set (INTENSET) register is written to '1'.

20.4.5. Debug Operation

When the CPU is halted in Debug mode, the WDT will halt normal operation.

20.5. Dependencies

20.5.1. I/O Lines

Not applicable.

20.5.2. Power Management

The WDT will continue to operate in any sleep mode. The WDT's interrupts can be used to wake up the device from sleep modes. Refer to the *PM – Power Manager* chapter for details on the different sleep modes.

20.5.3. Clocks

The WDT bus clock (CLK_WDT_APB) can be enabled or disabled in the main clock. See the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter for more information.

A 1.024 kHz oscillator clock (CLK_WDT_OSC) is required to clock the WDT internal counter.

The CLK_WDT_OSC clock is sourced from the internal 32.768 kHz Oscillator (OSC32K).

The CLK_WDT_OSC clock is asynchronous to the bus clock (CLK_WDT_APB). Due to this asynchronicity, writes to certain registers require synchronization between the clock domains. Refer to the *Synchronization* section for further details.

20.5.4. DMA

Not applicable.

20.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use WDT interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 20-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
WDT	EW	In Normal mode, the counter reaches the configured early warning value. In Window mode, the counter enters the open window.	In Normal mode, the Early Warning Offset is configured in the Early Warning Control register (EWCTRL.EWOFFSET)

20.5.6. Events

Not applicable.

20.5.7. Analog Connections

Not applicable.

20.6. Register Summary - WDT

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	ALWAYSON					WEN	ENABLE	
0x01	CONFIG	7:0	WINDOW[3:0]				PER[3:0]			
0x02	EWCTRL	7:0					EWOFFSET[3:0]			
0x03	Reserved									
0x04	INTENCLR	7:0								EW
0x05	INTENSET	7:0								EW
0x06	INTFLAG	7:0								EW
0x07	Reserved									
0x08	SYNCBUSY	7:0				CLEAR	ALWAYSON	WEN	ENABLE	
		15:8								
		23:16								
		31:24								
0x0C	CLEAR	7:0	CLEAR[7:0]							

20.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0xXX
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

The Reset value is determined from the Boot Configuration Fuses (BOOTCFG).

When the Always-On bit in the Control A (CTRLA.ALWAYSON) register is '1', this register will be read-only.

Bit	7	6	5	4	3	2	1	0
	ALWAYSON					WEN	ENABLE	
Access	R/W					R/W	R/W	
Reset	x					x	x	

Bit 7 – ALWAYSON Always-On
This bit controls whether the WDT will run continuously or not.
This bit is only cleared by a Reset.
This bit is set by writing a '1' to it.
Writing a '0' to this bit has no effect.
This bit is loaded from the *BOOTCFG – Boot Configuration Fuses* section at start-up.

Value	Description
0	The WDT can be enabled and disabled through the ENABLE bit
1	The WDT is enabled and can only be disabled by a Power-on Reset (POR)

Note: When this bit is '1', the Control A (CTRLA) register, the Configuration (CONFIG) register, and the Early Warning Control (EWCTRL) register will be read-only, and any writes to these registers will cause an error.

Bit 2 – WEN Watchdog Timer Window Mode Enable
This bit controls whether the Window mode is enabled or not.
This bit is loaded from the *BOOTCFG – Boot Configuration Fuses* section at start-up.

Value	Description
0	Window mode is disabled
1	Window mode is enabled

Bit 1 – ENABLE Enable
This bit controls whether the WDT is enabled or not.
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled or disabled. The value written to CTRL.ENABLE will read back immediately, and the Enable bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.
This bit is not Enable-Protected.
This bit is loaded from the *BOOTCFG – Boot Configuration Fuses* section at start-up.

Value	Description
0	The WDT is disabled
1	The WDT is enabled

20.6.2. Configuration

Name: CONFIG
Offset: 0x01
Reset: 0xXX
Property: PAC Write-Protection

The Reset value is determined from the Boot Configuration Fuses (BOOTCFG).

When the Always-On bit in the Control A register (CTRLA.ALWAYSON) is '1', this register will be read-only.

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, this bit field controls the WDT closed window period as a number of cycles of the 1.024 kHz CLK_WDT_OSC clock.
This bit field is loaded from the *BOOTCFG – Boot Configuration Fuses* section at start-up.

Value	Name	Description
0x0	CYC8	8 clock cycles
0x1	CYC16	16 clock cycles
0x2	CYC32	32 clock cycles
0x3	CYC64	64 clock cycles
0x4	CYC128	128 clock cycles
0x5	CYC256	256 clock cycles
0x6	CYC512	512 clock cycles
0x7	CYC1024	1024 clock cycles
0x8	CYC2048	2048 clock cycles
0x9	CYC4096	4096 clock cycles
0xA	CYC8192	8192 clock cycles
0xB	CYC16384	16384 clock cycles
Other	—	Reserved

Bits 3:0 – PER[3:0] Time-Out Period
This bit field controls the WDT time-out period as a number of 1.024 kHz CLK_WDT_OSC clock cycles.
In Window mode operation, this bit field defines the open window period.
This bit field is loaded from the *BOOTCFG – Boot Configuration Fuses* section at start-up.

Value	Name	Description
0x0	CYC8	8 clock cycles
0x1	CYC16	16 clock cycles
0x2	CYC32	32 clock cycles
0x3	CYC64	64 clock cycles
0x4	CYC128	128 clock cycles
0x5	CYC256	256 clock cycles
0x6	CYC512	512 clock cycles
0x7	CYC1024	1024 clock cycles
0x8	CYC2048	2048 clock cycles
0x9	CYC4096	4096 clock cycles
0xA	CYC8192	8192 clock cycles
0xB	CYC16384	16384 clock cycles
Other	—	Reserved

20.6.3. Early Warning Control

Name: EWCTRL
Offset: 0x02
Reset: 0xXX
Property: PAC Write-Protection

The Reset value is determined from the Boot Configuration Fuses (BOOTCFG).

When the Always-On bit in the Control A register (CTRLA.ALWAYSON) is '1', this register will be read-only.

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

Bits 3:0 – EWOFFSET[3:0] Early Warning Interrupt Time Offset
This bit field controls the number of CLK_WDT_OSC clock cycles between the start of the WDT time-out period and the generation of the early warning interrupt.
This bit field is loaded from the *BOOTCFG – Boot Configuration Fuses* section at start-up.

Value	Name	Description
0x0	CYC8	8 clock cycles
0x1	CYC16	16 clock cycles
0x2	CYC32	32 clock cycles
0x3	CYC64	64 clock cycles
0x4	CYC128	128 clock cycles
0x5	CYC256	256 clock cycles
0x6	CYC512	512 clock cycles
0x7	CYC1024	1024 clock cycles
0x8	CYC2048	2048 clock cycles
0x9	CYC4096	4096 clock cycles
0xA	CYC8192	8192 clock cycles
0xB - 0xF	—	Reserved

20.6.4. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x04
Reset: 0x00
Property: PAC Write-Protection

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.

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

Bit 0 – EW Early Warning Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Early Warning Interrupt Enable bit, which disables the early warning interrupt.

Value	Description
0	The early warning interrupt is disabled
1	The early warning interrupt is enabled

20.6.5. Interrupt Enable Set

Name: INTENSET
Offset: 0x05
Reset: 0x00
Property: PAC Write-Protection

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.

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

Bit 0 – EW Early Warning Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Early Warning Interrupt Enable bit, which enables the early warning interrupt.

Value	Description
0	The early warning interrupt is disabled
1	The early warning interrupt is enabled

20.6.6. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x06
Reset: 0x00
Property: –

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

Bit 0 – EW Early Warning
This flag is cleared by writing a '1' to it.
This flag is set when an early warning interrupt occurs, as defined by the EWOFFSET bit field in the EWCTRL register, and will generate an interrupt request if INTENSET/CLR.EW is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the early warning interrupt flag.

20.6.7. Synchronization Busy

Name: SYNCBUSY
Offset: 0x08
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				CLEAR	ALWAYSON	WEN	ENABLE	
Access				R	R	R	R	
Reset				0	0	0	0	

Bit 4 – CLEAR CLEAR Synchronization Busy

Value	Description
0	Write synchronization of the CLEAR register is complete
1	Write synchronization of the CLEAR register is ongoing

Bit 3 – ALWAYSON ALWAYSON Synchronization Busy

Value	Description
0	Write synchronization of the CTRLA.ALWAYSON bit is complete
1	Write synchronization of the CTRLA.ALWAYSON bit is ongoing

Bit 2 – WEN Window Enable Synchronization Busy

Value	Description
0	Write synchronization of the CTRLA.WEN bit is complete
1	Write synchronization of the CTRLA.WEN bit is ongoing

Bit 1 – ENABLE Enable Synchronization Busy

Value	Description
0	Write synchronization of the CTRLA.ENABLE bit is complete
1	Write synchronization of the CTRLA.ENABLE bit is ongoing

20.6.8. Clear

Name: CLEAR
Offset: 0x0C
Reset: 0x00
Property: 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[7:0] Watchdog Clear
This bit field controls clearing of the WDT.

Value	Name	Description
0xA5	KEY	In Normal mode, writing during the WDT time-out period will clear the WDT and restart the WDT time-out period.(1) In Window mode, writing during the open window will clear the WDT and restart the WDT time-out sequence.(2)
Other	—	Reserved. Will issue an immediate system Reset.

Notes:

1. In Normal mode, the WDT time-out period is given by CONFIG.PER.
2. In Window mode, the WDT time-out sequence period is given by CONFIG.WIN and CONFIG.PER.

21. RTC – Real-Time Counter

21.1. Features

- 32-bit Counter with 10-bit Prescaler
- 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- or 24-hour format)
  - Date in day of month, month, and year
  - Leap year correction
- Digital Prescaler Correction/Tuning for Increased Accuracy
- Overflow, Compare/Alarm Match and Prescaler Interrupts and Events
  - Optional clear on compare/alarm match

21.2. Overview

The Real-Time Counter (RTC) is a 32-bit counter with a 10-bit programmable prescaler and frequency correction functionality. The RTC typically runs continuously to keep track of time in the system. It can operate as a 32-bit or 16-bit counter, as well as a clock/calendar with alarm functionality. The RTC can wake the device from sleep modes and it can generate events and interrupts at overflow, periodically, or when a compare or alarm match occurs.

The 10-bit programmable prescaler can scale down the clock source. This allows a wide range of resolutions and time-out periods can be configured, ranging from microseconds to years, making it suitable for a variety of time-based functions.

With frequency correction, the RTC can compensate for oscillators operating too slowly or fast.

21.3. Block Diagram

Figure 21-1. RTC Block Diagram (Mode 0 – 32-Bit Counter)

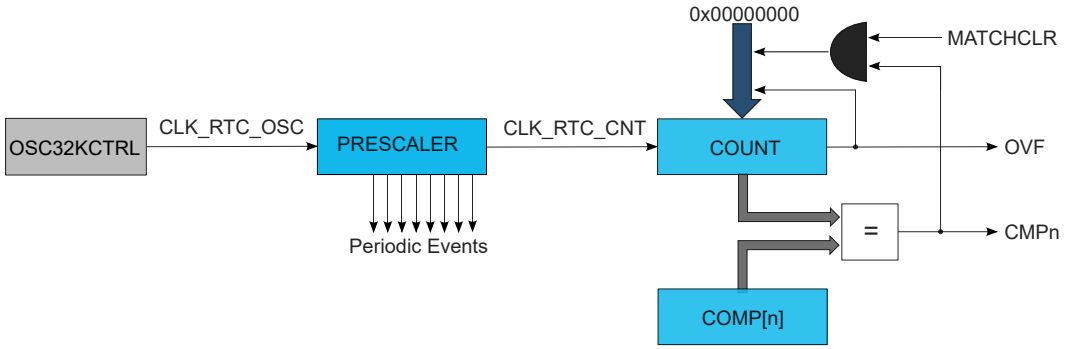


Figure 21-2. RTC Block Diagram (Mode 1 – 16-Bit Counter)

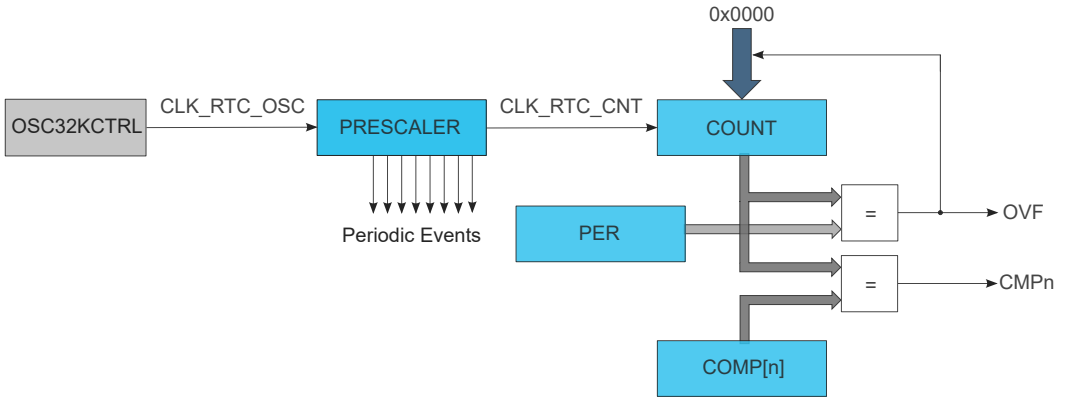
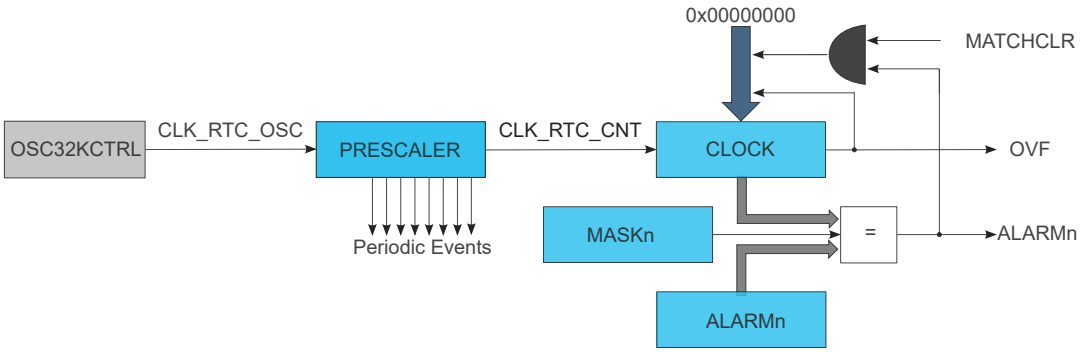


Figure 21-3. RTC Block Diagram (Mode 2 – Clock/Calendar)



21.3.1. Signal Description

Not applicable.

21.4. Functional Description

21.4.1. Initialization

To initialize and run the RTC in basic 32-bit counter mode, follow these steps:

1. Configure the mode of operation to 32-bit counter (CTRLA.MODE).
2. Enable continuous read resynchronization of the counter and wait for the operation to complete (CTRLA.COUNTSYNC and SYNCBUSY.COUNTSYNC).
3. Configure the counter value (COUNT.COUNT).
4. Enable the Overflow interrupt (INTENSET.OVF).
5. Enable the RTC and wait for the operation to complete (CTRLA.ENABLE and SYNCBUSY.ENABLE).
6. Wait for the Overflow interrupt flag to be set when the RTC has overflowed (INTFLAG.OVF).

21.4.2. Operation

21.4.2.1. Prescaler

The RTC prescaler divides the source clock for the RTC counter.

Note: In Clock/Calendar mode, the prescaler must be configured to provide a 1 Hz clock to the counter for correct operation.

The frequency of the RTC clock (CLK_RTC_CNT) is given by the following formula:

$$f_{CLK\_RTC\_CNT} = \frac{f_{CLK\_RTC\_OSC}}{2^{PRESCALER}}$$

The frequency of the oscillator clock, CLK_RTC_OSC, is given by $f_{CLK\_RTC\_OSC}$, and $f_{CLK\_RTC\_CNT}$ is the frequency of the internal prescaled RTC clock, CLK_RTC_CNT.

21.4.2.2. 32-Bit Counter (Mode 0)

When the RTC Operating Mode bit field in the Control A register (CTRLA.MODE) is configured to COUNT32, the counter operates in 32-bit Counter mode. The block diagram of this mode is shown in Figure 21-1. 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 (COUNT) register in 32-bit format. To read valid counter values, the COUNT register requires synchronization prior to reading. This is achieved by writing the COUNT Read Resynchronization Enable bit in the Control A register (CTRLA.COUNTSYNC) to '1' and waiting for the COUNT Read Resynchronization Enable Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.COUNTSYNC) to clear. Once enabled, the counter value is continuously resynchronized. Disabling the resynchronization will prevent valid values from being read from the COUNT register.

The counter value is continuously compared with the 32-bit Compare n Value (COMP[n]) register. When a compare match occurs, the Compare n Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn) is set on the next 0-to-1 transition of CLK_RTC_CNT.

If the Clear on Match bit in the Control A register (CTRLA.MATCHCLR) is '1', the counter is cleared on the next counter cycle when a compare match with COMP[n] occurs. This allows the RTC to generate periodic interrupts or events with longer periods than the prescaler events.
Note: When CTRLA.MATCHCLR is '1', INTFLAG.CMPn and INTFLAG.OVF will both be set simultaneously on a compare match with COMP[n].

21.4.2.3. 16-Bit Counter (Mode 1)

When the RTC Operating Mode bit field in the Control A register (CTRLA.MODE) is configured to COUNT16, the counter operates in 16-bit Counter mode, as shown in Figure 21-2. 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 (PER) register 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 (COUNT) register in 16-bit format. To read valid counter values, the COUNT register requires synchronization prior to reading. This is achieved by writing the COUNT Read Resynchronization Enable bit in the Control A register (CTRLA.COUNTSYNC) to '1' and waiting for the COUNT Read Resynchronization Enable Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.COUNTSYNC) to clear. Once enabled, the counter value is continuously resynchronized. Disabling the resynchronization will prevent reading valid values from the COUNT register.

The counter value is continuously compared with the 16-bit Compare n Value (COMP[n]) registers. When a compare match occurs, the corresponding Compare n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.CMPn) is set on the next 0-to-1 transition of CLK_RTC_CNT.

21.4.2.4. Clock/Calendar (Mode 2)

When the RTC Operating Mode bit field in the Control A register (CTRLA.MODE) is configured to CLOCK, the counter operates in Clock/Calendar mode, as shown in Figure 21-3. 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 1 Hz clock to the counter for correct operation in this mode.

The time and date can be read from or written to the Clock Value (CLOCK) register in a 32-bit time and date format. To read valid clock values, the CLOCK register requires synchronization prior to reading. This is achieved by writing the CLOCK Read Resynchronization Enable bit in the Control A register (CTRLA.CLOCKSYNC) to '1' and waiting for the CLOCK Read Resynchronization Enable Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.CLOCKSYNC) to complete. Once enabled, the clock value is continuously resynchronized. Disabling the resynchronization will prevent valid values from being read from the CLOCK register.

Time is represented as:

- Seconds (0–59)
- Minutes (0–59)
- Hours

Hours can be represented in either 12- or 24-hour format, selected by the Clock Representation bit in the Control A register (CTRLA.CLKREP). This bit can be changed only while the RTC is disabled. In 24-hour clock format, the Hour bit field in the CLOCK register (CLOCK.HOUR) can have values from 0 to 23. In 12-hour clock format, CLOCK.HOUR can have values from 1 to 12, while the Meridiem Indicator bit field (CLOCK.INDICATOR) indicates whether the clock value is before (AM) or after midday (PM).

The date is represented in the following form:

- Day as the numeric day of the month (starting at 1 and ending at 28–31, depending on the month and year)
- Month as the numeric month of the year (1 = January, 2 = February, ..., 12 = December)
- Year as a value from '0x00' to '0x3F'. This value must be added to a user-defined reference year. The reference year must be a leap year (2016, 2020 and so on). For example, the year value '0x2D', when added to the reference year 2016, represents the year 2061.

The RTC will increment until it reaches the top value of 23:59:59 on December 31 of year value '0x3F', and then wrap to 00:00:00 on January 1 of year value '0x00'. This will set the Overflow interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF).

The clock value is continuously compared with the 32-bit Alarm n Value (ALARMn) register. The time and date for the ALARMn register are defined similarly to those for the CLOCK register. When an alarm match occurs, the Alarm n interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.ALARMn) is set on the next 0-to-1 transition of CLK_RTC_CNT. For example, with a 1 Hz clock counter, the Alarm n interrupt flag is set with a delay of 1 second after the occurrence of an alarm match.

A valid alarm match depends on the configuration of the Alarm n Mask Selection bit field in the Alarm n Mask register (MASKn.SEL). This bit field determines which time and date fields of the clock and alarm values are used for comparison and which are ignored.

If the Clear on Match bit in the Control A register (CTRLA.MATCHCLR) is '1', the counter is cleared on the next counter cycle when an alarm match with ALARMn occurs. This allows the RTC to generate periodic interrupts or events with longer periods than would be possible with prescaler events alone. Refer to the *Periodic Intervals* section for details.
Note: When CTRLA.MATCHCLR is '1', INTFLAG.ALARMn and INTFLAG.OVF will both be set simultaneously on an alarm match with ALARMn.

21.4.3. Additional Features

21.4.3.1. Periodic Intervals

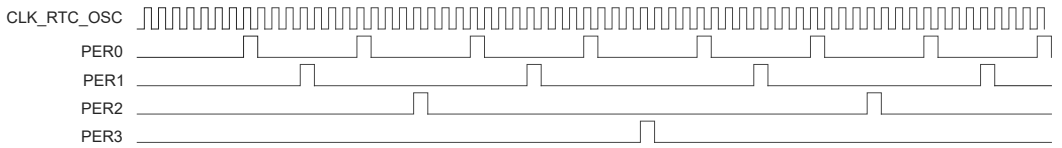
The RTC prescaler can generate interrupts and events at periodic intervals, allowing flexible system tick creation. Any of the upper eight bits of the 10-bit prescaler (bits 2 to 9) can be the source of an interrupt or event. When one of the eight Periodic Interval n Event Output Enable bits in the Event

Control register (EVCTRL.PEREOn) is '1', an event is generated on the 0-to-1 transition of the related bit in the prescaler (bit n+2), resulting in a periodic event frequency of

$$f_{\mathrm{PERIODIC(n)}} = \frac{f_{\mathrm{CLK\_RTC\_OSC}}}{2^{n+3}}$$

where $f_{CLK\_RTC\_OSC}$ is the frequency of the internal prescaler clock (CLK_RTC_OSC), and n is the position of the EVCTRL.PEREOn bit. For example, PER0 will generate an event every eight CLK_RTC_OSC cycles, PER1 every 16 cycles, and so on. This is shown in the following figure.

Figure 21-4. Example Periodic Events



Periodic events are independent of the prescaler setting used by the RTC counter, except when the Prescaler bit field in the Control A register (CTRLA.PRESCALER) is configured to OFF. In that case, no periodic events will be generated.

21.4.3.2. Frequency Correction

The RTC Frequency Correction functionality employs periodic counter corrections to compensate for an oscillator that is too slow or too fast. Frequency correction requires that the Prescaler bit field in the Control A register (CTRLA.PRESCALER) be configured to DIVn, where n > 1.

The digital correction circuit adds or subtracts cycles from the RTC prescaler to adjust the frequency in approximately 1 ppm (parts per million) steps. Digital correction is achieved by adding or skipping a single count in the prescaler once every 8,192 CLK_RTC_OSC cycles. The Correction Value bit field in the Frequency Correction register (FREQCORR.VALUE) determines the number of times the adjustment is applied over 128 of these periods. The resulting correction is as follows:

$$\text{Correction in ppm} = \frac{\text{FREQCORR.VALUE}}{8192 \cdot 128} \cdot 10^6 \text{ppm}$$

This results in a resolution of 0.95367 ppm.

The Correction Sign bit in the Frequency Correction register (FREQCORR.SIGN) determines the direction of the correction. A positive value will add counts and increase the period (reducing the frequency), while a negative value will reduce the counts per period (increasing the frequency).

Digital correction also affects the generation of 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.

21.4.4. Sleep Mode Operation

The RTC will continue to operate in any sleep mode where the source clock is active. RTC interrupts can be used to wake the device from a sleep mode. RTC events can trigger other operations in the system without exiting sleep mode.

An interrupt request will be generated after wake-up if the Nested Vectored Interrupt Controller (NVIC) is configured accordingly. Refer to the *NVIC – Nested Vectored Interrupt Controller* section for details. Otherwise, the CPU will wake up directly, without triggering any interrupt. In this case, the CPU will continue executing from the first instruction that followed the entry into sleep.

Periodic events can also wake up the CPU through the interrupt function of the Event System (EVSYS). In this case, the event must be enabled and connected to an event channel with its interrupt enabled. Refer to the *EVSYS – Event System* chapter for more information.

When the device is in Standby sleep mode, the Direct Memory Access (DMA) engine is not able to write to the Counter Value (COUNT) register. To write to the COUNT register with the DMA the device must be in Active mode or Idle sleep mode. Refer to the *DMAC – Direct Memory Access Controller* and *PM - Power Manager* chapters for details.

21.4.5. Debug Operation

When the CPU is halted in debug mode the RTC will halt normal operation. Setting the Debug Run bit in the Debug Control register (DBGCTRL.DBGRUN) forces the RTC to continue operating when the CPU is halted in debug mode.

If the RTC 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. Dependencies

21.5.1. I/O Lines

Not applicable.

21.5.2. Power Management

The RTC will continue to operate in any sleep mode where the selected source clock is running. RTC interrupts can be used to wake the device from sleep modes. Events connected to the event system can trigger other operations in the system without exiting sleep modes. Refer to the *PM - Power Manager* chapter for details on the different sleep modes.

21.5.3. 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 of the *MCLK – Main Clock* chapter.

A 32.768 kHz or 1.024 kHz oscillator clock (CLK_RTC_OSC) is required to clock the RTC. This clock must be configured and enabled in the *32.768 kHz Oscillators Controller* section before using the RTC.

CLK_RTC_OSC is asynchronous to CLK_RTC_APB. Due to this asynchronicity, writes to certain registers require synchronization between the clock domains.

21.5.4. DMA

Not applicable.

21.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use RTC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is located in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together at the system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine the interrupt condition.

Table 21-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
RTC	OVF	The counter has reached its top value and wrapped to zero	The Counter Value (COUNT) register
	CMPn/ ALARMn	**32-bit and 16-bit Counter (Mode 0 and 1):** The compare n value matches the counter value. **Clock/Calendar (Mode 2):** The alarm n value matches the clock value.	**32-bit and 16-bit Counter (Mode 0 and 1):** The Compare n Value (COMP[n]) and COUNT registers. **Clock/Calendar (Mode 2):** The Alarm n Value (ALARMn) and Clock Value (CLOCK) registers.
	PERn	The corresponding prescaler bit has toggled	The Counter Period (PER) register

21.5.6. Events

The RTC can generate the following events:

Table 21-2. Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
RTC	OVF	Overflow	Pulse	CLK_RTC_APB	One CLK_RTC_APB period
	CMP_n	**32-bit and 16-bit Counter (Mode 0 and 1):** Compare n. **Clock/Calendar (Mode 2):** Alarm n.	Pulse	CLK_RTC_APB	One CLK_RTC_APB period
	PER_n	Period Interval n	Pulse	CLK_RTC_APB	One CLK_RTC_APB period

Writing a '1' to an Event Output Enable bit in the Event Control register (EVCTRL.xxEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The RTC has no event users or actions.

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

21.5.7. Analog Connections

Not applicable.

21.6. Register Summary - RTC.MODE0

RTC in 32-bit counter mode (MODE0)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	MATCHCLR				MODE[1:0]		ENABLE	SWRST
		15:8	COUNTSYNC				PRESCALER[3:0]			
0x02 ... 0x03	Reserved									
0x04	EVCTRL	7:0	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
		15:8	OVFEO							CMPEO0
		23:16								
		31:24								
0x08	INTENCLR	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF							CMP0
0x0A	INTENSET	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF							CMP0
0x0C	INTFLAG	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF							CMP0
0x0E	DBGCTRL	7:0								DBGRUN
0x0F	Reserved									
0x10	SYNCBUSY	7:0			COMP0		COUNT	FREQCORR	ENABLE	SWRST
		15:8	COUNTSYNC							
		23:16								
		31:24								
0x14	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x15 ... 0x17	Reserved									
0x18	COUNT	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16	COUNT[23:16]							
		31:24	COUNT[31:24]							
0x1C ... 0x1F	Reserved									
0x20	COMP[0]	7:0	COMP[7:0]							
		15:8	COMP[15:8]							
		23:16	COMP[23:16]							
		31:24	COMP[31:24]							

21.6.1. Control A - MODE0

Name: CTRLA
Offset: 0x00
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	COUNTSYNC				PRESCALER[3:0]			
Access	R/W				R/W	R/W	R/W	R/W
Reset	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/W	R/W	R/W
Reset	0				0	0	0	0

Bit 15 – COUNTSYNC COUNT Read Resynchronization Enable
This bit controls the continuous read resynchronization of the COUNT register.

Value	Description
0	Continuous read resynchronization of COUNT is disabled. The values in COUNT are not continuously updated and are invalid.
1	Continuous read resynchronization of COUNT is enabled. The values in COUNT are continuously updated and valid.

Note: This bit is not enable-protected.

Bits 11:8 – PRESCALER[3:0] Prescaler
This bit field controls the prescaling factor for the RTC clock source (CLK_RTC_OSC) to generate the counter clock (CLK_RTC_CNT). Periodic events and interrupts are not available when the prescaler is off.

Value	Name	Description
0x0	OFF	CLK_RTC_CNT = GCLK_RTC/1
0x1	DIV1	CLK_RTC_CNT = GCLK_RTC/1
0x2	DIV2	CLK_RTC_CNT = GCLK_RTC/2
0x3	DIV4	CLK_RTC_CNT = GCLK_RTC/4
0x4	DIV8	CLK_RTC_CNT = GCLK_RTC/8
0x5	DIV16	CLK_RTC_CNT = GCLK_RTC/16
0x6	DIV32	CLK_RTC_CNT = GCLK_RTC/32
0x7	DIV64	CLK_RTC_CNT = GCLK_RTC/64
0x8	DIV128	CLK_RTC_CNT = GCLK_RTC/128
0x9	DIV256	CLK_RTC_CNT = GCLK_RTC/256
0xA	DIV512	CLK_RTC_CNT = GCLK_RTC/512
0xB	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
Other	—	Reserved

Note: This bit field is not synchronized.

Bit 7 – MATCHCLR Clear on Match
This bit controls whether the counter is cleared on a match.

Value	Description
0	The counter is not cleared on a Compare/Alarm match

Value	Description
1	The counter is cleared on a Compare/Alarm match

Note: This bit is not synchronized.

Bits 3:2 – MODE[1:0] Operating Mode
This bit field controls the operating mode of the RTC.

Value	Name	Description
0x0	COUNT32	Mode 0: 32-bit counter
0x1	COUNT16	Mode 1: 16-bit counter
0x2	CLOCK	Mode 2: Clock/calendar
Other	—	Reserved

Note: This bit field is not synchronized.

Bit 1 – ENABLE Enable
This bit controls whether the RTC is enabled.
Due to synchronization, there is a delay from writing to CTRLA.ENABLE until the RTC is enabled or disabled. The value written to CTRLA.ENABLE will be read back immediately and the Enable bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

Value	Description
0	The RTC is or being disabled
1	The RTC is or being enabled

Note: This bit is not enable-protected.

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.
Due to synchronization, there is a delay from writing to CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

Value	Description
0	There is no reset operation ongoing
1	The reset operation is ongoing

Notes:

1. This bit is not enable protected.
2. Writing a '1' to CTRL.SWRST will always take precedence, meaning that all other writes in the same write operation will be discarded.
3. The RTC should be disabled before the RTC is reset to avoid undefined behavior.

21.6.2. Event Control - MODE0

Name: EVCTRL
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVFEO							CMPEO0
Access	R/W							R/W
Reset	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

Value	Description
0	Overflow event is disabled and will not be generated
1	Overflow event is enabled and will be generated for every overflow

Bit 8 – CMPEOn Compare n Event Output Enable

Value	Description
0	Compare n event is disabled and will not be generated
1	Compare n event is enabled and will be generated for every compare match

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PEREOn Periodic Interval n Event Output Enable

Value	Description
0	Periodic Interval n event is disabled and will not be generated
1	Periodic Interval n event is enabled and will be generated

21.6.3. Interrupt Enable Clear - MODE0

Name: INTENCLR
Offset: 0x08
Reset: 0x0000
Property: PAC Write-Protection

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

Bit	15	14	13	12	11	10	9	8
	OVF							CMP0
Access	R/W							R/W
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, thereby disabling the Overflow interrupt.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

Bit 8 – CMPn Compare n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Compare n Interrupt Enable bit, thereby disabling the Compare n interrupt.

Value	Description
0	The Compare n interrupt is disabled
1	The Compare n interrupt is enabled

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Periodic Interval n Interrupt Enable bit, thereby disabling the Periodic Interval n interrupt.

Value	Description
0	The Periodic Interval n interrupt is disabled
1	The Periodic Interval n interrupt is enabled

21.6.4. Interrupt Enable Set - MODE0

Name: INTENSET
Offset: 0x0A
Reset: 0x0000
Property: PAC Write-Protection

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

Bit	15	14	13	12	11	10	9	8
	OVF							CMP0
Access	R/W							R/W
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Enable bit, thereby enabling the Overflow interrupt.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

Bit 8 – CMPn Compare n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Compare n Interrupt Enable bit, thereby enabling the Compare n interrupt.

Value	Description
0	The Compare n interrupt is disabled
1	The Compare n interrupt is enabled

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Periodic Interval n Interrupt Enable bit, thereby enabling the Periodic Interval n interrupt.

Value	Description
0	The Periodic Interval n interrupt is disabled
1	The Periodic Interval n interrupt is enabled

21.6.5. Interrupt Flag Status and Clear - MODE0

Name: INTFLAG
Offset: 0x0C
Reset: 0x0000
Property: –

Bit	15	14	13	12	11	10	9	8
	OVF							CMP0
Access	R/W							R/W
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs and will generate an interrupt if INTENCLR/SET.OVF is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow interrupt flag.

Bit 8 – CMPn Compare n
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition and will generate an interrupt if INTENCLR/SET.CMPn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Compare n interrupt flag.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n
This flag is cleared by writing a '1' to it.
This flag is set on the 0-to-1 transition of prescaler bit (n+2) and will generate an interrupt if INTENCLR/SET.PERn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Periodic Interval n interrupt flag.

21.6.6. Debug Control

Name: DBGCTRL
Offset: 0x0E
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Debug Run
This bit is not reset by a software reset.
This bit controls the functionality when the CPU is halted by an external debugger.

Value	Description
0	The RTC is halted when the CPU is halted by an external debugger
1	The RTC continues normal operation when the CPU is halted by an external debugger

21.6.7. Synchronization Busy - MODE0

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

Bit	7	6	5	4	3	2	1	0
			COMP0		COUNT	FREQCORR	ENABLE	SWRST
Access			R		R	R	R	R
Reset			0		0	0	0	0

Bit 15 – COUNTSYNC COUNT Read Resynchronization Enable Synchronization Busy
This bit is cleared when the synchronization of CTRLA.COUNTSYNC is complete.
This bit is set when the synchronization of CTRLA.COUNTSYNC is started.

Note: This bit will only be set when writing to CTRLA.COUNTSYNC, not as a result of ongoing read resynchronization of the COUNT register.

Bit 5 – COMPn Compare n Value Synchronization Busy
This bit is cleared when the synchronization of the COMP[n] register is complete.
This bit is set when the synchronization of the COMP[n] register is started.

Bit 3 – COUNT Counter Value Synchronization Busy
This bit is cleared when the synchronization of the COUNT register is complete.
This bit is set when the synchronization of the COUNT register is started.

Bit 2 – FREQCORR Frequency Correction Synchronization Busy
This bit is cleared when the synchronization of the FREQCORR register is complete.
This bit is set when the synchronization of the FREQCORR register is started.

Bit 1 – ENABLE Enable Synchronization Busy
This bit is cleared when the synchronization of CTRLA.ENABLE is complete.
This bit is set when the synchronization of CTRLA.ENABLE is started.

Bit 0 – SWRST Software Reset Synchronization Busy
This bit is cleared when the synchronization of CTRLA.SWRST is complete.
This bit is set when the synchronization of CTRLA.SWRST is started.

21.6.8. Frequency Correction

Name: FREQCORR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection, 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

Value	Description
0	The correction value is positive, that is, frequency will be decreased
1	The correction value is negative, that is, frequency will be increased

Bits 6:0 – VALUE[6:0] Correction Value
This bit field controls the amount of correction applied to the RTC prescaler.
Refer to the *Frequency Correction* section for details on how the correction value is calculated.

Value	Description
0x0	Correction is disabled, and the RTC frequency is unchanged
Other	The RTC frequency is adjusted according to the value

21.6.9. Counter Value - MODE0

Name: COUNT
Offset: 0x18
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. This register must be written with 32-bit accesses only.
2. Prior to any read access, this register requires continuous read resynchronization to be enabled by writing the COUNT Read Resynchronization Enable bit in the Control A register (CTRLA.COUNTSYNC) to '1' in order to read valid counter values.

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
This bit field controls the value of the RTC counter.

21.6.10. Compare n Value - MODE0

Name: COMP[n]
Offset: 0x20
Reset: 0x00000000
Property: PAC Write-Protection, 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
This bit field controls the value to which the RTC counter is continuously compared.

21.7. Register Summary - RTC.MODE1

RTC in 16-bit counter mode (MODE1)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0					MODE[1:0]		ENABLE	SWRST
		15:8	COUNTSYNC				PRESCALER[3:0]			
0x02 ... 0x03	Reserved									
0x04	EVCTRL	7:0	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
		15:8	OVFEO						CMPEO1	CMPEO0
		23:16								
		31:24								
0x08	INTENCLR	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF						CMP1	CMP0
0x0A	INTENSET	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF						CMP1	CMP0
0x0C	INTFLAG	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF						CMP1	CMP0
0x0E	DBGCTRL	7:0								DBGRUN
0x0F	Reserved									
0x10	SYNCBUSY	7:0		COMP1	COMP0	PER	COUNT	FREQCORR	ENABLE	SWRST
		15:8	COUNTSYNC							
		23:16								
		31:24								
0x14	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x15 ... 0x17	Reserved									
0x18	COUNT	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
0x1A ... 0x1B	Reserved									
0x1C	PER	7:0	PER[7:0]							
		15:8	PER[15:8]							
0x1E ... 0x1F	Reserved									
0x20	COMP[0]	7:0	COMP[7:0]							
		15:8	COMP[15:8]							
0x22	COMP[1]	7:0	COMP[7:0]							
		15:8	COMP[15:8]							

21.7.1. Control A - MODE1

Name: CTRLA
Offset: 0x00
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	COUNTSYNC				PRESCALER[3:0]			
Access	R/W				R/W	R/W	R/W	R/W
Reset	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/W	R/W
Reset					0	0	0	0

Bit 15 – COUNTSYNC COUNT Read Resynchronization Enable
This bit controls the continuous read resynchronization of the COUNT register.

Value	Description
0	Continuous read resynchronization of COUNT is disabled. The values in COUNT are not continuously updated and are invalid.
1	Continuous read resynchronization of COUNT is enabled. The values in COUNT are continuously updated and valid.

Note: This bit is not enable-protected.

Bits 11:8 – PRESCALER[3:0] Prescaler
This bit field controls the prescaling factor for the RTC clock source (CLK_RTC_OSC) to generate the counter clock (CLK_RTC_CNT). Periodic events and interrupts are not available when the prescaler is off.

Value	Name	Description
0x0	OFF	CLK_RTC_CNT = GCLK_RTC/1
0x1	DIV1	CLK_RTC_CNT = GCLK_RTC/1
0x2	DIV2	CLK_RTC_CNT = GCLK_RTC/2
0x3	DIV4	CLK_RTC_CNT = GCLK_RTC/4
0x4	DIV8	CLK_RTC_CNT = GCLK_RTC/8
0x5	DIV16	CLK_RTC_CNT = GCLK_RTC/16
0x6	DIV32	CLK_RTC_CNT = GCLK_RTC/32
0x7	DIV64	CLK_RTC_CNT = GCLK_RTC/64
0x8	DIV128	CLK_RTC_CNT = GCLK_RTC/128
0x9	DIV256	CLK_RTC_CNT = GCLK_RTC/256
0xA	DIV512	CLK_RTC_CNT = GCLK_RTC/512
0xB	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
Other	—	Reserved

Note: This bit field is not synchronized.

Bits 3:2 – MODE[1:0] Operating Mode
This bit field controls the operating mode of the RTC.

Value	Name	Description
0x0	COUNT32	Mode 0: 32-bit counter

Value	Name	Description
0x1	COUNT16	Mode 1: 16-bit counter
0x2	CLOCK	Mode 2: Clock/calendar
Other	—	Reserved

Note: This bit field is not synchronized.

Bit 1 – ENABLE Enable

This bit controls whether the RTC is enabled.
Due to synchronization, there is a delay from writing to CTRLA.ENABLE until the RTC is enabled or disabled. The value written to CTRLA.ENABLE will be read back immediately and the Enable bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

Value	Description
0	The RTC is or being disabled
1	The RTC is or being enabled

Note: This bit is not enable-protected.

Bit 0 – SWRST Software Reset

Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.
Due to synchronization, there is a delay from writing to CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

Value	Description
0	There is no reset operation ongoing
1	The reset operation is ongoing

Notes:

1. This bit is not enable protected.
2. Writing a '1' to CTRL.SWRST will always take precedence, meaning that all other writes in the same write operation will be discarded.
3. The RTC should be disabled before the RTC is reset to avoid undefined behavior.

21.7.2. Event Control - MODE1

Name: EVCTRL
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVFEO						CMPEO1	CMPEO0
Access	R/W						R/W	R/W
Reset	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

Value	Description
0	Overflow event is disabled and will not be generated
1	Overflow event is enabled and will be generated for every overflow

Bits 8, 9 – CMPEOn Compare n Event Output Enable

Value	Description
0	Compare n event is disabled and will not be generated
1	Compare n event is enabled and will be generated for every compare match

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PEREOn Periodic Interval n Event Output Enable

Value	Description
0	Periodic Interval n event is disabled and will not be generated
1	Periodic Interval n event is enabled and will be generated

21.7.3. Interrupt Enable Clear - MODE1

Name: INTENCLR
Offset: 0x08
Reset: 0x0000
Property: PAC Write-Protection

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

Bit	15	14	13	12	11	10	9	8
	OVF						CMP1	CMP0
Access	R/W						R/W	R/W
Reset	0						0	0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, thereby disabling the Overflow interrupt.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

Bits 8, 9 – CMPn Compare n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Compare n Interrupt Enable bit, thereby disabling the Compare n interrupt.

Value	Description
0	The Compare n interrupt is disabled
1	The Compare n interrupt is enabled

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Periodic Interval n Interrupt Enable bit, thereby disabling the Periodic Interval n interrupt.

Value	Description
0	The Periodic Interval n interrupt is disabled
1	The Periodic Interval n interrupt is enabled

21.7.4. Interrupt Enable Set - MODE1

Name: INTENSET
Offset: 0x0A
Reset: 0x0000
Property: PAC Write-Protection

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

Bit	15	14	13	12	11	10	9	8
	OVF						CMP1	CMP0
Access	R/W						R/W	R/W
Reset	0						0	0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Enable bit, thereby enabling the Overflow interrupt.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

Bits 8, 9 – CMPn Compare n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Compare n Interrupt Enable bit, thereby enabling the Compare n interrupt.

Value	Description
0	The Compare n interrupt is disabled
1	The Compare n interrupt is enabled

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Periodic Interval n Interrupt Enable bit, thereby enabling the Periodic Interval n interrupt.

Value	Description
0	The Periodic Interval n interrupt is disabled
1	The Periodic Interval n interrupt is enabled

21.7.5. Interrupt Flag Status and Clear - MODE1

Name: INTFLAG
Offset: 0x0C
Reset: 0x0000
Property: –

Bit	15	14	13	12	11	10	9	8
	OVF						CMP1	CMP0
Access	R/W						R/W	R/W
Reset	0						0	0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs and will generate an interrupt if INTENCLR/SET.OVF is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow interrupt flag.

Bits 8, 9 – CMPn Compare n
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition and will generate an interrupt if INTENCLR/SET.CMPn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Compare n interrupt flag.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n
This flag is cleared by writing a '1' to it.
This flag is set on the 0-to-1 transition of prescaler bit (n+2) and will generate an interrupt if INTENCLR/SET.PERn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Periodic Interval n interrupt flag.

21.7.6. Debug Control

Name: DBGCTRL
Offset: 0x0E
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Debug Run
This bit is not reset by a software reset.
This bit controls the functionality when the CPU is halted by an external debugger.

Value	Description
0	The RTC is halted when the CPU is halted by an external debugger
1	The RTC continues normal operation when the CPU is halted by an external debugger

21.7.7. Synchronization Busy - MODE1

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

Bit	7	6	5	4	3	2	1	0
		COMP1	COMP0	PER	COUNT	FREQCORR	ENABLE	SWRST
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bit 15 – COUNTSYNC COUNT Read Resynchronization Enable Synchronization Busy
This bit is cleared when the synchronization of CTRLA.COUNTSYNC is complete.
This bit is set when the synchronization of CTRLA.COUNTSYNC is started.

Note: This bit will only be set when writing to CTRLA.COUNTSYNC, not as a result of ongoing read resynchronization of the COUNT register.

Bits 5, 6 – COMPn Compare n Value Synchronization Busy
This bit is cleared when the synchronization of the COMP[n] register is complete.
This bit is set when the synchronization of the COMP[n] register is started.

Bit 4 – PER Counter Period Synchronization Busy
This bit is cleared when the synchronization of the PER register is complete.
This bit is set when the synchronization of the PER register is started.

Bit 3 – COUNT Counter Value Synchronization Busy
This bit is cleared when the synchronization of the COUNT register is complete.
This bit is set when the synchronization of the COUNT register is started.

Bit 2 – FREQCORR Frequency Correction Synchronization Busy
This bit is cleared when the synchronization of the FREQCORR register is complete.
This bit is set when the synchronization of the FREQCORR register is started.

Bit 1 – ENABLE Enable Synchronization Busy
This bit is cleared when the synchronization of CTRLA.ENABLE is complete.
This bit is set when the synchronization of CTRLA.ENABLE is started.

Bit 0 – SWRST Software Reset Synchronization Busy
This bit is cleared when the synchronization of CTRLA.SWRST is complete.
This bit is set when the synchronization of CTRLA.SWRST is started.

21.7.8. Frequency Correction

Name: FREQCORR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection, 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

Value	Description
0	The correction value is positive, that is, frequency will be decreased
1	The correction value is negative, that is, frequency will be increased

Bits 6:0 – VALUE[6:0] Correction Value
This bit field controls the amount of correction applied to the RTC prescaler.
Refer to the *Frequency Correction* section for details on how the correction value is calculated.

Value	Description
0x0	Correction is disabled, and the RTC frequency is unchanged
Other	The RTC frequency is adjusted according to the value

21.7.9. Counter Value - MODE1

Name: COUNT
Offset: 0x18
Reset: 0x0000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. This register must be written with 16-bit accesses only.
2. Prior to any read access, this register requires continuous read resynchronization to be enabled by writing the COUNT Read Resynchronization Enable bit in the Control A register (CTRLA.COUNTSYNC) to '1' in order to read valid counter values.

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
This bit field controls the value of the RTC counter.

21.7.10. Counter Period - MODE1

Name: PER
Offset: 0x1C
Reset: 0x0000
Property: PAC Write-Protection, 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
This bit field controls the value of the RTC period.

21.7.11. Compare n Value - MODE1

Name: COMP[n]
Offset: 0x20 + n*0x02 [n=0..1]
Reset: 0x0000
Property: PAC Write-Protection, 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
This bit field controls the value to which the RTC counter is continuously compared.

21.8. Register Summary - RTC.MODE2

RTC in Clock and Calendar mode (MODE2)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	MATCHCLR	CLKREP			MODE[1:0]		ENABLE	SWRST
		15:8	CLOCKSYNC				PRESCALER[3:0]			
0x02 ... 0x03	Reserved									
0x04	EVCTRL	7:0	PEREO7	PEREO6	PEREO5	PEREO4	PEREO3	PEREO2	PEREO1	PEREO0
		15:8	OVFEO							ALARMEO0
		23:16								
		31:24								
0x08	INTENCLR	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF							ALARM0
0x0A	INTENSET	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF							ALARM0
0x0C	INTFLAG	7:0	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
		15:8	OVF							ALARM0
0x0E	DBGCTRL	7:0								DBGRUN
0x0F	Reserved									
0x10	SYNCBUSY	7:0			ALARM0		CLOCK	FREQCORR	ENABLE	SWRST
		15:8	CLOCKSYNC				MASK0			
		23:16								
		31:24								
0x14	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x15 ... 0x17	Reserved									
0x18	CLOCK	7:0	MINUTE[1:0]		SECOND[5:0]					
		15:8	HOUR[3:0]				MINUTE[5:2]			
		23:16	MONTH[1:0]		DAY[4:0]					HOUR[4]
		31:24	YEAR[5:0]						MONTH[3:2]	
0x18	CLOCK	7:0	MINUTE[1:0]		SECOND[5:0]					
		15:8	HOUR[3:0]				MINUTE[5:2]			
		23:16	MONTH[1:0]		DAY[4:0]					INDICATOR
		31:24	YEAR[5:0]						MONTH[3:2]	
0x1C ... 0x1F	Reserved									
0x20	ALARM0	7:0	MINUTE[1:0]		SECOND[5:0]					
		15:8	HOUR[3:0]				MINUTE[5:2]			
		23:16	MONTH[1:0]		DAY[4:0]					HOUR[4]
		31:24	YEAR[5:0]						MONTH[3:2]	
0x20	ALARM0	7:0	MINUTE[1:0]		SECOND[5:0]					
		15:8	HOUR[3:0]				MINUTE[5:2]			
		23:16	MONTH[1:0]		DAY[4:0]					INDICATOR
		31:24	YEAR[5:0]						MONTH[3:2]	
0x24	MASK0	7:0						SEL[2:0]		

21.8.1. Control A - MODE2

Name: CTRLA
Offset: 0x00
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	CLOCKSYNC				PRESCALER[3:0]			
Access	R/W				R/W	R/W	R/W	R/W
Reset	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/W	R/W	R/W	R/W
Reset	0	0			0	0	0	0

Bit 15 – CLOCKSYNC CLOCK Read Resynchronization Enable
This bit controls the continuous read resynchronization of the CLOCK register.

Value	Description
0	Continuous read resynchronization of CLOCK is disabled. The values in CLOCK are not continuously updated and are invalid.
1	Continuous read resynchronization of CLOCK is enabled. The values in CLOCK are continuously updated and valid.

Note: This bit is not enable-protected.

Bits 11:8 – PRESCALER[3:0] Prescaler
This bit field controls the prescaling factor for the RTC clock source (CLK_RTC_OSC) to generate the counter clock (CLK_RTC_CNT). Periodic events and interrupts are not available when the prescaler is off.

Value	Name	Description
0x0	OFF	CLK_RTC_CNT = GCLK_RTC/1
0x1	DIV1	CLK_RTC_CNT = GCLK_RTC/1
0x2	DIV2	CLK_RTC_CNT = GCLK_RTC/2
0x3	DIV4	CLK_RTC_CNT = GCLK_RTC/4
0x4	DIV8	CLK_RTC_CNT = GCLK_RTC/8
0x5	DIV16	CLK_RTC_CNT = GCLK_RTC/16
0x6	DIV32	CLK_RTC_CNT = GCLK_RTC/32
0x7	DIV64	CLK_RTC_CNT = GCLK_RTC/64
0x8	DIV128	CLK_RTC_CNT = GCLK_RTC/128
0x9	DIV256	CLK_RTC_CNT = GCLK_RTC/256
0xA	DIV512	CLK_RTC_CNT = GCLK_RTC/512
0xB	DIV1024	CLK_RTC_CNT = GCLK_RTC/1024
Other	—	Reserved

Note: This bit field is not synchronized.

Bit 7 – MATCHCLR Clear on Match
This bit controls whether the counter is cleared on a match.

Value	Description
0	The counter is not cleared on a Compare/Alarm match

Value	Description
1	The counter is cleared on a Compare/Alarm match

Note: This bit is not synchronized.

Bit 6 – CLKREP Clock Representation
This bit determines how the hours are represented in the Clock Value (CLOCK) register.

Value	Name	Description
0	CLK24H	24 hour clock format
1	CLK12H	12 hour (AM/PM) clock format

Note: This bit is not synchronized.

Bits 3:2 – MODE[1:0] Operating Mode
This bit field controls the operating mode of the RTC.

Value	Name	Description
0x0	COUNT32	Mode 0: 32-bit counter
0x1	COUNT16	Mode 1: 16-bit counter
0x2	CLOCK	Mode 2: Clock/calendar
Other	—	Reserved

Note: This bit field is not synchronized.

Bit 1 – ENABLE Enable
This bit controls whether the RTC is enabled.
Due to synchronization, there is a delay from writing to CTRLA.ENABLE until the RTC is enabled or disabled. The value written to CTRLA.ENABLE will be read back immediately and the Enable bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE will be cleared when the operation is complete.

Value	Description
0	The RTC is or being disabled
1	The RTC is or being enabled

Note: This bit is not enable-protected.

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the RTC, except DBGCTRL, to their initial state, and the RTC will be disabled.
Due to synchronization, there is a delay from writing to CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

Value	Description
0	There is no reset operation ongoing
1	The reset operation is ongoing

Notes:

1. This bit is not enable protected.
2. Writing a '1' to CTRL.SWRST will always take precedence, meaning that all other writes in the same write operation will be discarded.
3. The RTC should be disabled before the RTC is reset to avoid undefined behavior.

21.8.2. Event Control - MODE2

Name: EVCTRL
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVFEO							ALARMEO0
Access	R/W							R/W
Reset	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

Value	Description
0	Overflow event is disabled and will not be generated
1	Overflow event is enabled and will be generated for every overflow

Bit 8 – ALARMEOn Alarm n Event Output Enable

Value	Description
0	Alarm n event is disabled and will not be generated
1	Alarm n event is enabled and will be generated for every compare match

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PEREOn Periodic Interval n Event Output Enable

Value	Description
0	Periodic Interval n event is disabled and will not be generated
1	Periodic Interval n event is enabled and will be generated

21.8.3. Interrupt Enable Clear - MODE2

Name: INTENCLR
Offset: 0x08
Reset: 0x0000
Property: PAC Write-Protection

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

Bit	15	14	13	12	11	10	9	8
	OVF							ALARM0
Access	R/W							R/W
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, thereby disabling the Overflow interrupt.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

Bit 8 – ALARMn Alarm n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Alarm n Interrupt Enable bit, thereby disabling the Alarm n interrupt.

Value	Description
0	The Alarm n interrupt is disabled
1	The Alarm n interrupt is enabled

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Periodic Interval n Interrupt Enable bit, thereby disabling the Periodic Interval n interrupt.

Value	Description
0	The Periodic Interval n interrupt is disabled
1	The Periodic Interval n interrupt is enabled

21.8.4. Interrupt Enable Set - MODE2

Name: INTENSET
Offset: 0x0A
Reset: 0x0000
Property: PAC Write-Protection

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

Bit	15	14	13	12	11	10	9	8
	OVF							ALARM0
Access	R/W							R/W
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Enable bit, thereby enabling the Overflow interrupt.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

Bit 8 – ALARMn Alarm n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Alarm n Interrupt Enable bit, thereby enabling the Alarm n interrupt.

Value	Description
0	The Alarm n interrupt is disabled
1	The Alarm n interrupt is enabled

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Periodic Interval n Interrupt Enable bit, thereby enabling the Periodic Interval n interrupt.

Value	Description
0	The Periodic Interval n interrupt is disabled
1	The Periodic Interval n interrupt is enabled

21.8.5. Interrupt Flag Status and Clear - MODE2

Name: INTFLAG
Offset: 0x0C
Reset: 0x0000
Property: –

Bit	15	14	13	12	11	10	9	8
	OVF							ALARM0
Access	R/W							R/W
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	PER7	PER6	PER5	PER4	PER3	PER2	PER1	PER0
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 – OVF Overflow
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_RTC_CNT cycle after an overflow condition occurs and will generate an interrupt if INTENCLR/SET.OVF is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow interrupt flag.

Bit 8 – ALARMn Alarm n
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_RTC_CNT cycle after a match with the compare condition and will generate an interrupt if INTENCLR/SET.ALARMn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Alarm n interrupt flag.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – PERn Periodic Interval n
This flag is cleared by writing a '1' to it.
This flag is set on the 0-to-1 transition of prescaler bit (n+2) and will generate an interrupt if INTENCLR/SET.PERn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Periodic Interval n interrupt flag.

21.8.6. Debug Control

Name: DBGCTRL
Offset: 0x0E
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Debug Run
This bit is not reset by a software reset.
This bit controls the functionality when the CPU is halted by an external debugger.

Value	Description
0	The RTC is halted when the CPU is halted by an external debugger
1	The RTC continues normal operation when the CPU is halted by an external debugger

21.8.7. Synchronization Busy - MODE2

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	CLOCKSYNC				MASK0			
Access	R				R			
Reset	0				0			

Bit	7	6	5	4	3	2	1	0
			ALARM0		CLOCK	FREQCORR	ENABLE	SWRST
Access			R		R	R	R	R
Reset			0		0	0	0	0

Bit 15 – CLOCKSYNC CLOCK Read Resynchronization Enable Synchronization Busy
This bit is cleared when the synchronization of CTRLA.CLOCKSYNC is complete.
This bit is set when the synchronization of CTRLA.CLOCKSYNC is started.

Note: This bit will only be set when writing to CTRLA.CLOCKSYNC, not as a result of ongoing read resynchronization of the CLOCK register.

Bit 11 – MASKn Alarm n Mask Synchronization Busy
This bit is cleared when the synchronization of the MASKn register is complete.
This bit is set when the synchronization of the MASKn register is started.

Bit 5 – ALARMn Alarm n Value Synchronization Busy
This bit is cleared when the synchronization of the ALARMn register is complete.
This bit is set when the synchronization of the ALARMn register is started.

Bit 3 – CLOCK Clock Value Synchronization Busy
This bit is cleared when the synchronization of the CLOCK register is complete.
This bit is set when the synchronization of the CLOCK register is started.

Bit 2 – FREQCORR Frequency Correction Synchronization Busy
This bit is cleared when the synchronization of the FREQCORR register is complete.
This bit is set when the synchronization of the FREQCORR register is started.

Bit 1 – ENABLE Enable Synchronization Busy
This bit is cleared when the synchronization of CTRLA.ENABLE is complete.
This bit is set when the synchronization of CTRLA.ENABLE is started.

Bit 0 – SWRST Software Reset Synchronization Busy
This bit is cleared when the synchronization of CTRLA.SWRST is complete.
This bit is set when the synchronization of CTRLA.SWRST is started.

21.8.8. Frequency Correction

Name: FREQCORR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection, 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

Value	Description
0	The correction value is positive, that is, frequency will be decreased
1	The correction value is negative, that is, frequency will be increased

Bits 6:0 – VALUE[6:0] Correction Value
This bit field controls the amount of correction applied to the RTC prescaler.
Refer to the *Frequency Correction* section for details on how the correction value is calculated.

Value	Description
0x0	Correction is disabled, and the RTC frequency is unchanged
Other	The RTC frequency is adjusted according to the value

21.8.9. Clock Value - 24H Format - MODE2

Name: CLOCK
Offset: 0x18
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. This register must be written with a 32-bit write access.
2. Refer to the *Clock/Calendar (Mode 2)* section for details on configuring the individual bit fields correctly.
3. Prior to any read access, this register requires continuous read resynchronization to be enabled by writing the CLOCK Read Resynchronization Enable bit in the Control A register (CTRLA.CLOCKSYNC) to '1' in order to read valid clock values.

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
This bit field controls the year of the clock value.

Bits 25:22 – MONTH[3:0] Month
This bit field controls the month of the clock value.

Value	Name	Description
0x1	JAN	January
0x2	FEB	February
0x3	MAR	March
0x4	APR	April
0x5	MAY	May
0x6	JUN	June
0x7	JUL	July
0x8	AUG	August
0x9	SEP	September
0xA	OCT	October
0xB	NOV	November
0xC	DEC	December

Value	Name	Description
Other	—	Reserved

Bits 21:17 – DAY[4:0] Day
This bit field controls the day of the clock value.

Bits 16:12 – HOUR[4:0] Hour
This bit field controls the hour of the clock value.

Bits 11:6 – MINUTE[5:0] Minute
This bit field controls the minute of the clock value.

Bits 5:0 – SECOND[5:0] Second
This bit field controls the second of the clock value.

21.8.10. Clock Value - 12H Format - MODE2

Name: CLOCK
Offset: 0x18
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. This register must be written with a 32-bit write access.
2. Refer to the *Clock/Calendar (Mode 2)* section for details on configuring the individual bit fields correctly.
3. Prior to any read access, this register requires continuous read resynchronization to be enabled by writing the CLOCK Read Resynchronization Enable bit in the Control A register (CTRLA.CLOCKSYNC) to '1' in order to read valid clock values.

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]					INDICATOR
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
This bit field controls the year of the clock value.

Bits 25:22 – MONTH[3:0] Month
This bit field controls the month of the clock value.

Value	Name	Description
0x1	JAN	January
0x2	FEB	February
0x3	MAR	March
0x4	APR	April
0x5	MAY	May
0x6	JUN	June
0x7	JUL	July
0x8	AUG	August
0x9	SEP	September
0xA	OCT	October
0xB	NOV	November
0xC	DEC	December

Value	Name	Description
Other	—	Reserved

Bits 21:17 – DAY[4:0] Day
This bit field controls the day of the clock value.

Bit 16 – INDICATOR Meridiem Indicator
This bit field controls the meridiem indicator (AM/PM) of the clock value.

Value	Name	Description
0	AM	Before midday (AM)
1	PM	After midday (PM)

Bits 15:12 – HOUR[3:0] Hour
This bit field controls the hour of the clock value.

Bits 11:6 – MINUTE[5:0] Minute
This bit field controls the minute of the clock value.

Bits 5:0 – SECOND[5:0] Second
This bit field controls the second of the clock value.

21.8.11. Alarm n Value - 24H Format - MODE2

Name: ALARMn
Offset: 0x20
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Note: Refer to the *Clock/Calendar (Mode 2)* section for details on configuring the individual bit fields correctly.

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
This bit field controls the year of the alarm value.

Bits 25:22 – MONTH[3:0] Month
This bit field controls the month of the alarm value.

Value	Name	Description
0x1	JAN	January
0x2	FEB	February
0x3	MAR	March
0x4	APR	April
0x5	MAY	May
0x6	JUN	June
0x7	JUL	July
0x8	AUG	August
0x9	SEP	September
0xA	OCT	October
0xB	NOV	November
0xC	DEC	December
Other	—	Reserved

Bits 21:17 – DAY[4:0] Day
This bit field controls the day of the alarm value.

Bits 16:12 – HOUR[4:0] Hour
This bit field controls the hour of the alarm value.

Bits 11:6 – MINUTE[5:0] Minute
This bit field controls the minute of the alarm value.

Bits 5:0 – SECOND[5:0] Second
This bit field controls the second of the alarm value.

21.8.12. Alarm n Value - 12H Format - MODE2

Name: ALARMn
Offset: 0x20
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Note: Refer to the *Clock/Calendar (Mode 2)* section for details on configuring the individual bit fields correctly.

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]					INDICATOR
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
This bit field controls the year of the alarm value.

Bits 25:22 – MONTH[3:0] Month
This bit field controls the month of the alarm value.

Value	Name	Description
0x1	JAN	January
0x2	FEB	February
0x3	MAR	March
0x4	APR	April
0x5	MAY	May
0x6	JUN	June
0x7	JUL	July
0x8	AUG	August
0x9	SEP	September
0xA	OCT	October
0xB	NOV	November
0xC	DEC	December
Other	—	Reserved

Bits 21:17 – DAY[4:0] Day
This bit field controls the day of the alarm value.

Bit 16 – INDICATOR Meridiem Indicator
This bit field controls the meridiem indicator (AM/PM) of the alarm value.

Value	Name	Description
0	AM	Before midday (AM)
1	PM	After midday (PM)

Bits 15:12 – HOUR[3:0] Hour
This bit field controls the hour of the alarm value.

Bits 11:6 – MINUTE[5:0] Minute
This bit field controls the minute of the alarm value.

Bits 5:0 – SECOND[5:0] Second
This bit field controls the second of the alarm value.

21.8.13. Alarm n Mask - MODE2

Name: MASKn
Offset: 0x24
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized

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

Bits 2:0 – SEL[2:0] Alarm Mask Selection
This bit field controls which bit fields in the CLOCK and ALARMn registers are used for comparison.

Value	Name	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
Other	—	Reserved

22. TC - Timer/Counter

22.1. Features

- Configurable timer/counter operation
  - 8-bit, 16-bit or 32-bit TC operation, where 32-bit operation requires combining two TC instances
- 2 compare/capture channels (CC) with:
  - Double buffered timer period setting
  - Double buffered compare channel
- Waveform generation
  - Frequency generation
  - Single-slope pulse-width modulation
- Input capture
  - Event or I/O pin edge capture
  - Frequency capture
  - Pulse-width capture
  - Timestamp capture
  - Minimum and maximum capture
- One input event
- Interrupts/output events on:
  - Counter overflow/underflow
  - Compare match or capture
- Internal prescaler
- DMA support

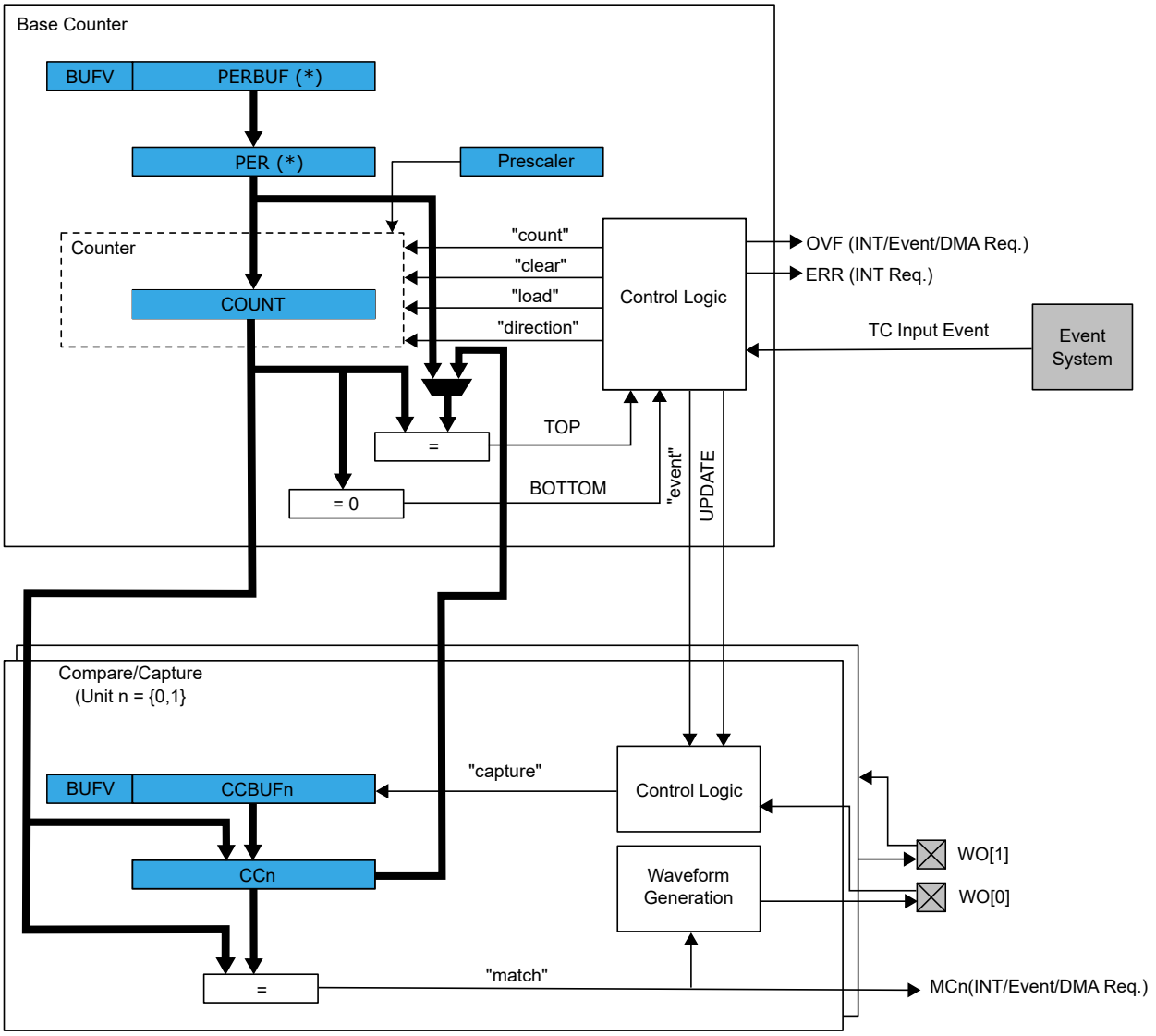
22.2. Overview

Each TC instance consists of a counter, a prescaler, compare/capture channels and control logic. The counter can be set to count events, or clock pulses. The counter, together with the compare/capture channels, can be configured to timestamp input events or I/O pin edges, allowing for capturing of frequency and/or pulse width.

A TC can also perform waveform generation, such as frequency generation and pulse-width modulation (PWM).

22.3. Block Diagram

Figure 22-1. Timer/Counter Block Diagram



Note: (*) PER and PERBUF only exist in 8-bit mode. In 16-bit and 32-bit, MAX will be used in the comparison that triggers TOP.

22.3.1. Signal Description

Table 22-1. Signal Description for TC

Signal Name	Type	Description
WO[1:0]	Digital output	Waveform output
	Digital input	Capture input

Refer to the *Pinout* chapter for details on the pin mapping for this peripheral. One signal can be mapped on several pins.

22.4. Functional Description

22.4.1. Initialization

The following registers are enable-protected, meaning that they can only be written when the TC is disabled (CTRLA.ENABLE =0):

- Control A register (CTRLA), except the Enable (ENABLE) and Software Reset (SWRST) bits
- Driver Control register (DRVCTRL)
- Waveform Generation Control register (WAVE)
- Event Control register (EVCTRL)

Writing to Enable-Protected bits and setting the CTRLA.ENABLE bit can be performed in a single 32-bit access of the CTRLA register. However, writing to Enable-Protected bits and clearing the CTRLA.ENABLE bit cannot be performed in a single 32-bit access.

Before enabling the TC, the peripheral must be configured by the following steps:

1. Enable the TC bus clock (CLK_TCn_APB).
2. Select 8-, 16- or 32-bit counter mode via the TC Mode bit field in the Control A register (CTRLA.MODE). The default mode is 16-bit.
3. Select a waveform generation operation using the Waveform Generation Mode bit field in the Waveform Generation Control register (WAVE.WAVEGEN).
4. If desired, the GCLK_TCn clock can be prescaled via the Prescaler bit field in the Control A register (CTRLA.PRESCALER).
   - If the prescaler is used, select a prescaler synchronization operation via the Prescaler and Counter Synchronization bit field in the Control A register (CTRLA.PRESYNC)
5. If desired, enable one-shot operation by writing a '1' to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT).
6. If desired, configure the counting direction to 'down' (starting from the TOP value) by writing a '1' to the Counter Direction bit in the Control B register (CTRLBSET.DIR).
7. If desired, for capture operation, enable the individual channels for capture by setting the Capture Channel n Enable bit field in the Control A register (CTRLA.CAPTEN).
8. If desired, enable inversion of the waveform output or I/O pin input signal for individual channels by setting the Invert Enable bit field in the Driver Control register (DRVCTRL.INVEN).

Note: Two instances of the TC may share a peripheral clock channel. In this case, they cannot be configured to use different clock frequencies. Refer to the peripheral clock channel mapping of the *Generic Clock Controller (GCLK)* section and the *Peripheral Channel Control (PCHCTRLm)* register section to identify shared peripheral clocks.

22.4.2. Operation

The following definitions are used throughout the documentation:

Table 22-2. Timer/Counter Definitions

Name	Description
TOP	The counter reaches TOP when it becomes equal to the highest value in the count sequence. The TOP value can be the same as Period (PER) or the Compare Channel 0 (CC0) register value depending on the Waveform Generator mode in the *Waveform Output Operations* section
ZERO	The counter is ZERO when it contains all zeroes
MAX	The counter reaches MAX when it contains all ones
UPDATE	The timer/counter signals an update when it reaches ZERO or TOP, depending on the direction settings

Table 22-2. Timer/Counter Definitions (continued)

Name	Description
Timer	TC mode where increment/decrement/clear/reload operations are performed on every prescaled clock cycle
Counter	TC mode where increment/decrement/clear/reload operations are performed on each detected event
CC	In compare operations, the CC registers are referred to as “compare channels.” In capture operations, the CC registers are referred to as “capture channels.”

Each TC instance has 2 compare/capture channels (CC[n]).

The counter in the TC can either count events from the Event System, or clock ticks of the GCLK_TCn clock, which may be divided by the prescaler.

The counter value is passed to the CC[n] where it can be either compared to user-defined values or captured.

The CC[n] registers are using buffer registers (CCBUF[n]) for optimized timing. Each buffer register has a Buffer Valid (BUFV) flag, which signals when the buffer holds a new value.

The Counter register (COUNT) and the Compare and Capture registers, including buffers (CC[n] and CCBUF[n]) can be configured as 8-, 16- or 32-bit registers, each with a corresponding MAX values. The mode settings (CTRLA.MODE) determine the maximum range of the Counter register.

In 8-bit mode, a Period (PER) register and its Period Buffer (PERBUF) register are also available. The counter range and the operating frequency define the maximum time resolution achievable with the TC peripheral.

The TC can be set to count up or down. The counter value is continuously compared to the TOP or ZERO value to determine whether the counter has reached that value. On a comparison match, the TC can request DMA transactions, generate interrupts, or events for the Event System.

In a compare operation, the counter value is continuously compared to the values in the CC[n] registers. When a match occurs, the TC can request DMA transactions, generate interrupts, or events for the Event System. In Waveform Generator mode, these comparisons are used to set the waveform period or pulse width.

Capture operation can be enabled to perform input signal period and pulse width measurements, or to capture selectable edges from an I/O pin or internal event from Event System.

22.4.2.1. Enabling, Disabling, and Resetting

The TC is enabled by writing a ‘1’ to the Enable bit in the Control A register (CTRLA.ENABLE). The TC is disabled by writing a ‘0’ to CTRLA.ENABLE.

The TC is reset by writing a ‘1’ 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. Refer to the CTRLA register for details.

The TC must be disabled before the TC is reset to avoid undefined behavior.

22.4.2.2. Prescaler Selection

The GCLK_TCn is fed into the internal prescaler.

The prescaler consists of a counter that counts up to the selected prescaler value; when this value is reached, the output of the prescaler toggles.

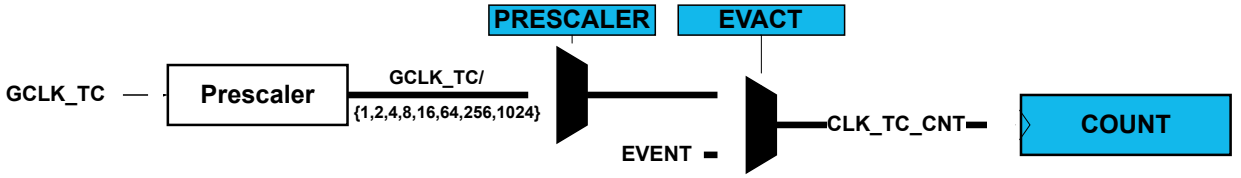
If the prescaler value is higher than one, the counter update condition can optionally be executed on either the next GCLK_TCn clock pulse or the next prescaled clock pulse. For further details, refer to Prescaler (CTRLA.PRESCALER) and Counter Synchronization (CTRLA.PRESYNC) description.

Prescaler outputs from GCLK_TC/1 to GCLK_TC/1024 are available. For a complete list of available prescaler outputs, see the register description for the Prescaler bit field in the Control A register (CTRLA.PRESCALER).

Note: When counting events, the prescaler is bypassed.

The combined stream of prescaler ticks and event action ticks is referred to as CLK_TC_CNT.

Figure 22-2. Prescaler



22.4.2.3. Counter Mode

The counter mode is selected by the Timer Counter Mode bit field in the Control A register (CTRLA.MODE). By default, the counter is set to 16-bit counter resolution. Three counter resolutions are available:

- COUNT8: 8-bit counter mode
- COUNT16: 16-bit counter mode
- COUNT32: 32-bit counter mode, achieved by pairing two 16-bit TC instances
  When paired, the TC instances are configured using the registers of the even-numbered TCn. The odd-numbered TCn instance acts as a client, and the Client bit in the Status register (STATUS.SLAVE) will be set. The register values of a client does not reflect the registers of the 32-bit counter. Writing to any of the client registers does not affect the 32-bit counter. Normal access to the client COUNT and CC[n] registers is not allowed.
  Note: If the highest-numbered TCn instance is even-numbered, this instance cannot be configured as a 32-bit counter.

22.4.2.4. Counter Operations

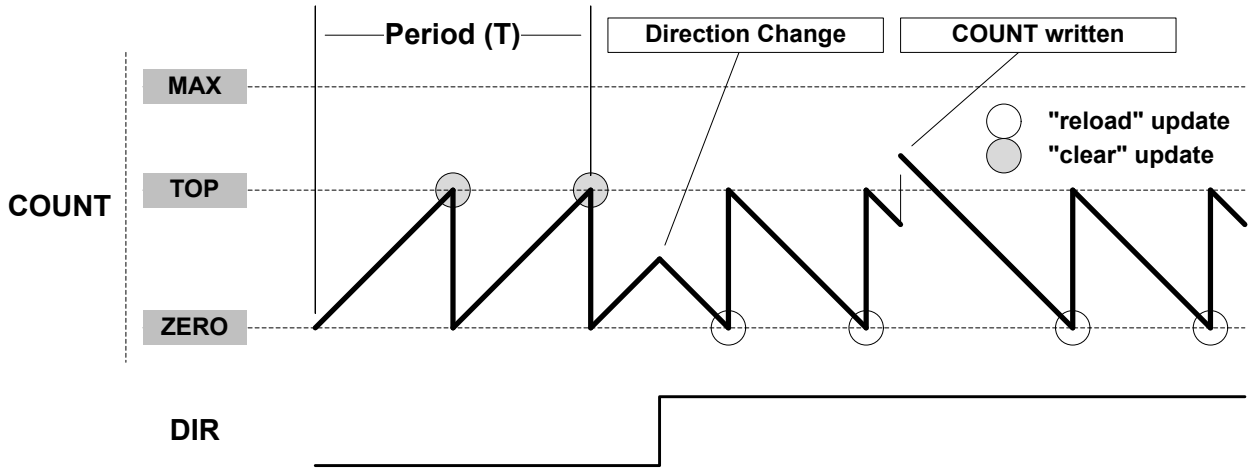
Depending on the mode of operation, the counter is cleared, reloaded, incremented, or decremented at each TC clock input (CLK_TC_CNT). A counter clear or reload marks the end of the current counter cycle and the beginning of a new one.

The counting direction is set by the Direction bit in the Control B Set register (CTRLBSET.DIR) and Control B Clear register (CTRLBCLR.DIR). The counter will count up by writing a '1' to CTRLBCLR.DIR bit, or count down by writing a '1' to CTRLBSET.DIR. The counter will count up or down for each tick (clock or event) until it reaches TOP or ZERO. When it is counting up and TOP is reached, the counter will be set to zero at the next tick (overflow), and the Overflow Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF) will be set. When it is counting down, the counter is reloaded with the TOP value when ZERO is reached (underflow), and INTFLAG.OVF is set.

INTFLAG.OVF can be used to trigger an interrupt, a DMA request, or an event. An overflow or underflow occurrence (i.e., a compare match with TOP/ZERO) will stop counting if the One-Shot bit in the Control B register is set (CTRLBSET.ONESHOT).

It is possible to change the counter value by writing directly in the COUNT register, even when the counter is running. When starting the TC, the COUNT value will be either ZERO or TOP (depending on the counting direction set by CTRLBSET.DIR or CTRLBCLR.DIR), unless a different value has been written to it, or the TC has been stopped at a value other than ZERO. Write access to COUNT register has higher priority than count, clear, or reload. The direction of the counter can also be changed when the counter is running. See the following figure.

Figure 22-3. Counter Operation



Due to asynchronous clock domains, internal counter settings are written only after the synchronization is complete.

22.4.2.4.1. Stop Command and Event Action

A Stop command can be issued from software by setting the Command bit field in the Control B Set register (CTRLBSET.CMD = 0x2, STOP). When a Stop is detected while the counter is running, the counter will be loaded with the starting value (ZERO or TOP, depending on direction set by CTRLBSET.DIR or CTRLBCLR.DIR). All waveforms are cleared, and the Stop bit in the Status register (STATUS.STOP) is set.

22.4.2.4.2. Retrigger Command and Event Action

A retrigger command can be issued from software by setting the Command bit field in the Control B Set register (CTRLBSET.CMD = 0x1, RETRIGGER), or from event when a retrigger event action is configured in the Event Control register (EVCTRL.EVACT = 0x1, RETRIGGER).

When the retrigger command is detected during counting operation, the counter will be reloaded or cleared, depending on the counting direction (CTRLBSET.DIR or CTRLBCLR.DIR). When the retrigger command is detected while the counter is stopped, the counter will resume counting from the current value in the COUNT register.

Note: When a retrigger event action is configured in the Event Action bits in the Event Control register (EVCTRL.EVACT = 0x1, RETRIGGER), enabling the counter will not start the counter. The counter will start on the next incoming event and will restart on the corresponding following event.

22.4.2.4.3. Count Event Action

The TC can count events. When an event is received, the counter increases or decreases the value, depending on the direction settings (CTRLBSET.DIR or CTRLBCLR.DIR). The count event action can be selected by setting the Event Action bit field in the Event Control register (EVCTRL.EVACT=0x2, COUNT).

Note: If this operation mode is selected, PWM generation is not supported.

22.4.2.4.4. Start Event Action

The TC can start counting operation on an event. In this configuration, the event has no effect if the counter is already counting. When the peripheral is enabled, the counter operation starts when the event is received or when a retrigger software command is applied.

The “Start TC on Event action” can be selected by setting the Event Action bit field in the Event Control register (EVCTRL.EVACT=0x3, START).

22.4.2.5. Compare Operations

By default, the Compare/Capture channel is configured for compare operations.

When using the TC and the Compare/Capture Value registers (CC[n]) for compare operations, the counter value is continuously compared to the values in the CC[n] registers. This functionality can be used for timer or for waveform operation.

The Channel n Compare Buffer (CCBUF[n]) registers provide double buffering capability. The double buffering synchronizes the update of the CC[n] register with the buffer value at the UPDATE condition or a forced update command (CTRLBSET.CMD = 0x3,UPDATE). For further details, refer to Double Buffering. This synchronization prevents the occurrence of odd-length, non-symmetrical pulses and ensures glitch-free output.

22.4.2.5.1. Waveform Output Operations

The compare channels can be used for waveform generation on output port pins. To make the waveform available on the connected pin, the following requirements must be fulfilled:

1. Choose a waveform generation mode using the Waveform Generation Mode bit field in the Waveform Generation Control register (WAVE.WAVEGEN).
2. Optionally invert the waveform output WO[n] by setting the corresponding Waveform Output n Invert Enable bit in the Driver Control register (DRVCTRL.INVENn).
3. Configure the pins as output using the I/O Pin Controller. Refer to the *I/O Pin Controller (PORT)* section for details.

The counter value is continuously compared with each CC[n] value. On a comparison match, the Match or Capture Channel n bit in the Interrupt Flag Status and Clear register (INTFLAG.MCn) will be set (see Normal Frequency Operation). An interrupt and/or event can be generated on comparison match if enabled. The same condition also generates a DMA request.

There are four waveform configurations for the Waveform Generation Operation bit group in the Waveform register (WAVE.WAVEGEN). These configurations influence how the waveform is generated and impose restrictions on the TOP value. The available configurations are:

- Normal frequency (NFRQ)
- Match frequency (MFRQ)
- Normal pulse-width modulation (NPWM)
- Match pulse-width modulation (MPWM)

When using NPWM or NFRQ configuration, the TOP value is determined by the counter resolution. In 8-bit Counter mode, the Period register (PER) is used as TOP, and the TOP can be changed by writing to the PER register. In 16- and 32-bit Counter mode, TOP is fixed to the maximum (MAX) value of the counter.

The table below shows the update counter and overflow event/interrupt generation conditions in different operation modes.

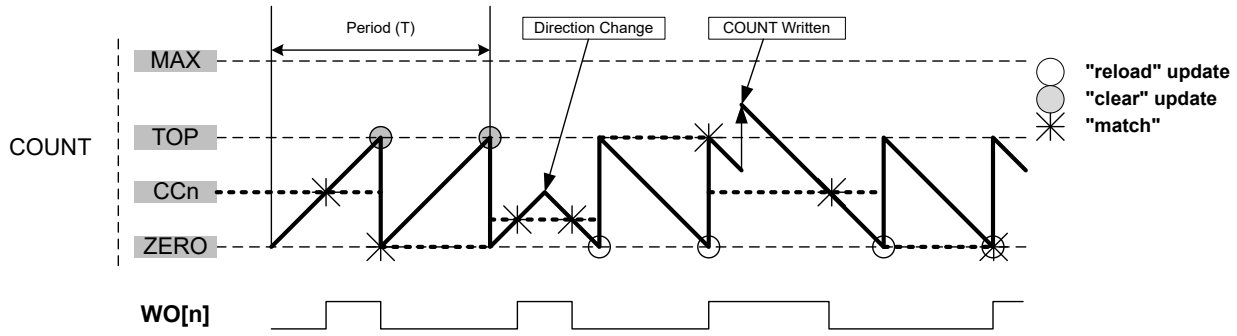
Table 22-3. Counter Update and Overflow Event/interrupt Conditions in TC

Name	Operation	TOP	Update	Output Waveform		OVF Interrupt/Event	
				On Match	On Update	Up	Down
NFRQ	Normal Frequency	PER	TOP/ ZERO	Toggle	Stable	TOP	ZERO
MFRQ	Match Frequency	CC[0]	TOP/ ZERO	Toggle	Stable	TOP	ZERO
NPWM	Normal PWM	PER	TOP/ ZERO	See description below		TOP	ZERO
MPWM	Match PWM	CC[0]	TOP/ ZERO	See description below		TOP	ZERO

Normal Frequency Generation (NFRQ)

For Normal Frequency Generation, the period time (T) is controlled by the TOP value. The waveform output (WO[n]) is toggled on each compare match between COUNT and CC[n], and the corresponding Match or Capture Channel n Interrupt Flag (INTFLAG.MCn) will be set.

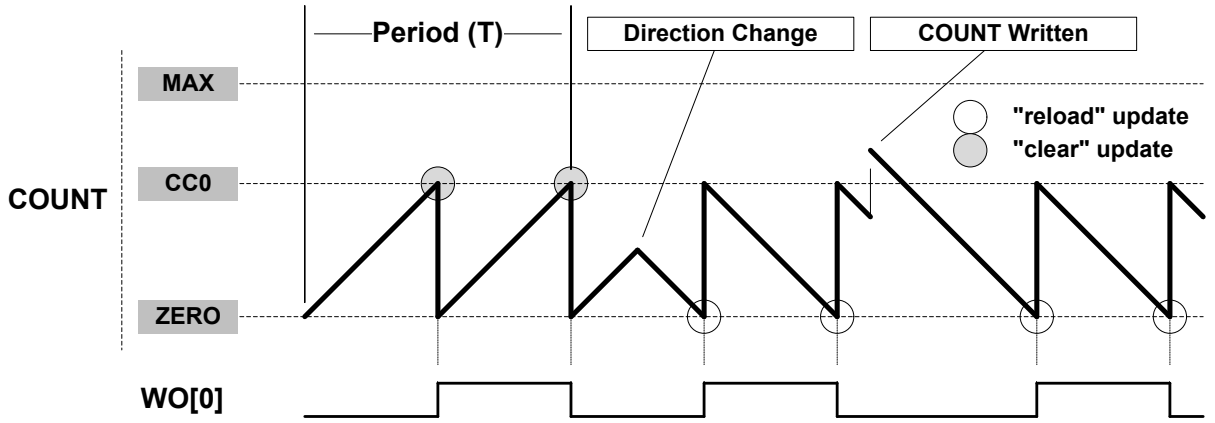
Figure 22-4. Normal Frequency Operation



Match Frequency Generation (MFRQ)

For Match Frequency Generation, the period time (T) is controlled by the CC[0] register instead of TOP. WO[0] toggles on each update condition.

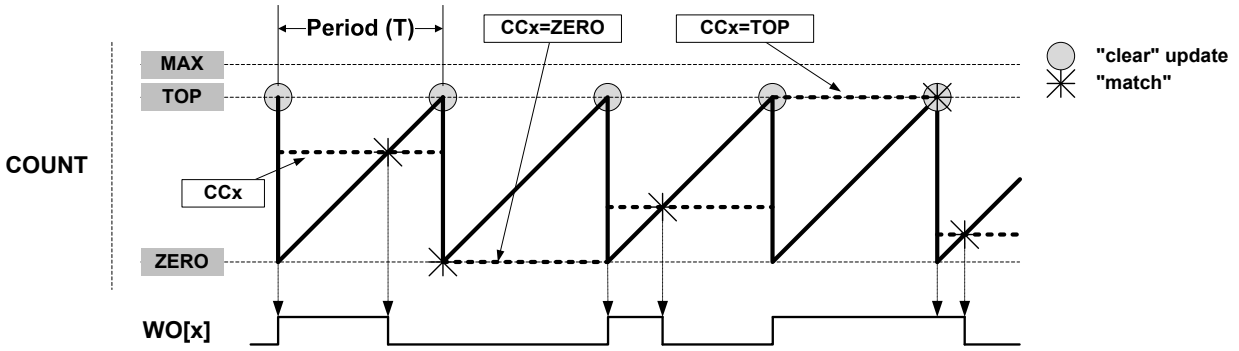
Figure 22-5. Match Frequency Operation



Normal Pulse-Width Modulation Operation (NPWM)

NPWM uses single-slope PWM generation.

For single-slope PWM generation, the period time (T) is controlled by the TOP value, and the CC[n] controls the duty cycle of the generated waveform output. When counting upwards, WO[n] is set at the start or on compare match between the COUNT and TOP values, and cleared on a compare match between COUNT and CC[n] register values. When counting downwards, the WO[n] is cleared at the start or on a compare match between the COUNT and ZERO values, and set on compare match between COUNT and the CC[n] register value.



The following equation calculates the exact resolution in bits for a single-slope PWM (R_{PWM_SS}) waveform:

$$R_{PWM\_SS} = \frac{\log(TOP+1)}{\log(2)}$$

The PWM frequency (f_{PWM_SS}) depends on the TOP value and the peripheral clock frequency (f_{GCLK_TC}), and can be calculated by the following equation:

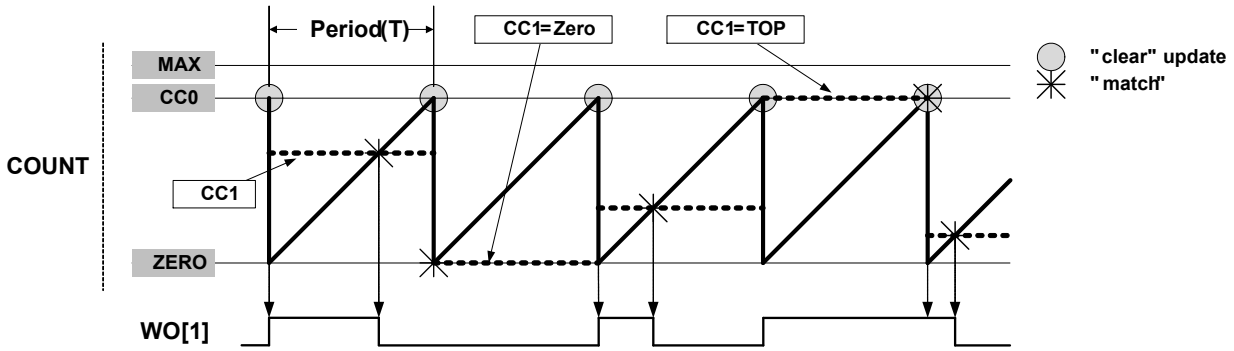
$$f_{PWM\_SS} = \frac{f_{GCLK\_TC}}{N(TOP+1)}$$

Where N represents the prescaler divider used (1, 2, 4, 8, 16, 64, 256, 1024).

Match Pulse-Width Modulation Operation (MPWM)

In MPWM mode, the output of WO[1] depends on CC1 as shown in the figure below. On every overflow or underflow, a one-TC-clock-cycle negative pulse is output on WO[0] (not shown in the figure).

Figure 22-6. Match PWM Operation



22.4.2.6. Double Buffering

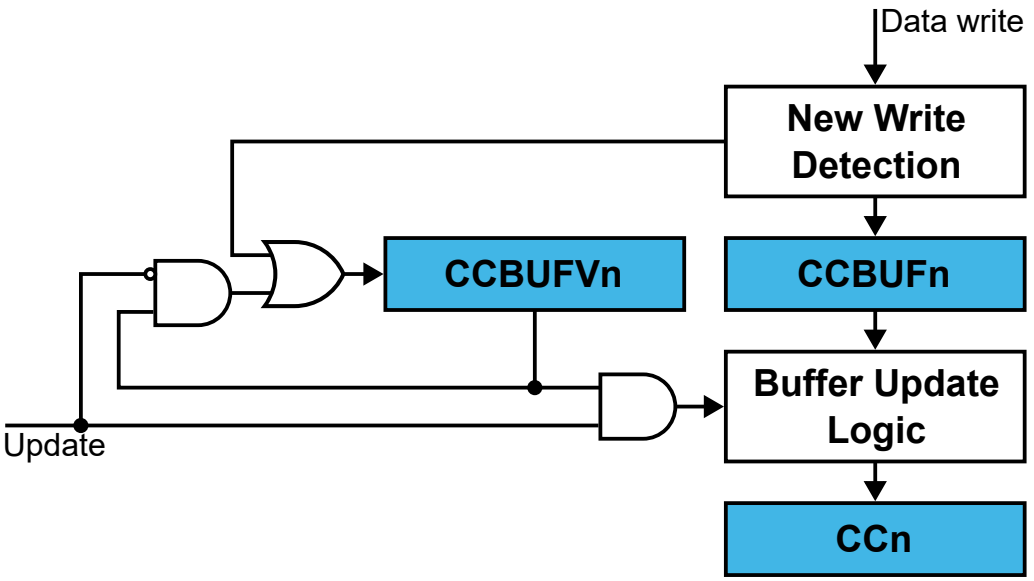
The Compare Channels (CC[n]) registers and the Period (PER) register are double buffered. Each buffer register has a buffer valid bit (CCBUFVn or PERBUFV) in the STATUS register, which indicates that the buffer register contains a new valid value that can be copied into the corresponding register. As long as the respective buffer valid status flags (PERBUFV or CCBUFVn) are set to '1', the related Synchronization Busy bits (SYNCBUSY.PER or SYNCBUSY.CCn) are set (SYNCBUSY.PER or SYNCBUSY.CCn). A write to the respective PER/PERBUF or CC[n]/CCBUF[n] registers will generate a PAC error, and access to the respective PER or CC[n] register is invalid.

When the Buffer Valid Flag bit in the STATUS register is '1', the data from buffer registers will be copied into the corresponding register under hardware UPDATE conditions. After the update the Buffer Valid Flag bits in the STATUS register are automatically cleared by hardware. This automatic update through hardware can be disabled by writing a '1' to the Lock Update bit in the Control B Set register (CTRLBSET.LUPD).

Note: The software update command (CTRLBSET.CMD=0x3) is acting independently of the LUPD value and can force an update even if hardware updates are locked.

A compare register is double buffered, as in the following figure.

Figure 22-7. Compare Channel Double Buffering

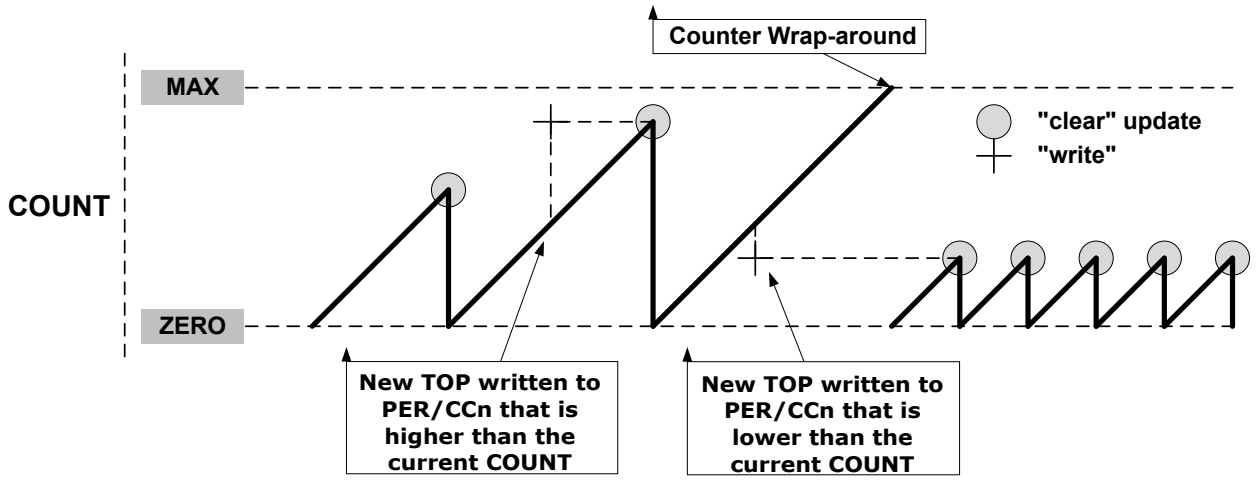


Both the registers (PER/CC[n]) and the corresponding buffer registers (PERBUF/CCBUF[n]) are available in the I/O register map, and the double buffering feature is not mandatory. Double buffering is disabled by writing a '1' to the Lock Update bit in the Control B Set register (CTRLBSET.LUPD).

Changing the Period

The counter period can be changed by writing a new TOP value to the Period register (PER or CC0, depending on the waveform generation and TC mode). Any TOP value update to the registers (PER or CC[n]) becomes effective only after the synchronization delay.

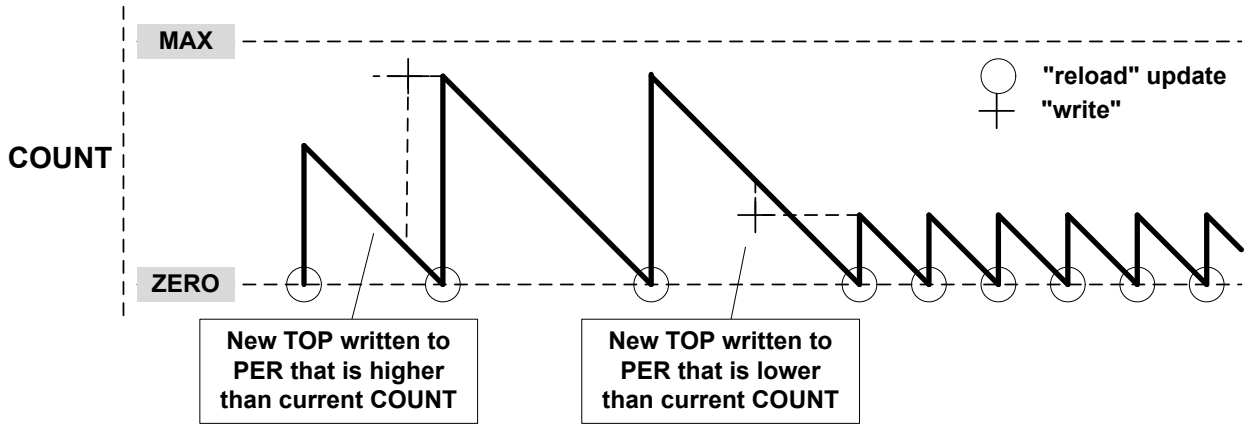
Figure 22-8. Unbuffered Single-Slope Up-Counting Operation



A counter wrap-around can occur in any operation mode when counting upwards without buffering, see Figure 22-8.

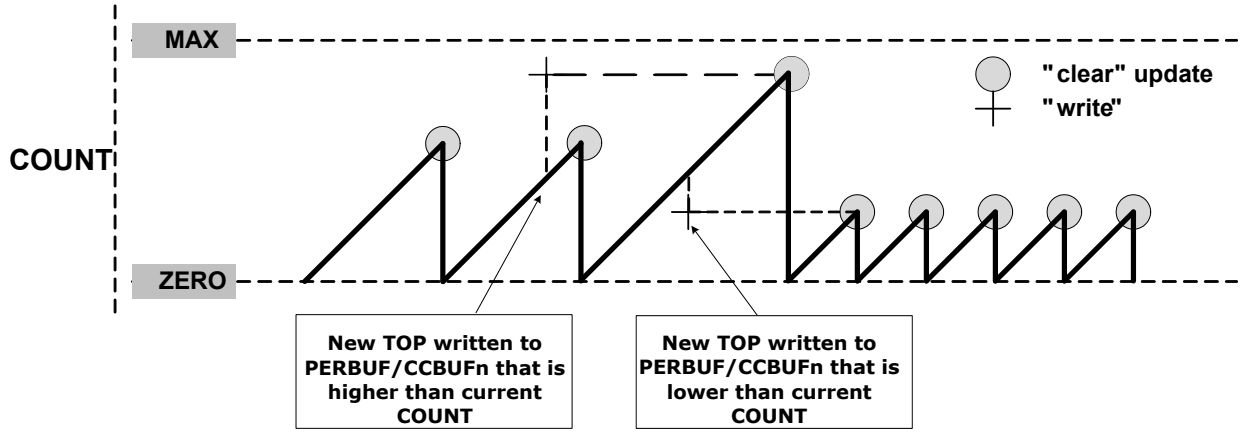
COUNT and TOP are continuously compared, so when a new TOP value that is lower than the current COUNT is written to TOP, COUNT will wrap before a compare match occurs.

Figure 22-9. Unbuffered Single-Slope Down-Counting Operation



When double buffering is used, the buffer register (PERBUF or CCBUF[n]) can be written at any time and the counter will still maintain correct operation. The period register (PER or CC[n]) is always updated on the update condition, as shown in Figure 22-10. This prevents wrap-around and the generation of odd waveforms.

Figure 22-10. Changing the Period Using Buffering



22.4.2.7. Capture Operations

To enable and use capture operations, the corresponding Capture Channel n Enable bit in the Control A register (CTRLA.CAPTENn) must be set to '1'.

A capture trigger can be provided by the input event line (TC_EV), or by an asynchronous I/O pin (WO[n]) for each capture channel, or by a TC event. To enable capture from the input event line, the Event Input Enable bit in the Event Control register (EVCTRL.TCEI) must be set to '1'. To enable capture from the I/O pin, the Capture On Pin x Enable bit in the CTRLA register (CTRLA.COPENn) must be set to '1'.

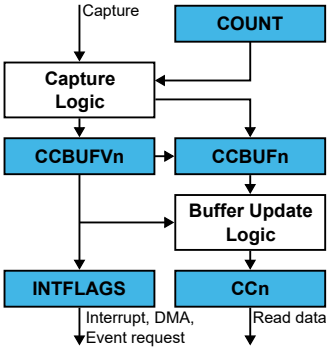
Notes:

1. Capture on I/Os is only possible in 'Event' and 'Time-Stamp' capture action modes. Other modes can only use internal events (if I/O toggling is needed in other modes, then the I/O edge must be configured for generating internal events).
2. Capture on an event from the Event System is possible in Event, PPW/PWP/PW, and Time-Stamp capture action modes. In this case, the event system channels must be configured to operate in asynchronous mode of operation.
3. Depending on CTRLA.COPENn, channel n can be configured for I/Os or internal event capture (both are mutually exclusive). One channel can be configured for I/Os capture while the other uses internal event capture.
4. Capture of the period and duty cycle on I/Os using PPW/PWP mode is possible using EIC and Event System.

By default, a capture operation is performed when a rising edge is detected on the input signal. Capture on the falling edge is also available, its activation depends on the input source:

- When the channel is used with an I/O pin, write a '1' to the corresponding Invert Enable bit in the Drive Control register (DRVCTRL.INVENn)
- When the channel is counting events from the Event System, write a '1' to the TC Event Input Invert Enable bit in the Event Control register (EVCTRL.TCINV)

Figure 22-11. Capture Double Buffering

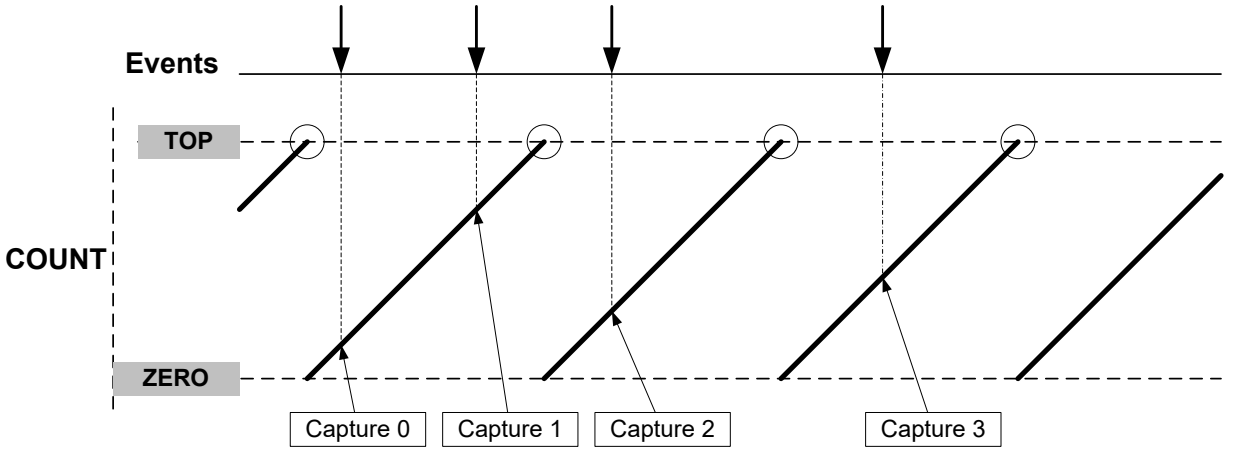


For input capture, the buffer register (CCBUF[n]) and the corresponding compare/capture register (CC[n]) function as a FIFO. When CC[n] is empty or read, any data in CCBUF[n] is transferred to CC[n]. The buffer valid flag is used to set the CC[n] interrupt flag (IF) and generate an optional interrupt, event, or DMA request. The CCBUF[n] register value can't be read, all captured data must be read from the CC[n] register.

22.4.2.7.1. Event Capture Action on Events or I/Os

The compare/capture channels can be used as input capture channels to capture events from the Event System or I/O pins and assign them a timestamp. This mode is selected when EVTCTRL.EVACT is configured either as OFF, RETRIGGER, COUNT or START. The following figure shows four capture events for one capture channel.

Figure 22-12. Input Capture Timing



The TC can detect capture overflow on the input capture channels: If a new capture event occurs while the Capture Interrupt flag (INTFLAG.MCn) is still set, the new timestamp will not be stored, and INTFLAG.ERR will be set.

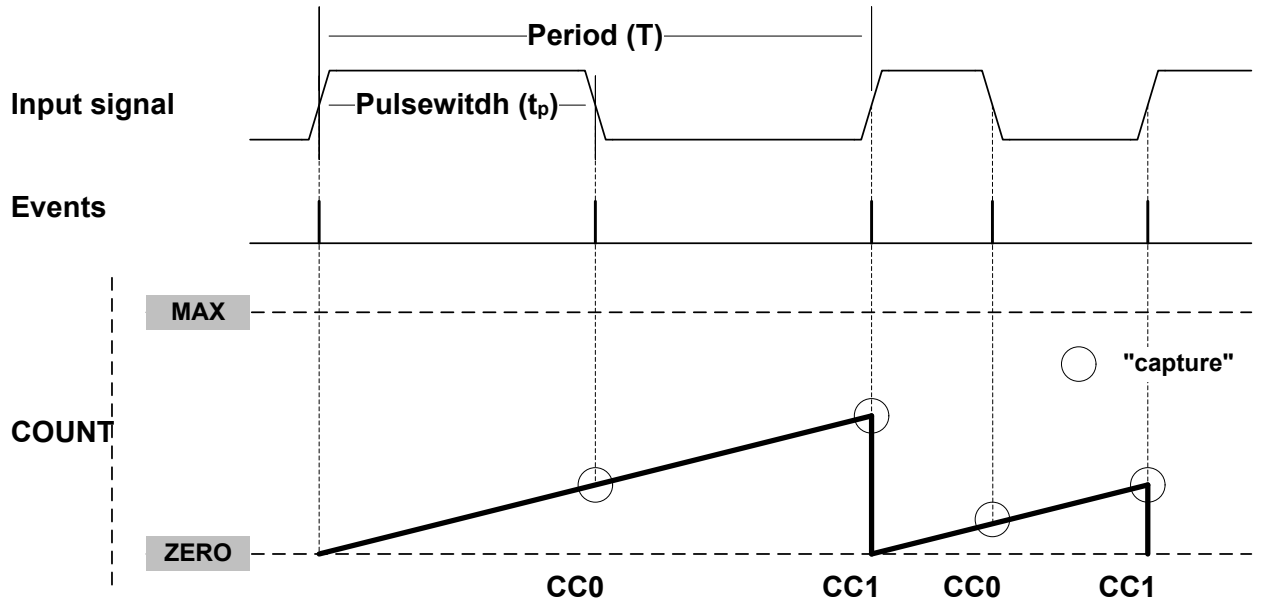
22.4.2.7.2. Period and Pulse-Width (PPW/PWP) Capture Action on Events

The TC can perform two input captures and restart the counter on one of the edges. This allows the TC to measure the pulse width and period, enabling characterization of the input signal's frequency f and duty cycle:

$$f = \frac{1}{T}$$

$$\text{dutyCycle} = \frac{t_p}{T}$$

Figure 22-13. PWP Capture



Selecting PWP in the Event Action bit field in the Event Control register (EVCTRL.EVACT) enables the TC to perform one capture action on the rising edge and the another on the falling edge. The period

T will be captured in CC1, and the pulse width t_p in CC0. When EVCTRL.EVACT is set to PPW (period and pulse-width), the functionality is identical, except that *T* is captured in CC0 and t_p in CC1.

The TC Event Input Invert Enable bit in the Event Control register (EVCTRL.TCINV) is used to select whether the wrap-around occurs on the rising edge or the falling edge. If EVCTRL.TCINV=1, the wrap-around occurs on the falling edge.

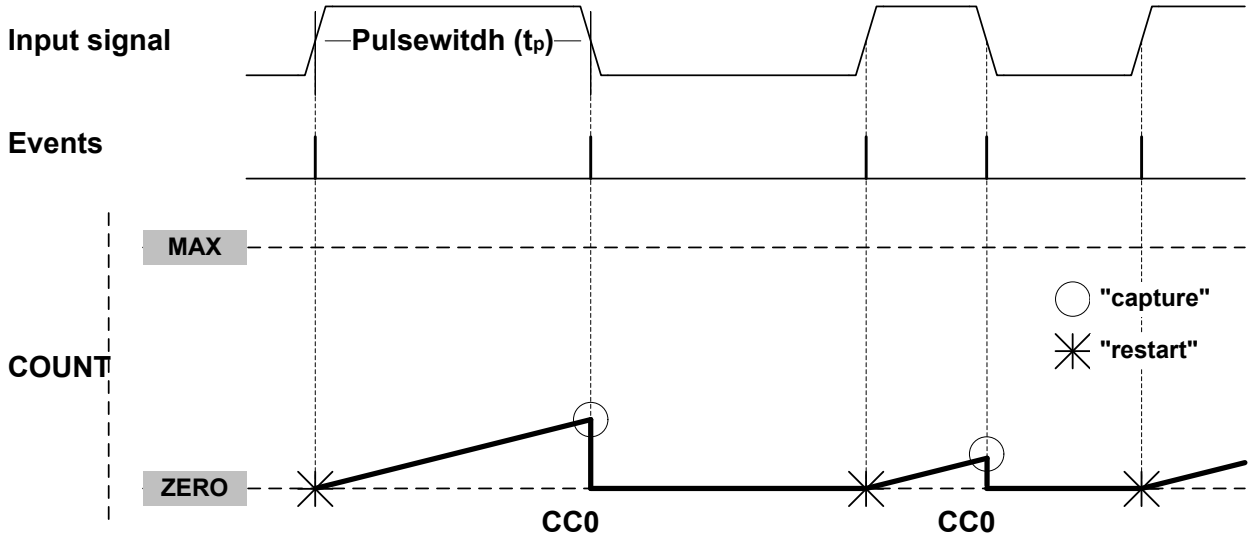
The TC can detect capture overflow of the input capture channels: If a new capture event occurs while the Capture Interrupt flag (INTFLAG.MCn) is still set, the new timestamp will not be stored, and INTFLAG.ERR will be set.

Note: The corresponding capture fucntion operates only if the channel is enabled in Capture mode (CTRLA.CAPTENn=1). Consequently, both channels must be enabled in order to fully characterize the input.

22.4.2.7.3. Pulse-Width (PW) Capture Action on Events

The TC performs input capture on the falling edge of the input signal. When this edge is detected, the counter value is cleared and the TC stops counting. When a rising edge is detected on the input signal, the counter resumes counting. To enable the operation on opposite edges, the input signal for capture must be inverted (refer to DRVCTRL.INVEN or EVCTRL.TCEINV).

Figure 22-14. Pulse-Width Capture on Channel 0



The TC can detect capture overflow of the input capture channels: If a new capture event occurs while the Capture Interrupt flag (INTFLAG.MCn) is still set, the new timestamp will not be stored, and INTFLAG.ERR will be set.

22.4.3. Additional Features

22.4.3.1. One-Shot Operation

When one-shot is enabled, the counter automatically stops at the next counter overflow or underflow condition. When the counter stopps, the Stop bit in the Status register (STATUS.STOP) is automatically set, and the waveform outputs are set to zero.

One-shot operation is enabled by writing a '1' to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT), and disabled by writing a '1' to CTRLBCLR.ONESHOT. When enabled, the TC will count until an overflow or underflow occurs and stops the counting operation. The one-shot operation can be restarted by a retrigger software command, a retrigger event, or a start event. When the counter restarts, STATUS.STOP is automatically cleared.

22.4.3.2. Time-Stamp Capture on Events or I/Os

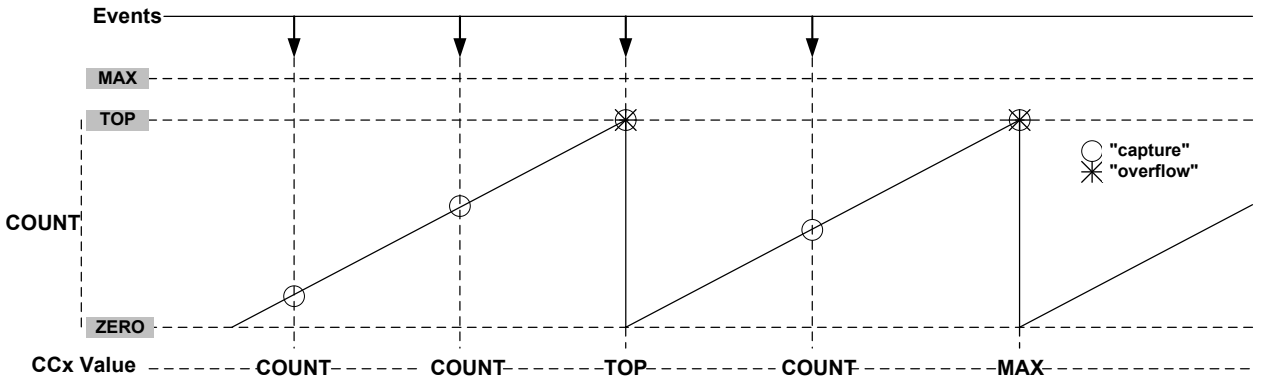
This feature is enabled when the Capture Time Stamp (STAMP) Event Action is selected in Event Control register (EVCTRL.EVACT). The counter TOP value must be smaller than MAX.

When a capture event is detected, the COUNT value is copied into the corresponding Channel x Compare/Capture Value (CC[n]) register. In the event of an overflow, the MAX value is copied into the corresponding CC[n] register.

When a valid captured value is present in the capture channel register, the corresponding Capture Channel n Interrupt Flag (INTFLAG.MCn) is set.

The TC can detect capture overflow of the input capture channels: When a new capture event is detected while the Capture Channel interrupt flag (INTFLAG.MCn) is still set, the new timestamp will not be stored, and INTFLAG.ERR will be set.

Figure 22-15. Time-Stamp



22.4.3.3. Minimum Capture

Minimum capture is enabled by selecting the CAPTMIN mode in the Channel n Capture Mode bits of the Control A register (CTRLA.CAPTMODEn = CAPTMIN).

CC[n] Content:

In CAPTMIN operations, the CC[n] register stores the Minimum captured values. Before enabling this capture mode, the user must initialize the corresponding CC[n] register value to a value other than ZERO. If the initial value of CC[n] register is ZERO, no captures will be performed using the corresponding channel.

MCn Behaviour:

In CAPTMIN operation, a capture is performed only if, at the time of capture event time, the counter value is lower than the last captured value. The MCn interrupt flag is set only if, at the time of capture event time, the counter value is greater or equal to the value captured during the previous event. Therefore, interrupt flag is set when a new absolute Minimum value has been detected.

22.4.3.4. Maximum Capture

Maximum capture is enabled by selecting the CAPTMAX mode in the Channel n Capture Mode bits in the Control A register (CTRLA.CAPTMODEn = CAPTMAX).

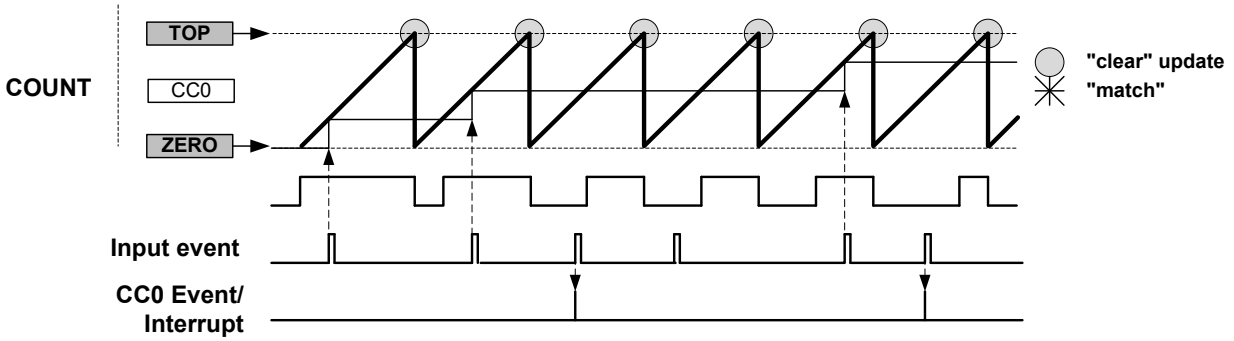
CC[n] Content:

In CAPTMAX operations, CC[n] register stores the Maximum captured values. Before enabling this capture mode, the user must initialize the corresponding CC[n] register value to a value other than TOP. If the initial value of CC[n] register is TOP, no captures will be performed on the corresponding channel.

MCn Behaviour:

In CAPTMAX operation, a capture is performed only if, at the time of capture event time, the counter value is greater than the last captured value. The MCn interrupt flag is set only if, at the time of capture event time, the counter value is lower or equal to the value captured on the previous event. So interrupt flag is set when a new absolute maximum value is detected.

Figure 22-16. Maximum Capture Operation with CC0 Initialized with ZERO Value



22.4.4. Sleep Mode Operation

The TC can be configured to operate in any sleep mode. To enable operation in Standby mode, the RUNSTDBY bit in the Control A register (CTRLA.RUNSTDBY) must be '1'. This peripheral can wake up the device from any sleep mode using interrupts, or perform actions through the Event System.

If the On Demand bit in the Control A register (CTRLA.ONDEMAND) is set to '1', the module stops requesting its peripheral clock when the STOP bit in STATUS register (STATUS.STOP) is set to '1'. When a retrigger or start condition is detected, the TC requests the clock before the operation begins.

22.4.5. Debug Operation

When the CPU is halted in Debug mode, this peripheral will halt normal operation. This peripheral can be forced to continue operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

22.5. Dependencies

22.5.1. I/O Lines

To use the TC's I/O lines, the corresponding I/O pins must be properly configured. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

22.5.2. Power Management

The TC will continue to operate in any sleep mode as long as the selected source clock remains running. The TC'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 the *PM – Power Manager* chapter for details on the different sleep modes.

22.5.3. Clocks

The TC bus clock (CLK_TCn_APB) can be enabled or disabled in the Main Clock. The default state of CLK_TCn_APB is described in the *Peripheral Clock Masking* section of the *MCLK – Main Clock* chapter.

A generic clock (GCLK_TCn) is required to clock the TC. This clock must be configured and enabled in the generic clock controller before using the TC. Refer to the *GCLK – Generic Clock Controller* chapter for details.

Note: The TC instances that can be configured for 32-bit operation mode share a peripheral clock channel. These two instances cannot be set to different clock frequencies, regardless of which operation mode they use.

These generic clocks are asynchronous to the user interface clock (CLK_TCn_APB). Because of this asynchronicity, writes to certain registers require synchronization between the clock domains.

22.5.4. DMA

The TC can generate the following DMA requests:

- Overflow (OVF): A DMA request is generated when an update condition (overflow, underflow or retrigger) is detected. The request is cleared by hardware upon DMA acknowledge.
- Match or Capture Channel n (MCn): For a compare channel, the DMA request is generated on each compare match detection and is cleared by hardware on the DMA acknowledge. For a capture channel, the DMA request is generated when valid data is present in the CC[n] register and is cleared when the CC[n] register is read.

The TC can change its DMA request trigger behavior by setting the DMA One-shot Trigger Mode bit in the Control A (CTRLA.DMAOS) register. In one-shot mode, DMA requests will only be generated on an OVF condition after writing '0x5' to the Command bit field of the Control B Set (CTRLBSET.CMD) register. After a request is generated, another one-shot command must be issued to generate a subsequent OVF DMA request. Issuing multiple one-shot commands before an OVF condition occurs has no effect and the TC will only generate one DMA request once the OVF condition occurs.

Note: After issuing a one-shot command (CTRLBSET.CMD=0x5), a request will only be generated on the next OVF condition. Any OVF condition that occurred before writing the command will not have any effect.

Note: While CTRLA.DMAOS is '1' MCn DMA requests are disabled.

The DMA request lines are connected to the DMA Controller (DMAC). Using the TC DMA requests needs to have the DMAC configured first. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

22.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use TC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 22-4. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
TCn	OVF	An overflow condition occurs	The Overflow Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.OVF) must be set
	ERR	A new capture occurs on a channel when the corresponding Match or Capture Channel x interrupt flag = '1'	The Error Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.ERR) must be set
	MCn	A match with the compare condition or once CCn register contain a valid capture value	Match or Capture Channel n Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.MCn) must be set

22.5.6. Events

The TC can generate the following events:

Table 22-5. Event Generators in TC

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCn	OVF	Overflow or Underflow	Pulse	GCLK_TCn	One GCLK_TCn period
	MC_n	Compare Match or Capture on compare/ capture channel n	Pulse	GCLK_TCn	One GCLK_TCn period

Note: The conditions for generating an event are identical to those that will raise the corresponding interrupt flag in the TCn.INTFLAGS register.

Writing a '1' to an Event Output Enable bit in the Event Control Register (EVCTRL.xxEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The TC can connect to the following events:

Table 22-6. Event Users

User Name		Description	Input Detection	Channel Path Type
Peripheral	Input			
TCn	EVU	Event Input	Level	Asynchronous Synchronous Resynchronized

Writing a '1' to an Event Input Enable bit in the Event Control register (EVCTRL.xxEI) enables the corresponding action on input event. Writing a '0' to this bit disables the corresponding action on input event.

Note: EVn input detection depends on the configuration of the Timer/Counter Event n Invert Enable bit in the Event Control register (EVCTRL.TCINVn) and the Timer/Counter Event Input n Action bit fields in the Event Control register (EVCTRL.EVACTn).

The event users can trigger the following actions:

Table 22-7. Event Actions

Event Input	Event Action	Description
EVU	RETRIGGER	Counter retrigger
	COUNT	Count on event (increment or decrement, depending on counter direction)
	START	Counter start - Start counting on the event rising edge. Further events will not restart the counter; the counter will keep on counting using prescaled GCLK_TCn, until it reaches TOP or ZERO, depending on the direction.
	STAMP	Capture Time-stamp (overflow)
	PPW	Period and pulse width capture. The period is captured in CC0, and the pulse width is captured in CC1.
	PWP	Pulse width and period capture. The period is captured in CC1, and the pulse width is captured in CC0.
	PW	Pulse width capture

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

22.5.7. Analog Connections

Not applicable.

22.6. Register Summary - TCn.COUNT8

Timer/Counter in 8-bit mode (COUNT8)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	ONDEMAND	RUNSTDBY	PRESCSYNC[1:0]		MODE[1:0]		ENABLE	SWRST
		15:8	DMAOS				ALOCK	PRESCALER[2:0]		
		23:16			COPEN1	COPEN0			CAPTEN1	CAPTEN0
		31:24				CAPTMODE1[1:0]			CAPTMODE0[1:0]	
0x04	CTRLBCLR	7:0	CMD[2:0]					ONESHOT	LUPD	DIR
0x05	CTRLBSET	7:0	CMD[2:0]					ONESHOT	LUPD	DIR
0x06	EVCTRL	7:0			TCEI	TCINV		EVACT[2:0]		
		15:8			MCEO1	MCEO0				OVFEO
0x08	INTENCLR	7:0			MC1	MC0			ERR	OVF
0x09	INTENSET	7:0			MC1	MC0			ERR	OVF
0x0A	INTFLAG	7:0			MC1	MC0			ERR	OVF
0x0B	STATUS	7:0			CCBUFV1	CCBUFV0	PERBUFV		SLAVE	STOP
0x0C	WAVE	7:0							WAVEGEN[1:0]	
0x0D	DRVCTRL	7:0							INVEN1	INVEN0
0x0E	Reserved									
0x0F	DBGCTRL	7:0								DBGRUN
0x10	SYNCBUSY	7:0	CC1	CC0	PER	COUNT	STATUS	CTRLB	ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x14	COUNT	7:0	COUNT[7:0]							
0x15 ... 0x1A	Reserved									
0x1B	PER	7:0	PER[7:0]							
0x1C	CC[0]	7:0	CC[7:0]							
0x1D	CC[1]	7:0	CC[7:0]							
0x1E ... 0x2E	Reserved									
0x2F	PERBUF	7:0	PERBUF[7:0]							
0x30	CCBUF[0]	7:0	CCBUF[7:0]							
0x31	CCBUF[1]	7:0	CCBUF[7:0]							

22.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized, Enable-Protected

Bit	31	30	29	28	27	26	25	24
				CAPTMODE1[1:0]			CAPTMODE0[1:0]	
Access				R/W	R/W		R/W	R/W
Reset				0	0		0	0

Bit	23	22	21	20	19	18	17	16
			COPEN1	COPEN0			CAPTEN1	CAPTEN0
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bit	15	14	13	12	11	10	9	8
	DMAOS				ALOCK	PRESCALER[2:0]		
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY	PRESCSYNC[1:0]		MODE[1:0]		ENABLE	SWRST
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	W
Reset	0	0	0	0	0	0	0	0

Bits 28:27 – CAPTMODE1[1:0] Capture mode Channel 1
This bit field selects the channel 1 capture mode.

Value	Name	Description
0x0	DEFAULT	Default capture
0x1	CAPTMIN	Minimum capture
0x2	CAPTMAX	Maximum capture
0x3	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bits 25:24 – CAPTMODE0[1:0] Capture mode Channel 0
This bit field selects the channel 0 capture mode.

Value	Name	Description
0x0	DEFAULT	Default capture
0x1	CAPTMIN	Minimum capture
0x2	CAPTMAX	Maximum capture
0x3	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bits 20, 21 – COPENn Capture On Pin n Enable
Bit n of COPEN[1:0] selects the trigger source for capture the operation, choosing between events or I/O pin input.

Value	Description
0	An event from the Event System is selected as trigger source for capture operation on channel n

Value	Description
1	An I/O pin is selected as trigger source for the capture operation on channel n

Note: This bit is enable-protected. This bit is not synchronized.

Bits 16, 17 – CAPTENn Capture Channel n Enable
Bit n of CAPTEN[1:0] selects whether channel n operates as a capture or a compare channel.

Value	Description
0	CAPTEN disables capture on channel n
1	CAPTEN enables capture on channel n

Note: This bit is enable-protected. This bit is not synchronized.

Bit 15 – DMAOS DMA One-Shot Trigger Mode
Writing a '1' to this bit will generate the DMA trigger on a TC clock cycle following a TC_CTRLBSET_CMD_DMAOS command. Writing a '0' to this bit will generate DMA triggers on every TC clock cycle.

Value	Description
0	Generate a DMA trigger on the TC clock cycle following a TC_CTRLBSET_CMD_DMAOS command
1	Generate DMA triggers on every TC clock cycle

Bit 11 – ALOCK Auto Lock
When this bit is set, the Lock bit update (LUPD) is set to '1' on each overflow, underflow or retrigger event.

Value	Description
0	The LUPD bit is not affected on overflow, underflow, and retrigger event
1	The LUPD bit is set on each overflow, underflow or retrigger event

Note: This bit is enable-protected. This bit is not synchronized.

Bits 10:8 – PRESCALER[2:0] Prescaler
This bit field selects the counter prescaler factor.

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

Note: This bit field is enable-protected. This bit field is not synchronized.

Bit 7 – ONDEMAND Clock On Demand
This bit selects the clock requirements when the TC is stopped.
In Standby mode, if the Run in Standby bit (CTRLA.RUNSTDBY) is '0', ONDEMAND is forced to '0'.

Value	Description
0	On Demand is disabled. When On Demand is disabled, the TC will continue to request the clock even when its operation is stopped (STATUS.STOP=1).

Value	Description
1	On Demand is enabled. When On Demand is enabled, the stopped TC will not request the clock. The clock is requested only when a software retrigger command is issued or when an event with start or retrigger action is detected.

Note: This bit is enable-protected. This bit is not synchronized.

Bit 6 – RUNSTDBY Run in Standby

This bit is used to keep the TC running in Standby mode.

Value	Description
0	The TC is halted in Standby
1	The TC continues to run in Standby

Note: This bit is enable-protected. This bit is not synchronized.

Bits 5:4 – PRESCSYNC[1:0] Prescaler and Counter Synchronization

This bit field controls whether the counter wraps around on the next GCLK_TCn clock or the next prescaled GCLK_TCn clock. It also allows the prescaler to be reset.

Note: This bit field is enable-protected. This bit field is not synchronized.

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.
Other	—	Reserved

Bits 3:2 – MODE[1:0] Timer Counter Mode

This bit field controls the counter 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
Other	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bit 1 – ENABLE Enable

Notes:

1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure that synchronization of CTRLA.ENABLE iscomplete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset

Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets all registers in the TC, except DBGCTRL, to their initial state, and the TC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write operation will be discarded.

Notes:

1. When CTRLA.SWRST is written, the user must poll the SYNCBUSY.SWRST bit to determine when the reset operation is complete.
2. During a SWRST operation, access to registers or bits without SWRST are disallowed until SYNCBUSY.SWRST is cleared by hardware.
3. This bit is not enable-protected.

22.6.2. Control B Clear

Name: CTRLBCLR
Offset: 0x04
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to modify its contents without performing a read-modify-write operation. Any changes made to this register will also be reflected in the Control B Set register (CTRLBSET).
Note: This register is read- and write-synchronized: SYNCBUSY.CTRLB must be checked to ensure that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0] Command
Writing 0x0 to this bit field has no effect.
Writing a value other than 0x0 to this bit field will clear the pending command. Refer CMD bit field in CTRLBSET register for commands.

Bit 2 – ONESHOT One-Shot on Counter
This bit controls one-shot operation of the TC.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will disable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting when the next overflow or underflow condition occurs
1	The TC will wrap around and stop on the next underflow or overflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TC buffered registers.
When CTRLB.LUPD is set, the registers are not updated with the values from their buffered registers upon a hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before a hardware update is performed. After all the buffer registers are loaded correctly, the buffered registers can be unlocked.
This bit has no effect when input capture operation is enabled.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the LUPD bit.

Value	Description
0	The CCBUF[n] and PERBUF buffer registers values are copied into the CC[n] and PER registers upon a hardware update condition
1	The CCBUF[n] and PERBUF buffer registers values are not copied into the CC[n] and PER registers upon a hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the bit and cause the counter count up.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

22.6.3. Control B Set

Name: CTRLBSET
Offset: 0x05
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to change its content without performing a read-modify-write operation. Any changes made tp this register will also be reflected in the Control B Clear register (CTRLBCLR).
Note: This register is read- and write-synchronized: SYNCBUSY.CTRLB must be checked to ensure the that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0] Command
This bit field is used for software control of the TC. Commands are executed on the next prescaled GCLK_TC clock cycle. When a command has been executed, the CMD bit field will be read back as zero.
Writing a `0x0` to this bit field has no effect.
Writing any value other than `0x0`, as specified in the following table will issue a command for execution.

Value	Name	Description
0x0	NONE	No action
0x1	RETRIGGER	Force a start, restart or retrigger
0x2	STOP	Force a stop
0x3	UPDATE	Force update of double buffered registers
0x4	READSYNC	Force a read synchronization of COUNT
0x5	DMAOS	One-shot DMA trigger
Other	—	Reserved

Important: This command requires synchronization before being executed. A valid sequence is:

- Issue the CMD command (CTRLBSET.CMD = command)
- Wait for CMD synchronization (SYNCBUSY.CTRLB = 0)
- Wait for CMD bit field read back as zero (CTRLBSET.CMD = 0)

Bit 2 – ONESHOT One-Shot on Counter
This bit controls one-shot operation of the TC.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will enable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting when the next overflow or underflow condition
1	The TC will wrap around and stop on the next underflow or overflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TC buffered registers.

When CTRLB.LUPD is set, the registers are not updated with the values from their buffered registers upon a hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before a hardware update is performed. After all buffer registers are loaded correctly, the buffered registers can be unlocked.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the LUPD bit.
This bit has no effect when the input capture operation is enabled.

Value	Description
0	The CCBUF[n] and PERBUF buffer registers values are copied into the CC[n] and PER registers upon hardware update condition
1	The CCBUF[n] and PERBUF buffer registers values are not copied into the CC[n] and PER registers upon hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the bit and cause the counter count down.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

22.6.4. Event Control

Name: EVCTRL
Offset: 0x06
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

Bit	15	14	13	12	11	10	9	8
			MCEO1	MCEO0				OVFEO
Access			R/W	R/W				R/W
Reset			0	0				0

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

Bits 12, 13 – MCEOn Match or Capture Channel n Event Output Enable
This bit field enables the generation of an event for every match or capture on channel n.

Value	Description
0	Match/Capture event on channel n is disabled and will not be generated
1	Match/Capture event on channel n is enabled and will be generated for every compare or capture operation

Bit 8 – OVFEO Overflow/Underflow Event Output Enable
This bit enables the Overflow/Underflow event. When enabled, an event will be generated when the counter overflows/underflows.

Value	Description
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

Bit 5 – TCEI TC Event Input Enable
This bit is used to enable asynchronous input events to the TC.

Value	Description
0	Incoming events are disabled
1	Incoming events are enabled

Bit 4 – TCINV TC Inverted Event Input Polarity
This bit inverts the asynchronous input event source.

Value	Description
0	Input event source is not inverted
1	Input event source is inverted

Bits 2:0 – EVACT[2:0] Event Action
This bit field defines the event action the TC will perform on an event.

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	STAMP	Time stamp capture
0x5	PPW	Period captured in CC0, pulse width in CC1

Value	Name	Description
0x6	PWP	Period captured in CC1, pulse width in CC0
0x7	PW	Pulse width capture

22.6.5. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x08
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to MCn will clear the corresponding Match or Capture Channel n Interrupt Enable bit, which disables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bit 1 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, which disables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

22.6.6. Interrupt Enable Set

Name: INTENSET
Offset: 0x09
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to MCn will set the corresponding Match or Capture Channel n Interrupt Enable bit, which enables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bit 1 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Enable bit, which enables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

22.6.7. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x0A
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n
This flag is set on a comparison match, or when the corresponding CC[n] register contains a valid capture value. This flag will generate an interrupt request if the corresponding Match or Capture Channel n Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.MCn) is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the corresponding Match or Capture Channel n interrupt flag.
In capture operation, this flag is automatically cleared when CC[n] register is read.

Bit 1 – ERR Error Interrupt Flag
This flag is set when a new capture occurs on a channel while the corresponding Match or Capture Channel n interrupt flag is set, in which case there is nowhere to store the new capture.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Error Interrupt flag.

Bit 0 – OVF Overflow Interrupt Flag
This flag is set after an overflow condition occurs and will generate an interrupt request if INTENSET.OVF is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Overflow Interrupt flag.

22.6.8. Status

Name: STATUS
Offset: 0x0B
Reset: 0x01
Property: Read-Synchronized, Write-Synchronized

Note: This register is read- and write-synchronized: SYNCBUSY.STATUS must be checked to ensure the STATUS register synchronization is complete.

Bit	7	6	5	4	3	2	1	0
			CCBUFV1	CCBUFV0	PERBUFV		SLAVE	STOP
Access			R/W	R/W	R/W		R	R
Reset			0	0	0		0	1

Bits 4, 5 – CCBUFVn Channel n Compare or Capture Buffer Valid
For a compare channel n, the bit n is set when a new value is written to the corresponding CCBUF[n] register.
The bit n is cleared by writing a '1' to it when CTRLB.LUPD is set, or it is cleared automatically by the hardware on UPDATE condition.
For a capture channel n, the bit n is set when a valid capture value is stored in the CCBUF[n] register. The bit n is cleared automatically when the CC[n] register is read.

Bit 3 – PERBUFV Period Buffer Valid
This bit is set when a new value is written to the PERBUF register. The bit is cleared by writing '1' to the corresponding location when CTRLB.LUPD is set, or automatically cleared by the hardware on UPDATE condition.

Bit 1 – SLAVE Client Status Flag
This bit is only available in 32-bit mode on the client (odd-numbered TCn). The bit is set when the associated host (even-numbered TCn) is set to run in 32-bit mode.

Bit 0 – STOP Stop Status Flag
This bit is set when the TC is disabled, on a Stop command, or on an overflow/underflow condition when the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) is '1'.

Value	Description
0	The counter is running
1	The counter is stopped

22.6.9. Waveform Generation Control

Name: WAVE
Offset: 0x0C
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

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

Bits 1:0 – WAVEGEN[1:0] Waveform Generation Mode
This bit field controls the waveform generation operation. It affects the top value, as shown in the *Waveform Output Operations* section. It also controls whether frequency or PWM waveform generation will be used. The waveform generation operations are explained in the *Waveform Output Operations* section.

Value	Name	Description
0x0	NFRQ	Normal Frequency Generation mode
0x1	MFRQ	Match Frequency Generation mode
0x2	NPWM	Normal Pulse-Width Modulation mode
0x3	MPWM	Match Pulse-Width Modulation mode

22.6.10. Driver Control

Name: DRVCTRL
Offset: 0x0D
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

Bit	7	6	5	4	3	2	1	0
							INVEN1	INVEN0
Access							R/W	R/W
Reset							0	0

Bits 0, 1 – INVENn Waveform Output n Invert Enable
The INVENn bit selects inversion of the output or capture trigger input of channel n.

Value	Description
0	Disable inversion of the WO[n] output and I/O input pin
1	Enable inversion of the WO[n] output and I/O input pin

22.6.11. Debug Control

Name: DBGCTRL
Offset: 0x0F
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Run in Debug Mode
This bit is not affected by a software Reset and must not be changed by the software while the TC is enabled.

Value	Description
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

22.6.12. Synchronization Busy

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	CC1	CC0	PER	COUNT	STATUS	CTRLB	ENABLE	SWRST
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 6, 7 – CCn Compare/Capture Channel n Synchronization Busy
For details on CC channels number, refer to each TC feature list.
This bit is cleared when the synchronization of CC[n] between the clock domains is complete.
This bit is set when the synchronization of CC[n] between clock domains is started.
This bit is also set when the CCBUF[n] is written, and cleared on update condition. The bit is automatically cleared when the STATUS.CCBUF[n] bit is cleared.

Bit 5 – PER PER Synchronization Busy
This bit is cleared when the synchronization of PER between the clock domains is complete.
This bit is set when the synchronization of PER between clock domains is started.
This bit is also set when the PERBUF is written, and cleared on update condition. The bit is automatically cleared when the STATUS.PERBUF bit is cleared.

Bit 4 – COUNT COUNT Synchronization Busy
This bit is cleared when the synchronization of COUNT between the clock domains is complete.
This bit is set when the synchronization of COUNT between clock domains is started.

Bit 3 – STATUS STATUS Synchronization Busy
This bit is cleared when the synchronization of STATUS between the clock domains is complete.
This bit is set when a '1' is written to the Capture Channel Buffer Valid status flags (STATUS.CCBUFVn) and the synchronization of STATUS between clock domains is started.

Bit 2 – CTRLB CTRLB Synchronization Busy
This bit is cleared when the synchronization of CTRLBSET or CTRLBCLR between the clock domains is complete.
This bit is set when the synchronization of CTRLBSET or CTRLBCLR between clock domains is started.

Bit 1 – ENABLE ENABLE Synchronization Busy
This bit is cleared when the synchronization of ENABLE bit between the clock domains is complete.
This bit is set when the synchronization of ENABLE bit between clock domains is started.

Bit 0 – SWRST SWRST Synchronization Busy
This bit is cleared when the synchronization of SWRST bit between the clock domains is complete.
This bit is set when the synchronization of SWRST bit between clock domains is started.

22.6.13. Counter - COUNT8

Name: COUNT
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. Prior to any read access, this register must be manually read synchronized by writing to the Command bit in the Control B Set register (CTRLBSET.CMD = READSYNC) and wait for the manual read synchronization to finish (SYNCBUSY.CTRLB and CTRLBSET.CMD are '0').
2. This register is write-synchronized. The COUNT Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.COUNT) must be checked to ensure that the COUNT register write synchronization is complete.

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
This bit field contains the current counter value.

22.6.14. Period - COUNT8

Name: PER
Offset: 0x1B
Reset: 0xFF
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.PER must be checked to ensure the PER register synchronization is complete.

Important: PER register updates using the DMA are not possible in Standby mode.

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
This bit field holds the value of the TC period count.

22.6.15. Channel n Compare/Capture - COUNT8

Name: CC[n]
Offset: 0x1C + n*0x01 [n=0..1]
Reset: 0x00
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.CC[n] must be checked to ensure the CC[n] register synchronization is complete.

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] Channel Compare/Capture Value
This bit field contains the compare/capture value in 8-bit TC mode. In Match frequency (MFRQ) or Match PWM (MPWM) waveform operation (WAVE.WAVEGEN), the CC0 register is used as a period register.

22.6.16. Period Buffer - COUNT8

Name: PERBUF
Offset: 0x2F
Reset: 0xFF
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.PER must be checked to ensure the PER register synchronization is complete.

Bit	7	6	5	4	3	2	1	0
	PERBUF[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 – PERBUF[7:0] Period Buffer Value
This bit field holds the value of the period buffer register. The value is copied to PER register on UPDATE condition.

22.6.17. Channel n Compare Buffer - COUNT8

Name: CCBUF[n]
Offset: 0x30 + n*0x01 [n=0..1]
Reset: 0x00
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.CC[n] must be checked to ensure the CC[n] register synchronization is complete.

Bit	7	6	5	4	3	2	1	0
	CCBUF[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 – CCBUF[7:0] Channel Compare Buffer Value
This bit field holds the value of the Channel n Compare Buffer Value. When the buffer valid flag is '1' and double buffering is enabled (CTRLBCLR.LUPD = 1), the data from buffer registers will be copied into the corresponding CC[n] register under UPDATE condition, including the software update command (CTRLBSET.CMD=0x3).

22.7. Register Summary - TCn.COUNT16

Timer/Counter in 16-bit mode (COUNT16)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	ONDEMAND	RUNSTDBY	PRESCSYNC[1:0]		MODE[1:0]		ENABLE	SWRST
		15:8	DMAOS				ALOCK	PRESCALER[2:0]		
		23:16			COPEN1	COPEN0			CAPTEN1	CAPTEN0
		31:24				CAPTMODE1[1:0]			CAPTMODE0[1:0]	
0x04	CTRLBCLR	7:0	CMD[2:0]					ONESHOT	LUPD	DIR
0x05	CTRLBSET	7:0	CMD[2:0]					ONESHOT	LUPD	DIR
0x06	EVCTRL	7:0			TCEI	TCINV		EVACT[2:0]		
		15:8			MCEO1	MCEO0				OVFEO
0x08	INTENCLR	7:0			MC1	MC0			ERR	OVF
0x09	INTENSET	7:0			MC1	MC0			ERR	OVF
0x0A	INTFLAG	7:0			MC1	MC0			ERR	OVF
0x0B	STATUS	7:0			CCBUFV1	CCBUFV0	PERBUFV		SLAVE	STOP
0x0C	WAVE	7:0							WAVEGEN[1:0]	
0x0D	DRVCTRL	7:0							INVEN1	INVEN0
0x0E	Reserved									
0x0F	DBGCTRL	7:0								DBGRUN
0x10	SYNCBUSY	7:0	CC1	CC0	PER	COUNT	STATUS	CTRLB	ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x14	COUNT	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
0x16 ... 0x1B	Reserved									
0x1C	CC[0]	7:0	CC[7:0]							
		15:8	CC[15:8]							
0x1E	CC[1]	7:0	CC[7:0]							
		15:8	CC[15:8]							
0x20 ... 0x2F	Reserved									
0x30	CCBUF[0]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
0x32	CCBUF[1]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							

22.7.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized, Enable-Protected

Bit	31	30	29	28	27	26	25	24
				CAPTMODE1[1:0]			CAPTMODE0[1:0]	
Access				R/W	R/W		R/W	R/W
Reset				0	0		0	0

Bit	23	22	21	20	19	18	17	16
			COPEN1	COPEN0			CAPTEN1	CAPTEN0
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bit	15	14	13	12	11	10	9	8
	DMAOS				ALOCK	PRESCALER[2:0]		
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY	PRESCSYNC[1:0]		MODE[1:0]		ENABLE	SWRST
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	W
Reset	0	0	0	0	0	0	0	0

Bits 28:27 – CAPTMODE1[1:0] Capture mode Channel 1
This bit field selects the channel 1 capture mode.

Value	Name	Description
0x0	DEFAULT	Default capture
0x1	CAPTMIN	Minimum capture
0x2	CAPTMAX	Maximum capture
0x3	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bits 25:24 – CAPTMODE0[1:0] Capture mode Channel 0
This bit field selects the channel 0 capture mode.

Value	Name	Description
0x0	DEFAULT	Default capture
0x1	CAPTMIN	Minimum capture
0x2	CAPTMAX	Maximum capture
0x3	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bits 20, 21 – COPENn Capture On Pin n Enable
Bit n of COPEN[1:0] selects the trigger source for capture the operation, choosing between events or I/O pin input.

Value	Description
0	An event from the Event System is selected as trigger source for capture operation on channel n

Value	Description
1	An I/O pin is selected as trigger source for the capture operation on channel n

Note: This bit is enable-protected. This bit is not synchronized.

Bits 16, 17 – CAPTENn Capture Channel n Enable
Bit n of CAPTEN[1:0] selects whether channel n operates as a capture or a compare channel.

Value	Description
0	CAPTEN disables capture on channel n
1	CAPTEN enables capture on channel n

Note: This bit is enable-protected. This bit is not synchronized.

Bit 15 – DMAOS DMA One-Shot Trigger Mode
Writing a '1' to this bit will generate the DMA trigger on a TC clock cycle following a TC_CTRLBSET_CMD_DMAOS command. Writing a '0' to this bit will generate DMA triggers on every TC clock cycle.

Value	Description
0	Generate a DMA trigger on the TC clock cycle following a TC_CTRLBSET_CMD_DMAOS command
1	Generate DMA triggers on every TC clock cycle

Bit 11 – ALOCK Auto Lock
When this bit is set, the Lock bit update (LUPD) is set to '1' on each overflow, underflow or retrigger event.

Value	Description
0	The LUPD bit is not affected on overflow, underflow, and retrigger event
1	The LUPD bit is set on each overflow, underflow or retrigger event

Note: This bit is enable-protected. This bit is not synchronized.

Bits 10:8 – PRESCALER[2:0] Prescaler
This bit field selects the counter prescaler factor.

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

Note: This bit field is enable-protected. This bit field is not synchronized.

Bit 7 – ONDEMAND Clock On Demand
This bit selects the clock requirements when the TC is stopped.
In Standby mode, if the Run in Standby bit (CTRLA.RUNSTDBY) is '0', ONDEMAND is forced to '0'.

Value	Description
0	On Demand is disabled. When On Demand is disabled, the TC will continue to request the clock even when its operation is stopped (STATUS.STOP=1).

Value	Description
1	On Demand is enabled. When On Demand is enabled, the stopped TC will not request the clock. The clock is requested only when a software retrigger command is issued or when an event with start or retrigger action is detected.

Note: This bit is enable-protected. This bit is not synchronized.

Bit 6 – RUNSTDBY Run in Standby

This bit is used to keep the TC running in Standby mode.

Value	Description
0	The TC is halted in Standby
1	The TC continues to run in Standby

Note: This bit is enable-protected. This bit is not synchronized.

Bits 5:4 – PRESCSYNC[1:0] Prescaler and Counter Synchronization

This bit field controls whether the counter wraps around on the next GCLK_TCn clock or the next prescaled GCLK_TCn clock. It also allows the prescaler to be reset.
Note: This bit field is enable-protected. This bit field is not synchronized.

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.
Other	—	Reserved

Bits 3:2 – MODE[1:0] Timer Counter Mode

This bit field controls the counter 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
Other	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bit 1 – ENABLE Enable

Notes:

1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure that synchronization of CTRLA.ENABLE iscomplete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset

Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the TC, except DBGCTRL, to their initial state, and the TC will be disabled.
Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write operation will be discarded.

Notes:

1. When CTRLA.SWRST is written, the user must poll the SYNCBUSY.SWRST bit to determine when the reset operation is complete.
2. During a SWRST operation, access to registers or bits without SWRST are disallowed until SYNCBUSY.SWRST is cleared by hardware.
3. This bit is not enable-protected.

22.7.2. Control B Clear

Name: CTRLBCLR
Offset: 0x04
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to modify its contents without performing a read-modify-write operation. Any changes made to this register will also be reflected in the Control B Set register (CTRLBSET).
Note: This register is read- and write-synchronized: SYNCBUSY.CTRLB must be checked to ensure that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0] Command
Writing 0x0 to this bit field has no effect.
Writing a value other than 0x0 to this bit field will clear the pending command. Refer CMD bit field in CTRLBSET register for commands.

Bit 2 – ONESHOT One-Shot on Counter
This bit controls one-shot operation of the TC.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will disable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting when the next overflow or underflow condition occurs
1	The TC will wrap around and stop on the next underflow or overflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TC buffered registers.
When CTRLB.LUPD is set, the registers are not updated with the values from their buffered registers upon a hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before a hardware update is performed. After all the buffer registers are loaded correctly, the buffered registers can be unlocked.
This bit has no effect when input capture operation is enabled.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the LUPD bit.

Value	Description
0	The CCBUF[n] and PERBUF buffer registers values are copied into the CC[n] and PER registers upon a hardware update condition
1	The CCBUF[n] and PERBUF buffer registers values are not copied into the CC[n] and PER registers upon a hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the bit and cause the counter count up.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

22.7.3. Control B Set

Name: CTRLBSET
Offset: 0x05
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to change its content without performing a read-modify-write operation. Any changes made tp this register will also be reflected in the Control B Clear register (CTRLBCLR).
Note: This register is read- and write-synchronized: SYNCBUSY.CTRLB must be checked to ensure the that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0] Command
This bit field is used for software control of the TC. Commands are executed on the next prescaled GCLK_TC clock cycle. When a command has been executed, the CMD bit field will be read back as zero.
Writing a `0x0` to this bit field has no effect.
Writing any value other than `0x0`, as specified in the following table will issue a command for execution.

Value	Name	Description
0x0	NONE	No action
0x1	RETRIGGER	Force a start, restart or retrigger
0x2	STOP	Force a stop
0x3	UPDATE	Force update of double buffered registers
0x4	READSYNC	Force a read synchronization of COUNT
0x5	DMAOS	One-shot DMA trigger
Other	—	Reserved

Important: This command requires synchronization before being executed. A valid sequence is:

- Issue the CMD command (CTRLBSET.CMD = command)
- Wait for CMD synchronization (SYNCBUSY.CTRLB = 0)
- Wait for CMD bit field read back as zero (CTRLBSET.CMD = 0)

Bit 2 – ONESHOT One-Shot on Counter
This bit controls one-shot operation of the TC.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will enable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting when the next overflow or underflow condition
1	The TC will wrap around and stop on the next underflow or overflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TC buffered registers.

When CTRLB.LUPD is set, the registers are not updated with the values from their buffered registers upon a hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before a hardware update is performed. After all buffer registers are loaded correctly, the buffered registers can be unlocked.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the LUPD bit.
This bit has no effect when the input capture operation is enabled.

Value	Description
0	The CCBUF[n] and PERBUF buffer registers values are copied into the CC[n] and PER registers upon hardware update condition
1	The CCBUF[n] and PERBUF buffer registers values are not copied into the CC[n] and PER registers upon hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the bit and cause the counter count down.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

22.7.4. Event Control

Name: EVCTRL
Offset: 0x06
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

Bit	15	14	13	12	11	10	9	8
			MCEO1	MCEO0				OVFEO
Access			R/W	R/W				R/W
Reset			0	0				0

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

Bits 12, 13 – MCEOn Match or Capture Channel n Event Output Enable
This bit field enables the generation of an event for every match or capture on channel n.

Value	Description
0	Match/Capture event on channel n is disabled and will not be generated
1	Match/Capture event on channel n is enabled and will be generated for every compare or capture operation

Bit 8 – OVFEO Overflow/Underflow Event Output Enable
This bit enables the Overflow/Underflow event. When enabled, an event will be generated when the counter overflows/underflows.

Value	Description
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

Bit 5 – TCEI TC Event Input Enable
This bit is used to enable asynchronous input events to the TC.

Value	Description
0	Incoming events are disabled
1	Incoming events are enabled

Bit 4 – TCINV TC Inverted Event Input Polarity
This bit inverts the asynchronous input event source.

Value	Description
0	Input event source is not inverted
1	Input event source is inverted

Bits 2:0 – EVACT[2:0] Event Action
This bit field defines the event action the TC will perform on an event.

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	STAMP	Time stamp capture
0x5	PPW	Period captured in CC0, pulse width in CC1

Value	Name	Description
0x6	PWP	Period captured in CC1, pulse width in CC0
0x7	PW	Pulse width capture

22.7.5. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x08
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to MCn will clear the corresponding Match or Capture Channel n Interrupt Enable bit, which disables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bit 1 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, which disables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

22.7.6. Interrupt Enable Set

Name: INTENSET
Offset: 0x09
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to MCn will set the corresponding Match or Capture Channel n Interrupt Enable bit, which enables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bit 1 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Enable bit, which enables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

22.7.7. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x0A
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n
This flag is set on a comparison match, or when the corresponding CC[n] register contains a valid capture value. This flag will generate an interrupt request if the corresponding Match or Capture Channel n Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.MCn) is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the corresponding Match or Capture Channel n interrupt flag.
In capture operation, this flag is automatically cleared when CC[n] register is read.

Bit 1 – ERR Error Interrupt Flag
This flag is set when a new capture occurs on a channel while the corresponding Match or Capture Channel n interrupt flag is set, in which case there is nowhere to store the new capture.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Error Interrupt flag.

Bit 0 – OVF Overflow Interrupt Flag
This flag is set after an overflow condition occurs and will generate an interrupt request if INTENSET.OVF is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Overflow Interrupt flag.

22.7.8. Status

Name: STATUS
Offset: 0x0B
Reset: 0x01
Property: Read-Synchronized, Write-Synchronized

Note: This register is read- and write-synchronized: SYNCBUSY.STATUS must be checked to ensure the STATUS register synchronization is complete.

Bit	7	6	5	4	3	2	1	0
			CCBUFV1	CCBUFV0	PERBUFV		SLAVE	STOP
Access			R/W	R/W	R/W		R	R
Reset			0	0	0		0	1

Bits 4, 5 – CCBUFVn Channel n Compare or Capture Buffer Valid
For a compare channel n, the bit n is set when a new value is written to the corresponding CCBUF[n] register.
The bit n is cleared by writing a '1' to it when CTRLB.LUPD is set, or it is cleared automatically by the hardware on UPDATE condition.
For a capture channel n, the bit n is set when a valid capture value is stored in the CCBUF[n] register.
The bit n is cleared automatically when the CC[n] register is read.

Bit 3 – PERBUFV Period Buffer Valid
This bit is set when a new value is written to the PERBUF register. The bit is cleared by writing '1' to the corresponding location when CTRLB.LUPD is set, or automatically cleared by the hardware on UPDATE condition.

Bit 1 – SLAVE Client Status Flag
This bit is only available in 32-bit mode on the client (odd-numbered TCn). The bit is set when the associated host (even-numbered TCn) is set to run in 32-bit mode.

Bit 0 – STOP Stop Status Flag
This bit is set when the TC is disabled, on a Stop command, or on an overflow/underflow condition when the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) is '1'.

Value	Description
0	The counter is running
1	The counter is stopped

22.7.9. Waveform Generation Control

Name: WAVE
Offset: 0x0C
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

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

Bits 1:0 – WAVEGEN[1:0] Waveform Generation Mode
This bit field controls the waveform generation operation. It affects the top value, as shown in the *Waveform Output Operations* section. It also controls whether frequency or PWM waveform generation will be used. The waveform generation operations are explained in the *Waveform Output Operations* section.

Value	Name	Description
0x0	NFRQ	Normal Frequency Generation mode
0x1	MFRQ	Match Frequency Generation mode
0x2	NPWM	Normal Pulse-Width Modulation mode
0x3	MPWM	Match Pulse-Width Modulation mode

22.7.10. Driver Control

Name: DRVCTRL
Offset: 0x0D
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

Bit	7	6	5	4	3	2	1	0
							INVEN1	INVEN0
Access							R/W	R/W
Reset							0	0

Bits 0, 1 – INVENn Waveform Output n Invert Enable
The INVENn bit selects inversion of the output or capture trigger input of channel n.

Value	Description
0	Disable inversion of the WO[n] output and I/O input pin
1	Enable inversion of the WO[n] output and I/O input pin

22.7.11. Debug Control

Name: DBGCTRL
Offset: 0x0F
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Run in Debug Mode
This bit is not affected by a software Reset and must not be changed by the software while the TC is enabled.

Value	Description
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

22.7.12. Synchronization Busy

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	CC1	CC0	PER	COUNT	STATUS	CTRLB	ENABLE	SWRST
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 6, 7 – CCn Compare/Capture Channel n Synchronization Busy
For details on CC channels number, refer to each TC feature list.
This bit is cleared when the synchronization of CC[n] between the clock domains is complete.
This bit is set when the synchronization of CC[n] between clock domains is started.
This bit is also set when the CCBUF[n] is written, and cleared on update condition. The bit is automatically cleared when the STATUS.CCBUF[n] bit is cleared.

Bit 5 – PER PER Synchronization Busy
This bit is cleared when the synchronization of PER between the clock domains is complete.
This bit is set when the synchronization of PER between clock domains is started.
This bit is also set when the PERBUF is written, and cleared on update condition. The bit is automatically cleared when the STATUS.PERBUF bit is cleared.

Bit 4 – COUNT COUNT Synchronization Busy
This bit is cleared when the synchronization of COUNT between the clock domains is complete.
This bit is set when the synchronization of COUNT between clock domains is started.

Bit 3 – STATUS STATUS Synchronization Busy
This bit is cleared when the synchronization of STATUS between the clock domains is complete.
This bit is set when a '1' is written to the Capture Channel Buffer Valid status flags (STATUS.CCBUFVn) and the synchronization of STATUS between clock domains is started.

Bit 2 – CTRLB CTRLB Synchronization Busy
This bit is cleared when the synchronization of CTRLBSET or CTRLBCLR between the clock domains is complete.
This bit is set when the synchronization of CTRLBSET or CTRLBCLR between clock domains is started.

Bit 1 – ENABLE ENABLE Synchronization Busy
This bit is cleared when the synchronization of ENABLE bit between the clock domains is complete.
This bit is set when the synchronization of ENABLE bit between clock domains is started.

Bit 0 – SWRST SWRST Synchronization Busy
This bit is cleared when the synchronization of SWRST bit between the clock domains is complete.
This bit is set when the synchronization of SWRST bit between clock domains is started.

22.7.13. Counter - COUNT16

Name: COUNT
Offset: 0x14
Reset: 0x0000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. Prior to any read access, this register must be manually read synchronized by writing to the Command bit in the Control B Set register (CTRLBSET.CMD = READSYNC) and wait for the manual read synchronization to finish (SYNCBUSY.CTRLB and CTRLBSET.CMD are '0').
2. This register is write-synchronized. The COUNT Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.COUNT) must be checked to ensure that the COUNT register write synchronization is complete.

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
This bit field contains the current counter value.

22.7.14. Channel n Compare/Capture - COUNT16

Name: CC[n]
Offset: 0x1C + n*0x02 [n=0..1]
Reset: 0x0000
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.CC[n] must be checked to ensure the CC[n] register synchronization is complete.

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] Channel Compare/Capture Value
This bit field contains the compare/capture value in 16-bit TC mode. In Match frequency (MFRQ) or Match PWM (MPWM) waveform operation (WAVE.WAVEGEN), the CC0 register is used as a period register.

22.7.15. Channel n Compare Buffer - COUNT16

Name: CCBUF[n]
Offset: 0x30 + n*0x02 [n=0..1]
Reset: 0x0000
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.CC[n] must be checked to ensure the CCBUF[n] register synchronization is complete.

Bit	15	14	13	12	11	10	9	8
	CCBUF[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
	CCBUF[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 – CCBUF[15:0] Channel Compare Buffer Value
This bit field holds the value of the Channel n Compare Buffer Value. When the buffer valid flag is '1' and double buffering is enabled (CTRLBCLR.LUPD=1), the data from buffer registers will be copied into the corresponding CC[n] register under UPDATE condition, including the software update command (CTRLBSET.CMD=0x3).

22.8. Register Summary - TCn.COUNT32

Timer/Counter in 32-bit mode (COUNT32)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	ONDEMAND	RUNSTDBY	PRESCSYNC[1:0]		MODE[1:0]		ENABLE	SWRST
		15:8	DMAOS				ALOCK	PRESCALER[2:0]		
		23:16			COPEN1	COPEN0			CAPTEN1	CAPTEN0
		31:24				CAPTMODE1[1:0]			CAPTMODE0[1:0]	
0x04	CTRLBCLR	7:0	CMD[2:0]					ONESHOT	LUPD	DIR
0x05	CTRLBSET	7:0	CMD[2:0]					ONESHOT	LUPD	DIR
0x06	EVCTRL	7:0			TCEI	TCINV		EVACT[2:0]		
		15:8			MCEO1	MCEO0				OVFEO
0x08	INTENCLR	7:0			MC1	MC0			ERR	OVF
0x09	INTENSET	7:0			MC1	MC0			ERR	OVF
0x0A	INTFLAG	7:0			MC1	MC0			ERR	OVF
0x0B	STATUS	7:0			CCBUFV1	CCBUFV0	PERBUFV		SLAVE	STOP
0x0C	WAVE	7:0							WAVEGEN[1:0]	
0x0D	DRVCTRL	7:0							INVEN1	INVEN0
0x0E	Reserved									
0x0F	DBGCTRL	7:0								DBGRUN
0x10	SYNCBUSY	7:0	CC1	CC0	PER	COUNT	STATUS	CTRLB	ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x14	COUNT	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16	COUNT[23:16]							
		31:24	COUNT[31:24]							
0x18 ... 0x1B	Reserved									
0x1C	CC[0]	7:0	CC[7:0]							
		15:8	CC[15:8]							
		23:16	CC[23:16]							
		31:24	CC[31:24]							
0x20	CC[1]	7:0	CC[7:0]							
		15:8	CC[15:8]							
		23:16	CC[23:16]							
		31:24	CC[31:24]							
0x24 ... 0x2F	Reserved									
0x30	CCBUF[0]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
		23:16	CCBUF[23:16]							
		31:24	CCBUF[31:24]							
0x34	CCBUF[1]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
		23:16	CCBUF[23:16]							
		31:24	CCBUF[31:24]							

22.8.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized, Enable-Protected

Bit	31	30	29	28	27	26	25	24
				CAPTMODE1[1:0]			CAPTMODE0[1:0]	
Access				R/W	R/W		R/W	R/W
Reset				0	0		0	0

Bit	23	22	21	20	19	18	17	16
			COPEN1	COPEN0			CAPTEN1	CAPTEN0
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bit	15	14	13	12	11	10	9	8
	DMAOS				ALOCK	PRESCALER[2:0]		
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ONDEMAND	RUNSTDBY	PRESCSYNC[1:0]		MODE[1:0]		ENABLE	SWRST
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	W
Reset	0	0	0	0	0	0	0	0

Bits 28:27 – CAPTMODE1[1:0] Capture mode Channel 1
This bit field selects the channel 1 capture mode.

Value	Name	Description
0x0	DEFAULT	Default capture
0x1	CAPTMIN	Minimum capture
0x2	CAPTMAX	Maximum capture
0x3	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bits 25:24 – CAPTMODE0[1:0] Capture mode Channel 0
This bit field selects the channel 0 capture mode.

Value	Name	Description
0x0	DEFAULT	Default capture
0x1	CAPTMIN	Minimum capture
0x2	CAPTMAX	Maximum capture
0x3	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bits 20, 21 – COPENn Capture On Pin n Enable
Bit n of COPEN[1:0] selects the trigger source for capture the operation, choosing between events or I/O pin input.

Value	Description
0	An event from the Event System is selected as trigger source for capture operation on channel n

Value	Description
1	An I/O pin is selected as trigger source for the capture operation on channel n

Note: This bit is enable-protected. This bit is not synchronized.

Bits 16, 17 – CAPTENn Capture Channel n Enable
Bit n of CAPTEN[1:0] selects whether channel n operates as a capture or a compare channel.

Value	Description
0	CAPTEN disables capture on channel n
1	CAPTEN enables capture on channel n

Note: This bit is enable-protected. This bit is not synchronized.

Bit 15 – DMAOS DMA One-Shot Trigger Mode
Writing a '1' to this bit will generate the DMA trigger on a TC clock cycle following a TC_CTRLBSET_CMD_DMAOS command. Writing a '0' to this bit will generate DMA triggers on every TC clock cycle.

Value	Description
0	Generate a DMA trigger on the TC clock cycle following a TC_CTRLBSET_CMD_DMAOS command
1	Generate DMA triggers on every TC clock cycle

Bit 11 – ALOCK Auto Lock
When this bit is set, the Lock bit update (LUPD) is set to '1' on each overflow, underflow or retrigger event.

Value	Description
0	The LUPD bit is not affected on overflow, underflow, and retrigger event
1	The LUPD bit is set on each overflow, underflow or retrigger event

Note: This bit is enable-protected. This bit is not synchronized.

Bits 10:8 – PRESCALER[2:0] Prescaler
This bit field selects the counter prescaler factor.

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

Note: This bit field is enable-protected. This bit field is not synchronized.

Bit 7 – ONDEMAND Clock On Demand
This bit selects the clock requirements when the TC is stopped.
In Standby mode, if the Run in Standby bit (CTRLA.RUNSTDBY) is '0', ONDEMAND is forced to '0'.

Value	Description
0	On Demand is disabled. When On Demand is disabled, the TC will continue to request the clock even when its operation is stopped (STATUS.STOP=1).

Value	Description
1	On Demand is enabled. When On Demand is enabled, the stopped TC will not request the clock. The clock is requested only when a software retrigger command is issued or when an event with start or retrigger action is detected.

Note: This bit is enable-protected. This bit is not synchronized.

Bit 6 – RUNSTDBY Run in Standby

This bit is used to keep the TC running in Standby mode.

Value	Description
0	The TC is halted in Standby
1	The TC continues to run in Standby

Note: This bit is enable-protected. This bit is not synchronized.

Bits 5:4 – PRESCSYNC[1:0] Prescaler and Counter Synchronization

This bit field controls whether the counter wraps around on the next GCLK_TCn clock or the next prescaled GCLK_TCn clock. It also allows the prescaler to be reset.
Note: This bit field is enable-protected. This bit field is not synchronized.

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.
Other	—	Reserved

Bits 3:2 – MODE[1:0] Timer Counter Mode

This bit field controls the counter 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
Other	—	Reserved

Note: This bit field is enable-protected. This bit field is not synchronized.

Bit 1 – ENABLE Enable

Notes:

1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure that synchronization of CTRLA.ENABLE iscomplete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset

Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the TC, except DBGCTRL, to their initial state, and the TC will be disabled.
Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write operation will be discarded.

Notes:

1. When CTRLA.SWRST is written, the user must poll the SYNCBUSY.SWRST bit to determine when the reset operation is complete.
2. During a SWRST operation, access to registers or bits without SWRST are disallowed until SYNCBUSY.SWRST is cleared by hardware.
3. This bit is not enable-protected.

22.8.2. Control B Clear

Name: CTRLBCLR
Offset: 0x04
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to modify its contents without performing a read-modify-write operation. Any changes made to this register will also be reflected in the Control B Set register (CTRLBSET).
Note: This register is read- and write-synchronized: SYNCBUSY.CTRLB must be checked to ensure that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0] Command
Writing 0x0 to this bit field has no effect.
Writing a value other than 0x0 to this bit field will clear the pending command. Refer CMD bit field in CTRLBSET register for commands.

Bit 2 – ONESHOT One-Shot on Counter
This bit controls one-shot operation of the TC.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will disable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting when the next overflow or underflow condition occurs
1	The TC will wrap around and stop on the next underflow or overflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TC buffered registers.
When CTRLB.LUPD is set, the registers are not updated with the values from their buffered registers upon a hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before a hardware update is performed. After all the buffer registers are loaded correctly, the buffered registers can be unlocked.
This bit has no effect when input capture operation is enabled.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the LUPD bit.

Value	Description
0	The CCBUF[n] and PERBUF buffer registers values are copied into the CC[n] and PER registers upon a hardware update condition
1	The CCBUF[n] and PERBUF buffer registers values are not copied into the CC[n] and PER registers upon a hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the bit and cause the counter count up.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

22.8.3. Control B Set

Name: CTRLBSET
Offset: 0x05
Reset: 0x00
Property: PAC Write-Protection, Read-Synchronized, Write-Synchronized

This register allows the user to change its content without performing a read-modify-write operation. Any changes made tp this register will also be reflected in the Control B Clear register (CTRLBCLR).
Note: This register is read- and write-synchronized: SYNCBUSY.CTRLB must be checked to ensure the that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]					ONESHOT	LUPD	DIR
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bits 7:5 – CMD[2:0] Command
This bit field is used for software control of the TC. Commands are executed on the next prescaled GCLK_TC clock cycle. When a command has been executed, the CMD bit field will be read back as zero.
Writing a `0x0` to this bit field has no effect.
Writing any value other than `0x0`, as specified in the following table will issue a command for execution.

Value	Name	Description
0x0	NONE	No action
0x1	RETRIGGER	Force a start, restart or retrigger
0x2	STOP	Force a stop
0x3	UPDATE	Force update of double buffered registers
0x4	READSYNC	Force a read synchronization of COUNT
0x5	DMAOS	One-shot DMA trigger
Other	—	Reserved

Important: This command requires synchronization before being executed. A valid sequence is:

- Issue the CMD command (CTRLBSET.CMD = command)
- Wait for CMD synchronization (SYNCBUSY.CTRLB = 0)
- Wait for CMD bit field read back as zero (CTRLBSET.CMD = 0)

Bit 2 – ONESHOT One-Shot on Counter
This bit controls one-shot operation of the TC.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will enable one-shot operation.

Value	Description
0	The TC will wrap around and continue counting when the next overflow or underflow condition
1	The TC will wrap around and stop on the next underflow or overflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TC buffered registers.

When CTRLB.LUPD is set, the registers are not updated with the values from their buffered registers upon a hardware UPDATE condition. Locking the update ensures that all buffer registers are valid before a hardware update is performed. After all buffer registers are loaded correctly, the buffered registers can be unlocked.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the LUPD bit.
This bit has no effect when the input capture operation is enabled.

Value	Description
0	The CCBUF[n] and PERBUF buffer registers values are copied into the CC[n] and PER registers upon hardware update condition
1	The CCBUF[n] and PERBUF buffer registers values are not copied into the CC[n] and PER registers upon hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the bit and cause the counter count down.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

22.8.4. Event Control

Name: EVCTRL
Offset: 0x06
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

Bit	15	14	13	12	11	10	9	8
			MCEO1	MCEO0				OVFEO
Access			R/W	R/W				R/W
Reset			0	0				0

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

Bits 12, 13 – MCEOn Match or Capture Channel n Event Output Enable
This bit field enables the generation of an event for every match or capture on channel n.

Value	Description
0	Match/Capture event on channel n is disabled and will not be generated
1	Match/Capture event on channel n is enabled and will be generated for every compare or capture operation

Bit 8 – OVFEO Overflow/Underflow Event Output Enable
This bit enables the Overflow/Underflow event. When enabled, an event will be generated when the counter overflows/underflows.

Value	Description
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

Bit 5 – TCEI TC Event Input Enable
This bit is used to enable asynchronous input events to the TC.

Value	Description
0	Incoming events are disabled
1	Incoming events are enabled

Bit 4 – TCINV TC Inverted Event Input Polarity
This bit inverts the asynchronous input event source.

Value	Description
0	Input event source is not inverted
1	Input event source is inverted

Bits 2:0 – EVACT[2:0] Event Action
This bit field defines the event action the TC will perform on an event.

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	STAMP	Time stamp capture
0x5	PPW	Period captured in CC0, pulse width in CC1

Value	Name	Description
0x6	PWP	Period captured in CC1, pulse width in CC0
0x7	PW	Pulse width capture

22.8.5. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x08
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to MCn will clear the corresponding Match or Capture Channel n Interrupt Enable bit, which disables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bit 1 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Enable bit, which disables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

22.8.6. Interrupt Enable Set

Name: INTENSET
Offset: 0x09
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to MCn will set the corresponding Match or Capture Channel n Interrupt Enable bit, which enables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bit 1 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Enable bit, which enables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

22.8.7. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x0A
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
			MC1	MC0			ERR	OVF
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 4, 5 – MCn Match or Capture Channel n
This flag is set on a comparison match, or when the corresponding CC[n] register contains a valid capture value. This flag will generate an interrupt request if the corresponding Match or Capture Channel n Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.MCn) is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the corresponding Match or Capture Channel n interrupt flag.
In capture operation, this flag is automatically cleared when CC[n] register is read.

Bit 1 – ERR Error Interrupt Flag
This flag is set when a new capture occurs on a channel while the corresponding Match or Capture Channel n interrupt flag is set, in which case there is nowhere to store the new capture.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Error Interrupt flag.

Bit 0 – OVF Overflow Interrupt Flag
This flag is set after an overflow condition occurs and will generate an interrupt request if INTENSET.OVF is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Overflow Interrupt flag.

22.8.8. Status

Name: STATUS
Offset: 0x0B
Reset: 0x01
Property: Read-Synchronized, Write-Synchronized

Note: This register is read- and write-synchronized: SYNCBUSY.STATUS must be checked to ensure the STATUS register synchronization is complete.

Bit	7	6	5	4	3	2	1	0
			CCBUFV1	CCBUFV0	PERBUFV		SLAVE	STOP
Access			R/W	R/W	R/W		R	R
Reset			0	0	0		0	1

Bits 4, 5 – CCBUFVn Channel n Compare or Capture Buffer Valid
For a compare channel n, the bit n is set when a new value is written to the corresponding CCBUF[n] register.
The bit n is cleared by writing a '1' to it when CTRLB.LUPD is set, or it is cleared automatically by the hardware on UPDATE condition.
For a capture channel n, the bit n is set when a valid capture value is stored in the CCBUF[n] register. The bit n is cleared automatically when the CC[n] register is read.

Bit 3 – PERBUFV Period Buffer Valid
This bit is set when a new value is written to the PERBUF register. The bit is cleared by writing '1' to the corresponding location when CTRLB.LUPD is set, or automatically cleared by the hardware on UPDATE condition.

Bit 1 – SLAVE Client Status Flag
This bit is only available in 32-bit mode on the client (odd-numbered TCn). The bit is set when the associated host (even-numbered TCn) is set to run in 32-bit mode.

Bit 0 – STOP Stop Status Flag
This bit is set when the TC is disabled, on a Stop command, or on an overflow/underflow condition when the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) is '1'.

Value	Description
0	The counter is running
1	The counter is stopped

22.8.9. Waveform Generation Control

Name: WAVE
Offset: 0x0C
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

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

Bits 1:0 – WAVEGEN[1:0] Waveform Generation Mode
This bit field controls the waveform generation operation. It affects the top value, as shown in the *Waveform Output Operations* section. It also controls whether frequency or PWM waveform generation will be used. The waveform generation operations are explained in the *Waveform Output Operations* section.

Value	Name	Description
0x0	NFRQ	Normal Frequency Generation mode
0x1	MFRQ	Match Frequency Generation mode
0x2	NPWM	Normal Pulse-Width Modulation mode
0x3	MPWM	Match Pulse-Width Modulation mode

22.8.10. Driver Control

Name: DRVCTRL
Offset: 0x0D
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

Bit	7	6	5	4	3	2	1	0
							INVEN1	INVEN0
Access							R/W	R/W
Reset							0	0

Bits 0, 1 – INVENn Waveform Output n Invert Enable
The INVENn bit selects inversion of the output or capture trigger input of channel n.

Value	Description
0	Disable inversion of the WO[n] output and I/O input pin
1	Enable inversion of the WO[n] output and I/O input pin

22.8.11. Debug Control

Name: DBGCTRL
Offset: 0x0F
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGRUN Run in Debug Mode
This bit is not affected by a software Reset and must not be changed by the software while the TC is enabled.

Value	Description
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

22.8.12. Synchronization Busy

Name: SYNCBUSY
Offset: 0x10
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	CC1	CC0	PER	COUNT	STATUS	CTRLB	ENABLE	SWRST
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 6, 7 – CCn Compare/Capture Channel n Synchronization Busy
For details on CC channels number, refer to each TC feature list.
This bit is cleared when the synchronization of CC[n] between the clock domains is complete.
This bit is set when the synchronization of CC[n] between clock domains is started.
This bit is also set when the CCBUF[n] is written, and cleared on update condition. The bit is automatically cleared when the STATUS.CCBUF[n] bit is cleared.

Bit 5 – PER PER Synchronization Busy
This bit is cleared when the synchronization of PER between the clock domains is complete.
This bit is set when the synchronization of PER between clock domains is started.
This bit is also set when the PERBUF is written, and cleared on update condition. The bit is automatically cleared when the STATUS.PERBUF bit is cleared.

Bit 4 – COUNT COUNT Synchronization Busy
This bit is cleared when the synchronization of COUNT between the clock domains is complete.
This bit is set when the synchronization of COUNT between clock domains is started.

Bit 3 – STATUS STATUS Synchronization Busy
This bit is cleared when the synchronization of STATUS between the clock domains is complete.
This bit is set when a '1' is written to the Capture Channel Buffer Valid status flags (STATUS.CCBUFVn) and the synchronization of STATUS between clock domains is started.

Bit 2 – CTRLB CTRLB Synchronization Busy
This bit is cleared when the synchronization of CTRLBSET or CTRLBCLR between the clock domains is complete.
This bit is set when the synchronization of CTRLBSET or CTRLBCLR between clock domains is started.

Bit 1 – ENABLE ENABLE Synchronization Busy
This bit is cleared when the synchronization of ENABLE bit between the clock domains is complete.
This bit is set when the synchronization of ENABLE bit between clock domains is started.

Bit 0 – SWRST SWRST Synchronization Busy
This bit is cleared when the synchronization of SWRST bit between the clock domains is complete.
This bit is set when the synchronization of SWRST bit between clock domains is started.

22.8.13. Counter - COUNT32

Name: COUNT
Offset: 0x14
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. Prior to any read access, this register must be manually read synchronized by writing to the Command bit in the Control B Set register (CTRLBSET.CMD = READSYNC) and wait for the manual read synchronization to finish (SYNCBUSY.CTRLB and CTRLBSET.CMD are '0').
2. This register is write-synchronized. The COUNT Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.COUNT) must be checked to ensure that the COUNT register write synchronization is complete.

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
This bit field contains the current counter value.

22.8.14. Channel n Compare/Capture - COUNT32

Name: CC[n]
Offset: 0x1C + n*0x04 [n=0..1]
Reset: 0x00000000
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.CC[n] must be checked to ensure the CC[n] register synchronization is complete.

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] Channel Compare/Capture Value
This bit field contains the compare/capture value in 32-bit TC mode. In Match frequency (MFRQ) or Match PWM (MPWM) waveform operation (WAVE.WAVEGEN), the CC0 register is used as a period register.

22.8.15. Channel n Compare Buffer - COUNT32

Name: CCBUF[n]
Offset: 0x30 + n*0x04 [n=0..1]
Reset: 0x00000000
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.CC[n] must be checked to ensure the CCBUF[n] register synchronization is complete.

Bit	31	30	29	28	27	26	25	24
	CCBUF[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
	CCBUF[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
	CCBUF[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
	CCBUF[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 – CCBUF[31:0] Channel Compare Buffer Value
This bit field holds the value of the Channel n Compare Buffer Value. When the buffer valid flag is '1' and double buffering is enabled (CTRLBCLR.LUPD=1), the data from buffer registers will be copied into the corresponding CC[n] register under UPDATE condition, including the software update command (CTRLBSET.CMD=0x3).

23. TCC - Timer/Counter for Control Applications

23.1. Features

- Up to four Compare/Capture Channels (CC) with:
  - Double buffered period setting
  - Double buffered compare or capture channel
  - Circular buffer on period and compare channel registers
- Waveform Generation:
  - Frequency generation
  - Single-slope pulse-width modulation (PWM)
  - Dual-slope PWM with half-cycle reload capability
- Input Capture:
  - Event capture
  - Frequency capture
  - Pulse-width capture
- Waveform Extensions:
  - Configurable distribution of compare channels outputs across port pins
  - Pattern generation with double buffer support
- Fault Protection for Safe Disabling of Drivers:
  - Two recoverable fault sources
  - Two non-recoverable fault sources
  - Debugger can be a source of non-recoverable fault
- Input Events:
  - Two input events (EV) for counter
  - One input event (MC) for each channel
- Output Events:
  - Three output events (Count, ReTrigger and Overflow) are available for counter
  - One Compare Match/Input Capture event output for each channel
- Interrupts:
  - Overflow and Retrigger interrupt
  - Compare Match/Input Capture interrupt
  - Interrupt on fault detection
  - Counter cycle interrupt
  - Error condition interrupt
- Can be used with Direct Memory Access (DMA) and can Trigger DMA Transactions

Table 23-1. TCC Configuration Summary

TCCn	Channels (CCn)	Waveform Output (WOn)	Counter Size	Fault	Output Matrix	Pattern Generation
0	4	4	16-bit	Yes	Yes	Yes

23.2. Overview

The device provides one instance of the Timer/Counter for Control (TCC) applications peripheral.

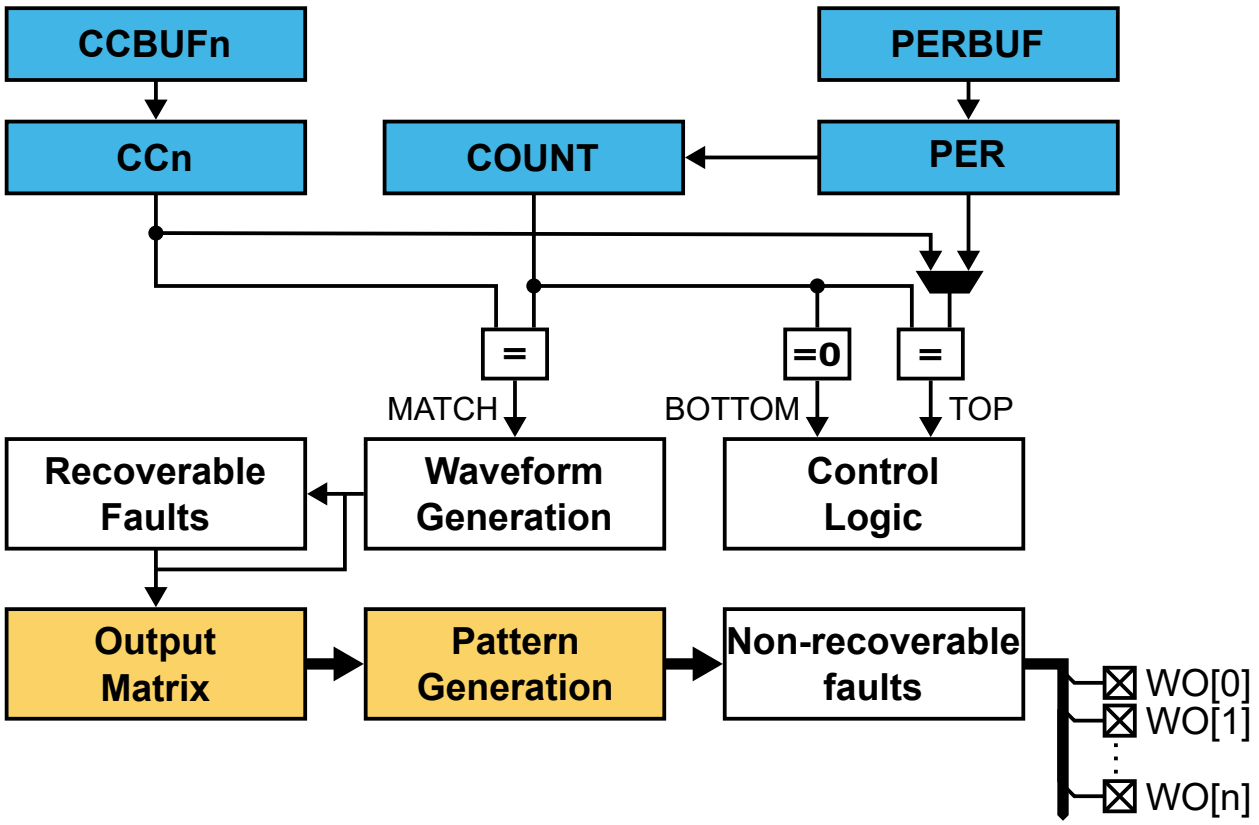
Each TCC instance consists of a counter, a prescaler, compare/capture channels and, control logic. The counter can be set to count events or clock pulses. Together with the compare/capture channels, the counter can be configured to timestamp input events, allowing for the capture of frequency and pulse-width. It can also perform waveform generation, such as frequency generation and pulse-width modulation.

Waveform extensions are featured for motor control, ballast, LED, H-bridge, power converters, and other types of power control applications. They allow for low-side and high-side output with optional dead-time insertion. Waveform extensions can also generate a synchronized bit pattern across the waveform output pins. The fault options enable fault protection for safe and deterministic handling, disabling and/or shut down of external drivers.

Note: The TCC configurations, such as channel numbers and features, may be reduced for some of the TCC instances.

23.3. Block Diagram

Figure 23-1. Timer/Counter for Control Applications - Block Diagram



23.3.1. Signal Description

Pin Name	Type	Description
WOn	Digital output	Compare channel n waveform output

Refer to the *Pinout and Multiplexing* section for details on the pin mapping for this peripheral. One signal can be mapped to several pins.

23.4. Functional Description

23.4.1. Initialization

The following registers are enable-protected, meaning that they can only be written when the TCC is disabled(CTRLA.ENABLE=0):

- Control A register (CTRLA), except for the Run Standby (RUNSTDBY), Enable (ENABLE) and Software Reset (SWRST) bits
- Fault Control x register (FCTRLx)
- Waveform Extension Control register (WEXCTRL)
- Driver Control register (DRVCTRL)
- Event Control register (EVCTRL)

Enable-protected bits in the CTRLA register can be written at the same time as CTRLA.ENABLE is written to '1', but not at the same time as CTRLA.ENABLE is written to '0'. Enable-protection is denoted by the “Enable-Protected” property in the register description.

Before the TCC is enabled, it must be configured as outlined by the following steps:

1. Enable the TCC bus clock (CLK_TCCn_APB).
2. If Capture mode is required, enable the channel in capture mode by writing a '1' to the Capture Enable bit in the Control A register (CTRLA.CPTEN).

Optionally, the following configurations can be set before enabling TCC:

1. Select desired setting in the Prescaler bit field in the Control A register (CTRLA.PRESCALER).
2. Select desired setting in the Prescaler and Counter Synchronization bit field in Control A register (CTRLA.PRESCSYNC).
3. If down-counting operation is desired, write '1' to the Counter Direction bit in the Control B Set register (CTRLBSET.DIR) .
4. Select the desired operation in the Waveform Generation bit field in the WAVE register (WAVE.WAVEGEN).
5. Select the desired output polarity in the Waveform Output Polarity bit field in the WAVE register (WAVE.POL).
6. The waveform output can be inverted for individual channels using the Waveform Output x Inversion bit in the Driver Control register (DRVCTRL.INVENx).

23.4.2. Operation

The following definitions are used throughout the documentation:

Table 23-2. Timer/Counter for Control Applications - Definitions

Name	Description
TOP	The counter reaches TOP when it becomes equal to the highest value in the count sequence. The TOP value can be the same as Period (PER) or the Compare Channel 0 (CC0) register value depending on the waveform generator mode in the *Waveform Output Generation Operations* section.
ZERO	The counter reaches ZERO when it contains all zeroes
MAX	The counter reaches maximum when it contains all ones
UPDATE	The timer/counter signals an update when it reaches ZERO or TOP, depending on the direction settings
Timer	Increment / decrement / clear / reload steps are done on each prescaled clock
Counter	Increment / decrement / clear / reload steps is done on each detected events
CCn	Capture/Compare channel n based on operation mode
MCn	Match/Compare channel n based on operation mode

Table 23-2. Timer/Counter for Control Applications - Definitions (continued)

Name	Description
CC_NUM	The number of Compare/Capture channels
WO_NUM	The number of Waveform Outputs

Each TCC instance has up to four Compare/Capture channels (CCn).

The Period register (PER), the Compare/Capture Channel n register (CCn) and the Pattern register (PATT) have buffered versions in the Period Buffer register (PERBUF), the Compare/Capture Buffer Channel x register (CCBUFx) and the Pattern Buffer register (PATTBUF). Each buffer register has a buffer valid flag in the Status register (STATUS) that indicates when the buffer contains a new value.

Under normal operation, the counter value is continuously compared to the TOP or ZERO value to determine whether the counter has reached TOP or ZERO. In either case, the TCC can generate interrupt requests, request Direct Memory Access (DMA) transactions, or generate events for the Event System. In waveform generator mode, these comparisons are used to set the waveform period or pulse alignment.

A prescaled generic clock (GCLK_TCCn) and events from the event system can be used to control the counter. The event system is used as a source for the input capture.

The Recoverable Fault Unit enables event controlled waveforms by acting directly on the generated waveforms of the TCC compare channels output. These events can restart, halt the timer/counter period, shorten the output pulse active time, or disable waveform output as long as the fault condition is present. This can typically be used for current sensing regulation, zero-crossing and demagnetization retriggering.

The MCE0 and MCE1 (Match or Capture Channel) asynchronous event sources are shared with the Recoverable Fault Unit. Only asynchronous events are used internally when fault unit extension is enabled. For further details on how to configure asynchronous events routing, refer to the *EVSYS - Event System* section.

Recoverable fault sources can be filtered and/or windowed to avoid false triggering, for example from I/O pin glitches, by using digital filtering, input blanking, and qualification options. See also the *Recoverable Faults* section.

To support different applications the following additional features are implemented in some or all of the TCC instances as optional and successive units:

- Recoverable faults and non-recoverable faults
- Output matrix
- Pattern generation

See the *Overview* section for more information on which TCC instance has additional features, and the *Block Diagram* section for how the additional features connect.

The output matrix (OTMX) can distribute and route the TCC waveform outputs across the port pins in different configurations, each optimized for different application types.

The pattern generation unit can be used to generate synchronized waveforms with a constant logic level on TCC UPDATE conditions. This is useful for easy stepper motor and full bridge control.

The non-recoverable fault module enables event controlled fault protection by acting directly on the generated waveforms of the timer/counter compare channel outputs. When a non-recoverable fault condition is detected, the output waveforms are forced to a per-configured value that is safe for the application. This is typically used for instant and predictable shutdown and disabling high-current or high-voltage drives.

The count event sources (TCE0 and TCE1) are shared with the non-recoverable fault extension. The events can be optionally filtered.

Note: MCE0 and MCE1 can also be used as non-recoverable event source.

If the filter options are not used, the non-recoverable faults provide an immediate asynchronous action on waveform output, even for cases where the clock is not present. For further details on how to configure asynchronous events routing, refer to the *EVSYS - Event System* section.

23.4.2.1. Enabling, Disabling and Resetting

The TCC is enabled by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE). The TCC is disabled by writing a '0' to CTRLA.ENABLE.

The TCC is reset by writing '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the TCC, except for the Debug control register (DBGCTRL), will be reset to their initial state, and the TCC will be disabled. Refer to the Control A (CTRLA) register for details.

The TCC should be disabled before it is reset to avoid undefined behavior.

23.4.2.2. Prescaler Selection

The GCLK_TCCn clock is fed into the internal prescaler.

The prescaler consists of a counter that counts up to the selected prescaler value, whereupon the output of the prescaler toggles.

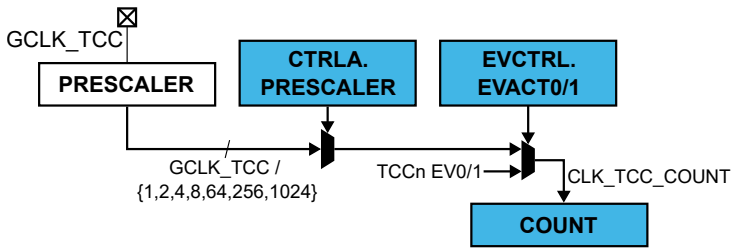
If the precaler factor in the Prescaler bit field in the Control A register (CTRLA.PRESCALER) is above DIV1, the counter update condition can optionally be executed on the next GCLK_TCCn clock pulse or the next prescaled clock pulse. For further details, refer to the description of the CTRLA.PRESCALER bit field and the Prescaler and Counter Synchronization bit field in the Control A register (CTRLA.PRESYNC).

Prescaler outputs from 1 to 1/1024 are available. For a complete list of available prescaler outputs, see the register description for the CTRLA.PRESCALER bit fields.

Note: When counting events, the prescaler is bypassed.

The joint stream of prescaler ticks and event action ticks is called CLK_TCC_COUNT.

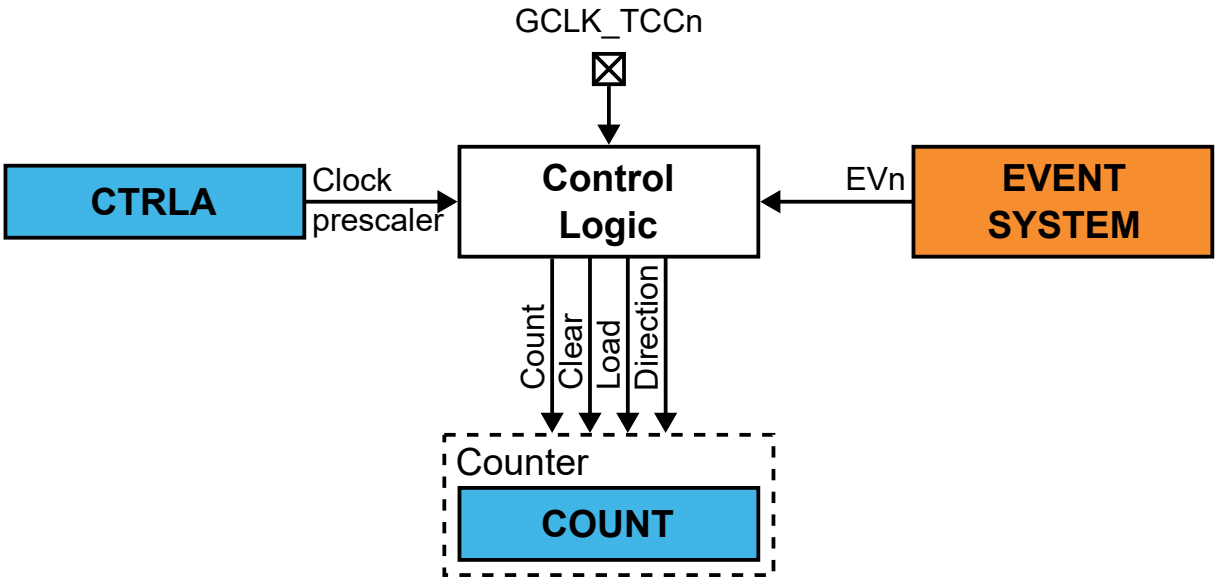
Figure 23-2. Prescaler



23.4.2.3. Counter Operation

Depending on the mode of operation, the counter is cleared, reloaded, incremented, or decremented at each TCC clock input (CLK_TCC_COUNT). A counter clear or reload marks the end of current counter cycle and the start of a new one.

Figure 23-3. TCC - Counter Operation

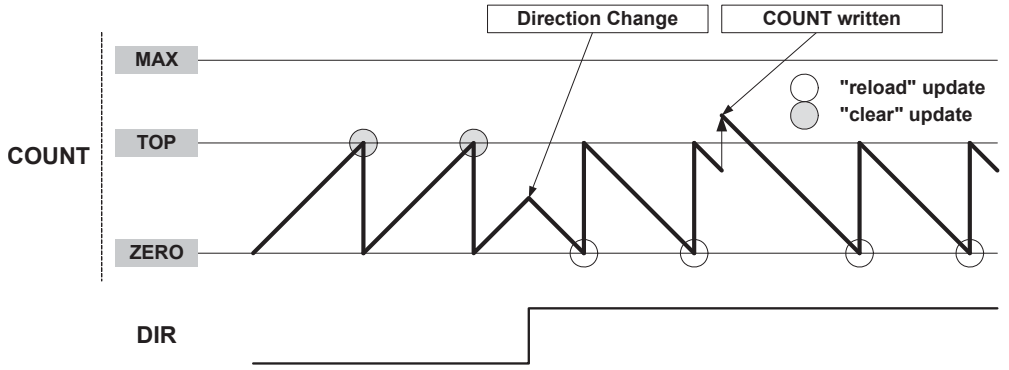


The counting direction is set by writing to the Counter Direction bit in the Control B Set register (CTRLBSET.DIR). The counter counts down if the bit is '1', and up when the bit is '0'.

The counter will count up or down for each tick (clock or event) until it reaches TOP or ZERO. When counting up and TOP is reached, the counter will be set to zero at the next tick (overflow), and the Overflow Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.OVF) will be set. When counting down, the counter is reloaded with the TOP value when ZERO is reached (underflow), and INTFLAG.OVF is set.

INTFLAG.OVF can be used to trigger an interrupt, a Direct Memory Access (DMA) request, or an event. An overflow or underflow occurrence (i.e, a compare match with TOP or ZERO) will stop counting if the One-Shot bit in the Control B register (CTRLBSET.ONESHOT) is set. The One-Shot feature is explained in the Additional Features section.

Figure 23-4. Counter Operation



It is possible to change the counter value (by writing directly to the Counter register (COUNT)) even when the counter is running. The COUNT value will always be ZERO or TOP, depending on the direction set by the Counter Direction bit in the Control B Clear or Set registers (CTRLBCLR/SET.DIR), when starting the TCC, unless a different value has been written to it or the TCC has been stopped at a value other than ZERO. Write access has higher priority than count, clear, or reload operations. The direction of the counter can also be changed during normal operation. See the *Counter Operation* figure above for further information.

Stop Command

A stop command can be issued from software by writing the STOP command value to the TCC Command bit field in the Control B Set register (CTRLBSET.CMD).

When a stop is detected while the counter is running, the counter will maintain its current value. If waveform generation (WG) is used, all waveforms are set to a state defined in Non-Recoverable State x Output Enable bit and Non-Recoverable State x Output Value bit in the Driver Control register (DRVCTRL.NREx and DRVCTRL.NRVx), and the Stop bit in the Status register (STATUS.STOP) is set.

Stop Event Action

A stop command can also be triggered by event input 1 by writing the STOP action value to the Input Event Action 1 bits in the Event Control register (EVCTRL.EVACT1).

When a stop action is detected, the counter can stop immediately, maintaining its current value and all waveforms keep their current state, as long as a start event action is detected through the: Input Event Action 0 bits in Event Control register (EVCTRL.EVACT0).

Retrigger Command and Event Action

A retrigger command can be issued by software by writing the RETRIGGER command value to the CTRLBSET.CMD bit field, or by events by writing the RETRIGGER action value to the Input Event 0/1 Action bits in Event Control register (EVCTRL.EVACTx).

When the command is detected during counting operation, the counter will be reloaded or cleared, depending on the counting direction set in CTRLBCLR/SET.DIR, and set the Retrigger Interrupt Flag bit (TRG) bit in the Interrupt Flag Status and Clear register (INTFLAG) . It is also possible to generate an event by writing a '1' to the Retrigger Event Output Enable bit in the Event Control register (EVCTRL.TRGEO).

Note: When a retrigger event action is configured in the EVCTRL.EVACTx bit field, enabling the counter will not start it. The counter will start on the next incoming event and restart on each subsequent corresponding event.

Start Event Action

The start action can be enabled by writing the START action value to the Timer/Counter Event Input 0 Action bit field in the Event Control register (EVCTRL.EVACT0), and can start the counting operation when previously stopped. The event has no effect if the counter is already counting. When the module is enabled, the counter operation starts when the event is received or when a retrigger software command is applied.

Note: When a start event action is configured in the EVCTRL.EVACT0 bit field, enabling the counter will not start the counter. The counter will start on the next incoming event, but it will not restart on subsequent events.

Count Event Action

The TCC can count events. When an event is received, the counter increments or decrements its value, depending on the direction set in CTRLBCLR/SET.DIR.

The count event action can be enabled by writing the COUNTEV or COUNT_ASYNC action value to the EVCTRL.EVACT0 bit field.

Count on Active State of Asynchronous Event

The TCC counts during the active state of an asynchronous event (incrementing or decrementing, regardless of the direction set in CTRLBCLR/SET.DIR).

Direction Event Action

The direction event action can be enabled by writing the DIR_ASYNC action value to the EVCTRL.EVACT1 bit field. When this event is triggered, the asynchronous event path specified in the event system must be configured or selected. The direction event action can be used to control the direction of the counter operation, depending on external events level. When received, the event level overrides the Direction settings in CTRLBCLR/SET.DIR and the direction bit value is updated accordingly.

Increment Event Action

The increment event action can be enabled by writing the INC action value to the EVCTRL.EVACT0 bit field and can change the counter state when an event is received. When the TCE0 event (TCCn_EV0) is received, the counter increments, regardless of the direction setting in CTRLBCLR/SET.DIR is.

Decrement Event Action

The decrement event action can be enabled by writing the DEC action value to the EVCTRL.EVACT1 bit field, and can change the counter state when an event is received. When the TCE1 (TCCn_EV1) event is received, the counter decrements, regardless of the direction setting in CTRLBCLR/SET.DIR is.

Non-recoverable Fault Event Action

Non-recoverable fault actions can be enabled by writing the FAULT_ASYNC action value to the EVCTRL.EVACTx bit field. When such an event is received, the counter will be stopped, and the output of the compare channels will be overridden according to the Driver Control register settings in the DRVCTRL.NREx and DRVCTRL.NRVx bit fields. TCE0 and TCE1 must be configured as asynchronous events.

Event Action Off

If the event action is disabled by writing the OFF action value to the EVCTRL.EVACTx bit field, enabling the counter will also start it.

23.4.2.4. Compare Operations

By default, the Compare/Capture channel is configured for compare operations. To perform capture operations, it must be reconfigured.

When using the TCC with the Compare/Capture Channel n registers (CCn) for compare operations, the counter value is continuously compared to the values in the CCn registers. This can be used for timer or waveform operations.

The Compare/Capture Buffer Channel x registers (CCBUFx) provide double buffering capability. Double buffering synchronizes the update of the CCn register with the buffer value at the UPDATE condition, or if a force update command is issued by writing the UPDATE command value to the TCC Command bit field in the Control B Set register (CTRLBSET.CMD). For further details, refer to the Double Buffering section. This synchronization prevents the occurrence of odd-length, non-symmetrical pulses and ensures glitch-free output.

23.4.2.4.1. Waveform Output Generation Operations

The compare channels can be used for waveform generation on output port pins. To make the waveform available on the connected pin, the following requirements must be fulfilled:

1. Choose a waveform generation mode by writing to the Waveform Generation Operation bit in the Waveform register (WAVE.WAVEGEN).
2. Optionally, invert the waveform output WO[x] by writing to the corresponding Waveform Output x Inversion bit in the Driver Control register (DRVCTRL.INVENx).
3. Configure the pins with the I/O Pin Controller. Refer to the *PORT - I/O Pin Controller* section for details.

Note: The MCx event must not be used when the compare channel is set in waveform output operating mode, except when it is used as a non-recoverable fault input.

The counter value is continuously compared with each CCn value. On a compare match, the Match or Capture Channel x bit in the Interrupt Flag Status and Clear register (INTFLAG.MCx) will be set on the next zero-to-one transition of CLK_TCC_COUNT (see Normal Frequency Operation). An interrupt and/or event can be generated under the same condition if a match or capture occurs, that is if the Match or Capture Channel x Interrupt Enable bit in the interrupt Enable Set register (INTENSET.MCx) and/or the Match or Capture Channel x Event Output Enable bit in the Event Control register (EVCTRL.MCEOx) is set to '1'. Both an interrupt and an event can be generated simultaneously. The same condition also generates a DMA request.

There are seven waveform configurations for the Waveform Generation Operation bit field in the Waveform register (WAVE.WAVEGEN). This will influence how the waveform is generated and impose restrictions on the top value. The configurations are:

- Normal Frequency (NFRQ)
- Match Frequency (MFRQ)
- Normal Pulse-Width Modulation (NPWM)
- Dual-slope, interrupt/event at TOP (DSTOP)
- Dual-slope, interrupt/event at ZERO (DSBOTTOM)
- Dual-slope, interrupt/event at Top and ZERO (DSBOTH)
- Dual-slope, critical interrupt/event at ZERO (DSCRITICAL)

When using the MFRQ configuration, the TOP value is defined by the CC0 register value. For the other waveform operations, the TOP value is defined by the Period (PER) register value.

For dual-slope waveform operations, the update occurs when the counter reaches ZERO. For the other waveform generation modes, the update occurs on counter wraparound, overflow, underflow, or retrigger.

The following table shows the update counter and overflow event/interrupt generation conditions in different operation modes.

Table 23-3. Counter Update and Overflow Event/interrupt Conditions

Name	Operation	TOP	Update	Output Waveform		OVFIF/Event	
				On Match	On Update	Up	Down
NFRQ	Normal Frequency	PER	TOP/ ZERO	Toggle	Stable	TOP	ZERO
MFRQ	Match Frequency	CC0	TOP/ ZERO	Toggle	Stable	TOP	ZERO
NPWM	Single-slope PWM	PER	TOP/ ZERO	See section 'Output Polarity' below		TOP	ZERO
DSCRITICAL	Dual-slope PWM	PER	ZERO			-	ZERO
DSBOTTOM	Dual-slope PWM	PER	ZERO			-	ZERO
DSBOTH	Dual-slope PWM	PER	TOP(1) & ZERO			TOP	ZERO
DSTOP	Dual-slope PWM	PER	ZERO			TOP	–

Note:

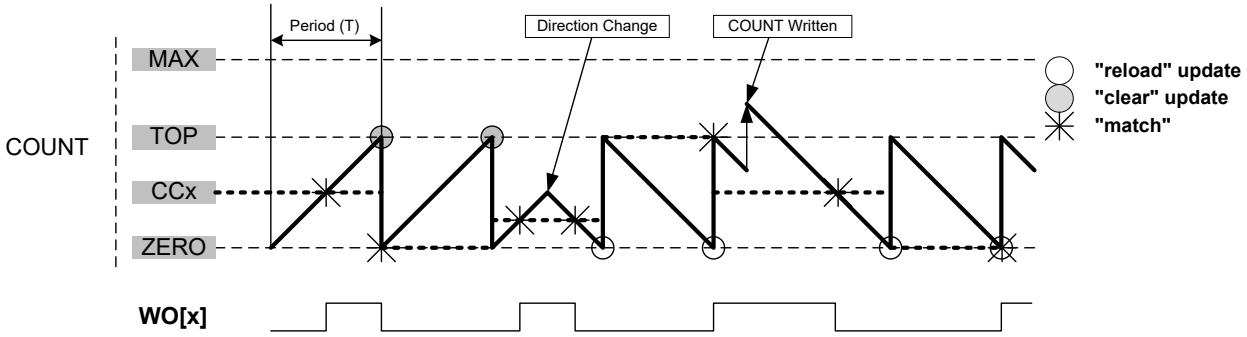
1. The UPDATE condition on TOP will only occur when the circular buffer is enabled for the channel. Refer to the *Circular Buffer* section for details.

23.4.2.4.2. Normal Frequency (NFRQ)

For Normal Frequency generation, the period time (T) is controlled by the Period register (PER). The phase of the waveform generation output (WO[n]) is controlled by the Compare/Capture Value

in the Compare/Capture Channel n register (CCn.CC). WO[n] is toggled on each compare match between Counter Value in the Counter register (COUNT.COUNT) and CCn.CC. The corresponding Match or Capture Channel n Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.MCn) will then be set.

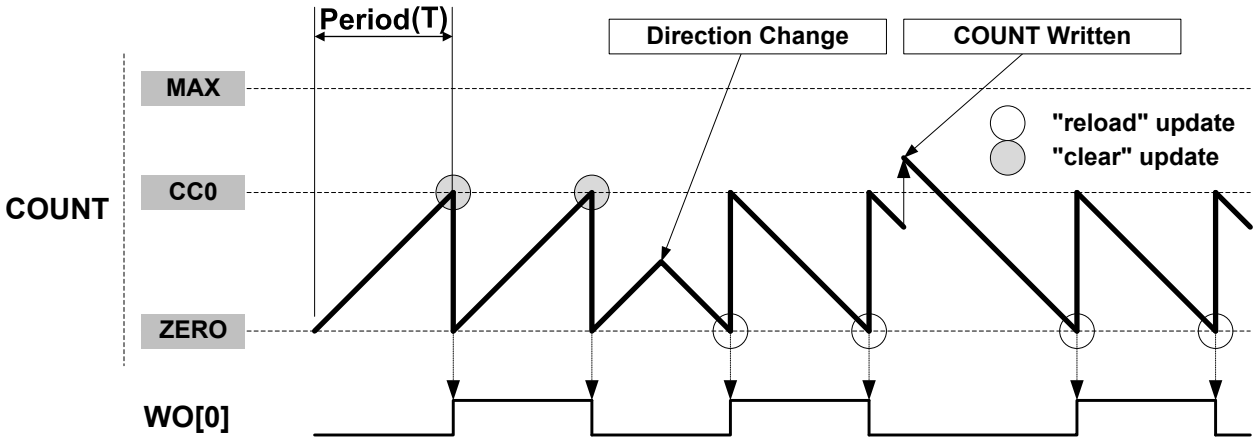
Figure 23-5. Normal Frequency Operation



23.4.2.4.3. Match Frequency (MFRQ)

For Match Frequency generation, the period time (T) is controlled by the Compare/Capture Channel 0 register (CC0) instead of PER. WO[0] toggles on each update condition.

Figure 23-6. Match Frequency Operation



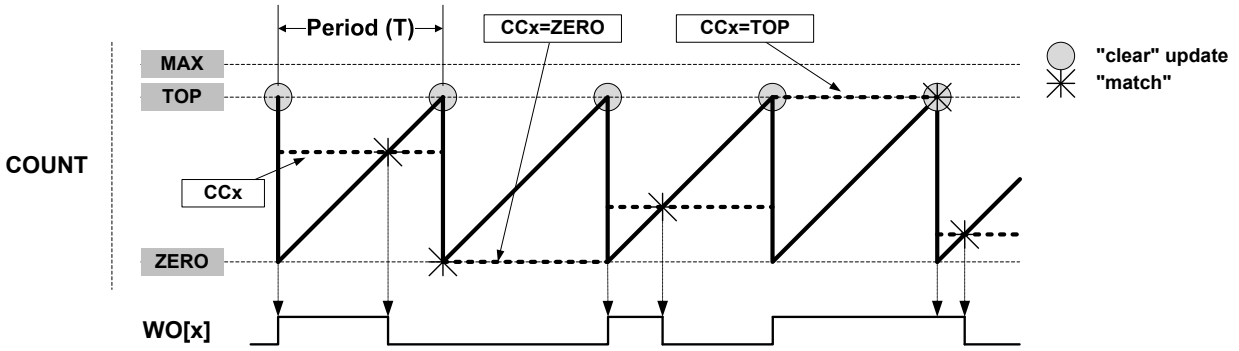
23.4.2.4.4. Normal Pulse-Width Modulation (NPWM)

NPWM uses single-slope PWM generation.

23.4.2.4.5. Single-Slope PWM Operation

For single-slope PWM generation, the period time (T) is controlled by the TOP value set in the Period Value bit field in the Period register (PER.PER), and the Compare/Capture value in the Compare/Capture Channel n register (CCn.CC) controls the duty cycle of the generated waveform output. When up-counting, WO[n] is set at the start or on a compare match between the COUNT and TOP values, and cleared on a compare match between COUNT and CCn.CC. When down-counting, the WO[n] is cleared at the start or on a compare match between COUNT and ZERO, and set on compare match between COUNT and CCn.CC.

Figure 23-7. Single-Slope PWM Operation



The following equation calculates the exact resolution for a single-slope PWM (R_{PWM_SS}) waveform:

$$R_{PWM\_SS} = \frac{\log(TOP+1)}{\log(2)}$$

The PWM frequency depends on the PER value and the peripheral clock frequency (f_{GCLK_TCC}), and can be calculated by the following equation:

$$f_{PWM\_SS} = \frac{f_{GCLK\_TCC}}{N(TOP+1)}$$

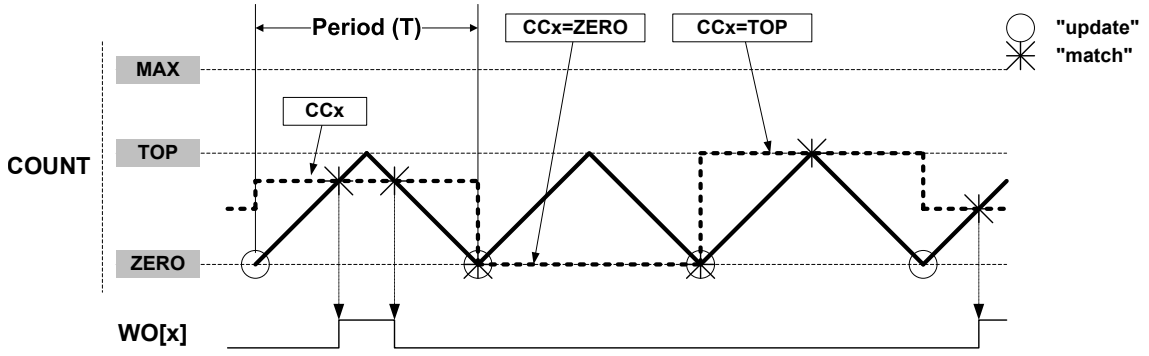
Where N represents the prescaler divider used (1, 2, 4, 8, 16, 64, 256, 1024).

23.4.2.4.6. Dual-Slope PWM Generation

For dual-slope PWM generation, the period time (T) is controlled by the TOP value set in the Period value bit field in the Period register (PER.PER), and the Compare/Capture Value in the Compare/Capture Channel n register (CCn.CC) controls the duty cycle of the generated waveform output. The figure below shows how the counter repeatedly counts from ZERO to TOP and then from TOP to ZERO. The waveform generator output is set on compare match when up-counting and cleared on a compare match when down-counting. An interrupt and/or event is generated on TOP (when counting upwards) and/or ZERO (when counting up or down).

In DSBOTH operation, the circular buffer must be enabled to enable the update condition on TOP.

Figure 23-8. Dual-Slope Pulse Width Modulation



Using dual-slope PWM results in a lower maximum operation frequency compared to single-slope PWM generation. The period (TOP) defines the PWM resolution. The minimum resolution is 1 bit (TOP=0x00000001).

The following equation calculates the exact resolution for dual-slope PWM (R_{PWM_DS}):

$$R_{PWM\_DS} = \frac{\log(TOP+1)}{\log(2)}.$$

The PWM frequency f_{PWM_DS} depends on the period setting (TOP) and the peripheral clock frequency f_{GCLK_TCC}, and can be calculated by the following equation (outside of DSBOTH mode):

$$f_{PWM\_DS} = \frac{f_{GCLK\_TCC}}{2N \cdot TOP}$$

N represents the prescaler divider used. The waveform generated will have a maximum frequency of half the TCC clock frequency (f_{GCLK_TCC}) when TOP is set to 0x00000001 and no prescaling is used.

The pulse width (P_{PWM_DS}) depends on the CCn.CC value and the peripheral clock frequency (f_{GCLK_TCC}), and can be calculated by the following equation:

$$P_{PWM\_DS} = \frac{2N \cdot (TOP - CCx)}{f_{GCLK\_TCC}}$$

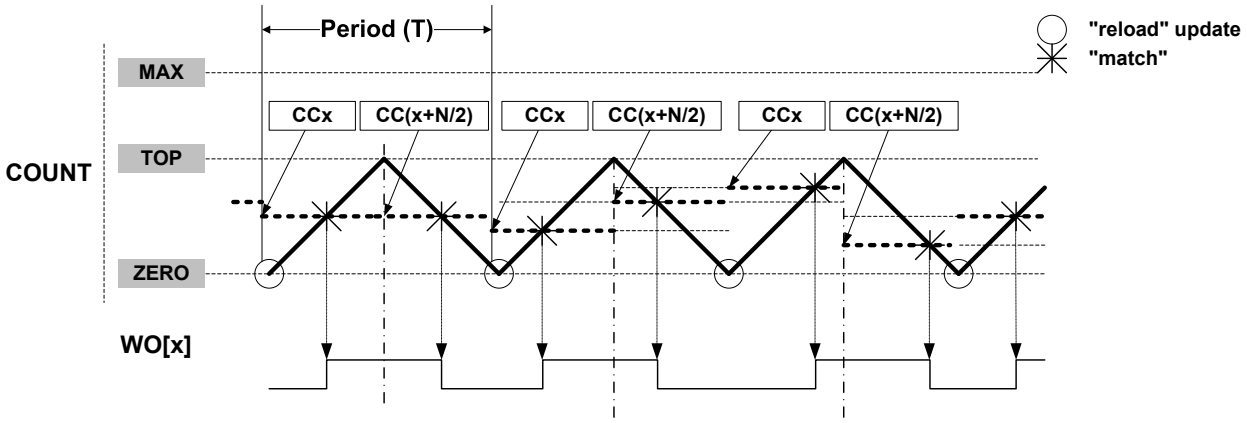
N represents the prescaler divider used.

Note: In DSTOP, DSBOTTOM and DSBOTH operation, when TOP is lower than MAX/2, the MSB bit of CCn.CC defines the ramp on which the CCn Match interrupt or event is generated (rising if CCn[MSB] = 0, falling if CCn[MSB] = 1.)

23.4.2.4.7. Dual-Slope Critical PWM Generation

Critical mode generation allows the generation of non-aligned centered pulses. In this mode, the period time (T) is controlled by TOP value set in the Period Value bit field in the Period register (PER.PER), while the Compare/Capture value in the Compare/Capture Channel n register (CCn.CC) controls the generated waveform output edge during up-counting, and the CC(n+CC_NUM/2).CC value controls the generated waveform output edge during down-counting.

Figure 23-9. Dual-Slope Critical Pulse Width Modulation (N=CC_NUM)



23.4.2.4.8. Output Polarity

The Channel Polarity n bit field in the Waveform register (WAVE.POLn) is available in all waveform output generation modes. In single-slope and dual-slope PWM operation, it is possible to invert the pulse edge alignment individually on start or end of a PWM cycle for each compare channels. The following table shows the waveform output set/clear conditions, depending on the settings of timer/counter, direction, and polarity.

Table 23-4. Waveform Generation Set/Clear Conditions

Waveform Generation operation	DIR	POLn	Waveform Generation Output Update	
			Set	Clear
Single-Slope PWM	0	0	Timer/counter matches TOP	Timer/counter matches CCn
		1	Timer/counter matches CC	Timer/counter matches TOP
	1	0	Timer/counter matches CC	Timer/counter matches ZERO
		1	Timer/counter matches ZERO	Timer/counter matches CC
Dual-Slope PWM	–	0	Timer/counter matches CC when counting up	Timer/counter matches CC when counting down
		1	Timer/counter matches CC when counting down	Timer/counter matches CC when counting up

In Normal and Match Frequency, the value of WAVE.POLn represents the initial state of the waveform output.

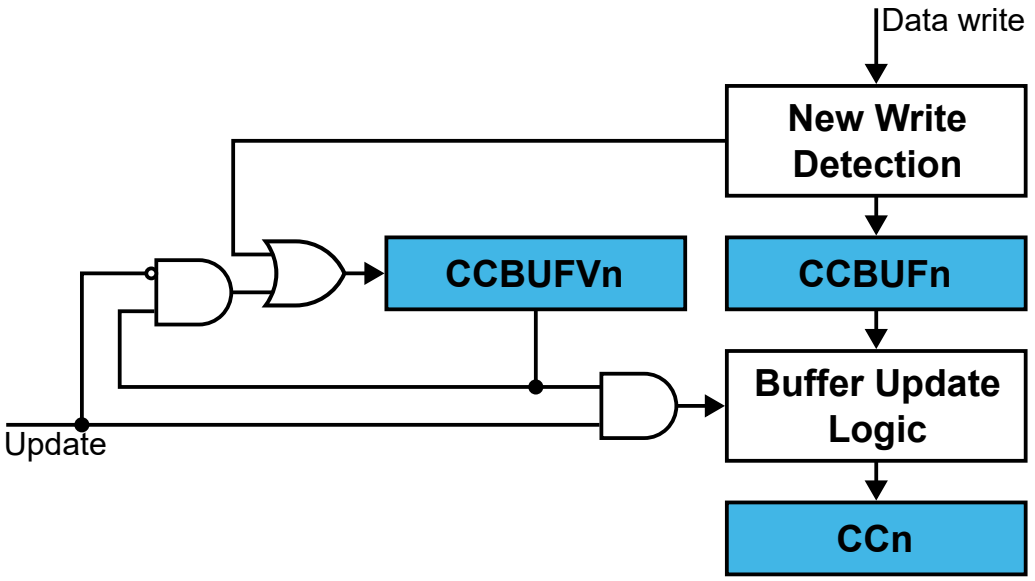
23.4.2.5. Double Buffering

The Pattern (PATT), Period (PER) and Compare Channels (CCn) registers are all double buffered. Each buffer register has a buffer valid bit in the Status register (STATUS), which indicates that the buffer register contains a valid value that can be copied into the corresponding register. As long as the respective Buffer Valid Status flag (PATTBUFV, PERBUFV or CCBUFVx) are set to '1', and the related SYNCBUSY bits are set (SYNCBUSY.PATT, SYNCBUSY.PER or SYNCBUSY.CCn), a write to the respective PATT/PATTBUF, PER/PERBUF or CCn/CCBUFx registers will generate a PAC error, and read access to the respective PATT, PER or CCn register is invalid.

When a Buffer Valid Flag bit in the STATUS register is '1' and the Lock Update bit in the Control B Clear register (CTRLBCLR.LUPD) is cleared, double buffering is enabled: The data from the buffer registers will be copied into the corresponding register under hardware UPDATE conditions, and then the Buffer Valid Flag bit in the STATUS register are automatically cleared by hardware. The buffer valid flag bits in the STATUS register can be cleared manually, but they must be cleared twice successively.
Note: A software update command issued by writing the UPDATE command value to the TCC Command bit field in the Control B Set register (CTRLBSET.CMD) acts independently of LUPD value.

A compare register is double buffered as shown in the following figure.

Figure 23-10. Compare Channel Double Buffering



Both the registers (PATT, PER, and CCn) and the corresponding buffer registers (PATTBUF, PERBUF, and CCBUFx) are available in the I/O register map, and the double buffering feature is not mandatory. Double buffering is disabled by writing a '1' to the Lock Update bit in the Control B Set register (CTRLBSET.LUPD).

Note: When NFRQ, MFRQ or PWM down-counting counter mode (with the Counter Direction bit in the Control B Clear or Set registers (CTRLBCLR/SET.DIR) set) is enabled and double buffering is enabled (CTRLBCLR/SET.LUPD is cleared), the PERBUF register is continuously copied into the PER independently of update conditions.

Figure 23-11. Unbuffered Single-Slope Up-Counting Operation

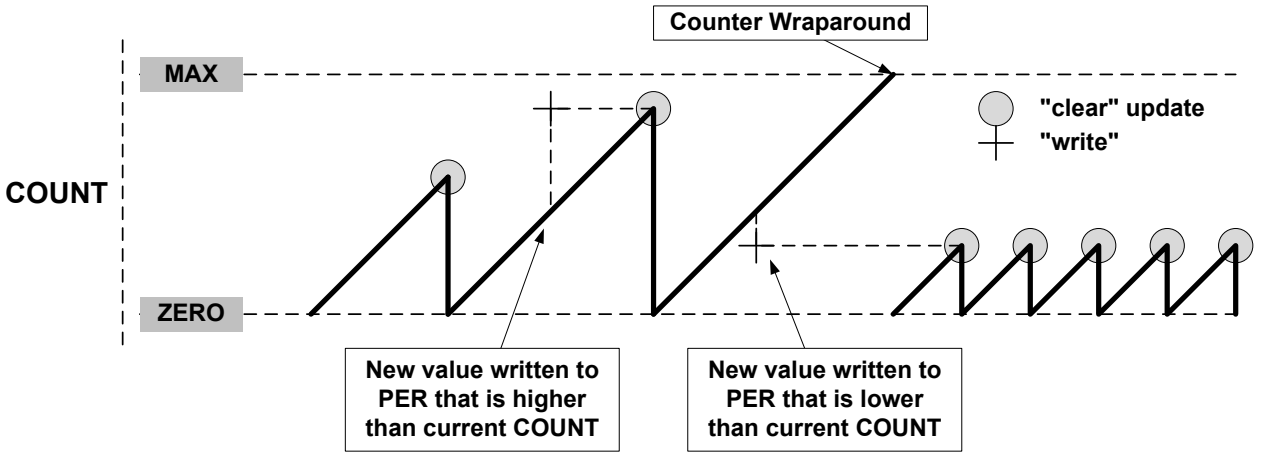
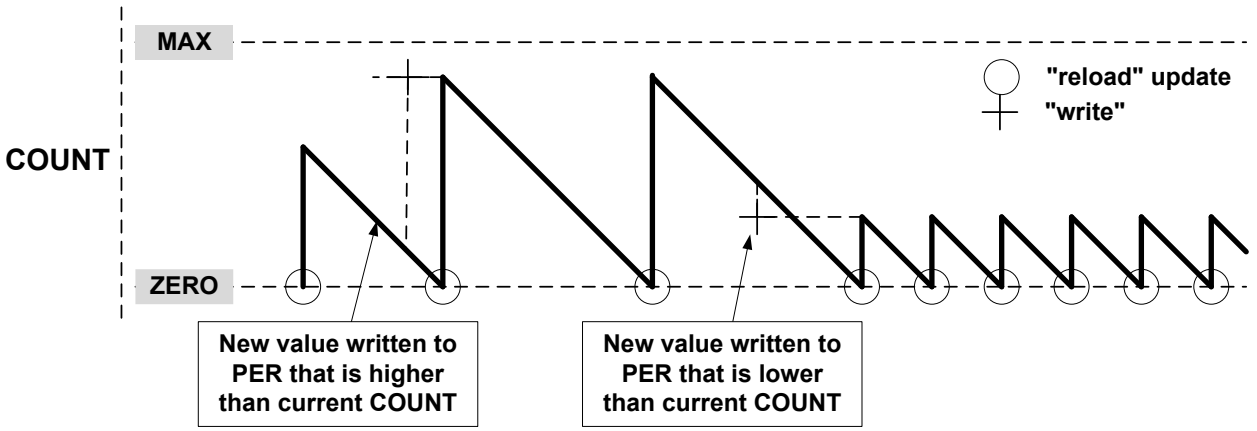
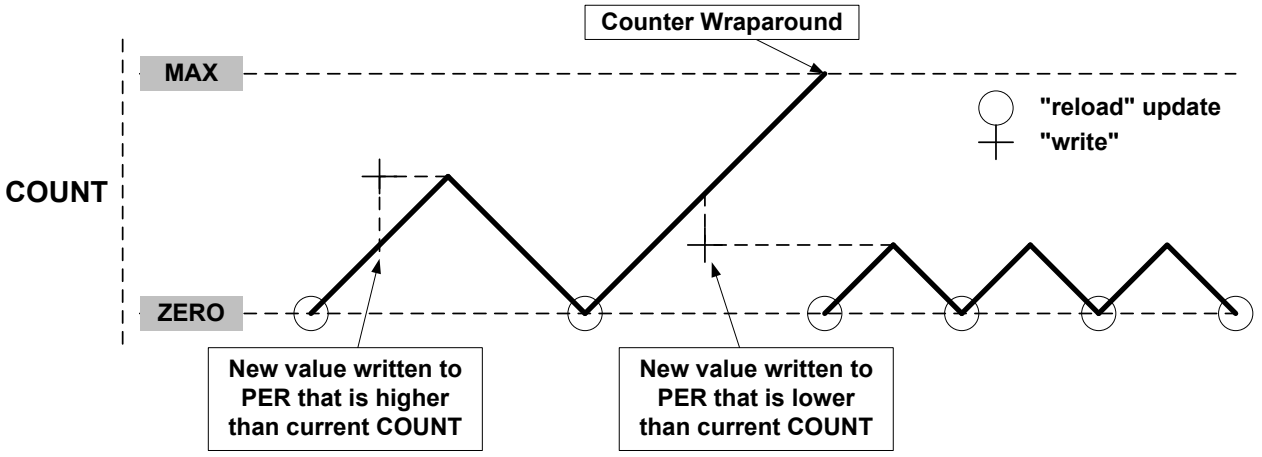


Figure 23-12. Unbuffered Single-Slope Down-Counting Operation



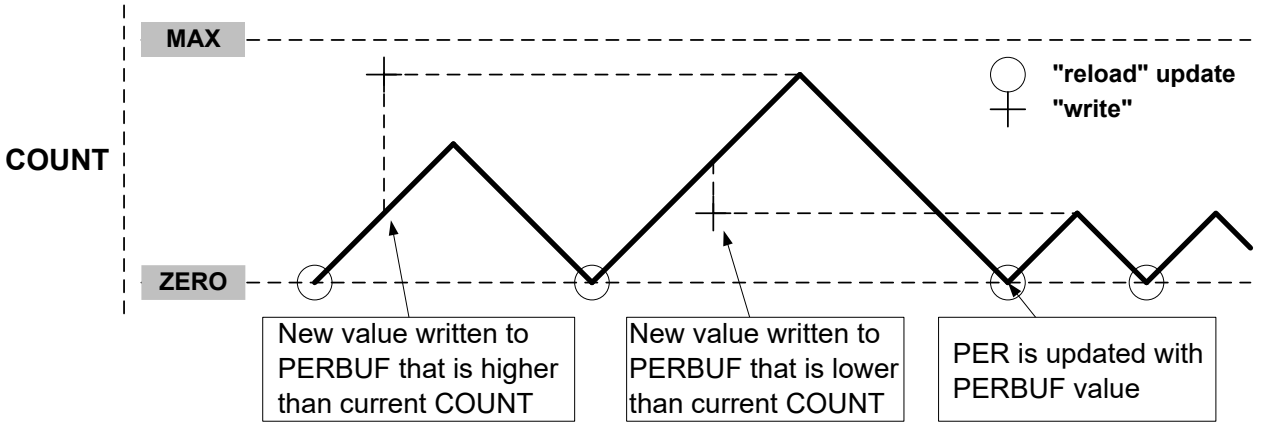
A counter wraparound can occur in any operation mode when up-counting without buffering. COUNT and TOP are continuously compared, so when a new value that is lower than the current COUNT is written to TOP, COUNT will wrap before a compare match occurs.

Figure 23-13. Unbuffered Dual-Slope Operation



When double buffering is used, the buffer can be written at any time, and the counter will still maintain correct operation. The period register is always updated on the update condition, as shown in the following figure. This prevents wraparound and the generation of irregular waveforms.

Figure 23-14. Changing the Period Using Buffering



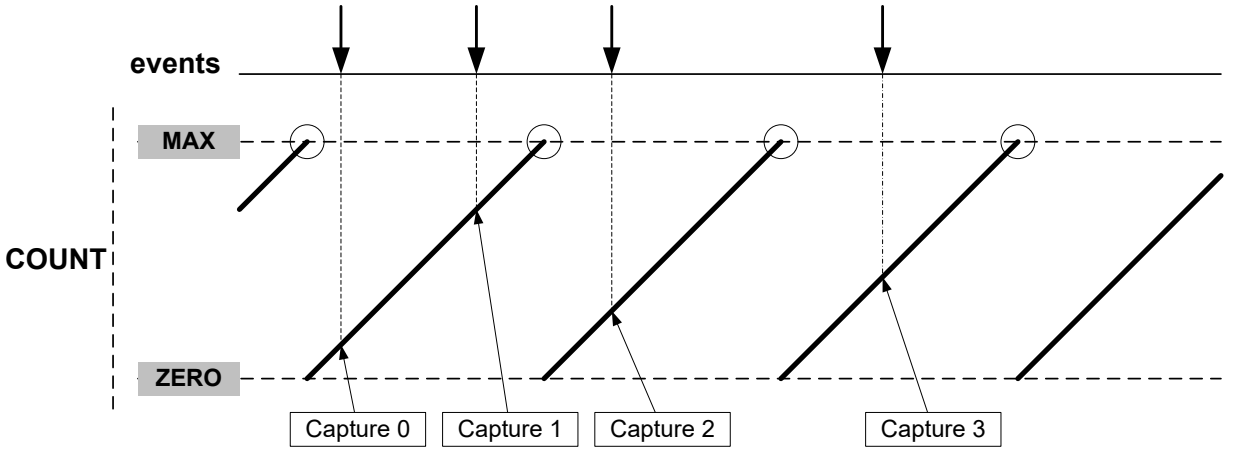
23.4.2.6. Capture Operations

To enable and use capture operations, the Match or Capture Channel x Event Input Enable bit in the Event Control register (EVCTRL.MCEIx) must be set to '1'. The capture channels to be used must also be enabled by setting the Capture Channel x Enable bit in the Control A register (CTRLA.CPTENx) before capturing can be performed. Event system channels must be configured to operate in asynchronous mode when used for capture operations.

Event Capture Action

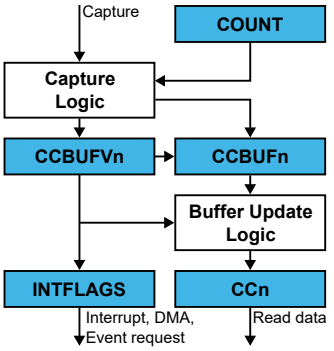
The compare/capture channels can be used as input capture channels to capture events from the Event System, and assign them a timestamp. The following figure shows four capture events for one capture channel.

Figure 23-15. Input Capture Timing



For input capture, the Compare/Capture Buffer Value in the Compare/Capture Buffer Channel x register (CCBUFx.CCBUF) and the corresponding Compare/Capture Value in the Compare/Capture Channel n register (CCn.CC) act like a FIFO. When CCn.CC is empty or read, any content in CCBUFx.CCBUF is transferred to CCn.CC. The buffer valid flag is used to set the Match or Capture Channel x Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.MCx) and generate the optional interrupt, event or DMA request. The CCBUFx.CCBUF value cannot be read all captured data must be read from the CCn register.

Figure 23-16. Capture Double Buffering



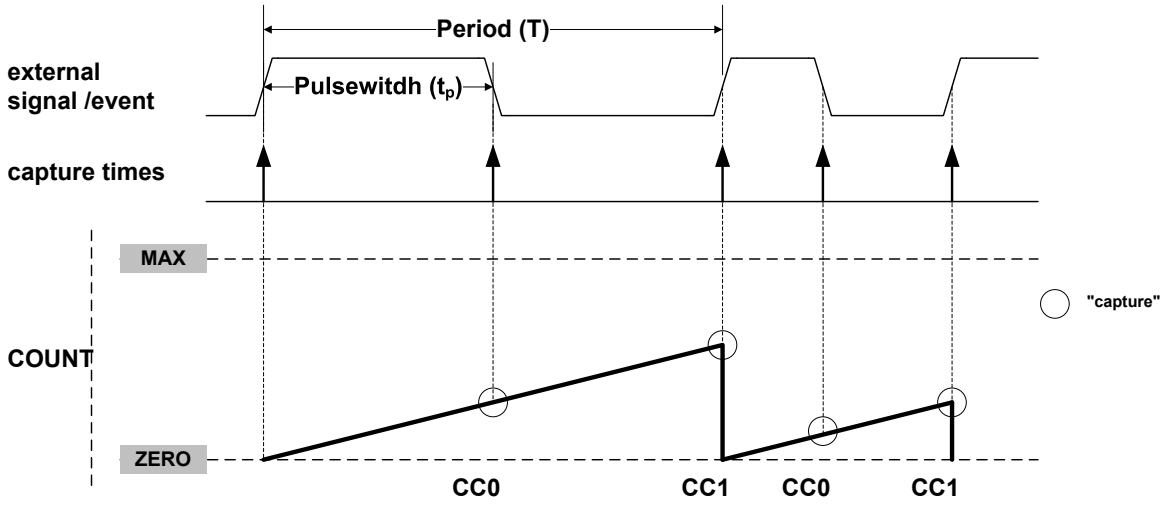
The TCC can detect capture overflow of the input capture channels: When a new capture event is detected while the Channel x Compare or Capture Buffer Valid flag in the Status register (STATUS.CCBUFV) is still set, the new timestamp will not be stored, and the Error Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.ERR) will be set.

Period and Pulse-Width (PPW) Capture Action

The TCC can perform two input captures and restart the counter on one of the edges. This enables the TCC to measure the pulse-width and period, and to characterize the frequency *f* and *dutyCycle* of an input signal:

$$f = \frac{1}{T} \quad , \quad dutyCycle = \frac{t_p}{T}$$

Figure 23-17. PWP Capture



Selecting PWP in the Timer/Counter Event Input 1 Action bit field in the Event Control register (EVCTRL.EVACT1) enables the TCC to perform one capture action on the rising edge and another on the falling edge. When using the event action, the period *T* will be captured into CC1 and the pulse width t_p into CC0.

The Timer/Counter Event n Invert Enable bit in the Event Control register (EVCTRL.TCEINVn) is used for event source x to select whether the wraparound should occur on the rising edge or the falling edge. If the EVCTRL.TCEINVn bit is set, the wraparound will occur on the falling edge.

The corresponding capture is performed only if the channel is enabled in capture mode (CTRLA.CPTENn = 1). If not, the capture action will be ignored, and the channel will be enabled in compare mode of operation. When only one of these channel is required, the other channel can be used for other purposes.

The TCC can detect capture overflow of the input capture channels: When a new capture event is detected while the Match or Capture Channel x Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.MCn) is still set, the new time-stamp will not be stored, and the INTFLAG.ERR flag will be set.

Note: When up-counting (with the Counter Direction bit in the Control B Clear or Set registers (CTRLBCLR/SET.DIR) is set), counter values lower than 1 cannot be captured in Capture Minimum mode (when the Recoverable Fault n Capture Action bit field in the Fault Control x register (FCTRLx.CAPTURE) is set to CAPTMIN). To capture the full range, including value 0, the TCC must be configured in down-counting mode (CTRLBCLR/SET.DIR is cleared).

Note: In dual-slope PWM operation, when TOP is lower than MAX/2, the MSB of CCn captures the CTRLBCLR/SET.DIR state to identify the ramp on which the capture has been performed. For rising ramps, the MSB of the CCn.CC bit field is zero, for falling ramps the MSb of the CCn.CC bit field is 1.

23.4.3. Additional Features

23.4.3.1. One-Shot Operation

When one-shot is enabled, the counter automatically stops on the next counter overflow or underflow condition. When the counter is stopped, the Stop bit in the Status register (STATUS.STOP) is set, and the waveform outputs are set to the value defined by the Non-Recoverable State x Output Enable and Non-Recoverable State x Output Value bits in the Driver Control register (DRVCTRL.NREx and DRVCTRL.NRVx).

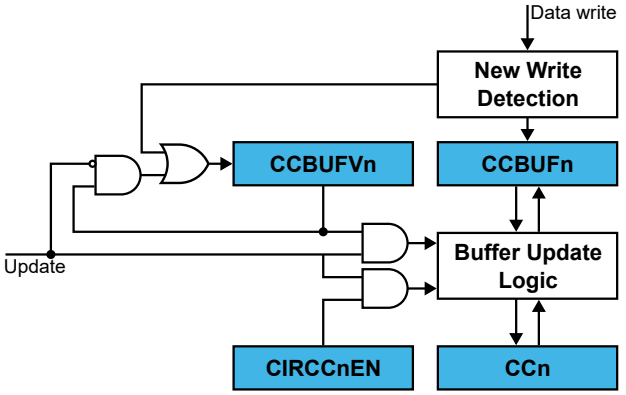
One-shot operation can be enabled by writing a '1' to the One-Shot bit in the Control B Set register (CTRLBSET.ONESHOT) and disabled by writing a '1' to the One-Shot bit in the Control B Clear register

(CTRLBCLR.ONESHOT). When enabled, the TCC will count until an overflow or underflow occurs and then stop counting. The one-shot operation can be restarted by a retrigger software command, a retrigger event, or a start event. When the counter restarts its operation, the STATUS.STOP bit is automatically cleared.

23.4.3.2. Circular Buffer

The Period register (PER) and the Compare/Capture Channel registers (CCn) support circular buffer operation. When circular buffer operation is enabled, the PER or CCn values are copied into the corresponding buffer registers at each update condition. Circular buffering is dedicated to RAMP2, RAMP2A, and DSBOTH operations.

Figure 23-18. Circular Buffer on Channel 0



23.4.3.3. Ramp Operations

Three ramp operation modes are supported. All of them require the timer/counter to be running in single-slope PWM generation. The ramp mode is selected by writing to the Ramp Operation bit field in the Waveform register (WAVE.RAMP).

RAMP1 Operation

This is the default PWM operation, described in Single-Slope PWM Generation.

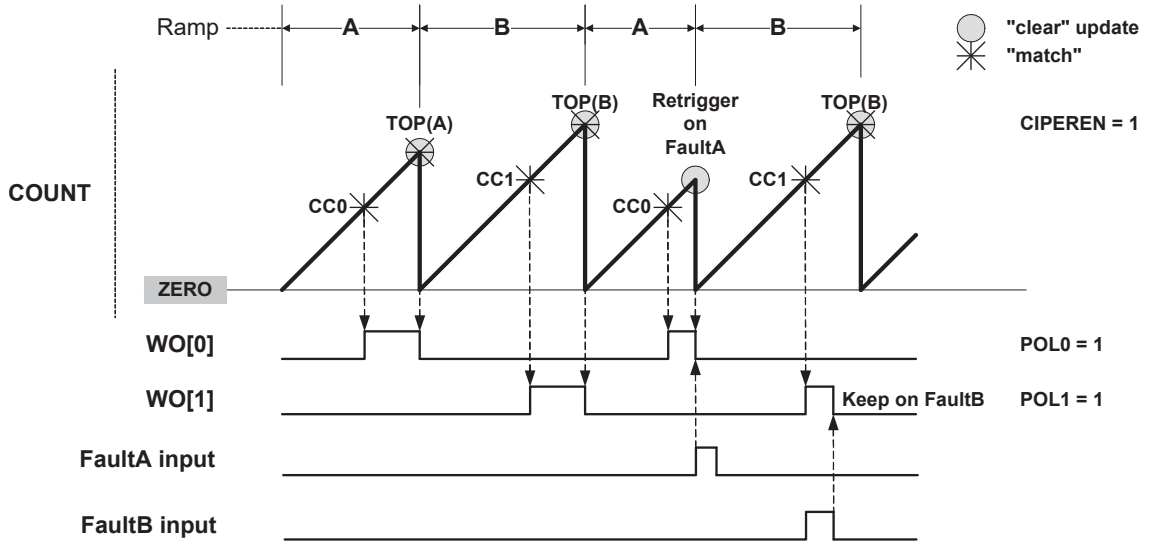
RAMP2 Operation

These operation modes are dedicated to power factor correction (PFC), half-bridge and push-pull SMPS topologies, where two consecutive timer/counter cycles are interleaved. In cycle A, the odd channel output is disabled, and in cycle B, the even channel output is disabled. The ramp index changes after each update, but it can be modified by software using the Ramp Index Command bit field in the Control B Set register (CTRLBSET.IDXCMD). All RAMP2 operations suport only up-counting mode (when the Counter Direction bit in the Control B Clear register (CTRLBCLR.DIR) is cleared).

Standard RAMP2 (RAMP2) Operation

Ramp A and B periods are controlled by the Period register (PER). The PER value can be different on each ramp if the Circular Period Enable bit in the Waveform register (WAVE.CIPEREN) is set. This mode uses a two-channel TCC to generate two output signals.

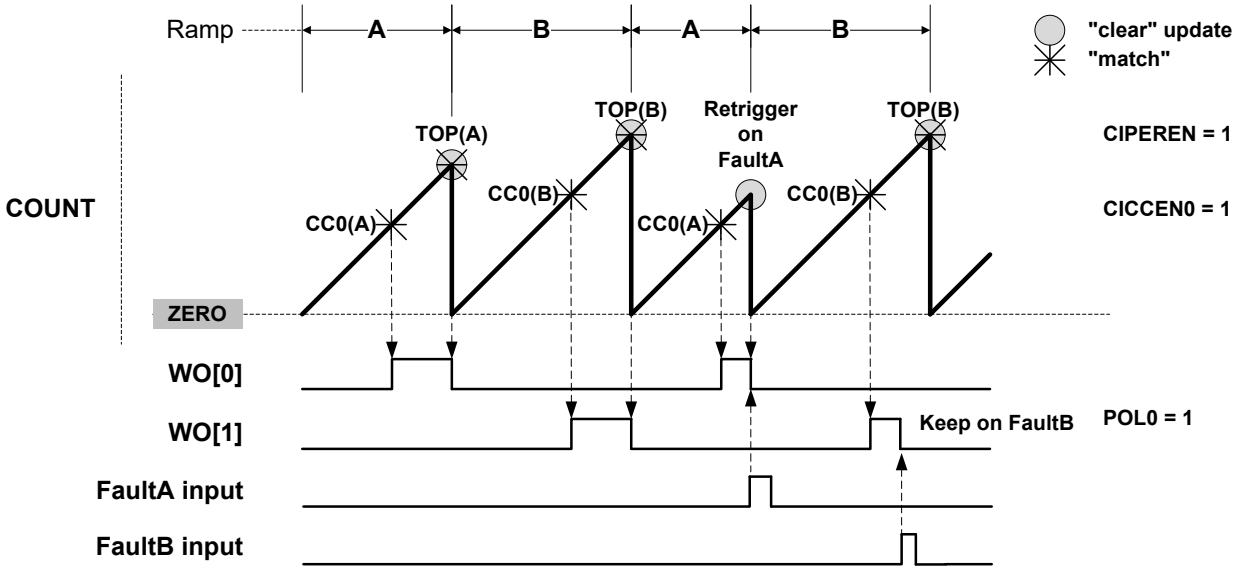
Figure 23-19. RAMP2 Standard Operation



Alternate RAMP2 (RAMP2A) Operation

Alternate RAMP2 operation is similar to RAMP2, but the Compare/Capture Value in the Compare/Capture Channel 0 register (CC0.CC) controls both WO[0] and WO[1] waveforms when the corresponding circular buffer option is enabled (WAVE.CIPEREN is set). The waveform polarity is the same on both outputs. Channel 1 can be used in capture mode.

Figure 23-20. RAMP2 Alternate Operation



Critical RAMP2 (RAMP2C) Operation

Critical RAMP2 operation provides a way to meet RAMP2 operation requirements without the update constraint associated with the use of circular buffers. In this mode, CC0.CC controls the period of ramp A, and PER.PER controls the period of ramp B. When using more than two channels, the WO[0] output is controlled by CC2.CC (HIGH) and CC0.CC (LOW). With two channels, a pulse on WO[0] will last the entire period of ramp A if the Channel Polarity 0 bit in the Waveform register (WAVE.POL0) is cleared.

Figure 23-21. RAMP2 Critical Operation With More Than 2 Channels

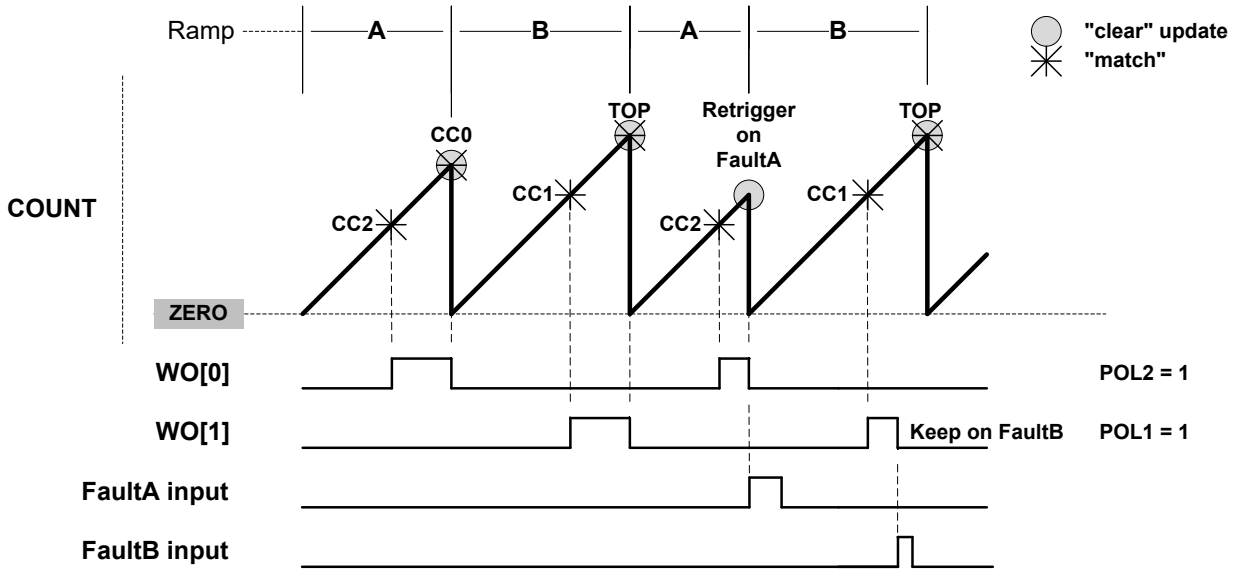
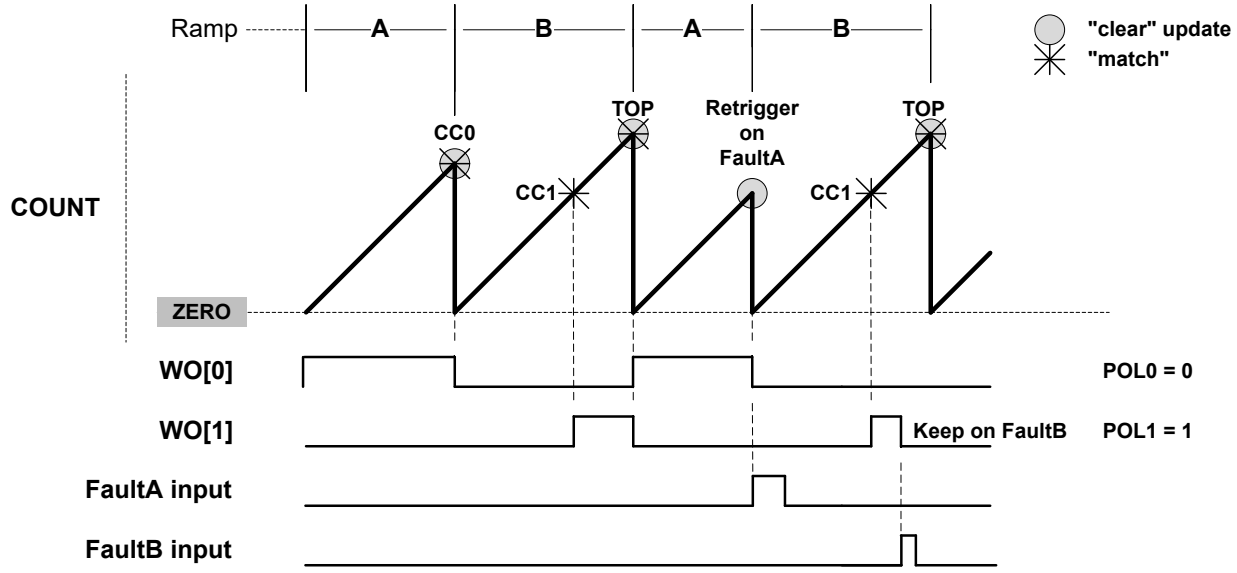


Figure 23-22. RAMP2 Critical Operation With 2 Channels



23.4.3.4. Recoverable Faults

Recoverable faults can restart or halt the timer/counter. Two faults, called Fault A and Fault B, can trigger recoverable fault actions on the Compare/Capture Channel n register (CC0 and CC1) of the TCC. The compare channels' outputs can be clamped to the inactive state either for as long as the fault condition is present, or from the first valid fault condition detection on until the end of the timer/counter cycle.

Fault Inputs

The first two channel input events (TCCnMC0 and TCCnMC1) can be used as Fault A and Fault B inputs, respectively. Event system channels connected to these fault inputs must be configured as asynchronous. The TCC must operate in a PWM mode.

Fault Filtering

There are three filters available for each input, Fault A and Fault B. They are configured in the corresponding Recoverable Fault Control x registers (FCTRLA and FCTRLB). The three filters can either be used independently or in any combination.

Input Filtering

By default, event detection is asynchronous. When the event occurs, the fault system will immediately and asynchronously perform the selected fault action on the compare channel output, even in device power modes where the clock is not available. To avoid false fault detection on external events (e.g., due to a glitch on an I/O port), a digital filter can be enabled and configured by the Recoverable Fault Filter Value bit field in the Recoverable Fault Control x register (FCTRLx.FILTERVAL). If the event width is less than FCTRLx.FILTERVAL (in clock cycles), the event will be discarded. A valid event will be delayed by FCTRLx.FILTERVAL clock cycles.

Fault Blanking

This ignores any fault input for a certain time immediately after a selected waveform output edge. This can be used to prevent false fault triggering due to signal bouncing, as shown in the figure below. Blanking can be enabled by writing an edge triggering configuration to the Recoverable Fault n Blanking Operation bit field (FCTRLx.BLANK). The desired duration of the blanking must be written to the Recoverable Fault x Blanking Value bit field (FCTRLx.BLANKVAL).
The blanking time, t_b is calculated by

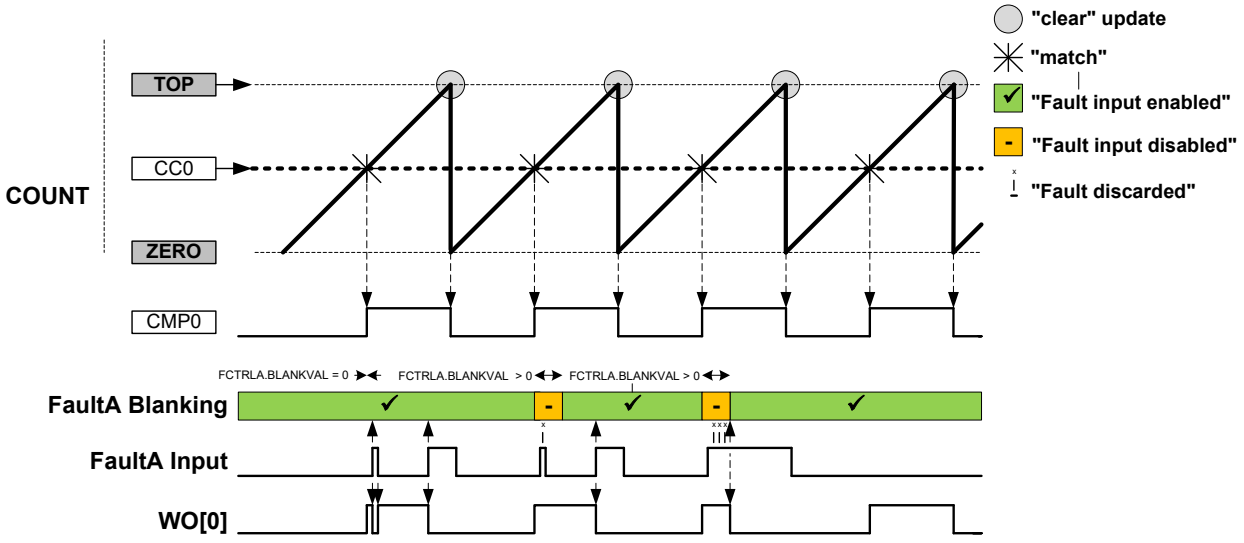
$$t_b = \frac{1 + \text{BLANKVAL}}{f_{\text{GCLK\_TCCn\_PRESC}}}$$

Where,

$f_{\text{GCLK\_TCCn\_PRESC}}$ is the frequency of the prescaled peripheral clock, $f_{\text{GCLK\_TCCn}}$. The prescaler is enabled by writing '1' to the Recoverable Fault n Blanking Value Prescaler bit (FCTRLx.BLANKPRESC). When disabled, $f_{\text{GCLK\_TCCn\_PRESC}}=f_{\text{GCLK\_TCCn}}$. When enabled, $f_{\text{GCLK\_TCCn\_PRESC}}=f_{\text{GCLK\_TCCn}}/64$.

The maximum blanking time (FCTRLx.BLANKVAL = 255) at $f_{\text{GCLK\_TCCn}}$=96 MHz is 2.67 µs (no prescaler) or 170 µs (with prescaling). For $f_{\text{GCLK\_TCCn}}$=1 MHz, the maximum blanking time is either 170 µs (no prescaling) or 10.9 ms (with prescaling enabled).

Figure 23-23. Fault Blanking in RAMP1 Operation with Inverted Polarity



Fault Qualification

This is enabled by writing a '1' to the Recoverable Fault x Qualification bit (FCTRLx.QUAL). When recoverable fault qualification is enabled (FCTRLx.QUAL is set),

the fault input is disabled whenever the corresponding channel output has an inactive level, as shown in the figures below.

Figure 23-24. Fault Qualification in RAMP1 Operation

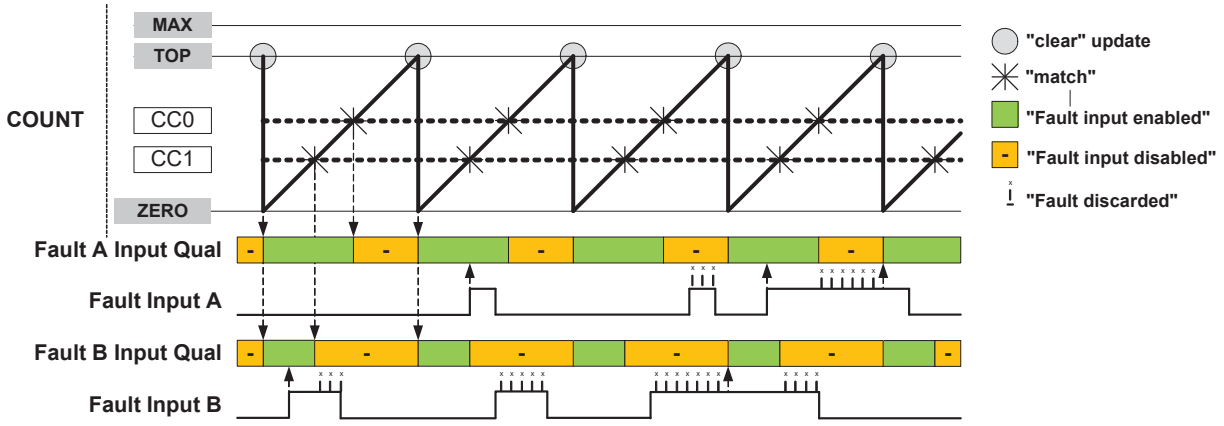
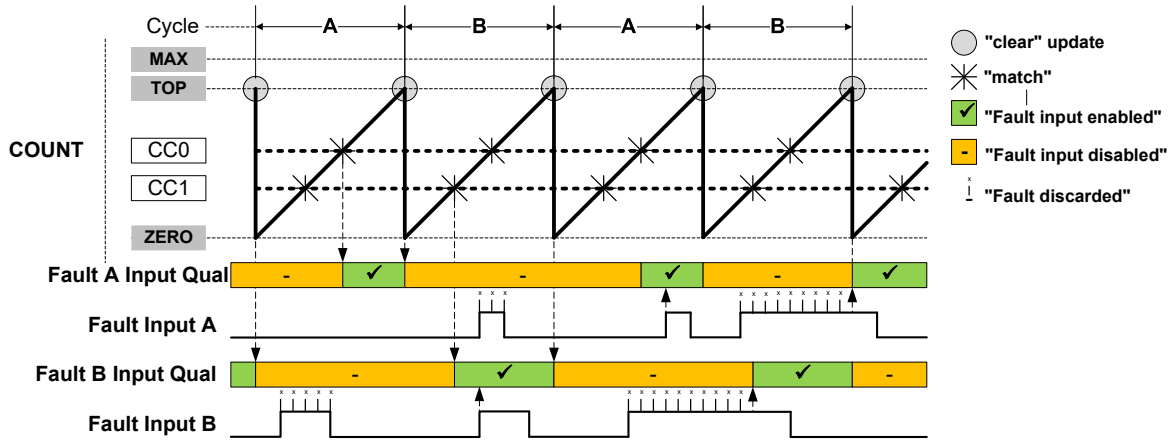


Figure 23-25. Fault Qualification in RAMP2 Operation with Inverted Polarity

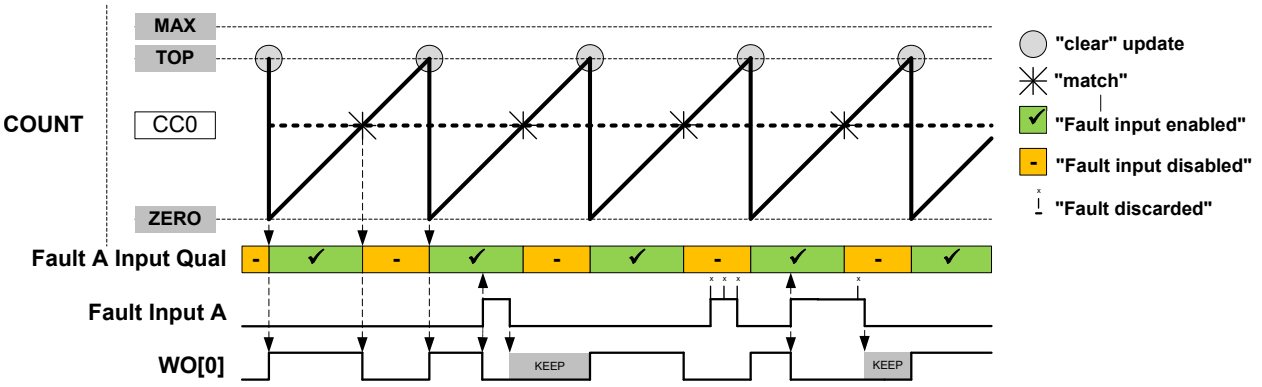


Fault Actions

Different fault actions can be configured individually for Fault A and Fault B. Most fault actions are not mutually exclusive; therefore, two or more actions can be enabled at the same time to achieve a result that is a combination of fault actions.

Keep Action This is enabled by writing a '1' to the Recoverable Fault x Keep bit (FCTRLx.KEEP) . When enabled, the corresponding channel output will be clamped to zero as long as the fault condition is present. The clamp will be released at the start of the first cycle after the fault condition is no longer present, see next Figure.

Figure 23-26. Waveform Generation with Fault Qualification and Keep Action



Restart Action

This is enabled by writing to '1' the Recoverable Fault x Restart bit (FCTRLx.RESTART). When enabled, the timer/counter will be restarted as soon as the corresponding fault condition is present. The ongoing cycle is stopped, and the timer/counter starts a new cycle (see the following figures). In Ramp 1 mode, when the new cycle starts, the compare outputs will be clamped to the inactive level as long as the fault condition is present.

Note: For RAMP2 operation, when a new timer/counter cycle starts, the cycle index will change automatically (see the following figures). Fault A and Fault B are qualified only during the cycle A and cycle B, respectively: Fault A is disabled during cycle B, and Fault B is disabled during cycle A.

Figure 23-27. Waveform Generation in RAMP1 mode with Restart Action

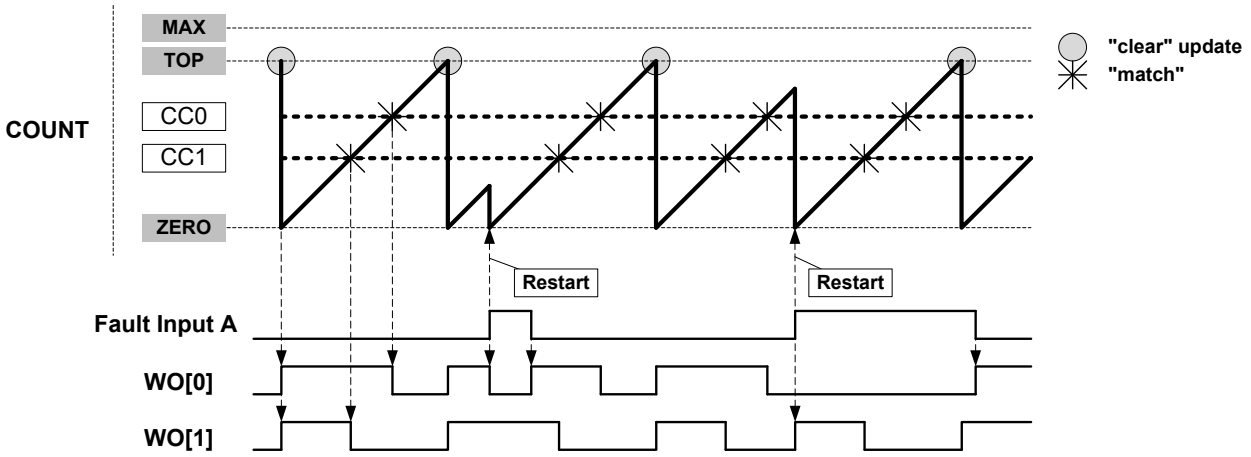
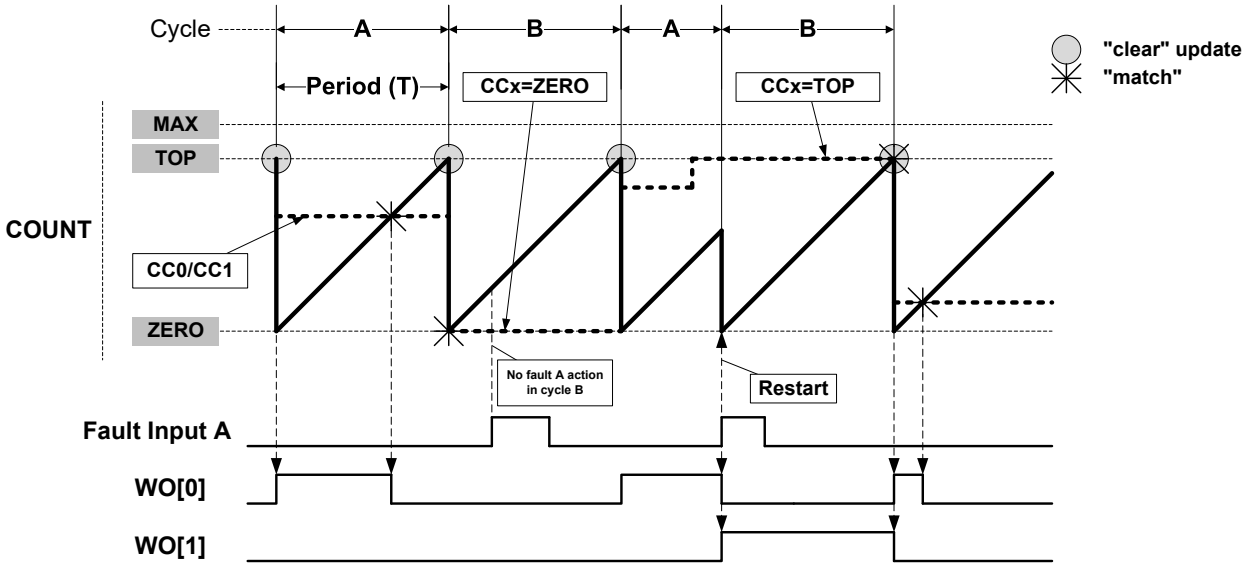


Figure 23-28. Waveform Generation in RAMP2 mode with Restart Action



Capture Action

Several capture actions can be selected by writing to the Recoverable Fault x Capture Action bit field (FCTRLx.CAPTURE). When one of the capture operations is selected, the counter value is captured when the fault occurs. These capture operations are available:

- CAPT - The equivalent to a standard capture operation. For further details, refer to the Capture Operations Section
- CAPTMIN - Captures the minimum time stamped value: On each new local minimum captured value, an event or interrupt is issued
- CAPTMAX - Captures the maximum time stamped value: On each new local maximum captured value, an event or interrupt (IT) is issued. For additional information, refer to the following figure, Capture Action “CAPTMAX”.
- LOCMIN - Notifies by event or interrupt when a local minimum captured value is detected
- LOCMAX - Notifies by event or interrupt when a local maximum captured value is detected
- DERIV0 - Notifies by event or interrupt when a local extreme captured value is detected, For more information, refer to the following figure Capture Action “DERIV0”

CCn.CC Content:

In CAPTMIN and CAPTMAX operations, CCn.CC stores the respective extreme captured values, For more information, refer to the following figure, Capture Action “CAPTMAX”. In LOCMIN, LOCMAX or DERIV0 operations, CCn.CC reflects the counter value at fault time, For more information, refer to the following figure, Capture Action “DERIV0”.

Before enabling CAPTMIN or CAPTMAX mode of capture, the user must initialize the corresponding CCn.CC bit field to a value to other than ZERO (for CAPTMIN) or TOP (for CAPTMAX). If the initial value of the CCn.CC bit field initial value is ZERO (for CAPTMIN) or TOP (for CAPTMAX), no captures will be performed using the corresponding channel.

When using advanced capture functions such as CAPTMIN, CAPTMAX, LOCMIN, LOCMAX and DERIV0, standard capture functions (CAPT) must be assigned to the lower CCn channels when used. See the example below.

Example: CC[0] = CAPT, CC[1] = CAPT, CC[2] = CAPTMIN, CC[3] = CAPTMAX.

MCx Behaviour:

In LOCMIN and LOCMAX operation, capture is performed on each capture event. The Match or Capture Channel n Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.MCx) is set only when the captured value is greater than or equal to (for LOCMIN) or less than or equal to (for LOCMAX) the previous captured value. Thus, the interrupt flag is set when a new relative local minimum (for CAPTMIN) or maximum (for CAPTMAX) value has been detected. DERIV0 is equivalent to a logical OR function of (LOCMIN, LOCMAX).

In CAPT operation, capture is performed on each capture event. The INTFLAG.MCx interrupt flag is set on each new capture.

In CAPTMIN and CAPTMAX operation, capture is performed only when, at the time of the capture event, the counter value is lower (for CAPTMIN) or higher (for CAPMAX) than the last captured value. The INTFLAG.MCx interrupt flag is set only at the time of the capture event, the counter value is higher or equal (for CAPTMIN) or lower or equal (for CAPTMAX) to the value captured on the previous event. Thus, the interrupt flag is set when a new absolute local minimum (for CAPTMIN) or maximum (for CAPTMAX) value has been detected.

Interrupt Generation

In CAPT mode, an interrupt is generated on each filtered Fault n and on each dedicated CCn channel capture counter value. In other modes, an interrupt is generated only on an extreme captured value.

Figure 23-29. Capture Action “CAPTMAX”

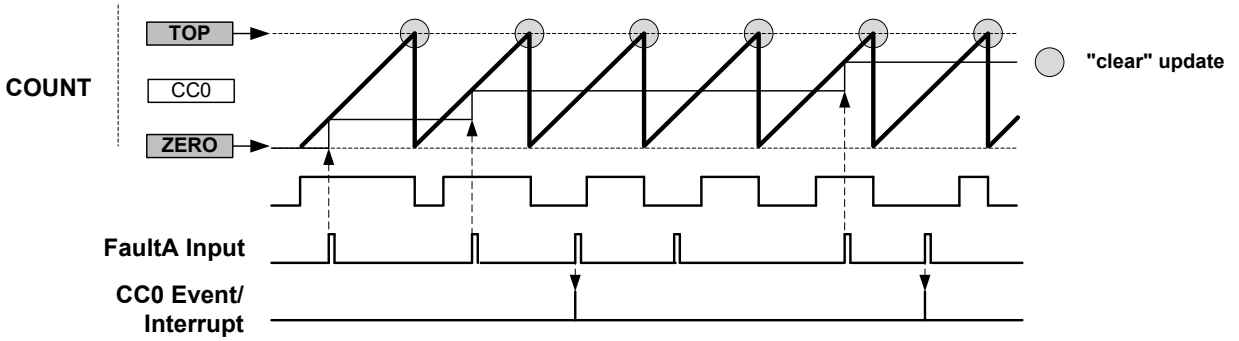
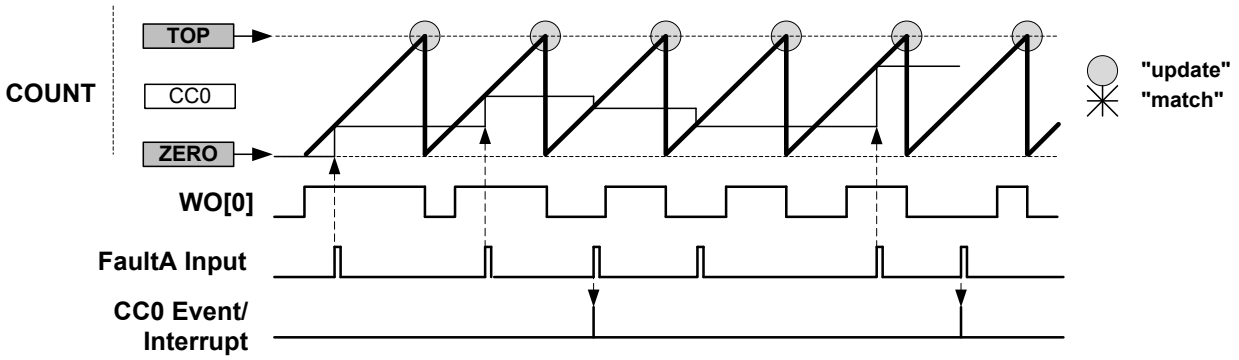


Figure 23-30. Capture Action “DERIV0”



Hardware Halt Action This is configured by writing 0x1 to the Recoverable Fault x Halt Operation bit field (FCTRLx.HALT). When enabled, the timer/counter is halted, and the cycle is extended for as long as the corresponding fault is present.

The next figure ('Waveform Generation with Halt and Restart Actions') shows an example where both the restart action and hardware halt action are enabled for Fault

A. The Compare channel 0 output is clamped to the inactive level as long as the timer/counter is halted. The timer/counter resumes counting operation as soon as the fault condition is no longer present. Since the restart action is enabled in this example, the timer/counter is restarted after the fault condition is no longer present.

The following figure ('Waveform Generation with Fault Qualification, Halt, and Restart Actions') shows a similar example, but with fault qualification additionally enabled. In this case, counting is resumed after the fault condition is no longer present.

Note that in RAMP2 and RAMP2A operations, when a new timer/counter cycle starts, the cycle index will automatically change.

Figure 23-31. Waveform Generation with Halt and Restart Actions

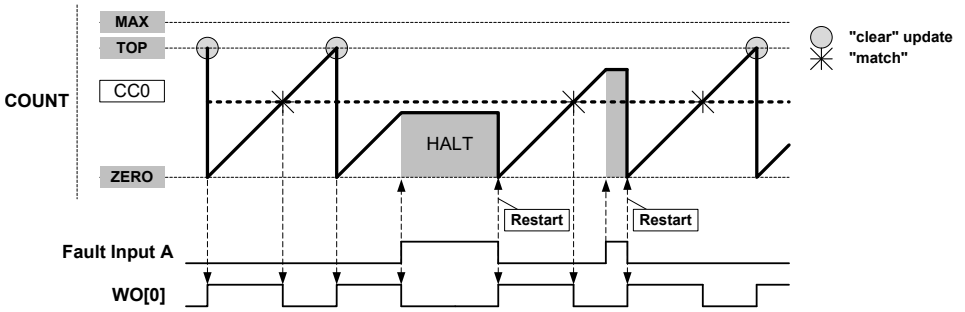
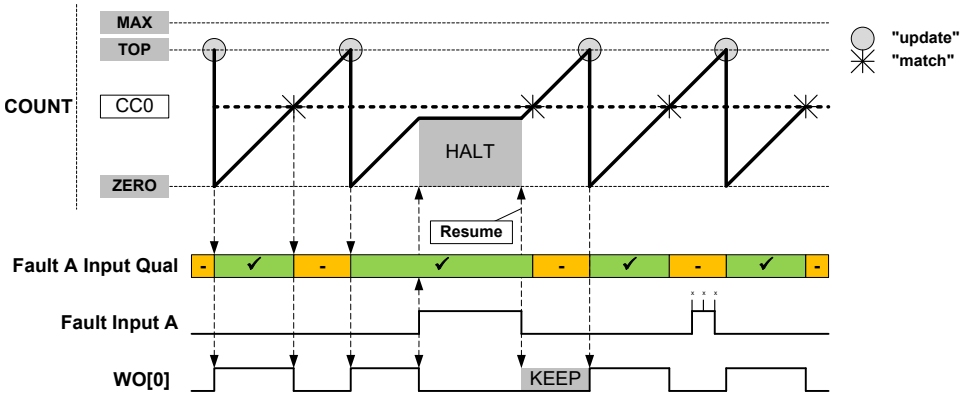


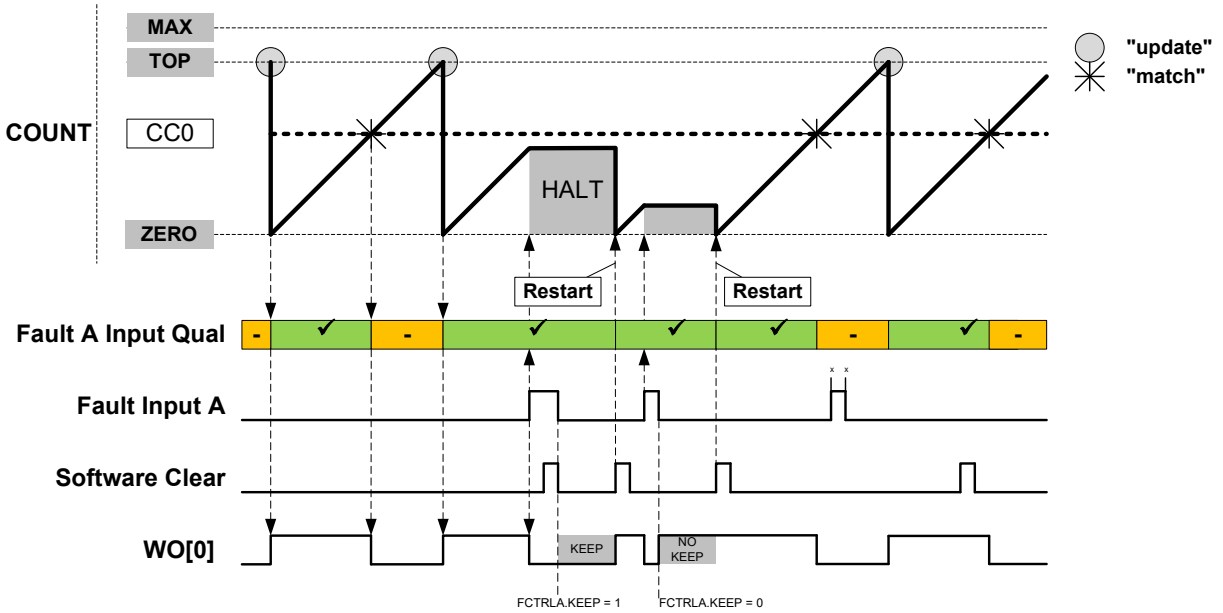
Figure 23-32. Waveform Generation with Fault Qualification, Halt, and Restart Actions



Software Halt Action

This is configured by writing 0x2 to the FCTRLx.HALT bit field. The software halt action is similar to the hardware halt action, but to restart the timer/counter, the corresponding fault condition must no longer be present, and the corresponding Non-recoverable Fault x State bit in the Status register (STATUS.FAULTx) must be cleared by software.

Figure 23-33. Waveform Generation with Software Halt, Fault Qualification, Keep and Restart Actions



23.4.3.5. Non-Recoverable Faults

The non-recoverable fault action will force all compare outputs to a predefined level set in the Non-Recoverable State x Output Enable and Non-Recoverable State x Output Value bit fields in the Driver Control register (DRVCTRL.NRE and DRVCTRL.NRV). The non-recoverable fault event input actions are enabled in the Timer/Counter Event Input n Action bit fields in the Event Control register (EVCTRL.EVACT0 and EVCTRL.EVACT1).

To avoid false fault detection on external events (e.g., a glitch on an I/O port), a digital filter can be enabled using the Non-Recoverable Fault Input n Filter Value bits in the Driver Control register (DRVCTRL.FILTERVAL0 and DRVCTRL.FILTERVAL1). As a result, event detection is synchronous, and event action is delayed by the selected digital filter value clock cycles.

When the Fault Detection on Debug Break Detection bit in the Debug Control register (DGBCTRL.FDDBD) is written to '1', a non-recoverable Debug Faults State (DFS) and an interrupt are generated when the system enters debug operation.

In RAMP2, RAMP2A, or DSBOTH operation, when the Lock Update bit in the Control B register (CTRLBSET.LUPD) is set and the ramp index or counter direction changes, a non-recoverable Update Fault State (UFS) and the respective interrupt are generated.

23.4.3.6. Time-stamp Capture

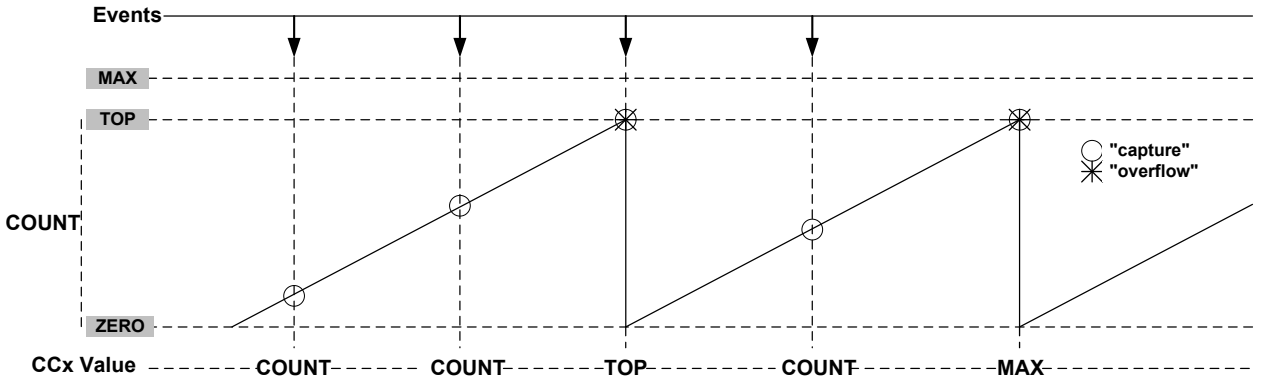
This feature is enabled when the Capture Time-stamp (STAMP) Event Action value is written to the Timer/Counter Event Input 0 Action bit field in the Event Control register (EVCTRL.EVACT0). The counter TOP value must be smaller than MAX.

When a capture event is detected, the COUNT value is copied into the corresponding Compare/Capture Channel n register (CCn). In case of an overflow, the MAX value is copied into the corresponding CCn register.

When a valid captured value is present in the capture channel register, the corresponding Match or Capture Channel n Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.MCn) is set.

The timer/counter can detect capture overflow of the input capture channels: When a new capture event is detected while INTFLAG.MCn is still set, the new time-stamp will not be stored, and the Error Interrupt Flag in the Interrupt Flag Status and Clear register (INTFLAG.ERR) will be set.

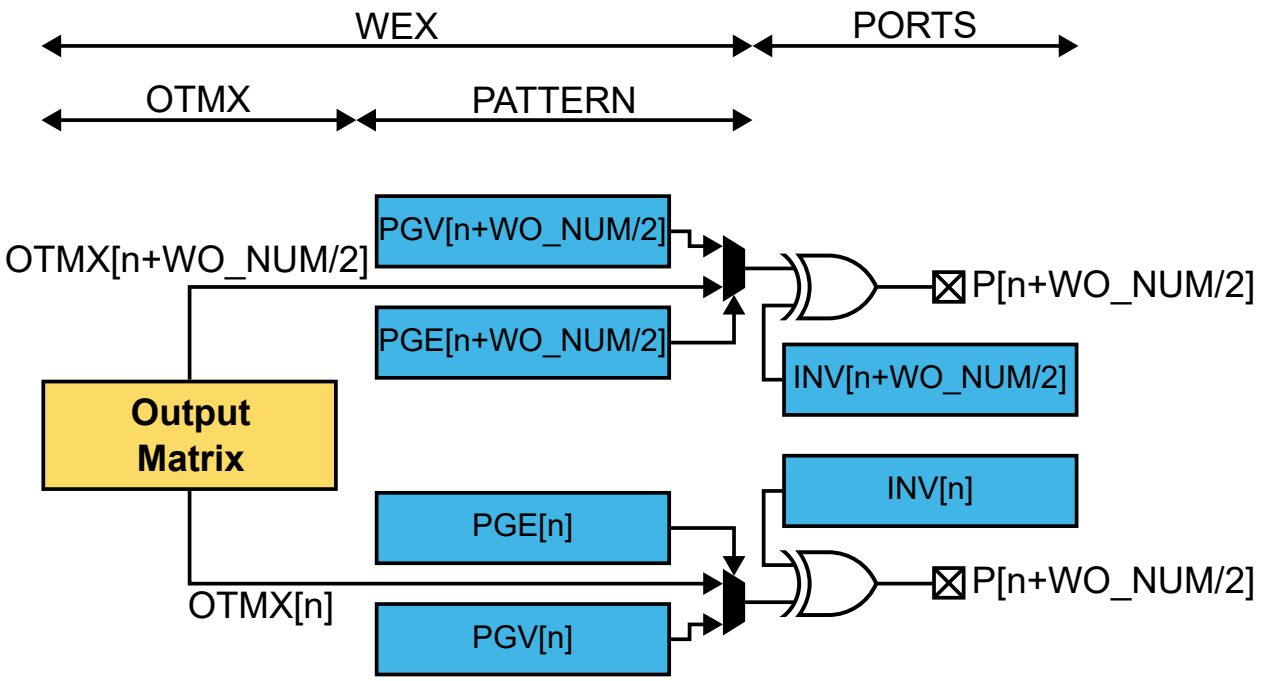
Figure 23-34. Time-stamp



23.4.3.7. Waveform Extension

The “Waveform Extension Stage Details” figure below shows a schematic diagram of the actions of the two optional units that follow the recoverable fault stage on a port pin pair: Output Matrix (OTMX) and Pattern Generation.

Figure 23-35. Waveform Extension Stage Details



The output matrix (OTMX) unit distributes Compare Channels (CCn), according to the selectable configurations shown in the table below.

Table 23-5. Output Matrix Channel Pin Routing Configuration

Value	OTMX[x]			
0x0	CC3	CC2	CC1	CC0
0x1	CC1	CC0	CC1	CC0
0x2	CC0	CC0	CC0	CC0
0x3	CC1	CC1	CC1	CC0

The following comments provide an explanation for each of the four Output Matrix Chanel Pin Routing Configurations. The configuration can be written to the Output Matrix bit field in the Waveform Extension Control register (WEXCTRL.OTMX).

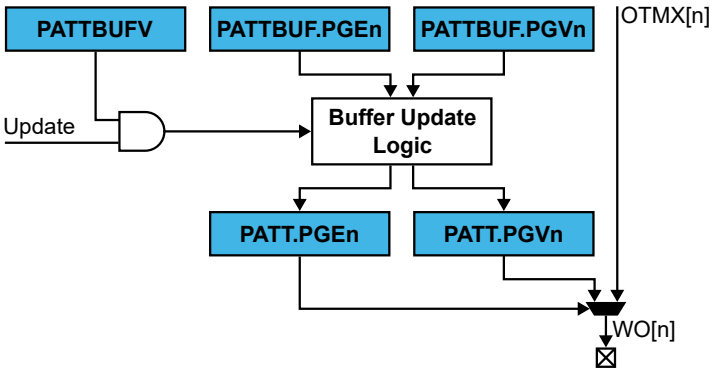
- Configuration 0x0 is the default configuration. The channel location is the default, and channels are distributed to outputs modulo the number of channels. Channel 0 is routed to the Output matrix output OTMX[0], and Channel 1 to OTMX[1]. If there are more outputs than channels, then channel 0 is duplicated to the Output matrix output OTMX[CC_NUM], channel 1 to OTMX[CC_NUM+1], and so on.
- Configuration 0x1 distributes the channels to outputs modulo half the number of channels. This assigns twice as many output locations to the lower channels compared to the default configuration. This can be used, for example, to control the four transistors of a full bridge using only two compare channels.
  With pattern generation, some of these four outputs can be overwritten by a constant level, enabling flexible drive of a full bridge in all quadrant configurations.
- Configuration 0x2 distributes Compare Channel 0 (CC0) to all port pins. With pattern generation, this configuration can be used to control a stepper motor.
- Configuration 0x3 distributes the Compare Channel 0 (CC0) to the first output, and the Compare Channel 1 (CC1) to all other outputs. Together with pattern generation and the fault extension, this configuration can control up to seven LED strings with a boost stage.

Table 23-6. Example: four compare channels on four outputs

Value	OTMX[3]	OTMX[2]	OTMX[1]	OTMX[0]
0x0	CC3	CC2	CC1	CC0
0x1	CC1	CC0	CC1	CC0
0x2	CC0	CC0	CC0	CC0
0x3	CC1	CC1	CC1	CC0

The pattern generator unit produces a synchronized bit pattern across the port pins to which it is connected. The pattern generation features are primarily intended for handling the commutation sequence in brushless DC motors (BLDC), stepper motors, and full bridge control. See also the “Pattern Generator Block Diagram” figure below.

Figure 23-36. Pattern Generator Block Diagram



As with other double-buffered timer/counter registers, the register update is synchronized to the UPDATE condition set by the timer/counter waveform generation operation. If synchronization is not required by the application, the software can simply access the bits directly in the Pattern Generation Output Enable and Pattern Generation Output Value fields in the Pattern register (PATT.PGE and PATT.PGV).

23.4.4. Sleep Mode Operation

The TCC can be configured to operate in any sleep mode. To run in standby the Run in Standby bit in the Control A register (CTRLA.RUNSTDBY) must be set. In any sleep mode, the TCC can wake up the device using interrupts or perform actions through the Event System.

When the device is in Standby sleep mode, the DMA is not able to write to the following registers: CTRLB, STATUS, COUNT, PATT, WAVE, PER, PERBUF, CC, and CCBUF. To write these registers using DMA, the device must be in Active mode or Idle sleep mode.

23.4.5. Debug Operation

When the CPU is halted in debug mode, the TCC will halt normal operation. Setting the Debug Running State in the Debug Control register (DBGCTRL.DBGRUN) forces the TCC to continue operating when the CPU is halted in debug mode.

23.5. Dependencies

23.5.1. I/O Lines

Using the TCC's I/O lines requires the I/O pins to be configured. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

23.5.2. Power Management

The TCC will continue to operate in any sleep mode where the selected source clock is running. TCC 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 the Power Manager chapter for details on the different sleep modes.

23.5.3. Clocks

The TCC bus clock (CLK_TCCn_APB) can be enabled and disabled in the Main Clock, and the default state of CLK_TCCn_APB can be found in the Peripheral Clock Masking section in the *MCLK – Main Clock* chapter.

A generic clock (GCLK_TCCn) is required to clock the TCC. This clock must be configured and enabled in the generic clock controller before using the TCC. Refer to the *GCLK – Generic Clock Controller* chapter for details.

This generic clock is asynchronous to the bus clock (CLK_TCCn_APB). Due to this asynchronicity, writes to certain registers will require synchronization between the clock domains.

23.5.4. DMA

The TCC generates the following DMA requests:

- Counter overflow (OVF):
  - The request is set if the Ones-shot Trigger mode in the control A register (CTRLA.DMAOS) is written to '0'. In this case, the TCC generates a DMA request on each cycle when an update condition (overflow, underflow or retrigger) is detected
  - The request is set when an update condition (overflow, underflow or retrigger) is detected while CTRLA.DMAOS=1.In this case, the TCC generates a DMA trigger on the cycle following the DMA One-Shot Command written to the Control B register (CTRLBSET.CMD=DMAOS)
  - In both cases, the request is cleared by hardware upon DMA acknowledgment
- Channel Match (MCn):
  - The request is set on a compare match if CTRLA.DMAOS = 0
  - When CTRLA.DMAOS = 1, the DMA requests are not generated
  - The request is cleared by hardware on DMA acknowledge

- Channel Capture (MCn):
  - The request is set when valid data is present in the CCn register, and cleared once the CCn register is read
  - In this operation mode, the CTRLA.DMAOS bit value is ignored

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the TCC's DMA requests. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

DMA Operation with Circular Buffer

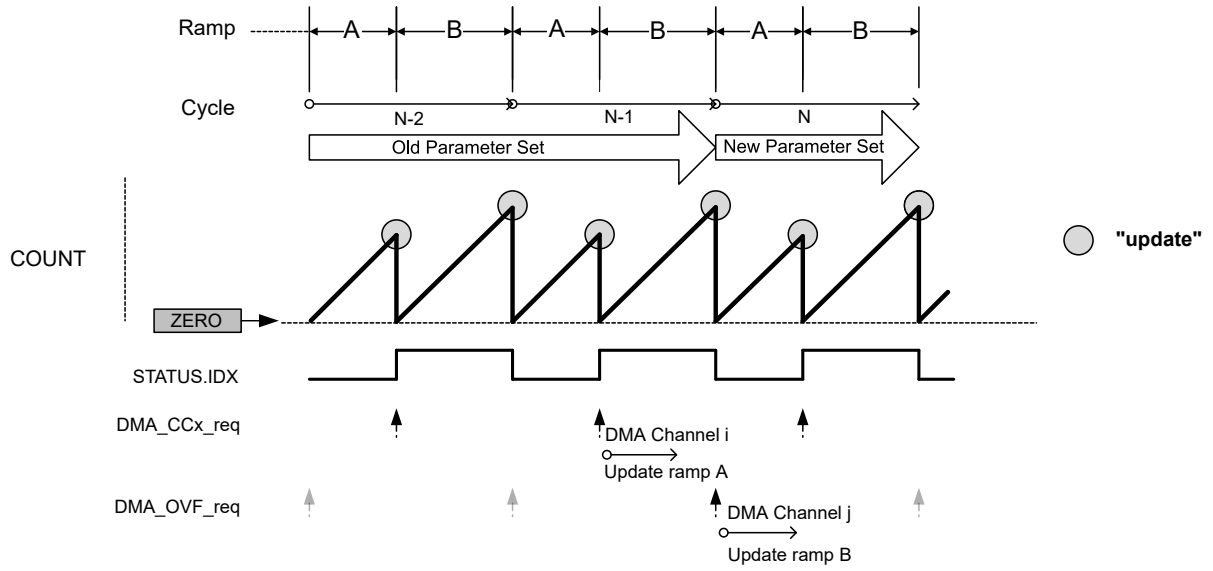
When circular buffer operation is enabled, the buffer registers must be written in the correct order and synchronized to the update times of the timer. The DMA triggers of the TCC provide a way to ensure a safe and correct update of circular buffers.
Note: Circular buffers are intended to be used only with RAMP2, RAMP2A and DSBOTH operations.

DMA Operation with Circular Buffer in RAMP2 and RAMP2A Mode
When a CCn channel is selected as a circular buffer, the related DMA request is not set on compare match detection, but on start of ramp B.

If at least one circular buffer is enabled, the DMA overflow request is triggered at the start of ramp A, with the actual DMA transfer occurring on the previous ramp B (DMA acknowledgment).

The update of all circular buffer values for ramp A can be performed through a DMA channel triggered on a MC trigger. The update of all circular buffer values for ramp B can be done through a second DMA channel triggered by the overflow DMA request.

Figure 23-37. DMA Triggers in RAMP and RAMP2 Operation Mode and Circular Buffer Enabled

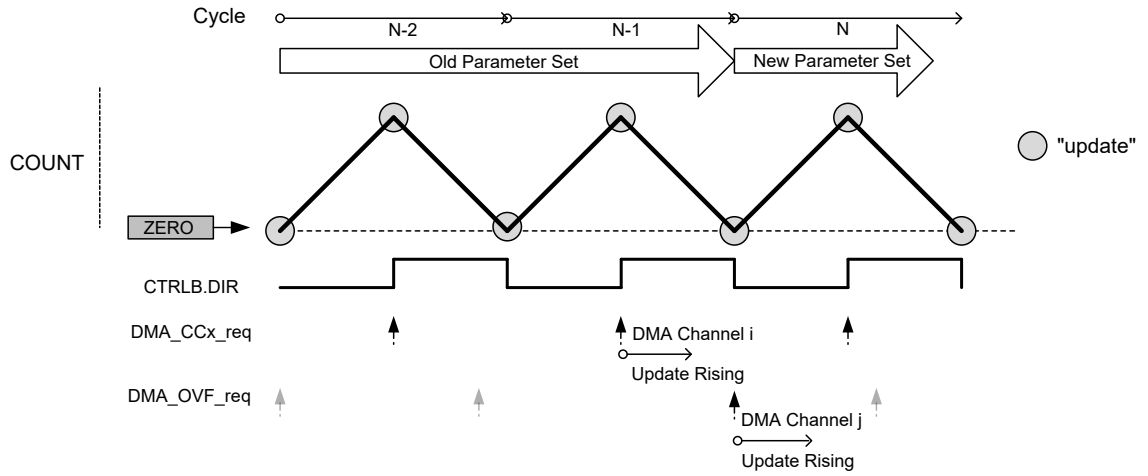


DMA Operation with Circular Buffer in DSBOTH Mode
When a CC channel is selected as a circular buffer, the related DMA request is not set on compare match detection, but on the start of the down-counting phase.

If at least one circular buffer is enabled, the DMA overflow request is triggered at the start of up-counting phase with an actual DMA transfer occurring during the previous down-counting phase (DMA acknowledgment).

When up-counting, all circular buffer values can be updated through a DMA channel triggered by an MC trigger. When down-counting, all circular buffer values can be updated through a second DMA channel, triggered by the OVF DMA request.

Figure 23-38. DMA Triggers in DSBOTH Operation Mode and Circular Buffer Enabled



23.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use TCC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 23-7. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
TCCn	OVF	An overflow condition occurs	
	TRG	A counter retrigger occurs	
	CNT	A counter event occurs	
	ERR	A new capture occurs on a channel when the corresponding Match or Capture Channel x interrupt flag = '1'	
	UFS	The RAMP index changes and the Lock Update bit is set (CTRLBSET.LUPD = '1')	
	DFS	A Debug Fault State occurs	
	FAULTx	A Recoverable Fault x occurs	
	FAULTn	A Non-Recoverable Fault n occurs	
	MCn	A match with the compare condition, or when CCn register contains a valid capture value	

23.5.6. Events

The TCC can generate the following events:

Table 23-8. TCC Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCCn	OVF	Overflow/Underflow	Pulse	GCLK_TCCn	One GCLK_TCCn period
	TRG	Trigger	Pulse	GCLK_TCCn	One GCLK_TCCn period
	CNT	Counter	Pulse	GCLK_TCCn	One GCLK_TCCn period
	MC_n	Compare Match or Capture on compare/capture channel n	Pulse	GCLK_TCCn	One GCLK_TCCn period

Writing a '1' to an Event Output Enable bit in the Event Control register (EVCTRL.xxEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The TCC can connect to the following events:

Table 23-9. Event Users

User Name		Description	Input Detection	Channel Path Type
Peripheral	Input			
TCCn	EV_n	Event Input n	Level/Edge	Asynchronous
	MC_n	Capture event n	Rising Edge	Asynchronous

Writing a '1' to an Event Input Enable bit in the Event Control register (EVCTRL.xxEI) enables the corresponding action on an input event. Writing a '0' to this bit disables the corresponding action on an input event.

Note: EVn input detection depends on the configuration of the Timer/Counter Event n Invert Enable bit in the Event Control register (EVCTRL.TCINVn) and the Timer/Counter Event Input n Action bit fields in the Event Control register (EVCTRL.EVACTn).

The event users can trigger the following actions:

Table 23-10. Event Actions

Event Input	Event Action	Description
EV_0	RETRIGGER	Counter retrigger
	COUNTEV	Count on event (increment or decrement, depending on counter direction)
	START	Counter start - Start counting on the event rising edge. Further events will not restart the counter; the counter will keep on counting using prescaled GCLK_TCCn, until it reaches TOP or ZERO, depending on the direction
	INC	Counter increment on event. This will increment the counter, irrespective of the counter direction
	COUNT_ASYNC	Count during active state of an asynchronous event (increment or decrement, depending on counter direction). In this case, the counter will be incremented or decremented on each cycle of the prescaled clock, as long as the event is active
	STAMP	Capture Time-stamp (overflow)
	FAULT_ASYNC	Non-recoverable fault
EV_1	RETRIGGER	Counter retrigger
	DIR_ASYNC	Counter direction control
	STOP	Stop the counter
	DEC	Decrement the counter on event
	PWP_ASYNC	Period and pulse width capture
	FAULT_ASYNC	Non-recoverable fault
MC_n	—	Capture event
	—	Generate a recoverable or non-recoverable fault

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

Notes:

1. Counter retrigger is not supported if RAMP2 operation is used with a prescaler (CTRLA.PRESCALER != 0). RAMP2 operation can use the retrigger option only if the counter retrigger is synchronized with the next prescaler clock (CTRLA.PRESCYNC = PRESC).
2. If a retrigger event occurs exactly at the time a channel compare match occurs, the next waveform will be corrupted. To avoid this issue, use two channels to store two successive CC register values (n and n+1) and combine the related waveform outputs to provide signal redundancy.
3. See the Capture Operations and Recoverable Faults sections for more information on the Match or Capture input events (MCn).

23.5.7. Analog Connections

Not applicable.

23.6. Register Summary - TCCn

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0							ENABLE	SWRST
		15:8		ALOCK	PRESCSYNC[1:0]		RUNSTDBY	PRESCALER[2:0]		
		23:16	DMAOS							FCYCLE
		31:24					CPTEN3	CPTEN2	CPTEN1	CPTEN0
0x04	CTRLBCLR	7:0	CMD[2:0]			IDXCMD[1:0]		ONESHOT	LUPD	DIR
0x05	CTRLBSET	7:0	CMD[2:0]			IDXCMD[1:0]		ONESHOT	LUPD	DIR
0x06 ... 0x07	Reserved									
0x08	SYNCBUSY	7:0	PER	WAVE	PATT	COUNT	STATUS	CTRLB	ENABLE	SWRST
		15:8					CC3	CC2	CC1	CC0
		23:16								
		31:24								
0x0C	FCTRLA	7:0	RESTART	BLANK[1:0]		QUAL	KEEP		SRC[1:0]	
		15:8	BLANKPRESC	CAPTURE[2:0]			CHSEL[1:0]		HALT[1:0]	
		23:16	BLANKVAL[7:0]							
		31:24					FILTERVAL[3:0]			
0x10	FCTRLB	7:0	RESTART	BLANK[1:0]		QUAL	KEEP		SRC[1:0]	
		15:8	BLANKPRESC	CAPTURE[2:0]			CHSEL[1:0]		HALT[1:0]	
		23:16	BLANKVAL[7:0]							
		31:24					FILTERVAL[3:0]			
0x14	WEXCTRL	7:0							OTMX[1:0]	
		15:8								
		23:16								
		31:24								
0x18	DRVCTRL	7:0					NRE3	NRE2	NRE1	NRE0
		15:8					NRV3	NRV2	NRV1	NRV0
		23:16					INVEN3	INVEN2	INVEN1	INVEN0
		31:24	FILTERVAL1[3:0]				FILTERVAL0[3:0]			
0x1C ... 0x1D	Reserved									
0x1E	DBGCTRL	7:0						FDDBD		DBGRUN
0x1F	Reserved									
0x20	EVCTRL	7:0	CNTSEL[1:0]			EVACT1[2:0]		EVACT0[2:0]		
		15:8	TCEI1	TCEI0	TCINV1	TCINV0		CNTEO	TRGEO	OVFEO
		23:16					MCEI3	MCEI2	MCEI1	MCEI0
		31:24					MCEO3	MCEO2	MCEO1	MCEO0
0x24	INTENCLR	7:0					ERR	CNT	TRG	OVF
		15:8	FAULT1	FAULT0	FAULTB	FAULTA	DFS	UFS		
		23:16					MC3	MC2	MC1	MC0
		31:24								
0x28	INTENSET	7:0					ERR	CNT	TRG	OVF
		15:8	FAULT1	FAULT0	FAULTB	FAULTA	DFS	UFS		
		23:16					MC3	MC2	MC1	MC0
		31:24								
0x2C	INTFLAG	7:0					ERR	CNT	TRG	OVF
		15:8	FAULT1	FAULT0	FAULTB	FAULTA	DFS	UFS		
		23:16					MC3	MC2	MC1	MC0
		31:24								
0x30	STATUS	7:0	PERBUFV		PATTBUFV		DFS	UFS	IDX	STOP
		15:8	FAULT1	FAULT0	FAULTB	FAULTA	FAULT1IN	FAULT0IN	FAULTBIN	FAULTAIN
		23:16					CCBUFV3	CCBUFV2	CCBUFV1	CCBUFV0
		31:24					CMP3	CMP2	CMP1	CMP0
0x34	COUNT	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16								
		31:24								

TCCn (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x38	PATT	7:0					PGE3	PGE2	PGE1	PGE0
		15:8					PGV3	PGV2	PGV1	PGV0
0x3A ... 0x3B	Reserved									
0x3C	WAVE	7:0	CIPEREN	RAMP[2:0]				WAVEGEN[2:0]		
		15:8					CICCEN3	CICCEN2	CICCEN1	CICCEN0
		23:16					POL3	POL2	POL1	POL0
		31:24								
0x40	PER	7:0	PER[7:0]							
		15:8	PER[15:8]							
		23:16								
		31:24								
0x44	CC[0]	7:0	CC[7:0]							
		15:8	CC[15:8]							
		23:16								
		31:24								
0x48	CC[1]	7:0	CC[7:0]							
		15:8	CC[15:8]							
		23:16								
		31:24								
0x4C	CC[2]	7:0	CC[7:0]							
		15:8	CC[15:8]							
		23:16								
		31:24								
0x50	CC[3]	7:0	CC[7:0]							
		15:8	CC[15:8]							
		23:16								
		31:24								
0x54 ... 0x63	Reserved									
0x64	PATTBUF	7:0					PGEB3	PGEB2	PGEB1	PGEB0
		15:8					PGVB3	PGVB2	PGVB1	PGVB0
0x66 ... 0x6B	Reserved									
0x6C	PERBUF	7:0	PERBUF[7:0]							
		15:8	PERBUF[15:8]							
		23:16								
		31:24								
0x70	CCBUF[0]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
		23:16								
		31:24								
0x74	CCBUF[1]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
		23:16								
		31:24								
0x78	CCBUF[2]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
		23:16								
		31:24								
0x7C	CCBUF[3]	7:0	CCBUF[7:0]							
		15:8	CCBUF[15:8]							
		23:16								
		31:24								

23.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
					CPTEN3	CPTEN2	CPTEN1	CPTEN0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DMAOS							FCYCLE
Access	R/W							R/W
Reset	0							0

Bit	15	14	13	12	11	10	9	8
		ALOCK	PRESCSYNC[1:0]		RUNSTDBY	PRESCALER[2:0]		
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

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

Bits 24, 25, 26, 27 – CPTENn Capture Channel n Enable
These bits are isused to select the capture or compare operation on channel x.
Writing a '1' to CPTENn enables capture on channel x.
Writing a '0' to CPTENn disables capture on channel x.
Note: These bits are enable-protected. These bits

Bit 23 – DMAOS DMA One-Shot Trigger Mode
This bit enables the DMA One-shot Trigger Mode.
Writing a '1' to this bit will generate a DMA trigger on the TCC cycle following a TCC_CTRLBSET_CMD_DMAOS command.
Writing a '0' to this bit will generate DMA triggers on each TCC cycle.
Note: This bit is enable-protected, and is not synchronized.

Bit 16 – FCYCLE Full Cycle Enable
Writing a '1' to this bit will cause the TCC to wait for the end of the current cycle before evaluating the stop condition.
Writing a '0' to this bit will make the TCC evaluate the stop condition immediately.
Note: This bit is enable-protected, and is not synchronized.

Value	Description
0	The stop condition is evaluated immediately
1	The stop condition is evaluated at the end of the cycle

Bit 14 – ALOCK Auto Lock
Note: This bit is enable-protected, and is not synchronized.

Value	Description
0	The Lock Update bit in the Control B Clear or Set registers (CTRLBCLR/SET.LUPD) is not affected by overflow, underflow, or retrigger events
1	CTRLBCLR/SET.LUPD is set to '1' on each overflow, underflow, or retrigger event

Bits 13:12 – PRESCSYNC[1:0] Prescaler and Counter Synchronization

This bit field selects whether, on a retrigger event, the counter is cleared or reloaded on either the next GCLK_TCCn clock, or the next prescaled GCLK_TCCn clock. It is also possible to reset the prescaler on retrigger event.

Note: This bit is enable-protected, and is not synchronized.

Value	Name	Description
0x0	GCLK	Reload or reset Counter on the next GCLK
0x1	PRESC	Reload or reset Counter on the next prescaler clock
0x2	RESYNC	Reload or reset Counter on the next GCLK and reset prescaler counter
Other	—	Reserved

Bit 11 – RUNSTDBY Run in Standby

This bit is used to keep the TCC running in Standby sleep mode.

Note: This bit is enable-protected, and is not synchronized..

Value	Description
0	The TCC is halted in standby
1	The TCC continues to run in standby

Bits 10:8 – PRESCALER[2:0] Prescaler

This bit field select the Counter prescaler factor.

Note: This bit is enable-protected, and is not synchronized.

Value	Name	Description
0x0	DIV1	Prescaler: GCLK_TCC
0x1	DIV2	Prescaler: GCLK_TCC/2
0x2	DIV4	Prescaler: GCLK_TCC/4
0x3	DIV8	Prescaler: GCLK_TCC/8
0x4	DIV16	Prescaler: GCLK_TCC/16
0x5	DIV64	Prescaler: GCLK_TCC/64
0x6	DIV256	Prescaler: GCLK_TCC/256
0x7	DIV1024	Prescaler: GCLK_TCC/1024

Bit 1 – ENABLE Enable

Note: This bit is not enable-protected.

Note: This bit is write-synchronized. The ENABLE Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE) must be checked to ensure that synchronization of the ENABLE bit between the clock domains is complete.

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset

Writing a '0' to this bit has no effect.

Writing a '1' to this bit resets all registers in the TCC (except DBGCTRL) to their initial state, and the TCC will be disabled.

Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write-operation will be discarded.

Note: This bit is not enable-protected.

Note: This bit is write-synchronized. The SWRST Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.SWRST) must be checked to ensure that synchronization of the SWRST bit between the clock domains is complete.

Important:

1. After CTRLA.SWRST is written to '1', the user must poll the SYNCB.SWRST bit to verify when the reset operation is complete.
2. During a Software Reset, access to registers or bits not affected by a Software Reset are not allowed before the SYNCBUSY.SWRST bit is cleared by hardware.

Value	Description
0	There is no reset operation ongoing
1	The reset operation is ongoing

23.6.2. Control B Clear

Name: CTRLBCLR
Offset: 0x04
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized, Read-Synchronized

This register allows the user to change its value without performing a read-modify-write operation. Changes in this register will also be reflected in the Control B Set (CTRLBSET) register.
Note: This register is write-synchronized - SYNCBUSY.CTRLB must be checked to ensure that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]			IDXCMD[1:0]		ONESHOT	LUPD	DIR
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 7:5 – CMD[2:0] TCC Command
Writing `0x0` to this bit field has no effect.
Writing a value other than `0x0` to these bit field bits will clear the pending command.

Bits 4:3 – IDXCMD[1:0] Ramp Index Command
This bit field can be used to force cycle A and cycle B changes in all RAMP2 operations. On timer/counter update condition, the command is executed, the IDX flag in the STATUS register is updated, and the IDXCMD command is cleared.
Writing `0x0` to this bit field has no effect.
Writing a value other than `0x0` to this bit field will clear the pending command.

Bit 2 – ONESHOT One-Shot
This bit controls the one-shot operation of the TCC. When one-shot operation is enabled, the TCC will stop counting on the next overflow or underflow condition or upon receiving a stop command.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will disable the one-shot operation.

Value	Description
0	The TCC will update the counter value on overflow or underflow condition and continue operation
1	The TCC will stop counting on the next overflow or underflow condition

Bit 1 – LUPD Lock Update
This bit controls the update operation of the TCC buffered registers.
When the Lock Update bit is cleared, the values of the buffer registers will be copied to their non-buffered counterparts on an UPDATE condition.
This bit has no effect when input capture operation is enabled.
Note:

Writing a '0' to this bit has no effect.
Writing a '1' to this bit enables buffer registers updates on hardware UPDATE condition.

Value	Description
0	The buffer registers values *are* copied into the corresponding non-buffer registers on a hardware update condition
1	The buffer registers values *are not* copied into the corresponding non-buffer registers on a hardware update condition

Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.

Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the bit and cause the counter to count up.

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

23.6.3. Control B Set

Name: CTRLBSET
Offset: 0x05
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized, Read-Synchronized

This register allows the user to change its register value without performing a read-modify-write operation. Changes in this register will also be reflected in the Control B Clear (CTRLBCLR) register.
Note: This register is write-synchronized - SYNCBUSY.CTRLB must be checked to ensure that synchronization of the CTRLBCLR register is complete.

Bit	7	6	5	4	3	2	1	0
	CMD[2:0]			IDXCMD[1:0]		ONESHOT	LUPD	DIR
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 7:5 – CMD[2:0] TCC Command
This bit field can be used for software control of retriggering and stopping commands of the TCC. When a command has been executed, the CMD bit field will be read back as zero. The commands are executed on the next prescaled GCLK_TCC clock cycle.
Writing `0x0` to this bit field has no effect
Writing a value other than `0x0` to this bit field will issue a command for execution.

Important: This command requires synchronization before being executed. A valid sequence is:

- Issue CMD command (CTRLBSET.CMD = command)
- Wait for CMD synchronization (SYNCBUSY.CTRLB = 0)
- Wait for CMD read back as zero (CTRLBSET.CMD = 0)

Value	Name	Description
0x0	NONE	No action
0x1	RETRIGGER	Force start, restart or retrigger
0x2	STOP	Force stop
0x3	UPDATE	Force update of double buffered registers
0x4	READSYNC	Force a read synchronization of COUNT
0x5	DMAOS	One-shot DMA trigger
Other	—	Reserved

Bits 4:3 – IDXCMD[1:0] Ramp Index Command
This bit field can be used to force cycle A and cycle B changes in RAMP2 and RAMP2A operation. On a timer/counter update condition, the command is executed, the IDX flag in the STATUS register is updated, and the IDXCMD command is cleared.
Writing `0x0` to this bit field has no effect.
Writing a valid value to this bit field will set a command.

Value	Name	Description
0x0	DISABLE	Command disabled: IDX toggles between cycles A and B
0x1	SET	Set IDX: Cycle B will be forced in the next cycle
0x2	CLEAR	Clear IDX: Cycle A will be forced in next cycle
0x3	HOLD	Hold IDX: The next cycle will be the same as the current cycle
Other	—	Reserved

Bit 2 – ONESHOT One-Shot

This bit controls one-shot operation of the TCC. When in one-shot operation, the TCC will stop counting on the next overflow/underflow condition or a stop command.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will enable the one-shot operation.

Value	Description
0	The TCC will count continuously
1	The TCC will stop counting on the next underflow/overflow condition

Bit 1 – LUPD Lock Update

This bit controls the update operation of the TCC buffered registers.
When the Lock Update bit is set, the value of the buffer registers will not be copied to their non-buffered counterparts during an UPDATE condition. Disabling the update ensures that all buffer registers can be updated to valid values before a hardware update is performed. After all the buffer registers have been loaded correctly, the buffered registers can be unlocked to allow the update to occur on the next UPDATE condition.
Note: This bit has no effect when input capture operation is enabled.

Writing a '0' to this bit has no effect.
Writing a '1' to this bit disables buffer registers updates on hardware UPDATE condition.

Value	Description
0	The values of the buffer registers *are* copied into the corresponding non-buffer registers on a hardware UPDATE condition
1	The values of the buffer registers *are not* copied into the corresponding non-buffer registers on a hardware UPDATE condition

Bit 0 – DIR Counter Direction

This bit is used to change the direction of the counter.
Writing a '0' to this bit has no effect
Writing a '1' to this bit will clear the bit and cause the counter to count up.
Note: When counting down, the COUNT register must be initialized to TOP value (PER or CC0 value depending on the mode).

Value	Description
0	The timer/counter is counting up (incrementing)
1	The timer/counter is counting down (decrementing)

23.6.4. Synchronization Busy

Name: SYNCBUSY
Offset: 0x08
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					CC3	CC2	CC1	CC0
Access					R	R	R	R
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PER	WAVE	PATT	COUNT	STATUS	CTRLB	ENABLE	SWRST
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 8, 9, 10, 11 – CCn Compare/Capture Channel n Synchronization Busy
This bit is cleared when the synchronization of Compare/Capture Channel n register between the clock domains is complete.
This bit is set when the synchronization of Compare/Capture Channel n register between clock domains is started.
The CCn bit is available only for existing Compare/Capture Channels. For details on the number of CC channels, refer to the feature list for each TCC.

Bit 7 – PER PER Synchronization Busy
This bit is cleared when the synchronization of PER register between the clock domains is complete.
This bit is set when the synchronization of the PER register between clock domains is started.

Bit 6 – WAVE WAVE Synchronization Busy
This bit is cleared when the synchronization of WAVE register between the clock domains is complete.
This bit is set when the synchronization of WAVE register between clock domains is started.

Bit 5 – PATT PATT Synchronization Busy
This bit is cleared when the synchronization of PATTERN register between the clock domains is complete.
This bit is set when the synchronization of PATTERN register between clock domains is started.

Bit 4 – COUNT COUNT Synchronization Busy
This bit is cleared when the synchronization of COUNT register between the clock domains is complete.
This bit is set when the synchronization of COUNT register between clock domains is started.

Bit 3 – STATUS STATUS Synchronization Busy
This bit is cleared when the synchronization of STATUS register between the clock domains is complete.
This bit is set when the synchronization of STATUS register between clock domains is started.

Bit 2 – CTRLB CTRLB Synchronization Busy
This bit is cleared when the synchronization of CTRLBSET or CTRLBCLR register between the clock domains is complete.
This bit is set when the synchronization of CTRLBSET or CTRLBCLR register between clock domains is started.

Bit 1 – ENABLE ENABLE Synchronization Busy
This bit is cleared when the synchronization of ENABLE bit between the clock domains is complete.
This bit is set when the synchronization of ENABLE bit between clock domains is started.

Bit 0 – SWRST SWRST Synchronization Busy
This bit is cleared when the synchronization of SWRST bit between the clock domains is complete.
This bit is set when the synchronization of SWRST bit between clock domains is started.

23.6.5. Recoverable Fault Control x

Name: FCTRLx
Offset: 0x0C + x*0x04 [x=0..1]
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
					FILTERVAL[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BLANKVAL[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
	BLANKPRESC	CAPTURE[2:0]			CHSEL[1:0]		HALT[1:0]	
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
	RESTART	BLANK[1:0]		QUAL	KEEP		SRC[1:0]	
Access	R/W	R/W	R/W	R/W	R/W		R/W	R/W
Reset	0	0	0	0	0		0	0

Bits 27:24 – FILTERVAL[3:0] Recoverable Fault Filter Value
This bit field defines the filter value applied to the MCEn (n = 0,1) event input line. The value must be set to zero when the MCEn event is used as a synchronous event.

Bits 23:16 – BLANKVAL[7:0] Recoverable Fault Blanking Value
This bit field determines the duration of the blanking of the fault input source. Activation and edge selection of the blank filtering are controlled by the BLANK bits (FCTRLn.BLANK).
When enabled, the fault input source is internally disabled for `BLANKVAL * prescaled GCLK_TCC` periods after the detection of the waveform edge.

Bit 15 – BLANKPRESC Recoverable Fault Blanking Value Prescaler
This bit enables a prescaler that multiplies the Blanking Value (BLANKVAL) by a factor of 64.

Value	Description
0	Blank time is `BLANKVAL * prescaled GCLK_TCCn`.
1	Blank time is `BLANKVAL * 64 * prescaled GCLK_TCCn`.

Bits 14:12 – CAPTURE[2:0] Recoverable Fault Capture Action
This bit field selects the capture and Fault n interrupt/event conditions.

Value	Name	Description
0x0	DISABLE	Capture on valid recoverable Fault is disabled
0x1	CAPT	On the rising edge of a valid recoverable fault, the counter value is captured on the channel selected by CHSEL[1:0] The INTFLAG.FAULTn flag is set on each new captured value.
0x2	CAPTMIN	On the rising edge of a valid recoverable fault, the counter value is captured on the channel selected by CHSEL[1:0] if the COUNT value is lower than the last stored capture value (CC). The INTFLAG.FAULTn flag is set on each local minimum detection.

Value	Name	Description
0x3	CAPTMAX	On the rising edge of a valid recoverable fault, the counter value is captured on the channel selected by CHSEL[1:0] if the COUNT value is higher than the last stored capture value (CC). The INTFLAG.FAULTn flag is set on each local maximun detection.
0x4	LOCMIN	On the rising edge of a valid recoverable fault, the counter value is captured on the channel selected by CHSEL[1:0]. The INTFLAG.FAULTn flag is set on each local minimum value detection.
0x5	LOCMAX	On the rising edge of a valid recoverable fault, the counter value is captured on the channel selected by CHSEL[1:0]. The INTFLAG.FAULTn flag is set on each local maximun detection.
0x6	DERIV0	On the rising edge of a valid recoverable fault, the counter value is captured on the channel selected by CHSEL[1:0]. The INTFLAG.FAULTn flag is set on each local maximun or minimum detection.
0x7	CAPTMARK	Capture with the ramp index as the MSB value

Bits 11:10 – CHSEL[1:0] Recoverable Fault Capture Channel
This bit field selects the channel for the capture operation triggered by a recoverable fault.

Value	Name	Description
0x0	CC0	The captured value stored in CC0
0x1	CC1	The captured value stored in CC1
0x2	CC2	The captured value stored in CC2
0x3	CC3	The captured value stored in CC3

Bits 9:8 – HALT[1:0] Recoverable Fault Halt Operation
This bit field selects the halt action for a recoverable fault.

Value	Name	Description
0x0	DISABLE	Halt action disabled
0x1	HW	Hardware halt action
0x2	SW	Software halt action
0x3	NR	Non-recoverable fault

Bit 7 – RESTART Recoverable Fault Restart
Setting this bit enables restart action for Fault.

Value	Description
0	The restart action for fault n is disabled
1	The restart action for fault n is enabled

Bits 6:5 – BLANK[1:0] Recoverable Fault Blanking Operation
This bit field selects the blanking start point for a recoverable fault.

Value	Name	Description
0x0	START	Blanking is applied from the start of the ramp period
0x1	RISE	Blanking is applied from the rising edge of the waveform output
0x2	FALL	Blanking is applied from the falling edge of the waveform output
0x3	BOTH	Blanking is applied from each toggle of the waveform output

Bit 4 – QUAL Recoverable Fault Qualification
Setting this bit enables the recoverable Fault input qualification.

Value	Description
0	The recoverable Fault n input is not disabled by the CMPn value condition
1	The recoverable Fault n input is disabled when the output signal is at the inactive level (CMPn = 0)

Bit 3 – KEEP Recoverable Fault Keep
Setting this bit enables the fault keep action.

Value	Description
0	The fault state is released as soon as the recoverable Fault n is cleared
1	The fault state is released at the end of the TCC cycle

Bits 1:0 – SRC[1:0] Recoverable Fault Source
This bit field selects the TCC event input for a recoverable fault.
When used as a recoverable fault input, the event system channel connected to the MCEn event input, must be configured to route the event asynchronously.

Value	Name	Description
0x0	DISABLE	Fault input disabled
0x1	ENABLE	MCEn (n = 0,1) event input
0x2	INVERT	Inverted MCEn (n = 0,1) event input
0x3	ALTFAULT	Alternate fault (A or B) state at the end of the previous period.

23.6.6. Waveform Extension Control

Name: WEXCTRL
Offset: 0x14
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bits 1:0 – OTMX[1:0] Output Matrix
This bit field defines the matrix routing of the TCC waveform generation outputs to the port pins, according to the *Waveform Extension* section.

23.6.7. Driver Control

Name: DRVCTRL
Offset: 0x18
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
	FILTERVAL1[3:0]				FILTERVAL0[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	23	22	21	20	19	18	17	16
					INVEN3	INVEN2	INVEN1	INVEN0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
					NRV3	NRV2	NRV1	NRV0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
					NRE3	NRE2	NRE1	NRE0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 31:28 – FILTERVAL1[3:0] Non-Recoverable Fault Input 1 Filter Value
This bit field defines the filter value applied to the TCE1 event input line. When the TCE1 event input line is configured as a synchronous event, this value must be `0x0`.

Bits 27:24 – FILTERVAL0[3:0] Non-Recoverable Fault Input 0 Filter Value
This bit field defines the filter value applied to the TCE0 event input line. When the TCE0 event input line is configured as a synchronous event, this value must be `0x0`.

Bits 16, 17, 18, 19 – INVENn Waveform Output n Inversion
This bit is used to select inversion on the output of channel n.
Writing a '0' to INVENn disables inversion of the output from WO[n].
Writing a '1' to INVENn inverts the output from WO[n].

Bits 8, 9, 10, 11 – NRVn Non-Recoverable State n Output Value
This bit defines the value of the enabled override output n under non-recoverable fault condition.

Bits 0, 1, 2, 3 – NREn Non-Recoverable State n Output Enable
This bit enables the override of output n by NRVn value under a non-recoverable fault condition.

Value	Description
0	A non-recoverable fault sets the output to tri-state
1	A non-recoverable faults set the output to NRVn level.

23.6.8. Debug control

Name: DBGCTRL
Offset: 0x1E
Reset: 0x00
Property: PAC Write-Protection

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

Bit 2 – FDDBD Fault Detection on Debug Break Detection
This bit is not affected by a software reset and should not be changed by software while the TCC is enabled.
By default, this bit is zero, and the on-chip debug (OCD) fault protection is disabled.
When writing a '1' to this bit, an OCD break request from the OCD system will trigger a non-recoverable fault. When this bit is set, OCD fault protection is enabled, and any OCD break request from the OCD system will trigger a non-recoverable fault.

Value	Description
0	No faults are generated when the TCC is halted in Debug mode
1	A non recoverable fault is generated, and FAULTD flag is set when the TCC is halted in Debug mode

Bit 0 – DBGRUN Debug Running State
This bit is affected by system software Reset and should not be changed by software while the TCC is enabled.

Value	Description
0	The TCC is halted when the device is halted in Debug mode
1	The TCC continues normal operation when the Device is halted in Debug mode

23.6.9. Event Control

Name: EVCTRL
Offset: 0x20
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
					MCEO3	MCEO2	MCEO1	MCEO0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
					MCEI3	MCEI2	MCEI1	MCEI0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TCEI1	TCEI0	TCINV1	TCINV0		CNTEO	TRGEO	OVFEO
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
	CNTSEL[1:0]		EVACT1[2:0]			EVACT0[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 24, 25, 26, 27 – MCEOn Match or Capture Channel n Event Output Enable
These bits control whether the match/capture event on channel n is enabled and will be generated for every match or capture.

Value	Description
0	The match/capture x event is disabled and will not be generated
1	The match/capture x event is enabled and will be generated for every compare or capture on channel n

Bits 16, 17, 18, 19 – MCEIn Match or Capture Channel n Event Input Enable
These bits indicate whether the match/capture n incoming event is enabled.
These bits are used to enable match or capture input events to the CCn channel of the TCC and to enable recoverable Faults A and B.

Value	Description
0	Incoming events are disabled
1	Incoming events are enabled

Bits 14, 15 – TCEIn Timer/Counter Event Input n Enable
This bit is used to enable input event n to the TCC.

Value	Description
0	Incoming event n is disabled
1	Incoming event n is enabled

Bits 12, 13 – TCINVn Timer/Counter Event n Invert Enable
This bit inverts the event n input.

Value	Description
0	Input event source n is not inverted
1	Input event source n is inverted

Bit 10 – CNTEO Timer/Counter Event Output Enable
This bit is used to enable the counter cycle event. When enabled, an event will be generated at the beginning or end of the counter cycle depending on the CNTSEL[1:0] settings.

Value	Description
0	The Counter cycle output event is disabled and will not be generated
1	The Counter cycle output event is enabled and will be generated depending on the CNTSEL[1:0] value

Bit 9 – TRGEO Retrigger Event Output Enable
This bit is used to enable the counter retrigger event. When enabled, an event will be generated when the counter retriggers.

Value	Description
0	The counter retrigger event is disabled and will not be generated
1	The counter retrigger event is enabled and will be generated for every counter retrigger

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 reaches the TOP or ZERO value.

Value	Description
0	The overflow/underflow counter event is disabled and will not be generated
1	The overflow/underflow counter event is enabled and will be generated for every counter overflow or underflow

Bits 7:6 – CNTSEL[1:0] Timer/Counter Interrupt and Event Output Selection
These bits define on which part of the counter cycle the counter event output is generated.

Value	Name	Description
0x0	START	An interrupt/event is generated at the beginning of each counter cycle
0x1	END	An interrupt/event is generated at the end of each counter cycle
0x3	BOUNDARY	An interrupt/event is generated at the beginning of the first counter cycle, and at the end of last counter cycle
Other	—	Reserved

Bits 5:3 – EVACT1[2:0] Timer/Counter Event Input 1 Action
This bit field defines the action the TCC will perform on the TCEI1 event input.

Value	Name	Description
0x0	OFF	Event action disabled
0x1	RETRIGGER	Restart or retrigger TCC on event
0x2	DIR	Direction control
0x3	STOP	Stop TCC on event
0x4	DEC	Decrement TCC on event
0x6	PWP	Period captured into CC1 Pulse Width on CC0
0x7	FAULT	Non-recoverable fault
Other	—	Reserved

Bits 2:0 – EVACT0[2:0] Timer/Counter Event Input 0 Action
This bit field defines the action the TCC will perform on the TCEI0 event input 0.

Value	Name	Description
0x0	OFF	Event action disabled
0x1	RETRIGGER	Restart or retrigger TCC on event

Value	Name	Description
0x2	COUNTEV	Count on event
0x3	START	Start TCC on event
0x4	INC	Increment TCC on event
0x5	COUNT	Count on the active state of an asynchronous event
0x6	STAMP	Capture Time-stamp (overflow)
0x7	FAULT	Non-recoverable fault
Other	—	Reserved

23.6.10. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x24
Reset: 0x00000000
Property: PAC Write-Protection

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 (INTENSET) register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					MC3	MC2	MC1	MC0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FAULT1	FAULT0	FAULTB	FAULTA	DFS	UFS		
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bit	7	6	5	4	3	2	1	0
					ERR	CNT	TRG	OVF
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the corresponding Match or Capture Channel n Interrupt Disable/Enable bit, thereby disabling the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel x interrupt is disabled
1	The Match or Capture Channel x interrupt is enabled

Bits 14, 15 – FAULTn Non-Recoverable Fault n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Non-Recoverable Fault n Interrupt Disable/Enable bit, which disables the Non-Recoverable Fault n interrupt.

Value	Description
0	The Non-Recoverable Fault 1 interrupt is disabled
1	The Non-Recoverable Fault 1 interrupt is enabled

Bits 12, 13 – FAULTx Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Recoverable Fault x Interrupt Disable/Enable bit, which disables the Recoverable Fault x interrupt.

Value	Description
0	The Recoverable Fault B interrupt is disabled

Value	Description
1	The Recoverable Fault B interrupt is enabled

Bit 11 – DFS Non-Recoverable Debug Fault Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Debug Fault State Interrupt Disable/Enable bit, which disables the Debug Fault State interrupt.

Value	Description
0	The Debug Fault State interrupt is disabled
1	The Debug Fault State interrupt is enabled

Bit 10 – UFS Non-Recoverable Update Fault Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Non-Recoverable Update Fault Interrupt Disable/Enable bit, which disables the Non-Recoverable Update Fault interrupt.
Note: This bit is only available on variant L devices. Refer to the *Configuration Summary* for more information.

Value	Description
0	The Non-Recoverable Update Fault interrupt is disabled
1	The Non-Recoverable Update Fault interrupt is enabled

Bit 3 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Error Interrupt Disable/Enable bit, which disables the Compare interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 2 – CNT Counter Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Counter Interrupt Disable/Enable bit, which disables the Counter interrupt.

Value	Description
0	The Counter interrupt is disabled
1	The Counter interrupt is enabled

Bit 1 – TRG Retrigger Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Retrigger Interrupt Disable/Enable bit, which disables the Retrigger interrupt.

Value	Description
0	The Retrigger interrupt is disabled
1	The Retrigger interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overflow Interrupt Disable/Enable bit, which disables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled

Value	Description
1	The Overflow interrupt is enabled

23.6.11. Interrupt Enable Set

Name: INTENSET
Offset: 0x28
Reset: 0x00000000
Property: PAC Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					MC3	MC2	MC1	MC0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FAULT1	FAULT0	FAULTB	FAULTA	DFS	UFS		
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bit	7	6	5	4	3	2	1	0
					ERR	CNT	TRG	OVF
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – MCn Match or Capture Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the corresponding Match or Capture Channel n Interrupt Disable/Enable bit, which enables the Match or Capture Channel n interrupt.

Value	Description
0	The Match or Capture Channel n interrupt is disabled
1	The Match or Capture Channel n interrupt is enabled

Bits 14, 15 – FAULTn Non-Recoverable Fault n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Non-Recoverable Fault n Interrupt Disable/Enable bit, which enables the Non-Recoverable Fault n interrupt.

Value	Description
0	The Non-Recoverable Fault 1 interrupt is disabled
1	The Non-Recoverable Fault 1 interrupt is enabled

Bits 12, 13 – FAULTx Recoverable Fault x Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Recoverable Fault x Interrupt Disable/Enable bit, which enables the Recoverable Fault x interrupt.

Value	Description
0	The Recoverable Fault B interrupt is disabled

Value	Description
1	The Recoverable Fault B interrupt is enabled

Bit 11 – DFS Non-Recoverable Debug Fault Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Debug Fault State Interrupt Disable/Enable bit, which enables the Debug Fault State interrupt.

Value	Description
0	The Debug Fault State interrupt is disabled
1	The Debug Fault State interrupt is enabled

Bit 10 – UFS Non-Recoverable Update Fault Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Non-Recoverable Update Fault Interrupt Disable/Enable bit, which enables the Non-Recoverable Update Fault interrupt.

Value	Description
0	The Non-Recoverable Update Fault interrupt is disabled
1	The Non-Recoverable Update Fault interrupt is enabled

Bit 3 – ERR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Error Interrupt Disable/Enable bit, which enables the Compare interrupt.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 2 – CNT Counter Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Retrigger Interrupt Disable/Enable bit, which enables the Counter interrupt.

Value	Description
0	The Counter interrupt is disabled
1	The Counter interrupt is enabled

Bit 1 – TRG Retrigger Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Retrigger Interrupt Disable/Enable bit, which enables the Retrigger interrupt.

Value	Description
0	The Retrigger interrupt is disabled
1	The Retrigger interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overflow Interrupt Disable/Enable bit, which enables the Overflow interrupt request.

Value	Description
0	The Overflow interrupt is disabled
1	The Overflow interrupt is enabled

23.6.12. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					MC3	MC2	MC1	MC0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FAULT1	FAULT0	FAULTB	FAULTA	DFS	UFS		
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bit	7	6	5	4	3	2	1	0
					ERR	CNT	TRG	OVF
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – MCn Match or Capture Channel n Interrupt Flag
This flag is set after a match with the compare condition or once the CCn register contains a valid capture value, and will generate an interrupt request if INTENSET.MCn = '1'.
Writing a '0' to one of these bits has no effect.
Writing a '1' to one of these bits will clear the corresponding Match or Capture Channel n interrupt flag.
In Capture operation, this flag is automatically cleared when the CCn register is read.

Bits 14, 15 – FAULTn Non-Recoverable Fault n Interrupt Flag
This flag is set after a Non-Recoverable Fault n occurs, and will generate an interrupt request if INTENSET.FAULTn = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Non-Recoverable Fault n interrupt flag.

Bits 12, 13 – FAULTx Recoverable Fault x Interrupt Flag
This flag is set after a Recoverable Fault x occurs, and will generate an interrupt request if INTENSET.FAULTx = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Non-Recoverable Fault x interrupt flag.

Bit 11 – DFS Non-Recoverable Debug Fault State Interrupt Flag
This flag is set after a Debug Fault State occurs, and will generate an interrupt request if INTENSET.DFS = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Debug Fault State interrupt flag.

Bit 10 – UFS Non-Recoverable Update Fault
This flag is set when the RAMP index changes and the Lock Update bit is set (CTRLBSET.LUPD = '1'), and will generate an interrupt request if INTENSET.UFS = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Non-Recoverable Update Fault interrupt flag.

Bit 3 – ERR Error Interrupt Flag
This flag is set if a new capture occurs on a channel when the corresponding Match or Capture Channel x interrupt flag = '1'. This means that it is not possible to save the new capture value, and it will generate an interrupt request if INTENSET.ERR = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Error Interrupt flag.

Bit 2 – CNT Counter Interrupt Flag
This flag is set after a counter event occurs, and it will generate an interrupt request if INTENSET.CNT = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the CNT Interrupt flag.

Bit 1 – TRG Retrigger Interrupt Flag
This flag is set after a counter retrigger occurs, and it will generate an interrupt request if INTENSET.TRG = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Retrigger Interrupt flag.

Bit 0 – OVF Overflow Interrupt Flag
This flag is set after an overflow condition occurs, and it will generate an interrupt request if INTENSET.OVF = '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Overflow Interrupt flag.

23.6.13. Status

Name: STATUS
Offset: 0x30
Reset: 0x00000001
Property: Read-Synchronized, Write-Synchronized

Note: This register is read-synchronized and write-synchronized. The SYNCBUSY.STATUS must be checked to ensure the STATUS register synchronization is complete.

Bit	31	30	29	28	27	26	25	24
					CMP3	CMP2	CMP1	CMP0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
					CCBUFV3	CCBUFV2	CCBUFV1	CCBUFV0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FAULT1	FAULT0	FAULTB	FAULTA	FAULT1IN	FAULT0IN	FAULTBIN	FAULTAIN
Access	R/W	R/W	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PERBUFV		PATTBUFV		DFS	UFS	IDX	STOP
Access	R/W		R/W		R/W	R/W	R	R
Reset	0		0		0	0	0	1

Bits 24, 25, 26, 27 – CMPn Compare Channel n Value
This bit reflects the channel n output compare value.

Value	Description
0	Channel compare output value is 0
1	Channel compare output value is 1

Bits 16, 17, 18, 19 – CCBUFVn Compare/Capture Channel n Buffer Valid
For compare channel:
This bit is set when a new value is written to the corresponding CCBUFn register.
This bit is cleared either by writing a '1' to the corresponding location when the CTRLB.LUPD is set, or automatically on an UPDATE condition.
For capture channel:
This bit is set when a valid capture value is stored in the CCBUFn register.
This bit is automatically cleared when the CCBUF value is copied into its CCn register.

Bits 14, 15 – FAULTn Non-recoverable Fault n State
This bit is set as soon as a non-recoverable Fault n condition occurs.
This bit is cleared by writing a '1' to this bit and the corresponding FAULTnIN status bit is '0'.
Once this bit is clear, the timer/counter will restart from the last COUNT value. To restart the timer/counter from BOTTOM, the timer/counter restart command must be executed before clearing the corresponding STATEn bit. For further details on timer/counter commands, refer to the commands listed in the TCC Command bit field of the Control B Set (CTRLBSET.CMD) register.

Bits 12, 13 – FAULTx Recoverable Fault x State
This bit is set as soon as recoverable Fault x condition occurs.
This bit is cleared when Fault B action is resumed, or by writing a '1' to this bit when the corresponding FAULTxIN bit is low. If the software halt command is enabled (FAULTB.HALT = SW), clearing this bit will release the timer/counter.

Bits 10, 11 – FAULTnIN Non-Recoverable Fault n Input
This bit is set while an active Non-Recoverable Fault n input is present.

Bits 8, 9 – FAULTxIN Recoverable Fault x Input
This bit is set while an active Recoverable Fault x input is present.

Bit 7 – PERBUFV Period Buffer Valid
This bit is set when a new value is written to the PERBUF register.
This bit is cleared on an UPDATE condition when the Lock Update bit in the Control B (CTRLB.LUPD) register is set, or by writing a '1' to this bit.

Bit 5 – PATTBUFV Pattern Generator Value Buffer Valid
This bit is set when a new value is written to the PATTBUF register.
This bit is cleared on an UPDATE condition when the Lock Update bit in the Control B (CTRLB.LUPD) register is set, or by writing a '1' to this bit.

Bit 3 – DFS Debug Fault State
This bit is set in Debug mode when the Fault Detection on Debug Break Detection bit in the Debug Control (DDBGCTRL.FDDBD) register is set.
This bit is cleared by writing a '1' to it and when the TCC is not in Debug mode.
When this bit is set, the counter is halted and the Waveforms state depends on the Non-Recoverable State n Output Enable (NREx) bits and the Non-Recoverable State n Output Value (NRVn) bits of the Driver Control (DRVCTRL) register.

Bit 2 – UFS Non-recoverable Update Fault State
This bit is set by hardware when the RAMP index changes and the Lock Update bit is set (CTRLBSET.LUPD). The bit is cleared by writing a one to this bit.
When the bit is set, the waveforms state depend on the DRVCTRL.NRE and DRVCTRL.NRV registers.

Bit 1 – IDX Ramp Index
For RAMP2 operation:
This bit is cleared during cycle A.
This bit is set during cycle B
For RAMP1 operation:
This bit always reads '0'.
For details on ramp operations, refer to the *Ramp Operations* section.

Bit 0 – STOP Stop
This bit is set when the TCC is disabled, either on a STOP command or on an UPDATE condition when One-Shot operation mode is enabled (CTRLBSET.ONESHOT = '1').
This bit is cleared on the next incoming counter increment or decrement.

Value	Description
0	Counter is running
1	Counter is stopped

23.6.14. Counter

Name: COUNT
Offset: 0x34
Reset: 0x00000000
Property: PAC Write-Protection, Write-Synchronized

Notes:

1. Prior to any read access, this register must be manually read synchronized by writing to the TCC Command bit in the Control B Set register (CTRLBSET.CMD = READSYNC) and wait for the manual read synchronization to finish (SYNCBUSY.CTRLB and CTRLBSET.CMD are '0').
2. This register is write-synchronized. The COUNT Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.COUNT) must be checked to ensure that the COUNT register write synchronization is complete.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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
This bit field hold the value of the Counter Value register.

23.6.15. Pattern

Name: PATT
Offset: 0x38
Reset: 0x0000
Property: Write-Synchronized

Note: This register is write-synchronized. The PATT Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.PATT) must be checked to ensure that the PATTERN register synchronization is complete.

Bit	15	14	13	12	11	10	9	8
					PGV3	PGV2	PGV1	PGV0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
					PGE3	PGE2	PGE1	PGE0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 8, 9, 10, 11 – PGVn Pattern Generation Output n Value
This register holds the values of pattern for each waveform output.

Bits 0, 1, 2, 3 – PGEn Pattern Generation Output n Enable
This register holds the enable status of the pattern generation for each waveform output.

23.6.16. Waveform

Name: WAVE
Offset: 0x3C
Reset: 0x00000000
Property: Write-Synchronized, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					POL3	POL2	POL1	POL0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
					CICCEN3	CICCEN2	CICCEN1	CICCEN0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

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

Bits 16, 17, 18, 19 – POLn Channel n Polarity
Setting these bits enables the output polarity in single- and dual-slope PWM and dual-compare operations.

Channel n Polarity in Single-Slope Mode

Value	Name	Description
0	NORMAL	Compare output is initialized to ~DIR and set to DIR when TCC counter matches CCn value.
1	INVERTED	Compare output is initialized to DIR and set to ~DIR when TCC counter matches CCn value.

Notes:

- These bits are not enable-protected
- These bits are write-synchronized. The WAVE Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.WAVE) must be checked to ensure that the WAVE.POLn synchronization is complete.

Channel n Polarity in Dual-Slope Mode

Value	Name	Description
0	NORMAL	Compare output is set to ~DIR when TCC counter matches CCn value
1	INVERTED	Compare output is set to DIR when TCC counter matches CCn value.

Notes:

- These bits are not enable-protected
- These bits are write-synchronized. The WAVE Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.WAVE) must be checked to ensure that the WAVE.POLn synchronization is complete.

Channel n Polarity in Dual-Compare Mode

Value	Name	Description
0	NORMAL	Compare output is initialized to ~DIR, set to DIR when TCC counter matches CCn value and set to ~DIR when TCC counter matches CC[n+WO_NUM/2] value.
1	INVERTED	Compare output is initialized to DIR, set to ~DIR when TCC counter matches CCn value and set to DIR when TCC counter matches CC[n+WO_NUM/2] value.

Notes:

- These bits are not enable-protected
- These bits are write-synchronized. The WAVE Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.WAVE) must be checked to ensure that the WAVE.POLn synchronization is complete.

Bits 8, 9, 10, 11 – CICCENn Circular Compare/Capture Channel n Enable

Setting these bits enables the compare circular buffer option on the channel. When the bit is set, the CCn register value is copied back into the CCn register on an UPDATE condition.

Notes:

- These bits are not enable-protected
- These bits are write-synchronized. The WAVE Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.WAVE) must be checked to ensure that the WAVE.CICCENn synchronization is complete.

Bit 7 – CIPEREN Circular Period Enable

Setting these bits enables the period circular buffer option. When the bit is set, the PER register value is copied back into the PERB register on an UPDATE condition.

Notes:

- This bit is not enable-protected
- This bit is write-synchronized. The WAVE Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.WAVE) must be checked to ensure that the WAVE.CIPEREN synchronization is complete.

Bits 6:4 – RAMP[2:0] Ramp Operation

This bit field select Ramp operation (RAMP).

Notes:

- This bit field is enable-protected
- This bit field is not synchronized

Value	Name	Description
0x0	RAMP1	RAMP1 operation
0x1	RAMP2A	Alternative RAMP2 operation
0x2	RAMP2	RAMP2 operation
0x3	RAMP2C	Critical RAMP2 operation
Other	—	Reserved

Bits 2:0 – WAVEGEN[2:0] Waveform Generation Operation

This bit field selects the waveform generation operation. The settings impact the top value and determine whether frequency or PWM frequency or PWM waveform generation should be used.

Notes:

- This bit field is enable-protected
- This bit field is not synchronized

Value	Name	Description
0x0	NFRQ	Normal Frequency

Value	Name	Description
0x1	MFRQ	Match Frequency
0x2	NPWM	Normal PWM
0x3	DPWM	Dual Compare PWM
0x4	DSCRITICAL	Dual-slope PWM
0x5	DSBOTTOM	Dual-slope PWM
0x6	DSBOTH	Dual-slope PWM
0x7	DSTOP	Dual-slope PWM
Other	—	Reserved

23.6.17. Compare/Capture Channel n

Name: CC[n]
Offset: 0x44 + n*0x04 [n=0..3]
Reset: 0x00000000
Property: Write-Synchronized

The Compare/Capture Channel n (CCn) register holds the 16-bit CCn value. The register has two functions, depending on the mode of operation.

For capture operation, this register represents the second buffer level and access point for the CPU and DMA.

For compare operation, this register is continuously compared to the counter value. Normally, the output form the comparator is then used for generating waveforms.

The CCn register is updated with the buffer value from its corresponding Compare/Capture Channel n Buffer (CCBUFn) register when an UPDATE condition occurs.

Note: In match frequency operation, the CC0 register controls the counter period.

Note: This register is write-synchronized. The CCn Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.CCn) must be checked to ensure that the CCn register synchronization is complete.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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 Channel Value
This bit field holds the value of the Compare/Capture Channel.

23.6.18. Period

Name: PER
Offset: 0x40
Reset: 0xFFFFFFFF
Property: Write-Synchronized

Note: This register is write-synchronized. The PER Synchronization Busy bit of the Synchronization Busy register (SYNCBUSY.PER) must be checked to ensure that the Period Value (PER) register synchronization is complete.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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	1	1	1	1	1	1	1	1

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 15:0 – PER[15:0] Period Value
This bit field holds the value of the TCC period register.

23.6.19. Pattern Buffer

Name: PATTBUF
Offset: 0x64
Reset: 0x0000
Property: Write-Synchronized

Notes:

- This register is write-synchronized. The PATT Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.PATT) must be checked to ensure that synchronization of the Pattern Buffer (PATTBUF) register is complete.
- This register must be written with 16-bit accesses only (no 8-bit writes)

Bit	15	14	13	12	11	10	9	8
					PGVB3	PGVB2	PGVB1	PGVB0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
					PGEB3	PGEB2	PGEB1	PGEB0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 8, 9, 10, 11 – PGVBn Pattern Generation Output Value Buffer n
These bits serves as the buffer for the PGV register. If double buffering is used, the valid content of this register is copied to the PGV register on an UPDATE condition or when the Update command is sent to the Control B Set register (CTRLBSET.CMD = `UPDATE`).

Bits 0, 1, 2, 3 – PGEBn Pattern Generation Output Enable Buffer n
These bits serve as the buffer of the PGE register. If double buffering is used,the valid content of this register is copied into the PGE register at an UPDATE condition or when the Update command is sent to the Control B Set register (CTRLBSET.CMD = `UPDATE`).

23.6.20. Period Buffer

Name: PERBUF
Offset: 0x6C
Reset: 0xFFFFFFFF
Property: Write-Synchronized

Note: This register is write-synchronized. The PER Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.PER) must be checked to ensure that synchronization of the Period Value Buffer (PERBUF) register is complete. This register must be written with 32-bit accesses only (no 8-bit or 16-bit writes).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	PERBUF[15:8]							
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

Bit	7	6	5	4	3	2	1	0
	PERBUF[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 15:0 – PERBUF[15:0] Period Buffer Value
This bit field holds the value of the Period Buffer register. The value is copied to the PER register on an UPDATE condition or when the Update command is sent to the Control B Set register (CTRLBSET.CMD = `UPDATE`).

23.6.21. Compare/Capture Channel n Buffer

Name: CCBUF[n]
Offset: 0x70 + n*0x04 [n=0..3]
Reset: 0x00000000
Property: Write-Synchronized

The Compare/Capture Channel n Buffer (CCBUFn) register serves as the buffer for the associated Compare/Capture Channel n (CCn) register. The CCBUFn register value is copied to the CCn register on an UPDATE condition or when writing the Update command to the Command bit field in the Control B Set register (CTRLBSET.CMD = `UPDATE`).

Notes:

- This register is write-synchronized. The CCn Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.CCn) must be checked to ensure that synchronization of the CCn register is complete..
- This register must be written with 32-bit access only (no 8- or 16-bit writes)
- Accessing this register using the CPU or DMA will affect the corresponding Compare/Capture Channel n Buffer Valid bit in the Status register (STATUS.CCBUFVn)

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	CCBUF[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
	CCBUF[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 – CCBUF[15:0] Compare/Capture Channel Buffer Value
This bit field holds the Compare/Capture Channel Buffer Value.

24. EVSYS – Event System

24.1. Features

- 4 Configurable Event Channels, where Each Channel Can:
  - Be connected to any event generator
  - Provide a pure asynchronous, resynchronized or synchronous path
- 62 Event Generators
- 22 Event Users
- Configurable Edge Detector
- Peripherals Can be Event Generators, Event Users, or Both
- SleepWalking and Interrupt Support for Operation in Sleep Modes
- Software Event Generation
- Each Event User Can Choose Which Channel to Respond to

24.2. Overview

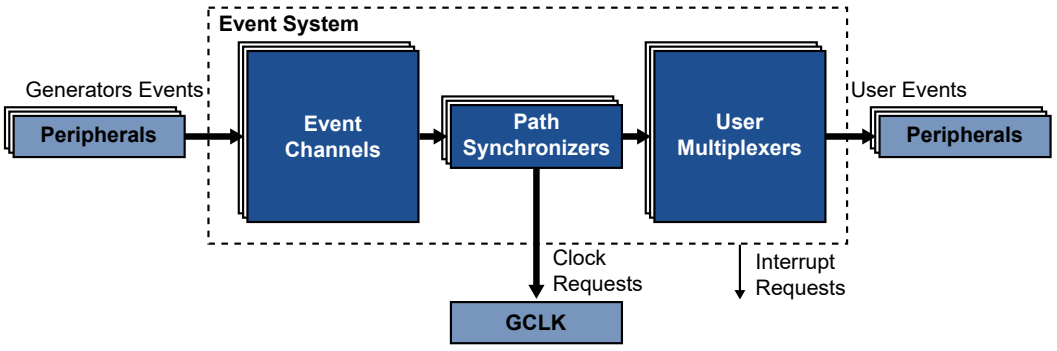
The Event System (EVSYS) enables autonomous, low-latency, and configurable communication between peripherals.

Several peripherals can be configured to generate and/or respond to signals known as events. The exact condition for generating an event, or the action taken upon receiving an event, is specific to each peripheral. Peripherals that respond to events are called event users, while peripherals that generate events are called event generators. A peripheral can have one or more event generators and one or more event users.

Communication is performed without CPU intervention and without consuming system resources such as bus or SRAM bandwidth. This reduces the load on the CPU and other system resources compared to a traditional interrupt-based system.

24.3. Block Diagram

Figure 24-1. Event System Block Diagram



24.3.1. Signal Description

Not applicable.

24.4. Functional Description

24.4.1. Initialization

To enable event routing within the system, the Event Users Multiplexer and Event Channels must be selected in the Event System (EVSYS), and the two peripherals that generate and use the event must be configured. The recommended sequence is:

1. In the event user peripheral, corresponding to the configuration in step 2a:
   a. Configure the action to be executed upon receiving an event by writing to the Event Action bits (EVACT) in the respective Event control register (e.g., TCn.EVCTRL.EVACT). Note: Not all peripherals require this step.
   b. Enable event input by writing a '1' to the respective Event Input Enable bit ("EI") in the peripheral's Event Control register (e.g., ADCn.EVCTRL.STARTEI).
2. Configure the EVSYS:
   a. Configure which channel Event User n should use by writing to the respective User n Channel Selection register (EVSYS.USER[n]). Refer to Event Users for all available event users.
   b. Configure which path event channel n should use by writing to the respective Path Selection bit field in the Channel n Control register (EVSYS.CHANNEL[n].PATH).
   c. Optional: Configure the edge detection of channel n by writing to the Edge Detection Selection bit field in the Channel n Control register (EVSYS.CHANNEL[n].EDGESEL).
   d. Configure which event generator channel n should use, corresponding to the configuration in step 1, by writing to the Event Generator bit field in the Channel n Control register (EVSYS.CHANNEL[n].EVGEN). Refer to Event Generators for all available event generators.

   Note: To use the EVSYS, other parts of the system must be configured first. Refer to the *Dependencies* section for further information.
3. In the peripheral generating the event, enable event output by writing a '1' to the respective Event Output Enable bit ("EO") in the peripheral's Event Control register (e.g., TCCn.EVCTRL.MCEO1, ACn.EVCTRL.WINEO0).

24.4.2. Operation

The Event System consists of several channels that route internal events from peripherals, known as event generators, to other internal peripherals or I/O pins, known as event users. Each event generator can be selected as the source for multiple channels, but a channel cannot be set to use multiple event generators at the same time. Each event user can only use one channel at a time.

24.4.2.1. Enabling, Disabling, and Resetting

The EVSYS is always enabled.

The EVSYS is reset by writing a '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the EVSYS will be reset to their initial state and all ongoing events will be canceled.

Refer to the CTRLA.SWRST register for details.

24.4.2.2. Event Users

Each event user has a dedicated multiplexer that controls which channel the event user is connected to. A user multiplexer receives the output of all event channels and must be configured to select one of these channels. The channel is selected by writing to the Channel bit field in the User x Channel Selection register (USER[x].CHANNEL).

The user multiplexer must always be configured before the channel.

The table below shows the available event users for this device family.

Table 24-1. Event Users

n	USER[n] Name		Description	Input Detection	Channel Path Type
	Peripheral	Input			
0-3	PORT	EV_n	PORT Event n Input	Level/Rising Edge	Asynchronous
4-5	DMAC	CH_n	Channel n	Level	Asynchronous Synchronous Resynchronized
6-8	TCn	EVU	TCn Event Input	Level	Asynchronous Synchronous Resynchronized
9-10	TCC0	EV_n	TCCn Event Input n	Level/Edge	Asynchronous
11-14		MC_n	TCCn Capture event n	Rising Edge	Asynchronous
15	ADC	START	ADC conversion start	Edge	Asynchronous
16-19	CCL	LUTIN_n	CCL Look-up Table Input Value	Level	Asynchronous
20	MTB	START	MTB Start event	Rising Edge	Asynchronous Synchronous Resynchronized
21		STOP	MTB Stop event	Rising Edge	Asynchronous Synchronous Resynchronized

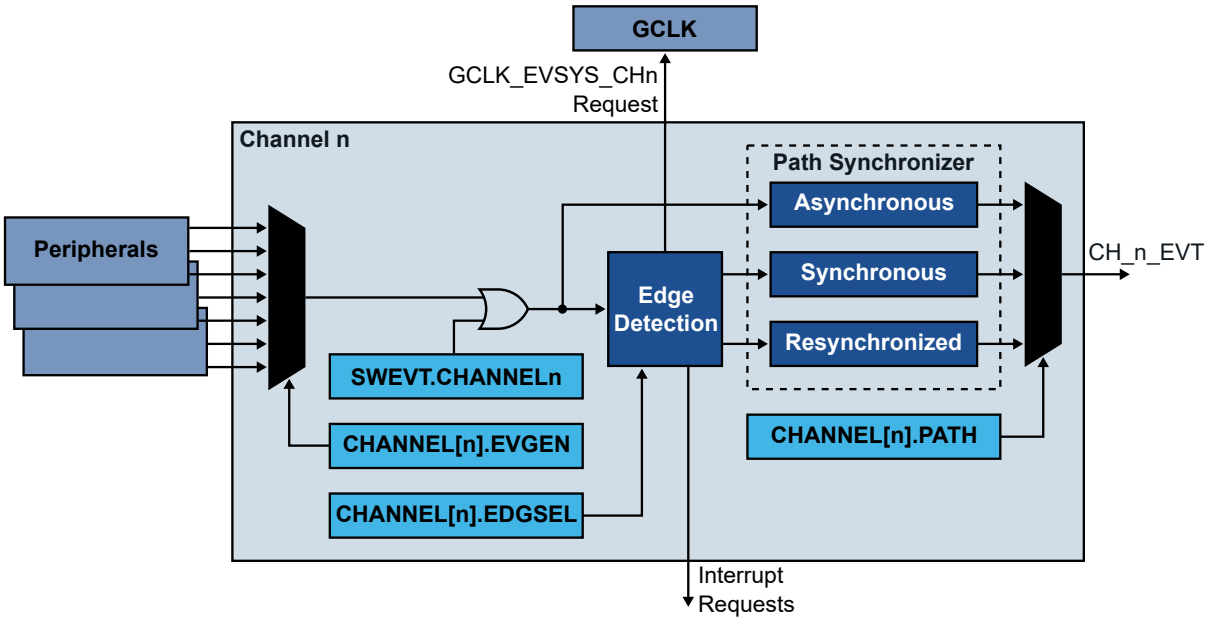
24.4.2.3. Event System Channel

An event channel can be connected to one of the event generators. Refer to *Event Generators* for further information.

Depending on the channel path configuration, the selected event can be sent to event users asynchronously, synchronously or resynchronized . Refer to the *Channel Path* section for further information.

When synchronous or resynchronized channel paths are used, the channel includes an internal edge detector. Refer to *Edge Detection* for further information.

Figure 24-2. Event Channel



24.4.2.4. Channel Path

There are three different paths for an event to propagate through an event system channel: Asynchronous, Synchronous or Resynchronized. The path is configured by writing to the Path Selection bit field of the Channel register (CHANNEL[n].PATH).

Asynchronous Path

When using the asynchronous path, events are propagated from the event generator to the event user without intervention from the Event System. The GCLK for this channel (GCLK_EVSYS_CHn) is not required.

When the asynchronous path is selected, the channel will not generate any interrupts, and the Channel Status register (CHSTATUS) will always read as zero. The Edge detection is non-functional and must be disabled. For further details, refer to the *Event Users* section and the description in each peripheral chapter description.

Synchronous Path

The synchronous path must only be used when the event generator and the event channel use the same generic clock generator, and the event user supports the synchronous path. For details on generic clock generators, refer to the *GCLK – Generic Clock Controller* chapter.

Resynchronized Path

The resynchronized path can be used when the event generator, the event channel, and the event user do not use the same clock source. The resynchronization of the event from the event generator to channel n, and from channel n to the event user, is automatically handled by EVSYS. For details on generic clock generators, refer to the *GCLK – Generic Clock Controller* chapter.

24.4.2.5. Event Latency

The latency from the event generator to the event user depends on the configuration of the channel:

- Asynchronous Path: An event will be propagated to the user without any clock latency.
- Synchronous Path: The maximum latency of an event is one GCLK_EVSYS_CHn clock cycle.

- Resynchronized Path: The maximum latency of an event is three GCLK_EVSYS_CHn clock cycles.

24.4.2.6. Edge Detection

When synchronous or resynchronized paths are used, edge detection must be enabled. The event system can execute 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 both rising and falling edges

Edge detection is selected by writing to the Edge Selection bit field of the Channel register (CHANNEL[n].EDGSEL).

When event channel n is configured and enabled, the channel will be ready to service incoming events after one GCLK_EVSYS_CHn cycle for rising edge detection (CHANNEL[n].EDGSEL=0x1), and after three GCLK_EVSYS_CHn cycles for falling edge detection (CHANNEL[n].EDGSEL=0x2).

24.4.2.7. Event Generators

Each event channel can be configured to receive events from a specific event generator by writing to the Event Generator bit field in the Channel n Control register (CHANNEL[n].EVGEN). By default, all channels are disabled, meaning they are not connected to any event generator (CHANNEL[n].EVGEN is '0').

For further details on event generation, refer to the *Events* section in the event generator's peripheral chapter.

The table below shows the available event generators for this device family.

Table 24-2. Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
OSC32KCTRL	CLKFAIL	XOSC32K Failure	Level	OSC32K	As long as INTFLAG.CLKFAIL is set
SUPC	MVIO	The voltage level on VDDIO2 is insufficient for communication	Level	CLK_SUPC_APB	As long as INTFLAG.VDDIO2OK is set
	VLM	Voltage Level Monitor is triggered	Level	CLK_SUPC_APB	As long as INTFLAG.VLM is set
RTC	OVF	Overflow	Pulse	CLK_RTC_APB	One CLK_RTC_APB period
	CMP_n	MODE0/MODE1: Compare n MODE2: Alarm n	Pulse	CLK_RTC_APB	One CLK_RTC_APB period
	PER_n	Period Interval n	Pulse	CLK_RTC_APB	One CLK_RTC_APB period
EIC	EXTINT_n	External Interrupt Event Output	Pulse	CLK_EIC_APB	One CLK_EIC_APB period
DMAC	CH_n	Transfer on channel n is complete	Pulse	CLK_DMAC_AHB	One CLK_DMAC_AHB period
TCn	OVF	Overflow/ Underflow	Pulse	GCLK_TCn	One GCLK_TCn period
	MC_n	Match/Compare n	Pulse	GCLK_TCn	One GCLK_TCn period

Table 24-2. Event Generators (continued)

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCCn	OVF	Overflow/ Underflow	Pulse	GCLK_TCCn	One GCLK_TCCn period
	TRG	Trigger	Pulse	GCLK_TCCn	One GCLK_TCCn period
	CNT	Count	Pulse	GCLK_TCCn	One GCLK_TCCn period
	MC_n	Match/Compare n	Pulse	GCLK_TCCn	One GCLK_TCCn period
ADCn	RESRDY	Result Ready	Pulse	CLK_ADCn_APB	One CLK_ADCn_APB period
	SAMPRDY	Sample Ready	Pulse	CLK_ADCn_APB	One CLK_ADCn_APB period
	WCMP	Window Compare	Pulse	CLK_ADCn_APB	One CLK_ADCn_APB period
ACn	COMP_n	Comparator n status	Level	Asynchronous	While COMPn-OUT is '1'
	WIN_n	Window n inside/ outside status	Level	Asynchronous	While the input signal is inside window n
CCL	LUTOUT_n	LUT Out n	Level	Asynchronous	As long as OUT is '1'
PAC	ACCERR	Access Error	Level	CLK_PAC_APB	As long as a interrupt flag is set

24.4.3. Additional Features

24.4.3.1. Overrun Channel Detection

If events occur faster than the event user of channel n is able to service them EVSYS will detect this as a channel overrun, and the Overrun Channel n bit in the Interrupt Flag Status and Clear register (INTFLAG.OVRn) will be set. Overrun channel detection can occur when:

- One or more event users on channel n are not ready when a new event occurs
- An event occurs when the previous event on channel n is not yet been handled by all event users connected channel n

The flag will only be set when using synchronous or resynchronized paths. When using the asynchronous path, INTFLAG.OVRn will always read as '0'.

If several events are received by the event channel in less than three GCLK_EVSYS_CHn periods, the overrun will not be detected, and only the first event will be serviced.

When using the synchronous path, ensure that the generic clock (GCLK_EVSYS_CHn) is always on by writing the On Demand bit in the Channel n register (CHANNEL[n].ONDEMAND) to '0'.

24.4.3.2. Channel Event Detection

When the peripheral connected to channel n generates an event, EVSYS will detect the event and the Event Detected Channel n bit in the Interrupt Flag Status and Clear register (INTFLAG.EVDn) will be set.

The flag will only be set when using a synchronous or resynchronized path. When using the asynchronous path, INTFLAG.EVDn will always read as '0'.

24.4.3.3. Software Event

A software event can be initiated on a channel by writing the Channel n bit in the Software Event register (SWEVT.CHANNELn) to '1', which will generate an event on channel n.

When using a software event on a channel with a resynchronized path, the CHSTATUS.CHBUSYn bit will not be set immediately. Wait three GCLK_EVSYS_CH_n clock cycles for the CHSTATUS.CHBUSYn bit to be set before issuing a new software event.

24.4.3.4. Channel Status

The Channel Status (CHSTATUS) register shows the status of the channels when using a synchronous or resynchronized path. There are two different status bits in CHSTATUS for each of the available channels:

- The CHSTATUS.CHBUSYn bit will be set when an event on the corresponding channel n has not been handled by all event users connected to that channel
- The CHSTATUS.USRRDYn bit will be set when all event users connected to the corresponding channel are ready to handle incoming events on that channel

24.4.4. Sleep Mode Operation

The EVSYS can generate interrupts to wake up the device from any sleep mode.

Event channel n will behave differently in different sleep modes depending on the combination of CHANNEL[n].PATH, CHANNEL[n].RUNSTDBY, and CHANNEL[n].ONDEMAND, as shown in the table below:

Table 24-3. Event Channel Sleep Behavior

CHANNEL[n].PATH	CHANNEL[n].ONDEMAND	CHANNEL[n].RUNSTDBY	Sleep Behavior
ASYNC(1)	0	0	Channel n will be available in both Idle and Standby sleep modes, but it will not wake up the device
SYNC/RESYNC	0	1	In both Idle and Standby Sleep modes, channel n will continuously request GCLK_EVSYS_CHn
RESYNC(2)	1	0	In Idle sleep mode, channel n will request GCLK_EVSYS_CHn only when an incoming event needs to be serviced
RESYNC(2)	1	1	In both Idle and Standby sleep modes, channel n will request GCLK_EVSYS_CHn only when an incoming event needs to be serviced

Notes:

1. Using an asynchronous path, with CHANNEL[n].RUNSTDBY = 1 and CHANNEL[n].ONDEMAND = 1 is not recommended. Refer to the bit field descriptions for further information.
2. With a synchronous path, CHANNEL[n].ONDEMAND = 1 is not recommended.

24.4.5. 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 periodic servicing by the CPU through interrupts or similar mechanisms, improper operation or data loss may occur during debugging.

24.5. Dependencies

24.5.1. I/O Lines

Not applicable.

24.5.2. Power Management

EVSYS interrupts 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 the *PM – Power Manager* chapter for details on the different sleep modes.

24.5.3. Clocks

The EVSYS bus clock (CLK_EVSYS_APB) can be enabled and disabled in the Main Clock (MCLK), and its default state can be found in the *Peripheral Clock Masking* section of the *MCLK – Main Clock* chapter.

Each EVSYS channel has a dedicated generic clock (GCLK_EVSYS_CHn), which is used for event detection and propagation for each channel. These clocks must be configured and enabled in the generic clock controller before using the EVSYS. Refer to the *GCLK – Generic Clock Controller* chapter for details.

24.5.4. DMA

Not applicable.

24.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use EVSYS interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC – Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the EVSYS is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 24-4. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
EVSYS	OVRn	An event occurs before the previous event on channel n has been handled by all event users, or when any of the event users are not ready. Refer to Overrun Channel Detection for further information.	The channel path synchronizer is configured to be either synchronous or resynchronized in the Path Selection bit field in Channel n Control register (CHANNEL[n].PATH is 0x0 or 0x1)
	EVDn	An event from the event generator configured for channel n is detected. Refer to Channel Event Detection for further information.	

24.5.6. Events

Not applicable.

24.5.7. Analog Connections

Not applicable.

24.6. Register Summary - EVSYS

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0								SWRST
0x01 ... 0x0B	Reserved									
0x0C	CHSTATUS	7:0					USRRDY3	USRRDY2	USRRDY1	USRRDY0
		15:8								
		23:16					CHBUSY3	CHBUSY2	CHBUSY1	CHBUSY0
		31:24								
0x10	INTENCLR	7:0					OVR3	OVR2	OVR1	OVR0
		15:8								
		23:16					EVD3	EVD2	EVD1	EVD0
		31:24								
0x14	INTENSET	7:0					OVR3	OVR2	OVR1	OVR0
		15:8								
		23:16					EVD3	EVD2	EVD1	EVD0
		31:24								
0x18	INTFLAG	7:0					OVR3	OVR2	OVR1	OVR0
		15:8								
		23:16					EVD3	EVD2	EVD1	EVD0
		31:24								
0x1C	SWEVT	7:0					CHANNEL3	CHANNEL2	CHANNEL1	CHANNEL0
		15:8								
		23:16								
		31:24								
0x20	CHANNEL[0]	7:0		EVGEN[6:0]						
		15:8	ONDEMAND	RUNSTDBY			EDGSEL[1:0]		PATH[1:0]	
		23:16								
		31:24								
0x24	CHANNEL[1]	7:0		EVGEN[6:0]						
		15:8	ONDEMAND	RUNSTDBY			EDGSEL[1:0]		PATH[1:0]	
		23:16								
		31:24								
0x28	CHANNEL[2]	7:0		EVGEN[6:0]						
		15:8	ONDEMAND	RUNSTDBY			EDGSEL[1:0]		PATH[1:0]	
		23:16								
		31:24								
0x2C	CHANNEL[3]	7:0		EVGEN[6:0]						
		15:8	ONDEMAND	RUNSTDBY			EDGSEL[1:0]		PATH[1:0]	
		23:16								
		31:24								
0x30 ... 0x7F	Reserved									
0x80	USER[0]	7:0				CHANNEL[4:0]				
		15:8								
		23:16								
		31:24								
...										
0xD4	USER[21]	7:0				CHANNEL[4:0]				
		15:8								
		23:16								
		31:24								

24.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the EVSYS to their initial state.
Note: Before applying a Software Reset, it is recommended to disable the event channels.

24.6.2. Channel Status

Name: CHSTATUS
Offset: 0x0C
Reset: 0x0000000F
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					CHBUSY3	CHBUSY2	CHBUSY1	CHBUSY0
Access					R	R	R	R
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					USRRDY3	USRRDY2	USRRDY1	USRRDY0
Access					R	R	R	R
Reset					1	1	1	1

Bits 16, 17, 18, 19 – CHBUSYn Channel n Busy
This bit is cleared when channel n is idle.
This bit is set if an event on channel n has not been handled by all event users connected to channel n. Refer to Event System Channel for further information.

Note: When the event channel path is asynchronous, this bit will always read as '0'

Bits 0, 1, 2, 3 – USRRDYn Users Ready on Channel n
This bit is cleared when at least one of the event users connected to channel n is not ready.
This bit is set when all event users connected to channel n are ready to handle incoming events.
Refer to Event System Channel for further information.

Note: When the event channel path is asynchronous, this bit will always read as '0'

24.6.3. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x10
Reset: 0x00000000
Property: PAC Write-Protection

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

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					EVD3	EVD2	EVD1	EVD0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OVR3	OVR2	OVR1	OVR0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – EVDn Event Detected Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Event Detected Channel n Interrupt Enable bit, which disables the Event Detected Channel n interrupt.

Value	Description
0	The Event Detected Channel n interrupt is disabled
1	The Event Detected Channel n interrupt is enabled

Bits 0, 1, 2, 3 – OVRn Overrun Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overrun Channel n Interrupt Enable bit, which disables the Overrun Channel n interrupt.

Value	Description
0	The Overrun Channel n interrupt is disabled
1	The Overrun Channel n interrupt is enabled

24.6.4. Interrupt Enable Set

Name: INTENSET
Offset: 0x14
Reset: 0x00000000
Property: PAC Write-Protection

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

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					EVD3	EVD2	EVD1	EVD0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OVR3	OVR2	OVR1	OVR0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – EVDn Event Detected Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Event Detected Channel n Interrupt Enable bit, which enables the Event Detected Channel n interrupt.

Value	Description
0	The Event Detected Channel n interrupt is disabled.
1	The Event Detected Channel n interrupt is enabled.

Bits 0, 1, 2, 3 – OVRn Overrun Channel n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Overrun Channel n Interrupt Enable bit, which enables the Overrun Channel n interrupt.

Value	Description
0	The Overrun Channel n interrupt is disabled
1	The Overrun Channel n interrupt is enabled

24.6.5. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x18
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					EVD3	EVD2	EVD1	EVD0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OVR3	OVR2	OVR1	OVR0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – EVDn Event Detected Channel n
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_EVSYS_APB cycle after an event is propagated through the channel, and an interrupt request will be generated if INTENSET.EVDn is '1'. Refer to Channel Event Detection for further information.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Event Detected Channel x interrupt flag.

Note: When the event channel path is asynchronous, the EVDn interrupt flag will not be set.

Bits 0, 1, 2, 3 – OVRn Overrun Channel n
This flag is cleared by writing a '1' to it.
This flag is set on the next CLK_EVSYS_APB cycle after an overrun channel condition occurs, and an interrupt request will be generated if INTENSET.OVRn is '1'. Refer to Overrun Channel Detection for further information.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Overrun Detected Channel n interrupt flag.

Note: When the event channel path is asynchronous, the OVRn interrupt flag will not be set.

24.6.6. Software Event

Name: SWEVT
Offset: 0x1C
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHANNEL3	CHANNEL2	CHANNEL1	CHANNEL0
Access					W	W	W	W
Reset					0	0	0	0

Bits 0, 1, 2, 3 – CHANNELn Channel n Software Selection
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will trigger a software event for channel n.

24.6.7. Channel n Control

Name: CHANNEL[n]
Offset: 0x20 + n*0x04 [n=0..3]
Reset: 0x00008000
Property: PAC Write-Protection

This register allows the user to configure channel n. To write to this register, perform a single 32-bit write of all the configuration data.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ONDEMAND	RUNSTDBY			EDGSEL[1:0]		PATH[1:0]	
Access	R/W	R/W			R/W	R/W	R/W	R/W
Reset	1	0			0	0	0	0

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

Bit 15 – ONDEMAND Generic Clock On Demand
This bit controls whether the generic clock for channel n (GCLK_EVSYS_CHn) is requested on demand.

Value	Description
0	GCLK_EVSYS_CHn is always on if channel n is configured (CHANNEL[n].EVGEN is not 0x00) and the generic clock source is enabled
1	GCLK_EVSYS_CHn is only on when requested by channel n

Note: This bit has no effect when channel n is configured as an asynchronous path.

Bit 14 – RUNSTDBY Run in Standby
This bit controls the behavior during Standby sleep mode.

Value	Description
0	Channel n is disabled in Standby sleep mode when configured with a synchronous or resynchronized path
1	Channel n is not stopped in Standby sleep mode and depends on CHANNEL[n].ONDEMAND

Bits 11:10 – EDGSEL[1:0] Edge Detection Selection
This bit field controls the edge detection for channel n.

Value	Name	Description
0x0	NO_EVT_OUTPUT	No event output when using the resynchronized or synchronous path
0x1	RISING_EDGE	Event detection occurs only on the rising edge of the signal from the event generator
0x2	FALLING_EDGE	Event detection occurs only on the falling edge of the signal from the event generator

Value	Name	Description
0x3	BOTH_EDGES	Event detection occurs on both the rising and falling edges of the signal from the event generator

Note: This bit field must be written to '0' when channel n is configured with asynchronous path.

Bits 9:8 – PATH[1:0] Path Selection
This bit field controls which path will be used for channel n.

Value	Name	Description
0x0	SYNCHRONOUS	Synchronous path
0x1	RESYNCHRONIZED	Resynchronized path
0x2	ASYNCHRONOUS	Asynchronous path
0x3	—	Reserved

Note: The path choice can be limited by the event user. Refer to the *Event Users* section for further information.

Bits 6:0 – EVGEN[6:0] Event Generator
This bit field controls which event generator is connected to channel n.

Value	Name	Description
0x00	OFF	Event channel n is disabled.
0x01	OSC32KCTRL_CLKFAIL	XOSC32K Failure
0x02	SUPC_MVIO	The voltage level on VDDIO2 is insufficient for communication
0x03	SUPC_VLM	Voltage Level Monitor is triggered
0x04	RTC_CMP_0	Compare Match 0 overflow
0x05	RTC_CMP_1	Compare Match 1 overflow
0x06	RTC_OVF	Counter overflow
0x07	RTC_PER_0	Periodic event 0
0x08	RTC_PER_1	Periodic event 1
0x09	RTC_PER_2	Periodic event 2
0x0A	RTC_PER_3	Periodic event 3
0x0B	RTC_PER_4	Periodic event 3
0x0C	RTC_PER_5	Periodic event 5
0x0D	RTC_PER_6	Periodic event 6
0x0E	RTC_PER_7	Periodic event 7
0x0F	EIC_EXTINT_0	Periodic event 3
0x10	EIC_EXTINT_1	External Interrupt 1 Event Output
0x11	EIC_EXTINT_2	External Interrupt 2 Event Output
0x12	EIC_EXTINT_3	External Interrupt 3 Event Output
0x13	EIC_EXTINT_4	External Interrupt 4 Event Output
0x14	EIC_EXTINT_5	External Interrupt 5 Event Output
0x15	EIC_EXTINT_6	External Interrupt 6 Event Output
0x16	EIC_EXTINT_7	External Interrupt 7 Event Output
0x17	EIC_EXTINT_8	External Interrupt 8 Event Output
0x18	EIC_EXTINT_9	External Interrupt 9 Event Output
0x19	EIC_EXTINT_10	External Interrupt 10 Event Output
0x1A	EIC_EXTINT_11	External Interrupt 11 Event Output
0x1B	EIC_EXTINT_12	External Interrupt 12 Event Output
0x1C	EIC_EXTINT_13	External Interrupt 13 Event Output
0x1D	EIC_EXTINT_14	External Interrupt 14 Event Output
0x1E	EIC_EXTINT_15	External Interrupt 15 Event Output
0x1F	DMAC_CH_0	Transfer on channel 0 is complete
0x20	DMAC_CH_1	Transfer on channel 1 is complete
0x21	TC0_OVF	TC0 Overflow

Value	Name	Description
0x22	TC0_MC_0	TC0 Match/Compare 0
0x23	TC0_MC_1	TC0 Match/Compare 1
0x24	TC1_OVF	TC1 Overflow
0x25	TC1_MC_0	TC1 Match/Compare 0
0x26	TC1_MC_1	TC1 Match/Compare 1
0x27	TC2_OVF	TC2 Overflow/Underflow
0x28	TC2_MC_0	TC2 Match/Compare 0
0x29	TC2_MC_1	TC2 Match/Compare 1
0x2A	TCC0_OVF	TCC0 Overflow/Underflow
0x2B	TCC0_TRG	TCC0 Trigger
0x2C	TCC0_CNT	TCC0 Count
0x2D	TCC0_MC_0	TCC0 Match/Compare 0
0x2E	TCC0_MC_1	TCC0 Match/Compare 1
0x2F	TCC0_MC_2	TCC0 Match/Compare 2
0x30	TCC0_MC_3	TCC0 Match/Compare 3
0x31	ADC0_RESRDY	ADC0 Result ready
0x32	ADC0_SAMPRDY	ADC0 Sample ready
0x33	ADC0_WCMP	ADC0 Window compare
0x34	AC0_COMP_0	AC0 Compare 0
0x35	AC0_COMP_1	AC0 Compare 1
0x36	AC0_WIN_0	AC0 Window n inside/outside status
0x37	CCL_LUTOUT_0	CCL LUT Out 0
0x38	CCL_LUTOUT_1	CCL LUT Out 1
0x39	CCL_LUTOUT_2	CCL LUT Out 2
0x3A	CCL_LUTOUT_3	CCL LUT Out 3
0x3C	PAC_ACCERR	PAC Access Error
Other	—	Reserved

24.6.8. User n Channel Selection

Name: USER[n]
Offset: 0x80 + n*0x04 [n=0..21]
Reset: 0x00000000
Property: PAC Write-Protection

Refer to the *Event Users* section for the complete list of available USER[n] registers for this device.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bits 4:0 – CHANNEL[4:0] Channel Event Selection
This bit field controls which event channel USER[n] is connected to.

Value	Name	Description
0x00	OFF	The Event System user is not connected to any channel
0x01	CHANNEL0	The Event System user is connected to Channel 0
0x02	CHANNEL1	The Event System user is connected to Channel 1
0x03	CHANNEL2	The Event System user is connected to Channel 2
0x04	CHANNEL3	The Event System user is connected to Channel 3
Other	—	Reserved

25. EIC – External Interrupt Controller

25.1. Features

- Up to 16 External Interrupt Pins (EXTINTn), Plus One Non-Maskable Interrupt Pin (NMI)
- Dedicated, Individually Maskable Interrupt for Each Pin
- Interrupt on Rising, Falling, or Both Edges
- Synchronous or Asynchronous Edge Detection Mode
- Interrupt on High or Low Levels
- Asynchronous Interrupts for Sleep Modes Without Clock
- Filtering of External Pins
- Event Generation From EXTINTn

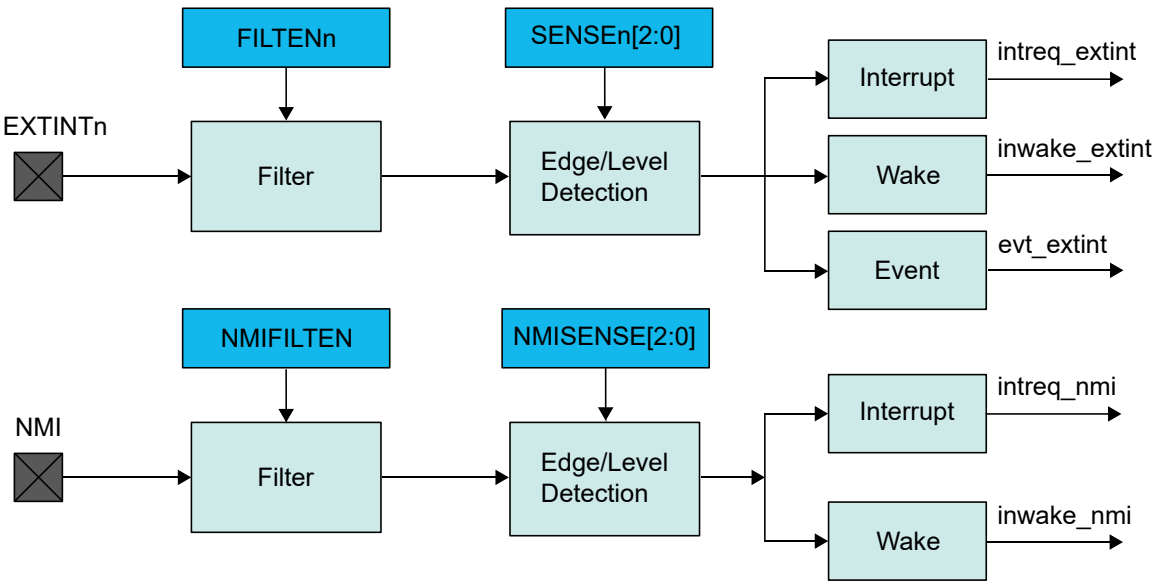
25.2. Overview

The External Interrupt Controller (EIC) allows external pins to be configured as interrupt lines. Each external pin can also generate an event.

A separate Non-Maskable Interrupt (NMI) connected to the NMI request of the CPU is also supported. The NMI is able to interrupt any other interrupt mode.

25.3. Block Diagram

Figure 25-1. EIC Block Diagram



25.3.1. Signal Description

Signal Name	Type	Description
EXTINT[15..0]	Digital input	External interrupt pin
NMI	Digital input	Non-maskable interrupt pin

One signal may be available on several pins.

25.4. Functional Description

25.4.1. Initialization

The EIC must be initialized in the following order:

1. Enable CLK_EIC_APB.
2. If required, configure the NMI by writing the Non-Maskable Interrupt Control (NMICTRL) register.
3. Enable GCLK_EIC or CLK_OSC32K when one of the following configurations is selected:
   - The NMI uses edge detection or filtering
   - One or more EXTINT uses filtering
   - One or more EXTINT uses edge detection

   GCLK_EIC is used when a frequency higher than 32.768 kHz is required for filtering.

   CLK_OSC32K is recommended when the power consumption is the priority. For CLK_OSC32K, write a '1' to the Clock Selection (CKSEL) bit in the Control A (CTRLA) register.
4. Configure the EIC input sense and filtering by writing the Configuration (CONFIG0 or CONFIG1) register.
5. Enable the EIC by writing a '1' to the Enable (ENABLE) bit in the Control A (CTRLA) register.

25.4.2. Operation

The EIC detects edge or level condition to generate interrupts to the CPU interrupt controller or events to the Event System (EVSYS). 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 interrupt pin (EXTINT) has a configurable filter to remove spikes.

Each external pin can also be configured to be asynchronous to wake up the device from sleep modes where all clocks have been disabled.

25.4.2.1. Enabling, Disabling and Resetting

The EIC is enabled or disabled by writing a '1' for enable or a '0' for disable to the Enable (ENABLE) bit in the Control A (CTRLA) register.

The EIC is reset by setting the Software Reset (SWRST) bit in the Control A (CTRLA) register. All registers in the EIC will be reset to their initial state, and the EIC will be disabled.

Refer to the CTRLA register description section for details.

25.4.2.2. 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 interrupt pins is configured by writing the Input Sense n (SENSEn) bit field in the CONFIG0 or CONFIG1 register. The corresponding interrupt (EXTINT[n]) flag in the Interrupt Flag Status and Clear (INTFLAG) register is set when the interrupt condition is met.

When the interrupt flag has been cleared in Edge-sensitive mode, INTFLAG.EXTINT[n] will only be set if a new interrupt condition is met.

In level-sensitive mode, when the interrupt has been cleared, INTFLAG.EXTINT[n] will be set immediately if the EXTINTn pin still matches the interrupt condition.

Each external pin can be filtered by a majority vote filtering, clocked by GCLK_EIC or CLK_OSC32K. Filtering is enabled if the Filter Enable n (FILTENn) bit in the CONFIG0 or CONFIG1 register is written to '1'. The majority vote filter samples the external pin three times with GCLK_EIC or CLK_OSC32K and outputs the value when two or more samples are equal.

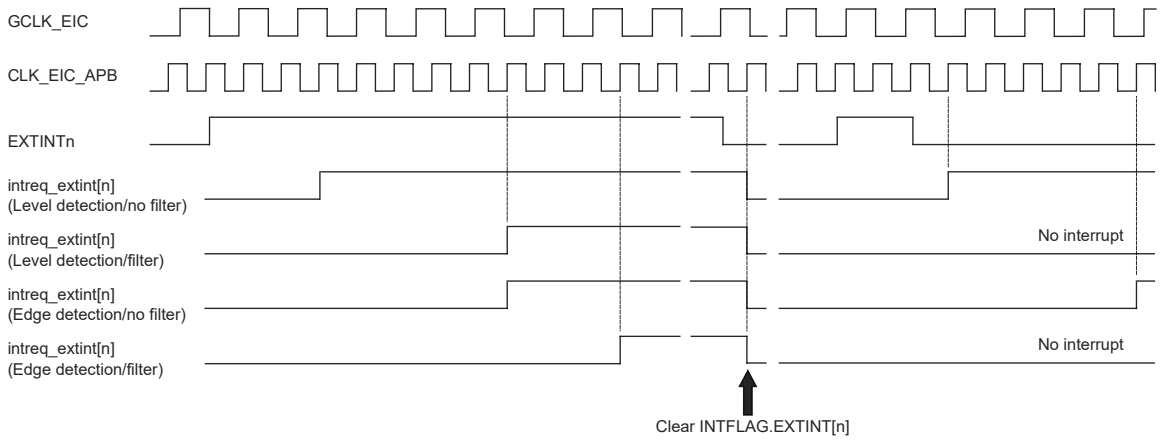
Table 25-1. Majority Vote Filter Logic

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 and when filtering is disabled, detection is done asynchronously. Level detection and asynchronous edge detection do not require GCLK_EIC or CLK_OSC32K and can generate asynchronous interrupts and events.

If filtering or synchronous edge detection is enabled, the EIC automatically requests GCLK_EIC or CLK_OSC32K to operate. The selection between these two clocks is done by writing to the Clock Selection (CKSEL) bit field in the Control A (CTRLA) register. GCLK_EIC must be enabled in the GCLK module. In these modes, the external pin is sampled at the EIC clock rate, thus, pulses with a duration lower than two EIC clock periods may not be properly detected.

Figure 25-2. Interrupt Detection Latency by Modes (Rising Edge)



Detection latency depends on the detection mode.

Table 25-2. Detection Latency

Detection Mode	Latency (Worst Case)
Level without filter	Five CLK_EIC_APB periods
Level with filter	Four GCLK_EIC/CLK_OSC32K periods + five CLK_EIC_APB periods
Edge without filter	Four GCLK_EIC/CLK_OSC32K periods + five CLK_EIC_APB periods
Edge with filter	Six GCLK_EIC/CLK_OSC32K periods + five CLK_EIC_APB periods

25.4.3. Additional Features

25.4.3.1. Non-Maskable Interrupt (NMI)

The non-maskable interrupt pin can also generate an interrupt-on-edge or level detection, but it is configured with the dedicated NMI Control (NMICTRL) register. To select the sense for NMI, write to the NMISENSE bit group in the NMI Control (NMICTRL) register. NMI filtering is enabled by writing a '1' to the NMI Filter Enable (NMIFILTEN) bit in the NMI Control (NMICTRL) register.

If edge detection or filtering is required, enable GCLK_EIC or CLK_OSC32K.

NMI detection is enabled only by the NMICTRL.NMISENSE value, and the EIC module is not required to be enabled.

When an NMI is detected, the non-maskable interrupt flag in the NMI Flag Status and Clear (NMIFLAG) register is set. NMI interrupt generation is always enabled, and NMIFLAG.NMI generates an interrupt request when set.

25.4.3.2. Asynchronous Edge Detection Mode

The EXTINT edge detection can operate synchronously by writing a '0' (default value) to the Asynchronous Edge Detection Mode (ASYNCH[n]) bit for the external pin 'n' in the External Interrupt Asynchronous Mode (ASYNCH) register, or asynchronously by writing a '1' to the Asynchronous Edge Detection Mode (ASYNCH[n]) bit in the ASYNCH register.

In Synchronous Edge Detection mode, the external interrupt (EXTINT) or non-maskable interrupt (NMI) pins are sampled using the EIC clock, as defined by the Clock Selection (CKSEL) bit in the Control A (CTRLA) register. The External Interrupt flag (INTFLAG.EXTINT[n]) or Non-Maskable Interrupt flag (NMIFLAG.NMI) is set when the last sampled state of the pin differs from the previously sampled state. The EIC clock is required in this mode.

The Synchronous Edge Detection mode can be used in the Idle and Standby sleep modes.

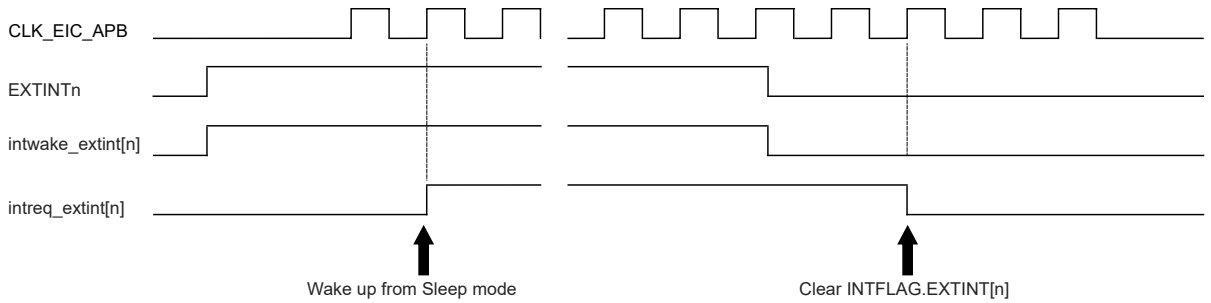
In Asynchronous Edge Detection mode, the external interrupt (EXTINTn) pins or the non-maskable interrupt (NMI) pins set the External Interrupt flag or Non-Maskable Interrupt flag (INTFLAG.EXTINT[n] or NMIFLAG) directly. The EIC clock is not required in this mode.

The asynchronous edge detection mode can be used in Idle and Standby sleep modes. When asynchronous edge detection is enabled in Standby sleep mode, only the first edge detected will trigger an event in the Event System. Subsequent asynchronous edges will not generate events until Standby sleep mode is exited. Synchronous edge detection will not exhibit this behavior.

25.4.4. Sleep Mode Operation

In sleep modes, an EXTINTn pin can wake up the device if the corresponding condition matches the configuration in the CONFIG0 or CONFIG1 register, and the corresponding bit in the Interrupt Enable Set (INTENSET) register is written to '1'.

Figure 25-3. Wake-up Operation Example (High-Level Detection, No Filter, Interrupt Enable Set)



25.4.5. Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset (SWRST) bit in the Control A (CTRLA) register
- Enable (ENABLE) bit in the Control A (CTRLA) register

Required write-synchronization is denoted by the “Write-Synchronized” property in the register description.

25.5. Dependencies

25.5.1. I/O Lines

Using the EIC’s I/O lines requires the I/O pins to be configured. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

25.5.2. Power Management

All interrupts are available down to Standby sleep mode, but the EIC can be configured to automatically mask some interrupts 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 the *PM – Power Manager* chapter for details on the different sleep modes.

25.5.3. Clocks

The EIC bus clock (CLK_EIC_APB) can be enabled and disabled by the Main Clock Controller. The default state of CLK_EIC_APB can be found in the *Peripheral Clock Masking* section.

Some optional functions require a peripheral clock, which can be either a generic clock (GCLK_EIC, for wider frequency selection) or a 32.768 kHz clock (CLK_OSC32K). One of the clock sources must be configured and enabled before using the peripheral:

- GCLK_EIC is configured and enabled in the Generic Clock Controller
- CLK_OSC32K is provided by the internal 32.768 kHz (OSC32K) oscillator in the OSC32KCTRL module

Both GCLK_EIC and CLK_OSC32K are asynchronous to the user interface clock (CLK_EIC_APB). Due to this asynchronicity, writes to certain registers require synchronization between the clock domains.

25.5.4. DMA

Not applicable.

25.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use EIC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a ‘1’ to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a ‘1’ 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Note:
The NMI interrupt request line is connected to the NVIC, but does not require the NVIC to be configured in advance.

Table 25-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
NMI	NMI	NMI pin matches the NMI sense configuration	The Non-Maskable Interrupt Sense Configuration in the Non-Maskable Interrupt Control register (NMICTRL.NMISENSE)
EIC	EXTINTn	The EXTINTn pin matches the external interrupt sense configuration	The Input Sense Configuration n bits in the External Interrupt Sense Configuration n registers (CONFIGn.SENSEn)

25.5.6. Events

The EIC can generate the following events:

Table 25-4. Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
EIC	EXTINT_n	External Interrupt Event Output	Pulse	CLK_EIC_APB	One CLK_EIC_APB period

Writing a '1' to an Event Output Enable bit in the Event Control Register (EVCTRL.EXTINTEOn) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

25.5.7. Analog Connections

Not applicable.

25.6. Register Summary - EIC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0				CKSEL			ENABLE	SWRST
0x01	NMICTRL	7:0				NMIASYNCH	NMIFILTEN	NMISENSE[2:0]		
0x02	NMIFLAG	7:0								NMI
		15:8								
0x04	SYNCBUSY	7:0							ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x08	EVCTRL	7:0	EXTINTEO7	EXTINTEO6	EXTINTEO5	EXTINTEO4	EXTINTEO3	EXTINTEO2	EXTINTEO1	EXTINTEO0
		15:8	EXTINTEO15	EXTINTEO14	EXTINTEO13	EXTINTEO12	EXTINTEO11	EXTINTEO10	EXTINTEO9	EXTINTEO8
		23:16								
		31:24								
0x0C	INTENCLR	7:0	EXTINT7	EXTINT6	EXTINT5	EXTINT4	EXTINT3	EXTINT2	EXTINT1	EXTINT0
		15:8	EXTINT15	EXTINT14	EXTINT13	EXTINT12	EXTINT11	EXTINT10	EXTINT9	EXTINT8
		23:16								
		31:24								
0x10	INTENSET	7:0	EXTINT7	EXTINT6	EXTINT5	EXTINT4	EXTINT3	EXTINT2	EXTINT1	EXTINT0
		15:8	EXTINT15	EXTINT14	EXTINT13	EXTINT12	EXTINT11	EXTINT10	EXTINT9	EXTINT8
		23:16								
		31:24								
0x14	INTFLAG	7:0	EXTINT7	EXTINT6	EXTINT5	EXTINT4	EXTINT3	EXTINT2	EXTINT1	EXTINT0
		15:8	EXTINT15	EXTINT14	EXTINT13	EXTINT12	EXTINT11	EXTINT10	EXTINT9	EXTINT8
		23:16								
		31:24								
0x18	ASYNCH	7:0	ASYNCH7	ASYNCH6	ASYNCH5	ASYNCH4	ASYNCH3	ASYNCH2	ASYNCH1	ASYNCH0
		15:8	ASYNCH15	ASYNCH14	ASYNCH13	ASYNCH12	ASYNCH11	ASYNCH10	ASYNCH9	ASYNCH8
		23:16								
		31:24								
0x1C	CONFIG0	7:0	FILTEN1	SENSE1[2:0]			FILTEN0	SENSE0[2:0]		
		15:8	FILTEN3	SENSE3[2:0]			FILTEN2	SENSE2[2:0]		
		23:16	FILTEN5	SENSE5[2:0]			FILTEN4	SENSE4[2:0]		
		31:24	FILTEN7	SENSE7[2:0]			FILTEN6	SENSE6[2:0]		
0x20	CONFIG1	7:0	FILTEN9	SENSE9[2:0]			FILTEN8	SENSE8[2:0]		
		15:8	FILTEN11	SENSE11[2:0]			FILTEN10	SENSE10[2:0]		
		23:16	FILTEN13	SENSE13[2:0]			FILTEN12	SENSE12[2:0]		
		31:24	FILTEN15	SENSE15[2:0]			FILTEN14	SENSE14[2:0]		

25.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: PAC Write-Protection, Write-Synchronized

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

Bit 4 – CKSEL Clock Selection
The EIC can be clocked either by GCLK_EIC (when a frequency higher than 32.768 kHz is required for filtering) or by CLK_OSC32K (when power consumption is the priority).
Note: Any writes to this bit while synchronization of CTRLA.ENABLE or CTRLA.SWRST is ongoing, or the CTRLA.ENABLE bit is '1', will cause a peripheral bus error. However, this bit itself is not synchronized.

Value	Name	Description
0	GCLK_EIC	The EIC is clocked by GCLK_EIC
1	CLK_OSC32K	The EIC is clocked by CLK_OSC32K

Bit 1 – ENABLE Enable
Note: This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure the CTRLA.ENABLE synchronization is complete. Any writes to this bit while synchronization is ongoing will cause a peripheral bus error.

Value	Description
0	The EIC is disabled
1	The EIC is enabled

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the EIC to their initial state, and the EIC will be disabled.
Writing a '1' to this bit will always take precedence, meaning that all other writes in the same write operation will be discarded.
Note: This bit is write-synchronized: SYNCBUSY.SWRST must be checked to ensure the CTRLA.SWRST synchronization is complete. Any register writes while CTRLA.SWRST is '1' will cause a peripheral bus error.

Value	Description
0	There is no ongoing reset operation
1	The reset operation is ongoing

25.6.2. Non-Maskable Interrupt Control

Name: NMICTRL
Offset: 0x01
Reset: 0x00
Property: PAC Write-Protection

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

Bit 4 – NMIASYNCH NMI Asynchronous Edge Detection Mode
The NMI edge detection can be operated synchronously or asynchronously to the EIC clock.

Value	Description
0	The NMI edge detection is synchronously operated
1	The NMI edge detection is asynchronously operated

Bit 3 – NMIFILTEN Non-Maskable Interrupt Filter Enable

Value	Description
0	NMI filter is disabled
1	NMI filter is enabled

Bits 2:0 – NMISENSE[2:0] Non-Maskable Interrupt Sense Configuration
These bits define on which edge or level the NMI triggers.

Value	Name	Description
0x0	NONE	No detection
0x1	RISE	Rising-edge detection
0x2	FALL	Falling-edge detection
0x3	BOTH	Both-edge detection
0x4	HIGH	High-level detection
0x5	LOW	Low-level detection
0x6-0x7	—	Reserved

25.6.3. Non-Maskable Interrupt Flag Status and Clear

Name: NMIFLAG
Offset: 0x02
Reset: 0x0000
Property: –

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 0 – NMI Non-Maskable Interrupt
Writing a '0' to this flag has no effect.
This flag is set when the NMI pin matches the NMI sense configuration, and will generate an interrupt request.
This flag is cleared by writing a '1' to it.

25.6.4. Synchronization Busy

Name: SYNCBUSY
Offset: 0x04
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ENABLE Enable Synchronization Busy Status

Value	Description
0	Write synchronization for the ENABLE bit is complete
1	Write synchronization for the ENABLE bit is ongoing

Bit 0 – SWRST Software Reset Synchronization Busy Status

Value	Description
0	Write synchronization for the SWRST bit is complete
1	Write synchronization for the SWRST bit is ongoing

25.6.5. Event Control

Name: EVCTRL
Offset: 0x08
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	EXTINTEO15	EXTINTEO14	EXTINTEO13	EXTINTEO12	EXTINTEO11	EXTINTEO10	EXTINTEO9	EXTINTEO8
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
	EXTINTEO7	EXTINTEO6	EXTINTEO5	EXTINTEO4	EXTINTEO3	EXTINTEO2	EXTINTEO1	EXTINTEO0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – EXTINTEOn External Interrupt Event Output Enable n
The bit n of EXTINTEO enables the event associated with the EXTINTn pin.

Value	Description
0	Event from pin EXTINTn is disabled
1	Event from pin EXTINTn is enabled and will be generated when the EXTINTn pin matches the external interrupt sensing configuration

25.6.6. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x0C
Reset: 0x00000000
Property: PAC Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	EXTINT15	EXTINT14	EXTINT13	EXTINT12	EXTINT11	EXTINT10	EXTINT9	EXTINT8
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
	EXTINT7	EXTINT6	EXTINT5	EXTINT4	EXTINT3	EXTINT2	EXTINT1	EXTINT0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – EXTINTn External Interrupt Enable n
The bit n of EXTINT disables the interrupt associated with the EXTINTn pin.
Writing a '0' to bit n has no effect.
Writing a '1' to bit n will clear the External Interrupt Enable bit n, which disables the external interrupt EXTINTn.

Value	Description
0	The external interrupt n is disabled
1	The external interrupt n is enabled

25.6.7. Interrupt Enable Set

Name: INTENSET
Offset: 0x10
Reset: 0x00000000
Property: PAC Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	EXTINT15	EXTINT14	EXTINT13	EXTINT12	EXTINT11	EXTINT10	EXTINT9	EXTINT8
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
	EXTINT7	EXTINT6	EXTINT5	EXTINT4	EXTINT3	EXTINT2	EXTINT1	EXTINT0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – EXTINTn External Interrupt Enable n
The bit n of EXTINT enables the interrupt associated with the EXTINTn pin.
Writing a '0' to bit n has no effect.
Writing a '1' to bit n will set the External Interrupt Enable bit x, which enables the external interrupt EXTINTn.

Value	Description
0	The external interrupt n is disabled
1	The external interrupt n is enabled

25.6.8. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x14
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	EXTINT15	EXTINT14	EXTINT13	EXTINT12	EXTINT11	EXTINT10	EXTINT9	EXTINT8
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
	EXTINT7	EXTINT6	EXTINT5	EXTINT4	EXTINT3	EXTINT2	EXTINT1	EXTINT0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – EXTINTn External Interrupt n
The flag bit n is cleared by writing a '1' to it.
This flag is set when the EXTINTn pin matches the external interrupt sense configuration and will generate an interrupt request if the External Interrupt Enable n (EXTINT[n]) in the Interrupt Enable Set (INTENSET) register is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the External Interrupt n flag.

25.6.9. External Interrupt Asynchronous Mode

Name: ASYNCH
Offset: 0x18
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ASYNCH15	ASYNCH14	ASYNCH13	ASYNCH12	ASYNCH11	ASYNCH10	ASYNCH9	ASYNCH8
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
	ASYNCH7	ASYNCH6	ASYNCH5	ASYNCH4	ASYNCH3	ASYNCH2	ASYNCH1	ASYNCH0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – ASYNCHn Asynchronous Edge Detection Mode n
The bit n of ASYNCH sets the Asynchronous Edge Detection Mode for the interrupt associated with the EXTINTn pin.

Value	Description
0	The EXTINTn edge detection is synchronously operated
1	The EXTINTn edge detection is asynchronously operated

25.6.10. External Interrupt Sense Configuration 0

Name: CONFIG0
Offset: 0x1C
Reset: 0x00000000
Property: PAC Write-Protection, Enable-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 3, 7, 11, 15, 19, 23, 27, 31 – FILTENn Filter Enable n
Note: The filter must be disabled if the asynchronous detection is enabled.

Value	Description
0	Filter is disabled for EXTINT[n] input
1	Filter is enabled for EXTINT[n] input

Bits 0:2, 4:6, 8:10, 12:14, 16:18, 20:22, 24:26, 28:30 – SENSEn Input Sense Configuration n
These bit fields define on which edge or level the interrupt or event for EXTINT[n] will be generated.

Value	Name	Description
0x0	NONE	No detection
0x1	RISE	Rising-edge detection
0x2	FALL	Falling-edge detection
0x3	BOTH	Both-edge detection
0x4	HIGH	High-level detection
0x5	LOW	Low-level detection
0x6-0x7	—	Reserved

25.6.11. External Interrupt Sense Configuration 1

Name: CONFIG1
Offset: 0x20
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
	FILTEN15	SENSE15[2:0]			FILTEN14	SENSE14[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
	FILTEN13	SENSE13[2:0]			FILTEN12	SENSE12[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
	FILTEN11	SENSE11[2:0]			FILTEN10	SENSE10[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
	FILTEN9	SENSE9[2:0]			FILTEN8	SENSE8[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 3, 7, 11, 15, 19, 23, 27, 31 – FILTENn Filter Enable n
Note: The filter must be disabled if the asynchronous detection is enabled.

Value	Description
0	Filter is disabled for EXTINT[n] input
1	Filter is enabled for EXTINT[n] input

Bits 0:2, 4:6, 8:10, 12:14, 16:18, 20:22, 24:26, 28:30 – SENSEn Input Sense Configuration n
These bit fields define on which edge or level the interrupt or event for EXTINT[n] will be generated.

Value	Name	Description
0x0	NONE	No detection
0x1	RISE	Rising-edge detection
0x2	FALL	Falling-edge detection
0x3	BOTH	Both-edge detection
0x4	HIGH	High-level detection
0x5	LOW	Low-level detection
0x6-0x7	—	Reserved

26. NVMCTRL – Non-Volatile Memory Controller

26.1. Features

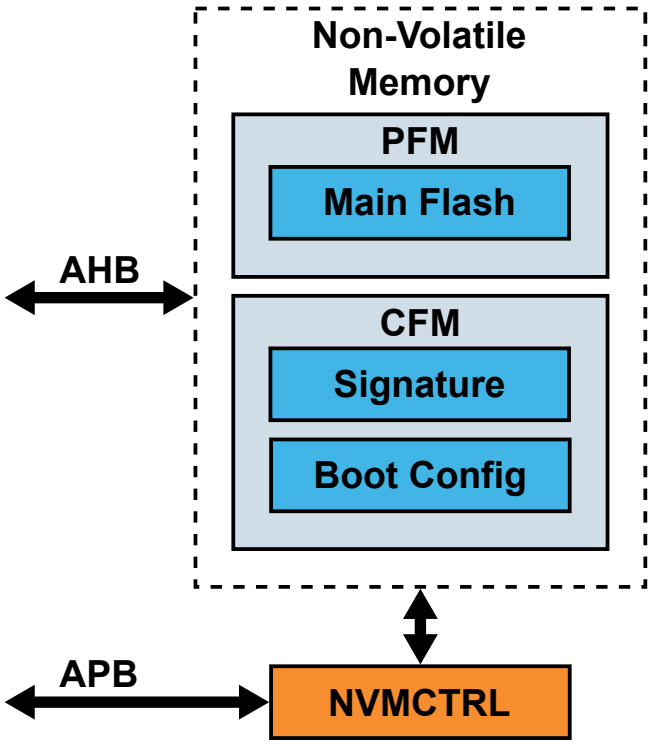
- In-System Programmable
- Self-Programming and Boot Loader Support
- 32-bit AHB Interface for Reads and Writes
- All NVM Sections are Memory Mapped to the AHB, Including Calibration and System Configuration
- 32-bit APB Interface for Commands and Control
- 16 Regions can be Individually Protected or Unprotected Against Erase and Writes
- Additional Protection for the Boot Section Against Erase and Writes
- Signature and Calibration Section for Factory-Programmed Data:
  - ID for each device type
  - Serial number for each device
  - Calibration bytes for factory-calibrated peripherals
- Boot Configuration Section for Application Data:
  - Can be read and written from software
  - Content is retained after chip erase

26.2. Overview

Nonvolatile Memory (NVM) is reprogrammable Flash memory that retains program and data storage even through a power cycle. It embeds a Program Flash Memory (PFM) array and a separate smaller Configuration Flash Memory (CFM) array. A size-configurable section at the beginning of the PFM array can be configured as write protected. The NVM Controller (NVMCTRL) connects to the Advanced High-performance Bus (AHB) and Advanced Peripheral Bus (APB) interfaces, providing system access to the NVM block. The AHB interface is used for direct reads and writes to the NVM block, while the APB interface is used for register access, allowing for commands and configuration.

26.3. Block Diagram

Figure 26-1. Block Diagram



26.3.1. Signal Description

Not applicable.

26.4. Functional Description

26.4.1. Initialization

This module is initialized in the boot ROM according to the configured fuses. If necessary, some settings can be changed by the application after the boot sequence.

26.4.2. Operation

26.4.2.1. Principle of Operation

The NVM Controller is a client on the AHB and APB buses. It responds to commands, read requests, and write requests based on user configuration.

After device reset, 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

26.4.2.2. Memory Organization

The Flash memory is divided into two memory sections.

1. Program Flash Memory (PFM): The main Flash array containing the user application code.
2. Configuration Flash Memory (CFM): A set of Flash pages that store data such as factory calibration, system configuration, and device ID.

The PFM is divided into a set of pages. A page is the basic unit addressed when erasing the Flash. It is only possible to erase a multiple of whole page at a time, while writes are performed per word. One page consists of several words.

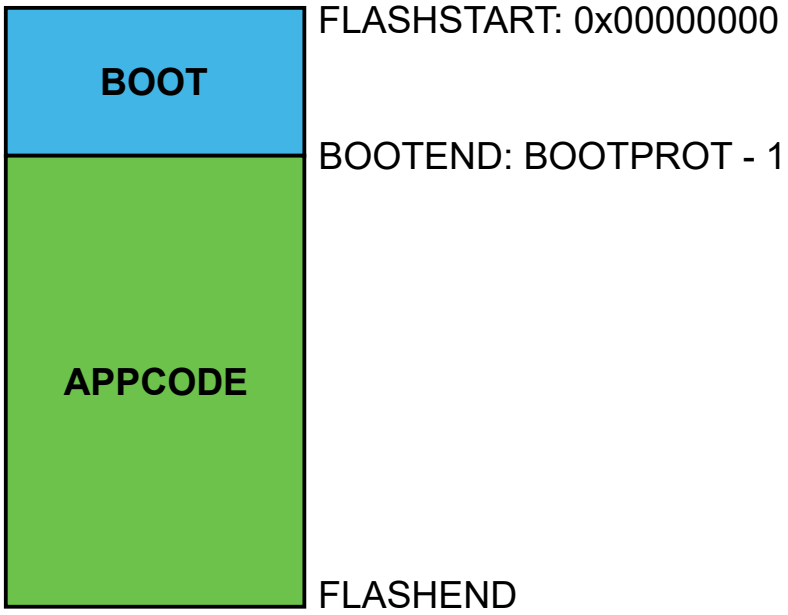
Refer to the *Physical Memory Map* section for memory sizes and addresses for each device.

26.4.2.2.1. Program Flash Memory

The Program Flash Memory section can be divided into two logical sections, in blocks of a configured number of bytes, for different security. The two sections are Boot Code (BOOT) and Application Code (APPCODE).

The sizes of these sections are set by the Boot Section End fuse, which can be read back in the Boot Section Size bit field in the NVM Parameters register (PARAM.BOOTPROT). See PARAM.BOOTPROT for the full configuration list.

Figure 26-2. NVM Memory Address Space



The BOOT section extends from the start of the Flash until BOOTEND. The APPCODE section runs from BOOTEND to FLASHEND. If PARAM.BOOTPROT is read as `0`, the entire Flash is regarded as the APPCODE section.

26.4.2.2.2. Signature

The SIGNATURE row contains a serial number for each manufactured device. The serial number consists of the production lot number, wafer number, and wafer coordinates for the device. The SIGNATURE row cannot be written or erased, but it can be read from application software and external programmers.

26.4.2.2.3. BOOTCFG

The BOOTCFG Row is a Flash section that can be used to store various data, such as configuration calibration data, as well as application serial numbers. This page is not erased by a chip erase.

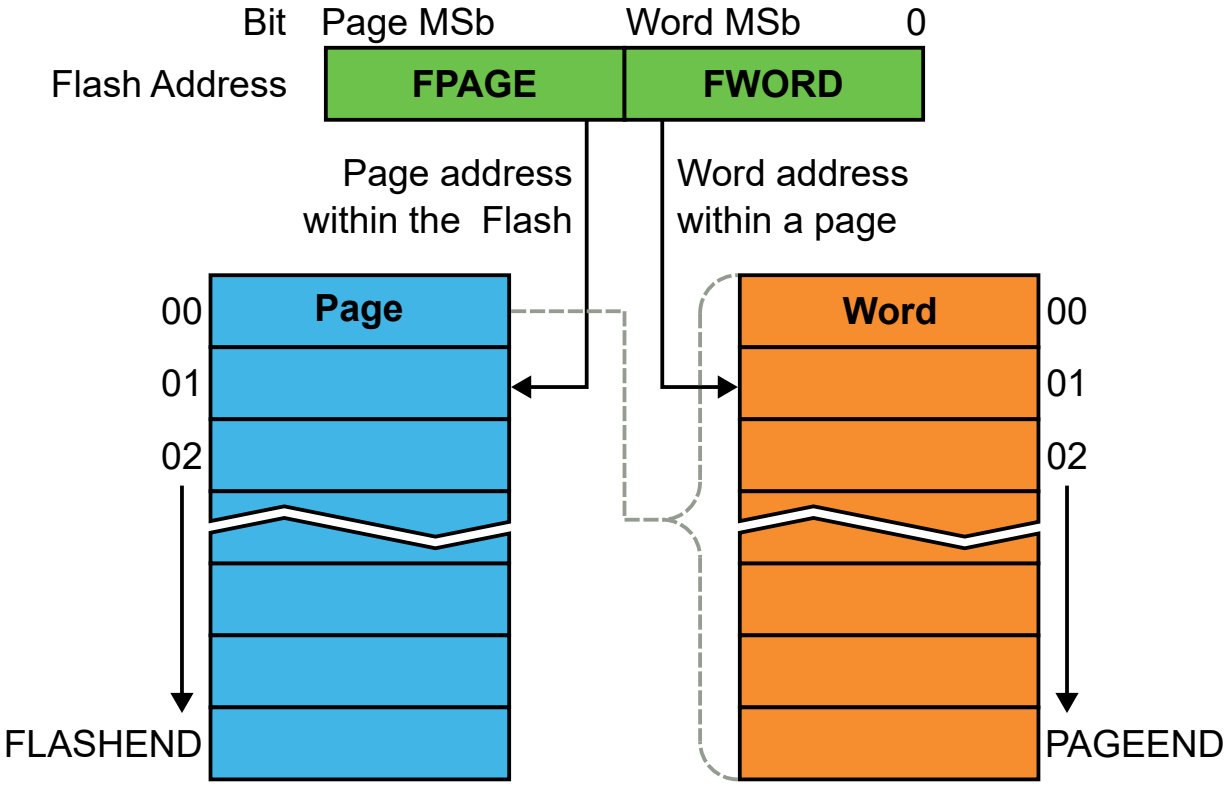
The BOOTCFG Row is erased and written like normal Flash. When erasing the user row the entire row is erased at once.

On a locked device, the BOOTCFG Row can only be written through boot ROM commands.

The BOOTCFG row is not erased by a Chip Erase.

26.4.2.3. Memory Access

Figure 26-3. Flash Addressing for Self-Programming



The Flash is word-accessed and organized in pages, so the Address Pointer can be considered as having two sections, as shown in the previous figure. The word address within the page (FWORD) is held by the Least Significant bits (LSbs) of the Address Pointer, while the Most Significant bits (MSbs) hold the Flash page address (FPAGE). Together, FWORD and FPAGE provide an absolute address to a word in the Flash.

The size of FWORD and FPAGE depend on the page and Flash size of the device.

26.4.2.3.1. Flash Read

Reading from the Flash is performed using AHB bus read transfers. Reading any of the Flash sections while a write or erase operation is in progress on that section will result in a bus wait, and the read will be suspended until the ongoing operation is complete.

Reads can be of byte, halfword or word size.

26.4.2.3.2. Flash Programming

When programming the Flash, it is written one word at a time. Writing is done by writing a 32-bit word to the desired address in the Flash, after programming has been enabled through the Command bit field in the Control B register (CTRLB.CMD). Writes of bytes or half-words to the Flash are discarded, and a bus error is returned.

Attempting to start a programming on a section that is busy will cause the Programming Error Status bit in the Status register (STATUS.PROGE) and the Error flag in the Interrupt Flag Status and Clear register (INTFLAG.ERROR) to be set.

26.4.2.3.3. Flash Erase

The NVM command set supports multiple types of Flash erase operations with different number of pages. This allows several pages to be erased in a single operation. The duration of the erase operation is the same for 1 page as for 32 pages.

Table 26-1. Programming Granularity

Memory Section	Erase Granularity	Write Granularity
PFM section	Page	Word
BOOTCFG Row	Page	Word[1]

Note:
1. BOOTCFG has page granularity when programming on a locked device.

26.4.2.3.4. Command and Data Interface

Reading from the Flash is handled with AHB read transfers. Writing and erasing is performed by writing a command to the Command bit field in the Control B register (CTRLB.CMD), and then writing the desired word data to the desired address in the memory array.

To issue a command, the CTRLB.CMD bits must be written along with the Command Execution bit field in the Control B register (CTRLB.CMDEX). When a command is issued, the Ready flag in the Interrupt Flag Status and Clear register (INTFLAG.READY) will be cleared until the command has completed. Any commands written while INTFLAG.READY is low will be ignored, and the Programming Error Status bit in the Status register (STATUS.PROGE) will be set. If Self-Programming is used, before entering any sleep mode, ensure that any commands written to the NVM controller have completed by confirming that INTFLAG.READY is '1'.

Erasing the entire Flash (i.e. Chip Erase) cannot be done by the application when Debug Access Level (DAL) is 0, it can only be performed through the boot ROM interface from an external debugger or programmer.

To perform an operation (write/erase) in the NVM:

1. Confirm that any previous operation is completed by reading the INTFLAG.READY flag.
2. Unprotect the NVMCTRL registers by clearing the Write Protection Enable bit in the Write Protection Control register (WPCTRL.WPEN).
3. Write the desired command value to the CTRLB.CMD bit field.
4. Write to the correct address in the array to start the operation.
5. Wait for INTFLAG.READY to be set to confirm that the operation is done
6. Write the NOOP or NOCMD command to the CTRLB.CMD bit field to clear the current command.
7. Protect the NVMCTRL registers by setting the WPCTRL.WPEN bit.

To execute a command in the NVM:

1. Confirm that any previous operation is completed by reading the INTFLAG.READY flag.
2. Unprotect the NVMCTRL registers by clearing the WPCTRL.WPEN bit.
3. Write the desired command value to the CTRLB.CMD bit field.

Flash Write Mode

The Flash Write mode (FLWR) of the Flash controller enables writes to the Flash array to start a program operation. Several writes can be done while the Flash write mode is enabled in the NVMCTRL.CTRLB register. When the write mode is enabled, one word is written at a time.

Before a write is performed, the address need to be erased using a suitable command.

If the row resides in a region that is locked (either through BOOTPROT and/or LOCK bits), the write will not be performed, and the Lock Error bit in the Status register (STATUS.LOCKE) set.

Flash Page Erase Mode

The Flash Page Erase mode (FLPER) causes the addressed page to be erased upon a write to the memory array. If the row resides in a region that is locked (either through BOOTPROT and/or LOCK bits), the erase will not be performed, and the Lock Error bit in the Status register (STATUS.LOCKE) will be set.

Flash Multi-page Erase Mode

The Multi-page Erase mode (FLMPERn) causes each write to the memory array to erase multiple pages. This mode can be configured to erase 2, 4, 8, 16 or 32 pages with a single write.

The LSbs of the page address is ignored when determining which Flash pages are erased. This means that, for a 4-page erase, providing an address to a word within page 8 to 11 will cause all pages 8-11 to be erased.

Table 26-2. Flash Multi-page Erase

CMD	Pages Erased	Description
FLMPER2	2	Pages matching FPAGE[N:1] are erased. The value in FPAGE[0] is ignored.
FLMPER4	4	Pages matching FPAGE[N:2] are erased. The value in FPAGE[1:0] is ignored.
FLMPER8	8	Pages matching FPAGE[N:3] are erased. The value in FPAGE[2:0] is ignored.
FLMPER16	16	Pages matching FPAGE[N:4] are erased. The value in FPAGE[3:0] is ignored.
FLMPER32	32	Pages matching FPAGE[N:5] are erased. The value in FPAGE[4:0] is ignored.
Note: FPAGE is the page number when doing a Flash erase. Refer to the Memory Access section for details.		
Note: "N" refers to the MSb of the FPAGE, i.e.: [N:1] refers to all bits in FPAGE except the LSb.		

All pages that are erased at once must be within a single logical NVM section; Boot or application code. If the number of pages selected crosses the boundary between two sections, the operation is aborted and STATUS.LOCKE is set.

If the rows reside in a region that is locked (either through BOOTPROT and/or LOCK bits), the erase will not be performed and the Lock Error bit in the Status register (STATUS.LOCKE) will be set.

Lock and Unlock Region Commands

The Lock (LR) and Unlock (UR) Region commands lock or unlock the region pointed to by the ADDR register. This change is not persistent and will revert to the configuration set in the ROMCFG.NVMLOCKREGION after a device reset. See WLOCKREGION for information on how to change the default values of LOCK.

Refer to the *Region Lock Bit* section for more information on the lockable regions.

BOOTCFG Erase Mode

The BOOTCFG Erase mode (EBOOTCFG) causes the BOOTCFG row to be erased upon a write to the memory array.

BOOTCFG Write Mode

The BOOTCFG Write mode (WBOOTCFG) enables writing to the BOOTCFG row of the CFM section.

ROMCFG Write Mode

The ROMCFG Write mode (WROMCFG) enables writing to the ROMCFG row in the CFM section. Writing to this row is, with one exception, limited to DAL == 2 or SYSINIT privileges.

The NVM Lock Region Write command (WLOCKREGION) enables writing to the ROMCFG.NVMLOCKREGION word. This allows the default states for the region lock bits in the LOCK register to be changed. Writing to this word can be done at any DAL, without any special privileges.

Chip Erase Command

The Chip Erase (CHER) command erases the Flash. All Flash bytes will read back 0xFF after this command. The command can only be executed if DAL == 2 or with SYSINIT privileges.

26.4.2.4. Preventing Flash Corruption

During periods of low V_{DD}, a Flash write or erase can cause memory corruption if the supply voltage is too low for the CPU and the Flash to operate properly. These issues are similar to those encountered in board-level systems using Flash. The internal or external Brown-out Detector (BOD) is recommended to ensure that the operating voltage is high enough.

Two circumstances may cause Flash corruption when the voltage is too low:

1. A regular write sequence to the Flash, which requires a minimum voltage to operate correctly.
2. The CPU can execute instructions incorrectly when the supply voltage is too low.

See the *Electrical Characteristics* chapter for Maximum Frequency vs. V_{DD}.

Attention: Taking the following measures may avoid Flash corruption:

1. Keep the device in Reset during periods of insufficient power supply voltage. Do this by enabling the internal BOD.
2. The Voltage Level Monitor (VLM) in the BOD can be used to prevent starting a write to the Flash close to the BOD level.
3. If the detection levels of the internal BOD do not match the required detection level, an external V_{DD} Reset protection circuit can be used. If a Reset occurs while a write operation is ongoing, the write operation will be aborted.

26.4.3. Other Dependencies

26.4.3.1. Region Lock Bits

The main PFM array is split into 16 equally sized regions. The region size is dependent on the Flash memory size, and is given in the following table. Each region has a dedicated lock bit preventing writing and erasing pages in the region. After production, all regions will be unlocked.

Table 26-3. Region Size

Memory Size [KB]	Region Size [KB]
128	8
64	4

To temporarily lock or unlock a region, the Lock Region (LR) and Unlock Region (UR) commands are provided. Writing one of these commands will temporarily lock or 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 remain in effect until the next Reset, or until it is 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 or unlock setting for a region, the WLOCKREGION command must be used. This command will take effect after the next Reset. Therefore, a device reboot is required for changes in the lock or unlock setting to take effect.

Note: The boot loader section is write protected by the BOOTPROT fuse and by the lock bit(s) corresponding to its address space.

26.4.4. Sleep Mode Operation

The NVMCTRL will enter sleep mode if the system is in sleep and there are no ongoing write or erase operations. DMA requests will be handled in IDLE sleep mode.

Notes:

- Make sure that there is no ongoing erase or write operation before entering STANDBY sleep mode
- The NVM Ready interrupt will wake the device only from Idle sleep mode

26.4.5. Debug Operation

When the CPU is halted in debug mode, the NVMCTRL will halt normal operation.

26.5. Dependencies

26.5.1. I/O Lines

Not applicable.

26.5.2. 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 the device from sleep modes. Refer to the *PM - Power Manager* chapter for details on the different sleep modes.

26.5.3. Clocks

Not applicable.

26.5.4. DMA

Not applicable.

26.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use NVMCTRL interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 26-4. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
NVMCTRL	ERROR	An error flag has been raised in the NVMCTRL	LOCKE and PROGE in the Status register (STATUS.LOCKE and STATUS.PROGE)
	READY	The Flash is ready for new write/erase operations	—

26.5.6. Events

Not applicable.

26.5.7. Analog Connections

Not applicable.

26.6. Register Summary - NVMCTRL

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	CTRLB	7:0		CMD[6:0]						
		15:8	CMDEX[7:0]							
		23:16								
		31:24								
0x08	PARAM	7:0	NVMP[7:0]							
		15:8	NVMP[15:8]							
		23:16						PSZ[2:0]		
		31:24						BOOTPROT[2:0]		
0x0C	INTENCLR	7:0							ERROR	READY
		15:8								
		23:16								
		31:24								
0x10	INTENSET	7:0							ERROR	READY
		15:8								
		23:16								
		31:24								
0x14	INTFLAG	7:0							ERROR	READY
		15:8								
		23:16								
		31:24								
0x18	INTFLAGSET	7:0							ERROR	READY
		15:8								
		23:16								
		31:24								
0x1C	STATUS	7:0					LOCKE	PROGE		
		15:8								
		23:16								
		31:24								
0x20	ADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x24	LOCK	7:0	LOCK7	LOCK6	LOCK5	LOCK4	LOCK3	LOCK2	LOCK1	LOCK0
		15:8	LOCK15	LOCK14	LOCK13	LOCK12	LOCK11	LOCK10	LOCK9	LOCK8
		23:16								
		31:24								
0x28	DATA	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0x2C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

26.6.1. Control B

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 15:8 – CMDEX[7:0] Command Execution

When this bit field is written with the key value `0xA5`, the command written to the Command bit field in the Control B register (CTRLB.CMD) will be executed. If a value different from the key value is used, the write will not be performed, and the Programming Error bit in the Status register (STATUS.PROGE) will be set. STATUS.PROGE is also set if a previously written command has not yet completed.

The key value must be written at the same time as CTRLB.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 both the NVM block and the AHB bus are idle.

The Ready flag in the Interrupt Flag Status and Clear register (INTFLAG.READY) must be '`1`' when the command is issued.

Bit `0` of the CTRLB.CMDEX bit field will read back as '`1`' until the command is issued.

Value	Name	Description
`0xA5`	KEY	Execution key
`Other`	—	Reserved

Bits 6:0 – CMD[6:0] Command

This bit field defines the command to be executed when the correct key is written to the Command Execution bit field in the Command register (CTRLB.CMDEX). Write to this bit field to enable or issue a command. The Chip Erase command is started when the command is written. The others enable an erase or write operation. The operation is started by performing a store to the address location. A change from one command to another should always go through NOCMD or NOOP. If an attempt is made to write a programming command while the Flash is busy, STATUS.PROGE is set.

Value	Name	Description
`0x00`	NOCMD	No command

Value	Name	Description
0x01	NOOP	No operation
0x02	FLWR	Flash Write Enable
0x08	FLPER	Flash Page Erase Enable
0x09	FLMPER2	Flash 2-page Erase Enable
0x0A	FLMPER4	Flash 4-page Erase Enable
0x0B	FLMPER8	Flash 8-page Erase Enable
0x0C	FLMPER16	Flash 16-page Erase Enable
0x0D	FLMPER32	Flash 32-page Erase Enable
0x0E	LR	Lock Region. Sets the bit in the Region n Lock Bits bit field in the Lock Section register (NVMCTRL.LOCK) corresponding to the address location in the ADDR register.
0x0F	UR	Unlock Region. Clears the bit in NVMCTRL.LOCK corresponding to the address location in the ADDR register.
0x10	EBOOTCFG	Erase BOOTCFG Page Enable
0x11	WBOOTCFG	Write BOOTCFG Page Enable
0x12	WLOCKREGION	Write Enable to ROMCFG.NVMLOCKREGION. Writes to other addresses will cause STATUS.PROGE to be set.
0x20	WROMCFG	Write ROMCFG Page Enable. The ROMCFG Page is used by the boot ROM to store Debug Access Level (DAL) bits and other security or protection bits for such as CEHL and Immutable boot.
0x21	CHER	Erase Flash Requires DAL == 2 or SYSINT privileges
Other	—	Reserved

26.6.2. NVM Parameters

Name: PARAM
Offset: 0x08
Reset: 0x0XXXXXXX
Property: –

This register is reset to `0x00000000` but are populated by values set by fuses during the boot sequence.

Bit	31	30	29	28	27	26	25	24
						BOOTPROT[2:0]		
Access						R	R	R
Reset						x	x	x

Bit	23	22	21	20	19	18	17	16
						PSZ[2:0]		
Access						R	R	R
Reset						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	R
Reset	x	x	x	x	x	x	x	x

Bits 26:24 – BOOTPROT[2:0] Boot Section Size
Indicates the number of bytes protected by the boot section.

Value	Name	Description
0x0	SIZE_32768BYTES	32768 bytes are protected
0x1	SIZE_16384BYTES	16384 bytes are protected
0x2	SIZE_8192BYTES	8192 bytes are protected
0x3	SIZE_4096BYTES	4096 bytes are protected
0x4	SIZE_2048BYTES	2048 bytes are protected
0x5	SIZE_1024BYTES	1024 bytes are protected
0x6	SIZE_512BYTES	512 bytes are protected
0x7	SIZE_0BYTES	0 bytes are protected (default)

Bits 18:16 – PSZ[2:0] Page Size
Indicates the page size. Not all devices in a device family will support all the page sizes listed in the table.

Value	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

Value	Name	Description
0x7	1024	1024 bytes

Bits 15:0 – NVMP[15:0] NVM Parameters
Other NVM parameters are read from fuses and written by the boot ROM.

26.6.3. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x0C
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ERROR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Error interrupt enable.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – READY NVM Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the NVM Ready interrupt enable.

Value	Description
0	The NVM Ready interrupt is disabled
1	The NVM Ready interrupt is enabled

26.6.4. Interrupt Enable Set

Name: INTENSET
Offset: 0x10
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ERROR Error Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Error interrupt enable.

Value	Description
0	The Error interrupt is disabled
1	The Error interrupt is enabled

Bit 0 – READY NVM Ready Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the NVM Ready interrupt enable.

Value	Description
0	The NVM Ready interrupt is disabled
1	The NVM Ready interrupt is enabled

26.6.5. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x14
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ERROR Error

This flag is set upon the occurrence of a LOCKE or PROGE error.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Error interrupt flag.

Value	Description
0x0	No errors have occurred since the last clear
0x1	At least one error has occurred since the last clear

Bit 0 – READY NVM Ready

This flag is set when the NVMCTRL is ready to accept new commands.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the NVM Ready interrupt flag.

Value	Description
0x0	The NVM controller is busy programming or erasing
0x1	The NVM controller is ready to accept a new command

26.6.6. Interrupt Flag Software Set

Name: INTFLAGSET
Offset: 0x18
Reset: 0x00000000
Property: Local Write-Protection

1. This register allows the user to manually set an interrupt flag by software to test the ISR, and should be used solely for that purpose.
2. Changes in this register will also be reflected in the Interrupt Flag Status and Clear (INTFLAG) register.
3. Reading a bit in this register returns the status of the corresponding bit in the INTFLAG register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ERROR Error
This flag is cleared by writing a '1' to INTFLAG.ERROR.
This flag is set by writing a '1' to this bit.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Error interrupt flag.

Bit 0 – READY NVM Ready
This flag is cleared by writing a '1' to INTFLAG.READY.
This flag is set by writing a '1' to this bit.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the NVM Ready interrupt flag.

26.6.7. Status

Name: STATUS
Offset: 0x1C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 3 – LOCKE Lock Error Status
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Lock Error Status bit.

Value	Description
0x0	No programming of any locked region has occurred since this bit was last cleared
0x1	Programming of at least one locked region has occurred since this bit was last cleared

Bit 2 – PROGE Programming Error Status
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Programming Error Status bit.

Value	Description
0x0	No invalid commands or incorrect keywords have been written to the NVM Command register since this bit was last cleared
0x1	An invalid command and/or a incorrect keyword has been written in the NVM Command register since this bit was last cleared

26.6.8. Address

Name: ADDR
Offset: 0x20
Reset: 0x00000000
Property: Local Write-Protection

Only the number of bits required to access the memory is used.

Note: Access to the Flash is double-word granular (32 bits), while the address space for the Flash array is byte granular. This means that not all bits of ADDR are needed to access a double word in Flash. As a result, ADDR[1:0] is ignored and will always read as 0.

Bit	31	30	29	28	27	26	25	24
	ADDR[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
	ADDR[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
	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 31:0 – ADDR[31:0] NVM Address
ADDR drives the address to the NVM when a command is executed using CMDEX.
This register is also automatically updated when writing to an address in the Flash.

26.6.9. Lock Section

Name: LOCK
Offset: 0x24
Reset: 0x0000XXXX
Property: –

Note: Default state after erase will be unlocked (`0x0000FFFF`).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	LOCK15	LOCK14	LOCK13	LOCK12	LOCK11	LOCK10	LOCK9	LOCK8
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
	LOCK7	LOCK6	LOCK5	LOCK4	LOCK3	LOCK2	LOCK1	LOCK0
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – LOCKn Region n Lock Bit
To set or clear this bit field, the CMD register must be used.
Note: This value can be set persistently using the WLOCKREGION command and temporarily with the LR/UR commands.

Value	Description
0x0	The corresponding lock region is locked
0x1	The corresponding lock region is not locked

26.6.10. Read Data

Name: DATA
Offset: 0x28
Reset: 0xXXXXXXXX
Property: –

Bit	31	30	29	28	27	26	25	24
	DATA[31:24]							
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
	DATA[23:16]							
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
	DATA[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
	DATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 31:0 – DATA[31:0] Read Data
Continuously updated with contents read from Flash. The data register will contain the last data value read from Flash.

26.6.11. Write Protection Control

Name: WPCTRL
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write operation to be successful.

Value	Name	Description
0x4E564D	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	NVMCTRL register write protection is disabled
1	Write protection is enabled on NVMCTRL registers with the Local Write-Protection property. Non-debugger writes to NVMCTRL registers with Local Write-Protection are ignored and will generate a bus error.

27. PORT - I/O Pin Controller

27.1. Features

- Selectable Input and Output (I/O) Configuration for Each Individual 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
  - Slew rate limiting
- Configurable Input Buffer and Pull Settings:
  - Internal pull-up
  - Input sampling criteria
  - Input buffer can be disabled if not needed for lower power consumption
  - Read-Modify-Write (RMW) support for output value (OUTCLR/OUTSET/OUTTGL) and pin direction (DIRCLR/DIRSET/DIRTGL)
- Input Event:
  - Up to four input event pins for each PORT group
  - Can be output to pin
  - SET, CLEAR or TOGGLE event actions on the output value of a pin

27.2. Overview

The I/O Pin Controller (PORT) controls the I/O pins of the device. The I/O pins are organized into groups with up to 32 pins, collectively referred to as PORT groups. The pins of a PORT group can be configured and controlled individually or as a group. The number of PORT groups on a device may depend on the package and the number of pins. Each pin may be used either for General-Purpose Input/Output (GPIO) under direct application control or be assigned to an embedded device peripheral. When used for GPIO, each pin can be configured as input or output, with highly configurable driver and pull settings.

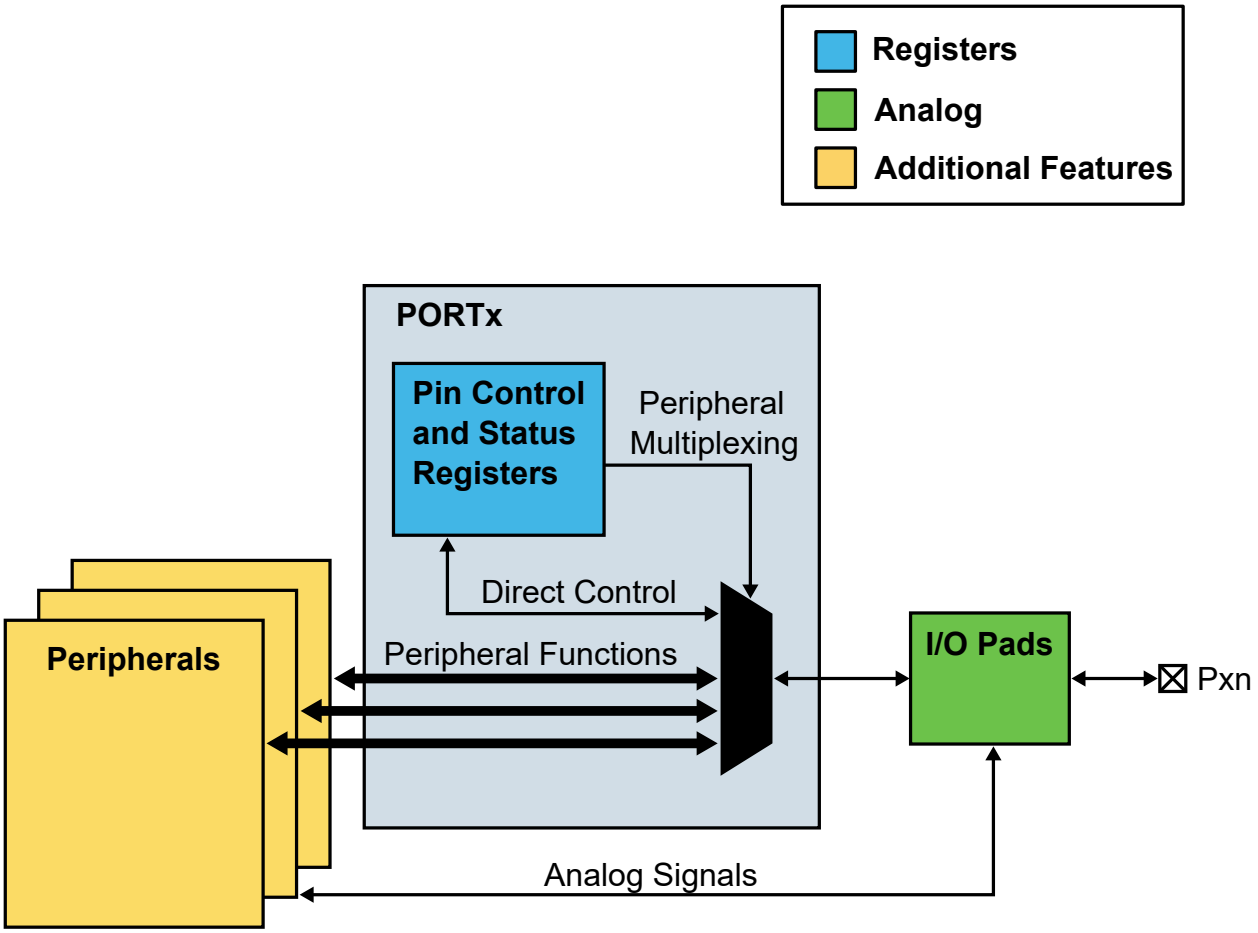
PORT groups are labeled using letters of the alphabet, starting with 'A', followed by 'B', 'C', and so on. Within each group, the pins are numbered starting from zero. Each pin is identified by combining the group letter with its zero-based index number. For example, the first pin in group A is named PA0, and the fourth pin in group B is named PB3.

All I/O pins have true Read-Modify-Write (RMW) functionality when used for GPIO. The direction or the output value of one or more pins may be changed (set, reset or toggled) explicitly without unintentionally changing the state of any other pins in the same port group, by a single atomic 8-, 16-, or 32-bit write.

The PORT is connected to the high-speed bus matrix through an AHB/APB bridge. A selection of registers can also be accessed using the low-latency ARM® single-cycle I/O port (IOBUS).

27.3. Block Diagram

Figure 27-1. PORT Block Diagram



27.3.1. Signal Description

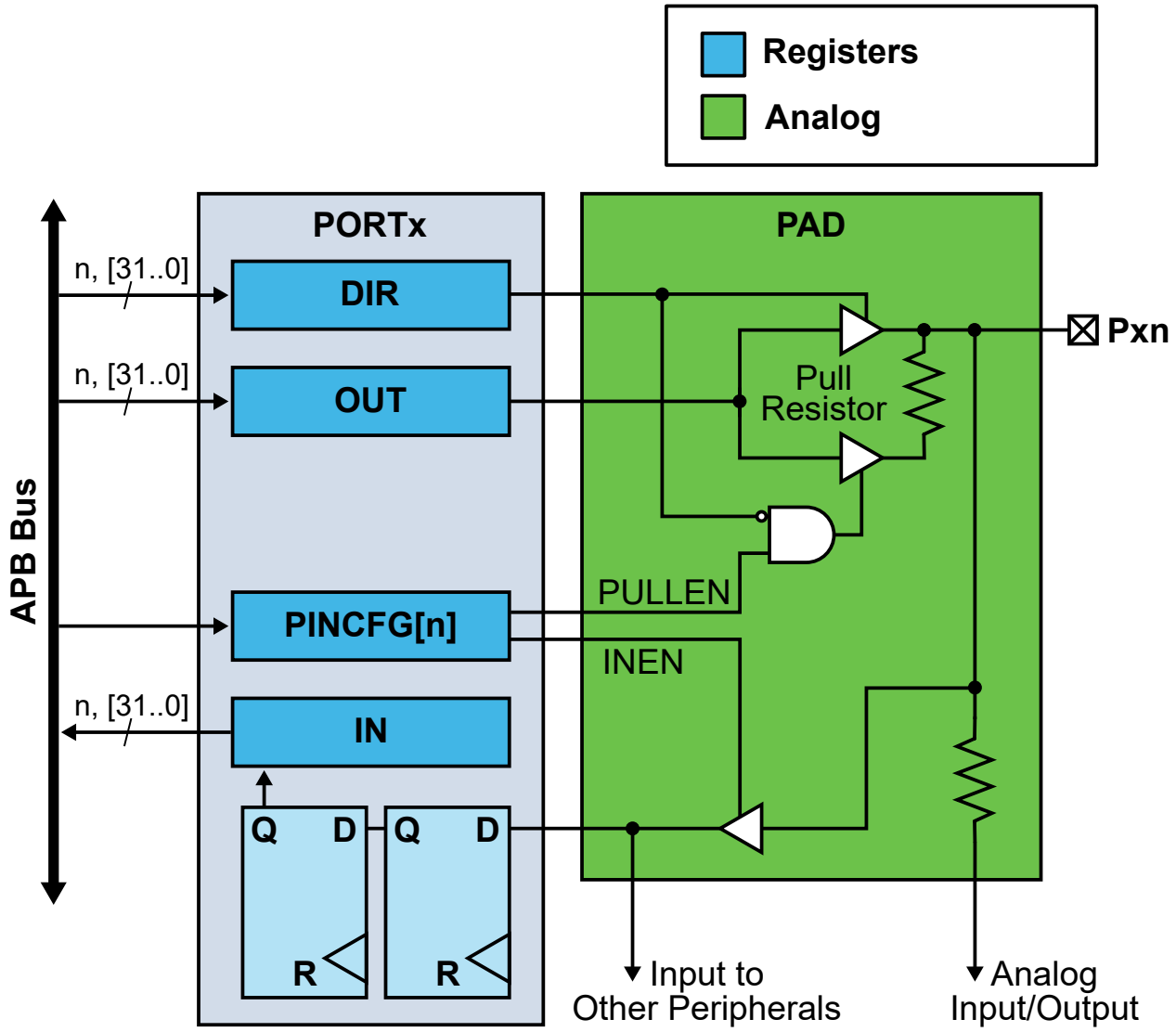
Table 27-1. Signal Description for PORT

Signal Name	Type	Description
Pxn	Digital I/O	General-Purpose I/O (GPIO) pin n in group x

Refer to the *Pinout* section for details on the pin mapping for this peripheral. A single signal can be mapped to several pins.

27.4. Functional Description

Figure 27-2. Overview of the PORT



27.4.1. 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 clock is running.

However, specific pins, such as those used for connection to a debugger, may be configured differently as required by their special function.

To configure pin n in PORT group x as an output and control the output drive level:

1. Set the direction (DIR[n]).
2. Configure the drive level (OUT[n]).

To configure pin n in PORT group x as an input and read the input pin level:

1. Enable the input buffer (PINCFG[n].INEN).
2. Read the input level (IN[n]).

27.4.2. Operation

Each I/O pin Pxn can be controlled by the registers in PORT. Each PORT group x has its own set of PORT registers, with a base address at (PORT + 0x80 × group index) (A corresponds to group index 0, B to 1, etc.). Within each set of registers, the pin index is n, ranging from 0 to 31.

The number of PORT groups depends on the package type and the number of pins. Refer to the *Pinout* chapter for details on the available pin configurations and PORT groups.

27.4.2.1. I/O Pin Configuration

The I/O pins of the device are controlled by the PORT peripheral registers. Each port pin has a corresponding bit in the Data Direction (DIR) and Data Output Value (OUT) registers to configure it as an input or output, and to control the output or pull state.

The Pin Configuration register (PINCFG[n]) is used for additional I/O pin configuration. A pin can be set to a totem-pole or pull configuration.

Since pull configuration is done through the PINCFG[n] register, all intermediate PORT states during switching of the pin direction and pin values are avoided.

The I/O pin configurations summarized in the following table are described in detail in this chapter.

Table 27-2. Pin Configurations Summary

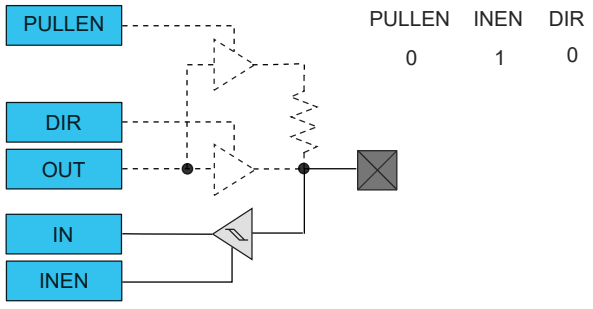
DIR	INEN	PULLEN	OUT	Configuration
0	0	0	X	All digital disabled
0	1	0	X	Input without pull
0	1	1	1	Input with pull-up
0	0	1	1	Pull-up output; input buffer disabled
1	0	X	X	Drive output; input buffer disabled
1	1	X	X	Drive output; input buffer enabled

27.4.2.1.1. Input Pin Configuration

To use pin Pxn as an input, the Input Buffer Enable bit in the Pin Configuration register (PINCFG[n].INEN) must be '1' and bit n in the Data Direction register (DIR) must be '0'. The physical pin state can be read from the corresponding bit in the Data Input Value register (IN).

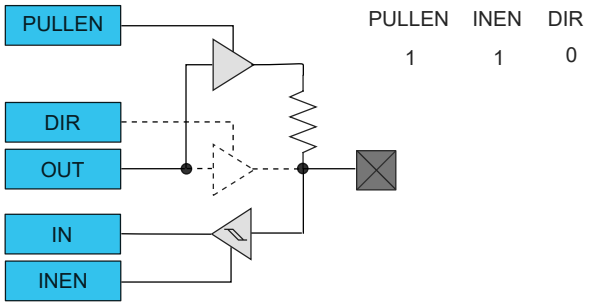
The pin will float if no external pull is connected and the Pull Enable bit in the Pin Configuration register (PINCFG[n].PULLEN) is '0'.

Figure 27-3. I/O Configuration - Standard Input



When the Pull Enable bit is '1', pull-up is enabled when the corresponding OUT value is written to '1'.

Figure 27-4. I/O Configuration - Input With Pull



To reduce power consumption, the input synchronizer is clocked only when an input read is requested. This will delay the read operation by two cycles of the PORT clock. To eliminate the delay, the input synchronizers for each PORT group of eight pins can be configured to be always active. This is enabled by writing a '1' to the corresponding bit in the Input Sampling Mode bit field of the Control register (CTRL.SAMPLING).

27.4.2.1.2. Output Pin Configuration

To use pin Pxn as an output, bit n in the Data Direction register (DIR) must be '1'. The pin is driven low or high according to the bit setting in the Data Output Value register (OUT). If bit n in OUT is written to '1', pin n is driven HIGH. If bit n in OUT is written to '0', pin n is driven LOW.

In this totem-pole (push-pull) configuration, there is no current limitation for sink or source other than what the pin is capable of.

Figure 27-5. I/O Configuration - Totem Pole Output With Disabled Input

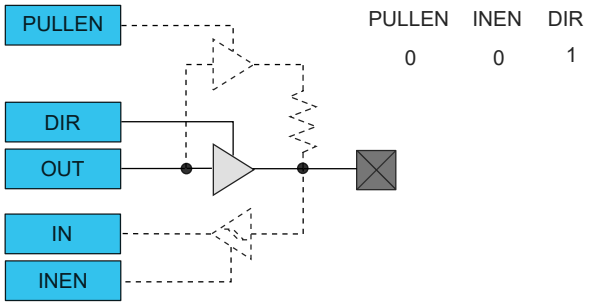
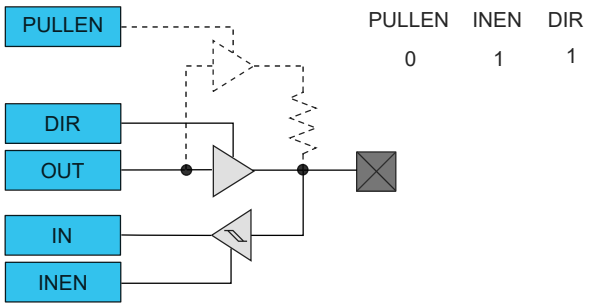


Figure 27-6. I/O Configuration - Totem Pole Output With Enabled Input

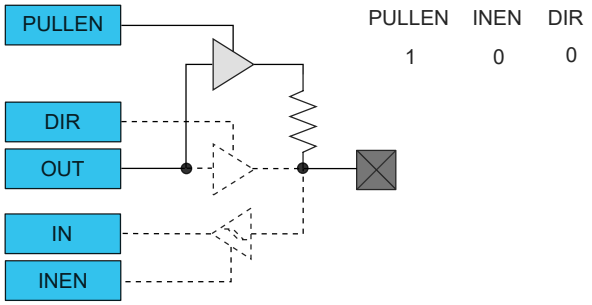


When the corresponding bit in the DIR register is '0' and the Pull Enable bit in the Pin Configuration register (PINCFG[n].PULLEN) is '1', the pin is pulled high when the bit setting in the OUT register is written to '1'.

This high-impedance configuration ensures the pin does not float and maintains a defined logic level unless overridden by an external driver.

Note: Enabling the output driver will automatically disable pull.

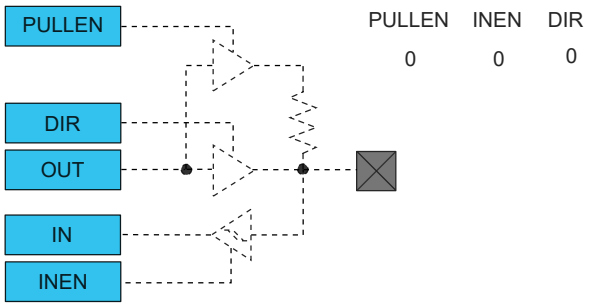
Figure 27-7. I/O Configuration - Output With Programmable Pull



27.4.2.1.3. Digital Functionality Disabled

After a Reset, all configuration bits are cleared, and neither input nor output functionality is enabled. This configuration is used for analog signals or unused pins.

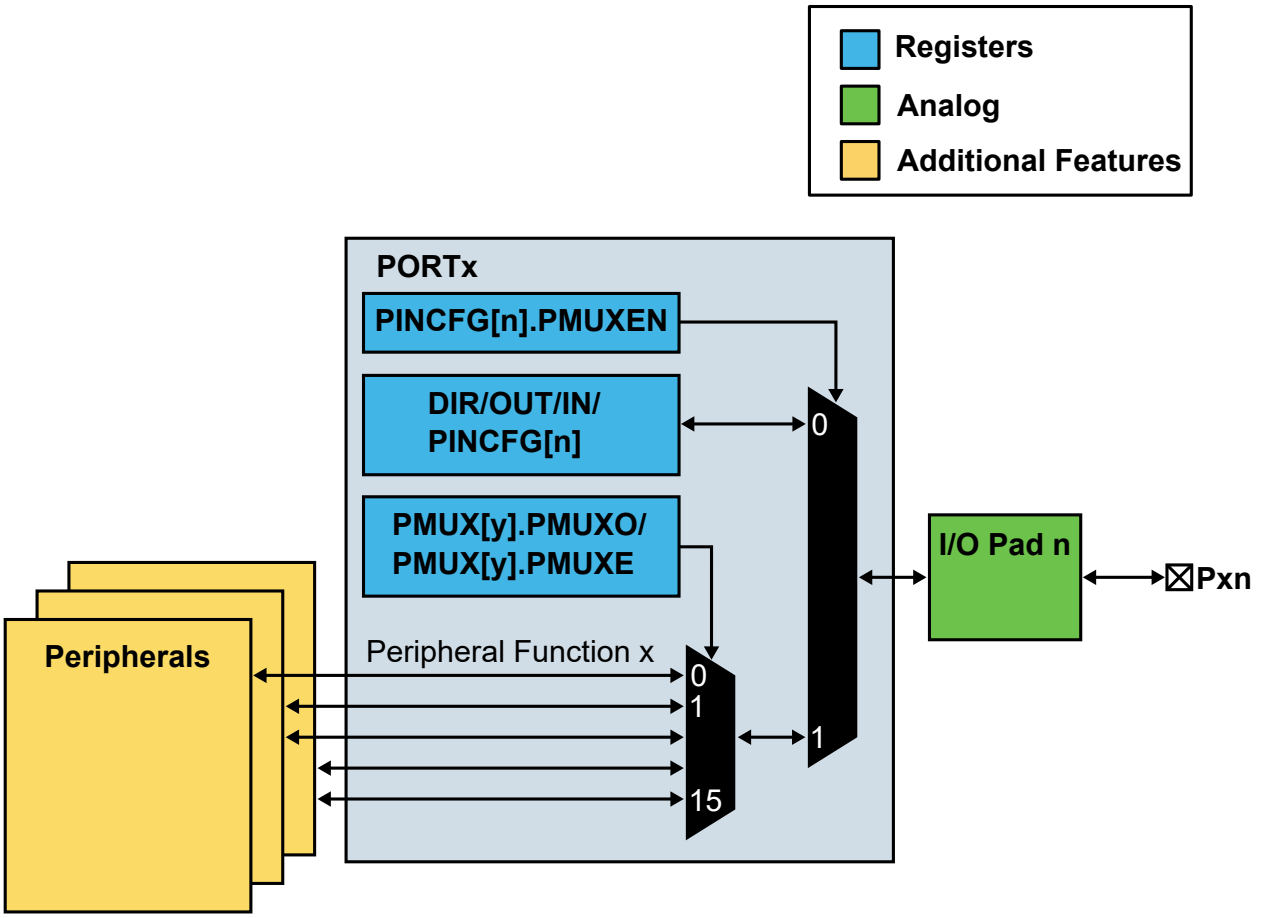
Figure 27-8. I/O Configuration - Reset or Analog I/O: Digital Output, Input and Pull Disabled



27.4.2.2. Peripheral Functions Multiplexing

Each I/O pin can be controlled either by the registers in PORT or by a peripheral, as shown in the following figure.

Figure 27-9. Overview of the Peripheral Functions Multiplexing



The Peripheral Multiplexer Enable bit in the Pin Configuration registers (PINCFG[n].PMUXEN) can be written to '1' to enable the connection between peripheral functions and individual I/O pins. This overrides the connection between the PORT and that I/O pin, connecting the selected peripheral signal to the particular I/O pin instead of the PORT line bundle.

The Peripheral Multiplexing registers (PMUX[m]) select the peripheral function for the corresponding pins. Each register controls the peripheral multiplexing for two pins, through the Peripheral Multiplexing for Odd-Numbered Pin (PMUXO) and Peripheral Multiplexing for Even-Numbered Pin (PMUXE) bit fields. For example, PA0 is configured in PMUX[0].PMUXE, and PA5 is configured in PMUX[2].PMUXO.

Each Peripheral Multiplexing bit field can be configured to peripheral functions A through J. Refer to the *Pinout* section for details on the functions each pin can connect to.

Note: The chosen peripheral must be configured and enabled.

27.4.3. Additional Features

27.4.3.1. Read-Modify-Write Pin Configuration

The Data Direction (DIR) and Data Output Value (OUT) registers each have three helper registers: Clear, Set and Toggle. These helper registers allow the user to update one or more I/O pins by performing a hardware Read-Modify-Write (RMW) operation on the register.

Writing a '1' to one or more bits in the Data Direction Clear (DIRCLR), Data Direction Set (DIRSET), and Data Direction Toggle (DIRTGL) registers will respectively clear, set or toggle the corresponding bit(s) in the DIR register.

Similarly, writing a '1' to one or more bits in the Data Output Value Clear (OUTCLR), Data Output Value Set (OUTSET), and Data Output Value Toggle (OUTTGL) registers will respectively clear, set or toggle the corresponding bit(s) in the OUT register.

Reading from the helper registers will return the value from the corresponding DIR or OUT register.

27.4.3.2. Multi-Pin Write Configuration

The Write Configuration register (WRCONFIG) allows writing the Pin Configuration registers (PINCFG[n]) and Peripheral Multiplexing registers (PMUX[m]) for up to 16 pins in one write.

The Half-Word Select bit (WRCONFIG.HWSEL) selects whether pins 0-15 or 16-31 are addressed, while the Pin Mask for Multiple Pin Configuration bit field (WRCONFIG.PINMASK) is used to specify which of those 16 pin registers are written.

The Write PMUX bit (WRCONFIG.WRPMUX) can be used to enable or disable writing to the selected Peripheral Multiplexing registers. If WRPMUX is written to '1', the Peripheral Multiplexing bit field (WRCONFIG.PMUX) is written to the respective Peripheral Multiplexing for Odd-Numbered Pin (PMUX[m].PMUXO) or Peripheral Multiplexing for Even-Numbered Pin (PMUX[m].PMUXE) bit fields for all the selected pins.

The Write PINCFG bit (WRCONFIG.WRPINCFG) can be used to enable or disable writing to the selected Peripheral Configuration registers. If WRPINCFG is written to '1', the PINCFG mirror bit fields in the WRCONFIG register are written to all the selected registers.

27.4.3.3. Slew Rate Limitation

The output pin internal slew rate limit can be enabled by writing a '1' to the Output Driver Slew Rate Limit Enable bit in the Pin Configuration register (PINCFG[n].SLEWLIM).

Refer to the *Electrical Characteristics* section for details on the slew rate limit.

27.4.4. Sleep Mode Operation

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

Event inputs connected to the event system can be triggered without exiting sleep modes. The SET, CLR and TGL events are not available in Standby sleep mode.

Refer to the *PM - Power Manager* chapter for additional information about sleep mode operation.

27.4.5. Debug Operation

When the CPU is halted in Debug mode, the PORT continues normal operation.

27.5. Dependencies

27.5.1. I/O Lines

Not applicable.

27.5.2. Power Management

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 clock is running.

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

Event inputs connected to the event system can be triggered without exiting sleep modes.

Refer to the *PM - Power Manager* chapter for details on the different sleep modes.

27.5.3. Clocks

The PORT bus clock (CLK_PORT_APB) can be enabled or disabled in the power manager, and its default state can be found in the Peripheral Clock Masking section of the *MCLK – Main Clock* chapter.

27.5.4. DMA

Not applicable.

27.5.5. Interrupts

Refer to the *EIC – External Interrupt Controller* chapter for details on external pin interrupts.

27.5.6. Events

The PORT allows up to four input events to control individual I/O pins. These input events are generated by the Event System (EVSYS) module and can originate from a different clock domain than the clock domain of the PORT module.

Table 27-3. Event Users

User Name		Description	Input Detection	Channel Path Type
Peripheral	Input			
PORT	EV_n	PORT Event n Input	Level/Rising Edge	Asynchronous

Writing a '1' to an Event Input Enable bit in the Event Control register (EVCTRL.xxEI) enables the corresponding action on an input event. Writing a '0' to this bit disables the corresponding action on input event.

The event users can trigger the following actions. Note that if connected to several events, the enabled action will be taken on any of the incoming events.

Table 27-4. Event Actions

Event Input	Event Action	Description	Input Detection	Async/Sync
EV_n	OUT	The Data Output Value bit of I/O pin PIDn will be set to the level of the event	Level	Asynchronous
	SET	The Data Output Value bit of I/O pin PIDn will be set	Rising Edge	Up to three clock cycles after a rising edge. Not available in Standby sleep mode.
	CLR	The Data Output Value bit of I/O pin PIDn will be cleared		
	TGL	The Data Output Value bit of I/O pin PIDn will be toggled		

The event actions are configured in the Event n Action bit fields in the Event Control register (EVCTRL.EVACTn).

Each event input can address one I/O pin per PORT group. The selection of the pin is indicated by the Event n Pin Identifier bit fields in the EVCTRL register (EVCTRL.PIDn).

To avoid action conflict on the output value of the register (OUT) for a particular I/O pin, only one action is executed.

- When only SET actions are triggered simultaneously, a SET action is executed
- When only CLR actions are triggered simultaneously, a CLR action is executed
- If any other combinations of actions are triggered simultaneously, a TGL action is executed

Because events are received asynchronously, the I/O pin may have unpredictable levels, depending on the timing of when the events are received. When several events are output to the same pin, the lowest event line will have access and all other events will be ignored.

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

27.5.7. Analog Connections

Analog signals are normally connected to pins by configuring the peripheral instance and do not need to use the Peripheral Multiplexer. However, configuring the PORT pin to use an analog function—typically peripheral function B—will disable the controls for pull-up, input, and output.

27.5.8. Access Priority

The PORT is accessed by the following systems:

- The ARM® CPU through the ARM single-cycle I/O port (IOBUS)
- The ARM CPU through the high-speed matrix and the AHB/APB bridge (APB)
- EVSYS through four asynchronous input events

The CPU local bus (IOBUS) is an interface that connects the CPU to the PORT. It is a single-cycle bus interface supporting 8-, 16-, and 32-bit data sizes. It does not support wait states.

This bus is generally used for low-latency operation. 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.

Because the IOBUS cannot wait for the IN register resynchronization, the Control register (CTRL) must be configured to continuous sampling for all pins that need to be read through the IOBUS to prevent stale data from being read.

IOBUS writes are not prevented on Peripheral Access Controller (PAC) write-protected registers when the PORT module is PAC protected.

The following priority is adopted:

1. ARM CPU IOBUS (no wait tolerated).
2. APB.
3. EVSYS input events, except for OUT events, in which the output pin follows the event input signal independently of the OUT register value.

Note: A one clock cycle latency may be observed on APB access in the case of concurrent PORT accesses.

27.6. Register Summary - PORT

Tip:
The I/O pins are assembled in pin groups (”PORT groups”) with up to 32 pins. Group 0 consists of the PA pins, group 1 is for the PB pins, etc. Each pin group has its own PORT registers, with a base address at byte address (PORT + 0x80 * group index) (A corresponds to group index 0, B to 1, etc.).

Within that set of registers, the pin index is y, from 0 to 31. For example, the register address offset for the Data Direction (DIR) register for group 0 (PA00 to PA31) is 0x00, and the register address offset for the DIR register for group 1 (PB00 to PB31) is 0x80.

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	DIR	7:0	DIR7	DIR6	DIR5	DIR4	DIR3	DIR2	DIR1	DIR0
		15:8	DIR15	DIR14	DIR13	DIR12	DIR11	DIR10	DIR9	DIR8
		23:16	DIR23	DIR22	DIR21	DIR20	DIR19	DIR18	DIR17	DIR16
		31:24	DIR31	DIR30	DIR29	DIR28	DIR27	DIR26	DIR25	DIR24
0x04	DIRCLR	7:0	DIR7CLR	DIR6CLR	DIR5CLR	DIR4CLR	DIR3CLR	DIR2CLR	DIR1CLR	DIR0CLR
		15:8	DIR15CLR	DIR14CLR	DIR13CLR	DIR12CLR	DIR11CLR	DIR10CLR	DIR9CLR	DIR8CLR
		23:16	DIR23CLR	DIR22CLR	DIR21CLR	DIR20CLR	DIR19CLR	DIR18CLR	DIR17CLR	DIR16CLR
		31:24	DIR31CLR	DIR30CLR	DIR29CLR	DIR28CLR	DIR27CLR	DIR26CLR	DIR25CLR	DIR24CLR
0x08	DIRSET	7:0	DIR7SET	DIR6SET	DIR5SET	DIR4SET	DIR3SET	DIR2SET	DIR1SET	DIR0SET
		15:8	DIR15SET	DIR14SET	DIR13SET	DIR12SET	DIR11SET	DIR10SET	DIR9SET	DIR8SET
		23:16	DIR23SET	DIR22SET	DIR21SET	DIR20SET	DIR19SET	DIR18SET	DIR17SET	DIR16SET
		31:24	DIR31SET	DIR30SET	DIR29SET	DIR28SET	DIR27SET	DIR26SET	DIR25SET	DIR24SET
0x0C	DIRTGL	7:0	DIR7TGL	DIR6TGL	DIR5TGL	DIR4TGL	DIR3TGL	DIR2TGL	DIR1TGL	DIR0TGL
		15:8	DIR15TGL	DIR14TGL	DIR13TGL	DIR12TGL	DIR11TGL	DIR10TGL	DIR9TGL	DIR8TGL
		23:16	DIR23TGL	DIR22TGL	DIR21TGL	DIR20TGL	DIR19TGL	DIR18TGL	DIR17TGL	DIR16TGL
		31:24	DIR31TGL	DIR30TGL	DIR29TGL	DIR28TGL	DIR27TGL	DIR26TGL	DIR25TGL	DIR24TGL
0x10	OUT	7:0	OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
		15:8	OUT15	OUT14	OUT13	OUT12	OUT11	OUT10	OUT9	OUT8
		23:16	OUT23	OUT22	OUT21	OUT20	OUT19	OUT18	OUT17	OUT16
		31:24	OUT31	OUT30	OUT29	OUT28	OUT27	OUT26	OUT25	OUT24
0x14	OUTCLR	7:0	OUT7CLR	OUT6CLR	OUT5CLR	OUT4CLR	OUT3CLR	OUT2CLR	OUT1CLR	OUT0CLR
		15:8	OUT15CLR	OUT14CLR	OUT13CLR	OUT12CLR	OUT11CLR	OUT10CLR	OUT9CLR	OUT8CLR
		23:16	OUT23CLR	OUT22CLR	OUT21CLR	OUT20CLR	OUT19CLR	OUT18CLR	OUT17CLR	OUT16CLR
		31:24	OUT31CLR	OUT30CLR	OUT29CLR	OUT28CLR	OUT27CLR	OUT26CLR	OUT25CLR	OUT24CLR
0x18	OUTSET	7:0	OUT7SET	OUT6SET	OUT5SET	OUT4SET	OUT3SET	OUT2SET	OUT1SET	OUT0SET
		15:8	OUT15SET	OUT14SET	OUT13SET	OUT12SET	OUT11SET	OUT10SET	OUT9SET	OUT8SET
		23:16	OUT23SET	OUT22SET	OUT21SET	OUT20SET	OUT19SET	OUT18SET	OUT17SET	OUT16SET
		31:24	OUT31SET	OUT30SET	OUT29SET	OUT28SET	OUT27SET	OUT26SET	OUT25SET	OUT24SET
0x1C	OUTTGL	7:0	OUT7TGL	OUT6TGL	OUT5TGL	OUT4TGL	OUT3TGL	OUT2TGL	OUT1TGL	OUT0TGL
		15:8	OUT15TGL	OUT14TGL	OUT13TGL	OUT12TGL	OUT11TGL	OUT10TGL	OUT9TGL	OUT8TGL
		23:16	OUT23TGL	OUT22TGL	OUT21TGL	OUT20TGL	OUT19TGL	OUT18TGL	OUT17TGL	OUT16TGL
		31:24	OUT31TGL	OUT30TGL	OUT29TGL	OUT28TGL	OUT27TGL	OUT26TGL	OUT25TGL	OUT24TGL
0x20	IN	7:0	IN7	IN6	IN5	IN4	IN3	IN2	IN1	IN0
		15:8	IN15	IN14	IN13	IN12	IN11	IN10	IN9	IN8
		23:16	IN23	IN22	IN21	IN20	IN19	IN18	IN17	IN16
		31:24	IN31	IN30	IN29	IN28	IN27	IN26	IN25	IN24
0x24	CTRL	7:0	SAMPLING7	SAMPLING6	SAMPLING5	SAMPLING4	SAMPLING3	SAMPLING2	SAMPLING1	SAMPLING0
		15:8	SAMPLING15	SAMPLING14	SAMPLING13	SAMPLING12	SAMPLING11	SAMPLING10	SAMPLING9	SAMPLING8
		23:16	SAMPLING23	SAMPLING22	SAMPLING21	SAMPLING20	SAMPLING19	SAMPLING18	SAMPLING17	SAMPLING16
		31:24	SAMPLING31	SAMPLING30	SAMPLING29	SAMPLING28	SAMPLING27	SAMPLING26	SAMPLING25	SAMPLING24
0x28	WRCONFIG	7:0	PINMASK[7:0]							
		15:8	PINMASK[15:8]							
		23:16				SLEWLIM		PULLEN	INEN	PMUXEN
		31:24	HWSEL	WRPINCFG		WRPMUX	PMUX[3:0]			

PORT (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x2C	EVCTRL	7:0	PORTEI0	EVACT0[1:0]		PID0[4:0]				
		15:8	PORTEI1	EVACT1[1:0]		PID1[4:0]				
		23:16	PORTEI2	EVACT2[1:0]		PID2[4:0]				
		31:24	PORTEI3	EVACT3[1:0]		PID3[4:0]				
0x30	PMUX[0]	7:0	PMUXO[3:0]				PMUXE[3:0]			
...										
0x3F	PMUX[15]	7:0	PMUXO[3:0]				PMUXE[3:0]			
0x40	PINCFG[0]	7:0				SLEWLIM		PULLEN	INEN	PMUXEN
...										
0x5F	PINCFG[31]	7:0				SLEWLIM		PULLEN	INEN	PMUXEN

27.6.1. Data Direction

Name: DIR
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to configure one or more I/O pins as an input or output.

This register can be manipulated without performing a Read-Modify-Write (RMW) operation by using the Data Direction Toggle (DIRTGL), Data Direction Clear (DIRCLR), and Data Direction Set (DIRSET) registers.

Bit	31	30	29	28	27	26	25	24
	DIR31	DIR30	DIR29	DIR28	DIR27	DIR26	DIR25	DIR24
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
	DIR23	DIR22	DIR21	DIR20	DIR19	DIR18	DIR17	DIR16
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
	DIR15	DIR14	DIR13	DIR12	DIR11	DIR10	DIR9	DIR8
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
	DIR7	DIR6	DIR5	DIR4	DIR3	DIR2	DIR1	DIR0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – DIRn Port Data Direction n

These bits control the data direction for the individual I/O pins in the PORT group.

Value	Description
0	The I/O pin in the PORT group corresponding to this bit is configured as an input
1	The I/O pin in the PORT group corresponding to this bit is configured as an output

27.6.2. Data Direction Clear

Name: DIRCLR
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to set one or more I/O pins as inputs, without performing a Read-Modify-Write (RMW) operation on the Data Direction register (DIR). Changes in this register will be reflected in the DIR, Data Direction Toggle (DIRTGL), and Data Direction Set (DIRSET) registers.

Bit	31	30	29	28	27	26	25	24
	DIR31CLR	DIR30CLR	DIR29CLR	DIR28CLR	DIR27CLR	DIR26CLR	DIR25CLR	DIR24CLR
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
	DIR23CLR	DIR22CLR	DIR21CLR	DIR20CLR	DIR19CLR	DIR18CLR	DIR17CLR	DIR16CLR
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
	DIR15CLR	DIR14CLR	DIR13CLR	DIR12CLR	DIR11CLR	DIR10CLR	DIR9CLR	DIR8CLR
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
	DIR7CLR	DIR6CLR	DIR5CLR	DIR4CLR	DIR3CLR	DIR2CLR	DIR1CLR	DIR0CLR
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – DIRnCLR Port Data Direction n Clear

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding bit in the DIR register, which configures the I/O pin as an input.

Value	Description
0	The I/O pin in the PORT group corresponding to this bit is configured as an input
1	The I/O pin in the PORT group corresponding to this bit is configured as an output

27.6.3. Data Direction Set

Name: DIRSET
Offset: 0x08
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to set one or more I/O pins as outputs, without performing a Read-Modify-Write (RMW) operation on the Data Direction register (DIR). Changes in this register will be reflected in the DIR, Data Direction Toggle (DIRTGL), and Data Direction Clear (DIRCLR) registers.

Bit	31	30	29	28	27	26	25	24
	DIR31SET	DIR30SET	DIR29SET	DIR28SET	DIR27SET	DIR26SET	DIR25SET	DIR24SET
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
	DIR23SET	DIR22SET	DIR21SET	DIR20SET	DIR19SET	DIR18SET	DIR17SET	DIR16SET
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
	DIR15SET	DIR14SET	DIR13SET	DIR12SET	DIR11SET	DIR10SET	DIR9SET	DIR8SET
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
	DIR7SET	DIR6SET	DIR5SET	DIR4SET	DIR3SET	DIR2SET	DIR1SET	DIR0SET
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – DIRnSET Port Data Direction n Set

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will set the corresponding bit in the DIR register, which configures the I/O pin as an output.

Value	Description
0	The I/O pin in the PORT group corresponding to this bit is configured as an input
1	The I/O pin in the PORT group corresponding to this bit is configured as an output

27.6.4. Data Direction Toggle

Name: DIRTGL
Offset: 0x0C
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to toggle the direction of one or more I/O pins without performing a Read-Modify-Write (RMW) operation on the Data Direction register (DIR). Changes in this register will be reflected in the DIR, Data Direction Set (DIRSET), and Data Direction Clear (DIRCLR) registers.

Bit	31	30	29	28	27	26	25	24
	DIR31TGL	DIR30TGL	DIR29TGL	DIR28TGL	DIR27TGL	DIR26TGL	DIR25TGL	DIR24TGL
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
	DIR23TGL	DIR22TGL	DIR21TGL	DIR20TGL	DIR19TGL	DIR18TGL	DIR17TGL	DIR16TGL
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
	DIR15TGL	DIR14TGL	DIR13TGL	DIR12TGL	DIR11TGL	DIR10TGL	DIR9TGL	DIR8TGL
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
	DIR7TGL	DIR6TGL	DIR5TGL	DIR4TGL	DIR3TGL	DIR2TGL	DIR1TGL	DIR0TGL
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – DIRnTGL Port Data Direction n Toggle

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will toggle the corresponding bit in the DIR register, which reverses the direction of the I/O pin.

Value	Description
0	The I/O pin in the PORT group corresponding to this bit is configured as an input
1	The I/O pin in the PORT group corresponding to this bit is configured as an output

27.6.5. Data Output Value

Name: OUT
Offset: 0x10
Reset: 0x00000000
Property: PAC Write-Protection

This register controls the output drive for the individual I/O pins in the PORT.

This register can be manipulated without performing a Read-Modify-Write (RMW) operation by using the Data Output Value Clear (OUTCLR), Data Output Value Set (OUTSET), and Data Output Value Toggle (OUTTGL) registers.

Bit	31	30	29	28	27	26	25	24
	OUT31	OUT30	OUT29	OUT28	OUT27	OUT26	OUT25	OUT24
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
	OUT23	OUT22	OUT21	OUT20	OUT19	OUT18	OUT17	OUT16
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
	OUT15	OUT14	OUT13	OUT12	OUT11	OUT10	OUT9	OUT8
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
	OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – OUTn PORT Data Output Value n

For pins configured as outputs through the Data Direction register (DIR), these bits set the logical output drive level.
For pins configured as inputs through the DIR register and with pull enabled through the Pull Enable bit in the Pin Configuration register (PINCFG.PULLEN), these bits configure the input pull direction.

Value	Description
0	The pin in the PORT group corresponding to this bit is driven low
1	The pin in the PORT group corresponding to this bit is driven high, or the input is connected to an internal pull-up

27.6.6. Data Output Value Clear

Name: OUTCLR
Offset: 0x14
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to set the low drive level for one or more output I/O pins, without performing a Read-Modify-Write (RMW) operation on the Data Output Value register (OUT). Changes in this register will be reflected in the OUT, Data Output Value Toggle (OUTTGL), and Data Output Value Set (OUTSET) registers.

Bit	31	30	29	28	27	26	25	24
	OUT31CLR	OUT30CLR	OUT29CLR	OUT28CLR	OUT27CLR	OUT26CLR	OUT25CLR	OUT24CLR
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
	OUT23CLR	OUT22CLR	OUT21CLR	OUT20CLR	OUT19CLR	OUT18CLR	OUT17CLR	OUT16CLR
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
	OUT15CLR	OUT14CLR	OUT13CLR	OUT12CLR	OUT11CLR	OUT10CLR	OUT9CLR	OUT8CLR
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
	OUT7CLR	OUT6CLR	OUT5CLR	OUT4CLR	OUT3CLR	OUT2CLR	OUT1CLR	OUT0CLR
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – OUTnCLR PORT Data Output Value n Clear

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will clear the corresponding bit in the OUT register.

Value	Description
0	The pin in the PORT group corresponding to this bit is driven low
1	The pin in the PORT group corresponding to this bit is driven high, or the input is connected to an internal pull-up

27.6.7. Data Output Value Set

Name: OUTSET
Offset: 0x18
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to set the high drive level for one or more output I/O pins, without performing a Read-Modify-Write (RMW) operation on the Data Output Value register (OUT). Changes in this register will be reflected in the OUT, Data Output Value Toggle (OUTTGL), and Data Output Value Clear (OUTCLR) registers.

Bit	31	30	29	28	27	26	25	24
	OUT31SET	OUT30SET	OUT29SET	OUT28SET	OUT27SET	OUT26SET	OUT25SET	OUT24SET
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
	OUT23SET	OUT22SET	OUT21SET	OUT20SET	OUT19SET	OUT18SET	OUT17SET	OUT16SET
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
	OUT15SET	OUT14SET	OUT13SET	OUT12SET	OUT11SET	OUT10SET	OUT9SET	OUT8SET
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
	OUT7SET	OUT6SET	OUT5SET	OUT4SET	OUT3SET	OUT2SET	OUT1SET	OUT0SET
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – OUTnSET PORT Data Output Value n Set

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will set the corresponding bit in the OUT register.

Value	Description
0	The pin in the PORT group corresponding to this bit is driven low
1	The pin in the PORT group corresponding to this bit is driven high, or the input is connected to an internal pull-up

27.6.8. Data Output Value Toggle

Name: OUTTGL
Offset: 0x1C
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to toggle the drive level of one or more output I/O pins without performing a Read-Modify-Write (RMW) operation on the Data Output Value register (OUT). Changes in this register will be reflected in the OUT, Data Output Value Set (OUTSET), and Data Output Value Clear (OUTCLR) registers.

Bit	31	30	29	28	27	26	25	24
	OUT31TGL	OUT30TGL	OUT29TGL	OUT28TGL	OUT27TGL	OUT26TGL	OUT25TGL	OUT24TGL
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
	OUT23TGL	OUT22TGL	OUT21TGL	OUT20TGL	OUT19TGL	OUT18TGL	OUT17TGL	OUT16TGL
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
	OUT15TGL	OUT14TGL	OUT13TGL	OUT12TGL	OUT11TGL	OUT10TGL	OUT9TGL	OUT8TGL
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
	OUT7TGL	OUT6TGL	OUT5TGL	OUT4TGL	OUT3TGL	OUT2TGL	OUT1TGL	OUT0TGL
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – OUTnTGL PORT Data Output Value n Toggle

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will toggle the corresponding bit in the OUT register.

Value	Description
0	The pin in the PORT group corresponding to this bit is driven low
1	The pin in the PORT group corresponding to this bit is driven high, or the input is connected to an internal pull-up

27.6.9. Data Input Value

Name: IN
Offset: 0x20
Reset: 0x00000000
Property: –

This register allows the user to read the logical level of one or more I/O pins.

Bit	31	30	29	28	27	26	25	24
	IN31	IN30	IN29	IN28	IN27	IN26	IN25	IN24
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
	IN23	IN22	IN21	IN20	IN19	IN18	IN17	IN16
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
	IN15	IN14	IN13	IN12	IN11	IN10	IN9	IN8
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
	IN7	IN6	IN5	IN4	IN3	IN2	IN1	IN0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – INn PORT Data Input Value n

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.

27.6.10. Control

Name: CTRL
Offset: 0x24
Reset: 0x00000000
Property: PAC Write-Protection

Bit	31	30	29	28	27	26	25	24
	SAMPLING31	SAMPLING30	SAMPLING29	SAMPLING28	SAMPLING27	SAMPLING26	SAMPLING25	SAMPLING24
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
	SAMPLING23	SAMPLING22	SAMPLING21	SAMPLING20	SAMPLING19	SAMPLING18	SAMPLING17	SAMPLING16
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
	SAMPLING15	SAMPLING14	SAMPLING13	SAMPLING12	SAMPLING11	SAMPLING10	SAMPLING9	SAMPLING8
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
	SAMPLING7	SAMPLING6	SAMPLING5	SAMPLING4	SAMPLING3	SAMPLING2	SAMPLING1	SAMPLING0
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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – SAMPLINGn Input Sampling Mode n

These bits control the input sampling functionality of the I/O pin input samplers, for pins configured as inputs through the Data Direction register (DIR).
The input samplers are enabled and disabled in subgroups of eight pins. Therefore, if any pin within a byte requests continuous sampling, all pins in that eight-pin subgroup will be continuously sampled.

Value	Description
0	On-demand sampling is enabled for the eight I/O pins in the PORT group corresponding to the byte location of this bit
1	Continuous sampling is enabled for the eight I/O pins in the PORT group corresponding to the byte location of this bit

27.6.11. Write Configuration

Name: WRCONFIG
Offset: 0x28
Reset: 0x00000000
Property: PAC Write-Protection

This write-only register is used to configure several pins simultaneously with the same configuration and peripheral multiplexing.

To avoid side effect from non-atomic access, 8- or 16-bit writes to this register will have no effect.

Reading this register always returns '0'.

Bit	31	30	29	28	27	26	25	24
	HWSEL	WRPINCFG		WRPMUX	PMUX[3:0]			
Access	W	W		W	W	W	W	W
Reset	0	0		0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				SLEWLIM		PULLEN	INEN	PMUXEN
Access				W		W	W	W
Reset				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 PORT group to be reconfigured in the atomic write operation.

Value	Description
0	The lower 16 pins of the PORT group will be configured
1	The upper 16 pins of the PORT group will be configured

Bit 30 – WRPINCFG Write PINCFG
This bit determines whether the atomic write operation will update the Pin Configuration register (PINCFG[n]) for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit updates the configuration of the selected pins with the values written toWRCONFIG.SLEWLIM, WRCONFIG.PULLEN, WRCONFIG.INEN, WRCONFIG.PMUXEN, and WRCONFIG.PINMASK values.

Bit 28 – WRPMUX Write PMUX
This bit determines whether the atomic write operation will update the Peripheral Multiplexing register (PMUX[m]) for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit updates the pin multiplexer configuration of the selected pins with the written WRCONFIG.PMUX value.

Bits 27:24 – PMUX[3:0] Peripheral Multiplexing

These bits determine the new value written to the Peripheral Multiplexing for Odd-Numbered Pin (PMUX[m].PMUXO) or Peripheral Multiplexing for Even-Numbered Pin (PMUX[m].PMUXE) for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPMUX bit is set.

Value	Name	Description
0x00	A	Peripheral function A selected
0x01	B	Peripheral function B selected
0x02	C	Peripheral function C selected
0x03	D	Peripheral function D selected
0x04	E	Peripheral function E selected
0x05	F	Peripheral function F selected
0x06	G	Peripheral function G selected
0x07	H	Peripheral function H selected
0x08	I	Peripheral function I selected
0x09	J	Peripheral function J selected
Other	—	Reserved

Bit 20 – SLEWLIM Output Driver Slew Rate Limit Enable

This bit determines the new value written to the PINCFG[n].SLEWLIM for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

Value	Description
0	Output Driver Slew Rate Limit is disabled
1	Output Driver Slew Rate Limit is enabled

Bit 18 – PULLEN Pull Enable

This bit determines the new value written to the PINCFG[n].PULLEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

Value	Description
0	The internal pull resistor is disabled, and the input is in a high-impedance configuration
1	The internal pull resistor is enabled, and the input is driven to a defined logic level in the absence of external input

Bit 17 – INEN Input Enable

This bit determines the new value written to the PINCFG[n].INEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

Value	Description
0	The input buffer for the I/O pin is disabled, and the input value will not be sampled
1	The input buffer for the I/O pin is enabled, and the input value will be sampled when required

Bit 16 – PMUXEN Peripheral Multiplexer Enable

This bit determines the new value written to the PINCFG[n].PMUXEN for all pins selected by the WRCONFIG.PINMASK and WRCONFIG.HWSEL bits, when the WRCONFIG.WRPINCFG bit is set.

Value	Description
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 function controls the direction and output drive value

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.

Writing a '0' to a bit has no effect.
Writing a '1' to a bit will update the configuration of the corresponding I/O pin in the half-word PORT group.

27.6.12. Event Control

Name: EVCTRL
Offset: 0x2C
Reset: 0x00000000
Property: PAC Write-Protection

This register allows the user to configure event actions for one or more I/O pins. There are up to four input events for each PORT group. Each byte of this register addresses one input event.

Bit	31	30	29	28	27	26	25	24
	PORTEI3	EVACT3[1:0]		PID3[4:0]				
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
	PORTEI2	EVACT2[1:0]		PID2[4:0]				
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
	PORTEI1	EVACT1[1:0]		PID1[4:0]				
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
	PORTEI0	EVACT0[1:0]		PID0[4:0]				
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, 15, 23, 31 – PORTEIn PORT Event n Input Enable
This bit field controls whether the input to Event n Action (EVACTn) is enabled.

Value	Description
0x00	The event action will not be triggered by any incoming event
0x01	The event action will be triggered by any incoming event

Bits 5:6, 13:14, 21:22, 29:30 – EVACTn Event n Action
This bit field controls the event action to execute on the pin configured by Event n Pin Identifier (PIDn).

Value	Name	Description
0x00	OUT	The output value of pin PIDn will be set to the level of the event
0x01	SET	Set the output value of pin PIDn on the rising edge of the event
0x02	CLR	Clear the output value of pin PIDn on the rising edge of the event
0x03	TGL	Toggle the output value of pin PIDn on the rising edge of the event

Bits 0:4, 8:12, 16:20, 24:28 – PIDn Event n Pin Identifier
This bit field selects the bit in the Data Output Value register (OUT) on which the Event n Action (EVACTn) will be executed.

Value	Name	Description
0x00	PIN0	Event actions will be executed on pin 0
0x01	PIN1	Event actions will be executed on pin 1
0x02	PIN2	Event actions will be executed on pin 2
0x03	PIN3	Event actions will be executed on pin 3

Value	Name	Description
0x04	PIN4	Event actions will be executed on pin 4
0x05	PIN5	Event actions will be executed on pin 5
0x06	PIN6	Event actions will be executed on pin 6
0x07	PIN7	Event actions will be executed on pin 7
0x08	PIN8	Event actions will be executed on pin 8
0x09	PIN9	Event actions will be executed on pin 9
0x0A	PIN10	Event actions will be executed on pin 10
0x0B	PIN11	Event actions will be executed on pin 11
0x0C	PIN12	Event actions will be executed on pin 12
0x0D	PIN13	Event actions will be executed on pin 13
0x0E	PIN14	Event actions will be executed on pin 14
0x0F	PIN15	Event actions will be executed on pin 15
0x10	PIN16	Event actions will be executed on pin 16
0x11	PIN17	Event actions will be executed on pin 17
0x12	PIN18	Event actions will be executed on pin 18
0x13	PIN19	Event actions will be executed on pin 19
0x14	PIN20	Event actions will be executed on pin 20
0x15	PIN21	Event actions will be executed on pin 21
0x16	PIN22	Event actions will be executed on pin 22
0x17	PIN23	Event actions will be executed on pin 23
0x18	PIN24	Event actions will be executed on pin 24
0x19	PIN25	Event actions will be executed on pin 25
0x1A	PIN26	Event actions will be executed on pin 26
0x1B	PIN27	Event actions will be executed on pin 27
0x1C	PIN28	Event actions will be executed on pin 28
0x1D	PIN29	Event actions will be executed on pin 29
0x1E	PIN30	Event actions will be executed on pin 30
0x1F	PIN31	Event actions will be executed on pin 31

27.6.13. Peripheral Multiplexing m

Name: PMUX[m]
Offset: 0x30 + m*0x01 [m=0..15]
Reset: 0x00
Property: PAC Write-Protection

There are up to 16 Peripheral Multiplexing registers in each group, one for every set of two consecutive I/O lines. The n denotes the number of the set of I/O lines.
Note: For the pin with the SWCLK peripheral function, the corresponding PMUX register reset value will reflect that configuration.

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 for Odd-Numbered Pin
These bits select the peripheral function for odd-numbered pins (2*n + 1) of a PORT group when the corresponding PINCFG[n].PMUXEN bit is '1'.
Not all possible values for this selection may be valid. For additional information, refer to the *Pinout* section.

Value	Name	Description
0x00	A	Peripheral function A selected
0x01	B	Peripheral function B selected
0x02	C	Peripheral function C selected
0x03	D	Peripheral function D selected
0x04	E	Peripheral function E selected
0x05	F	Peripheral function F selected
0x06	G	Peripheral function G selected
0x07	H	Peripheral function H selected
0x08	I	Peripheral function I selected
0x09	J	Peripheral function J selected
Other	—	Reserved

Bits 3:0 – PMUXE[3:0] Peripheral Multiplexing for Even-Numbered Pin
These bits select the peripheral function for even-numbered pins (2*n) of a PORT group when the corresponding PINCFG[n].PMUXEN bit is '1'.
Not all possible values for this selection may be valid. For additional information, refer to the *Pinout* section.

Value	Name	Description
0x00	A	Peripheral function A selected
0x01	B	Peripheral function B selected
0x02	C	Peripheral function C selected
0x03	D	Peripheral function D selected
0x04	E	Peripheral function E selected
0x05	F	Peripheral function F selected
0x06	G	Peripheral function G selected
0x07	H	Peripheral function H selected
0x08	I	Peripheral function I selected
0x09	J	Peripheral function J selected
Other	—	Reserved

27.6.14. Pin Configuration n

Name: PINCFG[n]
Offset: 0x40 + n*0x01 [n=0..31]
Reset: 0x00
Property: PAC Write-Protection

There are up to 32 Pin Configuration registers in each PORT group, one for each I/O line.

Bit	7	6	5	4	3	2	1	0
				SLEWLIM		PULLEN	INEN	PMUXEN
Access				R/W		R/W	R/W	R/W
Reset				0		0	0	0

Bit 4 – SLEWLIM Output Driver Slew Rate Limit Enable
This bit enables the internal slew rate limit of an I/O pin configured as an output.

Value	Description
0	Output Driver Slew Rate Limit is disabled
1	Output Driver Slew Rate Limit is enabled

Bit 2 – PULLEN Pull Enable
This bit enables the internal pull resistor of an I/O pin configured as an input.

Value	Description
0	The internal pull resistor is disabled, and the input is in a high-impedance configuration
1	The 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 Buffer Enable
This bit controls the input buffer of an I/O pin configured as either an input or output.
Writing a '0' 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.

Value	Description
0	The input buffer for the I/O pin is disabled, and the input value will not be sampled
1	The input buffer for the I/O pin is enabled, and the input value will be sampled when required

Bit 0 – PMUXEN Peripheral Multiplexer Enable
This bit enables or disables the peripheral multiplexer selection set in the Peripheral Multiplexing register (PMUX[m]), allowing alternative peripheral control over an I/O pin's direction and output drive value.
Writing a '0' to this bit allows the PORT to control the pad direction through the Data Direction register (DIR) and the output drive value through the Data Output Value register (OUT). The peripheral multiplexer value in PMUX[m] is ignored.
Writing a '1' to this bit enables the peripheral selection in PMUX[m] to control the pad. In this configuration, the physical pin state can still be read from the Data Input Value register (IN) if PINCFG[n].INEN is '1'.

Value	Description
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 function controls the direction and output drive value

28. SERCOM — Serial Communication Interface

28.1. Features

- Interface for Configuring as One of the Following:
  - Inter-Integrated Circuit (I^2C) Two-wire Serial Interface
  - Serial Peripheral Interface (SPI)
  - Universal Synchronous/Asynchronous Receiver/Transmitter (USART)
- Single Transmit Buffer and Double Receive Buffer
- Baud Rate Generator
- Address Match/Mask Logic
- Operational in all Sleep Modes with an Internal Clock Source
- Can be used with DMA

28.2. Overview

There are two instances of the serial communication interface (SERCOM) peripheral.

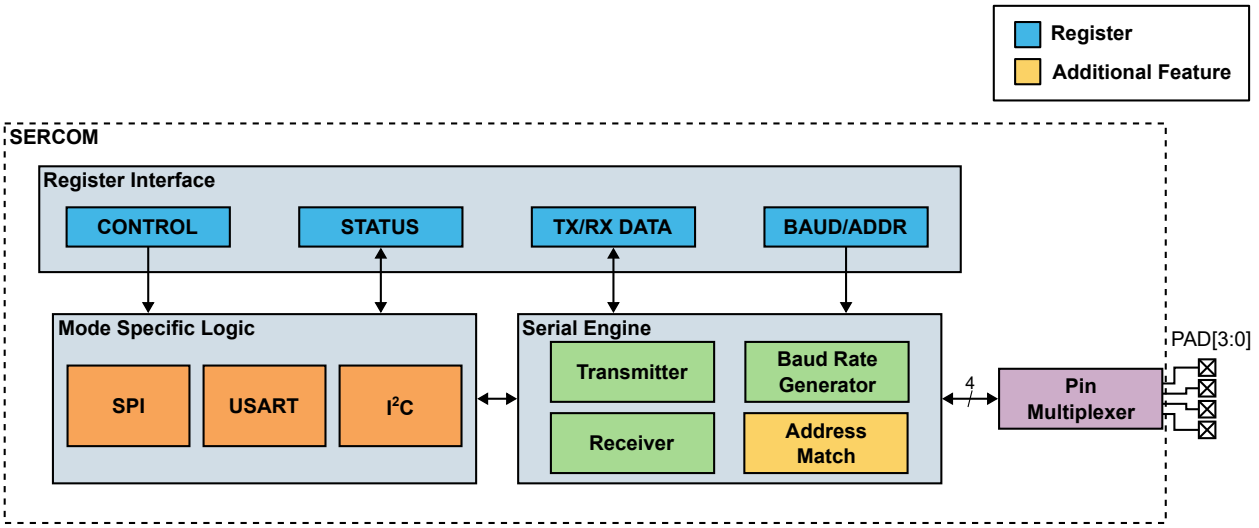
SERCOM0 can be configured to support all three modes: I^2C, SPI, and USART. SERCOM1 supports SPI and USART.

When an instance of SERCOM is configured and enabled, all of the resources of that SERCOM instance will be dedicated to the selected mode.

The SERCOM serial engine consists of a transmitter, a receiver, a baud rate generator, and address matching functionality. It can use either the internal generic clock or an external clock. Using an external clock allows the SERCOM to operate in all sleep modes.

28.3. Block Diagram

Figure 28-1. SERCOM Block Diagram



28.3.1. Signal Description

See the respective SERCOM mode chapters for details:

- SERCOM SPI
- SERCOM USART

- SERCOM I^2C

28.4. Functional Description

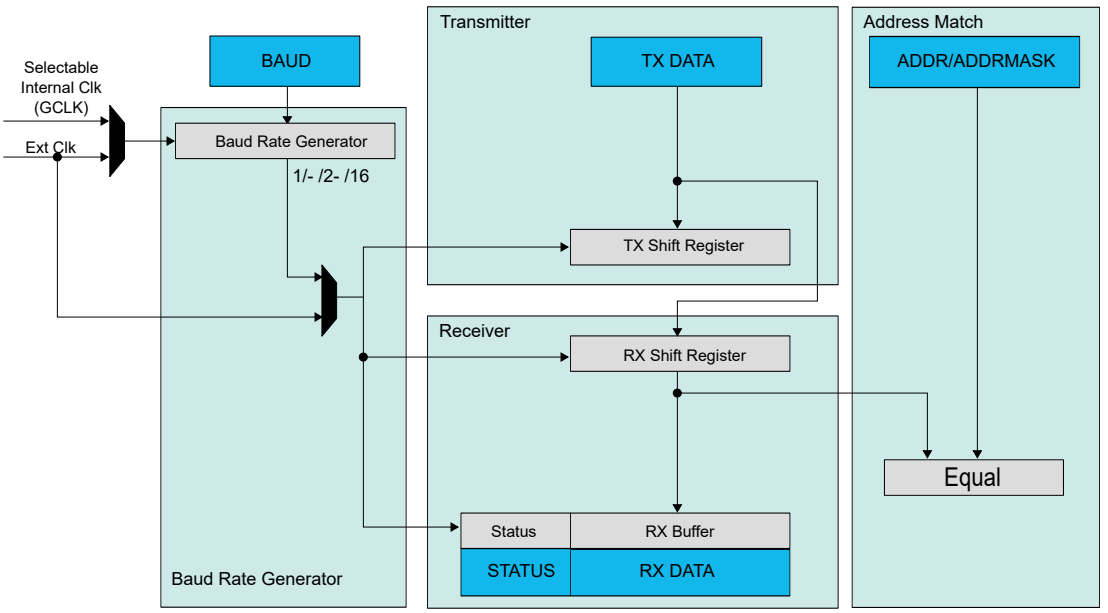
28.4.1. Initialization

The SERCOM must be configured to the desired mode by writing the Operating Mode bit field in the Control A register (CTRLA.MODE). Refer to each SERCOM mode for detailed initialization information.

28.4.2. Operation

The basic structure of the SERCOM serial engine is shown in Figure 28-2.

Figure 28-2. SERCOM Serial Engine



Registers with names in capital letters are synchronous to the system clock and accessible by the CPU, while those with names in lowercase letters can be configured to run on the GCLK_SERCOMx_CORE clock or an external clock.

The transmitter consists of a single write buffer and a shift register.

The receiver consists of a one-level (I^2C), two-level (USART, SPI) 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 incorporated to support both SPI and I^2C operations.

For further information, see the following chapters:

- SERCOM SPI
- SERCOM USART
- SERCOM I^2C

28.4.2.1. Clock Generation – Baud Rate Generator

The baud rate generator, as shown in Figure 28-3, generates internal clocks for both asynchronous and synchronous communication. The output frequency (f_{BAUD}) is determined by the configuration of the Baud register (BAUD) and the baud reference frequency (f_{ref}). The baud reference clock is the serial engine clock, which can be either internal or external.

For asynchronous communication, the /16 (divide-by-16) output is used when transmitting, whereas the /1 (divide-by-1) output is used while receiving.

For synchronous communication, the /2 (divide-by-2) output is used.

This functionality is automatically configured, depending on the selected operating mode.

Figure 28-3. Baud Rate Generator

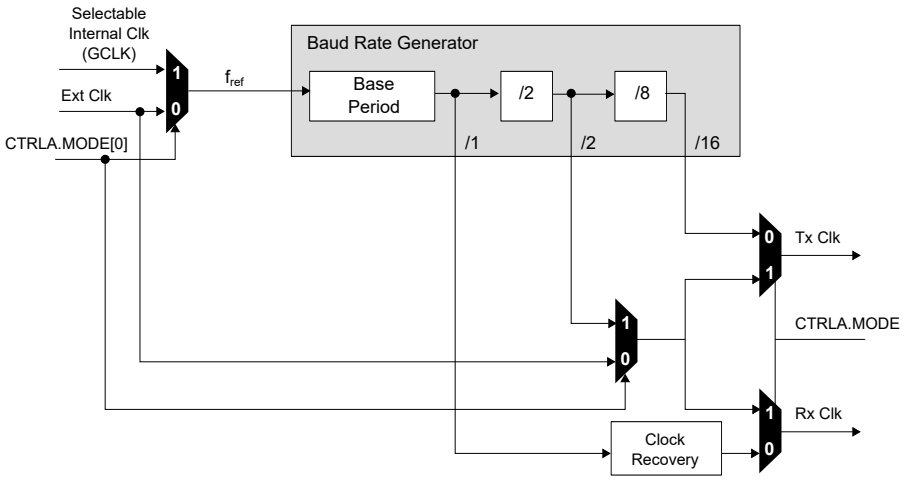


Table 28-1 contains equations for the baud rate (in bits per second) and the BAUD register value for each operating mode.

For asynchronous operation, there are two modes. In arithmetic mode, the BAUD register value is 16 bits (0 to 65,535). In fractional mode, the BAUD bit field is 13 bits, while the Fractional Part (BAUD.FP) bit field is 3 bits. The fractional part provides fine tuning of the baud rate. In the fractional mode, the BAUD register value must be 0x0001 or higher.

For synchronous operation, the BAUD register value is 8 bits (0 to 255).

Table 28-1. Baud Rate Equations

Operating Mode	Condition	Baud Rate (Bits Per Second)	BAUD Register Value Calculation
Asynchronous Arithmetic	$f_{BAUD} \leq \frac{f_{ref}}{16}$	$f_{BAUD} = \frac{f_{ref}}{16}\left(1 - \frac{BAUD}{65536}\right)$	$BAUD = 65536 \cdot \left(1 - 16 \cdot \frac{f_{BAUD}}{f_{ref}}\right)$
Asynchronous Fractional	$f_{BAUD} \leq \frac{f_{ref}}{S}$	$f_{BAUD} = \frac{f_{ref}}{S \cdot \left(BAUD + \frac{FP}{8}\right)}$	$BAUD = \frac{f_{ref}}{S \cdot f_{BAUD}} - \frac{FP}{8}$
Synchronous	$f_{BAUD} \leq \frac{f_{ref}}{2}$	$f_{BAUD} = \frac{f_{ref}}{2 \cdot (BAUD + 1)}$	$BAUD = \frac{f_{ref}}{2 \cdot f_{BAUD}} - 1$

S - Number of samples per bit, which can be 16, 8, or 3.

The Asynchronous Fractional option is used for auto-baud detection.

The baud rate error is represented by the following formula:

$$\text{Error} = 1 - \left(\frac{\text{ExpectedBaudRate}}{\text{ActualBaudRate}}\right)$$

28.4.2.1.1. Asynchronous Arithmetic Mode BAUD Value Selection

The formula given for f_{BAUD} calculates the average frequency over 65536 f_{ref} cycles. Although the BAUD register can be set to any value between 0 and 65536, the actual average frequency of f_{BAUD} over a single frame is more granular. BAUD register values that affect the average frequency over a single frame result in an integer increase in the cycles per frame (CPF).

$$CPF = \frac{f_{ref}}{f_{BAUD}}(D + S)$$

where

- *D* represents the number of data bits per frame
- *S* represents the sum of the start bit and the first stop bit, if present

Table 28-2 shows the BAUD register value versus baud frequency f_{BAUD} at a serial engine frequency of 48 MHz. This assumes a *D* value of eight bits and an *S* value of two bits (ten bits, including start and stop bits).

Table 28-2. BAUD Register Value vs. Baud Frequency

BAUD Register Value	Serial Engine CPF	f_{BAUD} at 48 MHz Serial Engine Frequency (f_{REF})
0 – 406	160	3 MHz
407 – 808	161	2.981 MHz
809 – 1205	162	2.963 MHz
...	...	...
65206	31775	15.11 kHz
65207	31871	15.06 kHz
65208	31969	15.01 kHz

28.4.3. Additional Features

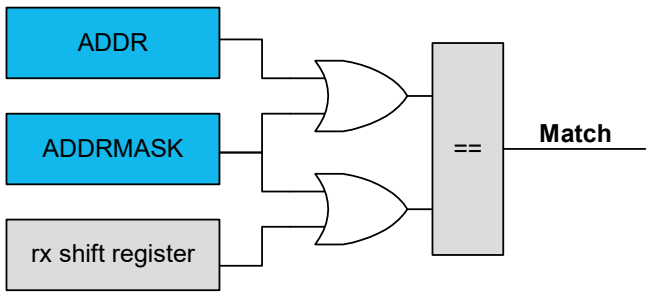
28.4.3.1. Address Match and Mask

The SERCOM address match and mask feature is capable of matching either 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.

28.4.3.1.1. Address With Mask

An address received that corresponds to the value written to the Address bits in the Address register (ADDR.ADDR), combined with a mask written to the Address Mask bits in the Address register (ADDR.ADDRMASK), results in an address match. All bits that are masked are not included in the match. Note that writing ADDR.ADDRMASK to all zeros will match a single unique address, while writing ADDR.ADDRMASK to all ones will result in all addresses being accepted.

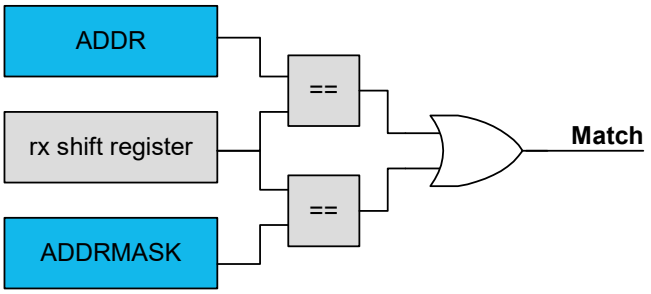
Figure 28-4. Address With Mask



28.4.3.1.2. Two Unique Addresses

The two addresses written to ADDR and ADDRMASK will result in a match.

Figure 28-5. Two Unique Addresses



28.4.3.1.3. Address Range

The range of addresses between, and including, ADDR.ADDR and ADDR.ADDRMASK will result in a match. ADDR.ADDR and ADDR.ADDRMASK can be set to any two addresses, with ADDR.ADDR serving as the upper limit and ADDR.ADDRMASK as the lower limit.

Figure 28-6. Address Range

28.4.4. 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.

28.4.5. Debug Operation

When the CPU is halted in debug mode the SERCOM will continue normal operation. Setting the Debug Stop Mode bit in the Debug Control register (DBGCTRL.DBGSTOP) forces the SERCOM to halt operation when the CPU is halted in debug mode.

28.5. Dependencies

28.5.1. I/O Lines

Using the SERCOM's I/O lines requires the I/O pins to be configured.

The SERCOM has four internal pads, PAD[3:0], and the signals from I^2C, SPI and USART are routed through these pads via a multiplexer. The configuration of the multiplexer is determined by the selected SERCOM mode

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

28.5.2. Power Management

The SERCOM will continue to operate in any sleep mode where the selected source clock is running. The SERCOM'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 the *PM – Power Manager* chapter for details on the different sleep modes.

28.5.3. Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the main clock, and the default state of CLK_SERCOMx_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

The SERCOM uses two generic clocks: The Core clock (GCLK_SERCOMx_CORE) and the Slow clock (GCLK_SERCOMx_SLOW). The Core clock is required for SERCOM operation as a host, while the Slow clock is only needed for certain functions. See specific mode chapters for details.

These clocks must be configured and enabled in the generic clock controller before using the SERCOM. Refer to the *GCLK – Generic Clock Controller* chapter for details.

The 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.

28.5.4. DMA

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the SERCOM's DMA requests. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

28.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use SERCOM interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

For peripheral specific configurations, refer to the corresponding section for each SERCOM mode.

28.5.6. Events

Not applicable.

28.5.7. Analog Connections

Not applicable.

29. SERCOM USART — SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter

29.1. Features

- Full-Duplex Operation
- Asynchronous Operation 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 Checking
- Selectable LSb or MSb First Data Transfer
- Buffer Overflow and Frame Error Detection
- Noise Filtering, Including False Start-Bit Detection and Digital Low-Pass Filtering
- Collision Detection
- Operates 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
- RTS (Request To Send) and CTS (Clear To Send) Flow Control
- IrDA Modulation and Demodulation up to 115.2 kbps
- LIN Host Support
- LIN Client Support
  - Auto-baud and break character detection
- RS485 Support
- Start-of-Frame Detection
- Can be used with DMA

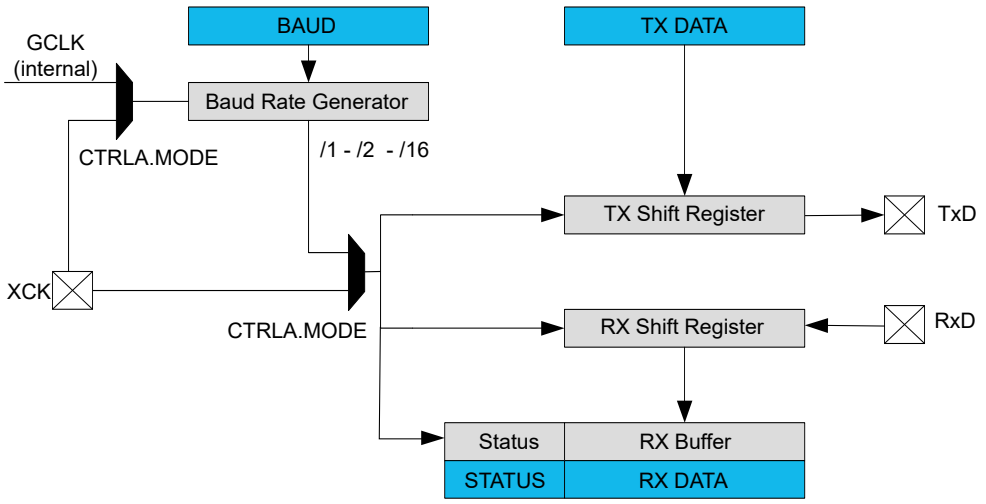
29.2. Overview

The Universal Synchronous and Asynchronous Receiver and Transmitter (USART) is one of the available modes in the SERCOM — Serial Communication Interface.

The transmitter consists of a single write buffer, a shift register, and control logic for different frame formats. The write buffer supports 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.

29.3. Block Diagram

Figure 29-1. USART Block Diagram



Registers labeled in uppercase are synchronous to CLK_SERCOMx_APB and accessible from the CPU, while those with lowercase labels can be programmed to run on the internal generic clock or an external clock.

29.3.1. Signal Description

Table 29-1. SERCOM USART Signals

Signal Name	Type	Description
PADn	Digital I/O	SERCOM pins

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter for details.

29.4. Functional Description

29.4.1. Initialization

1. Select either external or internal clock by writing the Operating Mode value in the Control A register (CTRLA.MODE).
2. Select either asynchronous or synchronous communication mode by writing the Communication Mode bit in the Control A register (CTRLA.CMODE).
3. Select the pin for receive data by writing the appropriate Receive Data Pinout value to the Control A register (CTRLA.RXPO).
4. Select the pads for the transmitter and external clock by writing the appropriate Transmit Data Pinout bit to the Control A register (CTRLA.TXPO).
5. When using an internal clock, write the Baud register (BAUD) to generate the desired baud rate.
6. Enable the transmitter and receiver by writing '1' to the Receiver Enable and Transmitter Enable bits in the Control B register (CTRLB.RXEN and CTRLB.TXEN).
7. Enable the module by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE).

29.4.2. Operation

The USART uses the following lines for data transfer:

- RxD for receiving
- TxD for transmitting

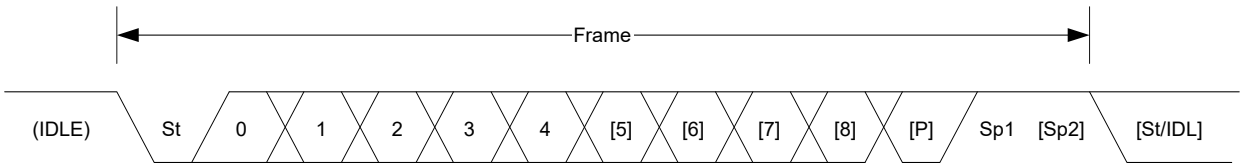
- XCK for the transmission clock in synchronous operation

USART data transfer is frame based. A serial frame consists of:

- One start bit
- From five to nine data bits (MSb or LSb first)
- No, even or odd parity bit
- One or two 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. After the stop bit(s) of a frame, the next frame can either follow immediately, or the communication line can return to the idle (high) state. The figure below illustrates the possible frame formats. Values inside brackets ([x]) denote optional bits.

Figure 29-2. Frame Formats



St	Start bit. Signal is always low.
n, [n]	Data bits. 0 to [5..9]
[P]	Parity bit. Either odd or even.
Sp, [Sp]	Stop bit. Signal is always high.
IDL	No frame is transferred on the communication line. Signal is always high in this state.

29.4.2.1. 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 '1' to the Communication Mode bit in the Control A register (CTRLA.CMODE), while Asynchronous mode is selected by writing '0' to CTRLA.CMODE.

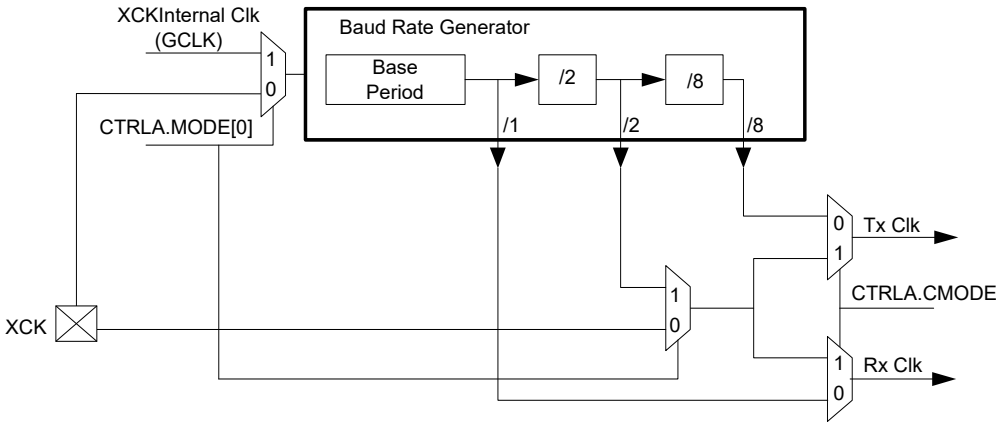
The internal clock source is selected by writing '1' to the Operation Mode bit field in the Control A register (CTRLA.MODE), while the external clock source is selected by writing '0' to CTRLA.MODE.

The SERCOM baud rate generator is configured as in the figure below.

In Asynchronous mode, the 16-bit Baud register value is used.

In Synchronous mode, the eight LSbs of the Baud register are used. Refer to Clock Generation – Baud-Rate Generator for details on configuring the baud rate.

Figure 29-3. Clock Generation



29.4.2.1.1. Synchronous Clock Operation

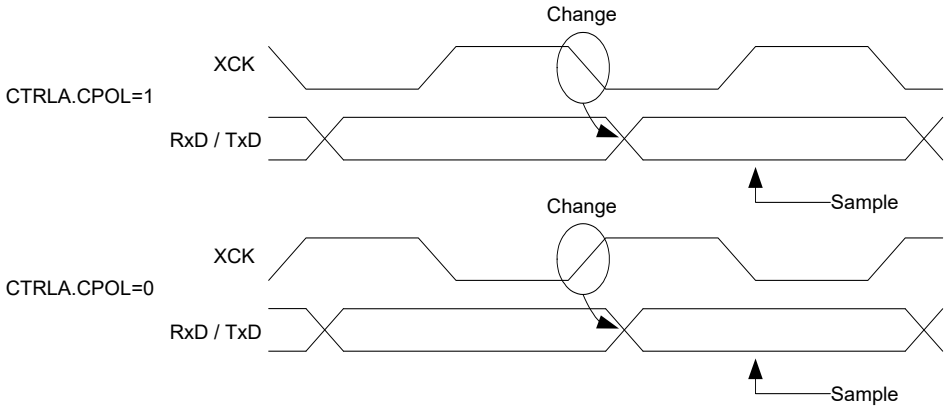
In Synchronous mode, the CTRLA.MODE bit field determines whether the transmission clock line (XCK) serves as an input or an output. The relationship between clock edges, data sampling, and data changes is the same for both internal and external clocks. Data input on the RxD pin is sampled on the opposite XCK clock edge from when 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:

When CTRLA.CPOL is '0', the data will be changed on the rising edge of XCK, and sampled on the falling edge of XCK.

When CTRLA.CPOL is '1', the data will be changed on the falling edge of XCK, and sampled on the rising edge of XCK.

Figure 29-4. Synchronous Mode XCK Timing



When the clock is provided through XCK, 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.

29.4.2.2. 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 TxDATA register. Reading the DATA register will return the contents of the RxDATA register.

29.4.2.3. Data Transmission

Data transmission is initiated by writing the data to be sent into the DATA register. Then, the data in TxDATA will be moved to the shift register when the shift register is empty and ready to send a new frame. After the shift register is loaded with data, the data frame will be transmitted.

When the entire data frame, including stop bit, has been transmitted (i.e., both the Tx buffer and shift register are empty) and no new data has been written to the DATA register, the Transmit Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set, and the optional interrupt will be generated.

The Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) indicates that the register is empty and ready for new data. The DATA register must only be written to when INTFLAG.DRE is set.

29.4.2.3.1. Disabling the Transmitter

The transmitter is disabled by writing '0' to the Transmitter Enable bit in the CTRLB register (CTRLB.TXEN).

Disabling the transmitter will only take effect after any ongoing and pending transmissions are completed; that is, there is no data in the transmit shift register or TxDATA to transmit.

29.4.2.4. Data Reception

The receiver accepts data when a valid Start bit is detected. Each bit following the Start bit is sampled according to the baud rate or XCK clock and shifted into the receive shift register until the first Stop bit of a frame is received. The second Stop bit will be ignored by the receiver.

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. Then, the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) will be set, and the optional interrupt can be generated.

The received data can be read from the DATA register when the Receive Complete Interrupt flag is set.

29.4.2.4.1. Disabling the Receiver

Writing '0' to the Receiver Enable bit in the CTRLB register (CTRLB.RXEN) will disable the receiver, flush the two-level receive buffer, and cause any data from ongoing receptions to be lost.

29.4.2.4.2. Error Bits

The USART receiver has three error bits in the Status (STATUS) register: Frame Error (FERR), Buffer Overflow (BUFOVF), and Parity Error (PERR). When an error occurs, the corresponding error bit is set and remains set until it is cleared by writing '1' to it. These bits are also cleared automatically when the receiver is disabled.

There are two methods for buffer overflow notification, selected by the Immediate Buffer Overflow Notification bit in the Control A register (CTRLA.IBON):

When CTRLA.IBON is set to '1', the STATUS.BUFOVF flag is triggered immediately upon a buffer overflow. At this point, software can clear the receive FIFO by reading from RxDATA until the Receiver Complete Interrupt flag (INTFLAG.RXC) is cleared.

When CTRLA.IBON is set to '0', the Buffer Overflow condition is indicated while data are still being received through the FIFO. After the received data are read, the STATUS.BUFOVF and INTFLAG.ERROR flags will be set, along with the Receiver Complete Interrupt flag INTFLAG.RXC.

29.4.2.4.3. Asynchronous Data Reception

The USART includes a clock recovery and data recovery unit for handling asynchronous data reception.

The clock recovery logic can 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.

29.4.2.4.4. Asynchronous Operational Range

The operational range of the asynchronous reception depends on the accuracy of the internal baud rate clock, the rate of the incoming frames, and the frame size (in number of bits). In addition, 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: First, the reference clock will always have some minor instability. Second, the baud rate generator cannot always do an exact division of the reference clock frequency to get the baud rate desired. In this case, the BAUD register value must be set to give the lowest possible error. Refer to Clock Generation – Baud-Rate Generator for details.

The recommended maximum receiver baud rate errors for various character sizes are shown in the following table.

Table 29-2. Asynchronous Receiver Error for 16-fold Oversampling

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

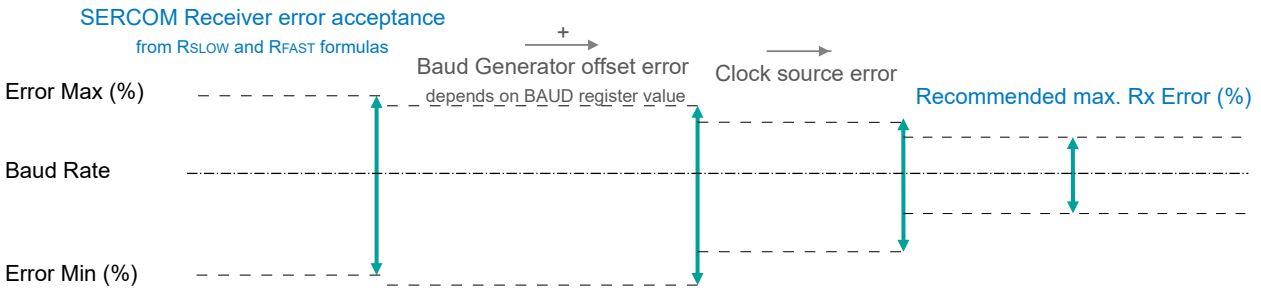
The following equations calculate the ratio of the incoming data rate and internal receiver baud rate:

$$R_{\mathrm{SLOW}} = \frac{(D+1)S}{S-1+D\cdot S+S_F} \quad , \qquad R_{\mathrm{FAST}} = \frac{(D+2)S}{(D+1)S+S_M}$$

- R_{SLOW} is the ratio of the slowest acceptable incoming data rate to the receiver baud rate
- R_{FAST} is the ratio of the fastest acceptable incoming data rate to the receiver baud rate
- D is the sum of the character size and the parity size (D = 5 to 10 bits)
- S is the number of samples per bit (S = 16, 8 or 3)
- S_F is the first sample number used for majority voting (S_F = 7, 3, or 2) when CTRLA.SAMPA=0
- S_M is the middle sample number used for majority voting (S_M = 8, 4, or 2) when CTRLA.SAMPA=0

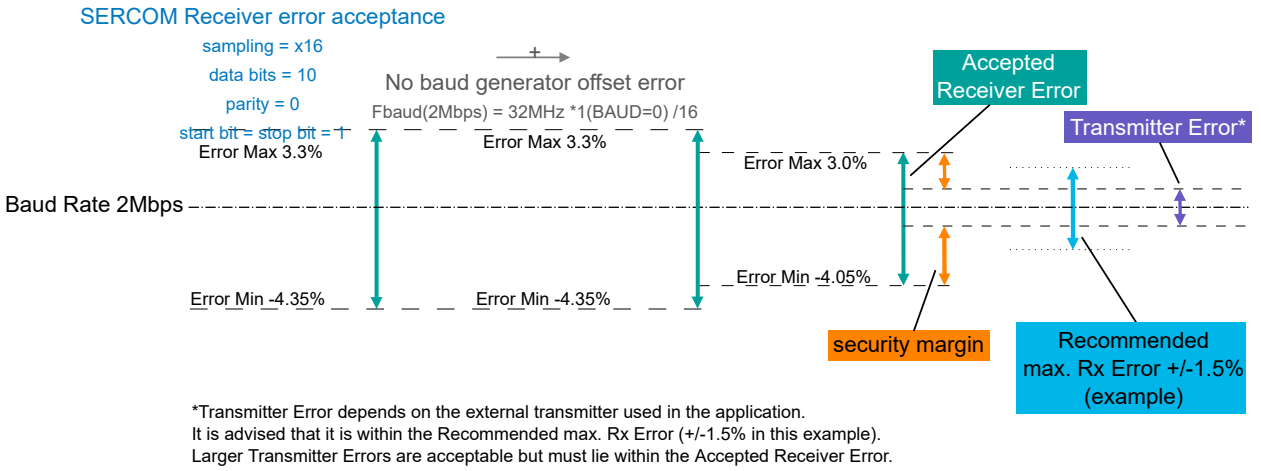
The recommended maximum Rx Error assumes that the receiver and transmitter equally divide the maximum total error. Its connection to the SERCOM Receiver error acceptance is depicted in this figure:

Figure 29-5. USART Rx Error Calculation



The recommendation values in Table 29-2 account for errors from both the clock source and the baud generator. The following figure provides an example for a baud rate of 3 Mbps:

Figure 29-6. USART Rx Error Calculation Example



*Transmitter Error depends on the external transmitter used in the application.
It is advised that it is within the Recommended max. Rx Error (+/-1.5% in this example).
Larger Transmitter Errors are acceptable but must lie within the Accepted Receiver Error.

29.4.3. Additional Features

29.4.3.1. Parity

Parity can be enabled for error checking by writing `USART_FRAME_WITH_PARITY` to the Frame Format bit field in the Control A register (CTRLA.FORM). Even parity is selected by writing the Parity mode bit in the Control B register (CTRLB.PMODE) to '0'. Odd parity is selected by writing CTRLB.PMODE to '1'.

If *even parity* is selected, the parity bit of an outgoing frame is '1' if the data contains an odd number of bits that are '1', making the total number of '1' even.

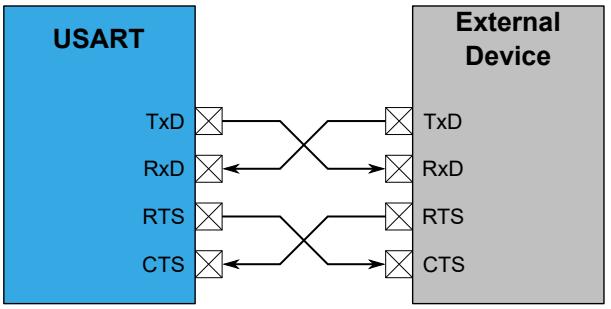
If *odd parity* is selected, the parity bit of an outgoing frame is '1' if the data contains an even number of bits that are '1', making the total number of '1' odd.

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.

29.4.3.2. Hardware Handshaking

The USART features an out-of-band hardware handshaking flow control mechanism, implemented by connecting the RTS and CTS pins with the remote device, as shown in the figure below.

Figure 29-7. Connection With a Remote Device for Hardware Handshaking

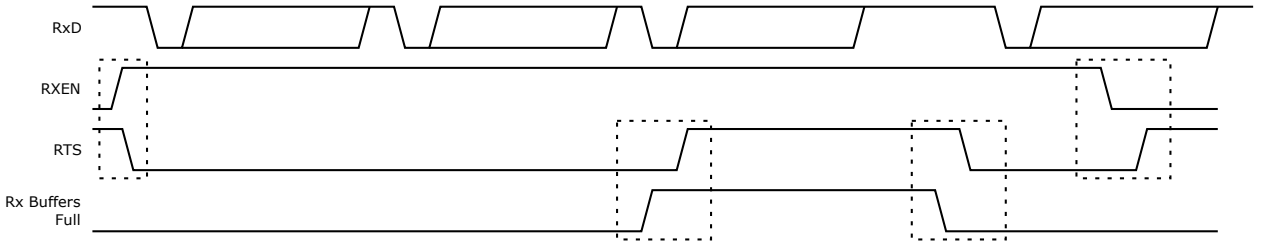


Hardware handshaking is only available in the following configuration:

- USART with internal clock (CTRLA.MODE=1)
- Asynchronous mode (CTRLA.CMODE=0)
- Flow control pinout (CTRLA.TXPO=2)

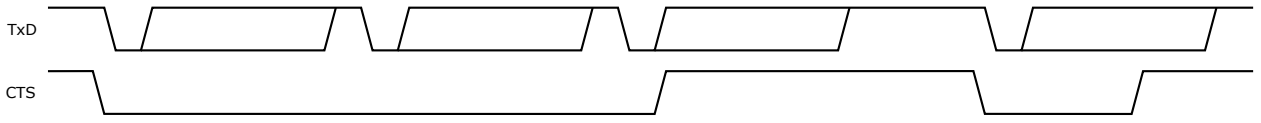
When the receiver is disabled or the receive FIFO is full, the receiver drives the RTS pin high. This notifies the remote device to stop transmission after the ongoing transfer. Enabling or disabling the receiver by writing to CTRLB.RXEN will set or clear the RTS pin after a synchronization delay. When the receive FIFO becomes full, RTS is set immediately, and the frame being received will be stored in the shift register until the receive FIFO is no longer full.

Figure 29-8. Receiver Behavior When Operating With Hardware Handshaking



The current CTS Status is reflected in the STATUS register (STATUS.CTS). Character transmission will start only when CTS is cleared. When CTS is set, the transmitter will complete the ongoing transmission and stop transmitting.

Figure 29-9. Transmitter Behavior When Operating With Hardware Handshaking



29.4.3.3. IrDA Modulation and Demodulation

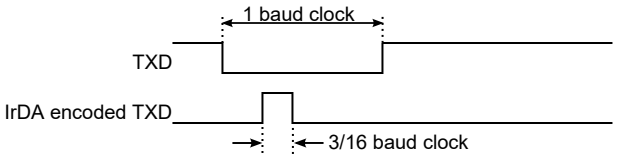
Transmission and reception can be encoded IrDA compliant up to 115.2 kb/s. IrDA modulation and demodulation work in the following configuration:

- IrDA encoding enabled (CTRLB.ENC = 1)
- Asynchronous mode (CTRLA.CMODE = 0)

- 16x sample rate (CTRLA.SAMPR[0] = 0)

During transmission, each low bit is transmitted as a high pulse. The pulse width is 3/16 of the baud rate period, as illustrated in the figure below.

Figure 29-10. IrDA Transmit Encoding



The reception decoder has two main functions.

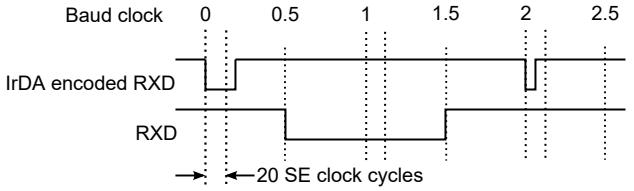
The first is to synchronize the incoming data to the IrDA baud rate counter. Synchronization is performed at the start of each zero pulse.

The second main function is to decode incoming Rx data. If a pulse width meets the minimum length set by the configuration (RXPL.RXPL), it is accepted. When the baud rate counter reaches its middle value (1/2 bit length), the data is transferred to the receiver.

Note: The polarity of the transmitter and receiver are opposite: During transmission, a '0' bit is transmitted as a '1' pulse, while during reception, an accepted '0' pulse is received as a '0' bit.

Example: The figure below illustrates reception with RXPL.RXPL set to 19. This means that the pulse width must be at least 20 SE clock cycles. When using BAUD = 0xE666, or 160 SE cycles per bit, this corresponds to a minimum required pulse width of 2/16 of the baud clock. In this case, the first bit is accepted as a '0', the second bit is a '1', and the third bit is also a '1'. A low pulse is rejected because it does not meet the minimum requirement of 2/16 of the baud clock.

Figure 29-11. IrDA Receive Decoding

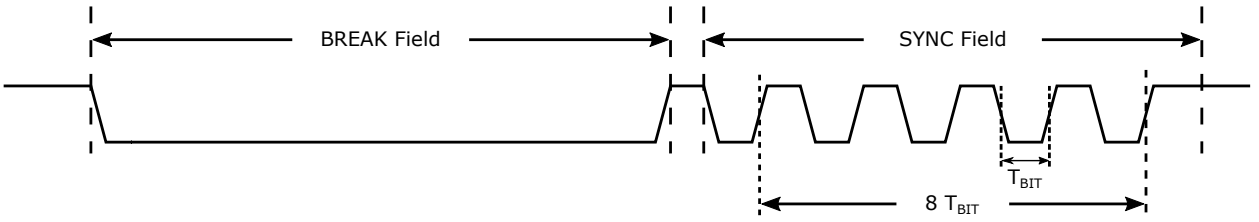


29.4.3.4. Break Character Detection and Auto-Baud

Break character detection and auto-baud are available in the following configuration:

- Auto-baud frame format (CTRLA.FORM = 0x04 or 0x05)
- Asynchronous mode (CTRLA.CMODE = 0)
- 16x sample rate using fractional baud rate generation (CTRLA.SAMPR = 1)

Figure 29-12. Auto-Baud Timing



The USART uses a break detection threshold of greater than 11 nominal bit times at the configured baud rate. At any time, if more than 11 consecutive dominant (`'0'`) bits are detected on the bus, the USART detects a Break Field. When a Break Field has been detected, the Receive Break interrupt flag (INTFLAG.RXBRK) is set and the USART expects the Sync Field character to be 0x55. This field is used to update the actual baud rate to stay synchronized. If the received Sync character is not 0x55, then the Inconsistent Sync Field error flag (STATUS.ISF) is set along with the Error interrupt flag (INTFLAG.ERROR), and the baud rate is unchanged.

After a break field is detected and the start bit of the Sync Field is detected, a counter is started. The counter is then incremented for the next eight T_{bit} periods of the Sync Field. At the end of these 8-bit times, the counter is stopped. At this point, the 13 most significant bits of the counter (counter value divided by 8) provide the new clock divider (BAUD.BAUD), and the three least significant bits (the remainder) provide the new Fractional Part (BAUD.FP).

When the Sync Field has been received, the clock divider (BAUD.BAUD) and the Fractional Part (BAUD.FP) are updated after a synchronization delay. After the Break and Sync Fields are received, n characters of data can be received.

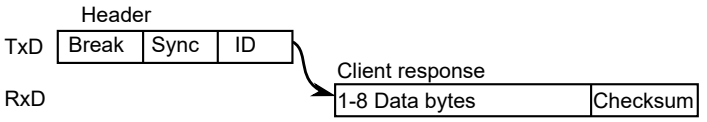
29.4.3.5. LIN Host

LIN Host is available with the following configuration:

- LIN Host format (CTRLA.FORM = 0x02)
- Asynchronous mode (CTRLA.CMODE = 0)
- 16x sample rate using fractional baud rate generation (CTRLA.SAMPR = 1)

LIN frames start with a header transmitted by the host. The header consists of the break, sync, and identifier fields. After the host transmits the header, the addressed client will respond with 1–8 bytes of data plus checksum.

Figure 29-13. LIN Frame Format



Using the LIN command field (CTRLB.LINCMD), the complete header can be automatically transmitted, or software can control transmission of the various header components.

When CTRLB.LINCMD = 0x1, software controls transmission of the LIN header. In this case, software uses the following sequence.

- CTRLB.LINCMD is written to 0x1
- Writing `0x00` to the DATA register triggers transmission of the break field by hardware. Note that writing any other value to the DATA register will also result in the transmission of the break field by the hardware.
- `0x55` written to the DATA register. The 0x55 value (sync) is transmitted.
- The identifier written to the DATA register. The identifier is transmitted.

Note: It is recommended to use the Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) to ensure no extra delay is added during the transmission of the data.

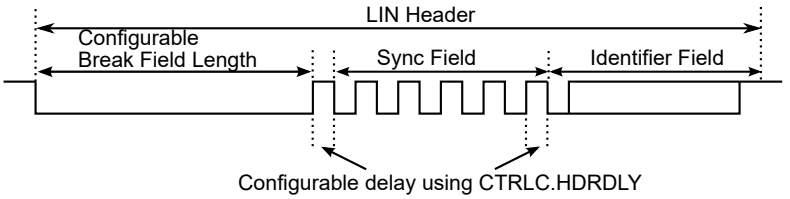
When CTRLB.LINCMD = 0x2, hardware controls transmission of the LIN header. In this case, software uses the following sequence.

- CTRLB.LINCMD is written to 0x2

- Writing the identifier to the DATA register triggers transmission of the complete header by hardware. First, the break field is transmitted. Next, the sync field is transmitted, followed by the sync field, and finally the identifier.

In LIN Host mode, the length of the break field is programmable using the break length field (CTRLC.BRKLEN). When the LIN header command is used (CTRLB.LINCMD = 0x2), the delay between the break and sync fields, in addition to the delay between the sync and ID fields, are configurable using the header delay field (CTRLC.HDRDLY). When manual transmission is used (CTRLB.LINCMD = 0x1), the software controls the delay between break and sync.

Figure 29-14. LIN Header Generation



After header transmission is complete, the client responds with 1–8 data bytes plus checksum.

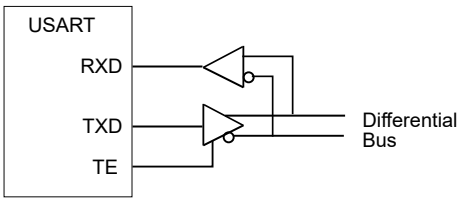
29.4.3.6. RS485

The RS485 is available with the following configuration:

- USART frame format (CTRLA.FORM = 0x00 or 0x01)
- RS485 pinout (CTRLA.TXPO = 0x3).

The RS485 feature enables control of an external line driver, as shown in the figure below. While operating in RS485 mode, the transmit enable pin (TE) is driven high when the transmitter is active.

Figure 29-15. RS485 Bus Connection



The TE pin will remain high for the complete frame including stop bits. If a Guard Time is programmed in the Control C register (CTRLC.GTIME), the line will remain driven after the last character completion. The following figure shows a transfer with one stop bit and CTRLC.GTIME = 3.

Figure 29-16. Example of TE Drive With Guard Time

The Transmit Complete interrupt flag (INTFLAG.TXC) will be raised after the guard time is complete and TE goes low.

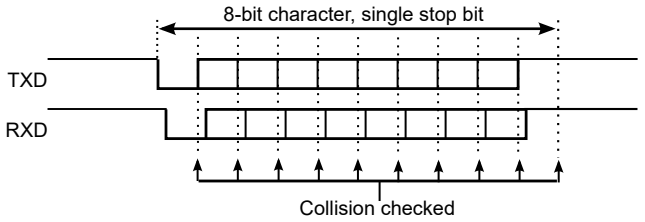
29.4.3.7. Collision Detection

When the receiver and transmitter are connected either through pin configuration or externally, transmit collision can be detected after selecting the Collision Detection Enable bit in the CTRLB register (CTRLB.COLDEN = 1). To detect collision, the receiver and transmitter must be enabled

(CTRLB.RXEN = 1 and CTRLB.TXEN = 1) and the peripheral bus (APB) must operate at or above the SERCOMx GCLK frequency.

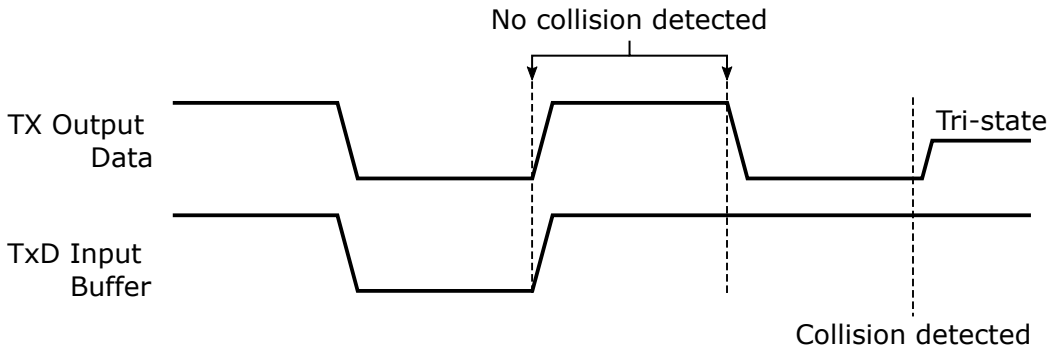
Collision detection is performed for each bit transmitted by comparing the received value with the transmit value, as shown in the figure below. While the transmitter is idle (no transmission in progress), characters can be received on RxD without triggering a collision.

Figure 29-17. Collision Checking



The figure below illustrates the conditions for a collision detection. In this case, the start bit and the first data bit are received with the same value as transmitted. The second received data bit is found to be different than the transmitted bit at the detection point, which indicates a collision.

Figure 29-18. Collision Detected



When a collision is detected, the USART follows this sequence:

1. Abort the current transfer.
2. Flush the transmit buffer.
3. Disable transmitter (CTRLB.TXEN = 0).
   - This is done after a synchronization delay. The CTRLB Synchronization Busy bit (SYNCBUSY.CTRLB) will be set until this is complete.
   - After disabling, the TxD pin will be tri-stated
4. Set the Collision Detected bit (STATUS.COLL) along with the Error interrupt flag (INTFLAG.ERROR).
5. Set the Transmit Complete interrupt flag (INTFLAG.TXC), because the transmit buffer no longer contains data.

After a collision, the software must manually enable the transmitter again before continuing, after assuring that the CTRLB Synchronization Busy bit (SYNCBUSY.CTRLB) is not set.

29.4.3.8. Loop-Back Mode

For Loop-Back mode, configure the Receive Data Pinout (CTRLA.RXPO) and Transmit Data Pinout (CTRLA.TXPO) to use the same data pin for transmit and receive. The loop-back is through the pad, therefore the signal is also available externally.

29.4.3.9. Start-of-Frame Detection

The USART start-of-frame detector can wake up the CPU when it detects a start bit. For this feature to work, the device must be in Standby sleep mode, the internal fast start-up oscillator must be selected as the GCLK_SERCOMx_CORE source, and the Run Standby bit (CTRLA.RUNSTDBY) must be set to '0'.

When a 1-to-0 transition is detected on RxD, the SERCOM requests the Generic Clock (GCLK) and the GCLK_SERCOMx_CORE starts. After start-up, the rest of the data frame can be received, provided that the baud rate is slow enough in relation to the fast start-up internal oscillator start-up time. Refer to the *Electrical Characteristics* section for details.

The USART start-of-frame detection works both in Asynchronous and Synchronous modes. It is enabled by writing '1' to the Start of Frame Detection Enable bit in the Control B register (CTRLB.SFDE).

If the Receive Start Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.RXS) is set, the Receive Start interrupt is generated immediately when a start is detected.

When using start-of-frame detection without the Receive Start interrupt, start detection requests the GCLK_SERCOMx_CORE to receive the rest of the frame. In this case, the CPU will not wake up until the Receive Complete interrupt is generated.

29.4.3.10. Sample Adjustment

In Asynchronous mode (CTRLA.CMODE = 0), three samples in the middle are used to determine the value based on majority voting. The three samples used for voting can be selected using the Sample Adjustment bit field in Control A register (CTRLA.SAMPA). When CTRLA.SAMPA = 0, samples 7-8-9 are used for 16x oversampling, and samples 3-4-5 are used for 8x oversampling.

Note: In Asynchronous mode, the start of frame may not occur at the UART clock reference rising edge, meaning the counter can start incrementing from 0 to 1 in less than one UART clock reference period. The counter will then continue to increment at each positive edge of the UART clock reference regardless of the incoming bits.

29.4.4. Sleep Mode Operation

The behavior in sleep mode depends on the clock source and the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY):

- Internal clocking, CTRLA.RUNSTDBY = 1: GCLK_SERCOMx_CORE can be enabled in all sleep modes. Any interrupt can wake up the device (except Frame error (FERR) and Parity error (PERR) that are part of the ERROR interrupt).
- External clocking, CTRLA.RUNSTDBY = 1: The Receive Start and the Receive Complete interrupts can wake up the device. Any interrupt can wake up the device (except Frame error (FERR) and Parity error (PERR) that are part of the ERROR interrupt).
- Internal clocking, CTRLA.RUNSTDBY = 0: The internal clock will be disabled after any ongoing transfer is completed. The Receive Start and Receive Complete interrupt(s) can wake up the device.
- External clocking, CTRLA.RUNSTDBY = 0: The external clock will be disconnected after any ongoing transfer is completed. All reception will be dropped.

When targeting the lowest STANDBY power consumption for a transmit-only application, the receiver must remain disabled, but the receiver data pinout (CTRLA.RXPO) must be set to match the transmit data pinout (CTRLA.TXPO).

29.4.5. Debug Operation

When the CPU is halted in debug mode the SERCOM will continue normal operation. Setting the Debug Stop Mode bit in the Debug Control register (DBGCTRL.DBGSTOP) forces the SERCOM to halt operation when the CPU is halted in debug mode.

29.5. Dependencies

29.5.1. I/O Lines

Using the SERCOM's I/O lines requires the I/O pins to be configured.

The SERCOM has four internal pads, PAD[3:0], and the signals from I^2C, SPI and USART are routed through these pads via a multiplexer. The configuration of the multiplexer is determined by the selected SERCOM mode

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

29.5.2. Power Management

The SERCOM will continue to operate in any sleep mode where the selected source clock is running. The SERCOM'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 the *PM – Power Manager* chapter for details on the different sleep modes.

29.5.3. Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the main clock, and the default state of CLK_SERCOMx_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

The SERCOM uses two generic clocks: The Core clock (GCLK_SERCOMx_CORE) and the Slow clock (GCLK_SERCOMx_SLOW). The Core clock is required for SERCOM operation as a host, while the Slow clock is only needed for certain functions. See specific mode chapters for details.

These clocks must be configured and enabled in the generic clock controller before using the SERCOM. Refer to the *GCLK – Generic Clock Controller* chapter for details.

The 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.

29.5.4. DMA

The USART generates the following DMA requests:

- Receive Complete (RXC): The request is set when data is available in the receive buffer (RX DATA). The request is cleared when DATA is read.
- Data Register Empty (DRE): The request is set when the transmit buffer (TX DATA) is empty. The request is cleared when DATA is written.

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the USART's DMA requests. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

29.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use SERCOM USART interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the

interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 29-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
SERCOMn	DRE	Data Register Empty	
	RXC	Receive Complete	
	TXC	Transmit Complete	
	RXS	Receive Start	
	CTSIC	Clear to Send Input Change	The Communication Mode, Transmit Data Pinout and Operating Mode bit fields in the Control A register (CTRLA.CMODE, CTRLA.TXPO and CTRLA.MODE).
	RXBRK	Received Break	The Communication Mode and Frame Format bit fields in the Control A register (CTRLA.CMODE and CTRLA.FORM).
	ERROR	Error	The Collision Detected, Inconsistent Sync Field, Buffer Overflow, Frame Error and Parity Error bits in the Status register (STATUS.COLL, STATUS.ISF, STATUS.BUFOVF, STATUS.FERR and STATUS.PERR).

29.5.6. Events

Not applicable.

29.5.7. Analog Connections

Not applicable.

29.6. Register Summary - SERCOMn.USART

SERCOM in USART Mode (External/Internal clock)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
		15:8	SAMPR[2:0]							IBON
		23:16	SAMPA[1:0]		RXPO[1:0]				TXPO[1:0]	
		31:24		DORD	CPOL	CMODE	FORM[3:0]			
0x04	CTRLB	7:0		SBMODE				CHSIZE[2:0]		
		15:8			PMODE			ENC	SFDE	COLDEN
		23:16							RXEN	TXEN
		31:24							LINCMD[1:0]	
0x08	CTRLC	7:0						GTIME[2:0]		
		15:8					HDRDLY[1:0]		BRKLEN[1:0]	
		23:16								
		31:24								
0x0C	BAUD	7:0	BAUD[7:0]							
		15:8	BAUD[15:8]							
0x0C	BAUD	7:0	BAUD[7:0]							
		15:8	FP[2:0]			BAUD[12:8]				
0x0E	RXPL	7:0	RXPL[7:0]							
0x0F ... 0x13	Reserved									
0x14	INTENCLR	7:0	ERROR		RXBRK	CTSIC	RXS	RXC	TXC	DRE
0x15	Reserved									
0x16	INTENSET	7:0	ERROR		RXBRK	CTSIC	RXS	RXC	TXC	DRE
0x17	Reserved									
0x18	INTFLAG	7:0	ERROR		RXBRK	CTSIC	RXS	RXC	TXC	DRE
0x19	Reserved									
0x1A	STATUS	7:0		TXE	COLL	ISF	CTS	BUFOVF	FERR	PERR
		15:8								
0x1C	SYNCBUSY	7:0						CTRLB	ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x20 ... 0x27	Reserved									
0x28	DATA	7:0	DATA[7:0]							
		15:8								DATA[8]
0x2A ... 0x2F	Reserved									
0x30	DBGCTRL	7:0								DBGSTOP

29.6.1. Control A - USART

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		DORD	CPOL	CMODE	FORM[3:0]			
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SAMPA[1:0]		RXPO[1:0]				TXPO[1:0]	
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bit	15	14	13	12	11	10	9	8
	SAMPR[2:0]							IBON
Access	R/W	R/W	R/W					R/W
Reset	0	0	0					0

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit 30 – DORD Data Order
This bit selects the data order when a character is shifted out from the Data register.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	MSB	MSb is transferred first
1	LSB	LSb is transferred first

Bit 29 – CPOL Clock Polarity
This bit selects the relationship between data output change and data input sampling in Synchronous mode.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	IDLE_LOW	TxD Change: Rising XCK edge, RxD Sample: Falling XCK edge
1	IDLE_HIGH	TxD Change: Falling XCK edge, RxD Sample: Rising XCK edge

Bit 28 – CMODE Communication Mode
This bit selects asynchronous or synchronous communication.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	ASYNC	Asynchronous communication
1	SYNC	Synchronous communication

Bits 27:24 – FORM[3:0] Frame Format
These bits define the frame format.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	USART_FRAME_NO_PARITY	USART frame
0x1	USART_FRAME_WITH_PARITY	USART frame with parity
0x2	USART_FRAME_LINBRKGEN	LIN Host - Break and sync generation. See LIN Command (CTRLB.LINCMD).
0x4	USART_FRAME_AUTO_BAUD_NO_PARITY	Auto-baud (LIN Client) - break detection and auto-baud
0x5	USART_FRAME_AUTO_BAUD_WITH_PARITY	Auto-baud - break detection and auto-baud with parity

Bits 23:22 – SAMPA[1:0] Sample Adjustment
These bits define the sample adjustment.
Notes:
1. This bit field is enable-protected. This bit field is not synchronized.
2. This adjustment depends on the over-sampling setting 16x (CTRLA.SAMPR = 0 or 1) or 8x (CTRLA.SAMPR = 2 or 3)

Value	Name	Description
0x0	ADJ0	16x Over-sampling = 7-8-9; 8x Over-sampling = 3-4-5
0x1	ADJ1	16x Over-sampling = 9-10-11; 8x Over-sampling = 4-5-6
0x2	ADJ2	16x Over-sampling = 11-12-13; 8x Over-sampling = 5-6-7
0x3	ADJ3	16x Over-sampling = 13-14-15; 8x Over-sampling = 6-7-8

Bits 21:20 – RXPO[1:0] Receive Data Pinout
These bits define the receive data (RxD) pin configuration.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	MUX0	SERCOM PAD[0] is used for data reception
0x1	MUX1	SERCOM PAD[1] is used for data reception
0x2	MUX2	SERCOM PAD[2] is used for data reception
0x3	MUX3	SERCOM PAD[3] is used for data reception

Bits 17:16 – TXPO[1:0] Transmit Data Pinout
These bits define the transmit data (TxD) and XCK pin configurations.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	MUX0	PAD[0]=TxD, PAD[1]=XCK
0x1	MUX1	PAD[2]=TxD, PAD[3]=XCK
0x2	MUX2	PAD[0]=TxD, PAD[2]=RTS/TE, PAD[3]=CTS
0x3	MUX3	PAD[0]=TxD, PAD[1]=XCK, PAD[2]=RTS/TE

Bits 15:13 – SAMPR[2:0] Sample Rate
These bits select the sample rate.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	16X_ARITHMETIC	16x over-sampling using arithmetic baud rate generation
0x1	16X_FRACTIONAL	16x over-sampling using fractional baud rate generation
0x2	8X_ARITHMETIC	8x over-sampling using arithmetic baud rate generation
0x3	8X_FRACTIONAL	8x over-sampling using fractional baud rate generation
0x4	3X_ARITHMETIC	3x over-sampling using arithmetic baud rate generation

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.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	STATUS.BUFOVF is asserted when a buffer overflow 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. Refer to Sleep Mode Operation for more information.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	The USART does not operate in Standby sleep mode
1	The USART operates in Standby sleep mode

Bits 4:2 – MODE[2:0] Operating Mode

This bit field controls the SERCOM mode.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	USART_EXT	USART with external clock
0x1	USART_INT	USART with internal clock
0x2	SPI_SLAVE	SPI Client mode
0x3	SPI_MASTER	SPI Host mode
0x4	I2C_SLAVE	I^2C Client mode
0x5	I2C_MASTER	I^2C Host mode

Bit 1 – ENABLE Enable

Notes:

1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure the CTRLA.ENABLE synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled or being disabled
1	The peripheral is enabled or being enabled

Bit 0 – SWRST Software Reset

Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Notes:

1. This bit is write-synchronized: SYNCBUSY.SWRST must be checked to ensure the CTRLA.SWRST synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	There is no reset operation in progress
1	The reset operation is in progress

29.6.2. Control B - USART

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
							LINCMD[1:0]	
Access							R/W	R/W
Reset							0	0

Bit	23	22	21	20	19	18	17	16
							RXEN	TXEN
Access							R/W	R/W
Reset							0	0

Bit	15	14	13	12	11	10	9	8
			PMODE			ENC	SFDE	COLDEN
Access			R/W			R/W	R/W	R/W
Reset			0			0	0	0

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

Bits 25:24 – LINCMD[1:0] LIN Command
These bits define the LIN header transmission control. This field is only valid in LIN Host mode (CTRLA.FORM = `USART_FRAME_LINBRKGEN`).
These bits will always read back as zero.
Notes:

1. This bit field is write-synchronized: SYNCBUSY.CTRLB must be checked to ensure the CTRLB.LINCMD synchronization is complete.
2. This bit field is enable-protected.

Value	Name	Description
0x0	NONE	Normal USART transmission
0x1	SOFTWARE_CONTROL_TRANSMIT_CMD	A break field is transmitted when DATA is written
0x2	AUTO_TRANSMIT_CMD	The break, sync, and identifier fields are automatically transmitted when the DATA is written with the identifier

Bit 17 – RXEN Receiver Enable
Writing '0' 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 '1' to CTRLB.RXEN when the USART is disabled will set CTRLB.RXEN immediately.
Writing '1' to CTRLB.RXEN when the USART is enabled will set SYNCBUSY.CTRLB, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as '1'.
Notes:

1. This bit is write-synchronized: SYNCBUSY.CTRLB must be checked to ensure the CTRLB.RXEN synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	The receiver is disabled or being enabled
1	The receiver is enabled or will be enabled when the USART is enabled

Bit 16 – TXEN Transmitter Enable

Writing '0' to this bit will disable the USART transmitter. Disabling the transmitter will not become effective until ongoing and pending transmissions are completed.
Writing '1' to CTRLB.TXEN when the USART is disabled will set CTRLB.TXEN immediately.
Writing '1' to CTRLB.TXEN when the USART is enabled will set SYNCBUSY.CTRLB, which will remain set until the transmitter is enabled, and CTRLB.TXEN will read back as '1'.

Notes:

1. This bit is write-synchronized: SYNCBUSY.CTRLB must be checked to ensure the CTRLB.TXEN synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	The transmitter is disabled or being enabled
1	The transmitter is enabled or will be enabled when the USART is enabled

Bit 13 – PMODE Parity Mode

This bit selects the type of parity used when parity is enabled. 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 bit of the incoming frame and, if a mismatch is detected, STATUS.PERR will be set.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	EVEN	Even parity
1	ODD	Odd parity

Bit 10 – ENC Encoding Format

This bit selects the data encoding format.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	DISABLE	Data is not encoded
1	IRDA	Data is IrDA encoded

Bit 9 – SFDE Start of Frame Detection Enable

This bit controls whether the start-of-frame detector will wake up the device when a start bit is detected on the RxD line.

Note: This bit is enable-protected. This bit is not synchronized.

SFDE	INTENSET.RXS	INTENSET.RXC	Description
0	X	X	Start-of-frame detection disabled
1	0	0	Reserved
1	0	1	Start-of-frame detection enabled. RXC wakes up the device from all sleep modes.
1	1	0	Start-of-frame detection enabled. RXS wakes up the device from all sleep modes.
1	1	1	Start-of-frame detection enabled. Both RXC and RXS wake up the device from all sleep modes.

Bit 8 – COLDEN Collision Detection Enable

This bit enables collision detection.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Collision detection is not enabled
1	Collision detection is enabled

Bit 6 – SBMODE Stop Bit Mode

This bit selects the number of stop bits transmitted.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	1_BIT	One stop bit
1	2_BIT	Two stop bits

Bits 2:0 – CHSIZE[2:0] Character Size

These bits select the number of bits in a character.

Note: This bit field is enable-protected. This bit is not synchronized.

Value	Name	Description
0x0	8_BIT	8-bits character
0x1	9_BIT	9-bits character
0x5	5_BIT	5-bits character
0x6	6_BIT	6-bits character
0x7	7_BIT	7-bits character
Other	—	Reserved

29.6.3. Control C - USART

Name: CTRLC
Offset: 0x08
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					HDRDLY[1:0]		BRKLEN[1:0]	
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

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

Bits 11:10 – HDRDLY[1:0] LIN Host Header Delay
These bits define the delay between break and sync transmission in addition to the delay between the sync and identifier (ID) fields when in LIN Host mode (CTRLA.FORM = `USART_FRAME_LINBRKGEN`).
This field is only valid when using the LIN header command (CTRLB.LINCMD = `AUTO_TRANSMIT_CMD`).

Value	Name	Description
0x0	1_BIT_TIME	Delay between break and sync transmission is one bit time. Delay between sync and ID transmission is one bit time.
0x1	4_BIT_TIME	Delay between break and sync transmission is four bit times. Delay between sync and ID transmission is four bit times.
0x2	8_BIT_TIME	Delay between break and sync transmission is eight bit times. Delay between sync and ID transmission is four bit times.
0x3	14_BIT_TIME	Delay between break and sync transmission is 14 bit times. Delay between sync and ID transmission is four bit times.

Bits 9:8 – BRKLEN[1:0] LIN Host Break Length
These bits define the length of the break field transmitted when in LIN Host mode (CTRLA.FORM = `USART_FRAME_LINBRKGEN`).

Value	Name	Description
0x0	13_BIT_TIME	The break field transmission is 13 bit times
0x1	17_BIT_TIME	The break field transmission is 17 bit times
0x2	21_BIT_TIME	The break field transmission is 21 bit times
0x3	26_BIT_TIME	The break field transmission is 26 bit times

Bits 2:0 – GTIME[2:0] Guard Time

These bits define the guard time when using RS485 mode.
For RS485 mode, the guard time is programmable from 0–7 bit times and defines the time that the transmit enable pin (TE) remains high after the last stop bit is transmitted and there is no remaining data to be transmitted.

29.6.4. Baud - Arithmetic Mode - USART

Name: BAUD
Offset: 0x0C
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

This register description is valid only in Arithmetic Baud Rate Generation mode (`CTRLA.SAMPR[0]=0`).

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[15:0] Baud Value
These bits control the clock generation, as described in the *Clock Generation – Baud-Rate Generator* section.

29.6.5. Baud - Fractional Mode - USART

Name: BAUD
Offset: 0x0C
Reset: 0x0000
Property: PAC Write-Protection, Enable-Protected

This register description is valid only in Fractional Baud Rate Generation mode (`CTRLA.SAMPR[0]=1`).

Bit	15	14	13	12	11	10	9	8
	FP[2:0]			BAUD[12:8]				
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 15:13 – FP[2:0] Fractional Part
These bits control the clock generation, as described in the *Clock Generation – Baud-Rate Generator* section.

Bits 12:0 – BAUD[12:0] Baud Value
These bits control the clock generation, as described in the *Clock Generation – Baud-Rate Generator* section.

29.6.6. Receive Pulse Length - USART

Name: RXPL
Offset: 0x0E
Reset: 0x00
Property: PAC Write-Protection, Enable-Protected

Bit	7	6	5	4	3	2	1	0
	RXPL[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 – RXPL[7:0] Receive Pulse Length
When the encoding format is set to IrDA (CTRLB.ENC = 1), these bits control the minimum pulse length required for a pulse to be accepted by the IrDA receiver with regards to the serial engine clock period SE_{per}.

$$PULSE \geq \left(\mathrm{RXPL} + 1\right) \cdot SE_{per}$$

29.6.7. Interrupt Enable Clear - USART

Name: INTENCLR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection

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).

Bit	7	6	5	4	3	2	1	0
	ERROR		RXBRK	CTSIC	RXS	RXC	TXC	DRE
Access	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	0		0	0	0	0	0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 5 – RXBRK Receive Break Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Receive Break Interrupt Enable bit, which disables the Receive Break interrupt.

Value	Description
0	Receive Break interrupt is disabled
1	Receive Break interrupt is enabled

Bit 4 – CTSIC Clear to Send Input Change Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Clear To Send Input Change Interrupt Enable bit, which disables the Clear To Send Input Change interrupt.

Value	Description
0	Clear To Send Input Change interrupt is disabled
1	Clear To Send Input Change interrupt is enabled

Bit 3 – RXS Receive Start Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Receive Start Interrupt Enable bit, which disables the Receive Start interrupt.

Value	Description
0	Receive Start interrupt is disabled
1	Receive Start interrupt is enabled

Bit 2 – RXC Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

Value	Description
0	Receive Complete interrupt is disabled
1	Receive Complete interrupt is enabled

Bit 1 – TXC Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Transmit Complete Interrupt Enable bit, which disables the Transmit Complete interrupt.

Value	Description
0	Transmit Complete interrupt is disabled
1	Transmit Complete interrupt is enabled

Bit 0 – DRE Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

Value	Description
0	Data Register Empty interrupt is disabled
1	Data Register Empty interrupt is enabled

29.6.8. Interrupt Enable Set - USART

Name: INTENSET
Offset: 0x16
Reset: 0x00
Property: PAC Write-Protection

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).

Bit	7	6	5	4	3	2	1	0
	ERROR		RXBRK	CTSIC	RXS	RXC	TXC	DRE
Access	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	0		0	0	0	0	0	0

Bit 7 – ERROR Error Interrupt Enable
The Frame Error (FERR) and Parity Error (PERR) error interrupts will not wake the device from Standby mode.
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 5 – RXBRK Receive Break Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Receive Break Interrupt Enable bit, which enables the Receive Break interrupt.

Value	Description
0	Receive Break interrupt is disabled
1	Receive Break interrupt is enabled

Bit 4 – CTSIC Clear to Send Input Change Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Clear To Send Input Change Interrupt Enable bit, which enables the Clear To Send Input Change interrupt.

Value	Description
0	Clear To Send Input Change interrupt is disabled
1	Clear To Send Input Change interrupt is enabled

Bit 3 – RXS Receive Start Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Receive Start Interrupt Enable bit, which enables the Receive Start interrupt.

Value	Description
0	Receive Start interrupt is disabled
1	Receive Start interrupt is enabled

Bit 2 – RXC Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

Value	Description
0	Receive Complete interrupt is disabled
1	Receive Complete interrupt is enabled

Bit 1 – TXC Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

Value	Description
0	Transmit Complete interrupt is disabled
1	Transmit Complete interrupt is enabled

Bit 0 – DRE Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

Value	Description
0	Data Register Empty interrupt is disabled
1	Data Register Empty interrupt is enabled

29.6.9. Interrupt Flag Status and Clear - USART

Name: INTFLAG
Offset: 0x18
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
	ERROR		RXBRK	CTSIC	RXS	RXC	TXC	DRE
Access	R/W		R/W	R/W	R/W	R	R/W	R
Reset	0		0	0	0	0	0	0

Bit 7 – ERROR Error
This flag is cleared by writing '1' to it.
This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. Errors that will set this flag are COLL, ISF, BUFOVF, FERR, and PERR.
The Frame Error (FERR) and Parity Error (PERR) error interrupts will not wake the device from Standby mode.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

Bit 5 – RXBRK Receive Break
This flag is cleared by writing '1' to it.
This flag is set when auto-baud is enabled (CTRLA.FORM) and a break character is received.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

Bit 4 – CTSIC Clear to Send Input Change
This flag is cleared by writing a '1' to it.
This flag is set when a change is detected on the CTS pin.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

Bit 3 – RXS Receive Start
This flag is cleared by writing '1' to it.
This flag is set when a start condition is detected on the RxD line and start-of-frame detection is enabled (CTRLB.SFDE is '1').
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Receive Start interrupt flag.

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 '0' to this bit has no effect.
Writing '1' to this bit has no effect.

Bit 1 – TXC Transmit Complete
This flag is cleared by writing '1' 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 '0' to this bit has no effect.
Writing '1' 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 '0' to this bit has no effect.
Writing '1' to this bit has no effect.

29.6.10. Status - USART

Name: STATUS
Offset: 0x1A
Reset: 0x0000
Property: –

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		TXE	COLL	ISF	CTS	BUFOVF	FERR	PERR
Access		R	R/W	R/W	R	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 6 – TXE Transmitter Empty
When CTRLA.FORM is set to LIN Host mode, this bit is set when any ongoing transmission is complete and TxDATA is empty.
When CTRLA.FORM is not set to LIN Host mode, this bit will always read back as zero.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear it.

Bit 5 – COLL Collision Detected
This bit is cleared by writing '1' to the bit or by disabling the receiver.
This bit is set when collision detection is enabled (CTRLB.COLDEN) and a collision is detected.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear it.

Bit 4 – ISF Inconsistent Sync Field
This bit is cleared by writing '1' to the bit or by disabling the receiver.
This bit is set when the frame format is set to auto-baud (CTRLA.FORM) and a sync field not equal to 0x55 is received.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear it.

Bit 3 – CTS Clear to Send
This bit indicates the current level of the CTS pin when flow control is enabled (CTRLA.TXPO).
Writing '0' to this bit has no effect.
Writing '1' to this bit has no effect.

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 '1' 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 '0' to this bit has no effect.
Writing '1' 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 '1' 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 '0'.
Writing '0' to this bit has no effect.
Writing '1' 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 '1' to the bit or by disabling the receiver.
This bit is set if parity checking is enabled and a parity error is detected.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear it.

29.6.11. Synchronization Busy - USART

Name: SYNCBUSY
Offset: 0x1C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						CTRLB	ENABLE	SWRST
Access						R	R	R
Reset						0	0	0

Bit 2 – CTRLB CTRLB Synchronization Busy
Writing to some Control B (CTRLB) register bits when the SERCOM is enabled requires synchronization. When writing to them, the SYNCBUSY.CTRLB bit will be set until synchronization is complete. If they are written while SYNCBUSY.CTRLB is asserted, an APB error will be generated.

Value	Description
0	CTRLB synchronization is not busy
1	CTRLB synchronization is busy

Bit 1 – ENABLE SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

Value	Description
0	Enable synchronization is not busy
1	Enable synchronization is busy

Bit 0 – SWRST Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

Value	Description
0	SWRST synchronization is not busy
1	SWRST synchronization is busy

29.6.12. Data - USART

Name: DATA
Offset: 0x28
Reset: 0x0000
Property: –

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

Reading these bits will return the contents of the Receive Data register. The register must 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 must be read before reading the DATA value to get any corresponding error.

Writing these bits will write the Transmit Data register. This register must be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

29.6.13. Debug Control - USART

Name: DBGCTRL
Offset: 0x30
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGSTOP Debug Stop Mode
This bit controls the baud rate generator functionality when the CPU is halted by an external debugger.

Value	Description
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

30. SERCOM SPI — SERCOM Serial Peripheral Interface

30.1. Features

- Full-Duplex, Four-Wire Interface
- One-Level Transmit Buffer, Two-Level Receive Buffer
- Supports all Four SPI Modes of Operation
- Single Data Direction Operation Allows Alternate Function on the MISO or MOSI Pin
- Selectable LSb- or MSb-First Data Transfer
- Can be used with DMA
- Host Operation:
  - 8-bit Clock Generator
  - Hardware-controlled $\overline{SS}$
- Client Operation:
  - 8-bit Address Match Operation
  - Operation in all Sleep Modes
  - Wake on $\overline{SS}$ Transition

30.2. Overview

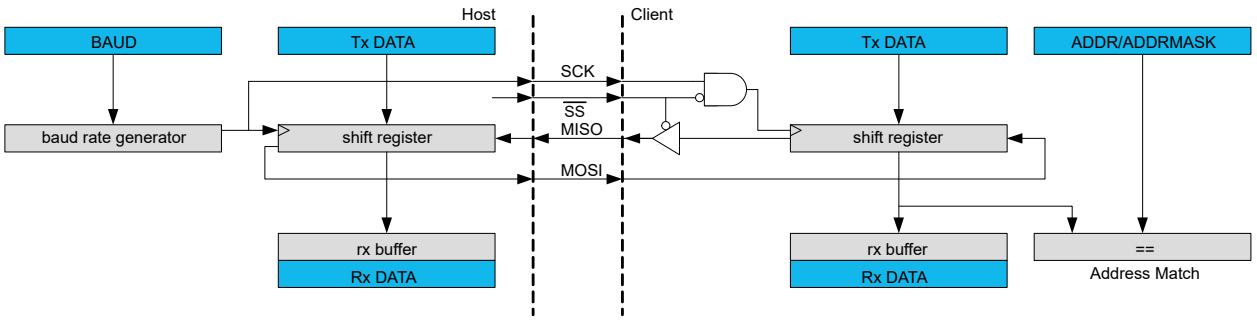
The serial peripheral interface (SPI) is one of the available modes in the Serial Communication Interface (SERCOM).

A SERCOM instance can be configured to operate either as an SPI host or an SPI client, as shown in the Block Diagram. Both the host and the client have an interface containing a shift register, a transmit buffer, and a receive buffer.

Additionally, the SPI host uses the SERCOM baud rate generator, while the SPI client uses the SERCOM address match logic.

30.3. Block Diagram

Figure 30-1. Full-Duplex SPI Host Client Interconnection



Registers labeled in uppercase are synchronous to CLK_SERCOMx_APB and accessible from the CPU, while those with lowercase labels are synchronous to the SCL clock.

30.3.1. Signal Description

Table 30-1. SERCOM SPI Signals

Signal Name	Type	Description
PADn	Digital I/O	SERCOM pins

One signal can be mapped to one of several pins. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter for details.

30.4. Functional Description

30.4.1. Initialization

1. Select the SPI mode for host or client operation using the Operating Mode bit field in the Control A register (CTRLA.MODE).
2. Select the transfer mode by configuring the Clock Polarity and Clock Phase bits in the Control A register (CTRLA.CPOL and CTRLA.CPHA).
3. Configure the Frame Format bit field in the Control A register (CTRLA.FORM).
4. Configure the Data In Pinout bit field in the Control A register (CTRLA.DIPO) to select SERCOM pads for the receiver.
5. Configure the Data Out Pinout bit field in the Control A register (CTRLA.DOPO) to select SERCOM pads for the transmitter.
6. If the SPI is used in Host mode:
   a. Select the desired baud rate by writing to the Baud register (BAUD).
   b. If Hardware $\overline{SS}$ control is required, write '1' to the Host SPI Select Enable bit in the CTRLB register (CTRLB.MSSEN).
7. Enable the receiver by writing a '1' to the Receiver Enable bit in the Control B register (CTRLB.RXEN).
8. Enable the module by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE)

30.4.2. Operation

The SPI is a synchronous data transfer interface. It allows high-speed communication between the device and peripheral devices.

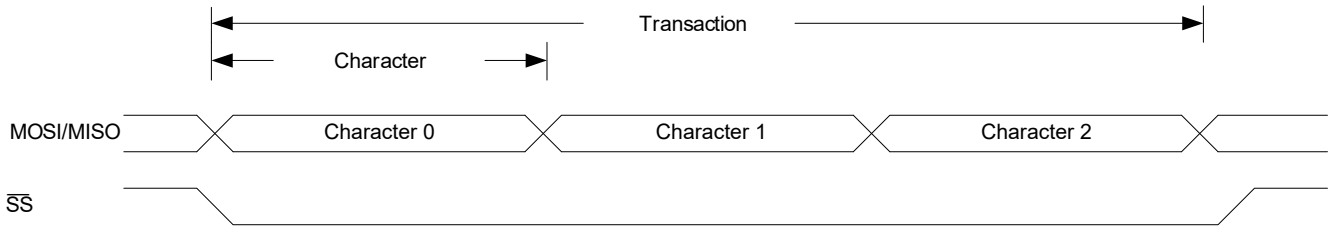
The SPI can operate as either a host or a client. As a host, the SPI initiates and controls all data transactions. The SPI is single-buffered for transmitting and double-buffered for receiving.

When transmitting data, the DATA register can be loaded with the next character to be transmitted during the current transmission.

When receiving, the data is transferred to the two-level receive buffer, and the receiver is ready for a new character.

The SPI transaction format is shown in Figure 30-2. Each transaction can contain one or more characters. The character size is configurable and can be either eight or nine bits.

Figure 30-2. SPI Transaction Format



The SPI host must pull the SPI select line ($\overline{SS}$) of the desired client low to initiate a transaction if multiple clients are connected to the bus. The SPI select line can be wired low if there is only one SPI client on the bus. The host and client prepare data to send via their respective shift registers, and the host generates the serial clock on the SCK line.

Data is always shifted from host to client on the Host Output Client Input line (MOSI); data is shifted from client to host on the Host Input Client Output line (MISO).

Each time a character is shifted out from the host, a character will be shifted out from the client simultaneously. To signal the end of a transaction, the host will pull the $\overline{SS}$ line high.

30.4.2.1. Enabling, Disabling, and Resetting

This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing '1' to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register.

30.4.2.2. Clock Generation

In SPI host operation, the serial clock (SCK) is generated internally by the SERCOM baud rate generator. The 8-bit Baud register (BAUD) value is used for generating SCK and to clock the shift register. Refer to Clock Generation – Baud-Rate Generator for more details.

In SPI client operation, the clock is provided by an external host on the SCK pin. This clock is used to clock the SPI shift register directly.

30.4.2.3. Data Register

The SPI Transmit Data register (TxDATA) and the SPI Receive Data register (RxDATA) share the same I/O address, referred to as the SPI Data register (DATA). Writing to the DATA register updates the Transmit Data register, while reading from the DATA register returns the contents of the Receive Data register.

30.4.2.4. SPI Transfer Modes

There are four combinations of SCK phase and polarity for transferring serial data. The SPI data transfer modes are shown in Table 30-2 and Figure 30-3.

SCK phase is configured by the Clock Phase bit in the CTRLA register (CTRLA.CPHA). SCK polarity is programmed by the Clock Polarity bit in the CTRLA register (CTRLA.CPOL). Data bits are shifted out and latched in on opposite edges of the SCK signal. This ensures sufficient time for the data signals to stabilize.

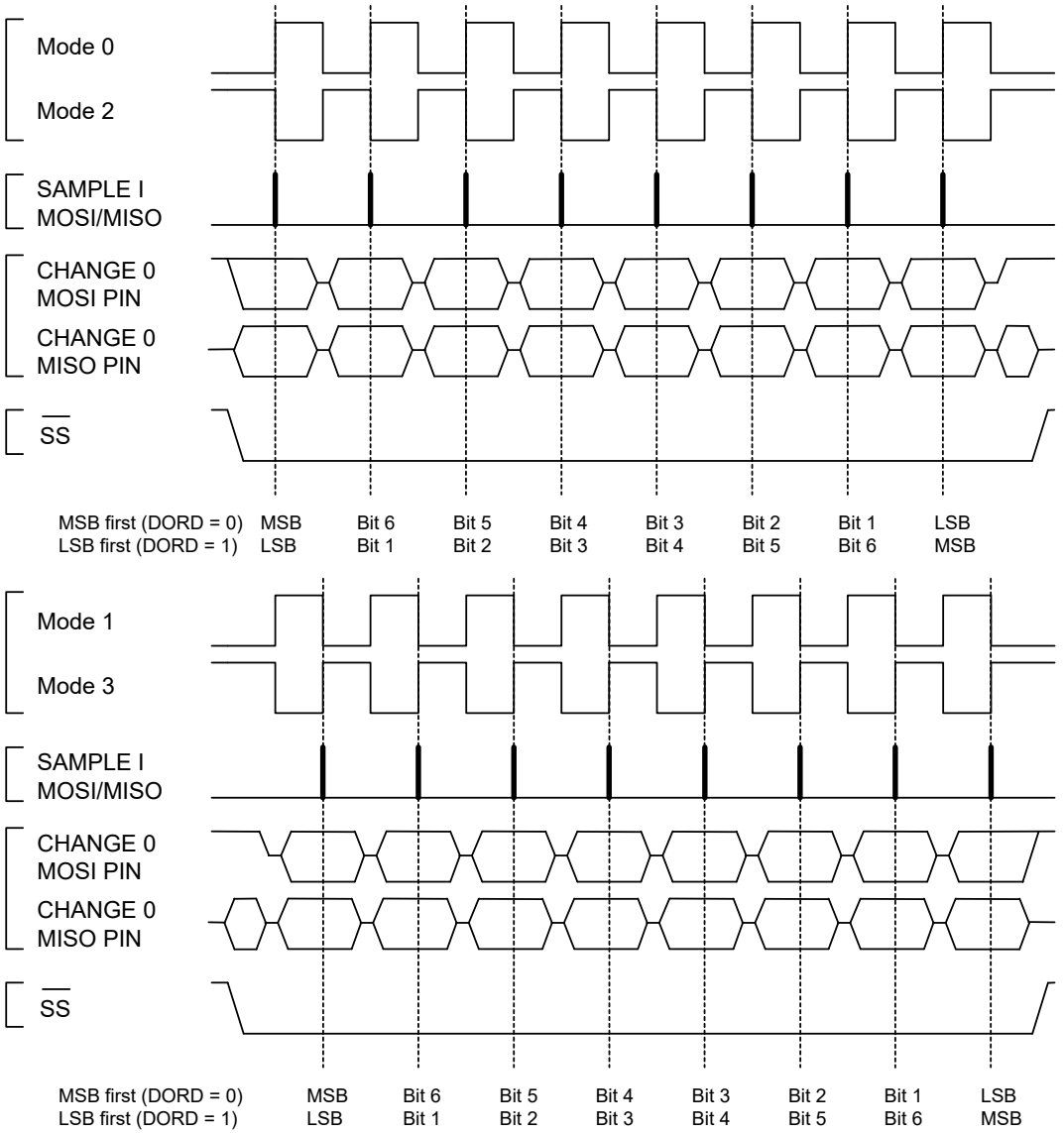
Table 30-2. SPI Transfer Modes

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

Note:
Leading edge is the first clock edge in a clock cycle.

Trailing edge is the second clock edge in a clock cycle.

Figure 30-3. SPI Transfer Modes



30.4.2.5. Transferring Data

30.4.2.5.1. Host

In Host mode, when Host SPI Select Enable (CTRLB.MSSEN) is '1', hardware controls the $\overline{SS}$ line.

When Host SPI Select Enable (CTRLB.MSSEN) is '0', the $\overline{SS}$ line must be configured as an output. Any general purpose I/O pin can be used as $\overline{SS}$. When the SPI is ready for a data transaction, software must pull the $\overline{SS}$ line low.

When writing a character to the Data register (DATA), the character will be transferred to the shift register. Once the content of TxDATA has been transferred to the shift register, the Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) will be set, and a new character can be written to DATA.

Each time a character is shifted out from the host, another character is simultaneously shifted in from the client. If the receiver is enabled (CTRLA.RXEN=1), the contents of the shift register are transferred to the two-level receive buffer. The transfer occurs in the same clock cycle as the last data bit is shifted in. Then, the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set. The received data can be retrieved by reading DATA register.

When the last character has been transmitted and there is no valid data in the DATA register, the Transmit Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set. When the transaction is finished, the host must pull the $\overline{SS}$ line high to notify the client. If Host SPI Select Enable (CTRLB.MSSEN) is set to '0', the software must pull the $\overline{SS}$ line high.

30.4.2.5.2. Client

In Client mode, the SPI interface will remain inactive with the MISO line tri-stated as long as the $\overline{SS}$ pin is held high. Software may update the contents of the DATA register at any time, as long as the Data Register Empty flag in the Interrupt Status and Clear register (INTFLAG.DRE) is set.

When $\overline{SS}$ is pulled low and SCK is running, the client will sample and shift out data according to the configured transaction mode. Once the contents of TxDATA have been loaded into the shift register, INTFLAG.DRE will be set, and new data can be written to the DATA register.

Similar to the host, the client will receive one character for each character transmitted. A character is transferred into the two-level receive buffer in the same clock cycle that its last data bit is received. The received character can be retrieved from the DATA register when the Receive Complete interrupt flag (INTFLAG.RXC) is set.

When the host pulls the $\overline{SS}$ line high, the transaction is done and the Transmit Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set.

After DATA is written, it takes up to three SCK clock cycles for the content of the DATA register is ready to load into the shift register at the next character boundary. As a result, the first character transferred in an SPI transaction will not be the content of the DATA register. This can be avoided by using the preloading feature. Refer to Preloading of the Client Shift Register for more information.

When transmitting several characters in one SPI transaction, data must be written to the DATA register with at least three SCK clock cycles remaining in the current character transmission. If this criteria is not met, the previously received character will be transmitted.

Once the DATA register is empty, it takes three CLK_SERCOM_APB cycles for INTFLAG.DRE to be set.

30.4.2.6. Receiver Error Bit

The SPI receiver has one error bit: The Buffer Overflow bit (BUFOVF), which can be read from the Status register (STATUS). Once an error occurs, the bit remains set until it is cleared by writing '1' to it. The bit is also automatically cleared when the receiver is disabled.

There are two methods for buffer overflow notification, selected by the immediate buffer overflow notification bit in the Control A register (CTRLA.IBON):

If CTRLA.IBON is set to '1', STATUS.BUFOVF is raised immediately upon buffer overflow. Software can then empty the receive FIFO by reading RxDATA until the Receive Complete Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) goes low.

If CTRLA.IBON is '0', the buffer overflow condition travels with data through the receive FIFO. After the received data is read, STATUS.BUFOVF and INTFLAG.ERROR will be set along with INTFLAG.RXC, and RxDATA will be zero.

30.4.3. Additional Features

30.4.3.1. Address Recognition

When the SPI is configured for client operation with address recognition (CTRLA.FORM configured to SPI_FRAME_WITH_ADDR), the SERCOM address recognition logic is enabled: the first character in a transaction is checked for an address match.

If there is a match, 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 the device is in Sleep mode, an address match can wake up the device to process the transaction.

If there is no match, the complete transaction is ignored.

If a 9-bit frame format is selected, only the lower eight bits of the shift register are checked against the Address register (ADDR).

Preload must be disabled (CTRLB.PLOADEN set to '0') to use this mode.

30.4.3.2. Preloading of the Client Shift Register

When starting a transaction, the client 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 received character in the previous transaction.

Preloading can be used to preload data into the shift register while $\overline{SS}$ is high: this eliminates sending a dummy character when starting a transaction.

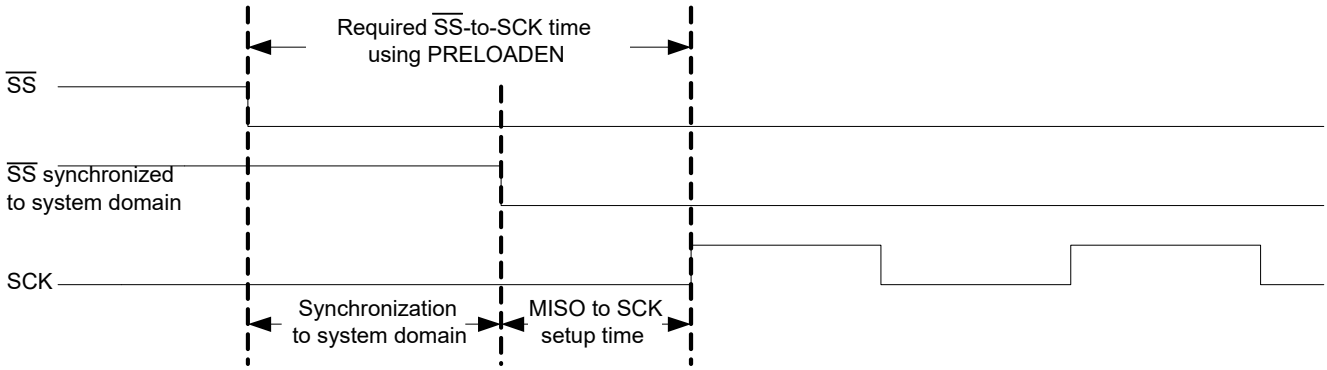
Only one data character can be preloaded into the shift register while the synchronized $\overline{SS}$ signal is high. If the next character is written to DATA before $\overline{SS}$ is pulled low, the second character will be stored in DATA until transfer begins.

For proper preloading, sufficient time must elapse between $\overline{SS}$ going low and the first SCK sampling edge, as shown in Figure 30-4. See also the *Electrical Characteristics* section for timing details.

Preloading is enabled by writing '1' to the Client Data Preload Enable bit in the CTRLB register (CTRLB.PLOADEN).

The Preload function must not be used when the Host cannot maintain the $\overline{SS}$ in a low state until transmission is complete.

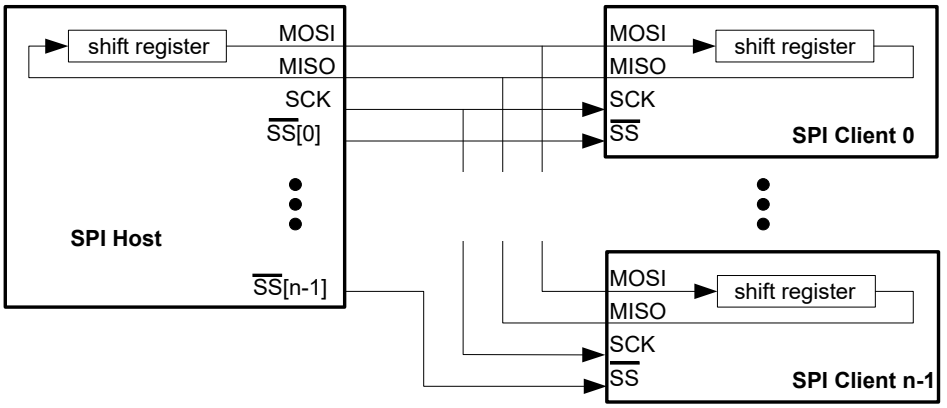
Figure 30-4. Timing Using Preloading



30.4.3.3. Host with Several Clients

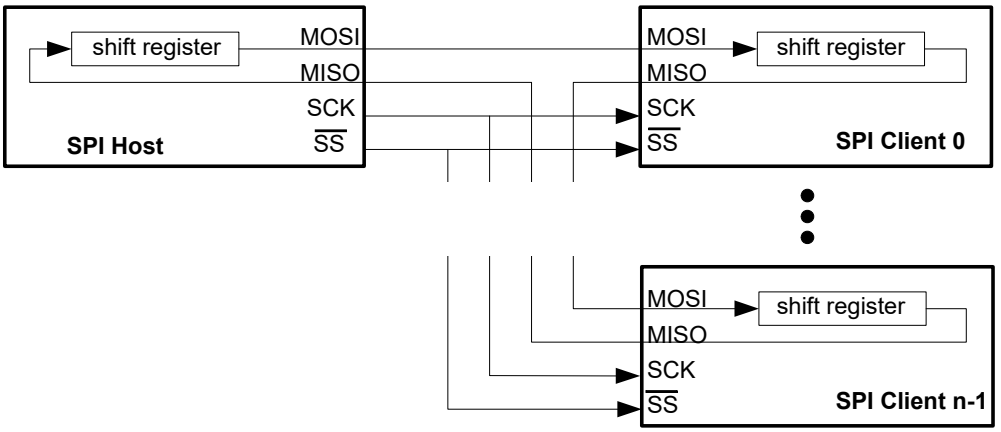
A host with multiple clients in parallel is only available when Host SPI Select Enable (CTRLB.MSSEN) is set to '0', disabling hardware $\overline{SS}$ control. If the bus consists of several SPI clients, the SPI host can use general purpose I/O pins to control the $\overline{SS}$ line for each client on the bus, as shown in the following figure. In this configuration, only the selected SPI client will drive the MISO line.

Figure 30-5. Multiple Clients in Parallel



Another configuration is multiple clients in series, as shown in the following figure. In this setup, all n attached clients are connected to the same $\overline{SS}$ line. Depending on the Host SPI Select Enable bit (CTRLB.MSSEN), the $\overline{SS}$ line can be controlled either by hardware or by user software using normal GPIO. The host must shift n characters for a complete transaction.

Figure 30-6. Multiple Clients in Series



30.4.3.4. Loop-Back Mode

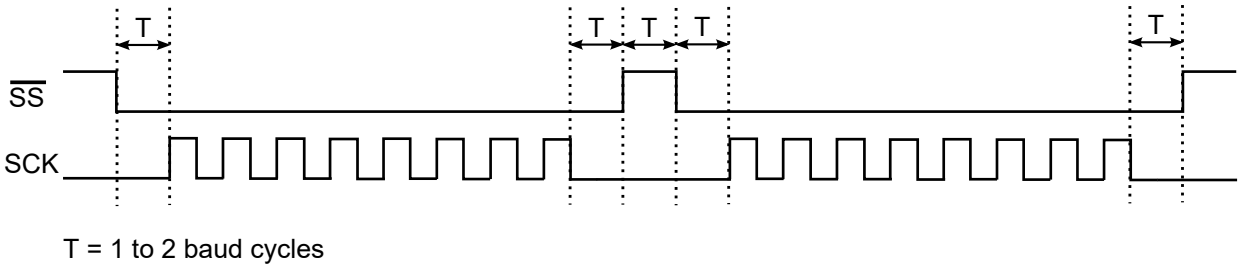
For Loop-Back mode, configure the Data In Pinout and Data Out Pinout in the Control A register (CTRLA.DIPO and CTRLA.DOPO) to use the same data pins for both transmit and receive. The loop-back occurs through the pad, so the signal is also available externally.

30.4.3.5. Hardware Controlled $\overline{SS}$

In Host mode, an SPI Select ($\overline{SS}$) line can be controlled by hardware by writing the Host SPI Select Enable (CTRLB.MSSEN) bit to '1'. In this mode, the $\overline{SS}$ pin is driven low for at least one baud cycle before transmission begins, and stays low for at least one baud cycle after transmission completes. The $\overline{SS}$ pin will always be driven high for at least one baud cycle between each data sent.

In Figure 30-7, the time T ranges from one to two baud cycles depending on the SPI transfer mode.

Figure 30-7. Hardware Controlled $\overline{SS}$



When CTRLB.MSSEN is '0', the $\overline{SS}$ pin(s) is/are controlled by user software and normal GPIO.

30.4.3.6. SPI Select Low Detection

In Client mode, the SPI can wake the CPU when the SPI Select ($\overline{SS}$) goes low. When SPI Select Low Detect is enabled in the Control B register (CTRLB.SSDE), a high-to-low transition sets the SPI Select Low interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.SSL), and the device will wake up if applicable.

30.4.4. Sleep Mode Operation

The behavior in sleep mode depends on the host/client configuration and the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY):

- Host operation with CTRLA.RUNSTDBY = 1: The peripheral clock GCLK_SERCOM_CORE will continue to run in Idle sleep mode and in Standby sleep mode. Any interrupt can wake up the device.
- Host operation with CTRLA.RUNSTDBY = 0: GLK_SERCOMx_CORE will be disabled after the ongoing transaction is finished. Any interrupt can wake up the device.
- Client operation with CTRLA.RUNSTDBY = 1: The Receive Complete interrupt can wake up the device.
- Client operation with CTRLA.RUNSTDBY = 0: All reception will be dropped, including any ongoing transaction.

30.4.5. Debug Operation

When the CPU is halted in debug mode the SERCOM will continue normal operation. Setting the Debug Stop Mode bit in the Debug Control register (DBGCTRL.DBGSTOP) forces the SERCOM to halt operation when the CPU is halted in debug mode.

30.5. Dependencies

30.5.1. I/O Lines

Using the SERCOM's I/O lines requires the I/O pins to be configured.

The SERCOM has four internal pads, PAD[3:0], and the signals from I^2C, SPI and USART are routed through these pads via a multiplexer. The configuration of the multiplexer is determined by the selected SERCOM mode

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

30.5.2. Power Management

The SERCOM will continue to operate in any sleep mode where the selected source clock is running. The SERCOM'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 the *PM – Power Manager* chapter for details on the different sleep modes.

30.5.3. Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the main clock, and the default state of CLK_SERCOMx_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

The SERCOM uses two generic clocks: The Core clock (GCLK_SERCOMx_CORE) and the Slow clock (GCLK_SERCOMx_SLOW). The Core clock is required for SERCOM operation as a host, while the Slow clock is only needed for certain functions. See specific mode chapters for details.

These clocks must be configured and enabled in the generic clock controller before using the SERCOM. Refer to the *GCLK – Generic Clock Controller* chapter for details.

The 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.

30.5.4. DMA

The SPI generates the following DMA requests:

- Receive Complete (RXC): The request is set when data is available in the receive buffer (RX DATA). The request is cleared when DATA is read.
- Data Register Empty (DRE): The request is set when the transmit buffer (TX DATA) is empty. The request is cleared when DATA is written.

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the SPI's DMA requests. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

30.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use SERCOM SPI interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 30-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
SERCOMn	DRE	Data Register Empty	
	RXC	Receive Complete	
	TXC	Transmit Complete	
	SSL	SPI Select Low	
	ERROR	Error	The Collision Detected, Inconsistent Sync Field, Buffer Overflow, Frame Error, and Parity Error bits in the Status register (STATUS.COLL, STATUS.ISF, STATUS.BUFOVF, STATUS.FERR and STATUS.PERR).

30.5.6. Events

Not applicable.

30.5.7. Analog Connections

Not applicable.

30.6. Register Summary - SERCOMn.SPI

SERCOM in SPI Modes (Client/Host)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
		15:8								IBON
		23:16			DIPO[1:0]				DOPO[1:0]	
		31:24		DORD	CPOL	CPHA	FORM[3:0]			
0x04	CTRLB	7:0		PLOADEN				CHSIZE[2:0]		
		15:8	AMODE[1:0]		MSSEN				SSDE	
		23:16							RXEN	
		31:24								
0x08 ... 0x0B	Reserved									
0x0C	BAUD	7:0	BAUD[7:0]							
0x0D ... 0x13	Reserved									
0x14	INTENCLR	7:0	ERROR				SSL	RXC	TXC	DRE
0x15	Reserved									
0x16	INTENSET	7:0	ERROR				SSL	RXC	TXC	DRE
0x17	Reserved									
0x18	INTFLAG	7:0	ERROR				SSL	RXC	TXC	DRE
0x19	Reserved									
0x1A	STATUS	7:0						BUFOVF		
		15:8								
0x1C	SYNCBUSY	7:0						CTRLB	ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x20 ... 0x23	Reserved									
0x24	ADDR	7:0	ADDR[7:0]							
		15:8								
		23:16	ADDRMASK[7:0]							
		31:24								
0x28	DATA	7:0	DATA[7:0]							
		15:8								DATA[8]
0x2A ... 0x2F	Reserved									
0x30	DBGCTRL	7:0								DBGSTOP

30.6.1. Control A - SPI

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		DORD	CPOL	CPHA	FORM[3:0]			
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			DIPO[1:0]				DOPO[1:0]	
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

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

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit 30 – DORD Data Order
This bit selects the data order when a character is shifted out from the shift register.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	MSB	MSb is transferred first
1	LSB	LSb is transferred first

Bit 29 – CPOL Clock Polarity
In combination with the Clock Phase bit (CPHA), this bit determines the SPI transfer mode.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	IDLE_LOW	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	IDLE_HIGH	SCK is high when IDLE. The leading edge of a clock cycle is a falling edge, while the trailing edge is a rising edge.

Bit 28 – CPHA Clock Phase
In combination with the Clock Polarity bit (CPOL), this bit determines the SPI transfer mode.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0	LEADING_EDGE	The data is sampled on a leading SCK edge and changed on a trailing SCK edge
1	TRAILING_EDGE	The data is sampled on a trailing SCK edge and changed on a leading SCK edge

Bits 27:24 – FORM[3:0] Frame Format

This bit field selects the various frame formats supported by the SPI in Client mode. When the SPI_FRAME_WITH_ADDR format is selected, the first byte received is checked against the ADDR register.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	SPI_FRAME	SPI Frame
0x2	SPI_FRAME_WITH_ADDR	SPI Frame with Address

Bits 21:20 – DIPO[1:0] Data In Pinout

These bits define the data in (DI) pad configurations.
In host operation, DI is MISO.
In client operation, DI is MOSI.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	MUX0	SERCOM PAD[0] is used as data input
0x1	MUX1	SERCOM PAD[1] is used as data input
0x2	MUX2	SERCOM PAD[2] is used as data input
0x3	MUX3	SERCOM PAD[3] is used as data input

Bits 17:16 – DOPO[1:0] Data Out Pinout

This bit defines the available pad configurations for data out (DO), the serial clock (SCK) and the SPI select ($\overline{SS}$).
In host operation, DO is MOSI.
In client operation, DO is MISO.

Notes:

- This bit field is enable-protected. This bit field is not synchronized.
- In host operation, the SPI select line ($\overline{SS}$) is only controlled by DOPO when CTRLB.MSSEN is set to '1'.

Value	Name	Description
0x0	MUX0	PAD[0]=DO, PAD[1]=SCK, PAD[2]=$\overline{SS}$
0x1	MUX1	PAD[2]=DO, PAD[3]=SCK, PAD[1]=$\overline{SS}$
0x2	MUX2	PAD[3]=DO, PAD[1]=SCK, PAD[2]=$\overline{SS}$
0x3	MUX3	PAD[0]=DO, PAD[3]=SCK, PAD[1]=$\overline{SS}$

Bit 8 – IBON Immediate Buffer Overflow Notification

This bit controls when the buffer overflow status bit (STATUS.BUFOVF) is set when a buffer overflow occurs.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	STATUS.BUFOVF is set when it occurs in the data stream
1	STATUS.BUFOVF is set immediately upon buffer overflow

Bit 7 – RUNSTDBY Run In Standby

This bit defines the functionality in Standby sleep mode.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	The SPI is disabled in Standby sleep mode. In Client mode, the SPI is disabled immediately when Standby sleep mode is entered. In Host mode, the SPI is disabled when the ongoing transaction is finished.
1	The SPI is enabled in Standby sleep mode

Bits 4:2 – MODE[2:0] Operating Mode
This bit field controls the SERCOM mode.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	USART_EXT	USART with external clock
0x1	USART_INT	USART with internal clock
0x2	SPI_SLAVE	SPI Client mode
0x3	SPI_MASTER	SPI Host mode
0x4	I2C_SLAVE	I^2C Client mode
0x5	I2C_MASTER	I^2C Host mode

Bit 1 – ENABLE Enable
Notes:
1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure the CTRLA.ENABLE synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled or being disabled
1	The peripheral is enabled or being enabled

Bit 0 – SWRST Software Reset
Writing '0' to this bit has no effect.
Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.
Writing '1' to CTRL.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.
Notes:
1. This bit is write-synchronized: SYNCBUSY.SWRST must be checked to ensure the CTRLA.SWRST synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	There is no reset operation in progress
1	The reset operation is in progress

30.6.2. Control B - SPI

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

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

Bit	15	14	13	12	11	10	9	8
	AMODE[1:0]		MSSEN				SSDE	
Access	R/W	R/W	R/W				R/W	
Reset	0	0	0				0	

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

Bit 17 – RXEN Receiver Enable
Writing '0' 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 '1' to CTRLB.RXEN when the SPI is enabled will set SYNCBUSY.CTRLB, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as '1'.
This bit is not enable-protected.

Value	Description
0	The receiver is disabled or being enabled
1	The receiver is enabled or it will be enabled when SPI is enabled

Bits 15:14 – AMODE[1:0] Address Mode
These bits set the Client addressing mode when the frame format (CTRLA.FORM) with address is used. They are unused in Host mode.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	MASK	ADDRMASK is used as a mask to the ADDR register
0x1	2ADDRS	The client responds to the two unique addresses in ADDR and ADDRMASK
0x2	RANGE	The client responds to the range of addresses between and including ADDR and ADDRMASK with ADDR as the upper limit

Bit 13 – MSSEN Host SPI Select Enable
This bit enables hardware SPI select ($\overline{SS}$) control. When Hardware SPI Select Control is enabled (CTRLB.MSSEN), the $\overline{SS}$ pin goes low before, and high after each transferred word. Refer to the Hardware Controlled SS section for details.

Note:

1. This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Hardware $\overline{SS}$ control is disabled
1	Hardware $\overline{SS}$ control is enabled

Bit 9 – SSDE SPI Select Low Detect Enable

This bit enables wake up when the SPI select ($\overline{SS}$) pin transitions from high to low.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	$\overline{SS}$ low detector is disabled
1	$\overline{SS}$ low detector is enabled

Bit 6 – PLOADEN Client Data Preload Enable

Setting this bit will enable preloading of the client shift register when there is no transfer in progress. If the $\overline{SS}$ line is high when DATA is written, it will be transferred immediately to the shift register.

The Preload function must not be used if the Host cannot maintain the $\overline{SS}$ low until transmission is complete.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Preloading of Client shift register when no transfer in progress is disabled
1	Preloading of Client shift register when no transfer in progress is enabled

Bits 2:0 – CHSIZE[2:0] Character Size

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0	8_BIT	Eight bits
1	9_BIT	Nine bits
Other	—	Reserved

30.6.3. Baud Rate - SPI

Name: BAUD
Offset: 0x0C
Reset: 0x00
Property: PAC Write-Protection, 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[7:0] Baud Register
These bits control the clock generation, as described in the *Clock Generation – Baud-Rate Generator* section.

30.6.4. Interrupt Enable Clear - SPI

Name: INTENCLR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
	ERROR				SSL	RXC	TXC	DRE
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 3 – SSL SPI Select Low Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the SPI Select Low Interrupt Enable bit, which disables the SPI Select Low interrupt.

Value	Description
0	SPI Select Low interrupt is disabled
1	SPI Select Low interrupt is enabled

Bit 2 – RXC Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

Value	Description
0	Receive Complete interrupt is disabled
1	Receive Complete interrupt is enabled

Bit 1 – TXC Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Transmit Complete Interrupt Enable bit, which disables the Transmit Complete interrupt.

Value	Description
0	Transmit Complete interrupt is disabled
1	Transmit Complete interrupt is enabled

Bit 0 – DRE Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

Value	Description
0	Data Register Empty interrupt is disabled
1	Data Register Empty interrupt is enabled

30.6.5. Interrupt Enable Set - SPI

Name: INTENSET
Offset: 0x16
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
	ERROR				SSL	RXC	TXC	DRE
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 3 – SSL SPI Select Low Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the SPI Select Low Interrupt Enable bit, which enables the SPI Select Low interrupt.

Value	Description
0	SPI Select Low interrupt is disabled
1	SPI Select Low interrupt is enabled

Bit 2 – RXC Receive Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

Value	Description
0	Receive Complete interrupt is disabled
1	Receive Complete interrupt is enabled

Bit 1 – TXC Transmit Complete Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

Value	Description
0	Transmit Complete interrupt is disabled
1	Transmit Complete interrupt is enabled

Bit 0 – DRE Data Register Empty Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

Value	Description
0	Data Register Empty interrupt is disabled
1	Data Register Empty interrupt is enabled

30.6.6. Interrupt Flag Status and Clear - SPI

Name: INTFLAG
Offset: 0x18
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
	ERROR				SSL	RXC	TXC	DRE
Access	R/W				R/W	R	R/W	R
Reset	0				0	0	0	0

Bit 7 – ERROR Error
This flag is cleared by writing '1' to it.
This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. The BUFOVF error will set this interrupt flag.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

Bit 3 – SSL SPI Select Low
This flag is cleared by writing '1' to it.
This bit is set when a high to low transition is detected on the $\overline{SS}$ pin in Client mode and SPI Select Low Detect (CTRLB.SSDE) is enabled.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

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 '0' to this bit has no effect.
Writing '1' to this bit has no effect.

Bit 1 – TXC Transmit Complete
This flag is cleared by writing '1' to it or by writing new data to DATA.
In Host mode, this flag is set when the data have been shifted out and there are no new data in DATA.
In Client mode, this flag is set when the $\overline{SS}$ pin is pulled high. If address matching is enabled, this flag is only set if the transaction was initiated with an address match.
Writing '0' to this bit has no effect.
Writing '1' 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 '0' to this bit has no effect.
Writing '1' to this bit has no effect.

30.6.7. Status - SPI

Name: STATUS
Offset: 0x1A
Reset: 0x0000
Property: –

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

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 '1' to the bit or by disabling the receiver.
This bit is set when a buffer overflow condition is detected. See also CTRLA.IBON for overflow handling.
When set, the corresponding RxDATA will be zero.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear it.

Value	Description
0	No Buffer Overflow has occurred
1	A Buffer Overflow has occurred

30.6.8. Synchronization Busy - SPI

Name: SYNCBUSY
Offset: 0x1C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						CTRLB	ENABLE	SWRST
Access						R	R	R
Reset						0	0	0

Bit 2 – CTRLB CTRLB Synchronization Busy
Writing to the CTRLB when the SERCOM is enabled requires synchronization. Ongoing synchronization is indicated by this bit. If CTRLB is written while this bit is set to '1', an APB error will be generated.

Value	Description
0	CTRLB synchronization is not busy
1	CTRLB synchronization is busy

Bit 1 – ENABLE SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. Ongoing synchronization is indicated by this bit.

Value	Description
0	Enable synchronization is not busy
1	Enable synchronization is busy

Bit 0 – SWRST Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. Ongoing synchronization is indicated by this bit.

Value	Description
0	SWRST synchronization is not busy
1	SWRST synchronization is busy

30.6.9. Address - SPI

Name: ADDR
Offset: 0x24
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

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								
Reset								

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 23:16 – ADDRMASK[7:0] Address Mask
These bits hold the address mask when the transaction format with address is used (CTRLA.FORM, CTRLB.AMODE).

Bits 7:0 – ADDR[7:0] Address
These bits hold the address when the transaction format with address is used (CTRLA.FORM, CTRLB.AMODE).

30.6.10. Data - SPI

Name: DATA
Offset: 0x28
Reset: 0x0000
Property: –

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

Reading these bits will return the contents of the receive data buffer. The register must 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 must be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

30.6.11. Debug Control - SPI

Name: DBGCTRL
Offset: 0x30
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGSTOP Debug Stop Mode
This bit controls the functionality when the CPU is halted by an external debugger.
This bit will be cleared after a software system reset.

Value	Description
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

31. SERCOM I²C — SERCOM Inter-Integrated Circuit

31.1. Features

- Host or Client Operation
- Can be used with DMA
- I²C Compatible
- SMBus™ Compatible
- PMBus Compatible
- Supports Standard-Mode (100 kbit/s), Fast-Mode (400 kbit/s) and Fast-Mode Plus (1 Mbit/s)
- 4-Wire Operation Supported
- Physical Interface Includes:
  - Slew-Rate Limited Outputs
  - Filtered Inputs
- Client Operation:
  - Operation in all Sleep Modes
  - Wake-Up on Address Match
  - 7-bit Address Match in Hardware

31.2. Overview

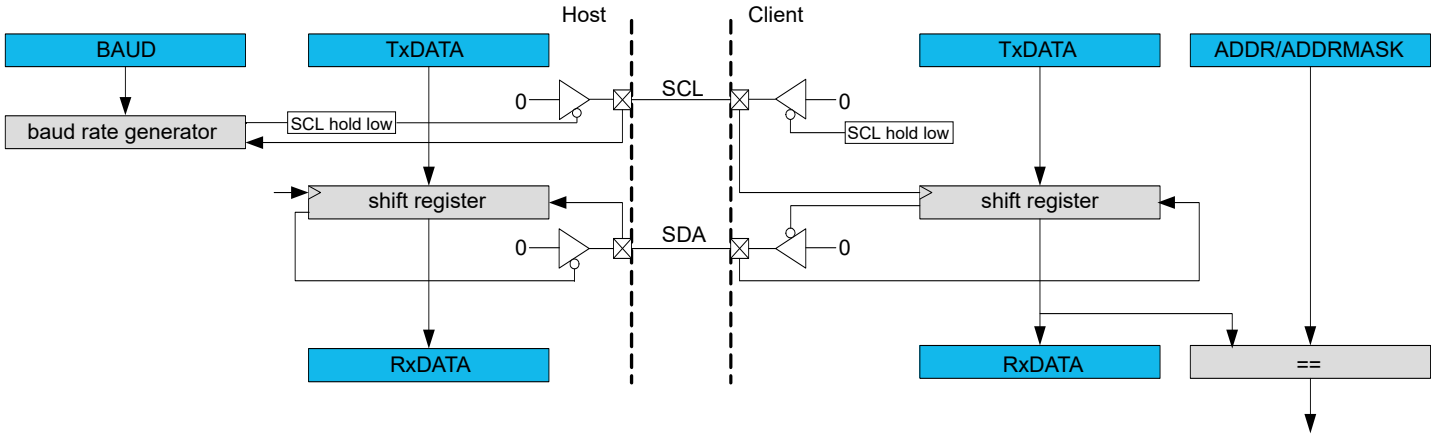
The inter-integrated circuit (I²C) interface is one of the available modes in the Serial Communication Interface (SERCOM).

A SERCOM instance can be configured to operate either as an I²C host or an I²C client as shown in Figure 31-1.

Both host and client have an interface containing a shift register, a transmit buffer and a receive buffer. In addition, the I²C host uses the SERCOM baud rate generator, while the I²C client uses the SERCOM address match logic.

31.3. Block Diagram

Figure 31-1. I²C Single-Host Single-Client Interconnection



Registers labeled in uppercase are accessible from the CPU, while those with lowercase labels are internal to the SERCOM.

31.3.1. Signal Description

Table 31-1. SERCOM I2C Signals

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 operation)
PAD[3]	Digital I/O	SCL_OUT (4-wire operation)

All SERCOM I^2C pins support the Fast-mode frequency.

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter for details.

31.4. Functional Description

31.4.1. Initialization

1. Select I^2C Host or Client mode by writing to the Operating Mode bits in the Control A register (CTRLA.MODE).
2. In Host mode:
   a. Select the inactive bus time-out in the Inactive Time-Out bit group in the Control A register (CTRLA.INACTOUT).
   b. Write the Baud Rate register (BAUD) to generate the desired baud rate.

   In Client mode:
   a. Configure the address match configuration by writing the Address Mode value in the Control B register (CTRLB.AMODE).
   b. Set the Address and Address Mask value in the Address register (ADDR.ADDR and ADDR.ADDRMASK) according to the address configuration.
3. Enable the module by writing to the Enable bit in the Control A register (CTRLA.ENABLE).

31.4.2. Operation

The I^2C interface uses two physical lines for communication:

- Serial Data Line (SDA) for data transfer
- Serial Clock Line (SCL) for the bus clock

A transaction starts with the I^2C host sending the start condition, followed by a 7-bit address and a direction bit that indicates whether there will be a read from or write to the client.

The addressed I^2C client will then acknowledge (ACK) the address, and data packet transactions can begin. Every 9-bit data packet consists of eight data bits followed by a one-bit reply indicating whether the data was acknowledged or not.

If a data packet is not acknowledged (NACK), whether by the I^2C client or host, the I^2C host takes action by either terminating the transaction by sending the stop condition, or by sending a repeated start to transfer more data.

The following figure illustrates the possible transaction formats and explains the transaction symbols. These symbols will be used in the following descriptions.

Figure 31-2. Transaction Diagram Symbols

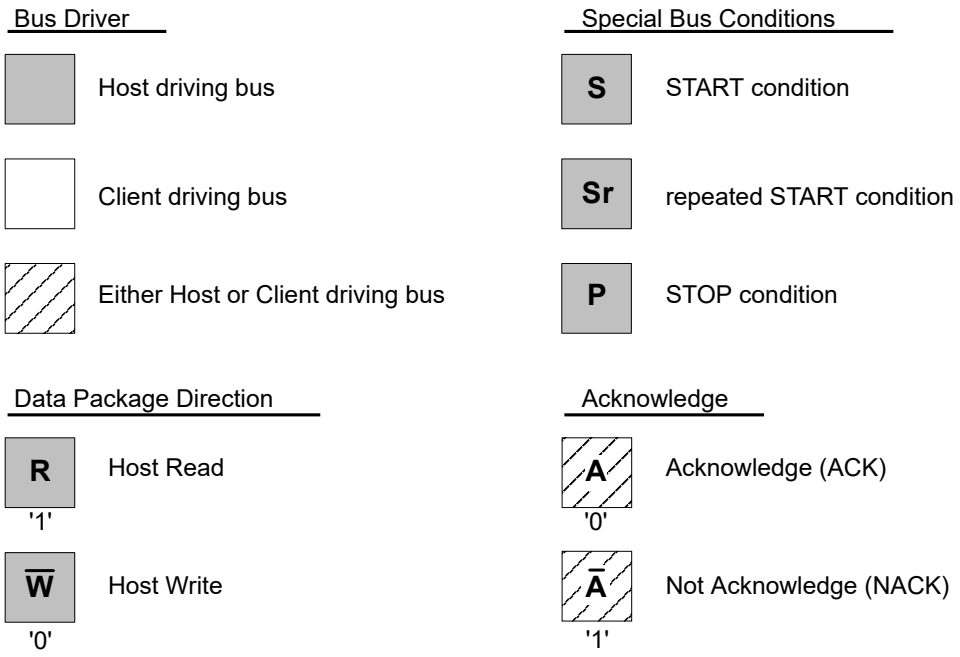
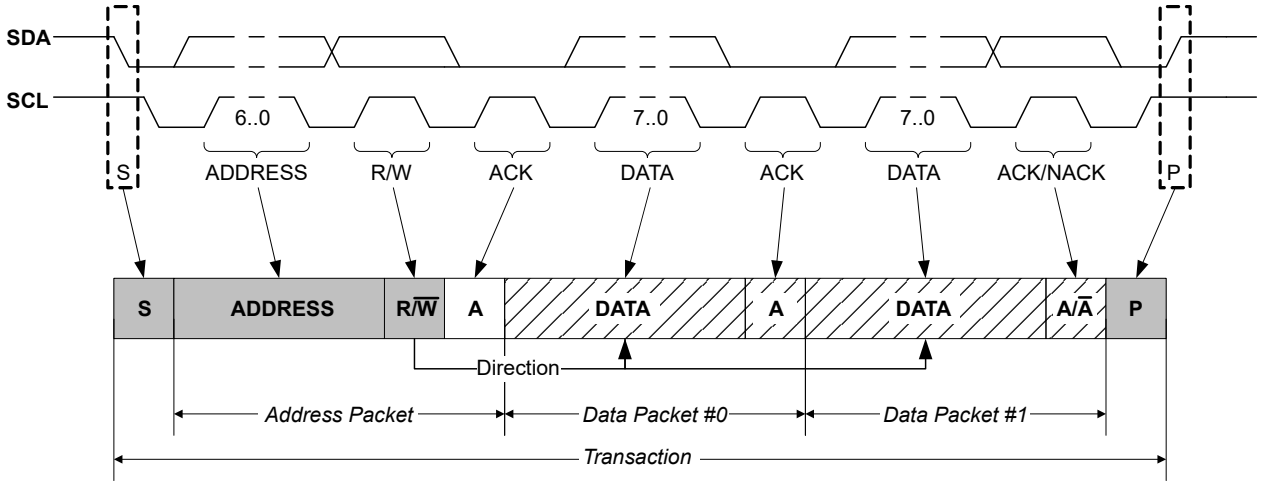


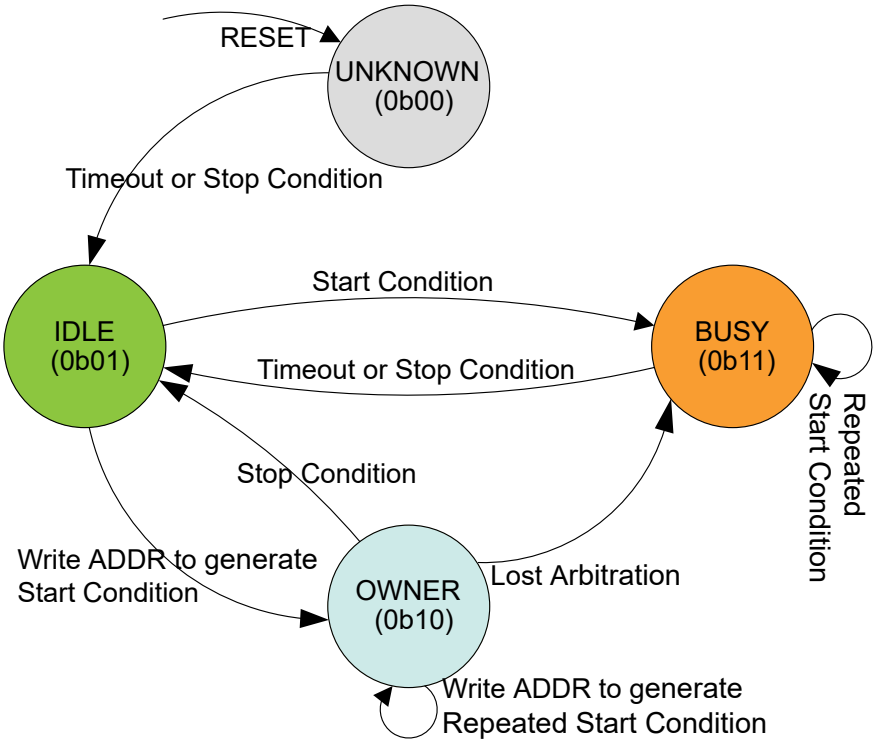
Figure 31-3. Basic I²C Transaction Diagram



31.4.2.1. 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 with running GCLK_SERCOM_x clocks. 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 Bus State Diagram. Software can get the current bus state by reading the Host Bus State bits in the Status register (STATUS.BUSSTATE). The value of STATUS.BUSSTATE in the figure is shown in binary.

Figure 31-4. Bus State Diagram



The bus state machine is active when the I²C host is enabled.

After the I²C host has been enabled, the bus state is UNKNOWN (`0b00`). From the UNKNOWN state, the bus will transition to IDLE (`0b01`) by either:

- Forcing by writing `0b01` to STATUS.BUSSTATE
- A stop condition is detected on the bus
- If the inactive bus time-out is configured for SMBus compatibility (CTRLA.INACTOUT) and a time-out occurs

Note: Once a known bus state is established, the bus state logic will not re-enter the UNKNOWN state.

When the bus is IDLE, it is ready for a new transaction. If a start condition is issued on the bus by another I²C host in a multi-host setup, the bus becomes BUSY (`0b11`). The bus will re-enter IDLE either when a stop condition is detected, or when a time-out occurs (inactive bus time-out needs to be configured).

If a start condition is generated internally by writing the Address bit group in the Address register (ADDR.ADDR) while IDLE, the OWNER state (`0b10`) is entered. If the complete transaction was performed without interference, i.e., arbitration was not lost, the I²C host can issue a stop condition, which will change the bus state back to IDLE.

However, if a packet collision is detected while in 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.

Note: Violating the protocol may cause the I²C to hang. If this happens it is possible to recover from this state by a software reset (CTRLA.SWRST='1').

31.4.2.2. I²C Host Operation

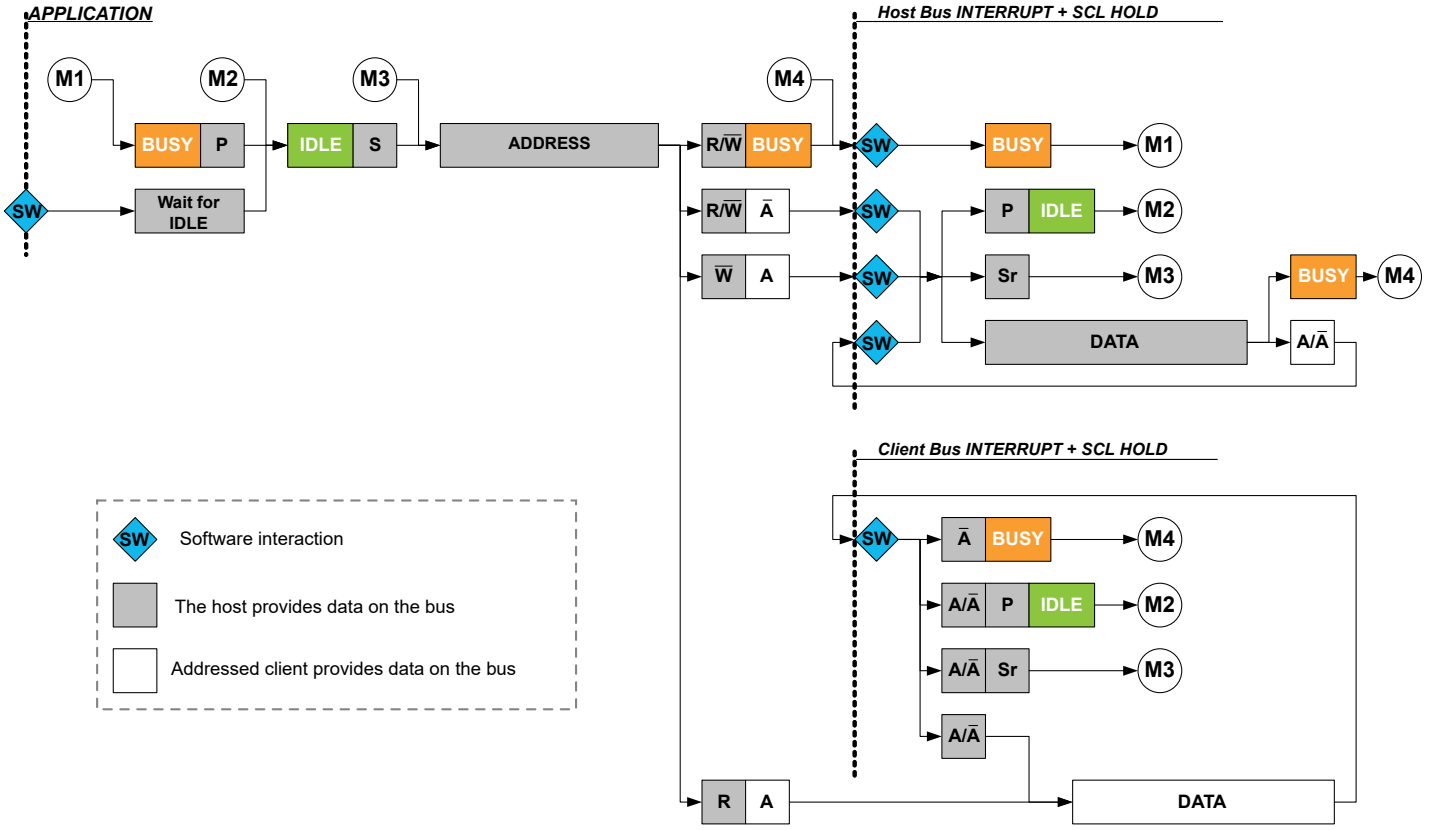
The I²C host is byte-oriented and interrupt based. The number of interrupts generated is kept at a minimum by automatic handling of most incidents. The software driver complexity and code size are reduced by auto-triggering of operations, and smart mode, which can be enabled by the Smart Mode Enable bit in the Control B register (CTRLB.SMEN).

The I²C host has two interrupt strategies.

When SCL Clock Stretch Mode (CTRLA.SCLSM) is '0', SCL is stretched before or after the Acknowledge bit. In this mode, the I²C host operates according to Host Behavioral Diagram (SCLSM=0). The circles labeled "M*n*" (M1, M2..) indicate the nodes the bus logic can jump to, based on software or hardware interaction.

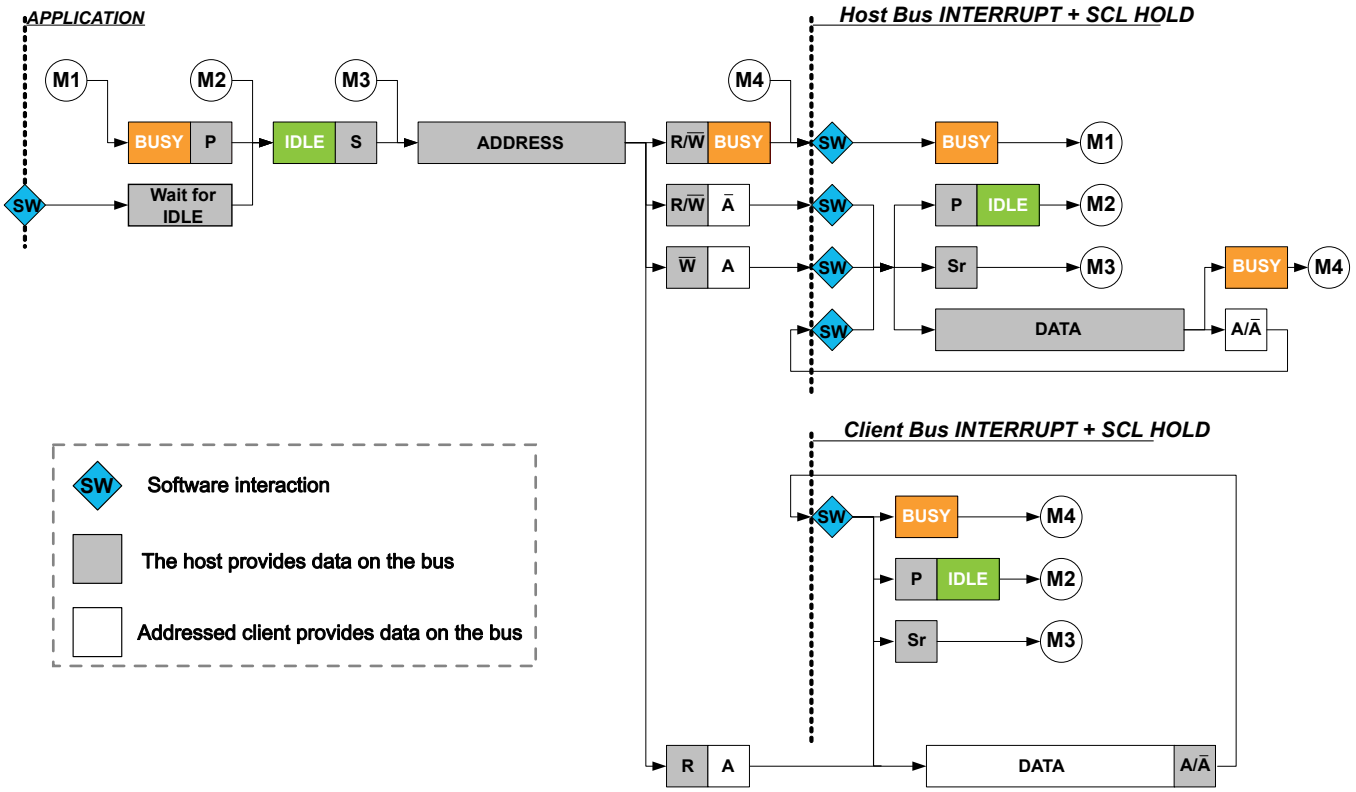
This diagram is used as reference for the description of the I²C host operation throughout the document.

Figure 31-5. I²C Host Behavioral Diagram (SCLSM=0)



In the second strategy (CTRLA.SCLSM='1'), interrupts only occur after the ACK bit, as in Host Behavioral Diagram (SCLSM=1). This strategy can be used when it is not necessary to check DATA before acknowledging.

Figure 31-6. I²C Host Behavioral Diagram (SCLSM=1)



31.4.2.2.1. Host Clock Generation

The SERCOM peripheral supports several I²C bidirectional modes:

- Standard-mode (*Sm*), with a bit rate up to 100 kbit/s
- Fast-mode (*Fm*), with a bit rate up to 400 kbit/s
- Fast-mode Plus (*Fm+*), with a bit rate up to 1 Mbit/s

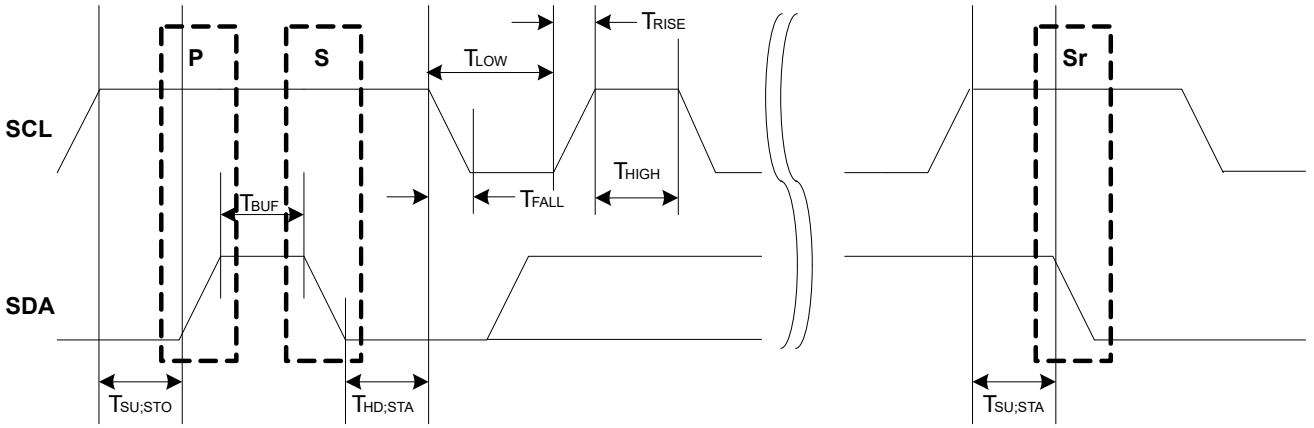
The Host clock configuration for *Sm, Fm,* and *Fm+* are described in Clock Generation (Standard-Mode, Fast-Mode, and Fast-Mode Plus).

Clock Generation (Standard-Mode, Fast-Mode, and Fast-Mode Plus)

In I²C *Sm, Fm*, and *Fm+* mode, the Host clock (SCL) frequency is determined as follows.

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 31-7. SCL Timing



The following parameters are timed using the SCL low time period T_{LOW}. This comes from the Host Baud Rate Low bit group in the Baud Rate register (BAUD.BAUDLOW). When BAUD.BAUDLOW='0', or the Host Baud Rate bit group in the Baud Rate register (BAUD.BAUD) determines it.

- T_{LOW} – Low period of the SCL clock
- $T_{SU;STO}$ – Set-up time for the stop condition
- T_{BUF} – Bus free time between stop and start conditions
- $T_{HD;STA}$ – Hold time for (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 the *Electrical Characteristics* section
- T_{FALL} is determined by the open-drain current limit and bus impedance; can typically be regarded as zero. Refer to the *Electrical Characteristics* section for details.

The SCL frequency is given by:

$$f_{\mathrm{SCL}} = \frac{1}{T_{\mathrm{LOW}} + T_{\mathrm{HIGH}} + T_{\mathrm{RISE}}}$$

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:

$$f_{\mathrm{SCL}} = \frac{f_{\mathrm{GCLK}}}{10 + 2BAUD + f_{\mathrm{GCLK}} \cdot T_{\mathrm{RISE}}}$$

When BAUD.BAUDLOW is non-zero, the following formula determines the SCL frequency:

$$f_{\mathrm{SCL}} = \frac{f_{\mathrm{GCLK}}}{10 + BAUD + BAUDLOW + f_{\mathrm{GCLK}} \cdot T_{\mathrm{RISE}}}$$

The following formulas can determine the SCL T_{LOW} and T_{HIGH} times:

$$T_{\mathrm{LOW}} = \frac{BAUDLOW + 5}{f_{\mathrm{GCLK}}}$$

$$T_{\mathrm{HIGH}} = \frac{BAUD + 5}{f_{\mathrm{GCLK}}}$$

Note: The I²C standard *Fm+* (Fast-mode plus) requires a nominal high to low SCL ratio of 1:2, and BAUD must be set accordingly. At a minimum, BAUD.BAUD and/or BAUD.BAUDLOW must be non-zero.

The minimum time between SDA transition and SCL rising edge is six APB cycles when the DATA register is written in Smart mode. If a greater start-up time is required due to long rise times, the time between DATA write and IF clear must be controlled by software.
Note: When timing is controlled by the user, the Smart mode cannot be enabled.

31.4.2.2.2. Transmitting Address Packets

The I²C host starts a bus transaction by writing the I²C client address to ADDR.ADDR and the direction bit. If the bus is busy, the I²C host will wait until the bus becomes idle before continuing the operation. When the bus is IDLE, the I²C host will issue a start condition on the bus. The I²C host will then transmit an address packet using the address written to ADDR.ADDR. After the address packet has been transmitted by the I²C host, one of four cases will arise according to 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 Host on Bus bit in the Interrupt Flag Status and Clear 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 host is no longer allowed to execute 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 Host Bus Error bit in the Status register (STATUS.BUSERR) are both set in addition to STATUS.ARBLOST.

The Host 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 moment, 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 there is no I²C client device responding to the address packet, then the INTFLAG.MB interrupt flag and STATUS.RXNACK will be set. The clock hold is active at this point, preventing further activity on the bus.

The missing ACK response can indicate that the I²C client is busy with other tasks or sleeping. Therefore, it is not able to respond. In this event, the next step can be either issuing a stop condition (recommended) or resending the address packet by a repeated start condition. When using SMBus logic, the client must ACK the address. If there is no response, it means that the client is not available on the bus.

Case 3: Address packet transmit complete – Write packet, Host on Bus set

If the I²C host receives an acknowledge response from the I²C client, INTFLAG.MB will be set and STATUS.RXNACK will be 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:

- Initiate a data transmit operation 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, Client on Bus set

If the I²C host receives an ACK from the I²C client, the I²C host proceeds to receive the next byte of data from the I²C client. When the first data byte is received, the Client on Bus bit in the Interrupt Flag register (INTFLAG.SB) will be set and STATUS.RXNACK will be 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:

- Let the I²C host continue to read data by acknowledging the data received. ACK can be sent by software, or automatically in Smart mode.
- Transmit a new address packet
- Terminate the transaction by issuing a stop condition

Note: An ACK or NACK will be automatically transmitted if Smart mode is enabled. The Acknowledge Action bit in the Control B register (CTRLB.ACKACT) determines whether ACK or NACK has to be sent.

31.4.2.2.3. Transmitting Data Packets

When an address packet with direction Host Write (see Figure 31-3) was transmitted successfully , INTFLAG.MB will be set. The I²C host will start transmitting data via the I²C bus by writing to DATA.DATA, and monitor continuously for packet collisions.

If a collision is detected, the I²C host will lose arbitration and STATUS.ARBLOST will be set. If the transmit was successful, the I²C host will receive an ACK bit from the I²C client, and STATUS.RXNACK will be cleared. INTFLAG.MB will be set in both cases, regardless of arbitration outcome.

It is recommended to read STATUS.ARBLOST and handle the arbitration lost condition in the beginning of the I²C Host on Bus interrupt. 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 host is not allowed to continue transmitting data packets if a NACK is received from the I²C client.

31.4.2.2.4. Receiving Data Packets (SCLSM=0)

When INTFLAG.SB is set, the I²C host will already have received one data packet. The I²C host must respond by sending either an ACK or NACK. Sending a NACK may be unsuccessful when arbitration is lost during the transmission. In this case, a lost arbitration will prevent setting INTFLAG.SB. Instead, INTFLAG.MB will indicate a change in arbitration. Handling of lost arbitration is the same as for data bit transmission.

31.4.2.2.5. Receiving Data Packets (SCLSM=1)

When INTFLAG.SB is set, the I²C host will already have received one data packet and transmitted an ACK or NACK, depending on CTRLB.ACKACT. At this point, CTRLB.ACKACT must be set to the correct value for the next ACK bit, and the transaction can continue by reading DATA and issuing a command if not in Smart mode.

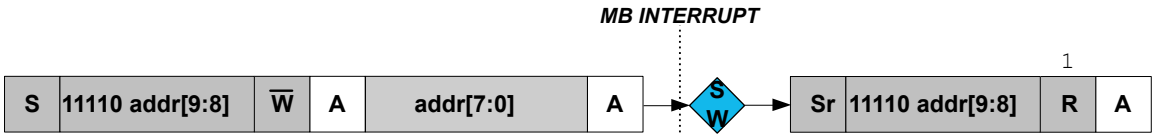
31.4.2.2.6. 10-Bit Addressing

When 10-bit addressing is enabled by the Ten Bit Addressing Enable bit in the Address register (ADDR.TENBITEN='1') and the Address bit field ADDR.ADDR is written, the two address bytes will be transmitted, see 10-bit Address Transmission for a Read Transaction. The addressed client acknowledges the two address bytes, and the transaction continues. Regardless of whether the transaction is a read or write, the host must start by sending the 10-bit address with the direction bit (ADDR.ADDR[0]) being zero.

If the host receives a NACK after the first byte, the write interrupt flag will be raised and the STATUS.RXNACK bit will be set. If the first byte is acknowledged by one or more clients, then the host will proceed to transmit the second address byte and the host will first see the write interrupt flag after the second byte is transmitted. If the transaction direction is read-from-client, the 10-bit

address transmission must be followed by a repeated start and the first seven bits of the address with the read/write bit equal to '1'.

Figure 31-8. 10-bit Address Transmission for a Read Transaction



This implies the following procedure for a 10-bit read operation:

1. Write the 10-bit address to ADDR.ADDR[10:1]. ADDR.TENBITEN must be '1', the direction bit (ADDR.ADDR[0]) must be '0' (can be written simultaneously with ADDR).
2. Once the Host on Bus interrupt is asserted, Write ADDR[7:0] register to '11110 address[9:8] 1'. ADDR.TENBITEN must be cleared (can be written simultaneously with ADDR).
3. Proceed to transmit data.

31.4.2.3. I²C Client Operation

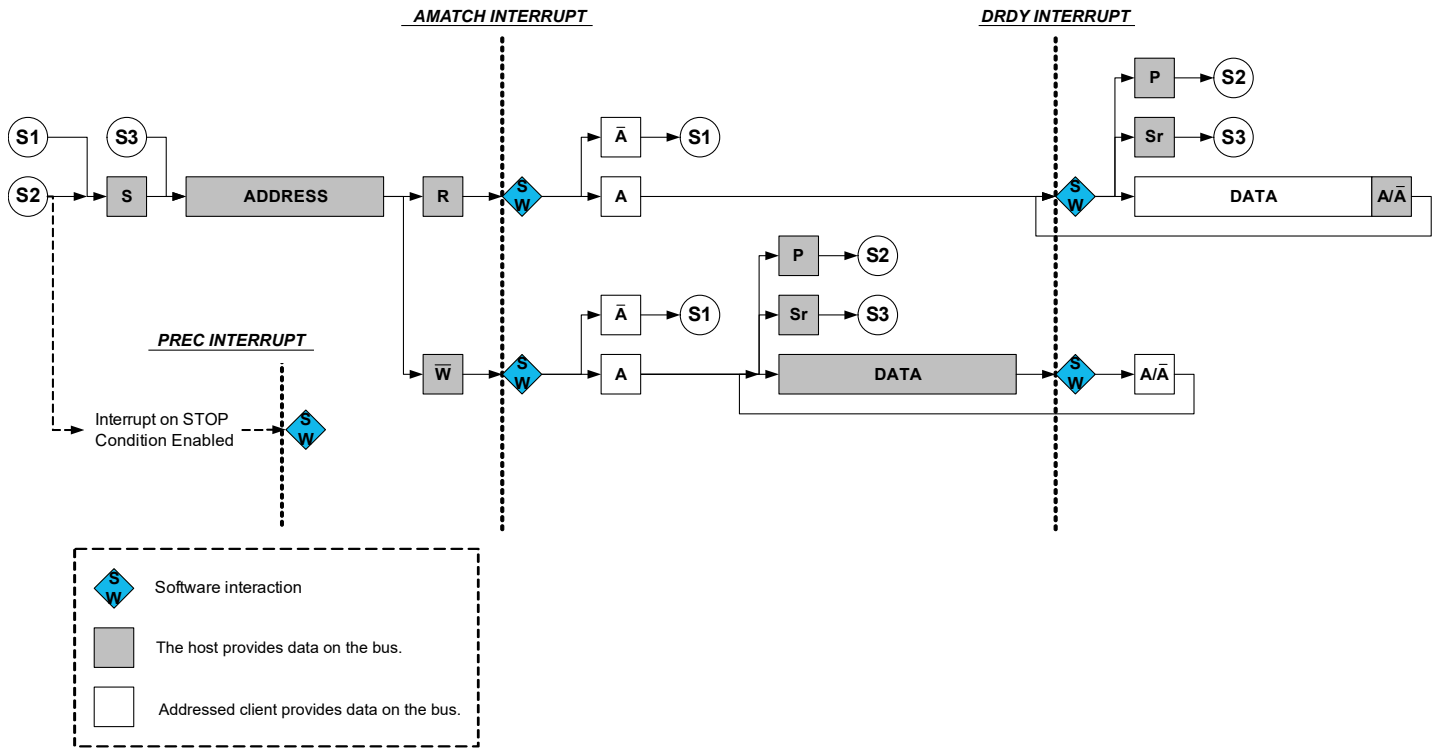
The I²C client is byte-oriented and interrupt-based. The number of interrupts generated is kept at a minimum by automatic handling of most events. The software driver complexity and code size are reduced by auto-triggering of operations, and a special Smart mode, which can be enabled by the Smart Mode Enable bit in the Control B register (CTRLB.SMEN).

The I²C client has two interrupt strategies.

When SCL Stretch Mode bit (CTRLA.SCLSM) is '0', SCL is stretched before or after the acknowledge bit. In this mode, the I²C client operates according to I²C Client Behavioral Diagram (SCLSM=0). The circles labeled "S*n*" (S1, S2..) indicate the nodes the bus logic can jump to, based on software or hardware interaction.

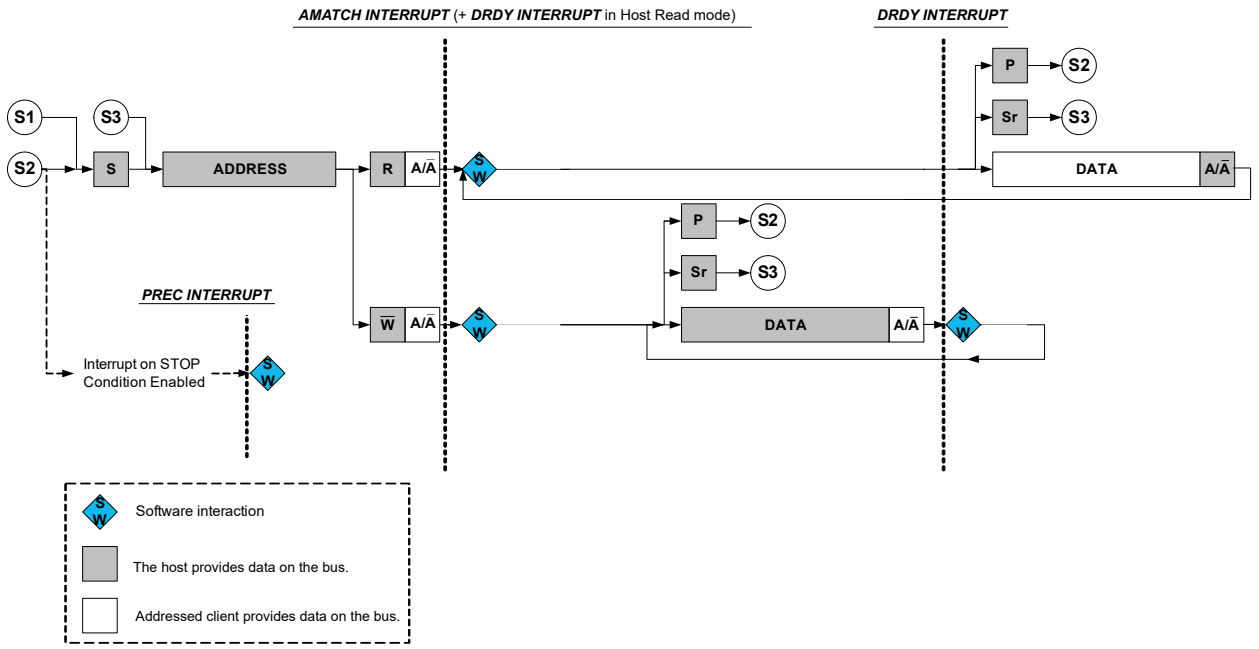
This diagram is used as reference for the description of the I²C client operation throughout the document.

Figure 31-9. I²C Client Behavioral Diagram (SCLSM=0)



In the second strategy (CTRLA.SCLSM='1'), interrupts only occur after the ACK bit is sent as shown in Client Behavioral Diagram (SCLSM=1). This strategy can be used when it is not necessary to check DATA before acknowledging. For host reads, an address and data interrupt will be issued simultaneously after the address acknowledge. However, for host writes, the first data interrupt will be seen after the first data byte has been received by the client and the acknowledge bit has been sent to the host.

Figure 31-10. I²C Client Behavioral Diagram (SCLSM=1)



31.4.2.3.1. Receiving Address Packets (SCLSM=0)

When CTRLA.SCLSM='0', the I²C client stretches the SCL line according to Figure 31-9. When the I²C client is properly configured, it will wait for a start condition.

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 will be rejected, and the I²C client will wait for a new start condition. If the received address is a match, the Address Match bit in the Interrupt Flag register (INTFLAG.AMATCH) will be set.

SCL will be stretched until the I²C client clears INTFLAG.AMATCH. As the I²C client holds the clock by forcing SCL low, the software has unlimited time to respond.

The direction of a transaction is determined by reading the Read / Write Direction bit in the Status register (STATUS.DIR). This 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 client had a packet collision. A collision causes the SDA and SCL lines to be released without any notification to software. Therefore, the next AMATCH interrupt is 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 host, one of two cases will arise based on transfer direction.

Case 1: Address packet accepted – Read flag set

The STATUS.DIR bit is '1', indicating an I²C host read operation. The SCL line is forced low, stretching the bus clock. If an ACK is sent, I²C client hardware will set the Data Ready bit in the Interrupt Flag register (INTFLAG.DRDY), indicating data are needed for transmit. If a NACK is sent, the I²C client 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 client Command bit field in the Control B register (CTRLB.CMD) can be written to '0x3' for both read and write operations as the command execution is dependent on the STATUS.DIR bit. Writing '1' 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 host write operation. The SCL line is forced low, stretching the bus clock. If an ACK is sent, the I²C client will wait for data to be received. Data, repeated start or stop can be received.

If a NACK is sent, the I²C client will wait for a new start condition and address match. Typically, software will immediately acknowledge the address packet by sending an ACK/NACK. The I²C client command CTRLB.CMD = 0x3 can be used for both read and write operation as the command execution is dependent on STATUS.DIR.

Writing '1' to INTFLAG.AMATCH will also cause an ACK/NACK to be sent corresponding to the CTRLB.ACKACT bit.

31.4.2.3.2. Receiving Address Packets (SCLSM=1)

When SCLSM=1, the I²C client will stretch the SCL line only after an ACK, see Client Behavioral Diagram (SCLSM=1). When the I²C client 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 will be rejected and the I²C client will wait for a new start condition.

If the address matches, the acknowledge action as configured by the Acknowledge Action bit Control B register (CTRLB.ACKACT) will be sent and the Address Match bit in the Interrupt Flag register (INTFLAG.AMATCH) is set. SCL will be stretched until the I²C client clears INTFLAG.AMATCH. As the I²C client 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). This 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, the last packet addressed to the I²C client 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 host, INTFLAG.AMATCH must be written to '1' to clear it.

31.4.2.3.3. Receiving and Transmitting Data Packets

After the I²C client 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. After receiving data, the I²C client will send an acknowledge according to CTRLB.ACKACT.

Case 1: Data received

INTFLAG.DRDY is set, and SCL is held low, pending for SW interaction.

Case 2: Data sent

When a byte transmission is successfully completed, the INTFLAG.DRDY interrupt flag is set. If NACK is received, indicated by STATUS.RXNACK=1, the I²C client must expect a stop or a repeated start to be received. The I²C client must release the data line to allow the I²C host to generate a stop or repeated start. Upon detecting a stop condition, the Stop Received bit in the Interrupt Flag register (INTFLAG.PREC) will be set and the I²C client will return to IDLE state.

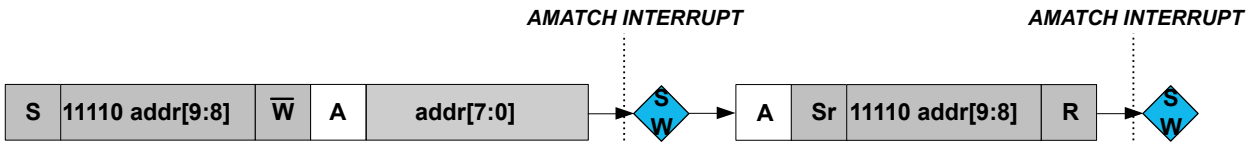
31.4.2.3.4. 10-Bit Addressing

When 10-bit addressing is enabled (ADDR.TENBITEN=1), the two address bytes following a START will be checked against the 10-bit client address recognition. The first byte of the address will always be acknowledged, and the second byte will raise the address interrupt flag, see 10-bit Addressing.

If the transaction is a write, then the 10-bit address will be followed by *N* data bytes.

If the operation is a read, the 10-bit address will be followed by a repeated START and reception of '11110 ADDR[9:8] 1', and the second address interrupt will be received with the DIR bit set. The client matches on the second address as it was addressed by the previous 10-bit address.

Figure 31-11. 10-bit Addressing



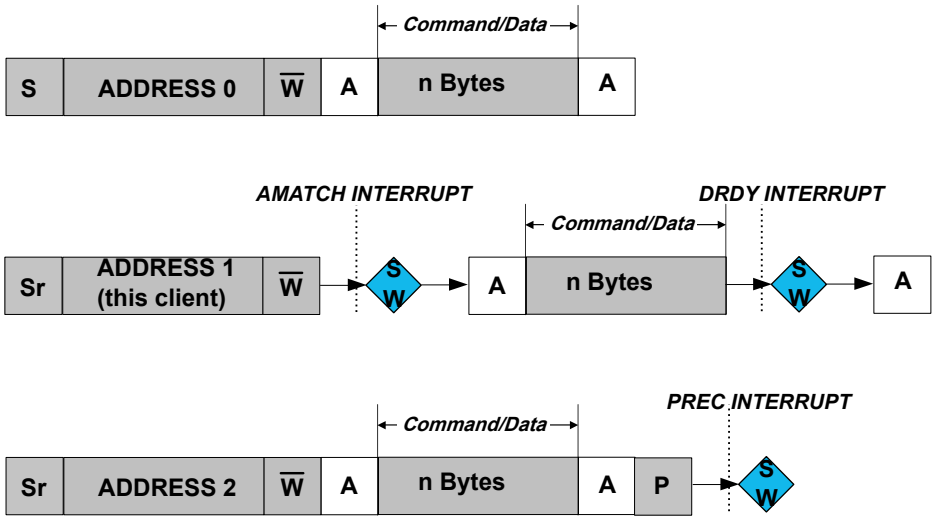
31.4.2.3.5. PMBus Group Command

When the PMBus Group Command bit in the CTRLB register is set (CTRLB.GCMD=1) and 7-bit addressing is used, INTFLAG.PREC will be set if the client has been addressed since the last STOP condition. When CTRLB.GCMD='0', a STOP condition without address match will not be set INTFLAG.PREC.

The group command protocol is used to send commands to more than one device. The commands are sent in one continuous transmission with a single STOP condition at the end. When the STOP condition is detected by the clients addressed during the group command, they all begin executing the command they received.

PMBus Group Command Example shows an example where this client, bearing ADDRESS 1, is addressed after a repeated START condition. There can be multiple clients addressed before and after this client. Eventually, at the end of the group command, a single STOP is generated by the host. At this point a STOP interrupt is asserted.

Figure 31-12. PMBus Group Command Example



31.4.3. Additional Features

31.4.3.1. SMBus

The I^{2}C includes three hardware SCL low time-outs which allow a time-out to occur for SMBus SCL low time-out, host extend time-out, and client extend time-out. This allows for SMBus functionality. These time-outs are driven by the GCLK_SERCOM_SLOW clock. The GCLK_SERCOM_SLOW clock is used to accurately time the time-out and must be configured to use a 32.768 kHz oscillator. The I^{2}C interface also allows for a SMBus compatible SDA hold time.

- $T_{TIMEOUT}$: SCL low time of 25-35 ms – Measured for a single SCL low period. It is enabled by CTRLA.LOWTOUTEN.
- $T_{LOW:SEXT}$: Cumulative clock low extend time of 25 ms – Measured as the cumulative SCL low extend time by a client device in a single message from the initial START to the STOP. It is enabled by CTRLA.SEXTTOEN.
- $T_{LOW:MEXT}$: Cumulative clock low extend time of 10 ms – Measured as the cumulative SCL low extend time by the host device within a single byte from START-to-ACK, ACK-to-ACK, or ACK-to-STOP. It is enabled by CTRLA.MEXTTOEN.

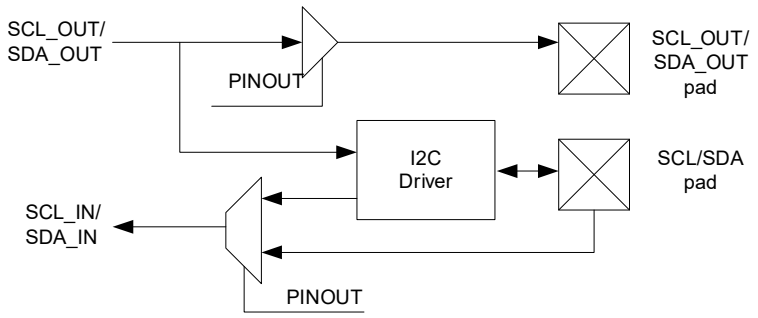
31.4.3.2. Smart Mode

The I^{2}C interface has a Smart mode that simplifies application code and minimizes the user interaction needed to adhere to the I^{2}C protocol. The Smart mode accomplishes this by automatically issuing an ACK or NACK (based on the content of CTRLB.ACKACT) as soon as DATA.DATA is read.

31.4.3.3. 4-Wire Mode

Writing a '1' to the Pin Usage bit in the Control A register (CTRLA.PINOUT) will enable 4-wire mode operation. 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 31-13. I²C Pad Interface



31.4.3.4. 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 (INTFLAG.SB or INTFLAG.MB) is set immediately after the client 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.

The use of the Quick Command mode (CTRLB.QCEN = '1') is only allowed if SCL Stretch Mode is not enabled.

31.4.4. Sleep Mode Operation

I²C Host 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 '1', the GLK_SERCOMx_CORE will also run in Standby Sleep mode. Any interrupt can wake up the device.

If CTRLA.RUNSTDBY=0, the GLK_SERCOMx_CORE will be disabled after any ongoing transaction is finished. Any interrupt can wake up the device.

I²C Client Operation

Writing CTRLA.RUNSTDBY=1 will allow the Address Match interrupt to wake up the device.

When CTRLA.RUNSTDBY=0, all receptions will be dropped.

31.4.5. Debug Operation

When the CPU is halted in debug mode the SERCOM will continue normal operation. Setting the Debug Stop Mode bit in the Debug Control register (DBGCTRL.DBGSTOP) forces the SERCOM to halt operation when the CPU is halted in debug mode.

31.5. Dependencies

31.5.1. I/O Lines

Using the SERCOM's I/O lines requires the I/O pins to be configured.

The SERCOM has four internal pads, PAD[3:0], and the signals from I²C, SPI and USART are routed through these pads via a multiplexer. The configuration of the multiplexer is determined by the selected SERCOM mode

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

31.5.2. Power Management

The SERCOM will continue to operate in any sleep mode where the selected source clock is running. The SERCOM'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 the *PM – Power Manager* chapter for details on the different sleep modes.

31.5.3. Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) can be enabled and disabled in the main clock, and the default state of CLK_SERCOMx_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

The SERCOM uses two generic clocks: The Core clock (GCLK_SERCOMx_CORE) and the Slow clock (GCLK_SERCOMx_SLOW). The Core clock is required for SERCOM operation as a host, while the Slow clock is only needed for certain functions. See specific mode chapters for details.

These clocks must be configured and enabled in the generic clock controller before using the SERCOM. Refer to the *GCLK – Generic Clock Controller* chapter for details.

The 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.

31.5.4. DMA

Smart mode must be enabled for DMA operation in the Control B register by writing CTRLB.SMEN='1'.

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the I^2C's DMA requests. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

31.5.4.1. Client DMA

When using the I^2C client with DMA, an address match will cause the address interrupt flag (INTFLAG.ADDRMATCH) to be raised. After the interrupt has been serviced, data transfer will be performed through DMA.

The I^2C client generates the following requests:

- Write data received (RX): The request is set when host write data is received. The request is cleared when DATA is read.
- Read data needed for transmit (TX): The request is set when data is needed for a host read operation. The request is cleared when DATA is written.

When using the DMA with Client mode, the client requires the transaction length of data that will be requested by the Host so the DMA can be configured properly.

31.5.4.2. Host DMA

When using the I^2C host with DMA, the ADDR register must be written with the desired address (ADDR.ADDR), transaction length (ADDR.LEN), and transaction length enable (ADDR.LENEN). When ADDR.LENEN is written to '1' along with ADDR.ADDR, ADDR.LEN determines the number of data bytes in the transaction from 0 to 255. DMA is then used to transfer ADDR.LEN bytes followed by an automatically generated NACK (for host reads) and a STOP.

If a NACK is received by the client for a host write transaction before ADDR.LEN bytes, a STOP will be automatically generated and the length error (STATUS.LENERR) will be raised along with the INTFLAG.ERROR interrupt.

The I^2C host generates the following requests:

- Read data received (RX): The request is set when host read data is received. The request is cleared when DATA is read.

- Write data needed for transmit (TX): The request is set when data is needed for a host write operation. The request is cleared when DATA is written.

31.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use SERCOM I^2C interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

The I^2C client has the following interrupt sources.

Table 31-2. Available Interrupt Vectors and Sources - I2C Client

Vector Name	Source Name	Condition	Dependency
SERCOMn	DRDY	Data Ready	
	AMATCH	Address Match	
	PREC	Stop Received	
	ERROR	Error	The Client SCL Low Extend Time-out, SCL Low Time-out, Transmit Collision, and Bus Error bits in the Status register (STATUS.SEXTTOUT, STATUS.LOWTOUT, STATUS.COLL and STATUS.BUSERR).

The I^2C host has the following interrupt sources.

Table 31-3. Available Interrupt Vectors and Sources - I2C Host

Vector Name	Source Name	Condition	Dependency
SERCOMn	SB	Client on Bus	
	MB	Host on Bus	
	ERROR	Error	The Transaction Length Error, Client SCL Low Extend Time-out, Host SCL Low Extend Time-out, SCL Low Time-out, Arbitration Lost, and Bus Error bits in the Status register (STATUS.LENERR, STATUS.SEXTTOUT, STATUS.MEXTTOUT, STATUS.LOWTOUT, STATUS.ARBLOST and STATUS.BUSERR).

31.5.6. Events

Not applicable.

31.5.7. Analog Connections

Not applicable.

31.6. Register Summary - SERCOMn.I2CS

SERCOM in I2C Client Mode

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
		15:8								
		23:16	SEXTTOEN		SDAHOLD[1:0]					PINOUT
		31:24		LOWTOUTEN			SCLSM		SPEED[1:0]	
0x04	CTRLB	7:0								
		15:8	AMODE[1:0]					AACKEN	GCMD	SMEN
		23:16						ACKACT	CMD[1:0]	
		31:24								
0x08 ... 0x13	Reserved									
0x14	INTENCLR	7:0	ERROR					DRDY	AMATCH	PREC
0x15	Reserved									
0x16	INTENSET	7:0	ERROR					DRDY	AMATCH	PREC
0x17	Reserved									
0x18	INTFLAG	7:0	ERROR					DRDY	AMATCH	PREC
0x19	Reserved									
0x1A	STATUS	7:0	CLKHOLD	LOWTOUT		SR	DIR	RXNACK	COLL	BUSERR
		15:8							SEXTTOUT	
0x1C	SYNCBUSY	7:0							ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x20 ... 0x23	Reserved									
0x24	ADDR	7:0	ADDR[6:0]							GENCEN
		15:8	TENBITEN							
		23:16	ADDRMASK[6:0]							
		31:24								
0x24	ADDR	7:0	ADDR[6:0]							GENCEN
		15:8	TENBITEN					ADDR[9:7]		
		23:16	ADDRMASK[6:0]							
		31:24						ADDRMASK[9:7]		
0x28	DATA	7:0	DATA[7:0]							

31.6.1. Control A - I2CS

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		LOWTOUTEN			SCLSM		SPEED[1:0]	
Access		R/W			R/W		R/W	R/W
Reset		0			0		0	0

Bit	23	22	21	20	19	18	17	16
	SEXTTOEN		SDAHOLD[1:0]					PINOUT
Access	R/W		R/W	R/W				R/W
Reset	0		0	0				0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit 30 – LOWTOUTEN SCL Low Time-Out
This bit enables the SCL low time-out. If SCL is held low for 25 ms–35 ms, the client will release its clock hold, if enabled, and reset the internal state machine. Any interrupt flags set at the time of time-out will remain set.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Time-out disabled
1	Time-out enabled

Bit 27 – SCLSM SCL Clock Stretch Mode
This bit controls when SCL will be stretched for software interaction.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	SCL stretch according to Figure 31-9
1	SCL stretch only after ACK bit according to Figure 31-10

Bits 25:24 – SPEED[1:0] Transfer Speed
These bits define the bus speed.
Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	SM_MODE	Standard-mode (Sm) and Fast-mode (Fm)
0x1	FASTPLUS_MODE	Fast-mode Plus (Fm+)
Other	—	Reserved

Bit 23 – SEXTTOEN Client SCL Low Extend Time-Out

This bit enables the client SCL low extend time-out. If SCL is cumulatively held low for greater than 25 ms from the initial START to a STOP, the client will release its clock hold if enabled and reset the internal state machine. Any interrupt flags set at the time of time-out will remain set. If the address was recognized, PREC will be set when a STOP is received.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Time-out disabled
1	Time-out enabled

Bits 21:20 – SDAHOLD[1:0] SDA Hold Time

These bits define the SDA hold time with respect to the negative edge of SCL.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	DIS	Disabled
0x1	75NS	50-100 ns hold time
0x2	450NS	300-600 ns hold time
0x3	600NS	400-800 ns hold time

Bit 16 – PINOUT Pin Usage

This bit sets the pin usage to either two- or four-wire operation:

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	4-wire operation is disabled
1	4-wire operation is enabled

Bit 7 – RUNSTDBY Run in Standby

This bit defines the functionality in Standby Sleep mode.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Disabled – All reception is dropped
1	Wake on address match, if enabled

Bits 4:2 – MODE[2:0] Operating Mode

This bit field controls the SERCOM mode.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	USART_EXT	USART with external clock
0x1	USART_INT	USART with internal clock
0x2	SPI_SLAVE	SPI Client mode
0x3	SPI_MASTER	SPI Host mode
0x4	I2C_SLAVE	I^2C Client mode
0x5	I2C_MASTER	I^2C Host mode

Bit 1 – ENABLE Enable

Notes:

1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure the CTRLA.ENABLE synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled or being disabled

Value	Description
1	The peripheral is enabled

Bit 0 – SWRST Software Reset

Writing '0' to this bit has no effect.
Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.
Writing '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Notes:

1. This bit is write-synchronized: SYNCBUSY.SWRST must be checked to ensure the CTRLA.SWRST synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	There is no reset operation in progress
1	The reset operation is in progress

31.6.2. Control B - I2CS

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						ACKACT	CMD[1:0]	
Access						R/W	W	W
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
	AMODE[1:0]					AACKEN	GCMD	SMEN
Access	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 18 – ACKACT Acknowledge Action

This bit defines the client's acknowledge behavior after an address or data byte is received from the host. The acknowledge action is executed when a command is written to the CMD bits. If Smart mode is enabled (CTRLB.SMEN='1'), the acknowledge action is performed when the DATA register is read.

Note: This bit is not enable-protected.

Value	Description
0	Send ACK
1	Send NACK

Bits 17:16 – CMD[1:0] Command

This bit field triggers the client operation as described below. The CMD bits are strobe bits, and always read as zero. The operation is dependent on the client interrupt flags, INTFLAG.DRDY and INTFLAG.AMATCH, in addition to STATUS.DIR.

All interrupt flags (INTFLAG.DRDY, INTFLAG.AMATCH and INTFLAG.PREC) are automatically cleared when a command is given.

Note: This bit is not enable-protected.

Table 31-4. 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 (Host write)	Execute acknowledge action succeeded by waiting for any start (S/Sr) condition
	1 (Host read)	Wait for any start (S/Sr) condition

Table 31-4. Command Description (continued)

CMD[1:0]	DIR	Action
0x3	Used in response to an address interrupt (AMATCH)	
	0 (Host write)	Execute acknowledge action succeeded by reception of next byte
	1 (Host read)	Execute acknowledge action succeeded by client data interrupt
	Used in response to a data interrupt (DRDY)	
	0 (Host write)	Execute acknowledge action succeeded by reception of next byte
	1 (Host 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.

Note: This bit field is enable-protected.

Value	Name	Description
0x0	MASK	The client responds to the address written in ADDR.ADDR masked by the value in ADDR.ADDRMASK. See SERCOM – Serial Communication Interface for additional information.
0x1	2ADDRS	The client responds to the two unique addresses in ADDR.ADDR and ADDR.ADDRMASK
0x2	RANGE	The client responds to the range of addresses between and including ADDR.ADDR and ADDR.ADDRMASK. ADDR.ADDR is the upper limit.

Bit 10 – AACKEN Automatic Acknowledge Enable

This bit enables the address to be automatically acknowledged if there is an address match.

Note: This bit is enable-protected.

Value	Description
0	Automatic acknowledge is disabled
1	Automatic acknowledge is enabled

Bit 9 – GCMD PMBus Group Command

This bit enables PMBus group command support. When enabled, the Stop Received interrupt flag (INTFLAG.PREC) will be set when a STOP condition is detected if the client has been addressed since the last STOP condition on the bus.

Note: This bit is enable-protected.

Value	Description
0	Group command is disabled.
1	Group command is enabled.

Bit 8 – SMEN Smart Mode Enable

When Smart mode is enabled, data is acknowledged automatically when DATA.DATA is read.

Note: This bit is enable-protected.

Value	Description
0	Smart mode is disabled
1	Smart mode is enabled

31.6.3. Interrupt Enable Clear - I2CS

Name: INTENCLR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection

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).

Bit	7	6	5	4	3	2	1	0
	ERROR					DRDY	AMATCH	PREC
Access	R/W					R/W	R/W	R/W
Reset	0					0	0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 2 – DRDY Data Ready Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Data Ready bit, which disables the Data Ready interrupt.

Value	Description
0	The Data Ready interrupt is disabled.
1	The Data Ready interrupt is enabled.

Bit 1 – AMATCH Address Match Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Address Match Interrupt Enable bit, which disables the Address Match interrupt.

Value	Description
0	The Address Match interrupt is disabled
1	The Address Match interrupt is enabled

Bit 0 – PREC Stop Received Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Stop Received Interrupt Enable bit, which disables the Stop Received interrupt.

Value	Description
0	The Stop Received interrupt is disabled
1	The Stop Received interrupt is enabled

31.6.4. Interrupt Enable Set - I2CS

Name: INTENSET
Offset: 0x16
Reset: 0x00
Property: PAC Write-Protection

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).

Bit	7	6	5	4	3	2	1	0
	ERROR					DRDY	AMATCH	PREC
Access	R/W					R/W	R/W	R/W
Reset	0					0	0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 2 – DRDY Data Ready Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Data Ready bit, which enables the Data Ready interrupt.

Value	Description
0	The Data Ready interrupt is disabled
1	The Data Ready interrupt is enabled

Bit 1 – AMATCH Address Match Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Address Match Interrupt Enable bit, which enables the Address Match interrupt.

Value	Description
0	The Address Match interrupt is disabled
1	The Address Match interrupt is enabled

Bit 0 – PREC Stop Received Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Stop Received Interrupt Enable bit, which enables the Stop Received interrupt.

Value	Description
0	The Stop Received interrupt is disabled
1	The Stop Received interrupt is enabled

31.6.5. Interrupt Flag Status and Clear - I2CS

Name: INTFLAG
Offset: 0x18
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
	ERROR					DRDY	AMATCH	PREC
Access	R/W					R/W	R/W	R/W
Reset	0					0	0	0

Bit 7 – ERROR Error
This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. The corresponding bits in STATUS are SEXTTOUT, LOWTOUT, COLL, and BUSERR.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

Bit 2 – DRDY Data Ready
This flag is set when a I^2C client byte transmission or reception 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 '0' to this bit has no effect.
Writing '1' to this bit will clear the Data Ready interrupt flag.

Bit 1 – AMATCH Address Match
This flag is set when the I^2C client address match logic detects that a valid address has been received.
The flag is cleared by hardware when CTRL.CMD is written.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Address Match interrupt flag. 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 host and another client will not set this flag, unless the PMBus Group Command is enabled in the Control B register (CTRLB.GCMD=1).
This flag is cleared by hardware after a command is issued on the next address match.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Stop Received interrupt flag.

31.6.6. Status - I2CS

Name: STATUS
Offset: 0x1A
Reset: 0x0000
Property: –

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

Bit 9 – SEXTTOUT Client SCL Low Extend Time-Out
This bit is set if a client SCL low extend time-out occurs.
This bit is cleared automatically if responding to a new start condition with ACK or NACK (write `0x3` to CTRLB.CMD).
This bit is not cleared when INTFLAG.AMATCH is cleared. Write to '1' to clear SEXTTOUT status.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the status.

Value	Description
0	No SCL low extend time-out has occurred
1	SCL low extend time-out has occurred

Bit 7 – CLKHOLD Clock Hold
The client Clock Hold bit (STATUS.CLKHOLD) is set when the client is holding the SCL line low, stretching the I^2C clock. Software has to consider this bit a read-only status flag that is set when INTFLAG.DRDY or INTFLAG.AMATCH is set. Do not clear the STATUS.CLKHOLD bit to preserve the current clock hold state.
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 `0x3` to CTRLB.CMD). This bit is not cleared when INTFLAG.AMATCH is cleared. Write to '1' to clear LOWTOUT status.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the status.

Value	Description
0	No SCL low time-out has occurred
1	SCL low time-out has occurred

Bit 4 – SR Repeated Start
When INTFLAG.AMATCH is raised due to an address match, SR indicates a repeated start or start condition.
This flag is only valid while the INTFLAG.AMATCH flag is one.

Value	Description
0	Start condition on last address match
1	Repeated start condition on last address match

Bit 3 – DIR Read / Write Direction

The Read/Write Direction (STATUS.DIR) bit stores the direction of the last address packet received from a host.

Value	Description
0	Host write operation is in progress
1	Host read operation is in progress

Bit 2 – RXNACK Received Not Acknowledge

This bit indicates whether the last data packet sent was acknowledged or not.

Value	Description
0	Host responded with ACK
1	Host responded with NACK

Bit 1 – COLL Transmit Collision

If set, the I^2C client was not able to transmit a high data or NACK bit, the I^2C client 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 it must be treated as a bus error.
Note that this status will not trigger any interrupt, and it must 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). This bit is not cleared when INTFLAG.AMATCH is cleared. Write to '1' to clear COLL status.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the status.

Value	Description
0	No collision detected on last data byte sent
1	Collision detected on last data byte sent

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). This bit is not cleared when INTFLAG.AMATCH is cleared. Write '1' to clear BUSERR status.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the status.

Value	Description
0	No bus error detected
1	Bus error detected

31.6.7. Synchronization Busy - I2CS

Name: SYNCBUSY
Offset: 0x1C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ENABLE SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

Value	Description
0	Enable synchronization is not busy
1	Enable synchronization is busy

Bit 0 – SWRST Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

Value	Description
0	SWRST synchronization is not busy
1	SWRST synchronization is busy

31.6.8. Address - 7-Bit Addressing mode - I2CS

Name: ADDR
Offset: 0x24
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

This register description is valid only in 7-Bit Addressing mode (`ADDR.TENBITEN=0`).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

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	
Reset	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8
	TENBITEN							
Access	R/W							
Reset	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 23:17 – ADDRMASK[6:0] Address Mask
These bits act as a second address match register, an address mask register or the lower limit of an address range, depending on the CTRLB.AMODE setting.

Bit 15 – TENBITEN Ten Bit Addressing Enable
This bit controls whether the 10-Bit Addressing mode is enabled.

Value	Description
0	10-Bit Addressing mode is disabled
1	10-Bit Addressing mode is enabled

Bits 7:1 – ADDR[6:0] Address
These bits contain the I^2C client address used by the client address match logic to determine if a host has addressed the client or the upper limit of an address range.
When using 7-bit addressing, the client address is represented by ADDR[6:0].
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
A general call address is an address consisting of all-zeroes, including the direction bit (host write).

Value	Description
0	General call address recognition is disabled
1	General call address recognition is enabled

31.6.9. Address - 10-Bit Addressing mode - I2CS

Name: ADDR
Offset: 0x24
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

This register description is valid only in 10-Bit Addressing mode (`ADDR.TENBITEN=1`).

Bit	31	30	29	28	27	26	25	24
						ADDRMASK[9:7]		
Access						R/W	R/W	R/W
Reset						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	
Reset	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8
	TENBITEN					ADDR[9:7]		
Access	R/W					R/W	R/W	R/W
Reset	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 26:17 – ADDRMASK[9:0] Address Mask
These bits act as a second address match register, an address mask register or the lower limit of an address range, depending on the CTRLB.AMODE setting.

Bit 15 – TENBITEN Ten Bit Addressing Enable
This bit controls whether the 10-Bit Addressing mode is enabled.

Value	Description
0	10-Bit Addressing mode is disabled
1	10-Bit Addressing mode is enabled

Bits 10:1 – ADDR[9:0] Address
These bits contain the I^2C client address used by the client address match logic to determine if a host has addressed the client or the upper limit of an address range.
When using 7-bit addressing, the client address is represented by ADDR[6:0].
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
A general call address is an address consisting of all-zeroes, including the direction bit (host write).

Value	Description
0	General call address recognition is disabled
1	General call address recognition is enabled

31.6.10. Data - I2CS

Name: DATA
Offset: 0x28
Reset: 0x00
Property: Write-Synchronized, Read-Synchronized

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 7:0 – DATA[7:0] Data

The client data register I/O location (DATA.DATA) provides access to the host 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 client (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.

31.7. Register Summary - SERCOMn.I2CM

SERCOM in I2C HOST Mode

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
		15:8								
		23:16	SEXTTOEN	MEXTTOEN	SDAHOLD[1:0]					PINOUT
		31:24		LOWTOUTEN	INACTOUT[1:0]		SCLSM		SPEED[1:0]	
0x04	CTRLB	7:0								
		15:8							QCEN	SMEN
		23:16						ACKACT	CMD[1:0]	
		31:24								
0x08 ... 0x0B	Reserved									
0x0C	BAUD	7:0	BAUD[7:0]							
		15:8	BAUDLOW[7:0]							
		23:16								
		31:24								
0x10 ... 0x13	Reserved									
0x14	INTENCLR	7:0	ERROR						SB	MB
0x15	Reserved									
0x16	INTENSET	7:0	ERROR						SB	MB
0x17	Reserved									
0x18	INTFLAG	7:0	ERROR						SB	MB
0x19	Reserved									
0x1A	STATUS	7:0	CLKHOLD	LOWTOUT	BUSSTATE[1:0]			RXNACK	ARBLOST	BUSERR
		15:8						LENERR	SEXTTOUT	MEXTTOUT
0x1C	SYNCBUSY	7:0						SYSOP	ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x20 ... 0x23	Reserved									
0x24	ADDR	7:0		ADDR[6:0]						
		15:8	TENBITEN		LENEN					
		23:16	LEN[7:0]							
		31:24								
0x24	ADDR	7:0	ADDR[7:0]							
		15:8	TENBITEN		LENEN				ADDR[9:8]	
		23:16	LEN[7:0]							
		31:24								
0x28	DATA	7:0	DATA[7:0]							
0x29 ... 0x2F	Reserved									
0x30	DBGCTRL	7:0								DBGSTOP

31.7.1. Control A - I2CM

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected, Write-Synchronized

Bit	31	30	29	28	27	26	25	24
		LOWTOUTEN	INACTOUT[1:0]		SCLSM		SPEED[1:0]	
Access		R/W	R/W	R/W	R/W		R/W	R/W
Reset		0	0	0	0		0	0

Bit	23	22	21	20	19	18	17	16
	SEXTTOEN	MEXTTOEN	SDAHOLD[1:0]					PINOUT
Access	R/W	R/W	R/W	R/W				R/W
Reset	0	0	0	0				0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY			MODE[2:0]			ENABLE	SWRST
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit 30 – LOWTOUTEN SCL Low Time-Out
This bit enables the SCL low time-out. If SCL is held low for 25–35 ms, the host 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.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Time-out is disabled
1	Time-out is enabled

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 arises when either an I^2C host or client is holding the SCL low.
Enabling this option is necessary for SMBus compatibility, but it can also be used in a non-SMBus set-up.
Calculated time-out periods are based on a 100 kHz bus clock.
Note: This bit is enable-protected. This bit is not synchronized.

Value	Name	Description
0x0	DIS	Disabled
0x1	55US	5-6 SCL cycle time-out (50-60 µs)
0x2	105US	10-11 SCL cycle time-out (100-110 µs)
0x3	205US	20-21 SCL cycle time-out (200-210 µs)

Bit 27 – SCLSM SCL Clock Stretch Mode

This bit controls when SCL will be stretched for software interaction.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	SCL stretch according to Figure 31-5
1	SCL stretch only after ACK bit, Figure 31-6

Bits 25:24 – SPEED[1:0] Transfer Speed

These bits define bus speed.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	SM_MODE	Standard-mode (Sm) and Fast-mode (Fm)
0x1	FASTPLUS_MODE	Fast-mode Plus (Fm+)
Other	—	Reserved

Bit 23 – SEXTTOEN Client SCL Low Extend Time-Out

This bit enables the client SCL low extend time-out. If SCL is cumulatively held low for greater than 25 ms from the initial START to a STOP, the host will release its clock hold if enabled, and complete the current transaction. A STOP will be transmitted automatically.
SB or MB will be set as normal, but CLKHOLD will be released. The MEXTTOUT and BUSERR status bits will also be set.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Time-out is disabled
1	Time-out is enabled

Bit 22 – MEXTTOEN Host SCL Low Extend Time-Out

This bit enables the host SCL low extend time-out. If SCL is cumulatively held low for greater than 10 ms from START-to-ACK, ACK-to-ACK, or ACK-to-STOP the host will release its clock hold if enabled, and complete the current transaction. A STOP will be transmitted automatically.
SB or MB will be set as normal, but CLKHOLD will be released. The MEXTTOUT and BUSERR status bits will also be set.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	Time-out is disabled
1	Time-out is enabled

Bits 21:20 – SDAHOLD[1:0] SDA Hold Time

These bits define the SDA hold time with respect to the negative edge of SCL.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	DIS	SDA hold time is disabled
0x1	75NS	SDA hold time is 50–100 ns
0x2	450NS	SDA hold time is 300–600 ns
0x3	600NS	SDA hold time is 400–800 ns

Bit 16 – PINOUT Pin Usage

This bit sets the pin usage to either two- or four-wire operation:

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	4-wire operation is disabled

Value	Description
1	4-wire operation is enabled

Bit 7 – RUNSTDBY Run in Standby

This bit defines the functionality in Standby Sleep mode.

Note: This bit is enable-protected. This bit is not synchronized.

Value	Description
0	GCLK_SERCOMx_CORE is disabled and the I²C host will not operate in Standby sleep mode
1	GCLK_SERCOMx_CORE is enabled in all sleep modes

Bits 4:2 – MODE[2:0] Operating Mode

This bit field controls the SERCOM mode.

Note: This bit field is enable-protected. This bit field is not synchronized.

Value	Name	Description
0x0	USART_EXT	USART with external clock
0x1	USART_INT	USART with internal clock
0x2	SPI_SLAVE	SPI Client mode
0x3	SPI_MASTER	SPI Host mode
0x4	I2C_SLAVE	I²C Client mode
0x5	I2C_MASTER	I²C Host mode

Bit 1 – ENABLE Enable

Notes:

1. This bit is write-synchronized: SYNCBUSY.ENABLE must be checked to ensure the CTRLA.ENABLE synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	The peripheral is disabled or being disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset

Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state and disables the SERCOM.

Writing '1' to CTRLA.SWRST always takes precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register during the reset will return the reset value of the register.

Notes:

1. This bit is write-synchronized: SYNCBUSY.SWRST must be checked to ensure the CTRLA.SWRST synchronization is complete.
2. This bit is not enable-protected.

Value	Description
0	There is no reset operation in progress
1	The reset operation is in progress

31.7.2. Control B - I2CM

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						ACKACT	CMD[1:0]	
Access						R/W	W	W
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
							QCEN	SMEN
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 18 – ACKACT Acknowledge Action

This bit defines the I^2C host's acknowledge behavior after a data byte is received from the I^2C client. 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.

Note: This bit is not enable-protected.

Value	Description
0	Send ACK
1	Send NACK

Bits 17:16 – CMD[1:0] Command

Writing to these bits triggers a host operation as described below. The CMD bits are strobe bits and always read as zero. The acknowledge action is only valid in Host Read mode. In Host 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 simultaneously, in which case the acknowledge action will be updated before the command is triggered.

Commands can only be issued when either the Client on Bus interrupt flag (INTFLAG.SB) or the Host on Bus interrupt flag (INTFLAG.MB) is '1'.

If CMD 0x1 is issued, a repeated start will be generated, 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 the transmission of the new address.

Issuing a command will set the System Operation bit in the Synchronization Busy register (SYNCBUSY.SYSOP).

Note: This bit field is not enable-protected.

Table 31-5. Command Description

CMD[1:0]	Direction	Action
0x0	X	(No action)
0x1	X	Execute acknowledge action succeeded by repeated Start
0x2	0 (Write)	No operation
	1 (Read)	Execute acknowledge action, followed by a byte read operation
0x3	X	Execute acknowledge action, followed by issuing a stop condition

Bit 9 – QCEN Quick Command Enable

Note: This bit is enable-protected.

Value	Description
0	Quick Command is disabled
1	Quick Command is enabled

Bit 8 – SMEN Smart Mode Enable

When Smart mode is enabled, acknowledge action is sent when DATA.DATA is read.

Note: This bit is enable-protected.

Value	Description
0	Smart mode is disabled
1	Smart mode is enabled

31.7.3. Baud Rate - I2CM

Name: BAUD
Offset: 0x0C
Reset: 0x00000000
Property: PAC Write-Protection, Enable-Protected

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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] Host Baud Rate Low

If this bit field is non-zero, the SCL low time will be described by the value written.
For more information on calculating the frequency, see the *Clock Generation – Baud-Rate Generator* section.

Bits 7:0 – BAUD[7:0] Host Baud Rate

This bit field 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 calculating the frequency, see the *Clock Generation – Baud-Rate Generator* section.

31.7.4. Interrupt Enable Clear - I2CM

Name: INTENCLR
Offset: 0x14
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
	ERROR						SB	MB
Access	R/W						R/W	R/W
Reset	0						0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 1 – SB Client on Bus Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Client on Bus Interrupt Enable bit, which disables the Client on Bus interrupt.

Value	Description
0	The Client on Bus interrupt is disabled
1	The Client on Bus interrupt is enabled

Bit 0 – MB Host on Bus Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the Host on Bus Interrupt Enable bit, which disables the Host on Bus interrupt.

Value	Description
0	The Host on Bus interrupt is disabled
1	The Host on Bus interrupt is enabled

31.7.5. Interrupt Enable Set - I2CM

Name: INTENSET
Offset: 0x16
Reset: 0x00
Property: PAC Write-Protection

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

Bit	7	6	5	4	3	2	1	0
	ERROR						SB	MB
Access	R/W						R/W	R/W
Reset	0						0	0

Bit 7 – ERROR Error Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 1 – SB Client on Bus Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Client on Bus Interrupt Enable bit, which enables the Client on Bus interrupt.

Value	Description
0	The Client on Bus interrupt is disabled
1	The Client on Bus interrupt is enabled

Bit 0 – MB Host on Bus Interrupt Enable
Writing '0' to this bit has no effect.
Writing '1' to this bit will set the Host on Bus Interrupt Enable bit, which enables the Host on Bus interrupt.

Value	Description
0	The Host on Bus interrupt is disabled
1	The Host on Bus interrupt is enabled

31.7.6. Interrupt Flag Status and Clear - I2CM

Name: INTFLAG
Offset: 0x18
Reset: 0x00
Property: –

Bit	7	6	5	4	3	2	1	0
	ERROR						SB	MB
Access	R/W						R/W	R/W
Reset	0						0	0

Bit 7 – ERROR Error

This flag is cleared by writing '1' to it.
This bit is set when any error is detected. Errors that set this flag have corresponding status bits in the STATUS register: LENERR, SEXTTOUT, MEXTTOUT, LOWTOUT, ARBLOST, and BUSERR.
Writing '0' to this bit has no effect.
Writing '1' to this bit will clear the flag.

Bit 1 – SB Client on Bus

The Client on Bus flag (SB) is set when a byte is successfully received in Host Read mode, meaning no arbitration loss or bus error occurred during the operation. When this flag is set, the host 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 '0' to this bit has no effect.
Writing '1' 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.

Bit 0 – MB Host on Bus

This flag is set when a byte is transmitted in Host Write mode, 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 Host Read mode, or when issuing a start condition if the bus state is unknown. When this flag is set and arbitration is not lost, the host 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

Writing '0' to this bit has no effect.
Writing '1' 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.

31.7.7. Status - I2CM

Name: STATUS
Offset: 0x1A
Reset: 0x0000
Property: Write-Synchronized

Bit	15	14	13	12	11	10	9	8
						LENERR	SEXTTOUT	MEXTTOUT
Access						R/W	R/W	R/W
Reset						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/W	R/W		R	R/W	R/W
Reset	0	0	0	0		0	0	0

Bit 10 – LENERR Transaction Length Error
This bit is set when automatic length is used for a DMA transaction and the client sends a NACK before all ADDR.LEN bytes have been written by the host.
Writing '0' to this bit has no effect.
Writing '1' to this bit location will clear STATUS.LENERR. This flag is automatically cleared when writing to the ADDR register.
Note: This bit is not synchronized.

Bit 9 – SEXTTOUT Client SCL Low Extend Time-Out
This bit is set if a client SCL low extend time-out occurs.
This bit is automatically cleared when writing to the ADDR register.
Writing '0' to this bit has no effect.
Writing '1' to this bit location will clear SEXTTOUT.
Note: This bit is not synchronized.

Bit 8 – MEXTTOUT Host SCL Low Extend Time-Out
This bit is set if a host SCL low time-out occurs.
Writing '0' to this bit has no effect.
Writing '1' to this bit location will clear STATUS.MEXTTOUT. This flag is automatically cleared when writing to the ADDR register.
Note: This bit is not synchronized.

Bit 7 – CLKHOLD Clock Hold
This bit is set when the host holds the SCL line low, stretching the I^2C clock. Software must consider this bit when INTFLAG.SB or INTFLAG.MB is set.
This bit is cleared when the corresponding interrupt flag is cleared and the next operation is performed.
Writing '0' to this bit has no effect.
Writing '1' to this bit has no effect.
Note: This bit is not synchronized.

Bit 6 – LOWTOUT SCL Low Time-Out
This bit is set if an SCL low time-out occurs.
Writing '0' to this bit has no effect.
Writing '1' to this bit location will clear this bit. This flag is automatically cleared when writing to the ADDR register.

Note: This bit is not synchronized.

Bits 5:4 – BUSSTATE[1:0] Bus State

These bits indicate the current I²C bus state.
When in UNKNOWN state, writing 0x1 to BUSSTATE forces the bus state into the IDLE state. The bus state cannot be forced into any other state.
Note: This bit field is write-synchronized: SYNCBUSY.SYSOP must be checked to ensure the STATUS.BUSSTATE synchronization is complete.

Value	Name	Description
0x0	UNKNOWN	The bus state is unknown to the I²C host, which will wait for a stop condition to be detected or for the bus to be forced into the IDLE state by software
0x1	IDLE	The bus state is waiting for a transaction to be initialized
0x2	OWNER	The I²C host is the current owner of the bus
0x3	BUSY	Some other I²C host owns the bus

Bit 2 – RXNACK Received Not Acknowledge

This bit indicates whether the last address or data packet sent was acknowledged or not.
Writing '0' to this bit has no effect.
Writing '1' to this bit has no effect.
Note: This bit is not synchronized.

Value	Description
0	Client responded with ACK
1	Client responded with NACK

Bit 1 – ARBLOST Arbitration Lost

This bit 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 Host 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 '0' to this bit has no effect.
Writing '1' to this bit will clear it.
Note: This bit is not synchronized.

Bit 0 – BUSERR Bus Error

This bit 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 I²C 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 I²C host 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 '0' to this bit has no effect.
Writing '1' to this bit will clear it.
Note: This bit is not synchronized.

31.7.8. Synchronization Busy - I2CM

Name: SYNCBUSY
Offset: 0x1C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						SYSOP	ENABLE	SWRST
Access						R	R	R
Reset						0	0	0

Bit 2 – SYSOP System Operation Synchronization Busy
Writing CTRLB.CMD, STATUS.BUSSTATE, ADDR, or DATA when the SERCOM is enabled requires synchronization. When written, the SYNCBUSY.SYSOP bit will be set until synchronization is complete.

Value	Description
0	System operation synchronization is not busy
1	System operation synchronization is busy

Bit 1 – ENABLE SERCOM Enable Synchronization Busy
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

Value	Description
0	Enable synchronization is not busy
1	Enable synchronization is busy

Bit 0 – SWRST Software Reset Synchronization Busy
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

Value	Description
0	SWRST synchronization is not busy
1	SWRST synchronization is busy

31.7.9. Address - 7-Bit Addressing mode - I2CM

Name: ADDR
Offset: 0x24
Reset: 0x00000000
Property: Write-Synchronized

This register description is valid only in 7-Bit Addressing mode (`ADDR.TENBITEN=0`).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LEN[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
	TENBITEN		LENEN					
Access	R/W		R/W					
Reset	0		0					

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

Bits 23:16 – LEN[7:0] Transaction Length
These bits define the transaction length of a DMA transaction from 0 to 255 bytes. The Transfer Length Enable (LENEN) bit must be written to '1' to use DMA.
Note: This bit field is not synchronized.

Bit 15 – TENBITEN Ten Bit Addressing Enable
This bit enables 10-bit addressing. This bit can be written simultaneously with ADDR to indicate a 10-bit or 7-bit address transmission.
Note: This bit is not synchronized.

Value	Description
0	10-bit addressing disabled
1	10-bit addressing enabled

Bit 13 – LENEN Transfer Length Enable
Note: This bit is not synchronized.

Value	Description
0	Automatic transfer length disabled
1	Automatic transfer length enabled

Bits 6:0 – ADDR[6:0] 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 host will await further operation until the bus becomes IDLE.

IDLE: The I^2C host 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. 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 host logic to perform any bus protocol related operations.
The I^2C host control logic uses bit 0 of ADDR as the bus protocol's read/write flag (R/W); '0' for write and '1' for read.
Note: This bit field is write-synchronized: SYNCBUSY.SYSOP must be checked to ensure the ADDR.ADDR synchronization is complete.

31.7.10. Address - 10-Bit Addressing mode - I2CM

Name: ADDR
Offset: 0x24
Reset: 0x00000000
Property: Write-Synchronized

This register description is valid only in 10-Bit Addressing mode (`ADDR.TENBITEN=1`).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LEN[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
	TENBITEN		LENEN				ADDR[9:8]	
Access	R/W		R/W				R/W	R/W
Reset	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 23:16 – LEN[7:0] Transaction Length
These bits define the transaction length of a DMA transaction from 0 to 255 bytes. The Transfer Length Enable (LENEN) bit must be written to '1' to use DMA.
Note: This bit field is not synchronized.

Bit 15 – TENBITEN Ten Bit Addressing Enable
This bit enables 10-bit addressing. This bit can be written simultaneously with ADDR to indicate a 10-bit or 7-bit address transmission.
Note: This bit is not synchronized.

Value	Description
0	10-bit addressing disabled
1	10-bit addressing enabled

Bit 13 – LENEN Transfer Length Enable
Note: This bit is not synchronized.

Value	Description
0	Automatic transfer length disabled
1	Automatic transfer length enabled

Bits 9:0 – ADDR[9:0] 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 host will await further operation until the bus becomes IDLE.

IDLE: The I^2C host 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. 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 host logic to perform any bus protocol related operations.
The I^2C host control logic uses bit 0 of ADDR as the bus protocol's read/write flag (R/W); '0' for write and '1' for read.
Note: This bit field is write-synchronized: SYNCBUSY.SYSOP must be checked to ensure the ADDR.ADDR synchronization is complete.

31.7.11. Data - I2CM

Name: DATA
Offset: 0x28
Reset: 0x00
Property: Write-Synchronized

Note: This register is write-synchronized: SYNCBUSY.SYSOP must be checked to ensure the DATA register synchronization is complete.

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 7:0 – DATA[7:0] Data

The host data register I/O location (DATA) provides access to the host 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 host (STATUS.CLKHOLD is set). An exception is 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.

31.7.12. Debug Control - I2CM

Name: DBGCTRL
Offset: 0x30
Reset: 0x00
Property: PAC Write-Protection

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

Bit 0 – DBGSTOP Debug Stop Mode
This bit controls functionality when the CPU is halted by an external debugger.
This bit will be cleared after a software system reset.

Value	Description
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

32. CCL - Configurable Custom Logic

32.1. Features

- Glue Logic for General-Purpose System Design
- Four Programmable Look-up Tables (LUTs)
- Combinatorial Logic Functions:
  AND, NAND, OR, NOR, XOR, XNOR, NOT
- Sequential Logic Functions:
  Gated D Flip-Flop, JK Flip-Flop, Gated D Latch, RS Latch
- Flexible LUT Input Selection:
  - I/Os
  - Events
  - Internal peripherals
  - Subsequent LUT output
- Output Can Be Connected to the I/O Pins or the Event System
- Optional Synchronizer, Filter, or Edge Detector Available on Each LUT Output

32.2. Overview

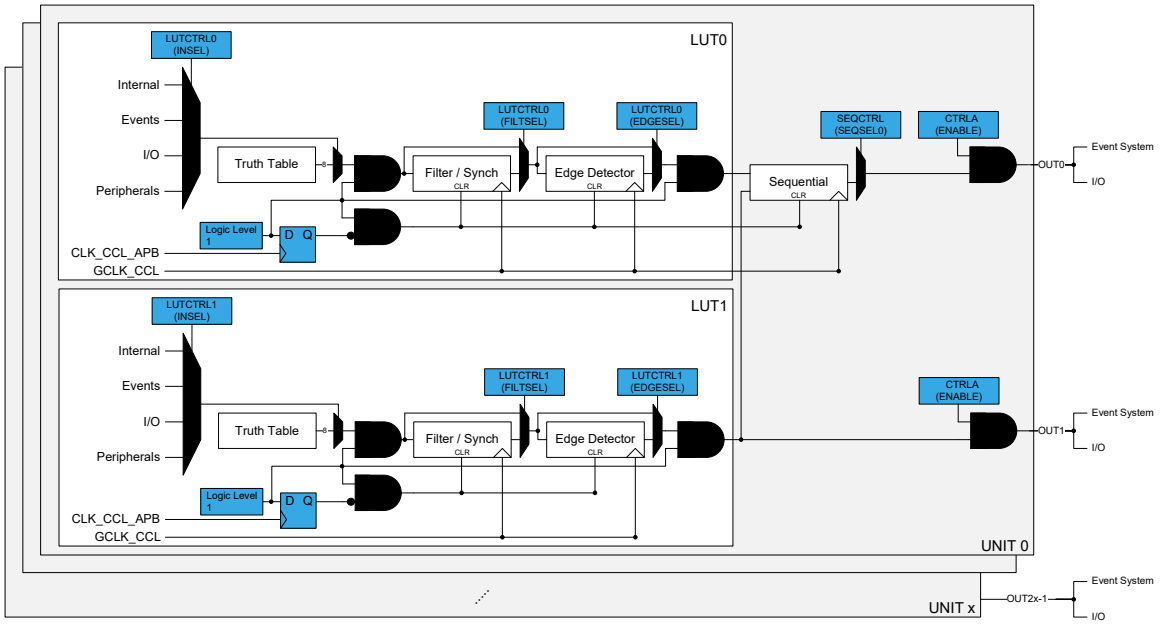
The Configurable Custom Logic (CCL) is a programmable logic peripheral that can be connected to the device pins, events, or other internal peripherals. This allows users to eliminate hardware logic gates for simple glue logic functions in system design.

Each look-up table (LUT) consists of three inputs and supports a user-defined truth table. Additionally, each LUT includes an optional synchronizer or filter, and an optional edge detector. Each LUT can generate an output based on a user-programmable logic expression with three inputs, and each input can be individually masked.

The output can be generated combinatorially from the inputs and can be filtered to remove spikes. Optional sequential logic can also be applied. The inputs to the sequential module are individually controlled by two independent, adjacent LUT outputs (e.g., LUT0/LUT1, LUT2/LUT3, etc.), enabling complex waveform generation.

32.3. Block Diagram

Figure 32-1. Configurable Custom Logic



32.3.1. Signal Description

Pin Name	Type	Description
LUTn-OUT	Digital output	Output from look-up table
LUTn-IN[2:0]	Digital input	Input to look-up table

Refer to the *Pinout* chapter for detailed information on the pin mapping for this peripheral. A single signal can be mapped to multiple pins.

32.4. Functional Description

32.4.1. Initialization

1. Enable the CCL bus clock in the Main Clock (MCLK)
2. Configure inputs to the LUTs (LUTCTRLn.INSELn)
3. Configure the Truth Table (LUTCTRLn.TRUTH)
4. Enable the CCL (CTRLA.ENABLE)

32.4.2. Operation

The Configurable Custom Logic (CCL) is a programmable logic block that can use the device port pins, internal peripherals, and the internal Event System as both input and output channels. The CCL can serve as glue logic between the device and external components, eliminating the need for additional hardware logic. Additionally, the CCL enables designers to overcome challenging real-time constraints by combining core independent peripherals in innovative ways to handle the most time-critical parts of an application, independent of the CPU.

32.4.2.1. Write Protection Control

CCL registers can be write-protected by writing a '1' to the Write Protection Enable (WPCTRL.WPEN) bit in the WPCTRL register and simultaneously writing the specific KEY value to the WPKEY bit field. For more information, refer to the description of the WPCTRL register.

32.4.2.2. Lookup Table Logic

The look-up table in each LUT unit can generate any logic expression at the output (OUT) as a function of three inputs (IN[2:0]), as illustrated in the figure below. One or more inputs can be masked. The truth table for the expression is defined by the Truth Table (LUTCTRL[n].TRUTH) bit field in the LUT Control n register.

Figure 32-2. Truth Table Output Value Selection

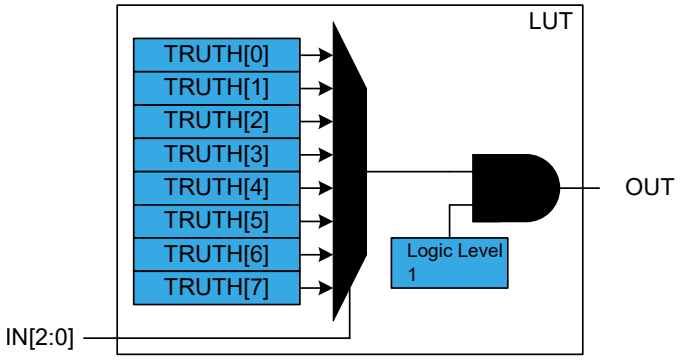


Table 32-1. Truth Table of LUT

IN[2]	IN[1]	IN[0]	OUT
0	0	0	TRUTH[0]
0	0	1	TRUTH[1]
0	1	0	TRUTH[2]
0	1	1	TRUTH[3]
1	0	0	TRUTH[4]
1	0	1	TRUTH[5]
1	1	0	TRUTH[6]
1	1	1	TRUTH[7]

32.4.2.3. Truth Table Inputs Selection

Input Overview

The inputs can be individually:

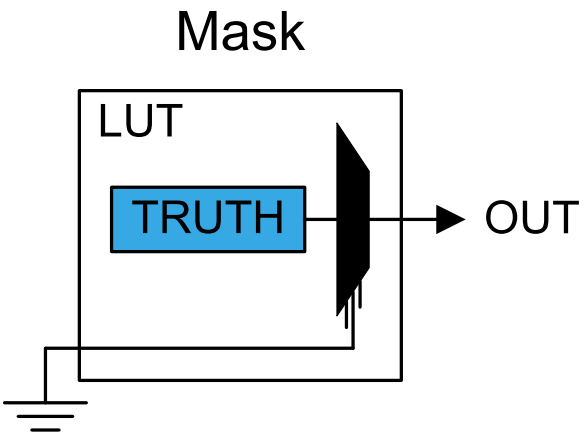
- Masked
- Driven by peripherals:
  - Analog Comparator output (AC)
  - Timer/Counters waveform outputs (TC)
  - Serial Communication output transmit interface (SERCOM)
  - Timer/Counters for Control Applications waveform outputs (TCC)
- Driven by internal events from Event System
- Driven by other CCL signals

The input selection for each input x to LUT n is configured by writing the Input x Source Selection (LUTCTRL[n].INSELx) bit field in the LUT n Control register.

Masked Inputs (MASK)

When a LUT input is masked (LUTCTRL[n].INSELx=MASK), the corresponding TRUTH input (IN) is internally tied to zero, as illustrated in this figure:

Figure 32-3. Masked Input Selection



Internal Feedback Inputs (FEEDBACK)

When feedback is selected (LUTCTRL[n].INSELx=FEEDBACK), the Sequential (SEQ) output is used as input to the corresponding LUT.

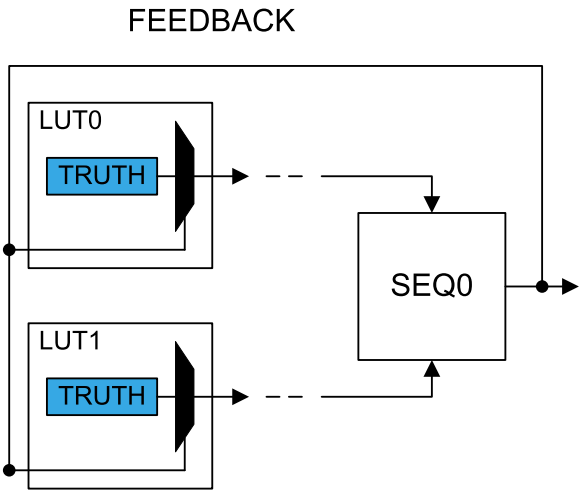
The output from an internal sequential sub-module can be used as input to the LUT; see the figure below for an example of LUT0 and LUT1. The sequential selection for each LUT follows the formula below:

$IN[2n][x] = \mathrm{SEQ}[n]$

$IN[2n+1][x] = \mathrm{SEQ}[n]$

In the formula above, *n* represents the sequencer number, and *x* (0,1,2) represents the LUT input index.

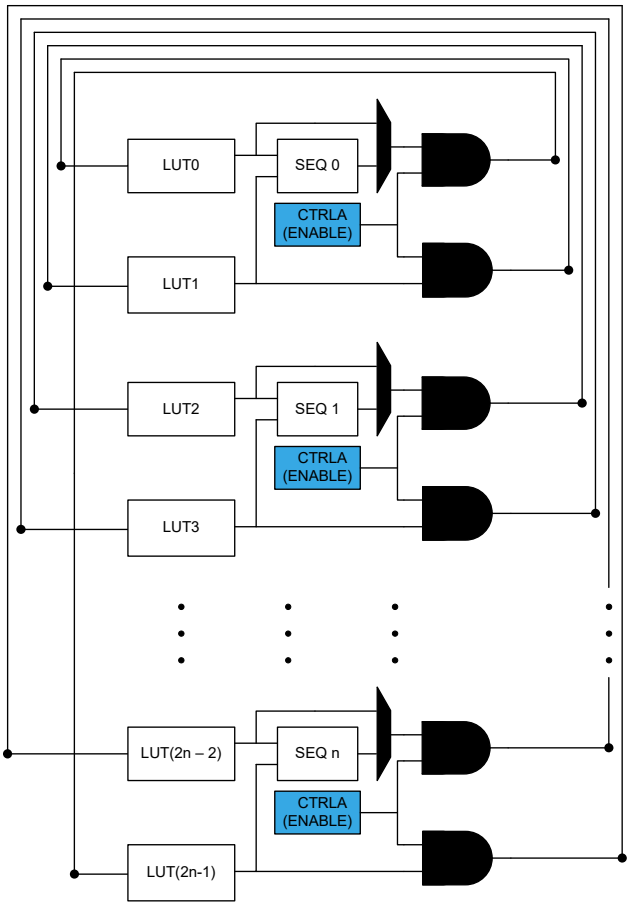
Figure 32-4. Feedback Input Selection



Linked LUT (LINK)

When linking is selected (LUTCTRL[n].INSELx=LINK), the output of the subsequent LUT is used as the input to the current LUT (e.g., LUT2 serves as the input to LUT1), as illustrated in the figure below.

Figure 32-5. Linked LUT Input Selection



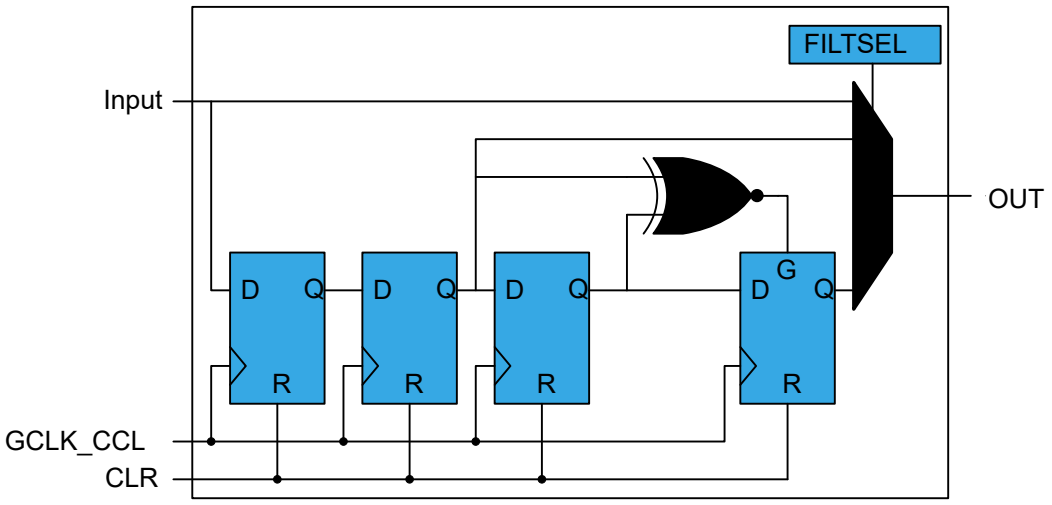
Internal Events Inputs Selection (EVENT)

Asynchronous events from the Event System can be used as input selections, as shown in the following figure. For each LUT, one event input line is available and can be selected on each LUT input. Before enabling event selection by writing LUTCTRL[n].INSELx = EVENT, the Event System must be configured.

By default, the CCL includes an edge detector. When the event is received, an internal strobe is generated upon detection of a rising edge, with a pulse duration of one GCLK_CCL clock cycle. Writing LUTCTRL[n].INSELx = ASYNCEVENT disables the edge detector. In this configuration, it is possible to combine an asynchronous event input with any other input source, which is particularly useful for event levels inputs (such as external I/O pin events). To ensure proper operation, follow these steps:

1. Enable the GCLK_CCL clock.
2. Configure the Event System to route the event asynchronously.
3. Select the event input type by configuring LUTCTRL[n].INSEL=ASYNCEVENT.
4. If a strobe must be generated on the event input falling edge, write a '1' to the Inverted Event Input Enable (LUTCTRL[n].INVEI) bit in the LUT Control register.
5. Enable the event input by writing a '1' to the Event Input Enable (LUTCTRL[n].LUTEI) bit in the LUT Control register.

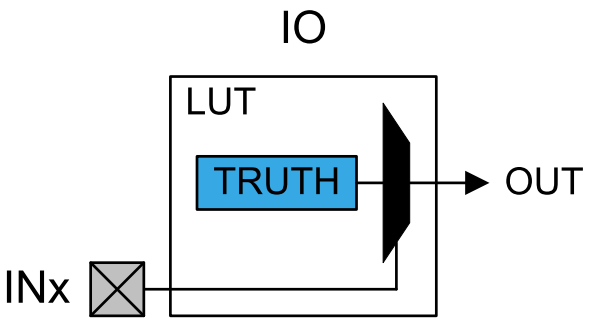
Figure 32-6. Event Input Selection



I/O Pin Inputs (IO)

When an I/O pin is selected as a LUT input (LUTCTRL[n].INSELx = IO), the corresponding LUT input will be connected to the pin, as illustrated in the figure below.

Figure 32-7. I/O Pin Input Selection



Analog Comparator Inputs (AC)

The AC outputs can be used as inputs to the LUTs by configuring LUTCTRL[n].INSELx = AC.

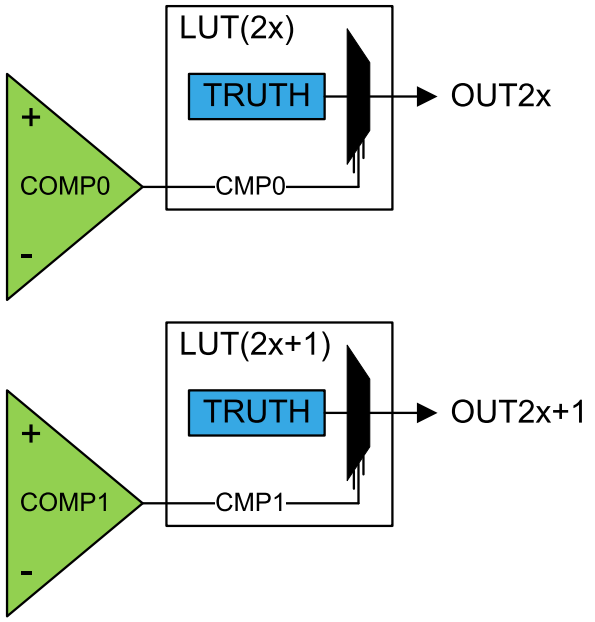
The analog comparator outputs are distributed according to the formula below:

```
IN[n][x]= AC[n % ComparatorOutput_Number]
```

In the formula above, *n* represents the LUT number and *x* (0,1,2) represents the LUT input index.

Before selecting the comparator output, the AC must be configured.

Figure 32-8. AC Input Selection



Timer/Counter Inputs (TC)

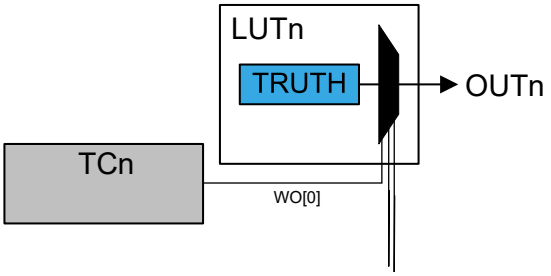
The WO[0] waveform outputs from the Timer/Counters can be used as inputs to the LUTs by configuring LUTCTRL[n].INSELx=TC. Specifically, TC0.WO[0], TC1.WO[0], and TC2.WO[0] are available as inputs to LUT0, LUT1, and LUT2, respectively. Additionally, TC0.WO[0] is available as an input to LUT3. See the figure below for an example of LUTn. More generally, the Timer/Counter selection for each LUT follows the formula below:

$$\mathrm{IN}[n][x] = TC[\mathrm{n} \,\%\, \mathrm{Number\_of\_TC\_Instances}].WO[0]$$

In the formula above, *n* represents the LUT number and *x* (0,1,2) represents the LUT input index.

Before selecting the waveform outputs, the Timer Counter (TC) must be configured.

Figure 32-9. TC Input Selection



Timer/Counter for Control Application Inputs (TCC)

The TCC waveform outputs can be used as inputs to the LUTs. Only WO[2:0] outputs can be selected and routed to the respective LUT input (i.e., IN0 is connected to WO0, IN1 to WO1, and IN2 to WO2), as illustrated in the figure below.

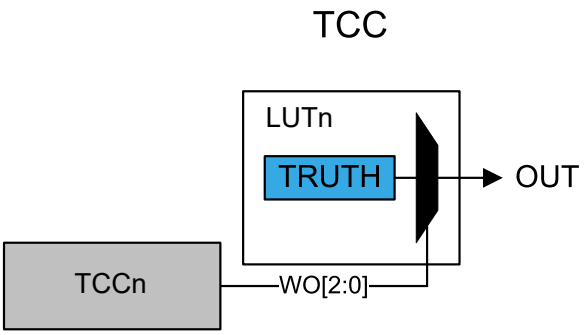
Note:
The TCC selection for each LUT follows the formula below:

$$\mathrm{IN}[n][x] = TCC[\mathrm{n} \,\%\, \mathrm{TCC\_Instance\_Number}].WO[x]$$

In the formula above, *n* represents the LUT number and *x* (0,1,2) represents the LUT input index.

Before selecting the waveform outputs, the TCC must be configured.

Figure 32-10. TCC Input Selection



Serial Communication Output Transmit Inputs (SERCOM)

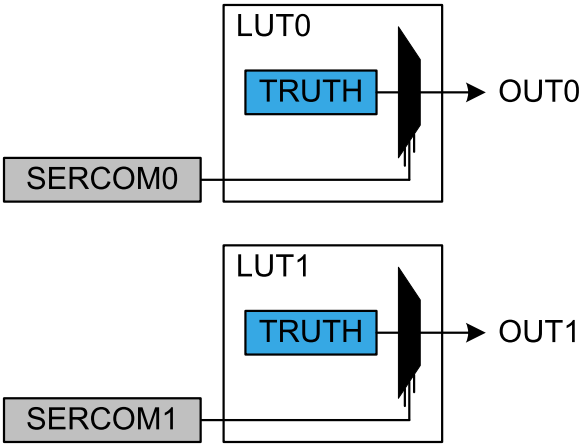
The serial engine transmitter output from Serial Communication Interface (SERCOM TX – TXd for USART, MOSI for SPI) can be used as an input to the LUT. The figure below shows an example for LUT0 and LUT1. The SERCOM selection for each LUT follows the formula below:

$\mathrm{IN}[n][x] = SERCOM[n \,\%\, \mathrm{SERCOM\_Instance\_Number}]$

In the formula above, *n* represents the LUT number and *x* (0,1,2) represents the LUT input index.

Before selecting SERCOM as input, the SERCOM must be configured. The SERCOM TX signal must be output on SERCOMn/pad[0], which serves as the input pad to the CCL.

Figure 32-11. SERCOM Input Selection



32.4.2.3.1. Input Selection Multiplexer

The following peripheral outputs are available as inputs into the CCL.

Table 32-2. CCL Input Selection Mux

Value	Input Source	LUT3			LUT2			LUT1			LUT0		
		INSEL2	INSEL1	INSEL0	INSEL2	INSEL1	INSEL0	INSEL2	INSEL1	INSEL0	INSEL2	INSEL1	INSEL0
0x00	MASK	None			None			None			None		
0x01	FEEDBACK	LUT3			LUT2			LUT1			LUT0		
0x02	LINK	LUT0			LUT3			LUT2			LUT1		
0x03	EVENT	LUTIN3			LUTIN2			LUTIN1			LUTIN0		
0x04	IO	LUT3-IN2	LUT3-IN1	LUT3-IN0	LUT2-IN2	LUT2-IN1	LUT2-IN0	LUT1-IN2	LUT1-IN1	LUT1-IN0	LUT0-IN2	LUT0-IN1	LUT0-IN0

Table 32-2. CCL Input Selection Mux (continued)

Value	Input Source	LUT3			LUT2			LUT1			LUT0		
		INSEL2	INSEL1	INSEL0	INSEL2	INSEL1	INSEL0	INSEL2	INSEL1	INSEL0	INSEL2	INSEL1	INSEL0
0x05	AC	CMP1			CMP0			CMP1			CMP0		
0x06	TC	TC0-WO0			TC2-WO0			TC1-WO0			TC0-WO0		
0x07	-	—											
0x08	TCC	TCC0-WO2	TCC0-WO1	TCC0-WO0	TCC0-WO2	TCC0-WO1	TCC0-WO0	TCC0-WO2	TCC0-WO1	TCC0-WO0	TCC0-WO2	TCC0-WO1	TCC0-WO0
0x09	SERCOM	SERCOM1-PAD0			SERCOM0-PAD0			SERCOM1-PAD0			SERCOM0-PAD0		
0x0A	-	—											
0x0B	ASYNCEVENT	LUTIN3			LUTIN2			LUTIN1			LUTIN0		

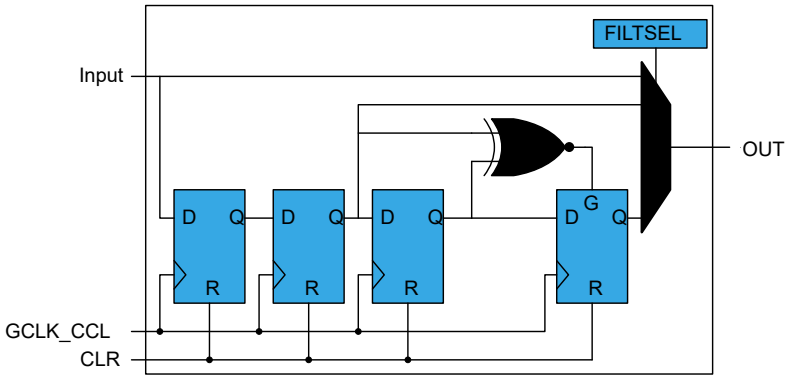
32.4.2.4. Filter

By default, the LUT output is a combinatorial function of the LUT inputs. This may result in some short glitches when the inputs change value. These glitches can be eliminated by clocking the output through filters, if required by the application.

The Filter Selection (LUTCTRL[n].FILTSEL) bits in the LUT Control register define the available synchronizer or digital filter options. When a filter is enabled, the OUT output will be delayed by two to five GCLK cycles. One APB clock cycle after the corresponding LUT is disabled, all internal filter logic is cleared.

Note: Events used as LUT inputs will also be filtered if the filter is enabled.

Figure 32-12. Filter



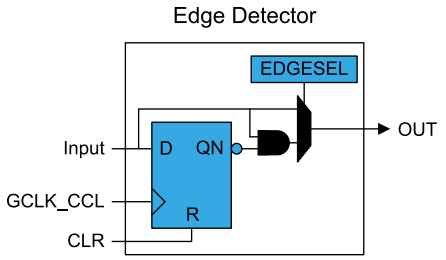
32.4.2.5. Edge Detector

The edge detector can be used to generate a pulse when a rising edge is detected on its input. To detect a falling edge, the TRUTH table has to be inverted.

The edge detector is enabled by writing a '1' to the Edge Selection (LUTCTRL[n].EDGESEL) bit in the LUT Control register. To avoid unpredictable behavior, either the filter or synchronizer must be enabled.

Edge detection is disabled by writing a '0' to LUTCTRL[n].EDGESEL. After an LUT is disabled, the corresponding internal edge detector logic is cleared one APB clock cycle later.

Figure 32-13. Edge Detector



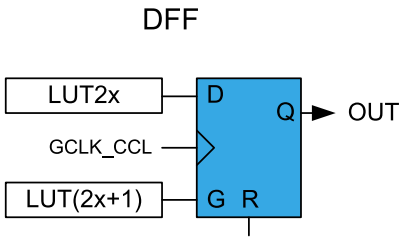
32.4.2.6. Sequential Logic

Each LUT pair can be connected to the internal sequential logic, which can be configured to operate as a D flip-flop, JK flip-flop, gated D latch, or RS latch by configuring the Sequential Selection bits in the corresponding Sequential Control (SEQCTRL) register. Before using sequential logic, the GCLK_CCL clock must be enabled, and, if required, each LUT filter or edge detector should also be enabled.

Gated D Flip-Flop (DFF)

When D flip-flop (DFF) mode is selected, the D input is driven by the even LUT output (LUT0 or LUT2), and the G input is driven by the odd LUT output (LUT1 or LUT3), as illustrated in the figure below.

Figure 32-14. D Flip-Flop



The flip-flop output (OUT) is updated on the rising edge of the GCLK_CCL, as shown in the table below.

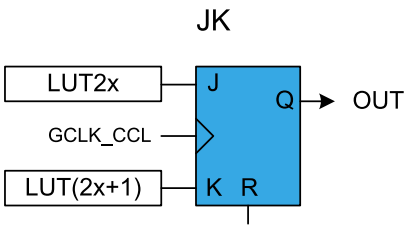
Table 32-3. DFF Characteristics

R	G	D	OUT
1	X	X	Clear
0	1	1	Set
		0	Clear
	0	X	Hold state (no change)

JK Flip-Flop (JK)

When this configuration is selected, the J input is driven by the even LUT output (LUT0 or LUT2), and the K input is driven by the odd LUT output (LUT1 or LUT3), as illustrated in the figure below.

Figure 32-15. JK Flip Flop



The flip-flop output (OUT) is updated on the rising edge of GCLK_CCL, as shown in the table below.

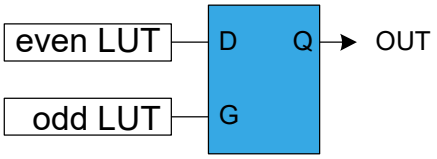
Table 32-4. JK Characteristics

R	J	K	OUT
1	X	X	Clear
0	0	0	Hold state (no change)
0	0	1	Clear
0	1	0	Set
0	1	1	Toggle

Gated D Latch (DLATCH)

When DLATCH is selected, the D input is driven by the even LUT output (LUT0 or LUT2), and the G input is driven by the odd LUT output (LUT1 or LUT3), as illustrated in the figure below.

Figure 32-16. D Latch



The latch output (OUT) is updated as shown in the table below.

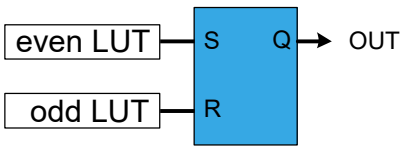
Table 32-5. D Latch Characteristics

G	D	OUT
0	X	Hold state (no change)
1	0	Clear
1	1	Set

RS Latch (RS)

When this configuration is selected, the S input is driven by the even LUT output (LUT0 or LUT2), and the R input is driven by the odd LUT output (LUT1 or LUT3), as illustrated in the figure below.

Figure 32-17. RS Latch



The latch output (OUT) is updated as shown in the table below.

Table 32-6. RS Latch Characteristics

S	R	OUT
0	0	Hold state (no change)
0	1	Clear
1	0	Set
1	1	Forbidden state

32.4.3. Sleep Mode Operation

When using the GCLK_CCL internal clocking, writing a '1' to the Run In Standby (CTRLA.RUNSTDBY) bit in the Control A register will allow GCLK_CCL to remain enabled in Standby sleep mode.

If CTRLA.RUNSTDBY=0, the GCLK_CCL will be disabled in Standby sleep mode. If the filter, edge detector or sequential logic are enabled, the LUT output will be forced to zero in Standby mode. In all other cases, the TRUTH table decoder will continue to operate, and the LUT output will be updated accordingly.

For further information, refer to the *PM - Power Manager* chapter.

32.4.4. Debug Operation

When the CPU is halted in debug mode the CCL continues normal operation.

32.5. Dependencies

32.5.1. I/O Lines

Using the CCL's I/O lines requires the I/O pins to be properly configured. Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

32.5.2. Power Management

The CCL will continue to operate in any sleep mode as long as its source clock is running. Events connected to the event system can trigger other operations within the system without exiting sleep modes. For details on the different sleep modes, refer to the *PM - Power Manager* chapter.

32.5.3. Clocks

The CCL bus clock (CLK_CCL_APB) can be enabled and disabled in the Main Clock Controller. The default state of CLK_CCL_APB is specified in the *Peripheral Clock Masking* section in the *MCLK – Main Clock Controller* chapter.

A generic clock (GCLK_CCL) may be required to operate the CCL, depending on which features are used. This clock must be configured and enabled in the Generic Clock Controller before using input events, filter, edge detection, or sequential logic. For more information, refer to the *GCLK – Generic Clock Controller* chapter.

32.5.4. DMA

Not applicable.

32.5.5. Interrupts

Not applicable.

32.5.6. Events

The CCL can generate the following events:

Table 32-7. Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
CCL	LUTOUT_n	Look-up Table Output Value	Level	Asynchronous	As long as OUT is '1'

Writing a '1' to an Event Output Enable bit in the LUT Control n (LUTCTRLn.LUTEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The CCL can connect to the following events:

Table 32-8. Event Users

User Name		Description	Input Detection	Channel Path Type
Peripheral	Input			
CCL	LUTIN_n	Look-up Table Input Value	Level	Asynchronous

Writing a '1' to an Event Input Enable bit in the LUT Control n (LUTCTRLn.LUTEO) enables the corresponding action on input event. Writing a '0' to this bit disables the corresponding action on input event.

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

32.5.7. Analog Connections

Not applicable.

32.6. Register Summary - CCL

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		RUNSTDBY					ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x04	SEQCTRL	7:0					SEQSEL0[3:0]			
		15:8					SEQSEL1[3:0]			
		23:16								
		31:24								
0x08	LUTCTRL[0]	7:0	EDGESEL		FILTSEL[1:0]					
		15:8	INSEL1[3:0]				INSEL0[3:0]			
		23:16		LUTEO	LUTEI	LUTINV	INSEL2[3:0]			
		31:24	TRUTH[7:0]							
0x0C	LUTCTRL[1]	7:0	EDGESEL		FILTSEL[1:0]					
		15:8	INSEL1[3:0]				INSEL0[3:0]			
		23:16		LUTEO	LUTEI	LUTINV	INSEL2[3:0]			
		31:24	TRUTH[7:0]							
0x10	LUTCTRL[2]	7:0	EDGESEL		FILTSEL[1:0]					
		15:8	INSEL1[3:0]				INSEL0[3:0]			
		23:16		LUTEO	LUTEI	LUTINV	INSEL2[3:0]			
		31:24	TRUTH[7:0]							
0x14	LUTCTRL[3]	7:0	EDGESEL		FILTSEL[1:0]					
		15:8	INSEL1[3:0]				INSEL0[3:0]			
		23:16		LUTEO	LUTEI	LUTINV	INSEL2[3:0]			
		31:24	TRUTH[7:0]							
0x18 ... 0x27	Reserved									
0x28	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

32.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: Local Write-Protection

Note: The RUNSTDBY bit in this register is enable-protected when CCL.CTRLA.ENABLE = 1.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 6 – RUNSTDBY Run in Standby
This bit indicates whether the GCLK_CCL clock must be kept running in Standby mode. This setting is ignored for configurations where the generic clock is not required. For details, refer to the *Sleep Mode Operation* section.

Important: This bit must be written before enabling the CCL.

Value	Description
0	A generic clock is not required in Standby sleep mode
1	A generic clock is required in Standby sleep mode

Bit 1 – ENABLE Enable

Value	Description
0	The peripheral is disabled
1	The peripheral is enabled

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the CCL to their initial state.

Value	Description
0	There is no reset operation in progress
1	The reset operation is in progress

32.6.2. Sequential Control

Name: SEQCTRL
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection, Enable-Protected

Note: The SEQCTRL register is enable-protected when CCL.CTRLA.ENABLE =1.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					SEQSEL1[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
					SEQSEL0[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 0:3, 8:11 – SEQSELx Sequential Selection x
These bit fields select the sequential configuration:

Value	Name	Description
0x0	DISABLE	Sequential logic is disabled
0x1	DFF	D flip-flop
0x2	JK	JK flip-flop
0x3	LATCH	D latch
0x4	RS	RS latch
0x5 - 0xF	—	Reserved

32.6.3. LUT Control n

Name: LUTCTRL[n]
Offset: 0x08 + n*0x04 [n=0..3]
Reset: 0x00000000
Property: Local Write-Protection, Enable-Protected

Note: The LUTCTRL[n] register is enable-protected when CCL.CTRLA.ENABLE=1.

Bit	31	30	29	28	27	26	25	24
	TRUTH[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	23	22	21	20	19	18	17	16
		LUTEO	LUTEI	LUTINV	INSEL2[3:0]			
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	INSEL1[3:0]				INSEL0[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	7	6	5	4	3	2	1	0
	EDGESEL		FILTSEL[1:0]					
Access	R/W		R/W	R/W				
Reset	0		0	0				

Bits 31:24 – TRUTH[7:0] Truth Table
This bit field defines the output value of the truth logic as a function of the inputs IN[2:0].

Bit 22 – LUTEO LUT Event Output Enable

Value	Description
0	LUT event output is disabled
1	LUT event output is enabled

Bit 21 – LUTEI LUT Event Input Enable

Value	Description
0	LUT incoming event is disabled
1	LUT incoming event is enabled

Bit 20 – LUTINV LUT Inverted Event Input Enable

Value	Description
0	Incoming event is not inverted
1	Incoming event is inverted

Bits 8:11, 12:15, 16:19 – INSELn LUT Input n Source Selection
These bit fields select the source for LUT input n. Refer to Input Selection Multiplexer to see the specific input signals that can be connected.

Value	Name	Description
0x0	MASK	Masked input

Value	Name	Description
0x1	FEEDBACK	Feedback input source
0x2	LINK	Linked LUT input source
0x3	EVENT	Event input source
0x4	IO	I/O pin input source
0x5	AC	AC input source
0x6	TC	TC input source
0x8	TCC	TCC input source
0x9	SERCOM	SERCOM input source
0xB	ASYNCEVENT	Asynchronous event input source
Other	—	Reserved

Bit 7 – EDGESEL Edge Selection

Value	Description
0	Edge detector is disabled
1	Edge detector is enabled

Bits 5:4 – FILTSEL[1:0] Filter Selection
This bit field selects the LUT output filter options:

Value	Name	Description
0x0	DISABLE	Filter disabled
0x1	SYNCH	Synchronizer enabled
0x2	FILTER	Filter enabled
0x3	—	Reserved

32.6.4. Write Protection Control

Name: WPCTRL
Offset: 0x28
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write operation to be successful.

Value	Name	Description
0x43434C	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	CCL register write protection is disabled
1	Write protection is enabled on CCL registers with the Local Write-Protection property. Non-debugger writes to CCL registers with Local Write-Protection are ignored and will generate a bus error.

33. ADC - Analog-to-Digital Converter

33.1. Features

- 12-bit Resolution
  - 12-bit, 10-bit and 8-bit resolution supported
  - 13-bit mode with reference range doubling
  - Up to 18-bit Effective Number of Bits (ENOB) with oversampling
- Conversion Rate Up to 800 ksps at 12 and 13-bit Resolution
- Supports Up to 29 Inputs
- Single-Ended and Differential Conversion
- Multiple Internal ADC Reference Voltages
- External Reference Input
- Single and Free-Running Conversions
- Series and Burst Accumulation Modes
  - Accumulates up to 1024 conversions
  - Left- or right-adjusted result
  - Offset Reduction using Sign Chopping
  - Averaged result
  - Low-pass filter
- Interrupts on Conversion Complete
- Event-Triggered Conversion
- DMA Transfer of Result and Sample
- Configurable Window Comparator

33.2. Overview

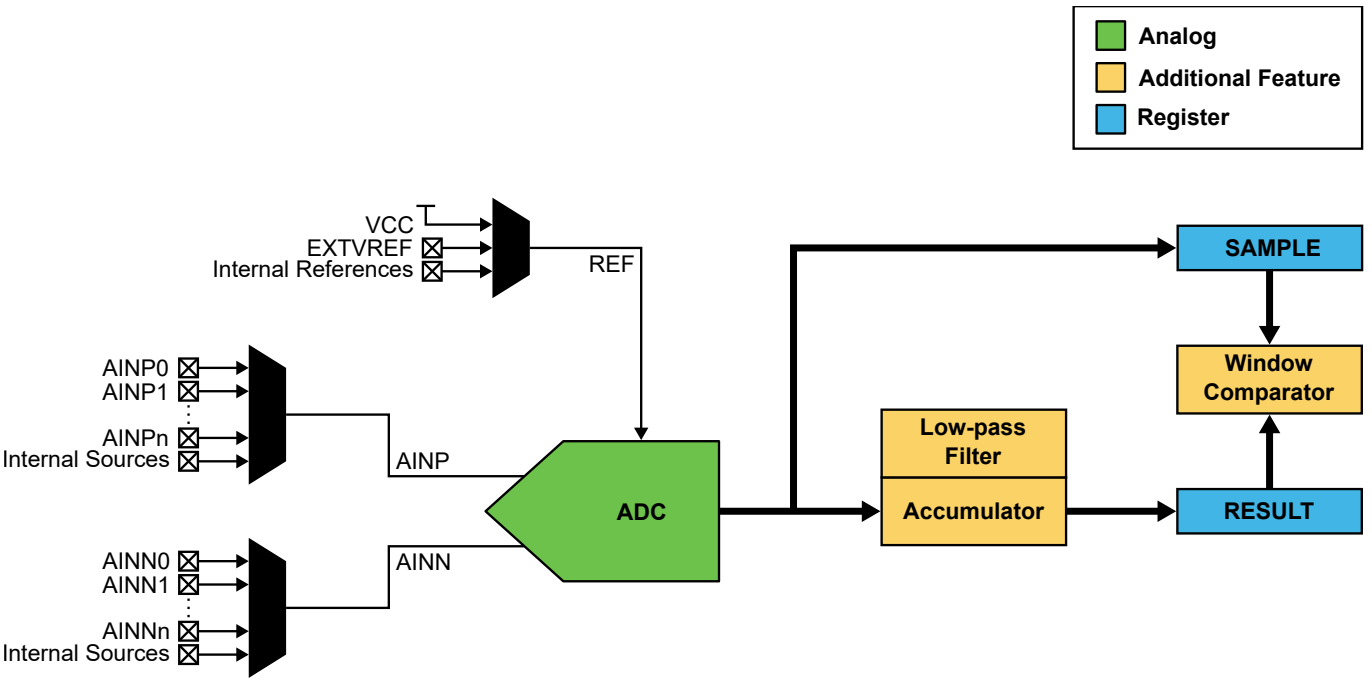
The Analog-to-Digital Converter (ADC) peripheral is a 12-bit differential and single-ended ADC, with a conversion rate up to 800 ksps at 12-bit resolution.

The ADC supports multiple analog input channels, selectable for single-ended or differential measurements. Internal and external reference voltage options are available, providing flexibility for various application requirements.

The ADC offers features such as accumulation modes for averaging or filtering results, a digital window comparator for threshold monitoring, and support for event-triggered conversions. These capabilities enable efficient and flexible analog signal measurement in a wide range of applications.

33.3. Block Diagram

Figure 33-1. Block Diagram



33.3.1. Signal Description

Signal	Type	Description
AINP	Analog input	Positive analog input
AINN	Analog input	Negative analog input
REF	Analog input	Reference voltage

For the mapping between the ADC signals and I/O pins, refer to the *Pinout and Multiplexing* section in the *Pinout* chapter. The External Voltage Reference (EXTVREF) pin for this device is named VREFA in this table.

33.4. Functional Description

33.4.1. Initialization

To initialize and run the ADC in basic single-ended mode, follow these steps:

1. Configure the Timebase bit field in the Control B register (CTRLB.TIMEBASE).
2. Configure the Reference Selection bit field in the Control C register (CTRLC.REFSEL).
3. Configure a positive input by writing to the Positive Input Multiplexer bit field in the Input Multiplexer Control register (INPUTCTRL.MUXPOS).
4. Configure the mode of operation for the ADC by writing to the Differential Mode bit and the Mode bit field in the Command register (COMMAND.DIFF and COMMAND.MODE).
5. Enable the ADC by writing a '1' to the Enable bit in the Control A register (CTRLA.ENABLE).
6. Configure how an ADC conversion will start by writing to the Start Conversion bit field in the Command register (COMMAND.START).

33.4.2. Operation

33.4.2.1. Operation Modes

The ADC supports several modes, each of which allows both single-ended and differential conversions.

The following operation modes are available:

- **Single mode** - One conversion per trigger
- **Series Accumulation mode** - One conversion per trigger, with an accumulation of n samples
- **Burst Accumulation mode** - A burst of n samples accumulated as quickly as possible after a single trigger. The SAMPRDY interrupt/event occurs after every conversion, and the RESRDY interrupt/event occurs after n conversions.

Series and Burst Accumulation modes can be configured with or without scaling of the accumulated result. The Sample Number bit field in the Control D register (CTRLD.SAMPNUM) controls how many samples are accumulated. The accumulator is always reset to '0' when a new Series or Burst accumulation is started.

The Series and Burst Accumulation modes can be used for oversampling to achieve up to five bits higher resolution, given suitable input signal, operating environment, and sampling frequency. Increasing the resolution by N bits can be achieved by accumulating 4^N samples and dividing the accumulated result by 2^N. The CTRLD.SAMPNUM bit field can be configured for up to $4^5 = 1024$ samples, resulting in up to 18-bit resolution (assuming 13-bit resolution is enabled).

33.4.2.2. Conversion Triggers

A conversion is started by one of the following triggers, depending on the configuration of the Start Conversion bit field in the Command register (COMMAND.START):

- Writing the IMMEDIATE value to COMMAND.START
  - This allows for software-controlled sampling
- Writing to the ADC Input Multiplexer Control (INPUTCTRL) register
  - This is useful when the ADC samples multiple inputs sequentially. When switching to the next input, the conversion automatically starts.
- Receiving an event input
  - This configuration allows the ADC to start automatically, either when a non-periodic event signals that a conversion is needed, or when a periodic event, such as the output from a timer, controls the sampling rate

Continuously repeating conversions or accumulations can be by setting the Free-running bit in the Control D register (CTRLD.FREERUN) to '1' before starting the first conversion.

If a new conversion is triggered before the ongoing conversion has finished, the Trigger Overrun Interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TRIGOVR) will be set, and the attempted trigger will be ignored.

33.4.2.3. Aborting a Conversion

There are several ways to stop an ongoing conversion:

- Writing STOP to the Start Conversion bit field in the Command register (COMMAND.START)
- Changing the value in the Input Multiplexer Control (INPUTCTRL) register
- Changing the Reference Selection bit field in the Control C register (CTRLC.REFSEL)

When an ongoing conversion is stopped, the Result (RESULT) and Sample (SAMPLE) registers will contain undefined values.

If a conversion is aborted while a series or burst accumulation is ongoing, the conversion series is aborted, the ADC peripheral returns to the idle state, and the accumulator is reset to '0'.

33.4.2.4. Output Formats

The resolution of an ADC output is determined by the number of bits in the conversion result:

- An **8-bit** ADC provides 256 discrete levels (from 0 to 255)
- A **10-bit** ADC provides 1024 discrete levels (from 0 to 1023)
- A **12-bit** ADC provides 4096 discrete levels (from 0 to 4095)
- A **13-bit** ADC provides 8192 discrete levels (from 0 to 8191)

A **single-ended** conversion measures one input voltage against ground. The result of an n-bit single-ended conversion is an unsigned (positive) integer between 0 and the maximum value 2^n - 1:

$$RESULT \in \mathbb{Z}: 0 \leq RESULT \leq 2^n - 1$$

The result of an n-bit single-ended conversion, as provided in the Result (RESULT) register, is given by the following equation:

$$RESULT = \frac{V_{\mathrm{INP}}}{V_{\mathrm{REF}}} \times 2^n$$

Notes:

- V_{INP}: Input voltage from the positive input multiplexer
- V_{REF}: Voltage reference

A **differential** conversion measures the difference between two input voltages. The result of an n-bit differential conversion is a signed (positive or negative) integer between -2^{n-1} and $2^{n-1} - 1$:

$$RESULT \in \mathbb{Z}: -2^{n-1} \leq RESULT \leq 2^{n-1} - 1$$

The result of an n-bit differential conversion, as provided in the Result (RESULT) register, is given by the following equation:

$$RESULT = \frac{V_{\mathrm{INP}} - V_{\mathrm{INN}}}{V_{\mathrm{REF}}} \times 2^{n-1}$$

Notes:

- V_{INP}: Input voltage from the positive input multiplexer
- V_{INN}: Input voltage from the negative input multiplexer
- V_{REF}: Voltage reference

Single-ended conversions are represented as unsigned values, while differential conversions use signed values in two's complement form. When the conversion result does not occupy the whole register, the unused bits are set to '0'. However, for differential conversion results, the unused bits to the left of the result replicate the sign bit. This process, known as sign extension, ensures that the value retains its correct positive or negative sign when interpreted as a larger bit-width number.

The ADC has two output registers: The Sample (SAMPLE) and Result (RESULT) registers. The SAMPLE register is always updated with the latest ADC conversion output (one sample). In series and burst accumulation modes, samples are added together in an internal sample accumulator, which is configured by the Sample Accumulation Number Select bit field in the Control D register (CTRLD.SAMPNUM). The accumulated result is automatically transferred to the RESULT register when all samples are completed. In single conversion modes, the RESULT register is updated with the latest sample, making it identical to the SAMPLE register.

The Result Scaling bit field in the Control D register (CTRLD.SCALING) determines how data are presented in the SAMPLE and RESULT registers, as shown in the following table.

Table 33-1. Scaling

Description	SAMPLE	RESULT
None	LSb at bit 0	Accumulated value with LSb at bit 0
Left Adjust	MSb at bit 15	Accumulated value is scaled corresponding to the number of samples. Up to 16 bits available, with MSb at bit 15.
Average	LSb at bit 0	Accumulated value divided by the number of samples, with LSb at bit 0

The data format for a single-ended conversion sample is an unsigned number, where `0x0000` represents zero and `0x0FFF` represents the largest value (full scale). For differential conversions, the data format is two's complement with sign extension.

The following tables show the RESULT register output formats for single-ended and differential conversions, based on mode of operation, scaling, and resolution.

Table 33-2. RESULT Register - Single-Ended Mode

MODE	SCALING	RES	RESULT																													
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SINGLE	NORMAL AVERAGE	8	0x00																								Conversion[7:0]							
		10	0x00																								Conversion[9:0]							
		12	0x00																								Conversion[11:0]							
		13	0x00																								Conversion[12:0]							
	LEFTADJ	8	0x00																				Conversion[7:0]								0x00			
		10	0x00																				Conversion[9:0]								0x00			
		12	0x00																				Conversion[11:0]								0x00			
		13	0x00																				Conversion[12:0]								0x00			
SERIES BURST	NORMAL	X	0x00																Accumulation[23:0]															
	AVERAGE	8	0x00																								Average[7:0]							
		10	0x00																								Average[9:0]							
		12	0x00																								Average[11:0]							
		13	0x00																								Average[12:0]							
	LEFTADJ	X	0x00																				Left Adjusted Accumulation[15:0]												

Table 33-3. RESULT Register - Differential Mode

MODE	SCALING	RES	RESULT																													
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SINGLE	NORMAL AVERAGE	8	Sign Extension																								Conversion[7:0]							
		10	Sign Extension																								Conversion[9:0]							
		12	Sign Extension																								Conversion[11:0]							
		13	Sign Extension																								Conversion[12:0]							
	LEFTADJ	8	Sign Extension																				Conversion[7:0]								0x00			
		10	Sign Extension																				Conversion[9:0]								0x00			
		12	Sign Extension																				Conversion[11:0]								0x00			
		13	Sign Extension																				Conversion[12:0]								0x00			
SERIES BURST	NORMAL	X	Sign Extension																Accumulation[23:0]															
	AVERAGE	8	Sign Extension																								Average[7:0]							
		10	Sign Extension																								Average[9:0]							
		12	Sign Extension																								Average[11:0]							
		13	Sign Extension																								Average[12:0]							
	LEFTADJ	X	Sign Extension																				Left Adjusted Accumulation[15:0]												

Table 33-4. SAMPLE Register

MODE	DIFF	SCALING	RES	SAMPLE															
				15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	0	NORMAL AVERAGE	8	0x00								Conversion[7:0]							
			10	0x00								Conversion[9:0]							
			12	0x00								Conversion[11:0]							
			13	0x00								Conversion[12:0]							
	1		8	Sign Extension								Conversion[7:0]							
			10	Sign Extension								Conversion[9:0]							
			12	Sign Extension								Conversion[11:0]							
			13	Sign Extension								Conversion[12:0]							
	X	LEFTADJ	8	Conversion[7:0]								0x00							
			10	Conversion[9:0]										0x00					
			12	Conversion[11:0]												0x00			
			13	Conversion[12:0]													0x00		

33.4.2.5. ADC Clock

The ADC clock (CLK_ADC) is derived by scaling down the APB clock (CLK_ADCn_APB). The amount of scaling is configured by the Prescaler bit field in the Control B register (CTRLB.PRESC). Refer to the ADC section of the *Electrical Characteristics* chapter for details on the minimum and maximum allowed values for ADC clock (CLK_ADC) period.

Some of the internal timings in the ADC are independent of CLK_ADC. The Timebase bit field in the Control B register (CTRLB.TIMEBASE) specifies the number of CLK_ADCn_APB cycles that is equivalent to or larger than 1 µs. This value is used for timing internal delays, such as ADC start-up times. The value must be rounded up to the nearest integer. The following code example shows how to do this using the `ceil` function.

```
#include <math.h>
#define CLK_ADCn_APB 24000000ul // 24 MHz
#define TIMEBASE_VALUE ((uint8_t) ceil(CLK_ADCn_APB*0.000001))
```

33.4.2.6. Input and Reference Selection

The input selection to the ADC is controlled by the Positive and Negative Input Multiplexer bit fields in the Input Control register (INPUTCTRL.MUXPOS and INPUTCTRL.MUXNEG). For single-ended conversions, only MUXPOS is used, while both MUXPOS and MUXNEG are used for differential conversions.

The reference voltage (V_{REF}) determines the conversion range of the ADC. It can be selected by configuring the Reference Selection bit field in the Control C register (CTRLC.REFSEL).

Except for V_{DD} and internal ground, the internal reference voltages are generated from an internal band gap reference.

Table 33-5. ADC Input Range and Resolution

Mode	Resolution	Negative Input Voltage (V_{INN})	Positive Input Voltage (V_{INP}) for Minimum Result	Input Voltage (V_{INP}) for 0 Result	Input Voltage (V_{INP}) for Maximum Result	LSb Resolution
Single-ended	8	GND	GND		V_{REF}	$V_{REF}/256$
	10					$V_{REF}/1024$
	12					$V_{REF}/4096$
	13	V_{REF}			$2 * V_{REF}$	$V_{REF}/4096$

Table 33-5. ADC Input Range and Resolution (continued)

Mode	Resolution	Negative Input Voltage (V_{INN})	Positive Input Voltage (V_{INP}) for Minimum Result	Input Voltage (V_{INP}) for 0 Result	Input Voltage (V_{INP}) for Maximum Result	LSb Resolution
Differential	8	MUXNEG	V_{INN} - V_{REF}	V_{INN}	V_{INN} + V_{REF}	V_{REF}/128
	10					V_{REF}/512
	12					V_{REF}/2048
	13					V_{REF}/4096

Changing the input or reference selections while a conversion is ongoing will corrupt the output. To avoid this, detect an ongoing conversion by reading the ADC Busy bit in the Status register (STATUS.ADCBUSY). Writing STOP to the Start Conversion bit field in the Command register (COMMAND.START) will stop an ongoing conversion.

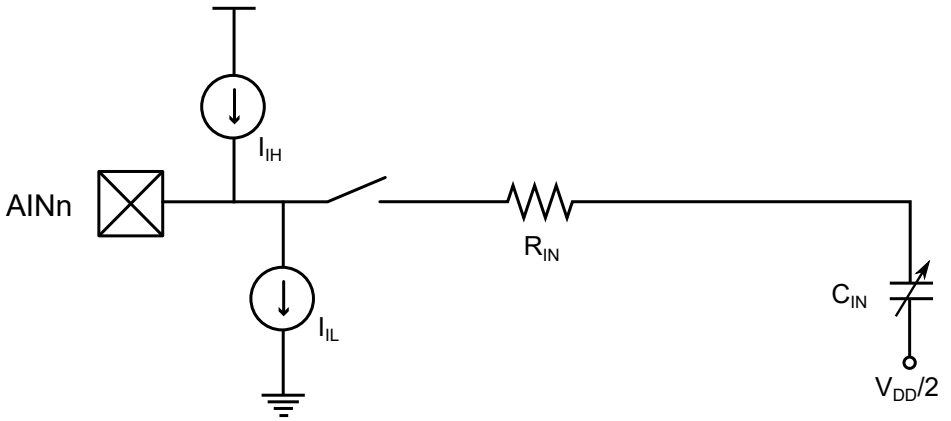
After switching the input or reference, the ADC requires time to settle. Refer to the *Electrical Characteristics* chapter for further details.

33.4.2.6.1. Analog Input Circuit

The figure below illustrates the analog input circuit. An analog source connected to an analog input (AINn) is subject to the pin capacitance and input leakage of that pin (represented by I_{IH} and I_{IL}). When the input is selected, the source must also drive the sample-and-hold capacitor (C_{IN}) through the combined resistance of the input path (represented by R_{IN}).

Refer to the *Electrical Characteristics* chapter for details on the input characteristics of the ADC.

Figure 33-2. Analog Input Schematic



If a source with high impedance is used, the sampling time may need to be increased. The required sample time depends on how long the source needs to charge the C_{IN} capacitor and can be configured using the Sample Length bit field in the Control E register (CTRLE.SAMPLEN).

33.4.2.7. Conversion Timing

Some of the analog components in the ADC are disabled between conversions and require time to initialize before a conversion starts. Only the components used by the current ADC configuration are enabled, and since the initializations run in parallel, the limiting factor is the module with the slowest initialization time. The ADC Busy (ADCBUSY) bit in the Status (STATUS) register can be used to check if initialization is ongoing. Refer to the *Electrical Characteristics* chapter for the initialization times needed by the analog modules.

When the On Demand bit in the Control A register (CTRLA.ONDEMAND) is '0', the latency of the ADC peripheral is reduced. This keeps the configured modules continuously enabled, effectively

eliminating all initialization time at the start of a conversion. However, initialization time is still required when enabling the ADC for the first time and when reconfiguring the ADC to use an input or reference that requires initialization.

The sampling period of the ADC is configured using the Sample Length bit field in the Control E register (CTRLE.SAMPLEN), and is `SAMPLEN + 1` CLK_ADC cycles.

Table 33-6. Conversion Timing

Mode	Resolution	Trigger Detection	Initialization	Sampling	Data Conversion	Result Update
Single-Ended	8	3 * CLK_ADCn_APB	t_{INIT}	(SAMPLEN + 1) * CLK_ADC	8	3 * CLK_ADCn_APB
	10				10	
	12				12	
	13				13	
Differential	8				9	
	10				11	
	12				13	
	13				13	

Figure 33-3. Timing Diagram - Single Conversion

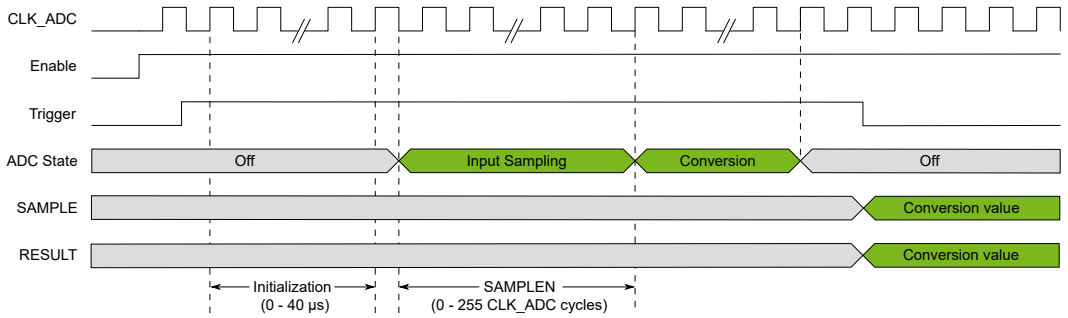


Figure 33-4. Timing Diagram - Series Accumulation

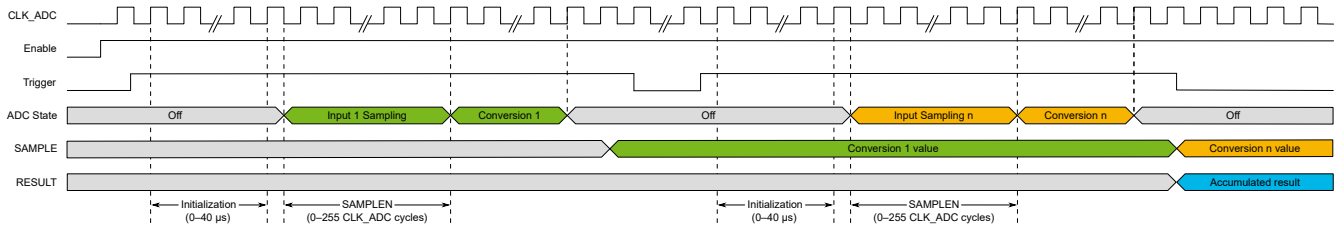
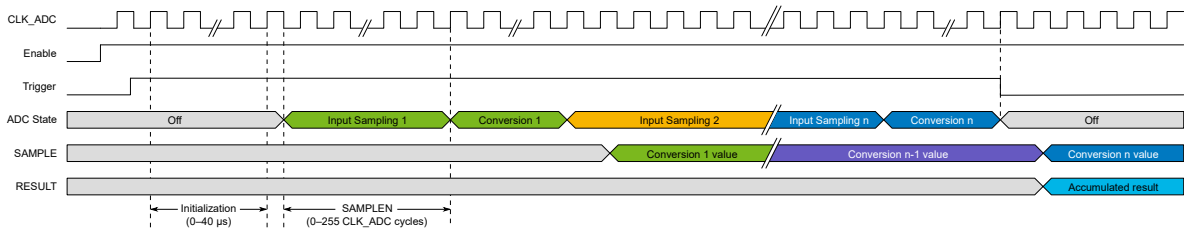


Figure 33-5. Timing Diagram - Burst Accumulation

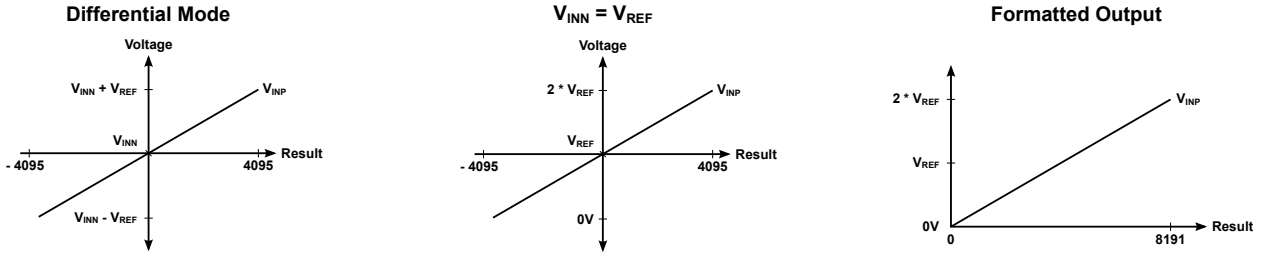


33.4.3. Additional Features

33.4.3.1. 13-bit Mode Operation

When 13-bit mode is selected, the ADC extends its resolution and input range by utilizing a differential conversion scheme for both single-ended and differential measurements. For differential mode, the ADC outputs a signed 13-bit result, representing the voltage difference between the positive and negative inputs, with a range from -4096 to +4095. For single-ended measurements, the ADC internally configures itself as a differential converter, assigning the selected reference voltage (V_{REF}) to the negative input. The output is formatted as an unsigned 13-bit value, shifting the range from -4096 to +4096 to 0 to 8191. This corresponds to an input voltage range of 0V to 2×V_{REF}. This effectively doubles the measurable voltage span compared to standard single-ended operation.

The graphs below show 13-bit operation in single-ended mode. The graph to the left shows a standard differential measurement, where the result is determined by the difference between the positive and negative inputs. The middle graph demonstrates how the measurement changes when V_{REF} is assigned to the negative input. Finally, the graph on the right displays the output of the 13-bit single-ended conversion, where the result is shifted to output only positive numbers.



33.4.3.2. Offset Reduction by Sign Chopping

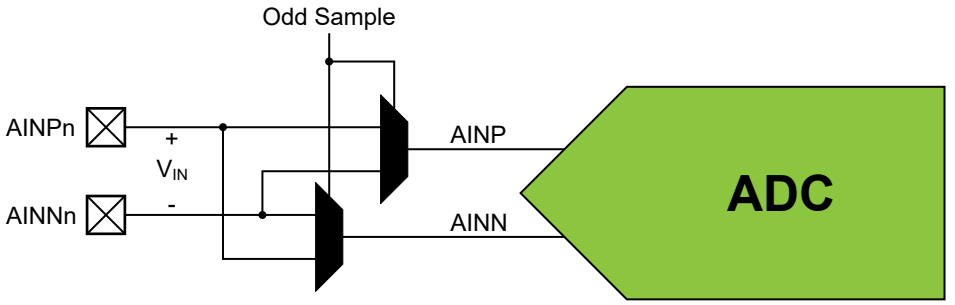
Sign Chopping is a feature that can mitigate offset error in the ADC. It is enabled by writing a '1' to the Sign Chopping bit in the Control D register (CTRLD.CHOPPING).

The feature works by swapping the inputs to the ADC every other sample and then subtracting the inverted sample (-V_{IN}) from the non-inverted sample (V_{IN}). Since the offset error (V_{OFFSET}) of the ADC is not proportional to the input voltage, this operation effectively cancels out the offset. This is illustrated in the following equation:

$$(V_{IN} + V_{OFFSET}) - (-V_{IN} + V_{OFFSET}) = V_{IN} + V_{OFFSET} + V_{IN} - V_{OFFSET} = 2 \times V_{IN}$$

The figure below illustrates the Sign Chopping feature. For even samples, the ADC operates as normal. For odd samples, the ADC swaps its inputs to invert the input signal without inverting the offset.

Figure 33-6. Sign Chopping



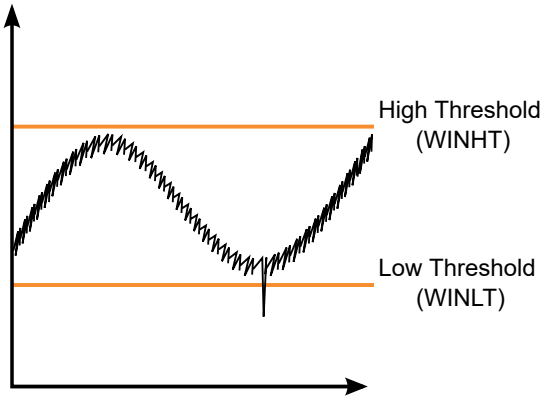
Sign Chopping can be used with either single-ended or differential conversions. It may only be used in accumulation modes (Series or Burst).

33.4.3.3. Window Comparator

The ADC features a window comparator that automatically compares the conversion result to two configurable thresholds. The outcome of the comparison is indicated by the Window Comparator Interrupt (WCMP) flag in the Interrupt Flag Status and Clear (INTFLAG) register, and can optionally trigger the corresponding interrupt. The available modes are:

- The value is below the low threshold
- The value is above the high threshold
- The value is inside the window (above the low threshold and below the high threshold)
- The value is outside the window (either below the low threshold or above the high threshold)

Figure 33-7. Window Comparator



The thresholds are set by writing to the Window Low and High Threshold (WINLT and WINHT) registers. The desired window mode is selected by configuring the Window Comparator Mode bit field in the Window Control register (WINCTRL.WINMODE).

The Window Mode Source bit field in Window Control register (WINCTRL.WINSRC) selects whether the comparison is performed on the 16 LSb of the Result (RESULT) register or the Sample (SAMPLE) register. If interrupt requests are enabled for the WCMP flag, WINSRC also determines which interrupt vector to request: RESRDY or SAMPRDY.

When accumulating multiple samples, if the Window Comparator source is the Result register, the comparison between the result and the threshold(s) occurs after the last conversion is complete. If the source is the Sample register, the comparison occurs after every conversion.

Assuming the ADC is already configured to run, follow these steps to use the Window Comparator mode:

1. Set the required threshold(s) by writing to the WINLT and WINHT registers.
2. Optional: Enable the interrupt request by writing a '1' to the WCMP bit in the Interrupt Enable Set (INTENSET) register.
3. Optional: Write the WINCTRL.WINSRC bit to select the source used by the Window Comparator.
4. Enable the Window Comparator by writing a non-zero value to WINCTRL.WINMODE.

The value in the RESULT register can exceed the number of bits available for window comparison when the Result Scaling bit field in the Control D register (CTRLD.SCALING) is configured to NORMAL and accumulating more than $2^{resolution - 8}$ (for example more than $2^4 = 16$ samples in 12-bit mode).

33.4.3.4. Low-pass Filter

Low-pass Filter mode works together with accumulation modes to effectively mitigating high-frequency noise.

When Low-pass Filter mode is enabled, the behavior of the accumulation modes changes. After an initial accumulation of samples, as configured in the Sample Accumulation Number Select bit field in the Control D register (CTRLD.SAMPNUM), the ADC continues to sample and accumulate indefinitely.

During the accumulation phase, while the specified number of samples are being accumulated, no filtering is applied. Once the specified number of samples have been converted, the Result (RESULT) register is updated as normal.

After the accumulation phase, the filter phase updates the accumulator with each new sample according to the following equation.

Equation 33-1. Filtering Phase

$$RESULT = \left(\frac{SAMPNUM - 1}{SAMPNUM} \times ACC\right) + SAMPLE$$

As shown in the equation, the average of the accumulator (ACC) is subtracted as the new sample is added. The RESULT register is updated with every new sample, and the Result Ready (RESRDY) interrupt flag is set to '1'.

When using Low-pass Filter mode together with Series accumulation mode, each sample requires a trigger, both during the accumulation phase and the filter phase.

For Burst mode, a single trigger causes all samples in the accumulation phase to complete back-to-back. In the filter phase, the Free-Running bit in CTRLD (CTRLD.FREERUN) determines whether a trigger results in a single sample (when FREERUN is '0') or back-to-back sampling (when FREERUN is '1') until stopped.

Low-pass Filter mode can be used together with Sign Chopping to reduce offset error. During the filter phase, each ADC sample automatically triggers a second sample with the input inverted. The output from Sign Chopping is added as normal in the filtering phase. For more information about Sign Chopping, see the section Offset Reduction by Sign Chopping.

33.4.3.5. Accumulator Test

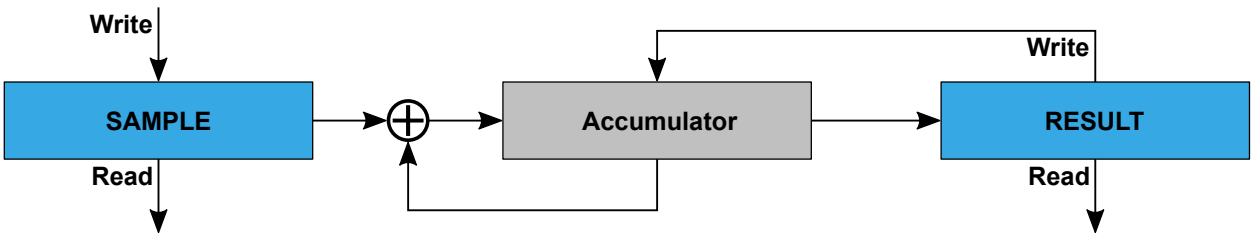
Accumulator Test mode aids in diagnostics of the accumulator and result formatting hardware, and can be used to test the integrity of the logic. It is enabled by configuring the Mode bit field in the Command register (COMMAND.MODE) to ACCTEST.

The Accumulator Test mode is used as follows:

- Writing to the Result (RESULT) register writes directly to the accumulator
- Writing to the Sample (SAMPLE) register accumulates the written value in the accumulator
- Reading the Result register will read the accumulator value

Note: Transitioning out of Accumulator Test mode clears the internal accumulator, as well as the RESULT and SAMPLE registers.

Figure 33-8. Accumulator Test



33.4.4. Sleep Mode Operation

The ADC can start conversions in Idle or Standby sleep mode if the Start Conversion bit field in the Command register (COMMAND.START) is configured to start a conversion on an event trigger. For

Standby sleep mode, the Run in Standby bit in the Control A register (CTRLA.RUNSTDBY) must also be set to '1'.

Table 33-7. ADC Behavior in FREERUN, RUNSTDBY and DBGRUN

Entering Power State	FREERUN	RUNSTDBY	DBGRUN	Behavior of Single or Series Conversion	Behavior of Burst Conversion
Standby	0	0	X	Finish the current conversion, then enter sleep mode. No interrupt request. The Series is aborted.	Finish the current burst, without interrupt request, then enter sleep mode.
	0	1	X	Finish the current conversion, then enter sleep mode. Then interrupt request. The Series is not aborted.	Finish the current burst, then enter sleep mode, followed by an interrupt request
	1	0	X	Finish the current conversion, then enter sleep mode. No interrupt request. Not applicable for series operation.	Finish the current burst without interrupt request, then enter sleep mode.
	1	1	X	Continuous free-running conversions with interrupt requests between conversions (relevant for window mode). Not applicable for series operation.	Continuous free-running bursts with interrupt requests between bursts (relevant for window mode)
Debug	0	X	0	Finish the ongoing conversion, then stop. New triggers are ignored.	Finish the current burst, then stop. New triggers are ignored.
	0	X	1	Finish the ongoing conversion, then stop. Will accept new triggers.	Finish the current burst, then stop. New triggers will be accepted.
	1	X	0	Finish the current conversion, then stop. Automatically restarts when break is released. Not applicable for series mode.	Finish the current burst, then stop. Automatically restarts when the break is released.
	1	X	1	Continuous free-running burst conversions, also active during break. Not applicable for series mode.	Continuous free-running burst conversions, also active during break.

Refer to the *PM - Power Manager* chapter for more information about sleep mode operation.

33.4.5. Debug Operation

When the CPU is halted in debug mode, the ADC will halt normal operation. Setting the Debug Run bit in the Debug Control register (DBGCTRL.DBGRUN) forces the ADC to continue operating while the CPU is halted in debug mode. Refer to Table 33-7 for details on ADC behavior based on the mode of operation and the configuration of the DBGRUN bit.

33.5. Dependencies

33.5.1. I/O Lines

The ADC samples the voltage present on the I/O pins. Typically, the I/O pin output buffer should be disabled to prevent interference with the input signal. This is done by writing the corresponding Port Data Direction bit in the Data Direction register (DIR.DIR) to '0', which is the default setting for the I/O pins. Additionally, to minimize switching noise from the I/O pin's input buffer, it is recommended to disable it by writing the Input Buffer Enable to '0' in the corresponding Pin Configuration register (PINCFG.INEN). For more details, refer to the *PORT – I/O Pin Controller* chapter.

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

33.5.2. Power Management

The ADC will continue to operate in any sleep mode where the selected source clock is running. ADC 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 the *Power Manager* chapter for details on the different sleep modes.

33.5.3. Clocks

The ADC bus clock (CLK_ADCn_APB) can be enabled or disabled in the Main Clock (MCLK). The default state of CLK_ADCn_APB is detailed in the Peripheral Clock Masking section of the *MCLK – Main Clock* chapter.

33.5.4. DMA

The ADC generates the following DMA requests:

- Result Ready (RESRDY): The request is set when a conversion result is available and is cleared when the Result (RESULT) register is read.
- Sample Ready (SAMPRDY): The request is set when a conversion sample is available and is cleared when the Sample (SAMPLE) register is read.

The DMA request lines are connected to the DMA Controller (DMAC). The DMAC must be configured before using the ADC's DMA requests. Refer to the *DMAC – Direct Memory Access Controller* chapter for details.

33.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use ADC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 33-8. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
ADCn	TRIGOVR	A new conversion is triggered while another conversion is already in progress	
	SAMPOVR	A new conversion overwrites an unread sample in the Sample (SAMPLE) register	
	RESOVR	A new conversion or accumulation overwrites an unread result in the Result (RESULT) register	
	RESRDY	A new result is available in the Result (RESULT) register	
	WCMP	A conversion or accumulation matches the conditions set by the window comparator	The Window Source bit in the Control D register (CTRLD.WINSRC) is '0'
	SAMPRDY	A new sample is available in the Sample (SAMPLE) register	
	WCMP	A sample matches the conditions set by the window comparator	CTRLD.WINSRC is '1'

33.5.6. Events

The ADC can generate the following events:

Table 33-9. ADC Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
ADCn	RESRDY	Result ready	Pulse	CLK_ADCn_APB	One CLK_ADCn_APB period
ADCn	SAMPRDY	Sample ready	Pulse	CLK_ADCn_APB	One CLK_ADCn_APB period
ADCn	WCMP	Window comparator	Pulse	CLK_ADCn_APB	One CLK_ADCn_APB period

Writing a '1' to an Event Output bit in the Event Control Register (EVCTRL.xxEO) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The ADC can connect to the following events:

Table 33-10. ADC Event Users and Available Event Actions

User Name		Description	Input Detection	Channel Path Type
Peripheral	Input			
ADCn	START	Conversion start	Edge	Asynchronous

Writing a '1' to an Event Input bit in the Event Control register (EVCTRL.xxEI) enables the corresponding action on input event. Writing a '0' to this bit disables the corresponding action on input event.

Refer to the *EVSYS - Event System* chapter for details on configuring the event system.

33.5.7. Analog Connections

Refer to the MUXPOS and MUXNEG bit fields in INPUTCTRL.

33.6. Register Summary - ADCn

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	ONDEMAND	RUNSTDBY					ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x04	CTRLB	7:0	PRESCALER[4:0]							
		15:8	TIMEBASE[4:0]							
		23:16								
		31:24								
0x08	CTRLC	7:0						REFSEL[2:0]		
		15:8								
		23:16								
		31:24								
0x0C	CTRLD	7:0					SAMPNUM[3:0]			
		15:8		FILTER	CHOPPING	FREERUN	SCALING[1:0]		RESOLUTION[1:0]	
		23:16								VPD
		31:24								
0x10	CTRLE	7:0	SAMPLEN[7:0]							
		15:8								
		23:16								
		31:24								
0x14 ... 0x1B	Reserved									
0x1C	WINCTRL	7:0					WINSRC	WINMODE[2:0]		
		15:8								
		23:16								
		31:24								
0x20	WINLT	7:0	WINLT[7:0]							
		15:8	WINLT[15:8]							
		23:16								
		31:24								
0x24	WINHT	7:0	WINHT[7:0]							
		15:8	WINHT[15:8]							
		23:16								
		31:24								
0x28	INPUTCTRL	7:0		MUXNEG[6:0]						
		15:8		MUXPOS[6:0]						
		23:16								
		31:24								
0x2C	EVCTRL	7:0				STARTINV				STARTEI
		15:8						WCMPEO	SAMPRDYEO	RESRDYEO
		23:16								
		31:24								
0x30	INTENCLR	7:0			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
		15:8								
		23:16								
		31:24								
0x34	INTENSET	7:0			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
		15:8								
		23:16								
		31:24								
0x38	INTFLAG	7:0			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
		15:8								
		23:16								
		31:24								
0x3C	INTFLAGSET	7:0			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
		15:8								
		23:16								
		31:24								

ADCn (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x40	STATUS	7:0								ADCBUSY
		15:8								
		23:16								
		31:24								
0x44 ... 0x47	Reserved									
0x48	COMMAND	7:0	DIFF		MODE[2:0]			START[2:0]		
		15:8								
		23:16								
		31:24								
0x4C ... 0x4F	Reserved									
0x50	RESULT	7:0	RESULT[7:0]							
		15:8	RESULT[15:8]							
		23:16	RESULT[23:16]							
		31:24	RESULT[31:24]							
0x54	SAMPLE	7:0	SAMPLE[7:0]							
		15:8	SAMPLE[15:8]							
		23:16								
		31:24								
0x58 ... 0x6B	Reserved									
0x6C	DBGCTRL	7:0								DBGRUN
		15:8								
		23:16								
		31:24								
0x70 ... 0x7B	Reserved									
0x7C	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

33.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 7 – ONDEMAND On Demand
This bit controls whether the required analog modules remain enabled.
Note: ONDEMAND does not keep the clock source enabled when the ADC is not converting, so a clock startup delay may occur even if ONDEMAND is cleared. To avoid this delay, ensure that the clock source is always enabled.

Value	Description
0	The analog modules remain enabled as long as they are selected as input to the ADC. This configuration minimizes the ADC initialization time.
1	The ADC enables the required analog modules only when starting a conversion. This reduces overall power consumption but increases the initialization time when starting an ADC conversion.

Bit 6 – RUNSTDBY Run in Standby
This bit controls whether the ADC operates in Standby sleep mode.

Value	Description
0	The ADC will not operate in Standby sleep mode. An ongoing conversion will complete before the ADC enters sleep mode.
1	The ADC will operate in Standby sleep mode. The main clock will be requested when the ADC is triggered to perform a conversion.

Bit 1 – ENABLE ADC Enable
This bit controls whether the ADC is enabled.

Value	Description
0	The ADC is disabled
1	The ADC is enabled

Bit 0 – SWRST Software Reset

Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the ADC, except the Debug Control (DBGCTRL) register, to their initial state, and disables the ADC.
Note: Writing a '1' to CTRLA.SWRST will always take precedence; all other writes in the same write operation will be discarded.

Value	Description
0	No effect
1	All registers in the ADC, except DBGCTRL, are reset to their initial state, and the ADC is disabled

33.6.2. Control B

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

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

Bits 12:8 – TIMEBASE[4:0] Timebase
This bit specifies the number of CLK_ADCn_APB cycles that is equivalent to or larger than 1 µs. The value must be rounded up to the closest integer. Refer to ADC Clock for details on how to calculate this.

Bits 4:0 – PRESCALER[4:0] Prescaler
This bit field controls the division factor from the peripheral clock (CLK_ADCn_APB) to the ADC clock (CLK_ADC).

Value	Name	Description
0x00	DIV2	APB clock divided by 2
0x01	DIV3	APB clock divided by 3
0x02	DIV4	APB clock divided by 4
0x03	DIV5	APB clock divided by 5
0x04	DIV6	APB clock divided by 6
0x05	DIV7	APB clock divided by 7
0x06	DIV8	APB clock divided by 8
0x07	DIV9	APB clock divided by 9
0x08	DIV10	APB clock divided by 10
0x09	DIV11	APB clock divided by 11
0x0A	DIV12	APB clock divided by 12
0x0B	DIV13	APB clock divided by 13
0x0C	DIV14	APB clock divided by 14
0x0D	DIV15	APB clock divided by 15
0x0E	DIV16	APB clock divided by 16
0x0F	DIV17	APB clock divided by 17
0x10	DIV18	APB clock divided by 18

Value	Name	Description
0x11	DIV19	APB clock divided by 19
0x12	DIV20	APB clock divided by 20
0x13	DIV21	APB clock divided by 21
0x14	DIV22	APB clock divided by 22
0x15	DIV23	APB clock divided by 23
0x16	DIV24	APB clock divided by 24
0x17	DIV25	APB clock divided by 25
0x18	DIV26	APB clock divided by 26
0x19	DIV27	APB clock divided by 27
0x1A	DIV28	APB clock divided by 28
0x1B	DIV29	APB clock divided by 29
0x1C	DIV30	APB clock divided by 30
0x1D	DIV31	APB clock divided by 31
0x1E	DIV32	APB clock divided by 32
Other	—	Reserved

33.6.3. Control C

Name: CTRLC
Offset: 0x08
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bits 2:0 – REFSEL[2:0] Reference Selection
This bit field selects the voltage reference for the ADC. Switching to internal references requires a 40 µs initialization time for the first conversion. Internal references can only be used if they are lower than VDD - 0.5V.

Value	Name	Description
0x0	VDD	V_{DD}
0x2	EXTVREF	External pin used as reference
0x4	1V024	Internal reference 1.024V
0x5	2V048	Internal reference 2.048V
0x6	4V096	Internal reference 4.096V
0x7	2V500	Internal reference 2.500V
Other	—	Reserved

33.6.4. Control D

Name: CTRLD
Offset: 0x0C
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

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

Bit	15	14	13	12	11	10	9	8
		FILTER	CHOPPING	FREERUN	SCALING[1:0]		RESOLUTION[1:0]	
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

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

Bit 16 – VPD Voltage Pump Disable
This bit controls the internal voltage pump for the ADC. The voltage pump must be disabled when using the PTC and operating above 4.5V; otherwise the pump should be enabled.

Value	Description
0	Voltage pump is enabled
1	Voltage pump is disabled

Bit 14 – FILTER Low-Pass Filter
This bit enables the low-pass filter. It is only used in Series and Burst modes.

Value	Description
0	Low-pass filter is disabled
1	Low-pass filter is enabled

Bit 13 – CHOPPING Sign Chopping
This bit controls whether sign chopping is enabled to reduce offset.
It is only used in Series and Burst modes.

Value	Description
0	Sign chopping is disabled
1	Sign chopping is enabled

Bit 12 – FREERUN Free-Running
This bit controls whether the ADC Free-Running mode is enabled.

Value	Description
0	The ADC Free-Running mode is disabled

Value	Description
1	The ADC Free-Running mode is enabled. A new conversion starts as soon as the previous conversion or accumulation is completed, as indicated by INTFLAG.RESRDY.

Bits 11:10 – SCALING[1:0] Result Scaling

This bitfield controls the scaling of the digital value in SAMPLE and RESULT registers. In NORMAL mode, the full accumulated result is returned. In LEFTADJ mode, the result is truncated if it exceeds 16 bits. In AVERAGE mode, the result is rounded to the configured resolution.

Value	Name	Description
0x0	NORMAL	The ADC output of the sample or accumulated result is right adjusted
0x1	LEFTADJ	The ADC output is left adjusted
0x2	AVERAGE	The accumulated ADC result is averaged and right adjusted
Other	—	Reserved

Bits 9:8 – RESOLUTION[1:0] ADC Resolution

This bit-field defines the number of bits output from the ADC after a conversion. The conversion time depends on the number of bits.

Value	Name	Description
0x0	12BIT	12-bit ADC Result
0x1	13BIT	13-bit ADC Result
0x2	10BIT	10-bit ADC Result
0x3	8BIT	8-bit ADC Result

Bits 3:0 – SAMPNUM[3:0] Sample Accumulation Number Select

This bit field controls the number of consecutive ADC samples that are automatically accumulated into the ADC Result (ADCn.RESULT) register. The most recent sample will be available in the ADC Sample (ADCn.SAMPLE) register.

Value	Name	Description
0x0	NONE	No accumulation, single sample per conversion result
0x1	ACC2	2 samples accumulated
0x2	ACC4	4 samples accumulated
0x3	ACC8	8 samples accumulated
0x4	ACC16	16 samples accumulated
0x5	ACC32	32 samples accumulated
0x6	ACC64	64 samples accumulated
0x7	ACC128	128 samples accumulated
0x8	ACC256	256 samples accumulated
0x9	ACC512	512 samples accumulated
0xA	ACC1024	1024 samples accumulated
Other	—	Reserved

33.6.5. Control E

Name: CTRLE
Offset: 0x10
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	SAMPLEN[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 – SAMPLEN[7:0] Sample Length
This bit field selects the ADC sample duration in ADC clock (CLK_ADC) cycles. The sample duration is (SAMPLEN+1) CLK_ADC cycles.

33.6.6. Window Control

Name: WINCTRL
Offset: 0x1C
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 3 – WINSRC Window Mode Source
This bit selects the source for the Window Comparator.

Value	Name	Description
0	RESULT	Use RESULT[15:0] as Window Comparator source
1	SAMPLE	Use SAMPLE[15:0] as Window Comparator source

Note: The number of bits populated in the result register may be more than 16 when using normal scaling. Refer to Window Comparator for details.

Bits 2:0 – WINMODE[2:0] Window Comparator Mode
This bit field enables and defines when the interrupt flag is set in Window Comparator mode. In the table below, OUTPUT refers to the 16-bit result or sample selected by WINCTRL.WINSRC. WINLT and WINHT are the 16-bit low and high threshold values configured in the Window Low Threshold (WINLT) and Window High Threshold (WINHT) registers, respectively.

Value	Name	Description
0x0	NONE	No Window Comparison
0x1	BELOW	OUTPUT < WINLT
0x2	ABOVE	OUTPUT > WINHT
0x3	INSIDE	WINLT < OUTPUT < WINHT
0x4	OUTSIDE	OUTPUT < WINLT or OUTPUT >WINHT
Other	—	Reserved

33.6.7. Window Low Threshold

Name: WINLT
Offset: 0x20
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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 Low Threshold

This register is the 16-bit Low Threshold for the digital comparator monitoring the ADC Result or Sample (RESULT or SAMPLE) registers. The data format must match the selected conversion mode and left adjustment setting. When monitoring the ADC Result register in an Accumulation mode, the window comparator thresholds are applied to the result after all accumulation (and optionally scaling) has been completed.

33.6.8. Window High Threshold

Name: WINHT
Offset: 0x24
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WINHT[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
	WINHT[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 – WINHT[15:0] Window High Threshold

This register is the 16-bit High Threshold for the digital comparator monitoring the ADC Result or Sample (RESULT or SAMPLE) registers. The data format must match the selected conversion mode and left adjustment setting. When monitoring the ADC Result register in an Accumulation mode, the window comparator thresholds are applied to the result after all accumulation (and optionally scaling) has been completed.

33.6.9. Input Multiplexer Control

Name: INPUTCTRL
Offset: 0x28
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

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

Bits 14:8 – MUXPOS[6:0] Positive Input Multiplexer
This bit field controls which analog input is connected to the positive input of the ADC. Changing this setting may require a settling time. Refer to the *Electrical Characteristics* chapter for further details.

Value	Name	Description
0x2	AIN2	ADC input 2
0x3	AIN3	ADC input 3
0x4	AIN4	ADC input 4
0x5	AIN5	ADC input 5
0x6	AIN6	ADC input 6
0x7	AIN7	ADC input 7
0x10	AIN16	ADC input 16
0x11	AIN17	ADC input 17
0x12	AIN18	ADC input 18
0x13	AIN19	ADC input 19
0x14	AIN20	ADC input 20
0x15	AIN21	ADC input 21
0x16	AIN22	ADC input 22
0x17	AIN23	ADC input 23
0x18	AIN24	ADC input 24
0x19	AIN25	ADC input 25
0x1A	AIN26	ADC input 26
0x1B	AIN27	ADC input 27
0x1C	AIN28	ADC input 28
0x1D	AIN29	ADC input 29
0x1F	AIN31	ADC input 31
0x28	AIN40	ADC input 40

Value	Name	Description
0x29	AIN41	ADC input 41
0x2A	AIN42	ADC input 42
0x2B	AIN43	ADC input 43
0x2C	AIN44	ADC input 44
0x2D	AIN45	ADC input 45
0x2E	AIN46	ADC input 46
0x2F	AIN47	ADC input 47
0x60	GND	Ground
0x61	VDDCOREDIV4	Internally generated core supply voltage divided by four
0x62	TEMPSENSE	Temperature sensor
0x64	VDDIODIV10	VDDIO divided by 10
0x65	VDDIO2DIV10	VDDIO2 divided by 10
0x69	ACVREFSCALE0	AC COMP0 Voltage Reference Scaler Output
0x6A	ACVREFSCALE1	AC COMP1 Voltage Reference Scaler Output
0x7E	PTC	Peripheral Touch Controller signal
Other	-	Reserved

Bits 6:0 – MUXNEG[6:0] Negative Input Multiplexer
This bit field controls which analog input is connected to the negative input of the ADC. Changing this setting may require a settling time. Refer to the *Electrical Characteristics* chapter for further details.

Value	Name	Description
0x2	AIN2	ADC input 2
0x3	AIN3	ADC input 3
0x4	AIN4	ADC input 4
0x5	AIN5	ADC input 5
0x6	AIN6	ADC input 6
0x7	AIN7	ADC input 7
0x10	AIN16	ADC input 16
0x11	AIN17	ADC input 17
0x12	AIN18	ADC input 18
0x13	AIN19	ADC input 19
0x14	AIN20	ADC input 20
0x15	AIN21	ADC input 21
0x16	AIN22	ADC input 22
0x17	AIN23	ADC input 23
0x18	AIN24	ADC input 24
0x19	AIN25	ADC input 25
0x1A	AIN26	ADC input 26
0x1B	AIN27	ADC input 27
0x1C	AIN28	ADC input 28
0x1D	AIN29	ADC input 29
0x1F	AIN31	ADC input 31
0x28	AIN40	ADC input 40
0x29	AIN41	ADC input 41
0x2A	AIN42	ADC input 42
0x2B	AIN43	ADC input 43
0x2C	AIN44	ADC input 44
0x2D	AIN45	ADC input 45
0x2E	AIN46	ADC input 46
0x2F	AIN47	ADC input 47
0x60	GND	Ground
0x61	VREFP	Positive voltage reference

Value	Name	Description
Other	-	Reserved

33.6.10. Event Control

Name: EVCTRL
Offset: 0x2C
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						WCMPEO	SAMPRDYEO	RESRDYEO
Access						R/W	R/W	R/W
Reset						0	0	0

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

Bit 10 – WCMPEO Window Comparator Event Output Enable
Writing a '0' to this bit disables the output of the window comparator event.
Writing a '1' to this bit enables the output of the window comparator event.

Value	Description
0	The Window Comparator event is disabled
1	The Window Comparator event is enabled

Bit 9 – SAMPRDYEO Sample Ready Event Output Enable
Writing a '0' to this bit disables the output of the sample ready event.
Writing a '1' to this bit enables the output of the sample ready event.

Value	Description
0	The Sample Ready event is disabled
1	The Sample Ready event is enabled

Bit 8 – RESRDYEO Result Ready Event Output Enable
Writing a '0' to this bit disables the output of the result ready event.
Writing a '1' to this bit enables the output of the result ready event.

Value	Description
0	The Result Ready event is disabled
1	The Result Ready event is enabled

Bit 4 – STARTINV Conversion Start Event Input Invert
Writing a '0' to this bit disables inversion of the input event for starting a conversion.
Writing a '1' to this bit enables inversion of the input event for starting a conversion.

Value	Description
0	The Conversion Start event is not inverted
1	The Conversion Start event is inverted

Bit 0 – STARTEI Conversion Start Event Input Enable
Writing a '0' to this bit disables event input for starting a conversion.
Writing a '1' to this bit enables event input for starting a conversion.

Value	Description
0	The Conversion Start event is disabled
1	The Conversion Start event is enabled

33.6.11. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x30
Reset: 0x00000000
Property: Local Write-Protection

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

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TRIGOVR Trigger Overrun Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Trigger Overrun Interrupt Enable bit, disabling the trigger overrun interrupt.

Value	Description
0	The Trigger Overrun interrupt is disabled
1	The Trigger Overrun interrupt is enabled

Bit 4 – SAMPOVR Sample Overwrite Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Sample Overwrite Interrupt Enable bit, disabling the sample overwrite interrupt.

Value	Description
0	The Sample Overwrite interrupt is disabled
1	The Sample Overwrite interrupt is enabled

Bit 3 – RESOVR Result Overwrite Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Result Overwrite Interrupt Enable bit, disabling the result overwrite interrupt.

Value	Description
0	The Result Overwrite interrupt is disabled

Value	Description
1	The Result Overwrite interrupt is enabled

Bit 2 – WCMP Window Comparator Interrupt Enable

Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Window Comparator Interrupt Enable bit, disabling the window comparator interrupt.

Value	Description
0	The Window Comparator interrupt is disabled
1	The Window Comparator interrupt is enabled

Bit 1 – SAMPRDY Sample Ready Interrupt Enable

Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Sample Ready Interrupt Enable bit, disabling the sample ready interrupt.

Value	Description
0	The Sample Ready interrupt is disabled
1	The Sample Ready interrupt is enabled

Bit 0 – RESRDY Result Ready Interrupt Enable

Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Result Ready Interrupt Enable bit, disabling the result ready interrupt.

Value	Description
0	The Result Ready interrupt is disabled
1	The Result Ready interrupt is enabled

33.6.12. Interrupt Enable Set

Name: INTENSET
Offset: 0x34
Reset: 0x00000000
Property: Local Write-Protection

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

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TRIGOVR Trigger Overrun Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Trigger Overrun Interrupt Enable bit, enabling the trigger overrun interrupt.

Value	Description
0	The Trigger Overrun interrupt is disabled
1	The Trigger Overrun interrupt is enabled

Bit 4 – SAMPOVR Sample Overwrite Interrupt Enable
Writing a '0'to this bit has no effect.
Writing a '1' to this bit sets the Sample Overwrite Interrupt Enable bit, enabling the sample overwrite interrupt.

Value	Description
0	The Sample Overwrite interrupt is disabled
1	The Sample Overwrite interrupt is enabled

Bit 3 – RESOVR Result Overwrite Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Result Overwrite Interrupt Enable bit, enabling the result overwrite interrupt.

Value	Description
0	The Result Overwrite interrupt is disabled

Value	Description
1	The Result Overwrite interrupt is enabled

Bit 2 – WCMP Window Comparator Interrupt Enable

Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Window Comparator Interrupt Enable bit, enabling the window comparator interrupt.

Value	Description
0	The Window Comparator interrupt is disabled
1	The Window Comparator interrupt is enabled

Bit 1 – SAMPRDY Sample Ready Interrupt Enable

Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Sample Ready Interrupt Enable bit, enabling the sample ready interrupt.

Value	Description
0	The Sample Ready interrupt is disabled
1	The Sample Ready interrupt is enabled

Bit 0 – RESRDY Result Ready Interrupt Enable

Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Result Ready Interrupt Enable bit, enabling the result ready interrupt.

Value	Description
0	The Result Ready interrupt is disabled
1	The Result Ready interrupt is enabled

33.6.13. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x38
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TRIGOVR Trigger Overrun Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set when a start trigger is received while a conversion is ongoing and will generate an interrupt if INTENCLR/SET.TRIGOVR is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Trigger Overrun interrupt flag.

Bit 4 – SAMPOVR Sample Overwrite Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set when an unread sample is overwritten in the Sample (SAMPLE) register and will generate an interrupt if INTENCLR/SET.SAMPOVR is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Sample Overwrite interrupt flag.

Bit 3 – RESOVR Result Overwrite Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set when an unread result is overwritten in the Result (RESULT) register and will generate an interrupt if INTENCLR/SET.RESOVR is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Result Overwrite interrupt flag.

Bit 2 – WCMP Window Comparator Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set when the conversion or accumulation is complete, and the thresholds match the selected window comparator source and mode, as configured in the Window Mode Source

and Window Comparator Mode bit fields in the Window Control register (WINCTRL.WINSRC and WINCTRL.WINMODE). When set, it will generate an interrupt if INTENCLR/SET.WCMP is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Window Comparator interrupt flag.

Bit 1 – SAMPRDY Sample Ready Interrupt Flag
This flag is cleared by writing a '1' to it or by reading the Sample (SAMPLE) register.
This flag is set when a conversion is complete, and a new sample is ready, and will generate an interrupt if INTENCLR/SET.SAMPRDY is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Sample Ready interrupt flag.

Bit 0 – RESRDY Result Ready Interrupt Flag
This flag is cleared by writing a '1' to it or by reading the Result (RESULT) register.
This flag is set when a conversion or accumulation is complete, and a new result is ready, and will generate an interrupt if INTENCLR/SET.RESRDY is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit clears the Result Ready interrupt flag.

33.6.14. Interrupt Flag Set

Name: INTFLAGSET
Offset: 0x3C
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TRIGOVR Trigger Overrun Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.TRIGOVR.
This flag is set when a start trigger is received while a conversion is ongoing and will generate an interrupt if INTENCLR/SET.TRIGOVR is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Trigger Overrun interrupt flag.

Bit 4 – SAMPOVR Sample Overwrite Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.SAMPOVR or by reading the Sample (SAMPLE) register.
This flag is set when an unread sample is overwritten in the Sample (SAMPLE) register and will generate an interrupt if INTENCLR/SET.SAMPOVR is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Sample Overwrite interrupt flag.

Bit 3 – RESOVR Result Overwrite Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.RESOVR or by reading the Result (RESULT) register..
This flag is set when an unread result is overwritten in the Result (RESULT) register and will generate an interrupt if INTENCLR/SET.RESOVR is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Result Overwrite interrupt flag.

Bit 2 – WCMP Window Comparator Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.WCMP.
This flag is set when the conversion or accumulation is complete, and the thresholds match the selected window comparator source and mode, as configured in the Window Mode Source

and Window Comparator Mode bit fields in the Window Control register (WINCTRL.WINSRC and WINCTRL.WINMODE). When set, it will generate an interrupt if INTENCLR/SET.WCMP is '1'
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Window Comparator interrupt flag.

Bit 1 – SAMPRDY Sample Ready Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.SAMPRDY.
This flag is set when a conversion is complete, and a new sample is ready, and will generate an interrupt if INTENCLR/SET.SAMPRDY is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Sample Ready interrupt flag.

Bit 0 – RESRDY Result Ready Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.RESRDY.
This flag is set when a conversion or accumulation is complete, and a new result is ready, and will generate an interrupt if INTENCLR/SET.RESRDY is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit sets the Result Ready interrupt flag.

33.6.15. Status

Name: STATUS
Offset: 0x40
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 0 – ADCBUSY ADC Busy
This bit indicates whether the ADC is busy performing a conversion or waiting for settling times due to configuration changes.

Value	Description
0	ADC is not busy
1	ADC is busy

33.6.16. Command

Name: COMMAND
Offset: 0x48
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 7 – DIFF Differential Mode
This bit controls whether the ADC conversion is Single-Ended or Differential.

Value	Description
0	Unsigned Single-Ended conversion. Only INPUTCTRL.MUXPOS is used.
1	Signed Differential conversion. Both INPUTCTRL.MUXPOS and INPUTCTRL.MUXNEG are used.

Bits 6:4 – MODE[2:0] Mode
This bit field controls the conversion mode for the ADC. Switching from one of the accumulation modes to Single mode will reset the accumulator.

Value	Name	Description
0x0	NONE	None
0x1	SINGLE	Single conversion
0x2	SERIES	Series mode with accumulation, using a separate trigger for each conversion
0x3	BURST	Burst mode with accumulation. A single trigger will initiate the number of conversions configured by CTRLD.SAMPNUM in one sequence.
0x7	ACCTEST	Accumulator diagnostics mode
Other	—	Reserved

Bits 2:0 – START[2:0] Start Conversion
This bit field starts or stops an ADC conversion, or controls how an ADC conversion will be initiated.

Value	Name	Description
0x0	STOP	Stop an ongoing conversion
0x1	IMMEDIATE	Start a conversion immediately. This bit will be reset to STOP when the conversion is complete, unless Free-Running mode is enabled.
0x2	INPUT	Start a conversion when a write to the INPUTCTRL register is performed

Value	Name	Description
0x4	EVENT	Start a conversion when an event is received by the ADC. This requires EVCTRL.STARTEI to be set to '1'.
Other	—	Reserved

Note: If CTRLA.ENABLE is '0' when this bit field is written to IMMEDIATE, it will automatically be set to STOP.

33.6.17. Result

Name: RESULT
Offset: 0x50
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
	RESULT[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
	RESULT[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
	RESULT[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
	RESULT[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 – RESULT[31:0] ADC Result

These bits can hold up to a 32-bit ADC conversion result, depending on the resolution and the number of samples in an accumulation mode.
If Accumulator Test mode (ACCTEST) is enabled in the Mode bit field of the Command register (COMMAND.MODE), data written to this register is written directly to the internal accumulator. Refer to the *Accumulator Test* section for more information.

33.6.18. Sample

Name: SAMPLE
Offset: 0x54
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	SAMPLE[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

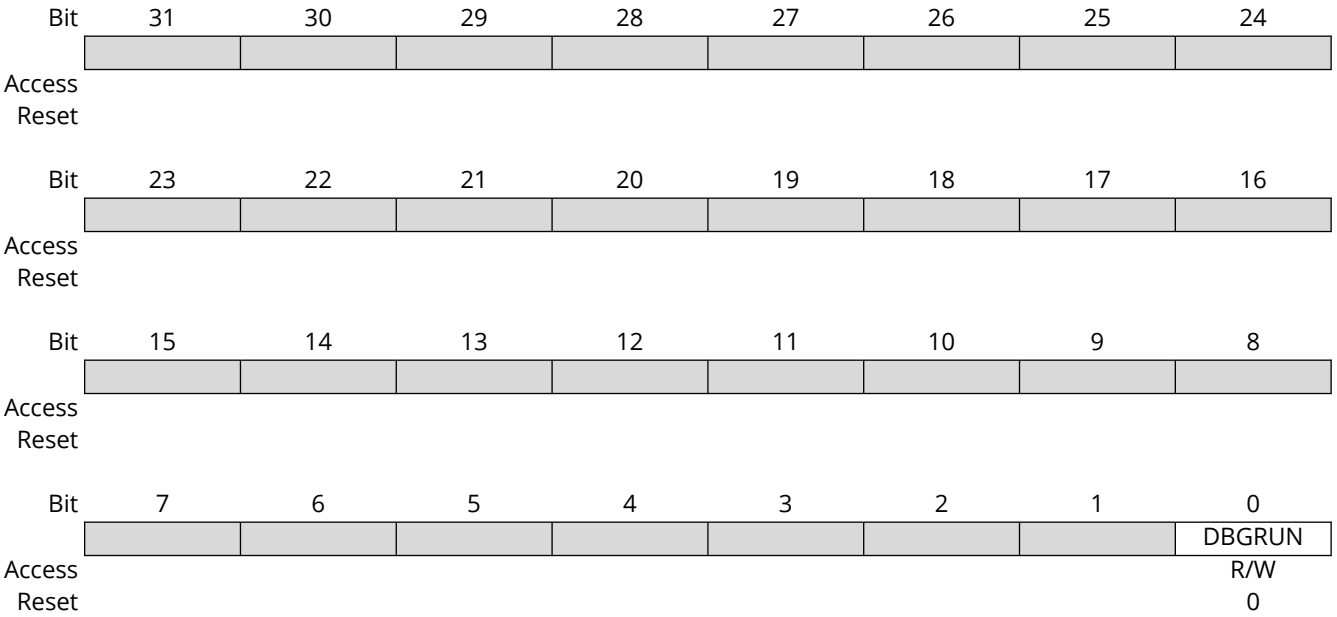
Bit	7	6	5	4	3	2	1	0
	SAMPLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – SAMPLE[15:0] ADC Sample

These bits will hold the ADC conversion result from last conversion.
If Accumulator Test mode (ACCTEST) is enabled in the Mode bit field in the Command Register (COMMAND.MODE), data written to this register is added to the internal accumulator. Refer to the *Accumulator Test* section for more information.

33.6.19. Debug Control

Name: DBGCTRL
Offset: 0x6C
Reset: 0x00000000
Property: Local Write-Protection



Bit 0 – DBGRUN Debug Run
This bit is not reset by a software reset.
This bit controls the functionality when the CPU is halted by an external debugger.

Value	Description
0	The ADC is halted when the CPU is halted by an external debugger. An ongoing conversion or burst accumulation will complete before the ADC stops.
1	The ADC continues normal operation when the CPU is halted by an external debugger

33.6.20. Write Protection Control

Name: WPCTRL
Offset: 0x7C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write to be successful.

Value	Name	Description
0x414443	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	The ADC register write protection is disabled
1	Write protection is enabled on ADC registers with the Local Write-Protection property. Non-debugger writes to ADC registers with Local Write-Protection are ignored and generate a bus error.

34. Temperature Sensor

An on-chip temperature sensor is available. To do a temperature measurement, the internal ADC peripheral acquires the sensor output. Using correction values from the System Controller (SYSCTRL) fuses, the application can calculate a temperature value in kelvin.

Follow these steps:

1. Configure the ADC voltage reference to internal 2.048V by writing to the Reference Selection (REFSEL) bit field in the Control C register of the ADC (ADC.CTRLC).
2. In the ADC Input Control (ADC.INPUTCTRL) register select the temperature sensor (TSENSE) as input in the Positive Input Multiplexer (MUXPOS) bit field.
3. Configure the ADC Sample Length by writing a value ≥ $32\ \mu s \times f_{CLK\_ADC}$ to the Sample Length (SAMPLEN) bit field in the Control E (ADC.CTRLE) register.
4. Acquire the temperature sensor output voltage by running a 12-bit single-ended conversion using the ADC.COMMAND register.
5. Process the measurement result, as described below.

The measured voltage has a linear relationship to the temperature. Due to process variations, the temperature sensor output voltage varies between individual devices at the same temperature. The individual compensation factors determined during the production test are stored in these System Controller (SYSCTRL) fuses:

- SYSCTRL.TEMPSENSE0 is a gain/slope correction
- SYSCTRL.TEMPSENSE1 is an offset correction

The calibration values can be used to improve the accuracy of the result. To achieve even more accurate results, it is possible to perform further calibration. For example, multi-point calibration is performed by imposing known temperatures on the device and recording the ADC results. This helps create a precise calibration curve which can be implemented in firmware to adjust the output result.

Use the following equation to calculate the temperature (in kelvin):

$$T = \frac{(\text{Offset Correction} - \text{ADC Result}) \times \text{Gain Correction}}{4096}$$

Example 34-1. Using the Calibration Values to Improve the Accuracy of the Temperature Readings

```
//Prerequisite: Configure ADC0 as described in the references

uint16_t sigrow_gain = SYSCTRL.TEMPSENSE0;    // Read unsigned gain/slope from
System Controller fuse
int16_t sigrow_offset = SYSCTRL.TEMPSENSE1;   // Read signed offset from
System Controller fuse

ADC_COMMAND_START
while (!(ADC0->INTFLAG.reg & ADC_INTFLAG_RESRDY)) {
        // Wait for result ready
    }

uint16_t adc_reading = ADC0.RESULT >> 2;    // 10-bit MSb of ADC result with
2.048V internal reference

uint32_t temp = adc_reading - sigrow_offset;
temp *= sigrow_gain;        // Result might overflow 16-bit variable (10-bit +
8-bit)
temp += 0x80;               // Add 256/2 to get correct integer rounding on
division below
temp >>= 12;                 // Divide result by 4096 to get processed
```

```
temperature in kelvin
uint16_t temperature_in_K = temp;
```

Refer to the *Electrical Characteristics* and the *ADC - Analog-Digital Converter* chapters for details on the configuration of the ADC for temperature measurements.

35. PTC - Peripheral Touch Controller

35.1. Features

- Low-Power, High-Sensitivity, Environmentally Robust Capacitive Touch Buttons, Sliders, Wheels, and 2D Surface
- Supports Mix-and-Match Mutual Capacitance and Self-Capacitance Sensing
  - Supports 29/25/20/16 PTC pins, for the 64/48/32/28 pin variants, respectively
- Supports Lumped Mode, Which Allows the Integration of Multiple Sensors, Configuring Them to Function as a Single Sensor
- One Pin per Electrode—No External Components Required
- Load Compensating Charge Sensing:
  - Parasitic capacitance compensation
  - Adjustable gain for superior sensitivity
- Zero Drift Over the Temperature and V_{DD} Range:
  - Auto-calibration and recalibration of sensors
- Hardware Noise Filtering and Noise Signal Desynchronization for High Conducted Immunity
- Driven Shield+ for Superior Noise Immunity and Moisture Tolerance in Self-Capacitance Systems
  - Any PTC X/Y line can be used for the Driven Shield
- Supports External Integration Capacitors (EXTCINT) for Large Capacitance Sensors
- Boost Mode for Doubled Signal-to-Noise Ratio or 4x Faster Response Time in Mutual Capacitance Systems
- Supported by the MPLAB® Code Configurator (MCC) Touch Configurator Development Tools

35.2. Overview

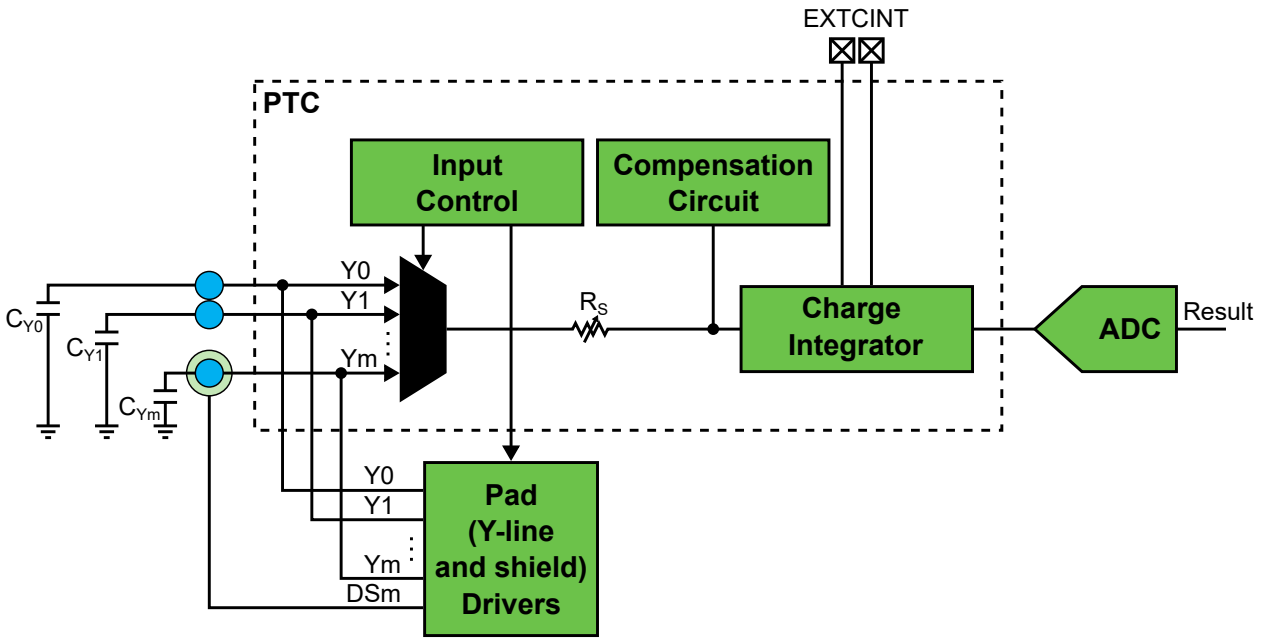
The Peripheral Touch Controller (PTC) detects touch inputs on capacitive sensors. The external capacitive touch sensor is typically formed on a PCB, and the sensor electrodes are connected to the analog front end of the PTC through the device's I/O pins. The PTC supports both mutual and self-capacitance sensors, which can be configured as buttons, wheels or sliders.

A self-capacitance sensor consists of a conductor plate with one wire connected directly to an I/O pad. In Self-Capacitance mode, the PTC requires one pin (Y-line) for each touch sensor, as shown in Figure 35-1.

A mutual capacitance sensor consists of two conductor plates, each connected to an I/O pad. The mutual capacitance between the two plates is measured by driving the first plate and sensing the second one. In Mutual Capacitance mode, sensing is performed using capacitive touch matrices that support various X-Y configurations. The PTC requires one pin per X-line and one pin per Y-line, as shown in Figure 35-2.

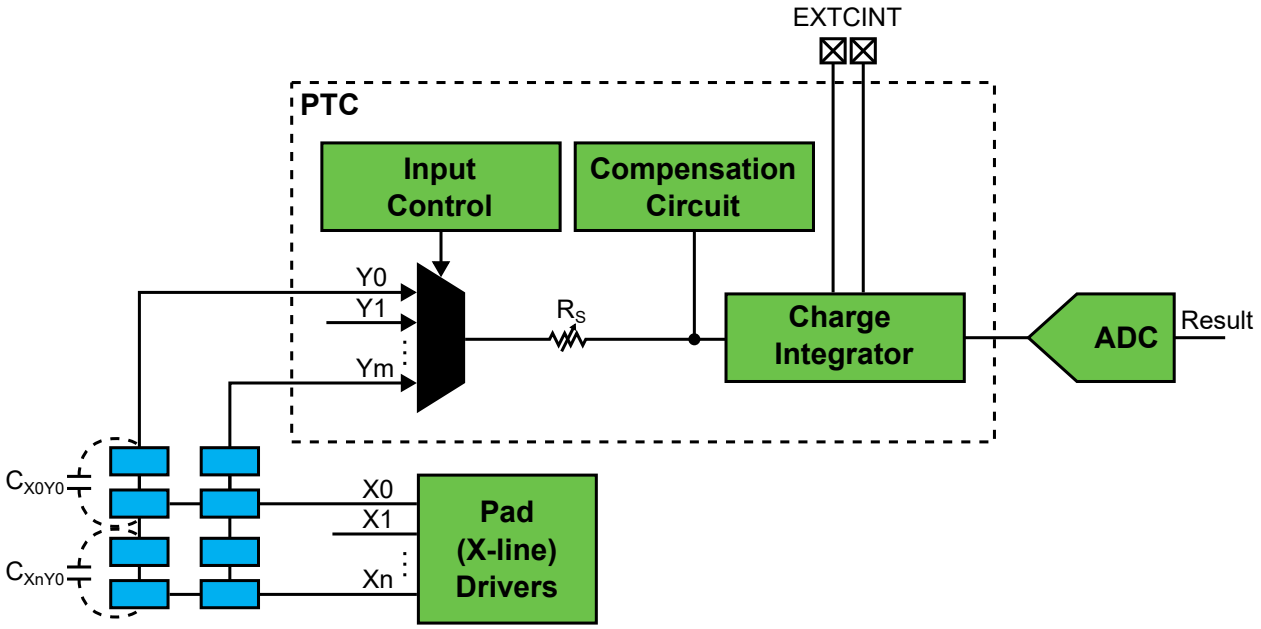
35.3. Block Diagram

Figure 35-1. PTC Block Diagram in Self-Capacitance Mode



Note: The internal series resistor, R_S, has limited effect in Self-Capacitance mode. It is always recommended to add an external series resistor.

Figure 35-2. PTC Block Diagram in Mutual Capacitance Mode



35.4. Signal Description

Table 35-1. Signal Description

Name	Type	Description
Y[m:0]	Analog	Y-line (Input/Output)

Table 35-1. Signal Description (continued)

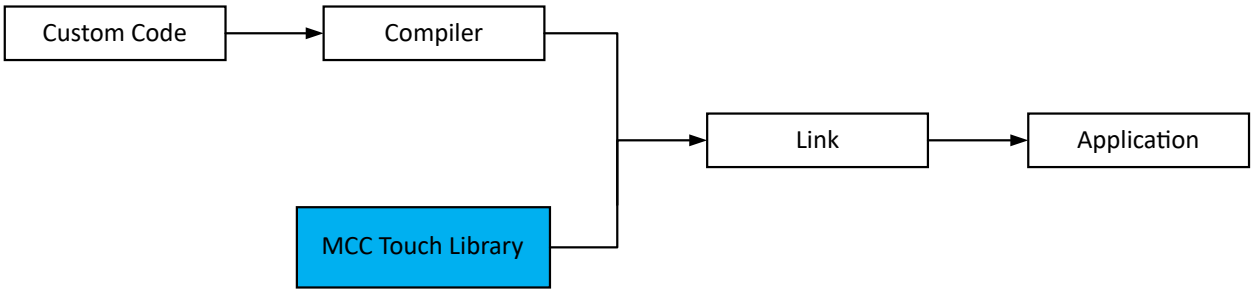
Name	Type	Description
X[n:0]	Digital	X-line (Output)
EXTCINT[1:0]	Analog	External Integration Capacitor (Input/ Output)

Note: The EXTCINT pins are used to connect an external capacitor to the charge integrator when working with high-capacitance sensors.

35.5. Functional Description

To access the PTC, the *MCC Touch Configurator* must be used to configure the *Touch Library* and link it to the application software. The MCC Touch Library can be used to implement buttons, sliders and wheels in various combinations on a single interface.

Figure 35-3. MCC Touch Library Usage



For more information about MCC Touch Library, refer to the "QTouch® Modular Library Peripheral Touch Controller User's Guide (DS40001986)".

35.5.1. Basic Operation

35.5.1.1. Initialization

The ADC used must be enabled and configured appropriately to ensure correct behavior of the PTC, which is managed by the MCC Touch library. Refer to the *ADC – Analog-to-Digital Converter* section for further details.

35.5.1.2. Enabling, Disabling and Resetting

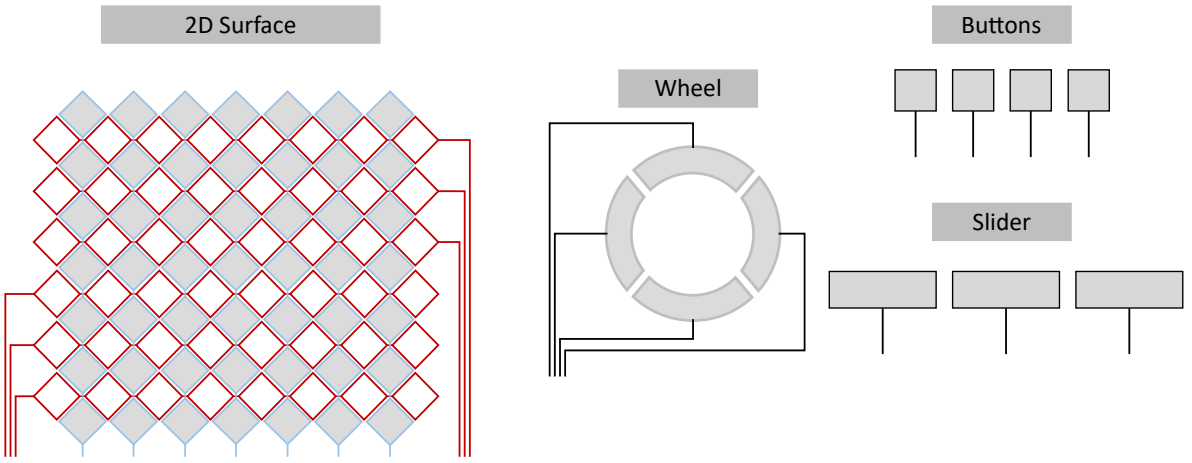
The PTC is enabled using the MCC Touch library. Refer to the *QTouch® Modular Library Peripheral Touch Controller User's Guide (DS40001986)*.

35.5.1.3. Sensor Configurations

The PTC supports different sensor configurations with its Self- and Mutual Capacitance modes. The mode to use depends on the type of sensor and whether single or multiple simultaneous touch points are required.

In addition, the number of available pins and the assignment of X- and Y-lines depend on both package type and device configuration.

Figure 35-4. Example of Supported Touch Sensors

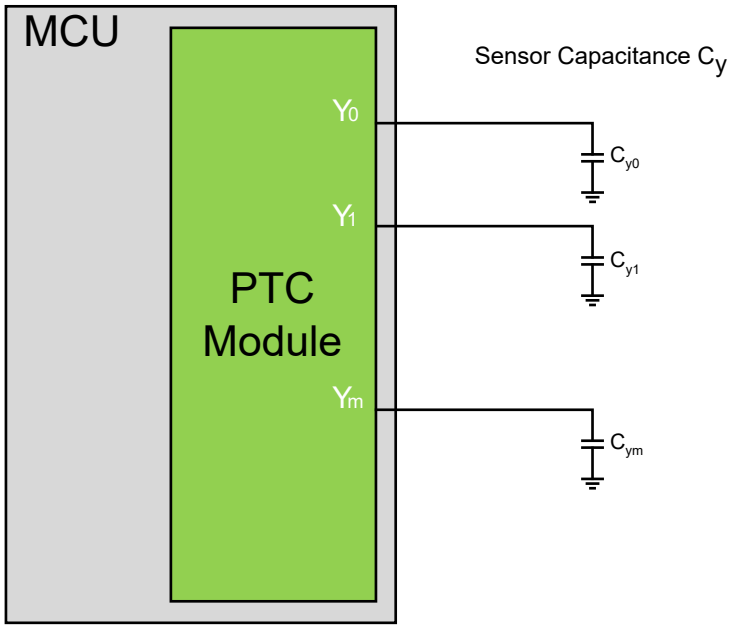


35.5.1.3.1. Self-Capacitance Sensor Arrangement

A self-capacitance sensor is connected to a single pin on the PTC through the Y-electrode. The PTC measures the capacitance of the sense electrode.

For a capacitive matrix array where only a single touch point is needed, self-capacitance sensing on both vertical and horizontal lines can effectively identify the touch point by locating the intersection of the touched lines.

Figure 35-5. Sensor Capacitance Sensor Arrangement



For more information about designing the touch sensor, refer to the AN2934—Capacitive Touch Sensor Design Guide (DS00002934).

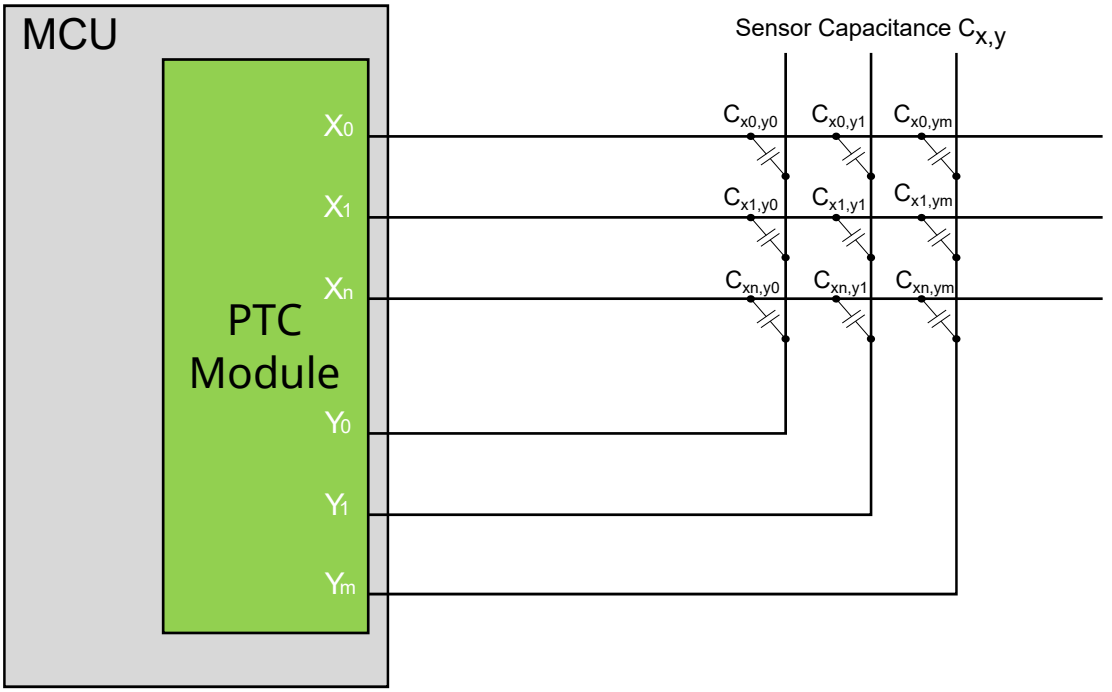
35.5.1.3.2. Mutual Capacitance Sensor Arrangement

For multiple touch points, self-capacitance scanning alone is insufficient for accurately determining their exact positions. To precisely detect multiple touch points, mutual-capacitance sensing is required.

A mutual capacitance sensor is formed between two I/O lines (X- and Y- electrodes) running either horizontally or vertically, with the intersection working as capacitors plates. The horizontal and vertical lines do not physically connect, forming a mutual capacitance sensor between the X- and Y-electrodes measured by the PTC.

A good way to reduce the number of I/O lines needed to connect a large number of sensors is to use a mutual capacitance sensor array.

Figure 35-6. Mutual Capacitance Sensor Arrangement



35.5.2. Sleep Mode Operation

Sleep Mode Operation for the PTC follows the operation of the ADC. See the *ADC - Analog-to-Digital Converter* chapter for more information.

35.5.3. Debug Operation

The PTC does not have separate debug signals, but follows the operation of the ADC. See the *ADC - Analog-to-Digital Converter* chapter for more information.

35.6. Dependencies

35.6.1. I/O Lines

Using the I/O lines of the PTC requires the I/O pins to be configured.

The I/O lines used for analog X- and Y-lines must be connected to external capacitive touch sensor electrodes. External components are not necessary for normal operation. However, to reduce the effects of electromagnetic interference, a series resistor of up to 100 kΩ can be added to the Y-lines. In Mutual Capacitance mode, a series resistor of up to 10 kΩ can be added to the X-lines to reduce EMC emissions.

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

35.6.2. Power Management

Power Management of the PTC follows the ADC. The PTC is disabled when not in use by the ADC. Refer to the *PM – Power Manager* chapter for details on the different sleep modes.

35.6.3. Clocks

The PTC bus clock (CLK_PTC_APB) can be enabled and disabled in the Power Manager, and the default state of CLK_PTC_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

35.6.4. Analog Connections

The PTC utilizes the in-built 12-bit ADC for signal acquisition and conversion.

When the PTC is enabled, it takes control of the ADC. If the application also requires the ADC for purposes other than touch sensing, the MCC Touch library provides APIs to facilitate multiplexing of the ADC between the PTC and non-touch applications.

When the PTC is disabled, the ADC becomes available for normal operation.

35.6.5. Events

Not applicable.

35.6.6. Interrupts

Not applicable.

36. AC - Analog Comparators

36.1. Features

- Two Individual Comparators
- Hysteresis, on or off
- Analog Comparator Outputs Available on Pins
- Flexible Comparator Input Selection:
  - 6 positive pins
  - 3 negative pins
  - Internal reference voltage generator
- Interrupt Generation on:
  - Rising edge
  - Falling edge
  - Both edges
- Window Function Interrupt Generation when:
  - Signal goes above window
  - Signal goes inside window
  - Signal goes below window
  - Signal goes outside window
- Event Generation on:
  - Comparator output
  - Window function inside/outside window

36.2. Overview

The Analog Comparators (AC) peripheral supports multiple individual comparators. Each comparator (COMPn) continuously compares the voltage levels on two inputs and provides a digital output based on this comparison. Each comparator can be configured to generate interrupt requests and peripheral events in response to various input change conditions.

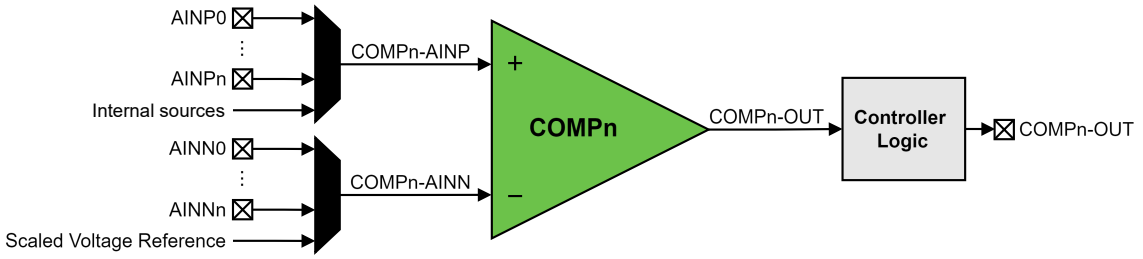
The input selection includes analog port pins and internally generated signals. Each comparator's output state can also be output on a pin for use by external devices.

The dynamic behavior of each comparator can be adjusted using a hysteresis feature.

The individual comparators can be used independently (Normal mode) or paired to form a window comparison (Window mode).

36.3. Block Diagram

Figure 36-1. Comparator n Block Diagram



36.3.1. Signal Description

Signal	Description	Type
COMPn-AINN	Comparator n negative input	Analog input
COMPn-AINP	Comparator n positive input	Analog input
COMPn-OUT	Comparator n output	Digital output

For the mapping between the AC signals and I/O pins, refer to the *Pinout and Multiplexing* section in the *Pinout* chapter. The external voltage reference pin (EXTVREF) is named VREFA in this table.

36.4. Functional Description

36.4.1. Initialization

For basic operation, follow these steps:

1. Configure the positive and negative input sources of COMPn by writing to the Positive and Negative Input Selection bit fields in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.MUXPOS/MUXNEG).
2. Enable COMPn by writing a '1' to the Comparator Enable bit in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.ENABLE).
3. Enable the AC by writing a '1' to the Global Enable bit in the Control A register (CTRLA.ENABLE).

36.4.2. Operation

36.4.2.1. Enabling, Disabling and Resetting

The AC is enabled by writing a '1' to the Global Enable bit in the Control A register (CTRLA.ENABLE). The AC is disabled by writing a '0' to CTRLA.ENABLE.

The AC is reset by writing a '1' to the Software Reset bit in the Control A register (CTRLA.SWRST). All registers in the AC will be reset to their initial state, and the AC will be disabled. Refer to the CTRLA register description for details.

The individual comparators are enabled by writing a '1' to the Comparator Enable bit in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.ENABLE). The individual comparators are disabled by writing a '0' to COMP[n].COMPCTRLA.ENABLE. Writing a '0' to CTRLA.ENABLE will also disable all the comparators, but will not clear their COMP[n].COMPCTRLA.ENABLE bits.

Note: During the start-up time after enabling comparator n, the output of COMPn may be invalid. Refer to the *Electrical Characteristics* section for details about the start-up time.

36.4.2.2. Selecting Comparator Inputs

Each comparator has one positive (COMPn-AINP) and one negative (COMPn-AINN) input, which may be chosen from a selection of shared analog I/O pins and internal inputs. COMPn-AINP and COMPn-AINN are configured by writing to the Positive and Negative Input Selection bit fields in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.MUXPOS/MUXNEG), respectively. For details about the possible input selections, refer to the MUXPOS and MUXNEG bits in the COMP[n].COMPCTRLA register description.

Notes:

1. Before using any I/O pins as comparator inputs, they must be configured as analog inputs. Refer to *I/O Lines* for details.
2. After switching inputs to I/O pins, COMPn requires time to settle. Refer to the *Electrical Characteristics* section for more details.

36.4.2.3. Comparator Output

The digital output from any comparator (COMPn-OUT) is '1' when the difference between the positive and the negative input voltage is positive, and '0' otherwise. The result of the latest comparison is available in the Comparator n State bit in the Status register (STATUS.COMPSTATEn).

The output of each comparator can be routed to an I/O pin for use by external devices by writing a '1' to the Output Enable bit field in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.OUT). When enabled, the output appears on the corresponding COMPn-OUT pin.

Note: Before using any I/O pins as comparator output, they must be configured as digital outputs. Refer to *I/O Lines* for details.

36.4.3. Additional Features

36.4.3.1. 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 writing a '1' to the Hysteresis Enable bit in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.HYSTEN).

36.4.3.2. Voltage Reference Selection

For each comparator, the reference voltage scaler generates a voltage ($V_{REFSCALE}$) available as an input, based on the following formula:

$$V_{REFSCALE} = V_{REF} \times \frac{VALUE}{2^{SIZE\_REFSCALE}}$$

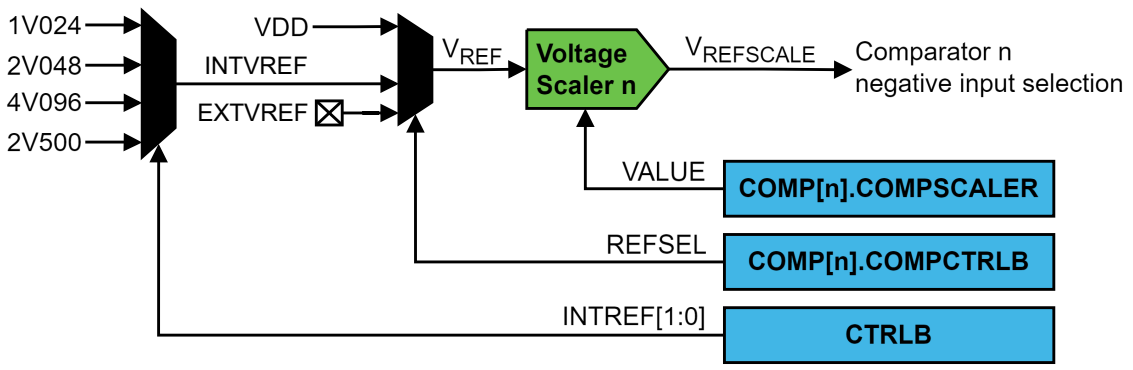
where V_{REF} is the reference voltage for the comparator, VALUE is the scaling factor for the comparator, and SIZE_REFSCALE = 8.

As shown in the following figure, the reference voltage (V_{REF}) is selected by writing to the Reference Selection bit field in the Comparator n Comparator Control B register (COMP[n].COMPCTRLB.REFSEL), and may be chosen from V_{DD}, internal voltage references (INTVREF), or an external reference pin (EXTVREF). The internal reference option is configured by writing to the Internal Reference Selection bit field in the Control B register (CTRLB.INTREF). The reference voltages, except for V_{DD} and EXTVREF, are generated from an internal band gap reference.

The scaling factor (VALUE) is configured by writing to the Scaled Reference Voltage bit field in the Comparator n Comparator Scaler register (COMP[n].COMPSCALER.VALUE).

Note: After selecting a new voltage reference, COMPn requires time to settle. Refer to the *Electrical Characteristics* section for more details.

Figure 36-2. Voltage Reference Selection



36.4.3.3. Window Mode

Two comparators can be configured to work together in Window mode. In this mode, a voltage range (the window) is defined, and the comparators indicate whether an input signal is within this range or not. Window mode is enabled by writing to the Window n Enable bit in the Window Control register (WINCTRL.WENn).

To operate in Window mode, the two comparators must be configured as shown in the following figure by writing to the Positive and Negative Input Selection bit fields in their respective Comparator n Comparator Control A registers (COMP[n].COMPCTRLA.MUXPOS/MUXNEG). For Window mode n, the even comparator (COMP(2n)) defines the lower limit and the odd comparator (COMP(2n+1)) defines the upper limit of the window. Note that the two comparators must use the same positive input source.

The state of the window function is available in the Window n State bit field in the Status register (STATUS.WINSTATEn). The state can be:

- Above window - the input signal is above the upper limit
- Inside window - the input signal is between the lower and upper limits
- Below window - the input signal is below the lower limit

The Window mode can be configured to request interrupts on one of the following conditions by writing to the Window n Interrupt Selection bit field in the Window Control register (WINCTRL.WINTSELn):

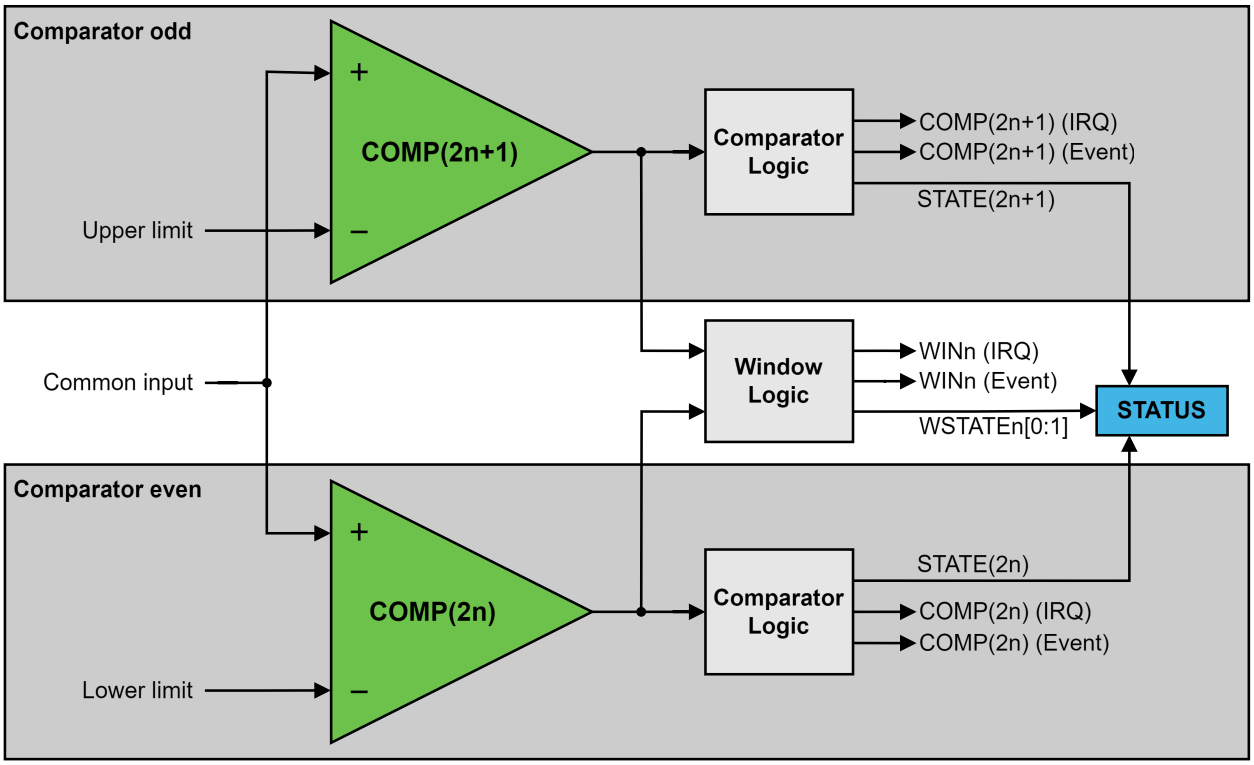
- Above window - the interrupt is issued on the edge when the input signal goes above the upper limit
- Inside window - the interrupt is issued on the edge when the input signal goes between the lower and upper limits
- Below window - the interrupt is issued on the edge when the input signal goes below the lower limit
- Outside window - the interrupt is issued on the edge when the input signal goes below the lower limit or above the upper limit

The window n interrupt is enabled by writing to the Window n Interrupt bit in the Interrupt Enable Set register (INTENSET.WINn). Once enabled, the corresponding interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.WINn) will be set when the input signal crosses one of the limits according to the configuration. For example, if WINCTRL.WINTSELn is configured to ABOVE, an interrupt will be generated when the signal exceeds the upper window limit, and INTFLAG.WINn will be set. If the interrupt flag is cleared while the signal remains above the window, the flag will not be set again until the signal falls below the upper limit and then exceeds it once more. The window n interrupt is disabled by writing to the Window n Interrupt bit in the Interrupt Enable Clear register (INTENCLR.WINn).

In addition, the Window mode can generate an event whenever the input signal is inside the window. The window n event has no configuration and is enabled by writing to the Window n Event Output Enable bit in the Event Control register (EVCTRL.WINEOn).

Note: The individual comparator outputs, interrupts and events continue to function normally during Window mode.

Figure 36-3. Comparators in Window Mode n



36.4.4. Sleep Mode Operation

In Idle sleep mode, each comparator will continue to operate as normal.

In Standby sleep mode, the behavior of COMPn is controlled by the Run in Standby bit field in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.RUNSTDBY). By default, COMPn is disabled during Standby sleep mode but retains its current configuration. Events and interrupts will not be triggered, and COMPn-OUT is not available on the pin. When COMP[n].COMPCTRLA.RUNSTDBY is written to '1', COMPn continues to operate as normal during Standby sleep mode, even if the clock is not running. Note that when COMP[n].COMPCTRLA.RUNSTDBY is '0', the analog blocks are powered off to achieve the lowest power consumption. As a result, a start-up delay is required when the system returns from sleep.

For Window mode operation, both comparators in a pair must have the same configuration in COMP[n].COMPCTRLA.RUNSTDBY.

When COMP[n].COMPCTRLA.RUNSTDBY is '1', any enabled AC interrupt source can wake up the CPU.

Refer to the *PM – Power Manager* chapter for more information about sleep mode operation.

36.4.5. Debug Operation

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

36.5. Dependencies

36.5.1. I/O Lines

Each comparator in the AC has multiple I/O pins that can be used as analog inputs or external voltage references. These pins should be configured for analog operation before using them as COMPn inputs.

Each comparator has one or more I/O pins that can be used as digital output. To avoid the pin being tri-stated when COMPn is disabled or in sleep mode, the COMPn-OUT pin must be configured as an output.

Refer to the *Pinout and Multiplexing* section in the *Pinout* chapter and the *PORT – I/O Pin Controller* chapter for details.

36.5.2. 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. Events connected to the event system can trigger other operations in the system without exiting sleep modes. Refer to the *PM – Power Manager* chapter for details on the different sleep modes.

36.5.3. Clocks

The AC bus clock (CLK_AC_APB) can be enabled and disabled in the Main Clock Controller, and the default state of CLK_AC_APB can be found in the *Peripheral Clock Masking* section in the *MCLK – Main Clock* chapter.

36.5.4. DMA

Not applicable.

36.5.5. Interrupts

The interrupt request line, also known as the interrupt vector, is connected to the interrupt controller. To use AC interrupts, the interrupt controller must be configured in advance, including enabling the interrupt line globally. For further information, refer to the *NVIC - Nested Vectored Interrupt Controller* section.

Each interrupt source has an interrupt flag which is in the Interrupt Flag Status and Clear (INTFLAG) register. The flag is set when the interrupt condition occurs. Each interrupt can be individually enabled by writing a '1' to the corresponding bit in the Interrupt Enable Set (INTENSET) register, and disabled by writing a '1' 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 source is enabled. The interrupt request remains active until the interrupt flag is cleared, the interrupt is disabled, or the peripheral is reset. Refer to the INTFLAG register description for details on how to clear interrupt flags.

All interrupt requests from the peripheral are ORed together on system level to generate a single combined interrupt request to the NVIC. Therefore, the INTFLAG register must be read to determine what the interrupt condition is.

Table 36-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
AC	COMPn	The comparator n output level matches the configuration	The Interrupt Selection bit field in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.INTSEL)
	WINn	The window n output level matches the configuration	The Window n Interrupt Selection bit field in the Window Control register (WINCTRL.WINTSEL)

36.5.6. Events

The AC can generate the following events:

Table 36-2. Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
AC	COMP_n	Comparator n status	Level	Asynchronous	While COMPn-OUT is '1'
	WIN_n	Window n inside/ outside status	Level	Asynchronous	While the input signal is inside window n

Writing a '1' to an Event Output bit in the Event Control register (EVCTRL.xxEOn) enables the corresponding output event. Writing a '0' to this bit disables the corresponding output event.

The AC has no event users or actions.

Refer to the *EVSYS – Event System* chapter for details on configuring the event system.

36.5.7. Analog Connections

Each comparator in the AC has internal voltage references that can be used as inputs. Before using an internal voltage reference as a comparator input, it must be configured and enabled.

36.6. Register Summary - AC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0							ENABLE	SWRST
		15:8								
		23:16								
		31:24								
0x04	CTRLB	7:0							INTREF[1:0]	
		15:8								
		23:16								
		31:24								
0x08 ... 0x13	Reserved									
0x14	WINCTRL	7:0						WINTSEL0[1:0]		WEN0
		15:8								
		23:16								
		31:24								
0x18 ... 0x1F	Reserved									
0x20	EVCTRL	7:0				WINEO0			COMPEO1	COMPEO0
		15:8								
		23:16								
		31:24								
0x24	INTENCLR	7:0				WIN0			COMP1	COMP0
		15:8								
		23:16								
		31:24								
0x28	INTENSET	7:0				WIN0			COMP1	COMP0
		15:8								
		23:16								
		31:24								
0x2C	INTFLAG	7:0				WIN0			COMP1	COMP0
		15:8								
		23:16								
		31:24								
0x30	INTFLAGSET	7:0				WIN0			COMP1	COMP0
		15:8								
		23:16								
		31:24								
0x34	STATUS	7:0			WINSTATE0[1:0]				COMPSTATE1	COMPSTATE0
		15:8								
		23:16								
		31:24								
0x38 ... 0x3F	Reserved									
0x40	COMP[0].COMPCTRLA	7:0		RUNSTDBY	INTSEL[1:0]		HYSTEN	OUT	ENABLE	
		15:8		MUXPOS[2:0]					MUXNEG[2:0]	
		23:16								
		31:24								
0x44	COMP[0].COMPCTRLB	7:0							REFSEL[1:0]	
		15:8								
		23:16								
		31:24								
0x48	COMP[0].COMPSCALER	7:0	VALUE[7:0]							
		15:8								
		23:16								
		31:24								

AC (continued)

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x4C ... 0x4F	Reserved									
0x50	COMP[1].COMPCTRLA	7:0		RUNSTDBY	INTSEL[1:0]		HYSTEN	OUT	ENABLE	
		15:8		MUXPOS[2:0]				MUXNEG[2:0]		
		23:16								
		31:24								
0x54	COMP[1].COMPCTRLB	7:0							REFSEL[1:0]	
		15:8								
		23:16								
		31:24								
0x58	COMP[1].COMPSCALER	7:0	VALUE[7:0]							
		15:8								
		23:16								
		31:24								
0x5C ... 0x7F	Reserved									
0x80	WPCTRL	7:0							WPLCK	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

36.6.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

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

Bit 1 – ENABLE Global Enable
This bit controls whether the AC is enabled.

Value	Description
0	The AC is or being disabled
1	The AC is or being enabled

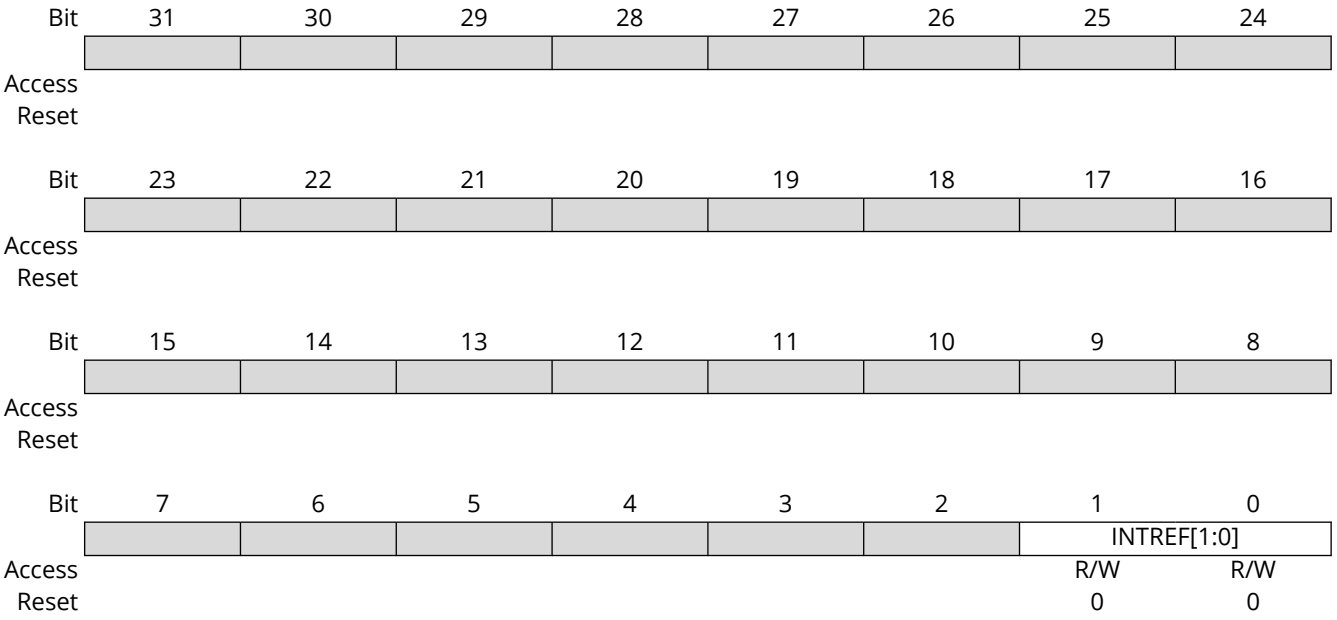
Note: Each comparator must also be enabled individually by the Enable (ENABLE) bit in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA).

Bit 0 – SWRST Software Reset
Writing a '0' to this bit has no effect.
Writing a '1' to this bit resets all registers in the AC to their initial state, and the AC will be disabled.
This bit will always read as '0'.

Note: Writing a '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded.

36.6.2. Control B

Name: CTRLB
Offset: 0x04
Reset: 0x00000000
Property: Local Write-Protection



Bits 1:0 – INTREF[1:0] Internal Reference Selection

This bit field controls the internal reference voltage of the AC. This configuration is common for all comparators using the internal reference.

Value	Name	Description
0x0	1V024	Internal 1.024V reference
0x1	2V048	Internal 2.048V reference
0x2	4V096	Internal 4.096V reference
0x3	2V500	Internal 2.500V reference

36.6.3. Window Control

Name: WINCTRL
Offset: 0x14
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						WINTSEL0[1:0]		WEN0
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 1:2 – WINTSELn[1:0] Window n Interrupt Selection
This bit field controls the configuration of the interrupt mode for window n.

Value	Name	Description
0x0	ABOVE	Interrupt when the signal goes above the window
0x1	INSIDE	Interrupt when the signal goes inside the window
0x2	BELOW	Interrupt when the signal goes below the window
0x3	OUTSIDE	Interrupt when the signal goes outside the window

Bit 0 – WENn Window n Enable

Value	Description
0	Window mode is disabled for comparators 2n and 2n+1
1	Window mode is enabled for comparators 2n and 2n+1

36.6.4. Event Control

Name: EVCTRL
Offset: 0x20
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				WINEO0			COMPEO1	COMPEO0
Access				R/W			R/W	R/W
Reset				0			0	0

Bit 4 – WINEOn Window n Event Output Enable
This bit controls whether the window n function can generate a peripheral event.

Value	Description
0	The Window n event is disabled
1	The Window n event is enabled

Bits 0, 1 – COMPEOn Comparator n Event Output Enable
This bit controls whether the comparator n output can generate a peripheral event.

Value	Description
0	The Comparator n event is disabled
1	The Comparator n event is enabled

36.6.5. Interrupt Enable Clear

Name: INTENCLR
Offset: 0x24
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				WIN0			COMP1	COMP0
Access				R/W			R/W	R/W
Reset				0			0	0

Bit 4 – WINn Window n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Window n Interrupt Enable bit, which disables the Window n interrupt.

Value	Description
0	The Window n interrupt is disabled
1	The Window n interrupt is enabled

Bits 0, 1 – COMPn Comparator n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Comparator n Interrupt Enable bit, which disables the Comparator n interrupt.

Value	Description
0	The Comparator n interrupt is disabled
1	The Comparator n interrupt is enabled

36.6.6. Interrupt Enable Set

Name: INTENSET
Offset: 0x28
Reset: 0x00000000
Property: Local Write-Protection

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.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				WIN0			COMP1	COMP0
Access				R/W			R/W	R/W
Reset				0			0	0

Bit 4 – WINn Window n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Window n Interrupt Enable bit, which enables the Window n interrupt.

Value	Description
0	The Window n interrupt is disabled
1	The Window n interrupt is enabled

Bits 0, 1 – COMPn Comparator n Interrupt Enable
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Comparator n Interrupt Enable bit, which enables the Comparator n interrupt.

Value	Description
0	The Comparator n interrupt is disabled
1	The Comparator n interrupt is enabled

36.6.7. Interrupt Flag Status and Clear

Name: INTFLAG
Offset: 0x2C
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				WIN0			COMP1	COMP0
Access				R/W			R/W	R/W
Reset				0			0	0

Bit 4 – WINn Window n Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set according to the Window n Interrupt Selection bit field in the Window Control register (WINCTRL.WINTSELn) and will generate an interrupt if INTENCLR/SET.WINn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Window n interrupt flag.

Bits 0, 1 – COMPn Comparator n Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set according to the Interrupt Selection bit field in the Comparator n Comparator Control A register (COMP[n].COMPCTRLA.INTSEL) and will generate an interrupt if INTENCLR/SET.COMPn is '1'.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Comparator n interrupt flag.

36.6.8. Interrupt Flag Set

Name: INTFLAGSET
Offset: 0x30
Reset: 0x00000000
Property: Local Write-Protection

Notes:

1. This register allows the user to manually set an interrupt flag by software to test the ISR, and should be used solely for that purpose.
2. Changes in this register will also be reflected in the Interrupt Flag Status and Clear (INTFLAG) register.
3. Reading a bit in this register returns the status of the corresponding bit in the INTFLAG register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				WIN0			COMP1	COMP0
Access				R/W			R/W	R/W
Reset				0			0	0

Bit 4 – WINn Window n Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.WINn.
This flag is set by writing a '1' to this bit.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Window n interrupt flag.

Bits 0, 1 – COMPn Comparator n Interrupt Flag
This flag is cleared by writing a '1' to INTFLAG.COMPn.
This flag is set by writing a '1' to this bit.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will set the Comparator n interrupt flag.

36.6.9. Status

Name: STATUS
Offset: 0x34
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			WINSTATE0[1:0]				COMPSTATE1	COMPSTATE0
Access			R	R			R	R
Reset			0	0			0	0

Bits 4:5 – WINSTATEn[1:0] Window n State
This bit field indicates the current state of the input signal if Window n mode is enabled.

Value	Name	Description
0x0	ABOVE	The input signal is above window n
0x1	INSIDE	The input signal is inside window n
0x2	BELOW	The input signal is below window n
Other	—	Reserved

Bits 0, 1 – COMPSTATEn Comparator n State
This bit is cleared when the positive input voltage is lower than or equal to the negative input voltage.
This bit is set when the positive input voltage is higher than the negative input voltage.

Value	Description
0	The COMPn-OUT signal is low
1	The COMPn-OUT signal is high

36.6.10. Comparator n Comparator Control A

Name: COMP[n].COMPCTRLA
Offset: 0x40 + n*0x10 [n=0..1]
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
		MUXPOS[2:0]				MUXNEG[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		RUNSTDBY	INTSEL[1:0]		HYSTEN	OUT	ENABLE	
Access		R/W	R/W	R/W	R/W	R/W	R/W	
Reset		0	0	0	0	0	0	

Bits 14:12 – MUXPOS[2:0] Positive Input Selection
This bit field defines the input signal to the positive input of comparator n.

Value	Name	Description
0x0	AINP0	Positive Pin 0
0x1	AINP1	Positive Pin 1
0x2	AINP2	Positive Pin 2
0x3	AINP3	Positive Pin 3
0x5	AINP5	Positive Pin 5
0x6	AINP6	Positive Pin 6
0x7	VDDIO2DIV10	VDDIO2 divided by 10
Other	—	Reserved

Bits 10:8 – MUXNEG[2:0] Negative Input Selection
This bit field defines the input signal to the negative input of comparator n.

Value	Name	Description
0x0	AINN0	Negative Pin 0
0x1	AINN1	Negative Pin 1
0x2	AINN2	Negative Pin 2
0x7	VREFSCALE	Voltage Reference Scaler
Other	—	Reserved

Bit 6 – RUNSTDBY Run in Standby
This bit controls whether comparator n will run during Standby sleep mode.

Value	Description
0	COMPn will not run in Standby sleep mode

Value	Description
1	COMPn will run in Standby sleep mode

Bits 5:4 – INTSEL[1:0] Interrupt Selection
This bit field defines the condition for comparator n to generate an interrupt or event.

Value	Name	Description
0x0	TOGGLE	Interrupt on comparator n output toggle
0x1	RISING	Interrupt on comparator n output rising edge
0x2	FALLING	Interrupt on comparator n output falling edge
Other	—	Reserved

Bit 3 – HYSTEN Hysteresis Enable
This bit controls the hysteresis function of comparator n.
For details on hysteresis values, refer to the *Electrical Characteristics* section.

Value	Description
0	Hysteresis is disabled
1	Hysteresis is enabled

Bit 2 – OUT Output Enable
This bit controls whether the comparator n output is enabled or not.

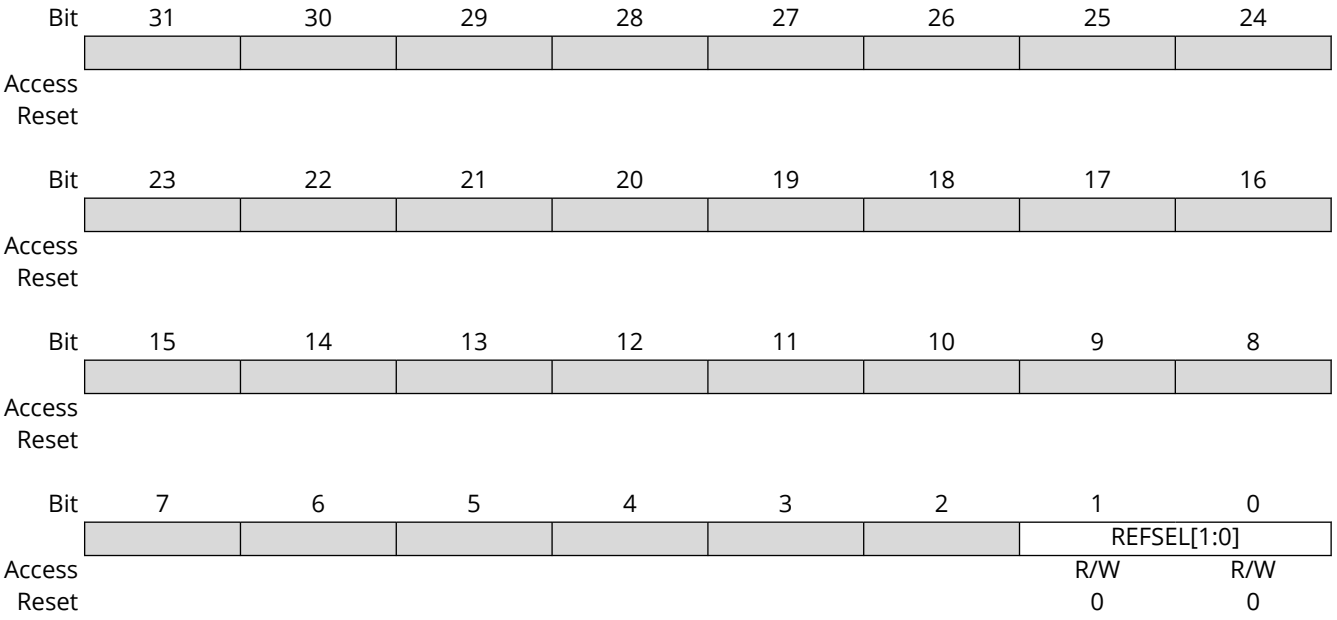
Value	Description
0	Output to COMPn-OUT pin is disabled
1	Output to COMPn-OUT pin is enabled

Bit 1 – ENABLE Comparator Enable
This bit controls whether comparator n is enabled.

Value	Description
0	COMPn is or being disabled
1	COMPn is or being enabled

36.6.11. Comparator n Comparator Control B

Name: COMP[n].COMPCTRLB
Offset: 0x44 + n*0x10 [n=0..1]
Reset: 0x00000000
Property: Local Write-Protection



Bits 1:0 – REFSEL[1:0] Reference Selection
This bit field controls the reference voltage for the reference scaler.
The internal reference can only be used if the reference is below V_{DD} - 0.5V.

Value	Name	Description
0x0	VDD	V_{DD}
0x1	INTVREF	Internal reference voltage, as defined by the Internal Reference Selection bit field in the Control B register (CTRLB.INTREF)
0x2	EXTVREF	External reference pin
Other	—	Reserved

36.6.12. Comparator n Comparator Scaler

Name: COMP[n].COMPSCALER
Offset: 0x48 + n*0x10 [n=0..1]
Reset: 0x00000000
Property: Local Write-Protection

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	VALUE[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 – VALUE[7:0] Scaled Reference Voltage
This bit field controls the scaling factor for the reference voltage scaler.

36.6.13. Write Protection Control

Name: WPCTRL
Offset: 0x80
Reset: 0x00000000
Property: –

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

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

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

Bits 31:8 – WPKEY[23:0] Write Protection Key
When writing to the WPCTRL register, this bit field must contain the specific KEY value for the write operation to be successful.

Value	Name	Description
0x414320	KEY	Allow writes to the WPCTRL register
Other	—	Reserved. The write operation to the WPCTRL register will be ignored, and a bus error will be generated.

Bit 1 – WPLCK Write Protection Lock

Value	Description
0	The WPCTRL register is not write-protected
1	The WPCTRL register is write-protected. Only a Reset can clear this bit.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	AC register write protection is disabled
1	Write protection is enabled on AC registers with the Local Write-Protection property. Non-debugger writes to AC registers with Local Write-Protection are ignored and will generate a bus error.

37. Electrical Characteristics @85°C

37.1. Disclaimer

Unless otherwise specified, all typical values are measured at T_A = 25°C and V_{DD} = 3.0V. All minimum and maximum values are valid across operating temperature and voltage ranges, unless otherwise specified.

The given typical values must be considered for design guidance only, and part-to-part variation around the values is expected.

37.2. Absolute Maximum Ratings

Stresses beyond those listed in this section may cause permanent damage to the device. These are stress ratings only. Functional operation of the device at these any 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 37-1. Absolute Maximum Ratings

Parameter	Rating	Units
Ambient temperature under bias	-40 to +85	°C
Maximum junction temperature	+150	°C
Storage temperature	-60 to +150	°C
Voltage on V_{DD} with respect to GND	-0.3 to +6.5	V
Voltage on V_{DDIO2} with respect to GND	-0.3 to +6.5	V
Voltage on AVDD with respect to GND	-0.3 to +6.5	V
Voltage on VREFA with respect to AVDD	-0.3 to (AVDD + 0.3)	V
Maximum current out of GND pins	140	mA
Maximum current into V_{DD} pins(1)	250	mA
Maximum current into V_{DDIO2} pins(1)	250	mA
Maximum DC current sunk by a standard I/O pin	50	mA
Maximum total power dissipation	800	mW
ESD Qualification		
Human Body Model (HBM) per JESD22-A114	2000	V
Charged Device Model (CDM) (ANSI/ESD STM 5.3.1) (All pins/Corner pins)	500 / 750	V

Note:

1. The maximum allowable current is a function of the device's maximum power dissipation. Power dissipation is calculated as follows: $P_{DIS} = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma\{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma(V_{OL} \times I_{OL})$.

37.3. Operating Frequencies and Thermal Limitations

Table 37-2. Operating Frequency vs. Voltage

Param. No.	V_{DDIO}, V_{DDIO2}, AVDD Range	Temp. Range	Max MCU Frequency	Comments
DC_5	1.8V to 5.5V	-40°C to +85°C	24 Mhz	Industrial

Table 37-3. Thermal Operating Conditions

Rating	Symbol	Min.	Typ.	Max.	Unit
Industrial Temperature Devices					
Operating ambient temperature range	T_A	-40	—	85	°C
Power Dissipation					

Table 37-3. Thermal Operating Conditions (continued)

Rating	Symbol	Min.	Typ.	Max.	Unit
Internal chip power dissipation: $P_{INT} = (V_{DDIO2} \times (I_{DD} - \Sigma I_{OHVDDIO2})) + (AVDD \times (I_{DDANA} - \Sigma I_{OHAVDD}) + (V_{DD} \times (I_{DD} - \Sigma I_{OHVDD}))$ I/O pin power dissipation: $P_{I/O} = \sum ((V_{DDIO2} - V_{OHVDDIO2}) \times I_{OHVDDIO2}) + \sum (V_{OL} \times I_{OLVDDIO2}) + \sum ((AVDD - V_{OHAVDD}) \times I_{OHAVDD}) + \sum (V_{OL} \times I_{OLAVDD}) + \sum ((V_{DD} - V_{OHVDD}) \times I_{OHVDD}) + \sum (V_{OL} \times I_{OLVDD})$	P_D		$P_{INT} + P_{I/O}$		W
Maximum allowed power dissipation	PDMAX		$(T_J - T_A)/\theta JA$		W

Table 37-4. Thermal Packaging Characteristics[1]

Characteristics	Symbol	Typ.	Max.	Unit
Thermal resistance for the 28-pin VQFN Wettable Flanks (4 x 4 x 1.0 mm) package	θJA	22.59	—	°C/W
Thermal resistance for the 28-pin SSOP (5.30 mm) package	θJA	57.3	—	°C/W
Thermal resistance for the 28-pin SPDIP (0.300) package	θJA	60.0	—	°C/W
Thermal resistance for the 32-pin TQFP (7 x 7 x 1.0 mm) package	θJA	56.63	—	°C/W
Thermal resistance for the 32-pin VQFN Wettable Flanks (5 x 5 x 0.9 mm) package	θJA	27.56	—	°C/W
Thermal resistance for the 48-pin TQFP (7 x 7 x 1.0 mm) package	θJA	62.8	—	°C/W
Thermal resistance for the 48-pin VQFN Wettable Flanks (6 x 6 x 0.9 mm) package	θJA	24.8	—	°C/W
Thermal resistance for the 64-pin TQFP (10 x 10 x 1.0 mm) package	θJA	49.83	—	°C/W
Thermal resistance for the 64-pin VQFN Wettable Flanks (9 x 9 x 1.0 mm) package	θJA	19.67	—	°C/W

Note:
1. Junction to ambient thermal resistance Theta-JA (θJA), values are obtained through package simulations.

37.4. Power Supply

Table 37-5. Power Supply DC Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated)
Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial

Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
REG_5	V_{DD_CIN}[3]	V_{DD} Input bypass parallel capacitor pair	—	0.1	—	µF	Bulk ceramic or solid tantalum capacitor with ESR < 0.5Ω
			—	10	—	nF	Ceramic X7R capacitor with ESR < 0.5Ω on all V_{DDIO2} pins
	V_{DDIO2_CIN}[3]	V_{DDIO2} Input bypass parallel capacitor pair	—	0.1	—	µF	Bulk ceramic or solid tantalum capacitor with ESR < 0.5Ω
			—	10	—	nF	Ceramic X7R capacitor with ESR < 0.5Ω on all V_{DDIO2} pins
REG_9	V_{REFA_CIN}[3]	External V_{REFA} Input bypass parallel capacitor pair	—	0.1	—	µF	Bulk ceramic or solid tantalum capacitor with ESR < 0.5Ω
			—	10	—	nF	Ceramic X7R capacitor with ESR < 0.5Ω
REG_17	$AVDD_{CIN}$[3]	AVDD Input bypass parallel capacitor pair	—	0.1	—	µF	Bulk Ceramic or solid tantalum capacitor with ESR < 0.5Ω
			—	10	—	nF	Ceramic X7R capacitor with ESR < 0.5Ω
REG_23	$AVDD_{LEXT}$[1]	AVDD series ferrite bead DC resistance (DCR)	—	—	0.1	Ω	≥ 600Ω @ 100 MHz
REG_25		Ferrite bead current rating	500	—	—	mA	—
REG_37	V_{DD}[2]	V_{DD} input voltage range	1.8	—	5.5	V	—
REG_37A	V_{DDIO2}	V_{DDIO2} input voltage range	1.6	—	5.5	V	—
REG_39	AVDD[2]	AVDD input voltage range	1.8	—	5.5	V	—

Table 37-5. Power Supply DC Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
REG_40	V_{DDCORE}[4]	Core voltage	—	2.0	—	V	With AVDD = V_{DD} > 2.4V
REG_43A	S_{VDD_R}	V_{DD} rise ramp rate required to ensure the internal Power-On Reset signal	—	—	0.05	V/ms	With BOD disabled, failure to meet this specification may lead to start-up issues or unexpected behaviors
		V_{DD} change rate required to prevent the internal Power-On Reset signal during operation	—	—	1.2	V/µs	1.8V ≤ V_{DD} ≤ 5.5V
REG_43B	S_{VDDIO2_R}	V_{DDIO2} rise ramp rate required to ensure the internal Power-On Reset signal	—	—	0.05	V/ms	—
REG_45	V_{POR}	Power-On Reset	—	1.6	—	V	V_{DD} power- up or power-down (see Parameter REG_43, V_{DD} ramp rate)
REG_47	V_{DD} / V_{DDIO2} / AVDD BOR	V_{DD} / V_{DDIO2} / AVDD Brown-Out Reset thresholds	1.8	1.9	2.1	V	BODLEVEL0
			2.3	2.45	2.6	V	BODLEVEL1
			2.55	2.7	2.85	V	BODLEVEL2
			2.7	2.85	3.0	V	BODLEVEL3

Notes:

1. Ferrite Bead I_{SAT}(min) ≥ (I_{DDANA}(max) x 1.15).
2. V_{DD} and AVDD must be at the same voltage level.
3. All bypass capacitors should be located immediately adjacent to pin(s) and on the same side of the PCB as the MCU.
4. For V_{DD} < 2.4 V_{DDCORE} can be lower than V_{DD}.

37.5. Power Consumption

37.5.1. MCU Active Power

Table 37-6. MCU Active Current Consumption Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 3.0V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Clock/Freq**	**Typ.**	**Max.**	**Units**	**Conditions**
APWR_1	I_{DD_ACTIVE}[1,2]	MCU I_{DD} in active mode	OSCHF	5.2	—	mA	FRQSEL = 24 MHz
APWR_2			EXTCLK_24MHz	5.3	—	mA	
APWR_3			OSC_32K	8.0	—	µA	
APWR_4[3]			XOSC_32K	10	—	µA	XOSC32K = 32 kHz crystal in LP mode

Notes:

1. Conditions:
   - No peripheral modules are operating (i.e., all peripherals are inactive)
   - All APB clocks are masked except for MCLK and NVMCTRL
   - MCLK.AHBMASK = `0x000003FF`
   - MCU is running on Flash
   - I/Os are in inactive input mode with the input trigger disabled
   - All clock generation sources are disabled unless otherwise specified
2. CPU Running While(1) Loop.
3. Characterized but not tested.

37.5.2. MCU Idle Power

Table 37-7. MCU Idle Current Consumption Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 3.0V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Clock/Freq	Typ.	Max.	Units	Conditions
IPWR_1	I_{DD_IDLE}(1)	MCU I_{DD} in idle mode	OSCHF	1.2	—	mA	FRQSEL = 24 MHz
IPWR_2			EXTCLK_24MHz	1.1	—	mA	
IPWR_3			OSC_32K	3.0	—	µA	
IPWR_4(2)			XOSC_32K	5.0	—	µA	XOSC32K = 32 kHz crystal in LP mode

Notes:

1. Conditions:
   - No peripheral modules are operating (i.e., all peripherals are inactive)
   - All APB clocks are masked except for MCLK and NVMCTRL
   - MCLK.AHBMASK = `0x000003FF`
   - I/Os are in inactive input mode with input trigger disabled
   - All clock generation sources are disabled unless otherwise specified
2. Characterized but not tested.

37.5.3. MCU Standby Power

Table 37-8. MCU Standby Current Consumption DC Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 3.0V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
SPWR_1	$I_{DD\_STANDBY}$(1)	MCU I_{DD} in Standby mode	—	2.0	—	µA	SUPC.VREG.RUNSTDBY = `0`
SPWR_3			—	190	—	µA	SUPC.VREG.RUNSTDBY = `1`

Note:

1. Conditions:
   - No peripheral modules are operating (i.e., all peripherals are inactive)
   - All APB clocks are masked except for MCLK and NVMCTRL
   - MCLK.AHBMASK = `0x000003FF`
   - I/Os are in inactive input mode with input trigger disabled
   - All clock generation sources are disabled unless otherwise specified

37.5.4. Peripheral Active Current

Table 37-9. Peripheral Active Current DC Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: T_A = 25°C for Industrial**						
Param. No.	Symbol	Characteristics	Typ.	Max.	Units	Conditions
PAI_10	I_{RTC}	RTC current	500	—	nA	
PAI_12	I_{WDT}	WDT current	210	—	nA	
PAI_14	I_{BOD32}	BOD current	625	—	nA	Sampling mode 32 Hz
PAI_16	$I_{XOSC32K}$	XOSC32 current	—	—	nA	
PAI_18	I_{OSC32K}	OSC32 Current	200	—	nA	
PAI_20	I_{BOD128}	BOD Current	600	—	nA	Sampling mode 128 Hz
PAI_30	I_{AC}	AC current	25	—	µA	With external inputs

Table 37-9. Peripheral Active Current DC Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: T_A = 25°C for Industrial**						
Param. No.	**Symbol**	**Characteristics**	**Typ.**	**Max.**	**Units**	**Conditions**
PAI_34	I_{BODA}	BOD current	6.0	—	µA	Continuous mode
PAI_40	$I_{ADCINTVREF}$	INTVREF current ADC	190	—	µA	
PAI_42	$I_{ACINTVREF}$	INTVREF current AC	80	—	µA	
PAI_50	I_{ADC}	ADC current	1.7	—	mA	Free-running mode, 800 ksps

37.6. Wake-up Timing

Table 37-10. Wake-Up Timings from Low-Power Modes Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
WKUP_3	$WKUP_{STDBY}$	Wake from Standby mode	—	120	—	µs	SUPC.VREG.RUNSTDBY = 0
WKUP_5			—	24	—	µs	SUPC.VREG.RUNSTDBY = 1

37.7. I/O Pin Electrical Specifications

Table 37-11. I/O Pin Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
DI_1A	V_{IL}(1)	I/O Pins	—	—	0.2 x V_{DD}	V	
DI_1B		With I^2C Levels	—	—	0.3 x V_{DD}		
DI_1C		$\overline{RESET}$ pin	—	—	0.2 x V_{DD}		
DI_3A	V_{IH}(1)	I/O pins	0.8 x V_{DD}	—	—	V	
DI_3B		With I^2C Levels	0.7 x V_{DD}	—	—		
DI_3C		$\overline{RESET}$ pin	0.8 x V_{DD}	—	—		
DI_5	V_{OL}	Output voltage low V_{DD} pins	—	—	0.6	V	V_{DD} = 3.0V @ I_{OL} = 10 mA
DI_7		Output voltage low V_{DDIO2} pins	—	—	0.6		V_{DDIO2} = 3.0V @ I_{OL} = 10 mA
DI_9	V_{OH}	Output voltage high V_{DD} pins	V_{DD} - 0.7	—	—		V_{DD} = 3.0V @ I_{OH} = 6 mA
DI_11		Output voltage high V_{DDIO2} pins	V_{DD} - 0.7	—	—		V_{DDIO2} = 3.0V @ I_{OH} = 6 mA
DI_13A	I_{IL}	Input pin leakage current on V_{DD} pins	—	5	125	nA	GND ≤ V_{PIN} ≤ V_{DD}(max) (V_{PIN} = Voltage present on pin)
DI_13B		Input pin leakage current on V_{DDIO2} pins	—	5	125	nA	GND ≤ V_{PIN} ≤ V_{DDIO2}(max) (V_{PIN} = Voltage present on pin)
DI_17	R_{PUP}(1)	Internal pull-up (DIR = 0, OUT = PULLEN = 1)	15	20	33	kΩ	V_{DD} = V_{DDIO2} = 3V
DI_25	t_{RISE}(1)	I/O pin rise time	—	22	—	ns	Slew rate limit disabled V_{DD} = V_{DDIO2} = 3V, C_{LOAD} = 30 pf(max)
			—	45	—	ns	Slew rate limit enabled V_{DD} = V_{DDIO2} = 3V, C_{LOAD} = 30 pf(max)
DI_27	t_{FALL}(1)	I/O pin fall time	—	16	—	ns	Slew rate limit disabled V_{DD} = V_{DDIO2} = 3V, C_{LOAD} = 30 pf(max)
			—	30	—	ns	Slew rate limit enabled V_{DD} = V_{DDIO2} = 3V, C_{LOAD} = 30 pf(max)

Table 37-11. I/O Pin Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions

Note:

1. The figures in *I/O Pin Graphics* section are valid for all I/O ports regardless of if they are connected to the V_{DDIO} or V_{DDIO2} power domain.

37.8. Internal Voltage Reference Specifications

Table 37-12. Internal Voltage Reference Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
VR_1	I_{REF}	Internal voltage reference accuracy	-4	—	4	%	ADC.CTRLC.REFSEL = 1V024
VR_3			-4	—	4	%	ADC.CTRLC.REFSEL = 2V048
VR_5			-4	—	4	%	ADC.CTRLC.REFSEL = 4V096
VR_7			-4	—	4	%	ADC.CTRLC.REFSEL = 2V500
VR_9	I_{REFVDD}	V_{REF}, V_{DD} range	2.7	—	5.5	V	ADC.CTRLC.REFSEL = 1V024
VR_11			2.7	—	5.5	V	ADC.CTRLC.REFSEL = 2V048
VR_13			4.55	—	5.5	V	ADC.CTRLC.REFSEL = 4V096
VR_15			2.95	—	5.5	V	ADC.CTRLC.REFSEL = 2V500
VR_20	I_{REFEX}	$EXTV_{REF0}$ pin voltage	1.8	—	V_{DD}	V	V_{DD} < 2.7V
VR_22			1.024	—	V_{DD}	V	V_{DD} ≥ 2.7V
VR_25	I_{REFD}	Delay for changing voltage reference	—	2.0	—	µA	—
VR_30	I_{REFST}	Internal V_{REF} start-up time	—	10	—	µs	V_{DD} = 3.0V

37.9. Clock Controller (CLKCTRL) Characteristics

37.9.1. Maximum Clock Frequencies Electrical Specifications

Table 37-13. Maximum Clock Frequencies Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**				
Param. No.	Symbol	Characteristics	Max	Units
FCLK_1	f_{CY}	MCU clock frequency	24	MHz
FCLK_3	f_{AHB}	AHB clock frequency	24	MHz
FCLK_5	f_{APBn}	APBA, APBB, APBC, APBD clock frequency	24	MHz
FCLK_13	f_{GCLK_EIC}	EIC input clock frequency	24	MHz
FCLK_19	$f_{GCLK\_EVSYS\_CHANNELn}$	EVSYS channel n input clock frequency	24	MHz
FCLK_21	$f_{GCLK\_SERCOMn\_SLOW}$	Common SERCOM slow input clock frequency	24	MHz
FCLK_23	$f_{GCLK\_SERCOMn\_CORE}$	SERCOMn input clock frequency	24	MHz
FCLK_35	f_{GCLK_TCCn}	TCCn input clock frequency	24	MHz
FCLK_37	f_{GCLK_TCn}	TC0, TC1, TC2, TC3 input clock frequency	24	MHz
FCLK_43	f_{GCLK_CCL}	CCL input clock frequency	24	MHz
FCLK_45	$f_{GCLK\_GCLKINn}$	External GCLKn input clock frequency	24	MHz
FCLK_46	$f_{GCLK\_GCLKOUTn}$	External GCLKn output clock frequency	$1/(Max(t_{RISE}, t_{FALL}) \times \pi$	MHz
FCLK_49	f_{GCLK_AC}	Analog comparator peripheral module clock frequency	24	MHz
FCLK_51	f_{GCLK_ADCn}	ADCn input clock frequency	24	MHz

Table 37-13. Maximum Clock Frequencies Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial				
Param. No.	**Symbol**	**Characteristics**	**Max**	**Units**
FCLK_55	f_{GCLK_PTC}	PTC input clock frequency	24	MHz
FCLK_63	f_{GCLK_PTC}	PTC (Peripheral Touch Controller), input clock frequency	24	MHz
FCLK_67	f_{GCLK_SPI}	SERCOM SPI internal GCLK frequency	24	MHz
FCLK_69	f_{GCLK_I2C}	SERCOM I^2C internal GCLK frequency	24	MHz
FCLK_71	f_{GCLK_USART}	SERCOM USART internal GCLK freq (Asynchronous mode)	24	MHz
		SERCOM USART internal GCLK freq (Synchronous mode)	24	MHz

37.9.2. External 32 kHz Crystal Oscillator (XOSC32K) Electrical Specifications

Table 37-14. 32.768 kHz Crystal Oscillator (XOSC32K) Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
XOSC32_1	f_{OSC_XOSC32}	XOSC32 crystal oscillator frequency	—	32.768	—	kHz	—
XOSC32_3	C_{XIN32}	XOSC32 XIN32 parasitic pin capacitance	—	3.5	—	pF	—
XOSC32_5	C_{XOUT32}	XOSC32 XOUT32 parasitic pin capacitance	—	3.6	—	pF	—
XOSC32_11	C_{LOAD_X32} (2)	32.768 kHz crystal load capacitance	—	—	9	pF	XOSC32KCTRL.LPMODE = 1 XOSC32KCTRL.XTALEN = 1 XOSC32KCTRL.ENABLE = 1
XOSC32_12			—	—	12.5	pF	XOSC32KCTRL.LPMODE = 0 XOSC32KCTRL.XTALEN = 1 XOSC32KCTRL.ENABLE = 1
XOSC32_13	ESR_{X32}	32.768 kHz crystal ESR	—	—	35	kΩ	XOSC32KCTRL.LPMODE = 1 XOSC32KCTRL.XTALEN = 1 XOSC32KCTRL.ENABLE = 1
XOSC32_14			—	—	70	kΩ	XOSC32KCTRL.LPMODE = 0 XOSC32KCTRL.XTALEN = 1 XOSC32KCTRL.ENABLE = 1
XOSC32_15	T_{OSC32}	$T_{OSC32} = 1/f_{OSC\_XOSC32}$	—	30.5176	—	µs	See parameter XOSC32_1 for F_{OSC_XOSC32} value
XOSC32_17	$XOSC32_{ST}$ (1)	XOSC32 Crystal Start-up Time	—	51500	(3)	TOSC	XOSC32KCTRL.LPMODE = 1 Crystal stabilization time only not oscillator ready
XOSC32_18			—	5800	(3)	TOSC	XOSC32KCTRL.LPMODE = 0 Crystal stabilization time only not oscillator ready
XOSC32_19	f_{OSC_XCLK32}	Ext clock oscillator input freq (XIN32 pin)	—	32.768	—	kHz	XOSC32KCTRL.XTALEN = 0
XOSC32_21	$XCLK32_{DC}$	Ext clock oscillator duty cycle	—	50	—	%	XOSC32KCTRL.XTALEN = 0
XOSC32_23	$XCLK32_{FST}$	XIN32 clock fail safe time-out period	—	4 x T_{OSC32} (4)	—	µs	—

Table 37-14. 32.768 kHz Crystal Oscillator (XOSC32K) Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions

Notes:

1. This is for guidance only. A major component of crystal start-up time is based on the second party crystal MFG parasitics that are outside the scope of this specification. If this is a major concern the customer would need to characterize this based on their design choices.
2. CRYSTAL LOAD CAPACITOR CALCULATION GIVEN:
   - Standard PCB trace capacitance = 1.5 pF per 12.5 mm (0.5 inches) (i.e. PCB STD TRACE W = 0.175 mm, H = 36 µm, T = 113 µm)
   - Xtal PCB capacitance typical therefore ~= 2.5 pF for a tight PCB xtal layout
   - For C_{XIN} and C_{XOUT} within 4pF of each other, assume C_{XTAL_EFF} = ((C_{XIN} + C_{XOUT})/2)
     Note: Averaging C_{XIN} and C_{XOUT} will affect final calculated C_{LOAD} value by less than the tolerance of the capacitor selection.

   EQUATION 1:

   MFG C_{LOAD} Spec = {([C_{XIN} + C1] x [C_{XOUT} + C2])/[C_{XIN} + C1 + C2 + C_{XOUT}] } + estimated oscillator PCB stray capacitance
   - Assuming C1 = C2 and C_{XIN} ~= C_{XOUT}, the formula can be further simplified and restated to solve for C1 and C2 by:

   EQUATION 2: (Simplified Equation 1)

   C1 = C2 = ((2 x MFG C_{LOAD} spec) - C_{XTAL_EFF} - (2 x PCB capacitance))

   EXAMPLE ONLY:
   - XTAL Mfg C_{LOAD} Data Sheet Spec = 12 pF
   - PCB XTAL trace Capacitance = 2.5 pF
   - C_{XIN} pin = 6.5 pF, C_{XOUT} pin = 4.5 pF. Therefore C_{XTAL_EFF} = ((C_{XIN} + C_{XOUT})/2)

   C_{XTAL_EFF} = ((6.5 + 4.5)/2) = 5.5 pF

   C1 = C2 = ((2 x MFG C_{LOAD} spec) - C_{XTAL_EFF} - (2 x PCB capacitance))

   C1 = C2 = (24 - 5.5 - (2 x 2.5))

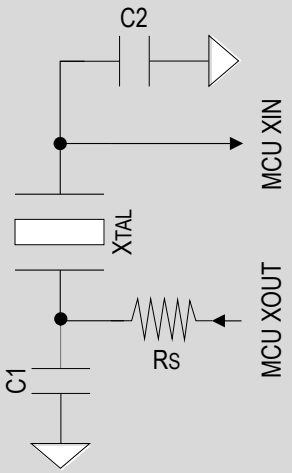
   C1 = C2 = (24 - 5.5 - 5)

   C1 = C2 = 13.5 pF (Always rounded down)

   C1 = C2 = 13 pF (i.e., for hypothetical example crystal external load capacitors)

   User C1 = C2 = 13 pF ≤ C_{LOAD}_X32 (max) spec

Figure 37-1. XTAL



3. User Selectable in XOSC32KCTRL.CSUT[1:0].
4. Period of four safe clocks, monitored the Clock Failure Detection watching for XOSC32K clock activity.

37.9.3. Internal 32.768 kHz RC Oscillator (OSC32K)

Table 37-15. Internal 32.768 kHz RC Oscillator (OSC32K) Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
OSC32K_1	f_{OSC_OSC32K}	Output frequency	—	32.768	—	kHz	T_A = 25 °C, V_{DD} = 5.0V
OSC32K_3	$OSC32K_{ACC}$	Accuracy	-10	—	10	%	
OSC32K_9	$OSC32K_{Duty}$	OSC32 duty cycle	—	50	—	%	

37.9.4. Internal RC Oscillator (OSCHF)

Table 37-16. Internal RC Oscillator (OSCHF) Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
OSCHF_1	f_{OSCHF}	Precision calibrated OSCHF oscillator frequency	-5	—	5	%	f_{OSCHF} ≥ 4 MHz -40°C ≤ TA ≤ 0°C
			-2	—	2	%	f_{OSCHF} ≥ 4 MHz 0°C ≤ TA ≤ +60°C
			-3	—	3	%	f_{OSCHF} ≥ 4 MHz +60°C ≤ TA ≤ +85°C
OSCHF_2(1)	f_{OSCHF}	OSCHF oscillator frequency	—	1	—	MHz	OSCCTRL.OSCHFCTRL.FRQSEL = 0x00
			—	2	—	MHz	OSCCTRL.OSCHFCTRL.FRQSEL = 0x01
			—	3	—	MHz	OSCCTRL.OSCHFCTRL.FRQSEL = 0x02
OSCHF_5	$OSCHF_{Duty}$	OSCHF oscillator duty cycle	40	—	60	%	
OSCHF_6	$OSCHF_{Duty}$	OSCHF oscillator tune step size	—	0.4	—	%	
OSCHF_7	$OSCHF_{ST}$	OSCHF oscillator start-up time	—	24	30	µs	SUPC.VREG.RUNSTDBY = 0
			—	120	180	µs	SUPC.VREG.RUNSTDBY = 1

Note:

1. These OSCHF settings are not calibrated.

37.10. Analog-to-Digital Converter (ADC) Electrical Specifications

Table 37-17. ADC Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
Device Supply							
ADC_1	AVDD	ADC module supply	AVDD(min)	—	AVDD(max)	V	
Reference Inputs							
ADC_3	V_{REF}(1)	ADC reference voltage	The greater of ≥ AVDD(min) or 2.4V	—	AVDD	V	External reference
					AVDD	V	Internal reference
ADC_5	I_{VREF}	External V_{REF} input load current	—	—	—	µA	@ f_{CNV}(max) w/ V_{REF} input buffer disabled or bypassed
Analog Input Range							

Table 37-17. ADC Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
ADC_7	AFS	Full-scale analog input signal range	AGND	—	V_{REF}	V	Single-ended mode
			$-V_{REF}$	—	$+V_{REF}$	V	Differential mode
ADC_9	V_{CMIN}	Input common mode voltage	AGND	—	AVDD	V	
ADC_11	$t_{SETTLING}$	ADC stabilization time	—	10	—	µs	

Note:

1. ADC functional device operation with either internal or external V_{REF} < 2.4V is functional, but not characterized. ADC will function, but with degraded accuracy of approximately ~((0.006 * 2^n)/V_{REF}) LSB's over full scale range, where "n" is the number of bits. ADC accuracy is limited by internal V_{REF} accuracy and drift, MCU-generated noise and the user's application noise/accuracy on AVDD, AGND.

Table 37-18. ADC Single-Ended Mode AC Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 3.0V (Unless otherwise stated) Operating temperature: T_A = 25°C							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
Single Ended Mode ADC Accuracy(4)							
SADC_11	Res	Resolution	8	—	12	bits	Selectable 8-, 10-, 12-bit resolution ranges
SADC_13	E_{NOB} (1,2)	Effective number of bits	—	9.3	—	bits	12-bit, 800 ksps, Internal V_{REF}
SADC_15			—	9.3	—	bits	10-bit, 930 ksps, Internal V_{REF}
SADC_17			—	7.8	—	bits	8-bit, 1000 ksps, Internal V_{REF}
SADC_19	INL	Integral nonlinearity	—	4	—	LSb	12-bit, 800 ksps, Internal V_{REF}
SADC_25	DNL	Differential nonlinearity	—	1	—	LSb	12-bit, 800 ksps, Internal V_{REF}
SADC_31	G_{ERR}	Gain error	—	3.5	—	LSb	12-bit, 800 ksps, Internal V_{REF}
SADC_37	E_{OFF}	Offset error	—	6	—	LSb	12-bit, 800 ksps, Internal V_{REF}
Single Ended Mode ADC Dynamic Performance(1,2,4)							
SADC_49	SINAD	Signal-to-noise and distortion	—	58	—	dB	V_{REF} = AVDD = V_{DD} = 3.0V @ 12-bit maximum sampling rate
SADC_51	SNR	Signal-to-noise ratio	—	65	—		
SADC_53	SFDR	Spurious-free dynamic range	—	50	—		
SADC_55	THD(3)	Total harmonic distortion	—	59	—		

1. Characterized with an analog input sine wave = (F_{TP}(max)/100). For example, F_{TP}(max) = 1 Msps/100 = 10 kHz sine wave.
2. Sine wave peak amplitude is 96% of the ADC full-scale input amplitude with 12-bit resolution.
3. The value taken over seven harmonics.
4. The ADC is configured in 12-bit mode. All registers are at the reset default value unless otherwise stated.

Table 37-19. ADC Differential Mode AC Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 3.0V (Unless otherwise stated) Operating temperature: T_A = 25°C							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
Differential Mode ADC Accuracy(4)							
DADC_11	Res	Resolution	8	—	12	bits	Selectable 8-, 10-, 12-bit resolution ranges
DADC_13	E_{NOB} (1,2)	Effective number of bits	—	10.1	—	bits	12-bit, 800 ksps, Internal V_{REF}
DADC_15			—	9.5	—	bits	10-bit, 930 ksps, Internal V_{REF}
DADC_17			—	7.9	—	bits	8-bit, 1000 ksps, Internal V_{REF}
DADC_19	INL	Integral nonlinearity	—	2.5	—	LSb	12-bit, 800 ksps, Internal V_{REF}

Table 37-19. ADC Differential Mode AC Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 3.0V (Unless otherwise stated) Operating temperature: T_A = 25°C							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
DADC_25	DNL	Differential nonlinearity	—	1.0	—	LSb	12-bit, 800 ksps, Internal V_{REF}
DADC_31	G_{ERR}	Gain error	—	1.2	—	LSb	12-bit, 800 ksps, Internal V_{REF}
DADC_37	E_{OFF}	Offset error	—	0.7	—	LSb	12-bit, 800 ksps, Internal V_{REF}
Differential Mode ADC Dynamic Performance(1,2,4)							
DADC_49	S_{INAD}	Signal to noise and distortion	—	62	—	dB	V_{REF} = AVDD = V_{DD} = 3.0V @ 12-bit max sampling rate
DADC_51	S_{NR}	Signal to noise ratio	—	68	—		
DADC_53	S_{FDR}	Spurious free dynamic range	—	63	—		
DADC_55	T_{HD}(3)	Total harmonic distortion	—	63	—		

Notes:

1. Characterized with an analog input sine wave = (F_{TP} (max)/100). For example, F_{TP}(max) = 1 Msps/100 = 10 kHz sine wave.
2. Sine wave peak amplitude is 96% of the ADC full-scale input amplitude with 12-bit resolution.
3. The value taken over seven harmonics.
4. The ADC is configured in 12-bit mode. All registers are at the reset default value unless otherwise stated.

Table 37-20. ADC Conversion and Sample Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
ADC Clock Requirements							
ADC_57	T_{AD}	ADC clock period	83	—		ns	
ADC_58	f_{GCLK_ADCn}	ADCn module GCLK maximum input frequency	—	—	FCLK_51	MHz	
ADC Single-Ended Throughput Rates							
ADC_59	F_{TP}(1) (single-ended mode)	Throughput Rate (single-ended)	—	—	800	ksps	12-bit resolution
			—	—	930		10-bit resolution
			—	—	1000		8-bit resolution
ADC Differential Mode Throughput Rates							
ADC_61	F_{TP}(1) (differential mode)	Throughput rate (differential mode)	—	—	800	ksps	12-bit resolution
			—	—	930		10-bit resolution
			—	—	1000		8-bit resolution

37.11. Analog Comparator (AC) Electrical Specifications

Table 37-21. Analog Comparator Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
CMP_1	V_{IOFF_00}	Input offset voltage	—	± 5	—	mV	COMPCTRLA.HYSTEN = 0x0 Comparator ref voltage = AVDD/2 V_{DD} = 3.0V
	V_{IOFF_01}		—	±15	—	mV	COMPCTRLA.HYSTEN = 0x0 0.1V < V_{IN} < (V_{DD} - 0.1V) V_{DD} = 3.0V
CMP_4	V_{IN}	Input voltage range	GND	—	V_{DD}	mV	With respect to GND and AVDD
CMP_5	V_{HYST_00}	Input hysteresis voltage	—	10	—	mV	Comparator ref voltage = AVDD/2 COMPCTRLA.HYSTEN = 0x1

Table 37-21. Analog Comparator Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
CMP_15	t_{RESPSS}	Small signal response time	—	120	—	ns	Comparator ref voltage = AVDD/2 COMPCTRLA.HYSTEN = 0x0 V_{DD} = 3.0V Input overdrive = ±50 mV
CMP_19	C_{OUTVAL}	Comparator enabled to output valid	—	1.3	—	µs	Comparator module is configured before enabling it
CMP_25	f_{GCLK_AC}	Analog comparator peripheral module clock frequency	—	—	FCLK_49	MHz	—

37.12. Peripheral QTouch® Controller (PTC) Electrical Specifications

Table 37-22. Peripheral QTouch Controller Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
PTC_1	C_{LOAD_SC}(1)	Self capacitance mode (PTC channel Y0 - Y31) Y0	—	—	58	pF	Maximum sensor load capacitance(1)
		Yn-1	—	—	—	pF	
		Yn	—	—	—	pF	
PTC_3	C_{LOAD_MC}(1)	Mutual capacitance mode (PTC channel Y0 - Y31)	—	—	28	pF	
PTC_4A	f_{PTC}(2)	PTC frequency	—	—	FCLK_55	MHz	
PTC_4B	t_{wake}	Wake-up time	—	—	—	µs	
PTC_5	t_{wake}	Supply voltage	2.5	—	5.5	V	

Notes:

1. Maximum capacitive load that the PTC circuitry can compensate on each channel.
2. f_{PTC} = $GCLK_{PTC}$/PTC prescaler.

37.13. Serial Peripheral Interface (SERCOM SPI) Mode Electrical Specifications

Figure 37-2. SERCOMn SPI Host Module CPHA=0 Timing Diagrams

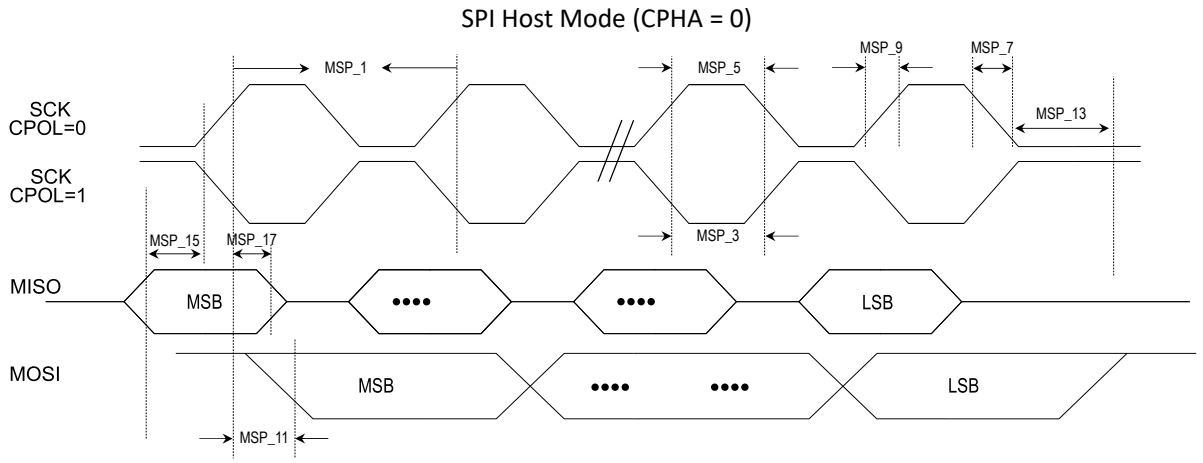


Figure 37-3. SERCOMn SPI Host Module CPHA=1 Timing Diagrams

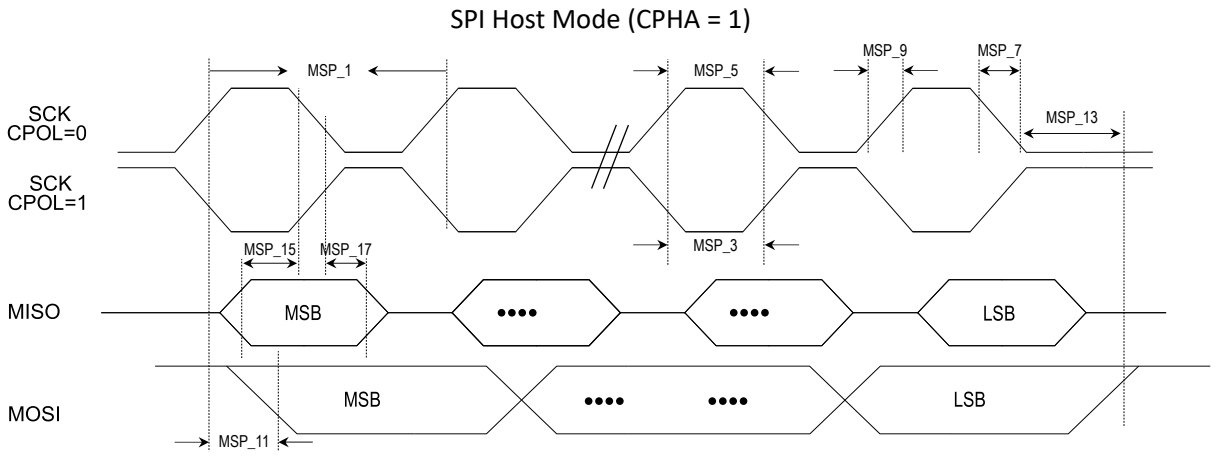


Table 37-23. SPIn Module Host Mode Electrical Specifications [1]

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics[1]**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
MSP_1	f_{SCK}	SCK Frequency	—	—		MHz	Transmitter mode, CTRLB.RXEN = 0, C_{LOAD} = 30 $pF_{(max)}$
			—	—			Full duplex transmit and receive mode, C_{LOAD} = 30 $pF_{(max)}$ Loop back mode with one SPI talking to another SPI on the same MCU
			—	—	1/(2 x (t_{MIS} + $NOTE2_{TV}$))[2]		Full duplex transmit and receive mode, C_{LOAD} = 30 $pf_{(max)}$. The maximum SPI speed of the MCU is partially dependent on the performance characteristics external SPI device. Faster speeds than the loop back mode above may therefore be possible using the formula.
MSP_3	t_{SCL}	SCK output low time	1/(2 x F_{SCK})	—	—	µs	—
MSP_5	t_{SCH}	SCK output high time	1/(2 x F_{SCK})	—	—	µs	—
MSP_7	t_{SCF}	SCK and MOSI output fall time	—	—	DI_27	ns	See parameter DI_27 in the I/O specifications
MSP_9	t_{SCR}	SCK and MOSI output rise time	—	—	DI_25	ns	See parameter DI_25 in the I/O specifications
MSP_11	t_{MOV}	MOSI data output valid after SCK	—	—	—	ns	$V_{DDIO(min)}$, C_{LOAD} = 30 $pF_{(max)}$
MSP_13	t_{MOH}	MOSI hold after SCK	—	1	—	ns	
MSP_15	t_{MIS}	MISO setup time of data input to SCK	—	10	—	ns	
MSP_17	t_{MIH}	MISO hold time of data input to SCK	—	10	—	ns	
MSP_19	SPI_{GCLK}	SERCOM SPI input clk freq, GCLK_SPI	—	—	FCLK_23	MHz	—

Notes:

1. Assumes V_{DDIO}(min) and a 30 pF external load on all SPIn pins unless, otherwise noted.
2. $NOTE2_{TV}$ is the client external device data output valid time from clock edge specification.

Figure 37-4. SERCOMn SPI Client Module (CPHA=0) Timing Diagram

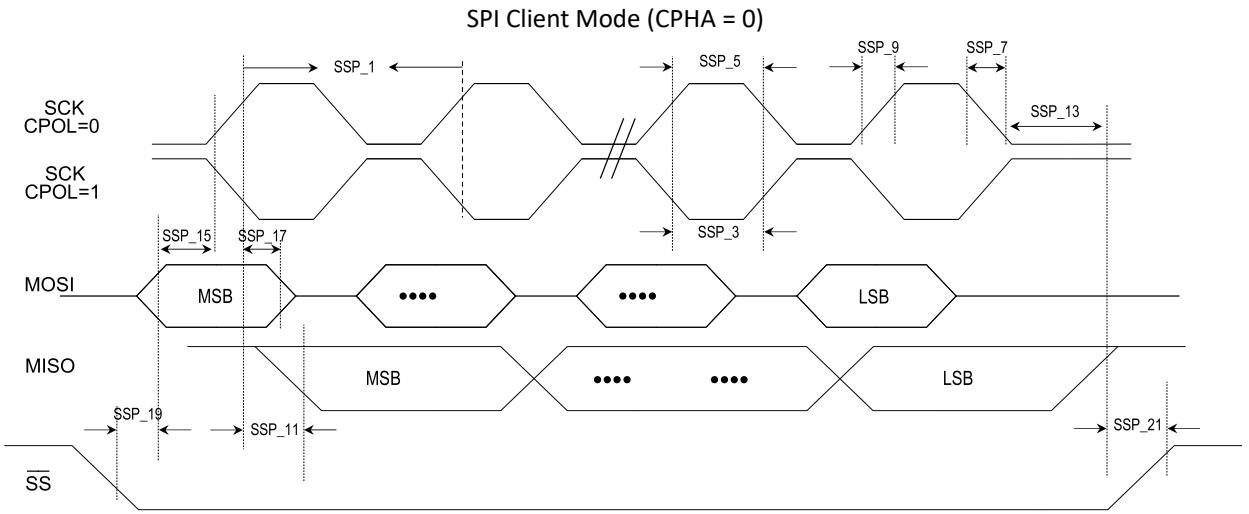


Figure 37-5. SERCOMn SPI Client Module (CPHA=1) Timing Diagram

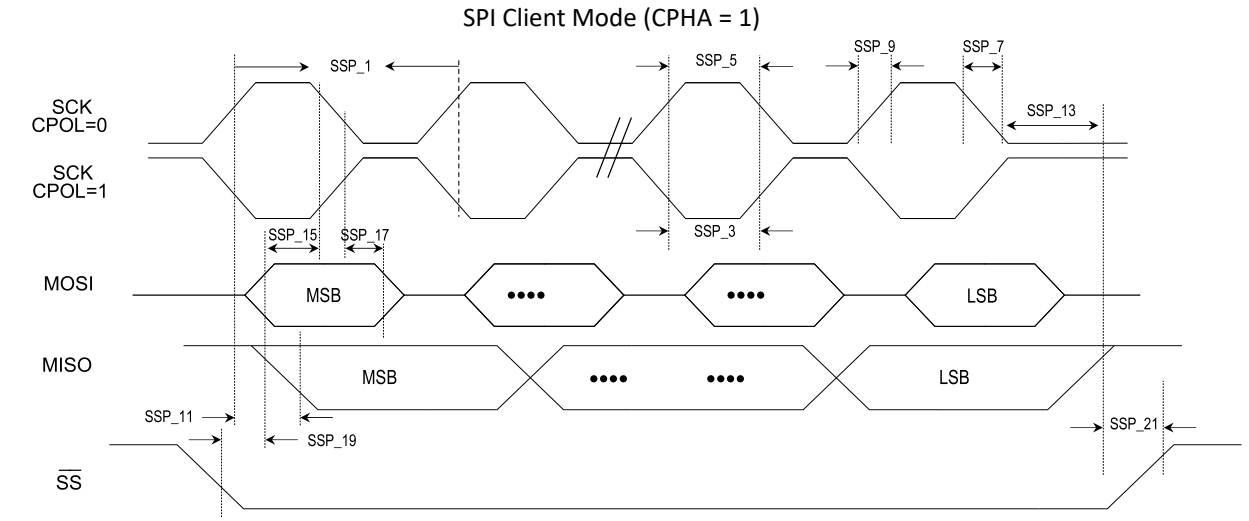


Table 37-24. SPIn Module Client Mode Electrical Specifications[1]

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics[1]**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
SSP_1	f_{SCK}	SCK Frequency	—	—	—	MHz	Receiver mode, C_{LOAD} = 30 $pF_{(max)}$
			—	—	—		Full duplex transmit and receive mode , C_{LOAD} = 30 $pF_{(max)}$ Loop back mode with one SPI communicating with another SPI on the same MCU
			—	—	1/(2 x (T_{SOV} + NOTE2_T_{MIS}))[2]		Full duplex transmit and receive mode, C_{LOAD} = 30 $pf_{(max)}$. The maximum SPI speed of the MCU is partially dependent on the performance characteristics of the external SPI device. Faster speeds than the loop back mode above may therefore be possible using the formula.
SSP_3	t_{SCL}	SCK output low time	1/(2 x f_{SCK})	—	—	µs	—
SSP_5	t_{SCH}	SCK output high time	1/(2 x f_{SCK})	—	—	µs	—
SSP_7	t_{SCF}	SCK and MOSI output fall time	—	—	DI_27	ns	See parameter DI_27 in the I/O specifications
SSP_9	t_{SCR}	SCK and MOSI output rise time	—	—	DI_25	ns	See parameter DI_25 in the I/O specifications
SSP_11	t_{SOV}	MISO data output valid after SCK	—	10	—	ns	V_{DDIO}(min), C_{LOAD} = 30 $pF_{(max)}$
SSP_13	t_{SOH}	MISO hold after SCK	—	13	—	ns	
SSP_15	t_{SIS}	MOSI setup time of data input to SCK	3	—	—	ns	
SSP_17	t_{SIH}	MOSI hold time of data input to SCK	t_{CLK_PER}	—	—	ns	
SSP_19	t_{SSS}	SS setup to SCK (PRELOADEN = 1)	21	—	—	ns	
		SS setup to SCK (PRELOADEN = 0)	21	—	—	ns	
SSP_21	t_{SSH}	SS hold after SCK client	20	—	—	ns	
SSP_23	SPI_{GCLK}	SERCOM SPI input clk freq, $GCLK_{SPI}$	—	—	FCLK_23	MHz	—

Notes:

1. Assumes V_{DDIO}(min) and a 30 pF external load on all SPIn pins unless, otherwise noted.
2. NOTE2_T_{MIS} is the setup time for the host external device.

37.14. SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter (SERCOM USART)

Table 37-25. SERCOM USART Electrical Specifications(1)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**								
Param. No.	**Symbol**	**Characteristics**		**Min.**	**Typical**	**Max.**	**Units**	**Conditions**
UT_1	F_{BRATE}	Baud Rate	Asynchronous SAMPR = 16x mode	—	—	—	Mbps	V_{DD} = 3.3V, C_{LOAD} = 30 pF(max)
				—	—	—	Mbps	V_{DD} = 1.8V, C_{LOAD} = 30 pF(max)
UT_3			Asynchronous SAMPR = 8x mode	—	—	—	Mbps	V_{DD} = 3.3V, C_{LOAD} = 30 pF(max)
				—	—	—	Mbps	V_{DD} = 1.8V, C_{LOAD} = 30 pF(max)
UT_5			Asynchronous SAMPR = 3x mode	—	—	—	Mbps	V_{DD} = 3.3V, C_{LOAD} = 30 pF(max)
				—	—	—	Mbps	V_{DD} = 1.8V, C_{LOAD} = 30 pF(max)
UT_19			Synchronous mode X2 mode	—	—	—	Mbps	V_{DD} = 3.3V, C_{LOAD} = 30 pF(max)
				—	—	—	Mbps	V_{DD} = 1.8V, C_{LOAD} = 30 pF(max)
UT_21			Synchronous mode X1 mode	—	—	—	Mbps	V_{DD} = 3.3V, C_{LOAD} = 30 pF(max)
				—	—	—	Mbps	V_{DD} = 1.8V, C_{LOAD} = 30 pF(max)
UT_23	f_{USART}	USART max $GCLK_{SERCOM}$		—	—	FCLK_23	MHz	—
UT_25	f_{XCK}	USART external clock input		—	—	—	MHz	—

Note:
1. These parameters are characterized, but not tested in manufacturing.

37.15. SERCOM Inter-Integrated Circuit (SERCOM I²C) Electrical Specifications

Figure 37-6. I²C Start/Stop Bits Host Mode Timing Diagrams

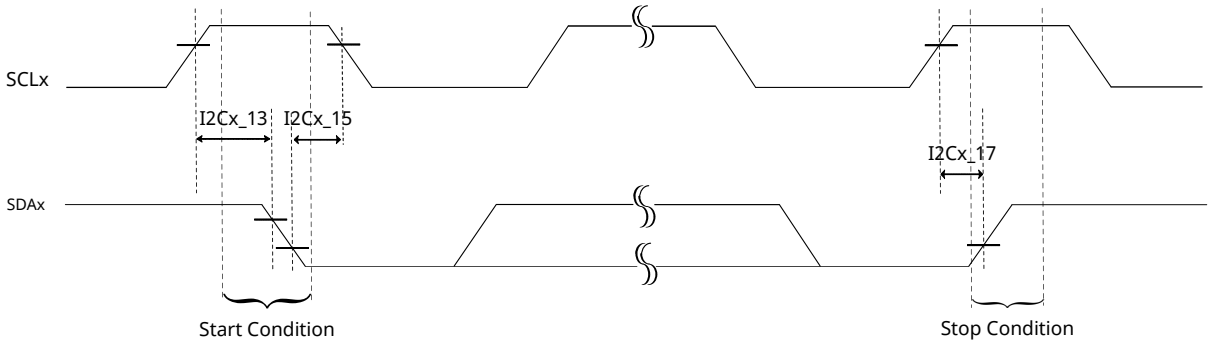


Figure 37-7. I²C Bus Data Host Mode Timing Diagrams

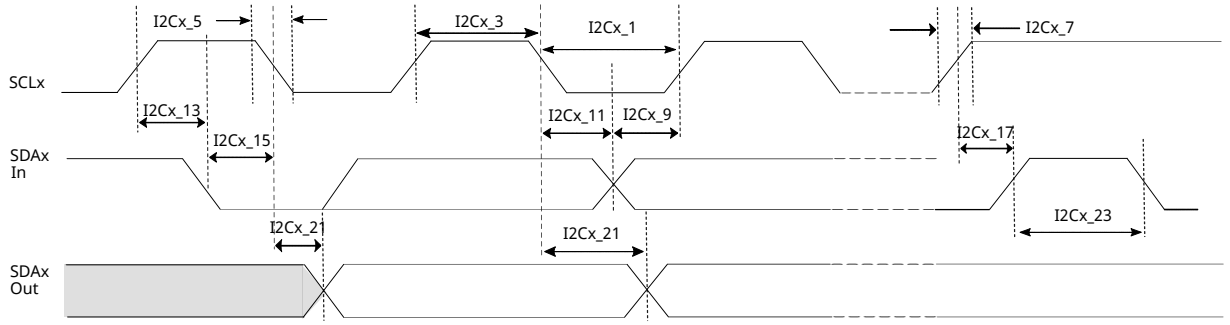


Table 37-26. I^2C Host Mode Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics[1]**		**Min.**	**Max.**	**Units**	**Conditions**
I2CM_1	$t_{L0:SCL}$	Host Clock Low Time	100 kHz mode	4.7	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	1.3	—	µs	
			1 MHz mode	0.5	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	320	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_3	$t_{HI:SCL}$	Host Clock High Time	100 kHz mode	4.0	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	120	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	60	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_5	$t_{F:SCL}$	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	—	300	ns	
			1 MHz mode	—	120	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	—	80	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	—	40	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_7	$t_{R:SCL}$	SDAn and SCLn Rise Time	100 kHz mode	—	1000	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	—	300	ns	
			1 MHz mode	—	120	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	—	80	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	—	40	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF

Table 37-26. I²C Host Mode Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated)
Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial

Param. No.	Symbol	Characteristics[1]		Min.	Max.	Units	Conditions
I2CM_9	$t_{SU:DAT}$	Data Setup Time	100 kHz mode	250	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	100	—	ns	
			1 MHz mode	50	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	10	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	10	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_11	$t_{HD:DAT}$[1]	Data Hold Time	100 kHz mode	300	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	300	—	ns	
			1 MHz mode	300	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	5.0	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	5.0	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_13	$t_{SU:STA}$	Start Condition Setup Time	100 kHz mode	4.7	—	µs	V_{DDIO}= 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_15	$t_{HD:STA}$	Start Condition Hold Time	100 kHz mode	4.0	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO2} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF

Table 37-26. I²C Host Mode Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics[1]**		**Min.**	**Max.**	**Units**	**Conditions**
I2CM_17	$t_{SU:ST0}$	Stop Condition Setup Time	100 kHz mode	4.0	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_21	$t_{AA:SCL}$	Output Valid from Clock	100 kHz mode	—	3.45	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	—	0.9	µs	
			1 MHz mode	—	0.45	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	—	100	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	—	100	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CM_23	$t_{BF:SDA}$[2]	Bus Free Time	100 kHz mode	4.7	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	1.3	—	µs	
			1 MHz mode	0.5	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	320	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF

Notes:

1. Longest delay between data hold timing based on bit field SDAHOLD of register CTRLA from SERCOM module and timing based on four periods of $GCLK_{SERCOM}$ for 100 kHz/400 kHz/1 MHz mode.
2. The amount of time the bus must be free before a new transmission starts (from STOP condition to START condition).

Figure 37-8. I²C Start/Stop Bits Client Mode Timing Diagram

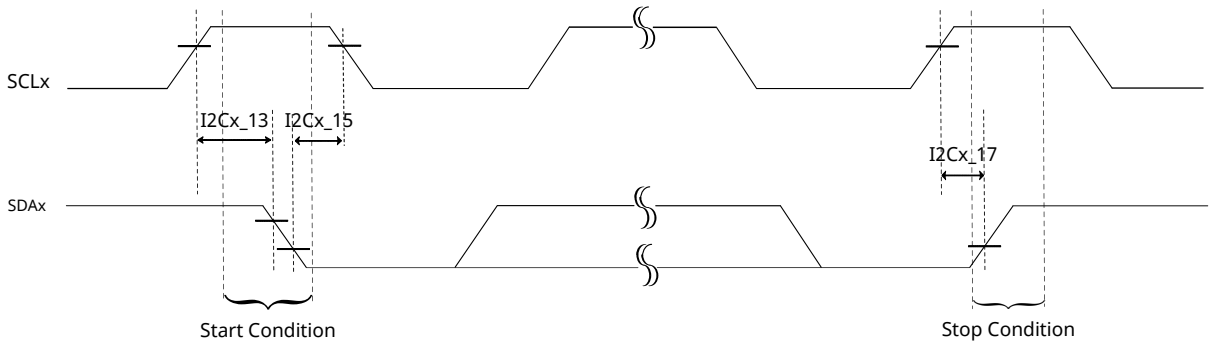


Figure 37-9. I²C Bus Data Client Mode Timing Diagrams

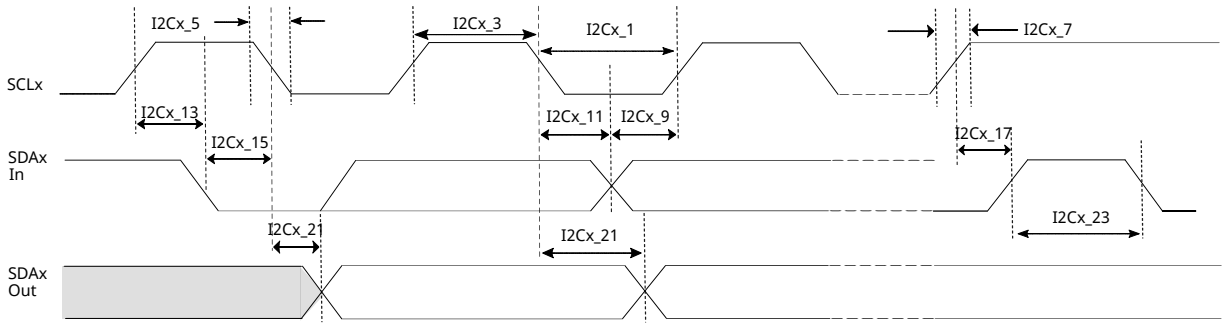


Table 37-27. I²C Client Mode AC Electrical Specifications

Standard Operating Conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics(1)**		**Min.**	**Max.**	**Units**	**Conditions**
I2CS_1	$t_{L0:SCL}$	Host Clock Low Time	100 kHz mode	4.7	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	1.3	—	µs	
			1 MHz mode	0.5	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz	320	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO}= 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_3	$t_{HI:SCL}$	Host Clock High Time	100 kHz mode	4.0	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	120	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	60	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_5	$t_{F:SCL}$	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	—	300	ns	
			1 MHz mode	—	120	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz	—	80	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	—	40	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF

Table 37-27. I²C Client Mode AC Electrical Specifications (continued)

Standard Operating Conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated)
Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial

Param. No.	Symbol	Characteristics[1]		Min.	Max.	Units	Conditions
I2CS_7	$t_{R:SCL}$	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	—	300	ns	
			1 MHz mode	—	120	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz	—	80	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	—	40	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_9	$t_{SU:DAT}$	Data Setup Time	100 kHz mode	250	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	100	—	ns	
			1 MHz mode	50	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz	10	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	10	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_11	$t_{HD:DAT}$[1]	Data Hold Time	100 kHz mode	300	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	300	—	ns	
			1 MHz mode	300	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz	5.0	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	5.0	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_13	$t_{SU:STA}$	Start Condition Setup Time	100 kHz mode	4.7	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_15	$t_{HD:STA}$	Start Condition Hold Time	100 kHz mode	4.0	—	µs	V_{DDIO} = 5.0V, $I_{PULL-UP}$ = 3 ma, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 5.0V, $I_{PULL-UP}$ = 20 ma, C_{LOAD} = 550 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 5.0V, $I_{PULL-UP}$ = 20 ma, C_{LOAD} = 100 pF

Table 37-27. I²C Client Mode AC Electrical Specifications (continued)

Standard Operating Conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated)
Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial

Param. No.	Symbol	Characteristics[1]		Min.	Max.	Units	Conditions
I2CS_17	$t_{SU:ST0}$	Stop Condition Setup Time	100 kHz mode	4.0	—	µs	V^{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	0.6	—	µs	
			1 MHz mode	0.26	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_21	$t_{AA:SCL}$	Output Valid from Clock	100 kHz mode	—	3.45	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	—	0.9	µs	
			1 MHz mode	—	0.45	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	—	100	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	—	100	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF
I2CS_23	$t_{BF:SDA}$[2]	Bus Free Time	100 kHz mode	4.7	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 3 mA, C_{LOAD} = 400 pF
			400 kHz mode	1.3	—	µs	
			1 MHz mode	0.5	—	µs	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 20 mA, C_{LOAD} = 550 pF
			1.7 MHz mode	320	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 400 pF
			3.4 MHz mode	160	—	ns	V_{DDIO} = 3.3V, $I_{PULL-UP}$ = 12 mA, C_{LOAD} = 100 pF

Notes:

1. Longest delay between data hold timing based on bit field SDAHOLD of register CTRLA from SERCOM module and timing based on four period of $GCLK_{SERCOM}$ for 100 kHz/400 kHz/1 MHz mode.
2. The amount of time the bus must be free before a new transmission starts (from STOP condition to START condition).

37.16. Timer Counter (TC) Module Electrical Specifications

Figure 37-10. TCn Timer Capture Input Module Timing Diagram

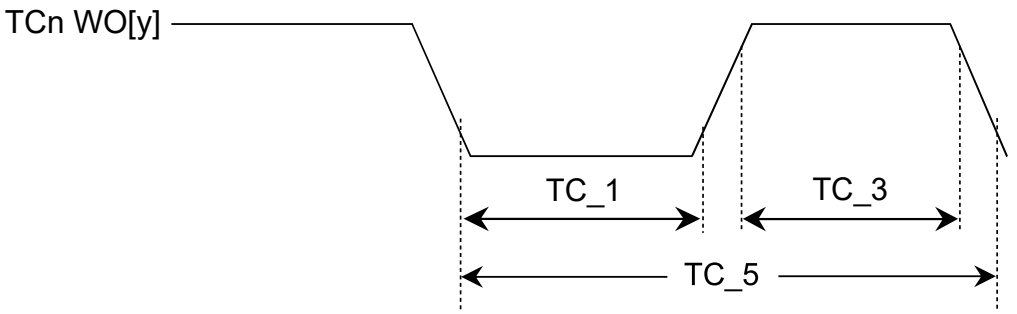


Figure 37-11. TCn Timer Compare Output Module Timing Diagram

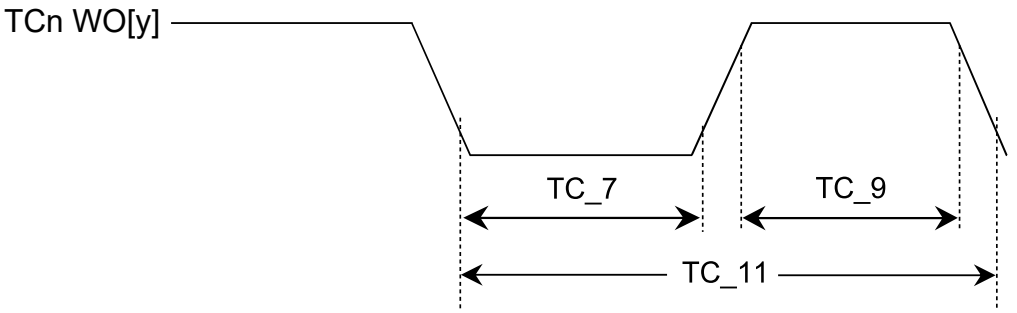


Table 37-28. TCn Timer Capture Module Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units	Conditions
TC_1	TC_{INLOW}	Capture TCn Input Low Time	$2/f_{GLK\_TCn}$	—	—	µs	V_{DDIO}(min) and meet TC_5 specifications
TC_3	TC_{INHIGH}	Capture TCn Input High Time	$2/f_{GLK\_TCn}$	—	—	µs	V_{DDIO}(min) and meet TC_5 specifications
TC_5	$TC_{INPERIOD}$	Capture Input Period	$4/f_{GLK\_TCn}$	—	—	µs	V_{DDIO}(min)
TC_7	TC_{OUTLOW}	Compare TCn Output Low Time	3 x DI_27	—	—	ns	V_{DDIO}(min) and meet TC_11 specifications
TC_9	$TC_{OUTHIGH}$	Compare TCn Output High Time	3 x DI_25	—	—	ns	V_{DDIO}(min) and meet TC_11 specifications
TC_11	$TC_{OUTPERIOD}$	Compare Output Period	TC_7 + TC_9	—	—	ns	V_{DDIO}(min)
TC_13	f_{GCLK_TCn}	TC peripheral module clock frequency	—	—	FCLK_37	MHz	V_{DDIO}(min)

37.17. Timer/Counter for Control (TCC) Application Electrical Specifications

Figure 37-12. TCCn Timer Capture Input Module Timing Diagrams

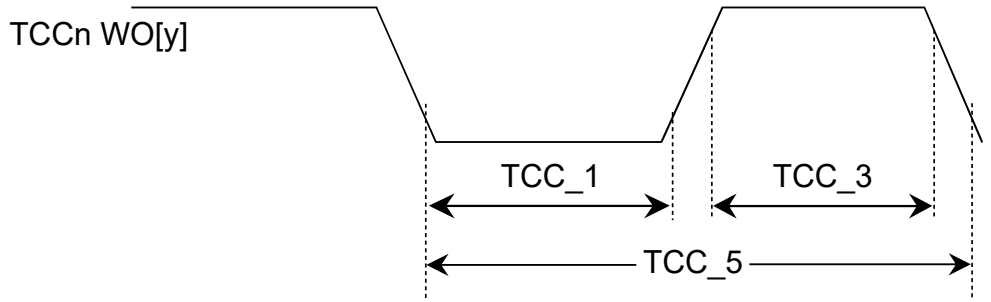


Figure 37-13. TCCn Timer Compare Output Module Timing Diagrams

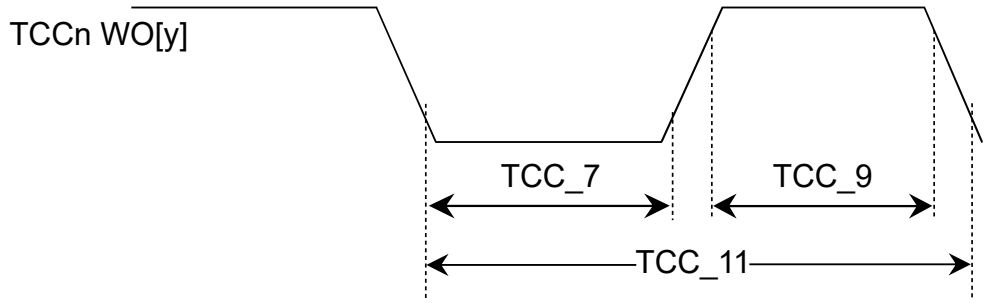


Figure 37-14. TCCn Timer Compare Fault Output Module Timing Diagrams

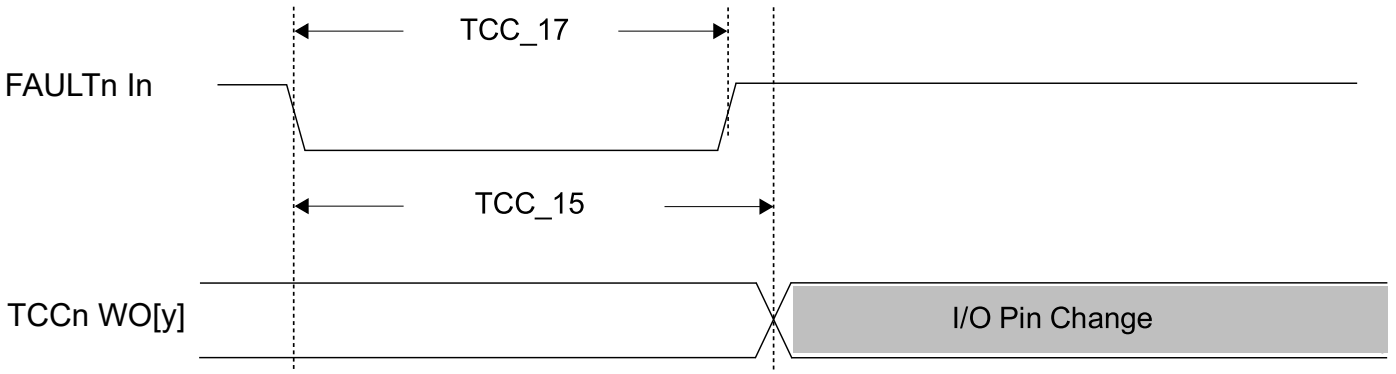


Table 37-29. TCCn Timer Capture Module Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) **Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial**							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
TCC_1	TCC_{INLOW}	Capture TCCn Input Low Time	$2/f_{GLK\_TCCn}$	—	—	µs	V_{DDIO}(min) and meet TCC_5 specifications
TCC_3	TCC_{INHIGH}	Capture TCCn Input High Time	$2/f_{GLK\_TCCn}$	—	—	µs	V_{DDIO}(min) and meet TCC_5 specifications
TCC_5	$TCC_{INPERIOD}$	Capture Input Period	$4/f_{GLK\_TCCn}$	—	—	µs	V_{DDIO}(min)
TCC_7	TCC_{OUTLOW}	Compare TCCn Output Low Time	3 x DI_27	—	—	ns	V_{DDIO}(min) and meet TCC_11 specifications

Table 37-29. TCCn Timer Capture Module Electrical Specifications (continued)

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
TCC_9	$TCC_{OUTHIGH}$	Compare TCCn Output High Time	3 x DI_25	—	—	ns	V_{DDIO}(min) and meet TCC_11 specifications
TCC_11	$TCC_{OUTPERIOD}$	Compare Output Period	TCC_7 + TCC_9	—	—	ns	V_{DDIO}(min)
TCC_13	f_{GCLK_TCCn}	TCC peripheral module clock frequency	—	—	FCLK_35	MHz	
TCC_15	TCC_{FD}	Fault input to I/O pin change	—	43	—	ns	V_{DDIO}(typ)
TCC_17	TCC_{FLT}	Fault input pulse width	1/(2 x (t_{RISE} + t_{FALL}))	—	—	ns	V_{DDIO}(min)

37.18. NVM Block (Flash, Data Flash and NVM Configuration Rows) Electrical Specifications

Table 37-30. Flash NVM Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial								
Param. No.	**Symbol**	**Characteristics**		**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
NVM_1	F_{RETEN}	Flash data retention		20	—	—	Yrs	Under all conditions less than the absolute maximum rating specifications
NVM_3	E_P	Cell endurance (Flash erase and write operation)		1k	—	—	Cycles	
NVM_4		High endurance Flash (Flash erase and write operation)		10k	—	—		
NVM_7	T_{FPW}	Program cycle time	Word write	—	—	150	µs	V_{DD} = 1.8V or V_{DD}(min) whichever is greater
NVM_9	T_{CE}		Chip erase	—	—	11	ms	
NVM_11	T_{FEB}		Page erase	—	—	11	ms	
NVM_13	I_{DDPROG}	Supply current during programming		—	—	10	mA	V_{DD} = 3.0V
NVM_15	$I_{DDERASE}$	Supply current during NVM erase		—	—	7	mA	V_{DD} = 3.0V

Notes:

1. The high endurance Flash applies to the memory locations from `0x0C00FC00` to `0x0C00FFFF`.
2. Times are with the OSCHF enabled.

37.19. Configurable Custom Logic (CCL) Electrical Specifications

Table 37-31. Configurable Custom Logic (CCL) Electrical Specifications

Standard operating conditions: V_{DDIO} = AVDD = 1.8V to 5.5V (Unless otherwise stated) Operating temperature: -40°C ≤ T_A ≤ +85°C for Industrial							
Param. No.	**Symbol**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
CCL_1	$t_{SU\_CCL\_FF}$	Setup time for CCL data to clock for flip-flop modes	—	—	—	ns	
CCL_3	f_{GCLK_CCL}	CCL peripheral clock module	—	—	FCLK_43	MHz	

38. Graphical Characteristics

38.1. I/O Pin Graphical Characteristics

Figure 38-1. Fall Time vs. V_{DD} (SLEWLIM = b'0)

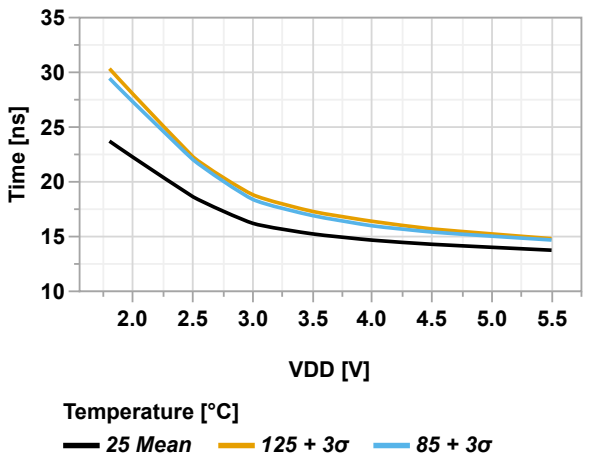


Figure 38-2. Fall Time vs. V_{DD} (SLEWLIM = b'1)

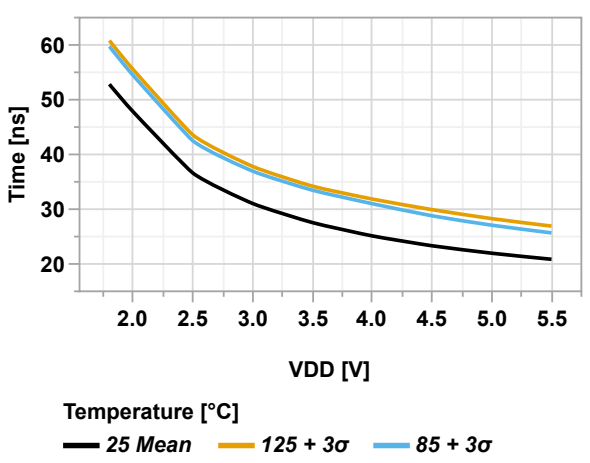


Figure 38-3. Rise Time vs. V_{DD} (SLEWLIM = b'0)

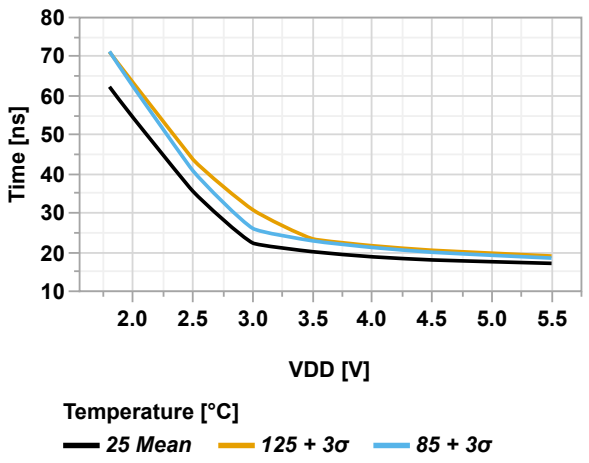


Figure 38-4. Rise Time vs. V_{DD} (SLEWLIM = b'1)

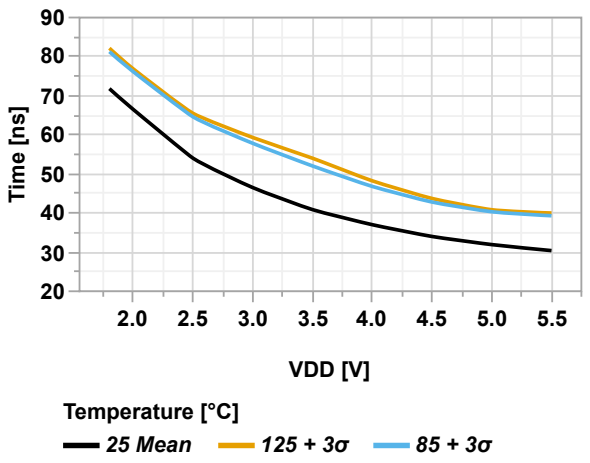


Figure 38-5. Input Pin with Schmitt Trigger - Maximum V_{IL} vs. V_{DD}

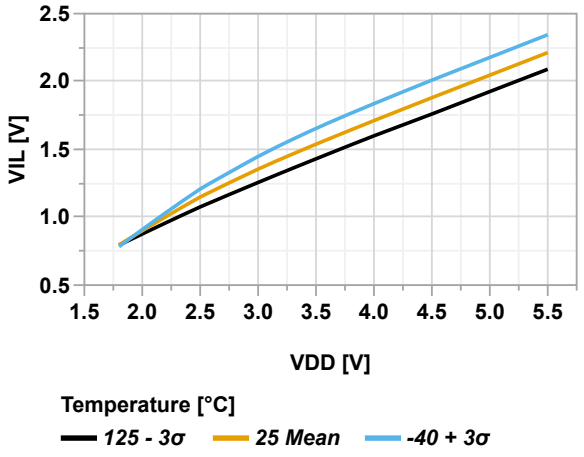


Figure 38-6. Input Pin with Schmitt Trigger - Minimum V_{IH} vs. V_{DD}

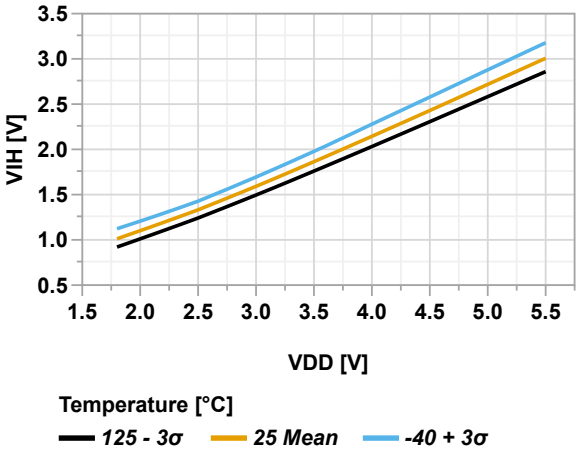


Figure 38-7. Input Pin with Schmitt Trigger - Hysteresis vs. V_{DD}

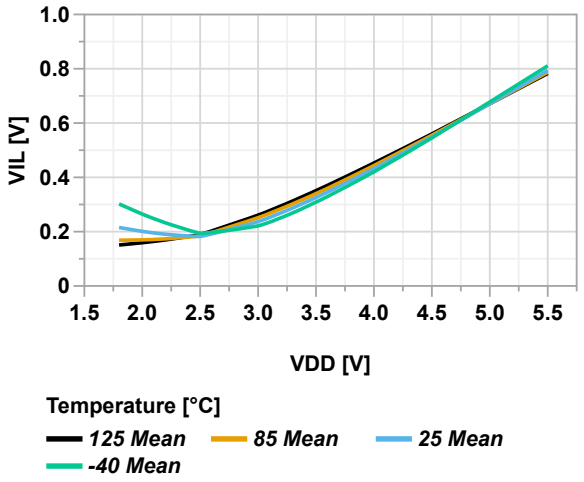


Figure 38-8. Input Pin with I^2C Trigger - Maximum V_{IL} vs. V_{DD}

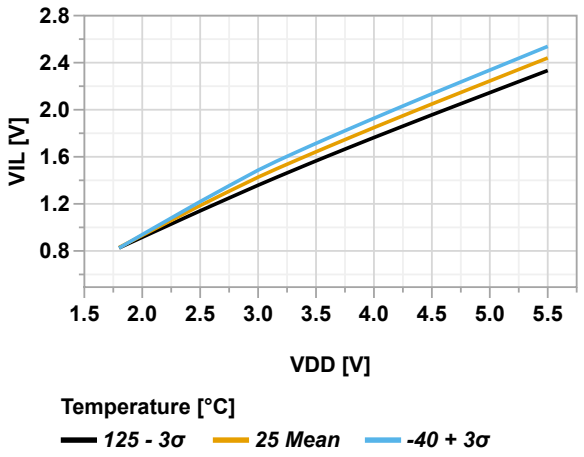


Figure 38-9. Input Pin with I^2C Trigger - Minimum V_{IH} vs. V_{DD}

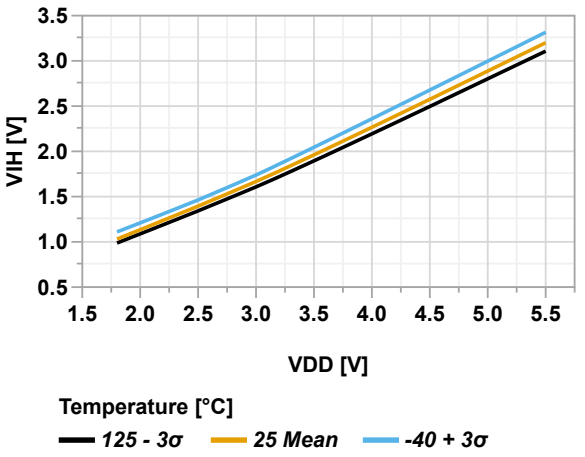


Figure 38-10. Output Pin - Maximum V_{OL} vs. Current, V_{DD} = 1.8V

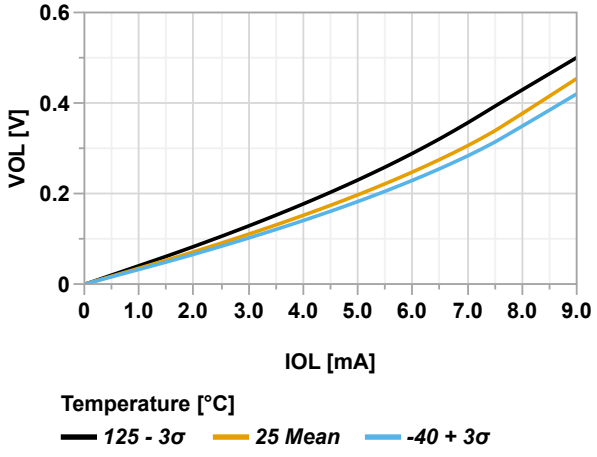


Figure 38-11. Output Pin - Minimum V_{OH} vs. Current, V_{DD} = 1.8V

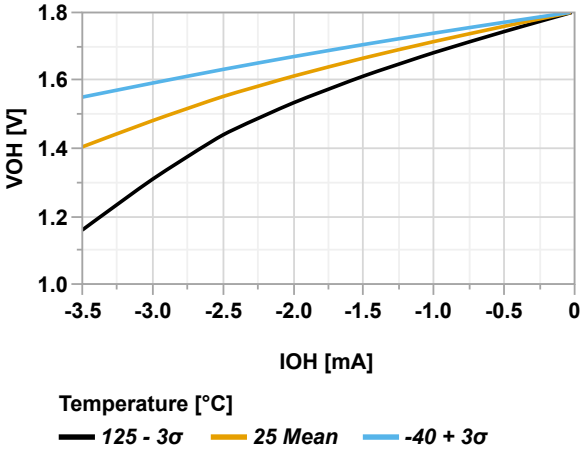


Figure 38-12. Output Pin - Maximum V_{OL} vs. Current, V_{DD} = 3.0V

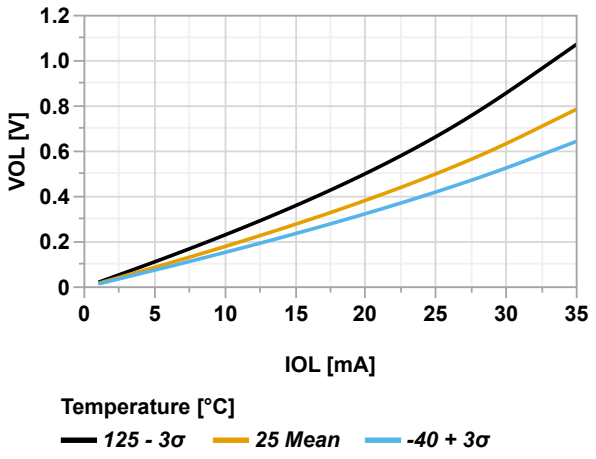


Figure 38-13. Output Pin - Minimum V_{OH} vs. Current, V_{DD} = 3.0V

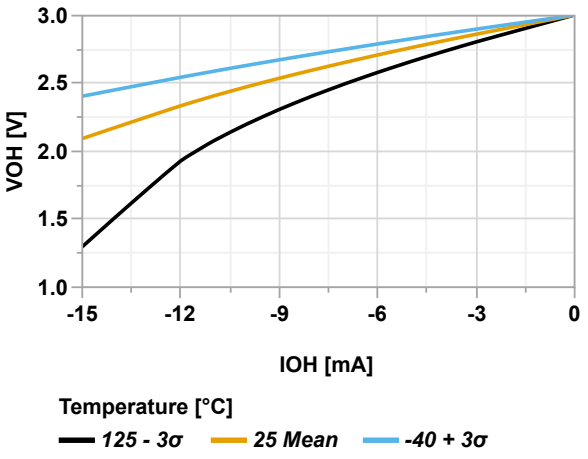


Figure 38-14. Output Pin - Maximum V_{OL} vs. Current, V_{DD} = 5.5V

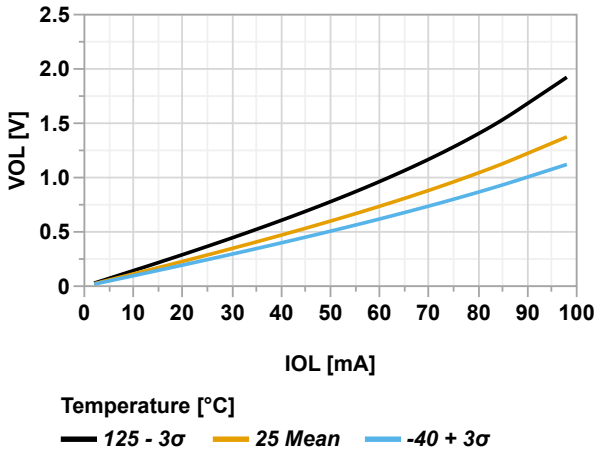
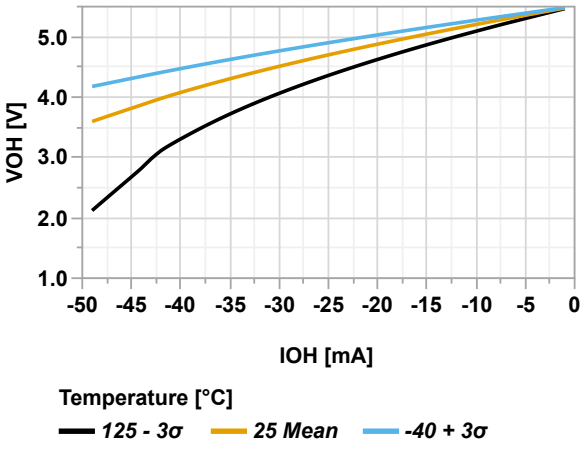


Figure 38-15. Output Pin - Minimum V_{OH} vs. Current, V_{DD} = 5.5V



39. Packaging Information

39.1. Package Marking Information

All devices are marked with the Microchip logo, the ordering code (full or truncated), the package style, and a trace code.

Legend:	XX...X	Customer-specific information or Microchip part number
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC® designator for Matte Tin (Sn)
Note:	In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.	

39.1.1. 64-Pin VQFN Wettable Flanks

Figure 39-1. General

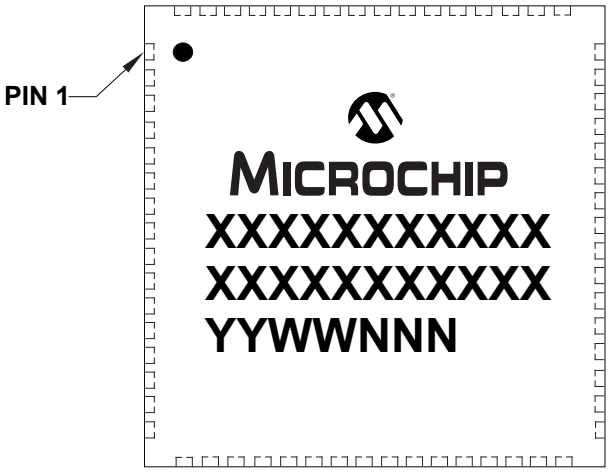
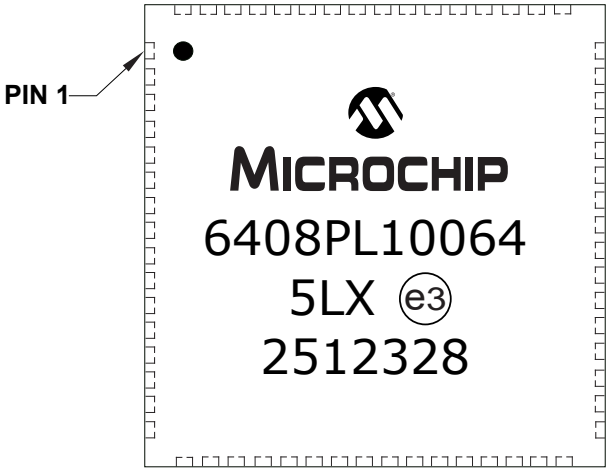


Figure 39-2. Example



39.1.2. 64-Pin TQFP

Figure 39-3. General

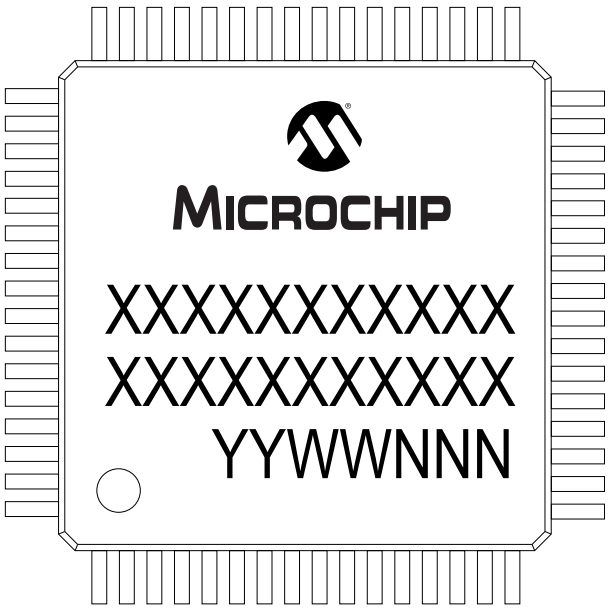
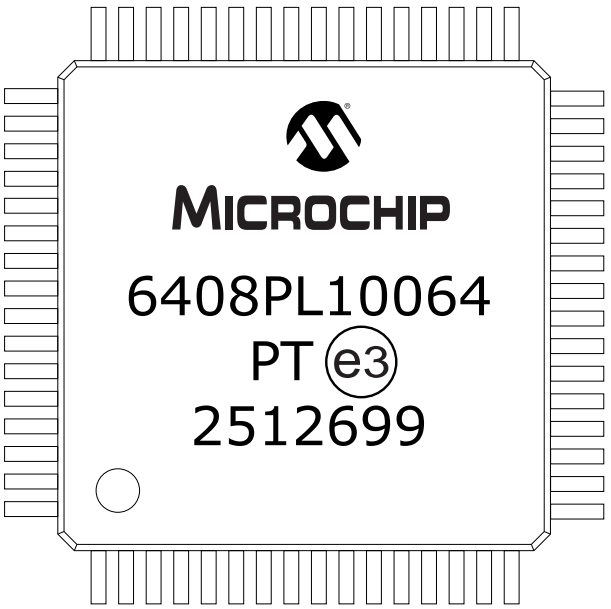


Figure 39-4. Example



39.1.3. 48-Pin VQFN Wettable Flanks

Figure 39-5. General

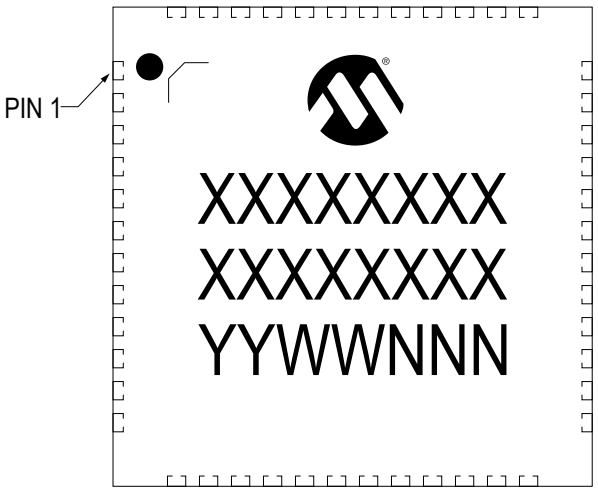
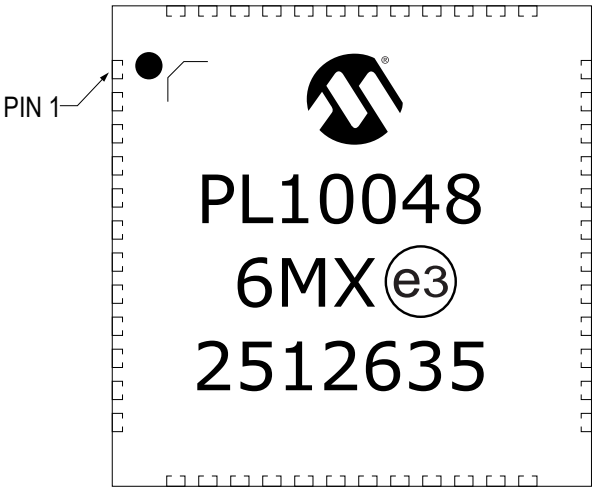


Figure 39-6. Example



39.1.4. 48-Pin TQFP

Figure 39-7. General

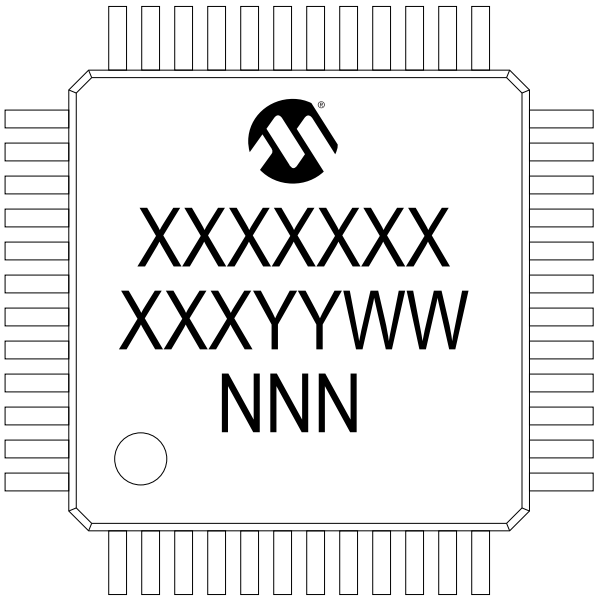
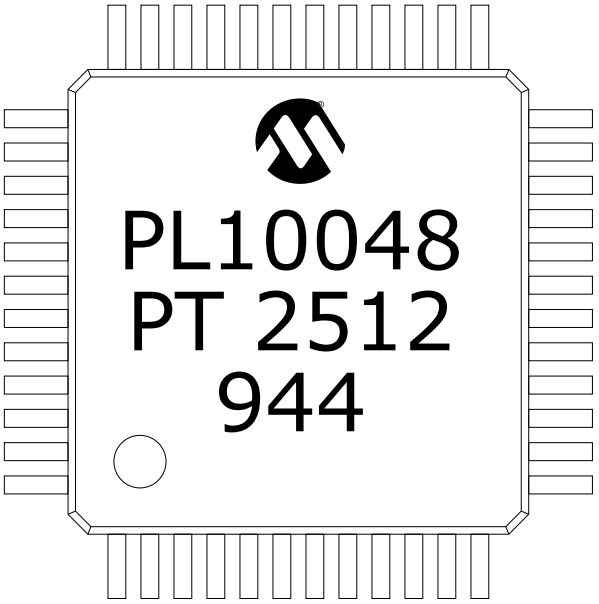


Figure 39-8. Example



39.1.5. 32-Pin VQFN Wettable Flanks

Figure 39-9. General

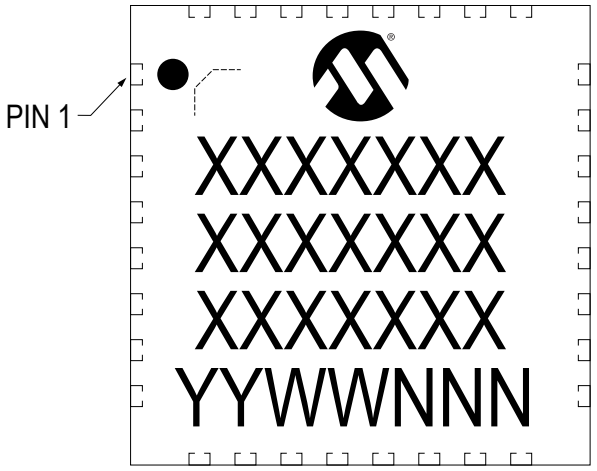
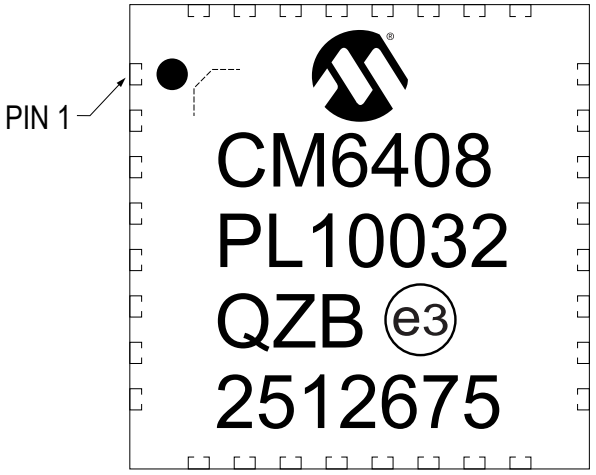


Figure 39-10. Example



39.1.6. 32-Pin TQFP

Figure 39-11. General

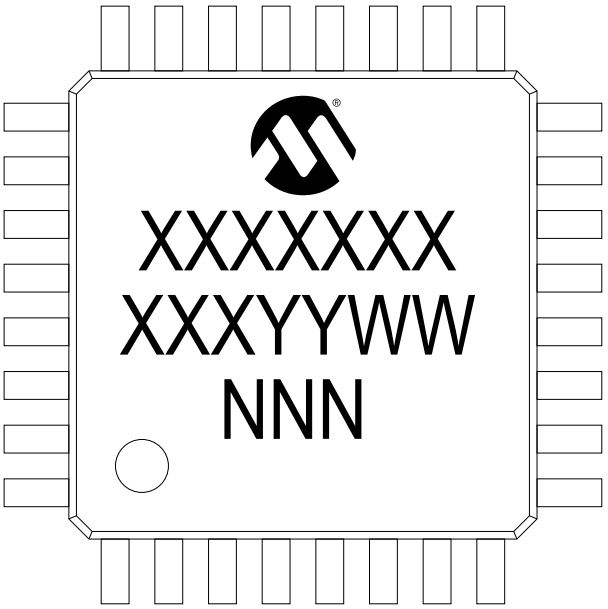
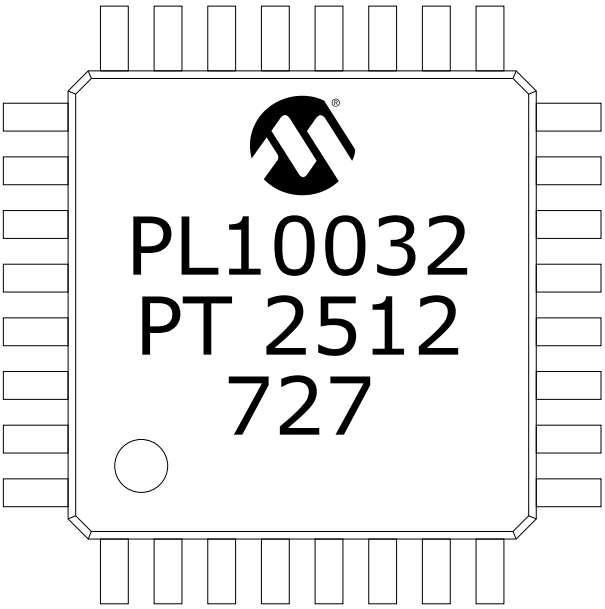


Figure 39-12. Example



39.1.7. 28-Pin VQFN Wettable Flanks

Figure 39-13. General

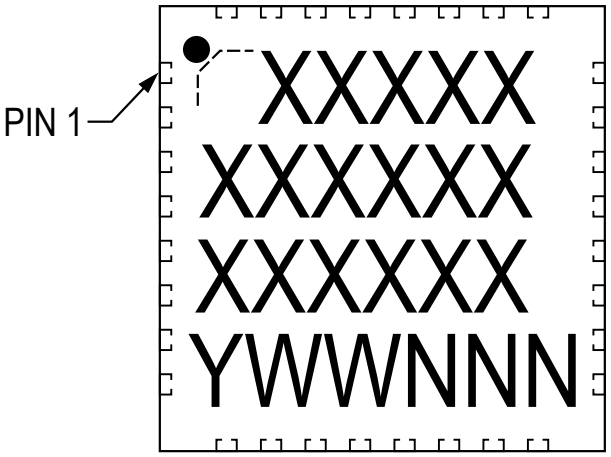
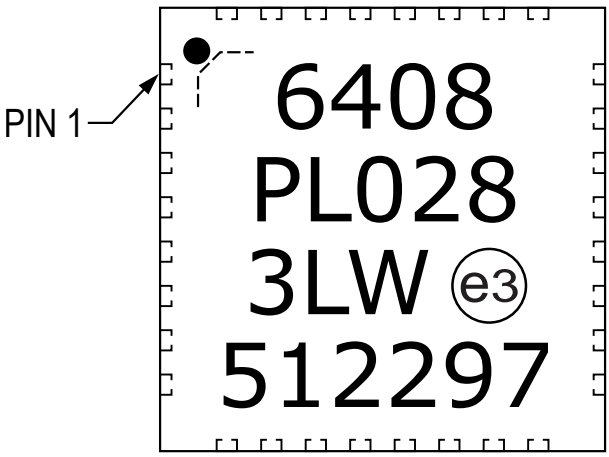


Figure 39-14. Example



39.1.8. 28-Pin SSOP

Figure 39-15. General

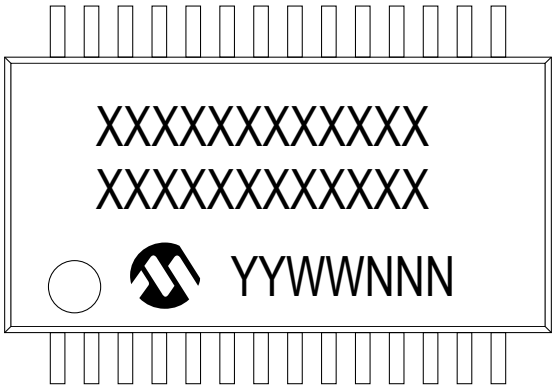
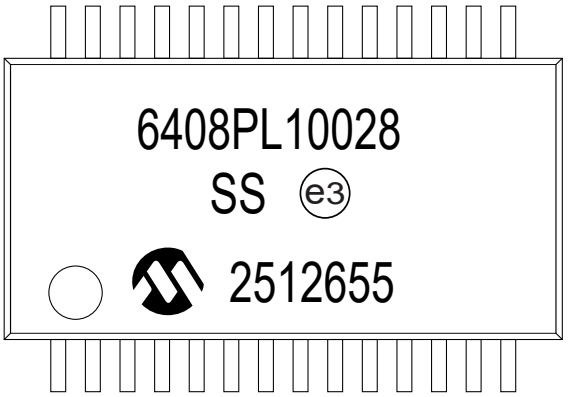


Figure 39-16. Example



39.1.9. 28-Pin SPDIP

Figure 39-17. General

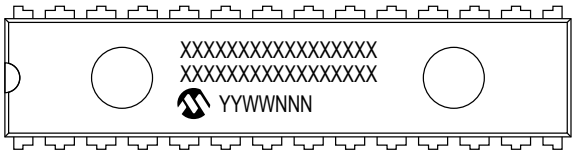


Figure 39-18. Example

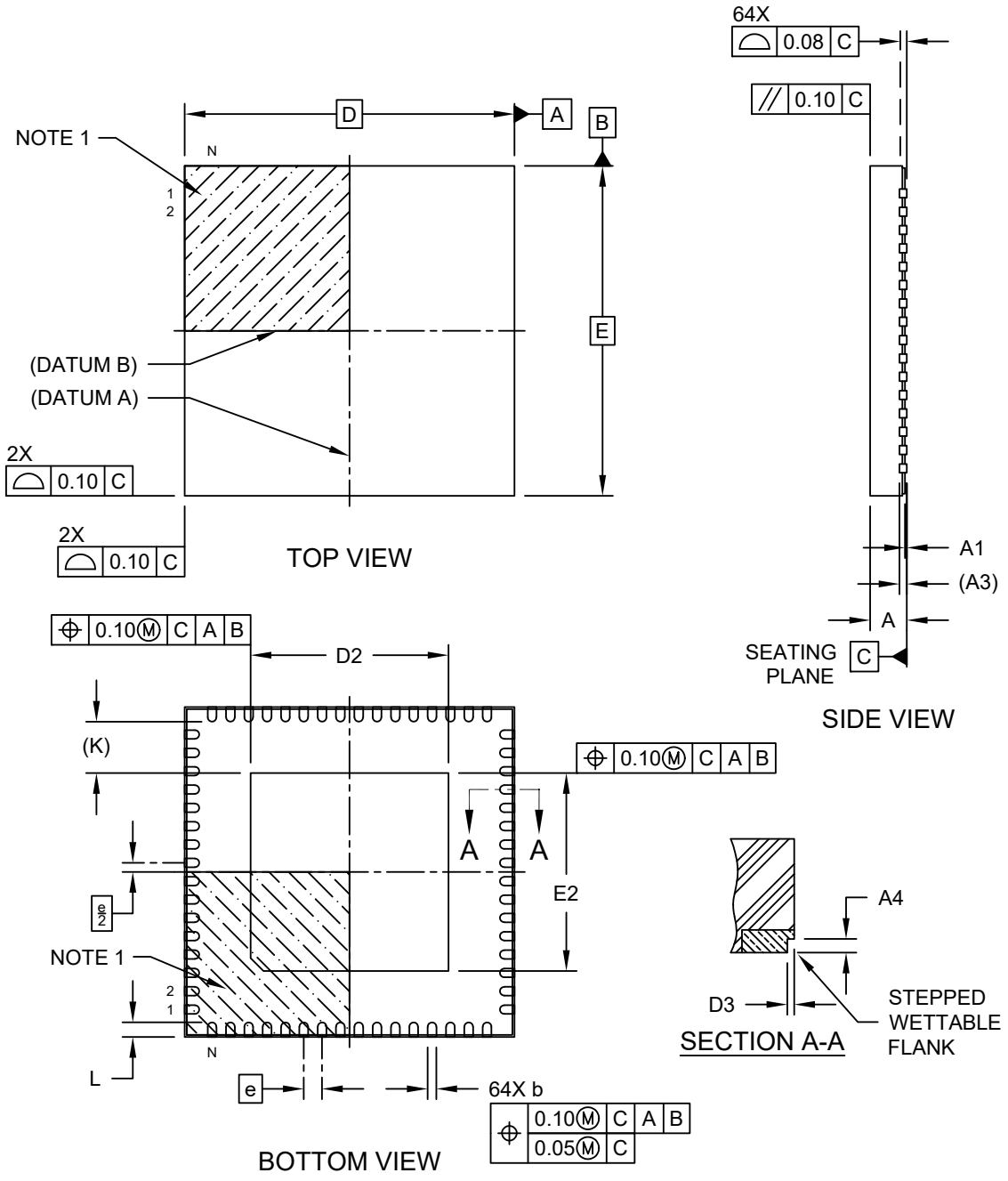
39.2. Package Drawings

Note: For current package drawings, refer to the Microchip Packaging Specification, which is available at http://www.microchip.com/packaging.

39.2.1. 64-Pin VQFN Wettable Flanks

64-Lead Very Thin Plastic Quad Flat, No Lead Package (5LX) - 9x9x1.0 mm Body [VQFN] With 5.4 mm Exposed Pad and Stepped Wettable Flanks

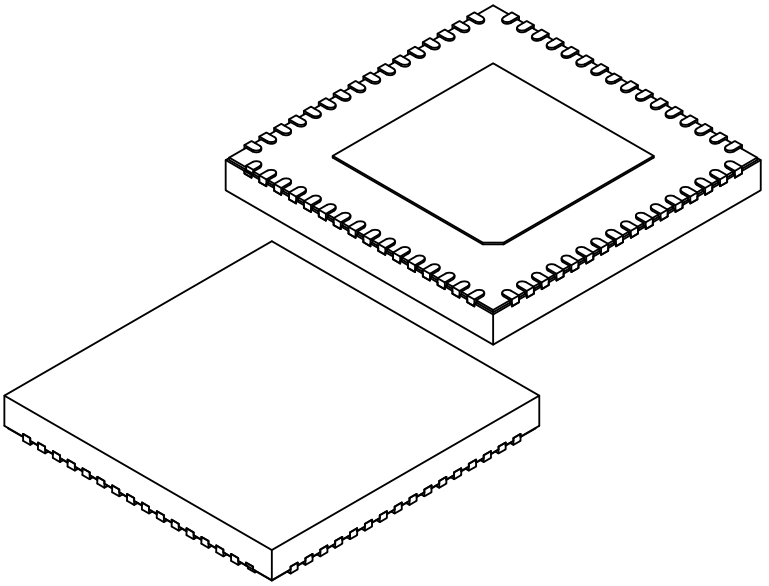
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-483-5LX Rev F Sheet 1 of 2

64-Lead Very Thin Plastic Quad Flat, No Lead Package (5LX) - 9x9x1.0 mm Body [VQFN] With 5.4 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		64	
Pitch	e		0.50 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		9.00 BSC	
Exposed Pad Length	D2	5.30	5.40	5.50
Overall Width	E		9.00 BSC	
Exposed Pad Width	E2	5.30	5.40	5.50
Terminal Width	b	0.20	0.25	0.30
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K		1.40 REF	
Wettable Flank Step Length	D3	0.035	0.060	0.085
Wettable Flank Step Height	A4	0.10	-	0.19

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

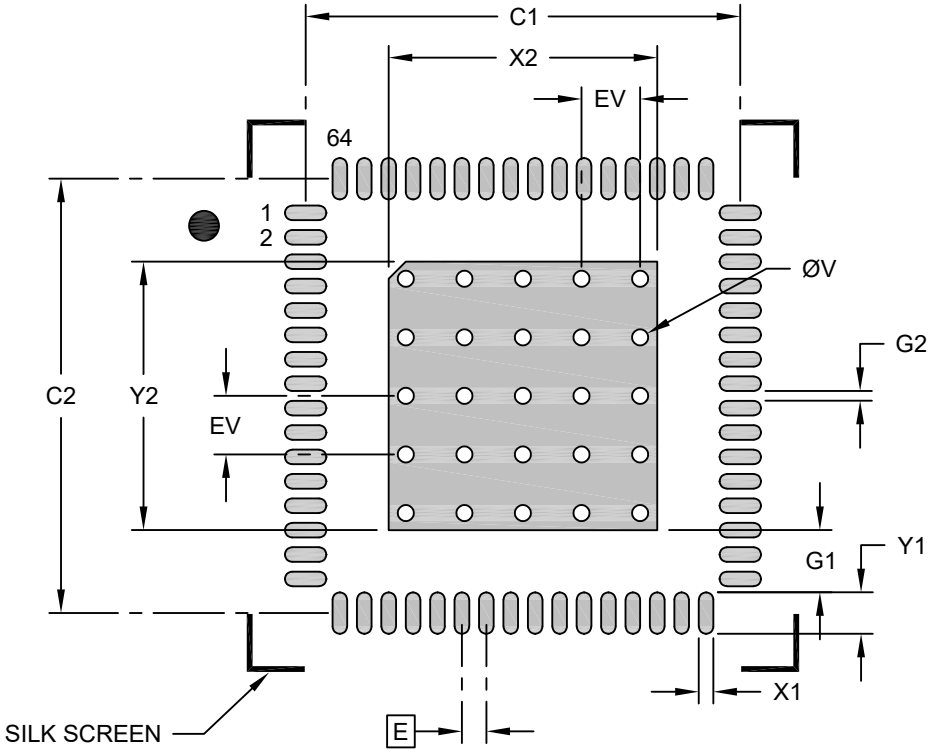
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-483-5LX Rev F Sheet 2 of 2

64-Lead Very Thin Plastic Quad Flat, No Lead Package (5LX) - 9x9x1.0 mm Body [VQFN] With 5.4 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Optional Center Pad Width	X2			5.50
Optional Center Pad Length	Y2			5.50
Contact Pad Spacing	C1		8.90	
Contact Pad Spacing	C2		8.90	
Contact Pad Width (X64)	X1			0.30
Contact Pad Length (X64)	Y1			0.85
Contact Pad to Center Pad (X64)	G1	1.28		
Contact Pad to Contact Pad (X60)	G2	0.20		
Thermal Via Diameter	V		0.33	
Thermal Via Pitch	EV		1.20	

Notes:

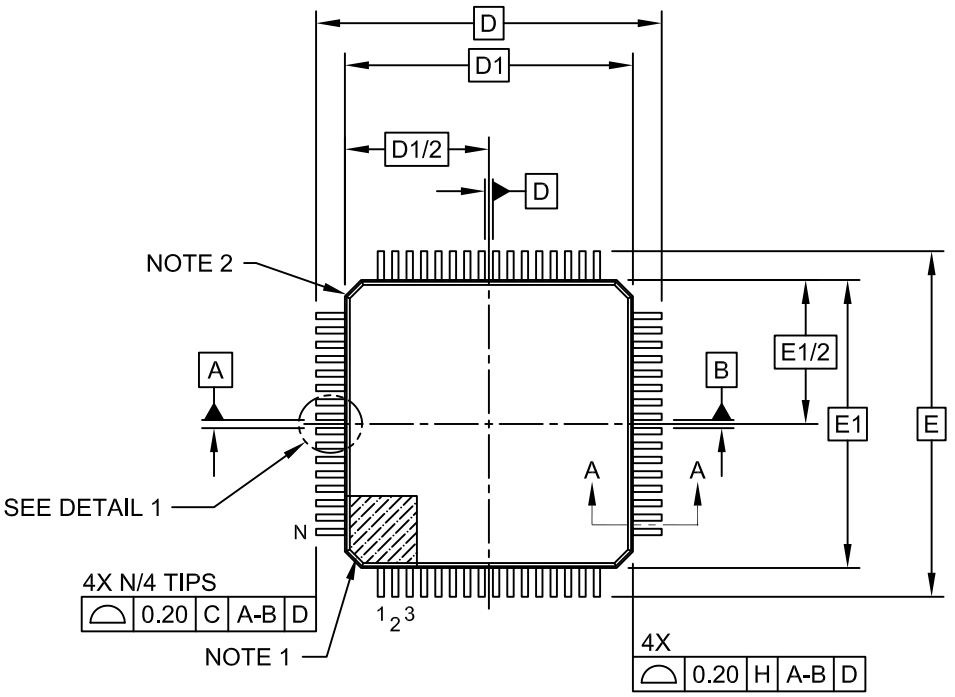
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2483-5LX Rev F

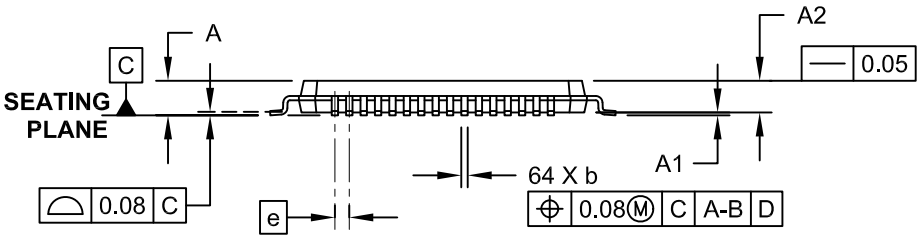
39.2.2. 64-Pin TQFP

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

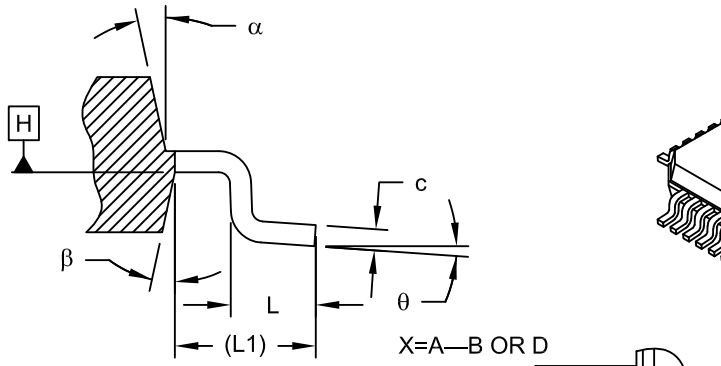


SIDE VIEW

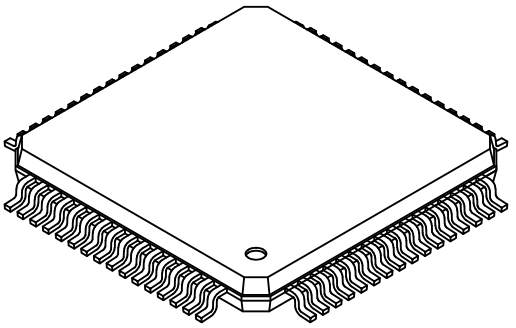
Microchip Technology Drawing C04-085-PT Rev E Sheet 1 of 2

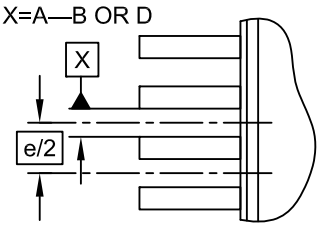
64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



SECTION A-A





DETAIL 1

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		64	
Lead Pitch	e		0.50 BSC	
Overall Height	A	-	-	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	-	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	θ	0°	3.5°	7°
Overall Width	E		12.00 BSC	
Overall Length	D		12.00 BSC	
Molded Package Width	E1		10.00 BSC	
Molded Package Length	D1		10.00 BSC	
Lead Thickness	c	0.09	-	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

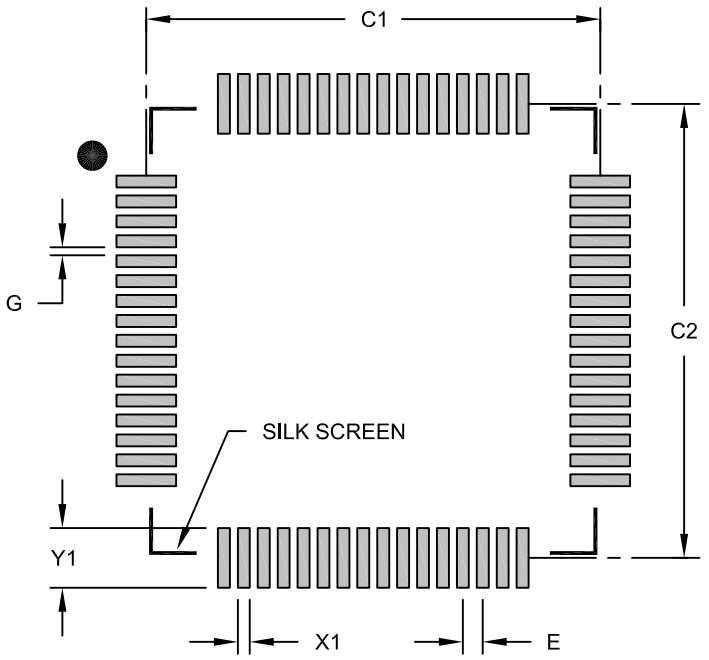
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085-PT Rev E Sheet 2 of 2

64-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 2.00 mm Footprint [TQFP]



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X64)	X1			0.30
Contact Pad Length (X64)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

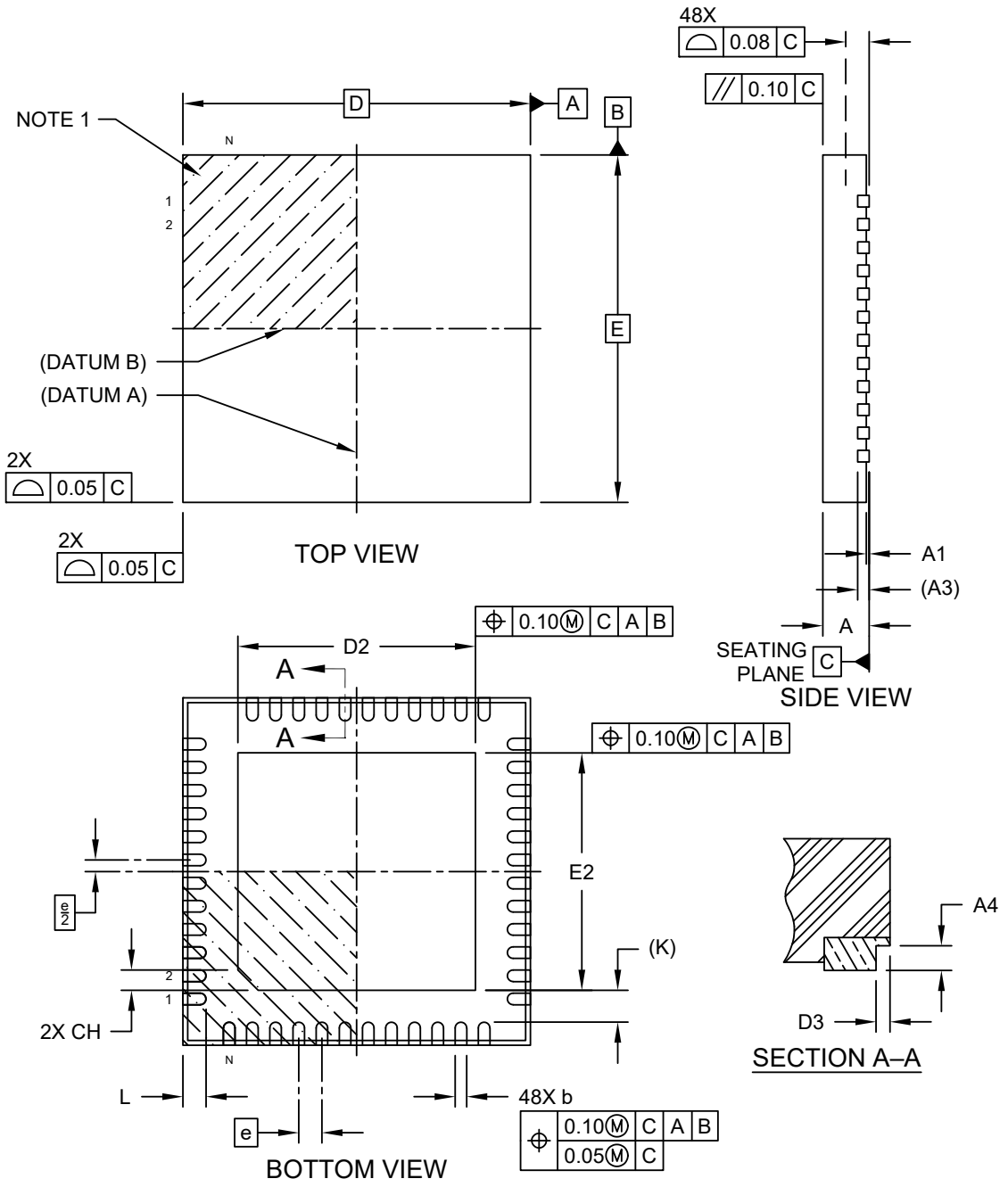
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2085-PT Rev E

39.2.3. 48-Pin VQFN Wettable Flanks

48-Lead Very Thin Plastic Quad Flat, No Lead Package (6MX) - 6x6 mm Body [VQFN] With 4.1x4.1 mm Exposed Pad and Stepped Wettable Flanks

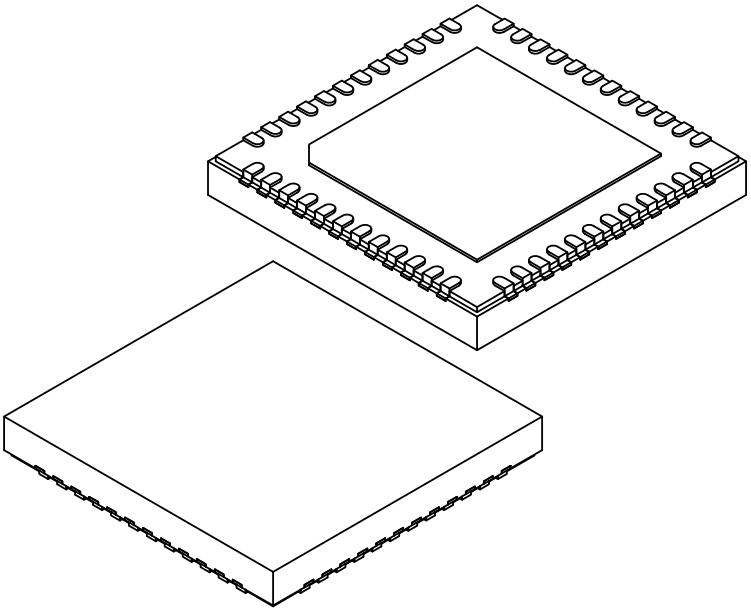
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-504-6MX Rev B Sheet 1 of 2

48-Lead Very Thin Plastic Quad Flat, No Lead Package (6MX) - 6x6 mm Body [VQFN] With 4.1x4.1 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		48	
Pitch	e		0.40 BSC	
Overall Height	A	0.80	0.85	0.90
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.20 REF	
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	4.00	4.10	4.20
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	4.00	4.10	4.20
Exposed Pad Corner Chamfer	CH		0.35 REF	
Terminal Width	b	0.15	0.20	0.25
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K		0.55 REF	
Wettable Flank Step Length	D3	-	-	0.085
Wettable Flank Step Height	A4	0.10	-	0.19

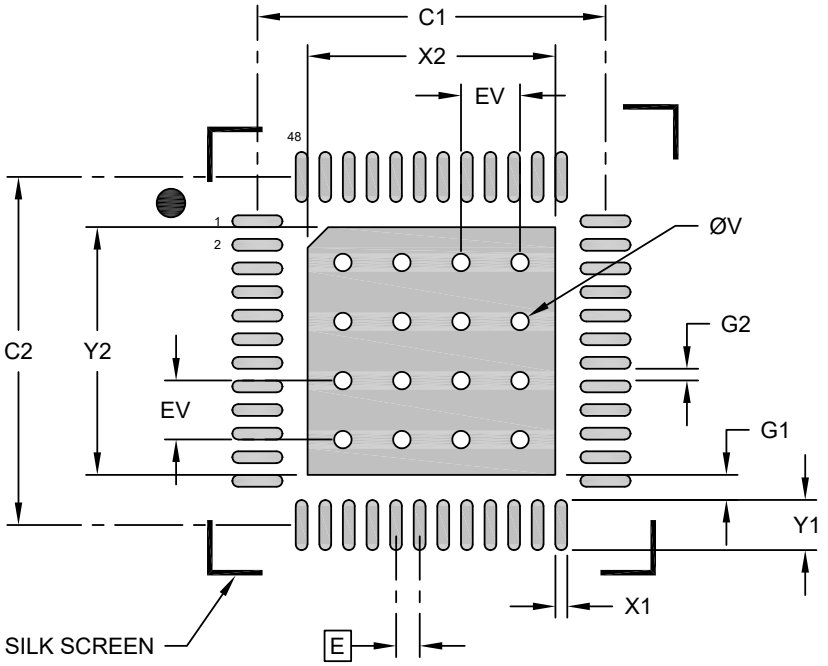
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-504-6MX Rev B Sheet 2 of 2

48-Lead Very Thin Plastic Quad Flat, No Lead Package (6MX) - 6x6 mm Body [VQFN] With 4.1x4.1 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.40 BSC	
Optional Center Pad Width	X2			4.20
Optional Center Pad Length	Y2			4.20
Contact Pad Spacing	C1		5.90	
Contact Pad Spacing	C2		5.90	
Contact Pad Width (X48)	X1			0.20
Contact Pad Length (X48)	Y1			0.85
Contact Pad to Center Pad (X48)	G1	0.20		
Contact Pad to Contact Pad (X44)	G2	0.20		
Thermal Via Diameter	V		0.30	
Thermal Via Pitch	EV		1.00	

Notes:

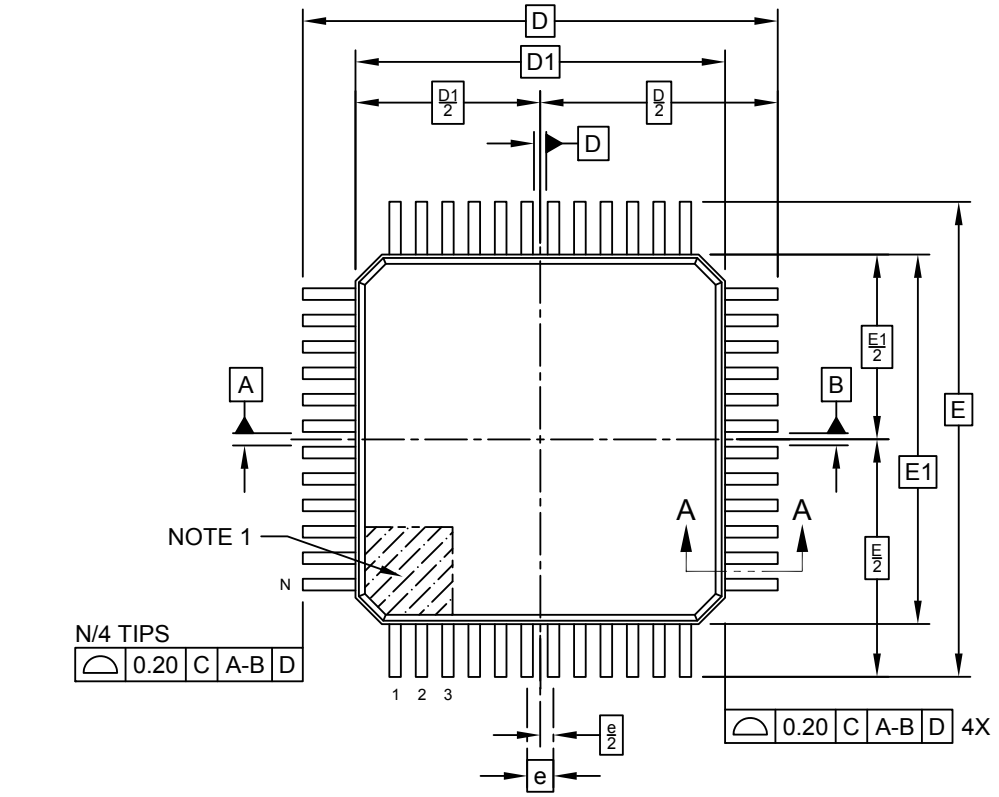
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2504-6MX Rev B

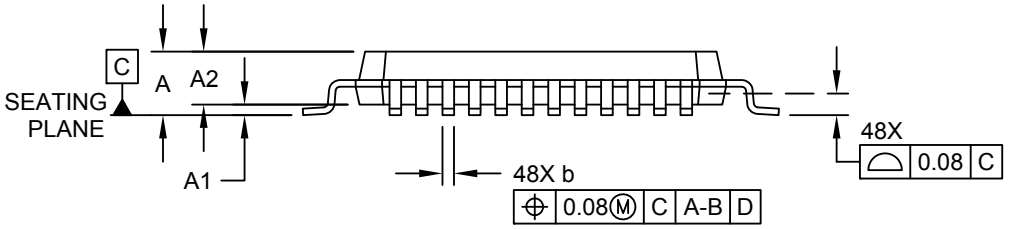
39.2.4. 48-Pin TQFP

48-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

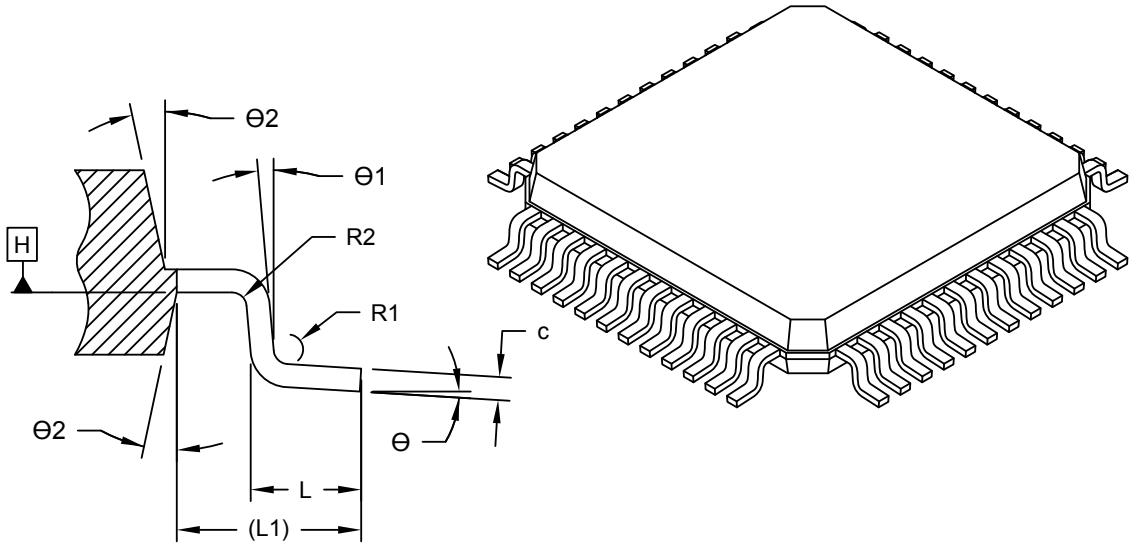


SIDE VIEW

Microchip Technology Drawing C04-300-PT Rev D Sheet 1 of 2

48-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



SECTION A-A

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		48	
Pitch	e		0.50 BSC	
Overall Height	A	-	-	1.20
Standoff	A1	0.05	-	0.15
Molded Package Thickness	A2	0.95	1.00	1.05
Overall Length	D		9.00 BSC	
Molded Package Length	D1		7.00 BSC	
Overall Width	E		9.00 BSC	
Molded Package Width	E1		7.00 BSC	
Terminal Width	b	0.17	0.22	0.27
Terminal Thickness	c	0.09	-	0.16
Terminal Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Lead Bend Radius	R1	0.08	-	-
Lead Bend Radius	R2	0.08	-	0.20
Foot Angle	Θ	0°	3.5°	7°
Lead Angle	Θ1	0°	-	-
Mold Draft Angle	Θ2	11°	12°	13°

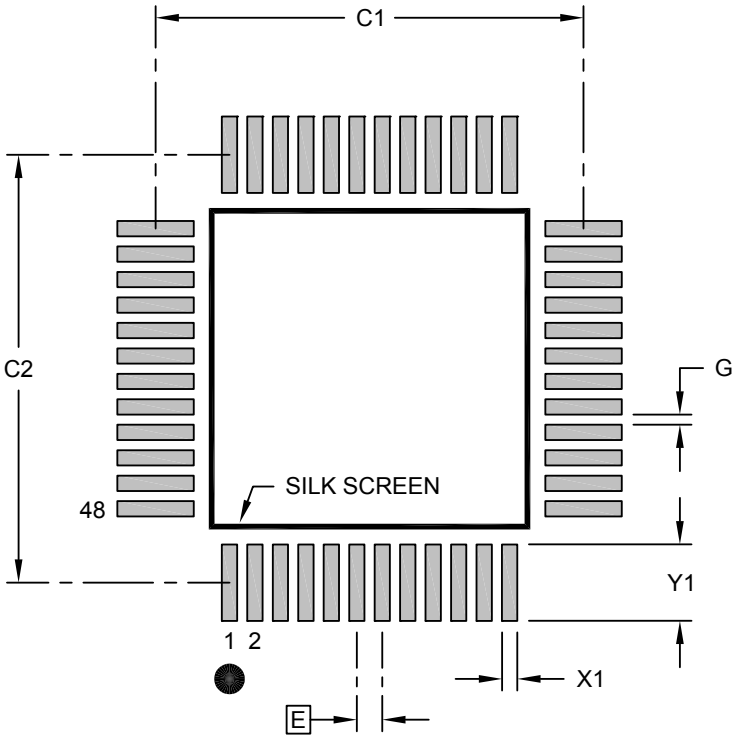
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-300-PT Rev D Sheet 2 of 2

48-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Contact Pad Spacing	C1		8.40	
Contact Pad Spacing	C2		8.40	
Contact Pad Width (X48)	X1			0.30
Contact Pad Length (X48)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

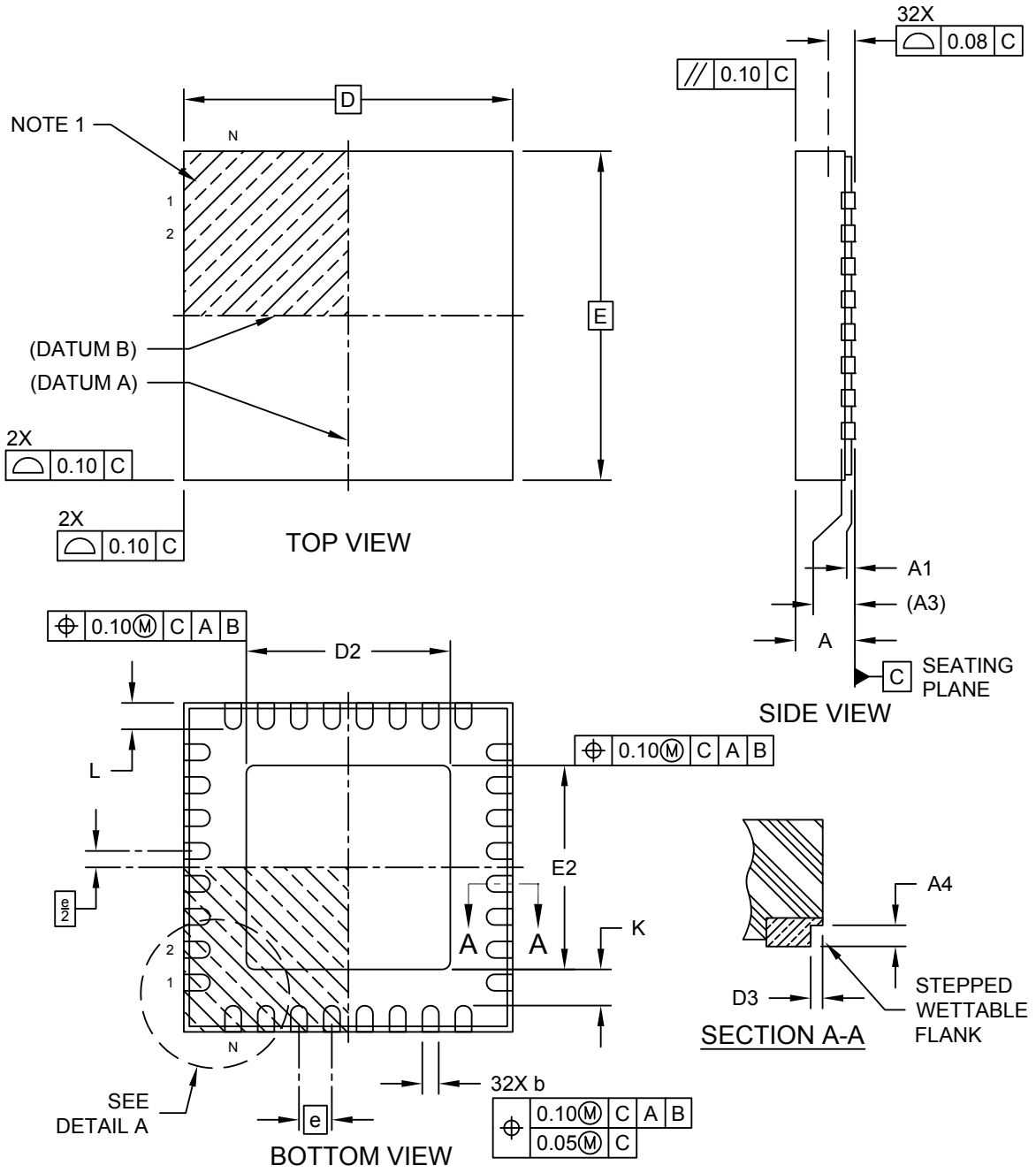
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2300-PT Rev D

39.2.5. 32-Pin VQFN Wettable Flanks

32-Lead Ultra Thin Plastic Quad Flat, No Lead Package (QZB) - 5x5x0.9 mm Body [VQFN] With 3.1 mm Exposed Pad and Stepped Wettable Flanks

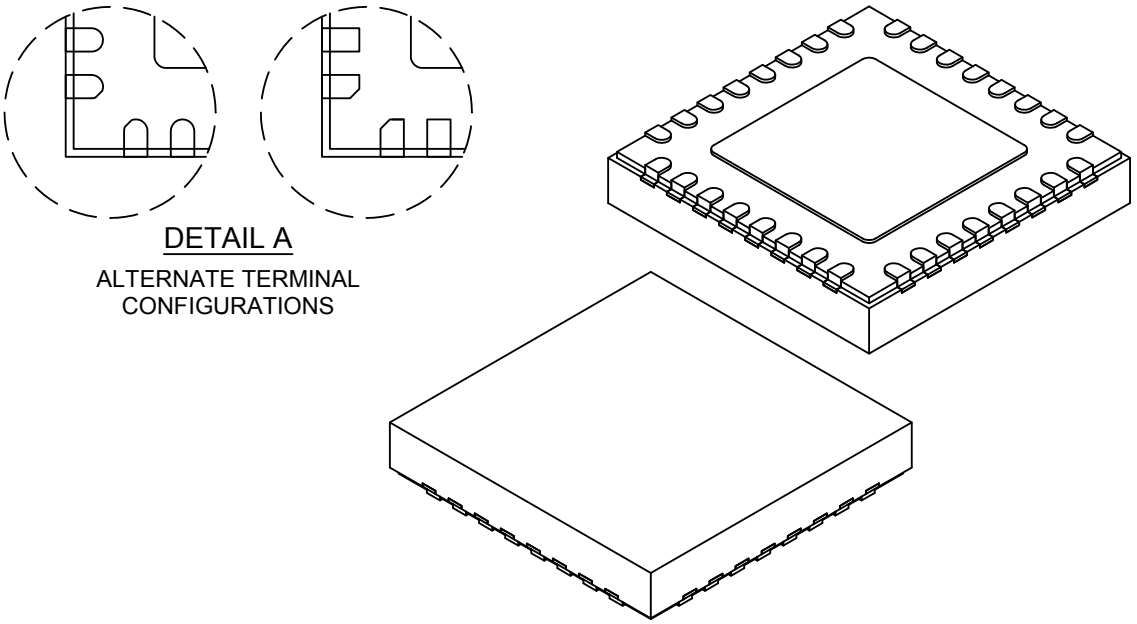
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-21511 Rev A Sheet 1 of 2

32-Lead Ultra Thin Plastic Quad Flat, No Lead Package (QZB) - 5x5x0.9 mm Body [VQFN] With 3.1 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		32	
Pitch	e		0.50 BSC	
Overall Height	A	0.80	0.85	0.90
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		5.00 BSC	
Exposed Pad Length	D2	3.00	3.10	3.20
Overall Width	E		5.00 BSC	
Exposed Pad Width	E2	3.00	3.10	3.20
Terminal Width	b	0.18	0.25	0.30
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K	0.20	-	-
Wettable Flank Step Length	D3	-	-	0.085
Wettable Flank Step Height	A4	0.10	-	0.19

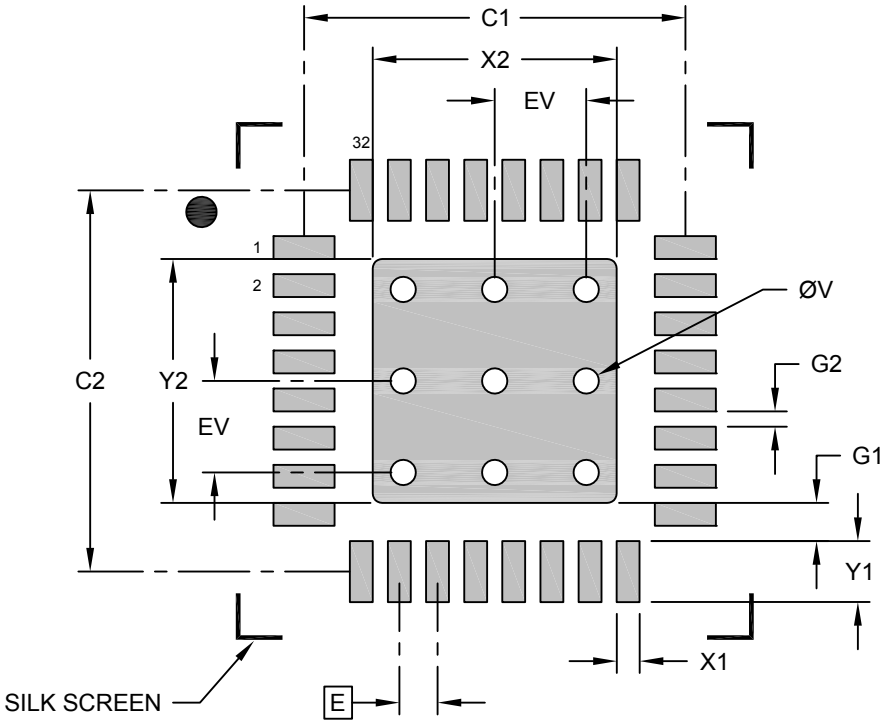
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21511 Rev A Sheet 1 of 2

32-Lead Ultra Thin Plastic Quad Flat, No Lead Package (QZB) - 5x5x0.9 mm Body [VQFN] With 3.1 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Center Pad Width	X2			3.20
Center Pad Length	Y2			3.20
Contact Pad Spacing	C1		5.00	
Contact Pad Spacing	C2		5.00	
Contact Pad Width (X32)	X1			0.30
Contact Pad Length (X32)	Y1			0.80
Contact Pad to Center Pad (X32)	G1	0.20		
Contact Pad to Contactr Pad (X28)	G2	0.20		
Thermal Via Diameter	V		0.33	
Thermal Via Pitch	EV		1.20	

Notes:

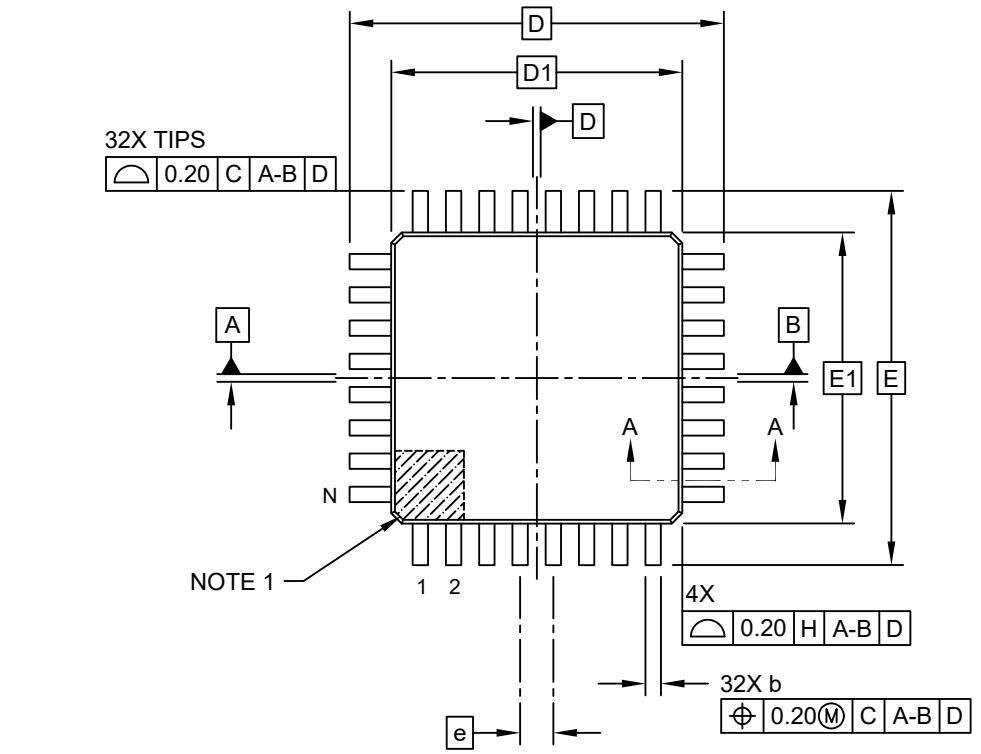
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-23511 Rev A

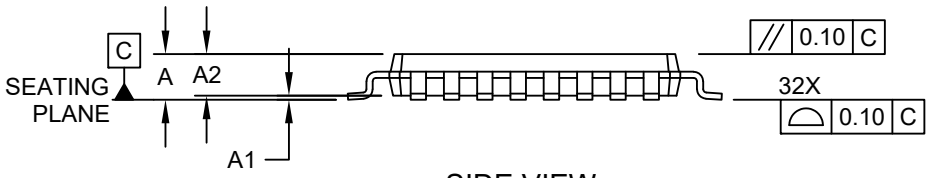
39.2.6. 32-Pin TQFP

32-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]
2.00 mm Footprint; Also Atmel Legacy Global Package Code AUT

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

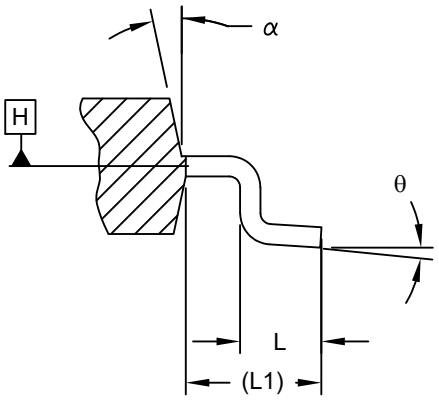


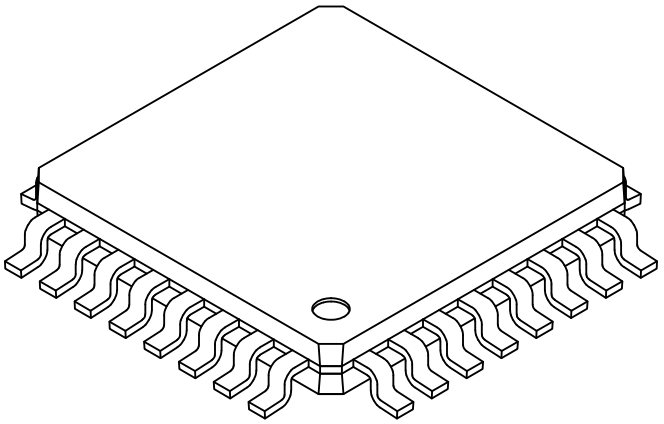
SIDE VIEW

Microchip Technology Drawing C04-074-PT Rev D Sheet 1 of 2

32-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]
2.00 mm Footprint; Also Atmel Legacy Global Package Code AUT

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging





SECTION A-A

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		32	
Lead Pitch	e		0.80 BSC	
Overall Height	A	-	-	1.20
Standoff	A1	0.05	-	0.15
Molded Package Thickness	A2	0.95	1.00	1.05
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	θ	0°	-	7°
Overall Width	E		9.00 BSC	
Overall Length	D		9.00 BSC	
Molded Package Width	E1		7.00 BSC	
Molded Package Length	D1		7.00 BSC	
Lead Width	b	0.30	0.37	0.45
Mold Draft Angle Top	α	11°	-	13°

Notes:

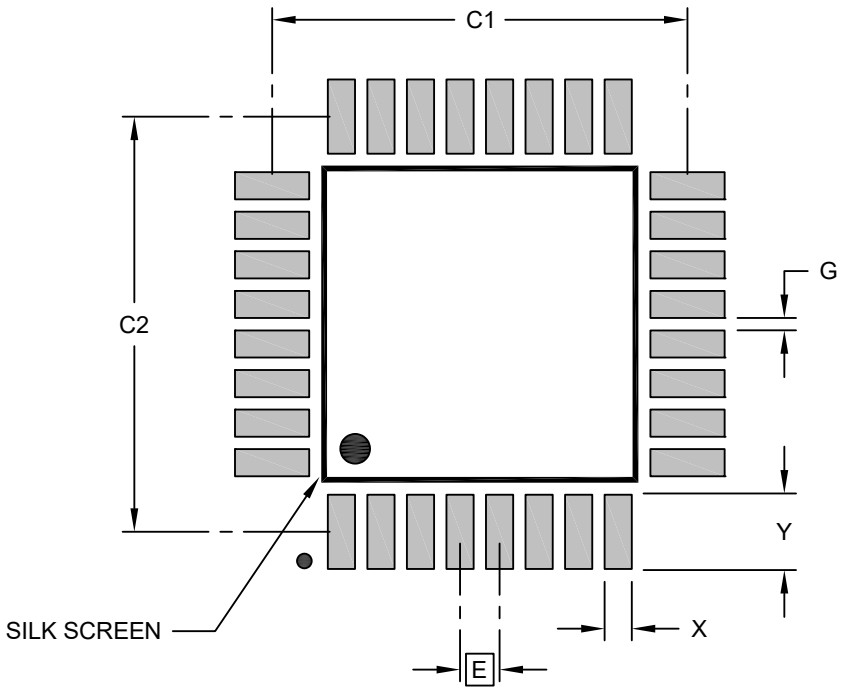
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-074-PT Rev D Sheet 2 of 2

32-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]
2.00 mm Footprint; Also Atmel Legacy Global Package Code AUT

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.80 BSC	
Contact Pad Spacing	C1		8.40	
Contact Pad Spacing	C2		8.40	
Contact Pad Width (X32)	X			0.55
Contact Pad Length (X32)	Y			1.55
Contact Pad to Contact Pad (X28)	G	0.25		

Notes:

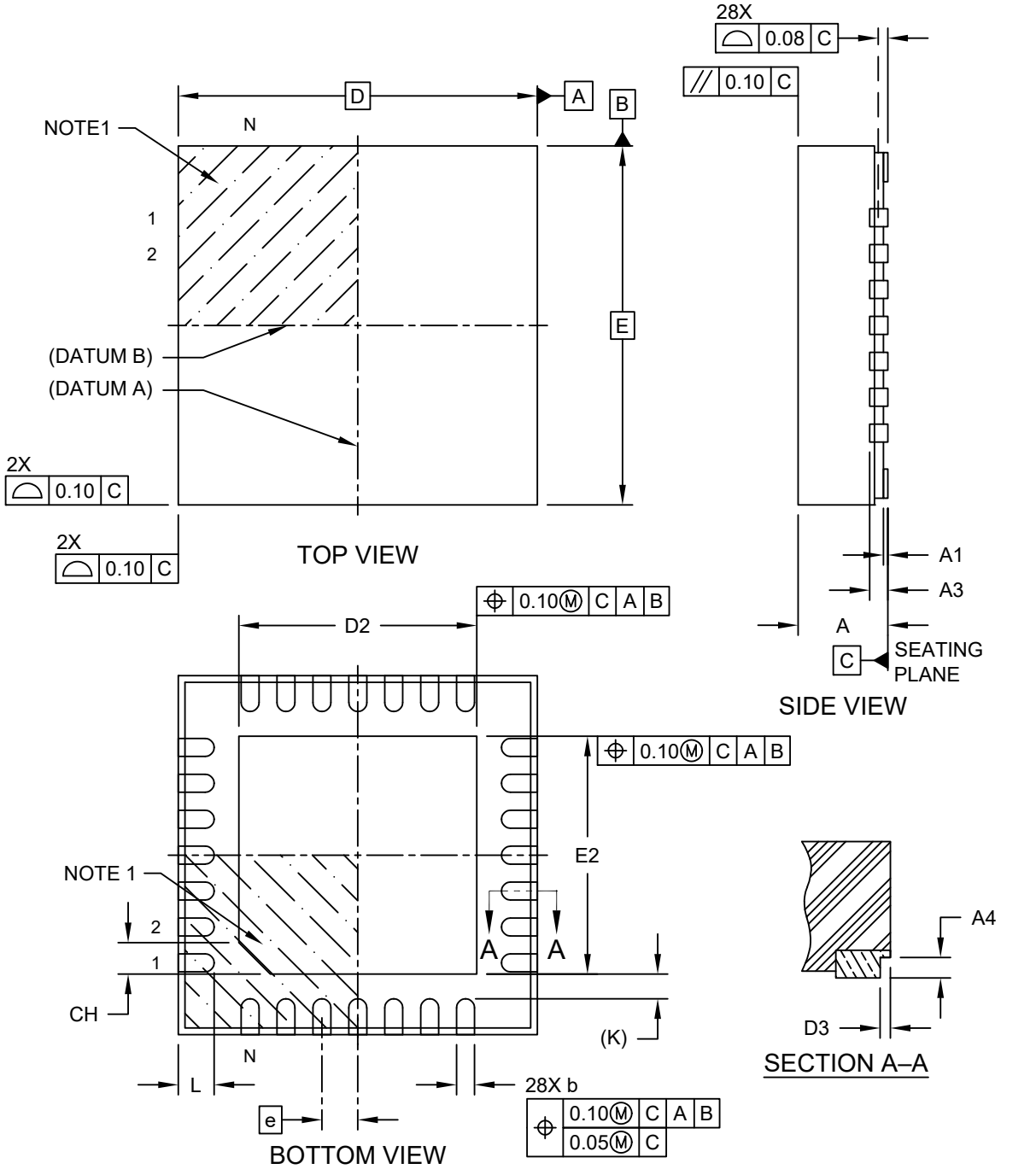
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2074-PT Rev D

39.2.7. 28-Pin VQFN Wettable Flanks

28-Lead Very Thin Plastic Quad Flat, No Lead Package (3LW) – 4x4x1.0 mm Body [VQFN] With 2.65 mm Exposed Pad and Stepped Wettable Flanks

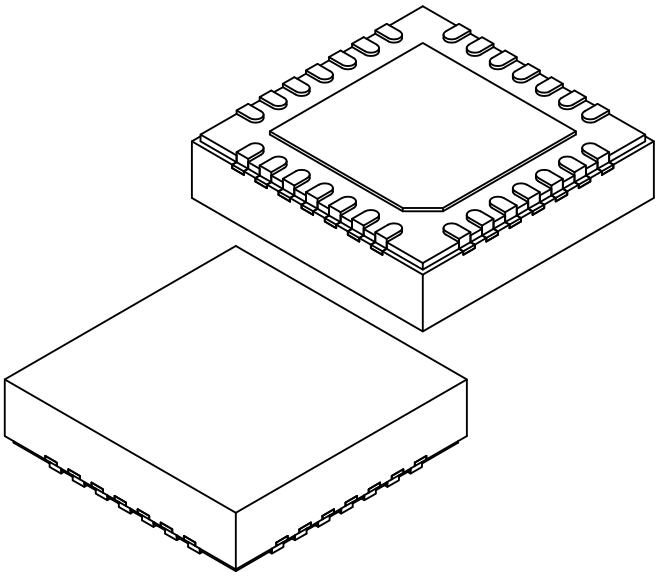
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-565-3LW Rev B Sheet 1 of 2

28-Lead Very Thin Plastic Quad Flat, No Lead Package (3LW) – 4x4x1.0 mm Body [VQFN] With 2.65 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		28	
Pitch	e		0.40 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		4.00 BSC	
Exposed Pad Length	D2	2.55	2.65	2.75
Overall Width	E		4.00 BSC	
Exposed Pad Width	E2	2.55	2.65	2.75
Exposed Pad Index Chamfer	CH		0.35 REF	
Terminal Width	b	0.15	0.20	0.25
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K		0.275 REF	
Wettable Flank Step Cut Length	D3	–	–	0.085
Wettable Flank Step Cut Height	A4	0.10	–	0.19

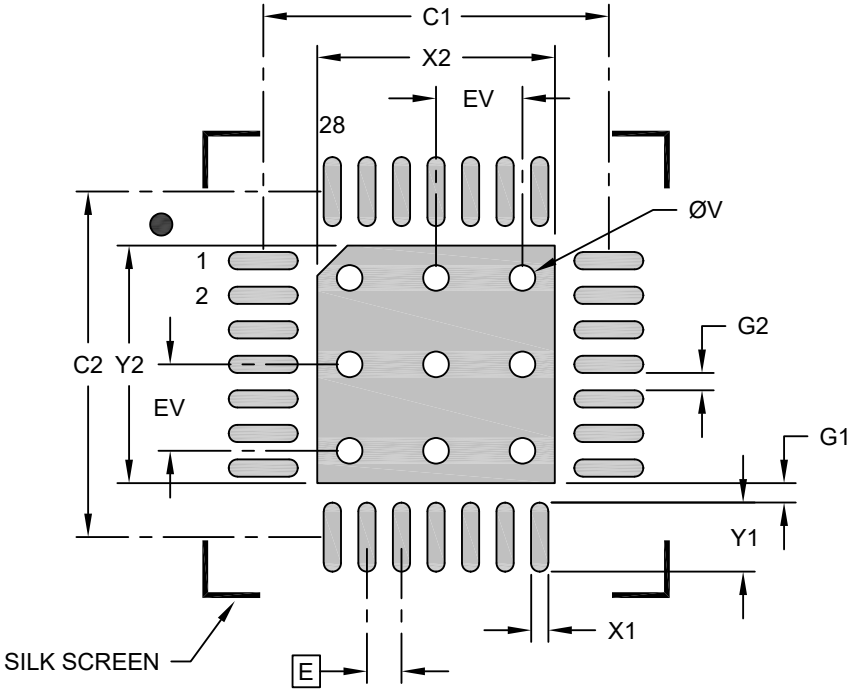
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-565-3LW Rev B Sheet 2 of 2

28-Lead Very Thin Plastic Quad Flat, No Lead Package (3LW) – 4x4x1.0 mm Body [VQFN] With 2.65 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.40 BSC	
Center Pad Width	X2			2.75
Center Pad Length	Y2			2.75
Contact Pad Spacing	C1		4.00	
Contact Pad Spacing	C2		4.00	
Contact Pad Width (X28)	X1			0.20
Contact Pad Length (X28)	Y1			0.80
Contact Pad to Center Pad (X28)	G1	0.23		
Contact Pad to Contact Pad (X24)	G2	0.20		
Thermal Via Diameter	V		0.30	
Thermal Via Pitch	EV		1.00	

Notes:

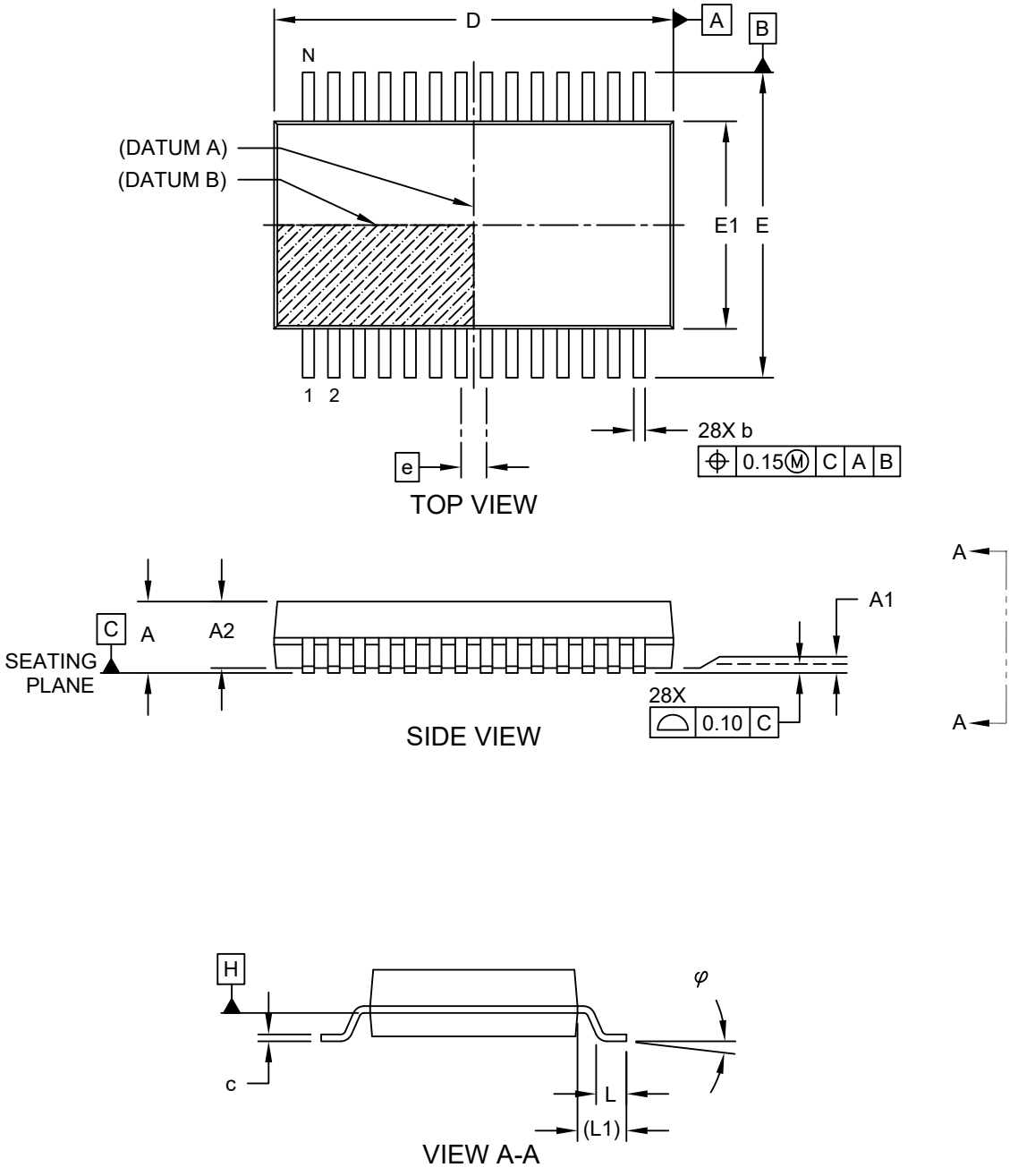
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2565-3LW Rev B

39.2.8. 28-Pin SSOP

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

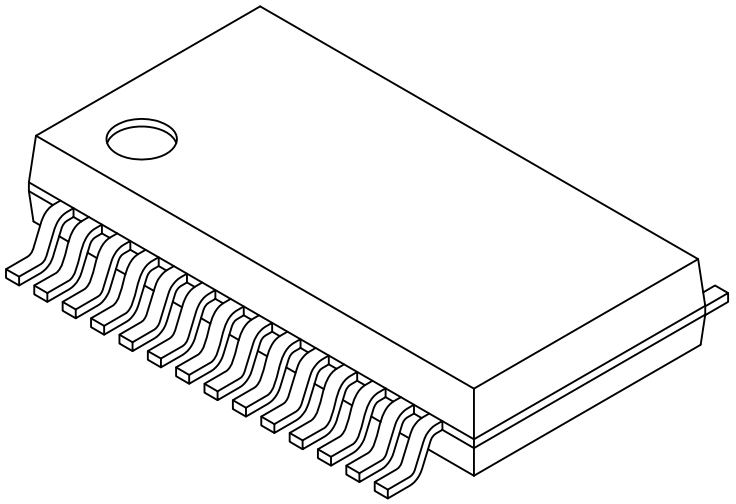
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-073 Rev C Sheet 1 of 2

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	-	-	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	-	-
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1		1.25 REF	
Lead Thickness	c	0.09	-	0.25
Foot Angle	φ	0°	4°	8°
Lead Width	b	0.22	-	0.38

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M

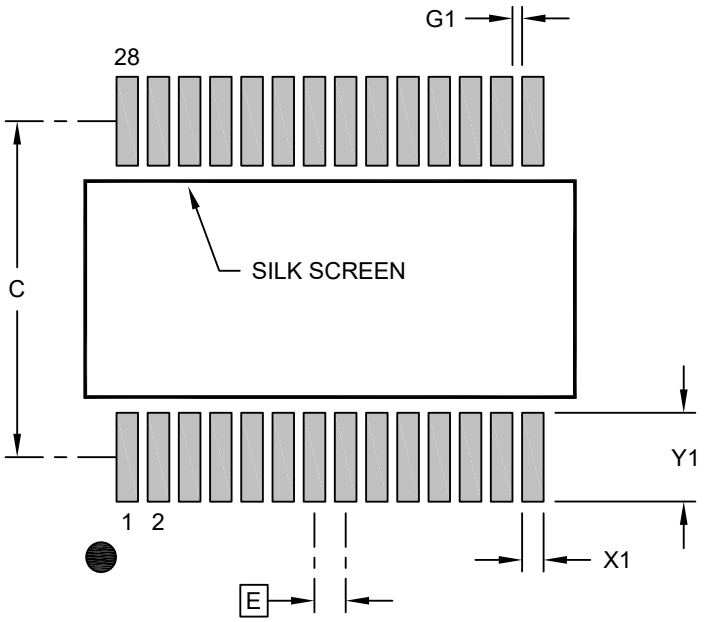
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073 Rev C Sheet 2 of 2

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Contact Pad Spacing	C		7.00	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.85
Contact Pad to Center Pad (X26)	G1	0.20		

Notes:

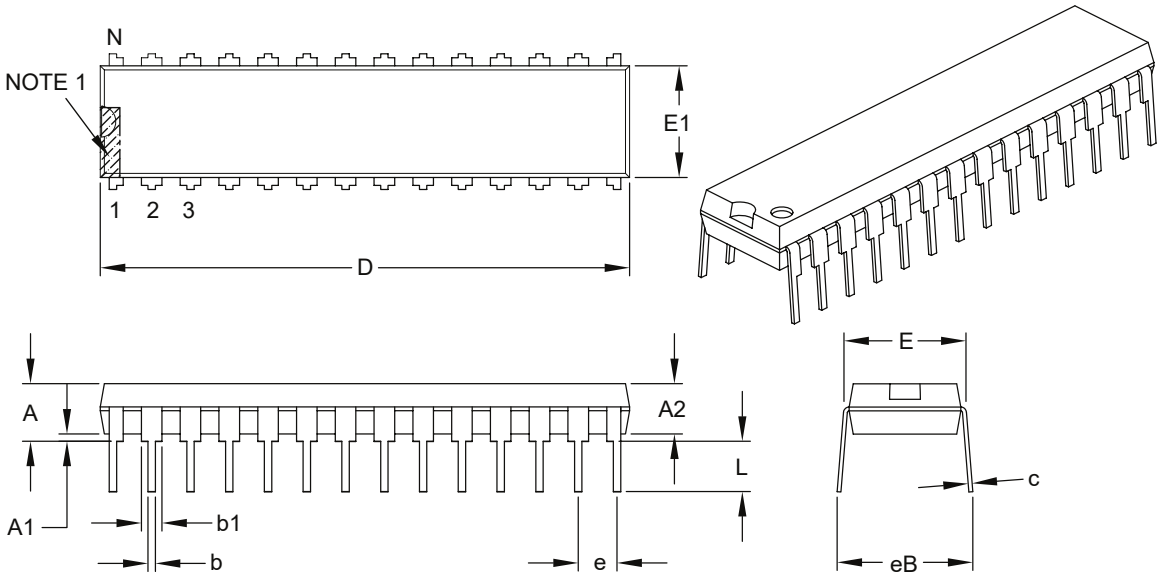
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2073 Rev B

39.2.9. 28-Pin SPDIP

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

39.3. Soldering Profile

The following table provides the recommended soldering profile from J-STD-20.

Table 39-1. Recommended 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 From 25°C to Peak Temperature	8 minutes max.

A maximum of three reflow passes is allowed per component.

40. Schematic Checklist

40.1. Introduction

This chapter describes a common checklist that must be used when starting and reviewing the schematics for a PIC32CM6408PL10 design. It illustrates the recommended power supply connections, how to connect external analog references, programmer, debugger, oscillator, and crystal.

Note: Refer to the Basic 32-Bit MCU Design and Troubleshooting Checklist document on the Microchip website, www.microchip.com.

40.2. Operation in Noisy Environment

If the device is operating in an environment with significant electromagnetic noise, it must be protected from this noise to ensure reliable operation. In addition to following best practice EMC design guidelines, the recommendations listed in the schematic checklist sections must be followed. In particular, placing decoupling capacitors very close to the power pins, an RC-filter on the $\overline{\text{RESET}}$ pin, and a pull-up resistor on the SWCLK pin are critical for reliable operations. A weak pull-up resistor is also recommended on the SWDIO pin. It is also important to eliminate or attenuate noise to prevent it from reaching supply pins, I/O pins, and crystals.

Refer to the EMI, EMC, EFT and ESD Circuit Design Consideration for 32-bit Microcontrollers guide on the Microchip website, www.microchip.com.

40.3. Power Supply

This device supports either a single power supply or dual power supplies ranging from 1.8V to 5.5V.

40.3.1. Power Supply Connections

The basics and details of power supply design are beyond the scope of these guidelines.

A decoupling capacitor C_1 must be placed close to the microcontroller for each supply pin pair (V_{DD} or other power supply pin and its corresponding GND pin). If the decoupling capacitor is placed too far from the microcontroller, a high-current loop can form, resulting in increased noise and radiated emissions.

Each supply pin pair (power input pin and ground pin) must have a separate decoupling capacitor.

It is recommended to place the decoupling capacitor on the same side of the PCB as the microcontroller. If space does not allow this, the decoupling capacitor may be placed on the other side through a via, but be sure to keep the distance to the supply pin as short as possible.

If the board is experiencing high-frequency noise (upwards of tens of MHz), add a second ceramic-type capacitor, C2, in parallel with the decoupling capacitor, C_1. Place this second capacitor right next to the primary decoupling capacitor.

In the board layout, starting from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins, ensuring that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

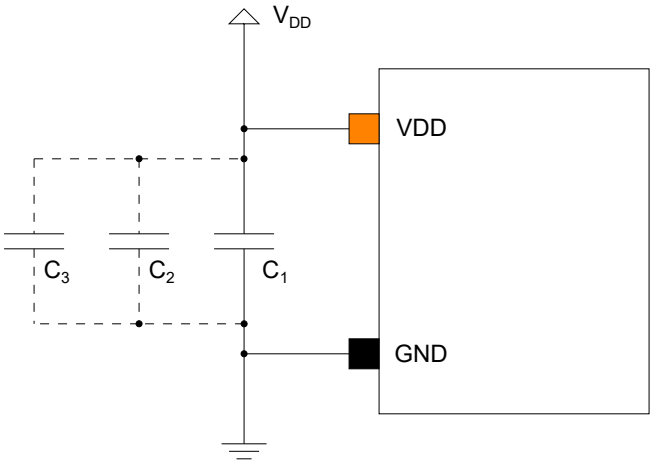
All values used in the examples are typical values. The actual design may require different values.

40.3.1.1. Power Supply

For higher pin-count package types, there are several V_{DD} and corresponding GND pins. All the V_{DD} pins in the microcontroller are internally connected. The same voltage must be applied to each of the V_{DD} pins.

The figure below shows the recommended connection of the power supply to the device's V_{DD} pin(s).

Figure 40-1. Recommended V_{DD} Connection Circuit Schematic



Typical values (recommended):
C_1: 100 nF (primary decoupling capacitor)
C_2: 1-10 nF (HF decoupling capacitor)
C_3(*): 1 µF (decoupling capacitor - optional)

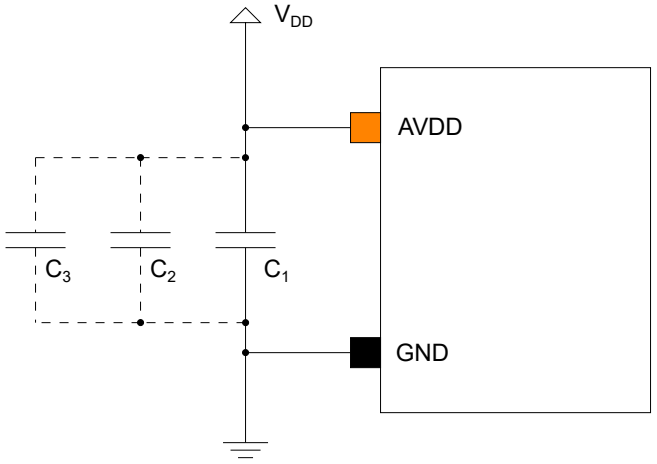
Important: For systems that frequently cycle V_{DD} or experience fast V_{DD} transients, it is recommended to add a decoupling capacitor (C_3) if the power supply slew rate exceeds the specified limits. Refer to the *Supply Voltage* section in the *Electrical Characteristics* for details about the power supply's slew rate limits.

40.3.1.2. Analog Power Supply

These devices have a separate analog supply pin, AVDD, for future compatibility. In this device, the AV_{DD} and V_{DD} power domains are internally connected.

The following figure shows the recommended method for connecting a power supply to the AVDD pin of the device.

Figure 40-2. Recommended AVDD Connection Circuit Schematic



Typical values (recommended):
C_1: 100 nF (primary decoupling capacitor)
C_2: 1-10 nF (HF decoupling capacitor)
C_3: 1 µF (decoupling capacitor - optional)

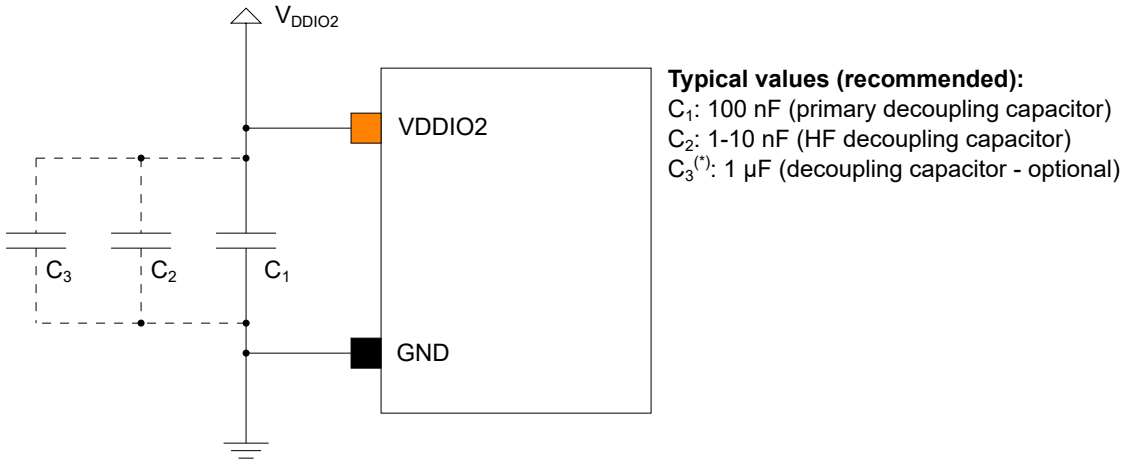
Important: For systems that frequently cycle V_{DD} or experience fast V_{DD} transients, it is recommended to add an additional decoupling capacitor (C_3) if the power supply slew rate exceeds the slew rate limits. Refer to the *Supply Voltage* section in the *Electrical Characteristics* for details about power supply slew rate limits.

40.3.1.3. Multi-Voltage I/O

This additional Multi-Voltage I/O (MVIO) power supply input pin and its corresponding grounding pin must be treated the same way as any other power supply pin pair: Connect a separate decoupling capacitor to the pin pair, and keep the trace distance from the pins as short as possible. Each supply pin and its corresponding ground pin must have a decoupling capacitor if there is more than one MVIO power supply pin.

The following figure shows the recommended method for connecting a power supply to the VDDIO2 pin(s) of the device.

Figure 40-3. Recommended VDDIO2 Connection Circuit Schematic



Important: For systems that frequently cycle V_{DDIO2} or experience fast V_{DDIO2} transients, it is recommended to add a decoupling capacitor (C_3) if the power supply slew rate exceeds the specified limits. Refer to the *Supply Voltage* section in the *Electrical Characteristics* for details about the power supply's slew rate limits.

40.4. External Analog Reference Connections

The following schematic checklist is only required if the application is using the external analog reference. For unused VREFx inputs, the following circuit is not necessary.

Figure 40-4. External Analog Reference Schematic With One Reference

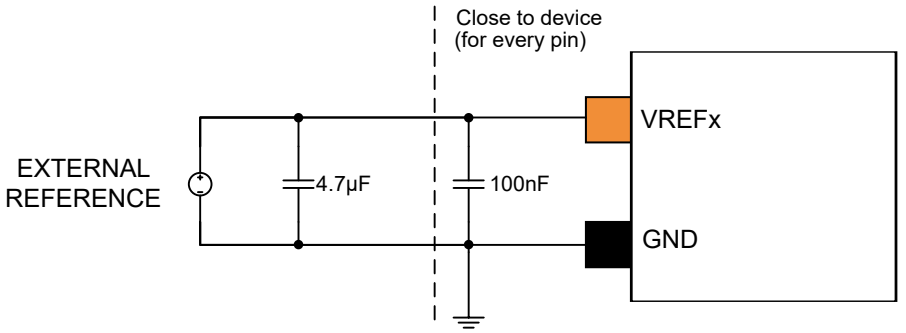


Table 40-1. External Analog Reference Connections

Signal Name	Recommended Pin Connection	Description
VREFx	2.4V to V_{DDANA} - 0.6V for ADC (2) 2.4V to V_{DDANA} - 0.6V for DAC (2) Decoupling/filtering capacitors: 100 nF(1,2,3) and 4.7 µF(1)	External reference from the VREFx pin on the analog port
GND		Ground

Notes:

1. Refer to the *Power Supply Electrical Specifications* in the *Electrical Characteristics* chapter.
2. Refer to the *Electrical Characteristics* chapter for the intended peripheral to be used with VREFx to determine the suggested operational conditions. Examples are peripherals such as the ADC, DAC, AC, and SDADC.
3. A decoupling capacitor must be placed close to the device for each supply pin pair in the signal group.

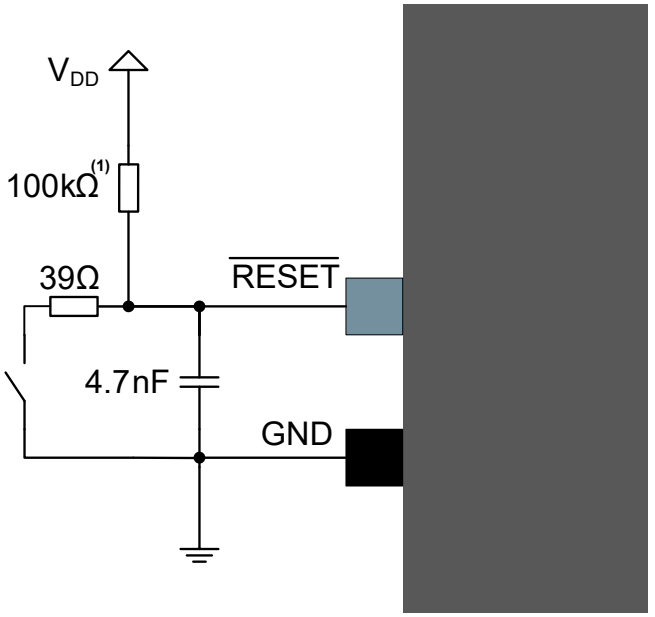
40.5. External Reset Circuit

The external Reset circuit is connected to the $\overline{\text{RESET}}$ pin when the external Reset function is used. The circuit is not necessary when the $\overline{\text{RESET}}$ pin is not driven low externally by the application circuitry.

The reset switch can be removed if a manual reset is not desired. The $\overline{\text{RESET}}$ pin itself has an internal pull-up resistor, therefore, adding an external pull-up resistor is optional.

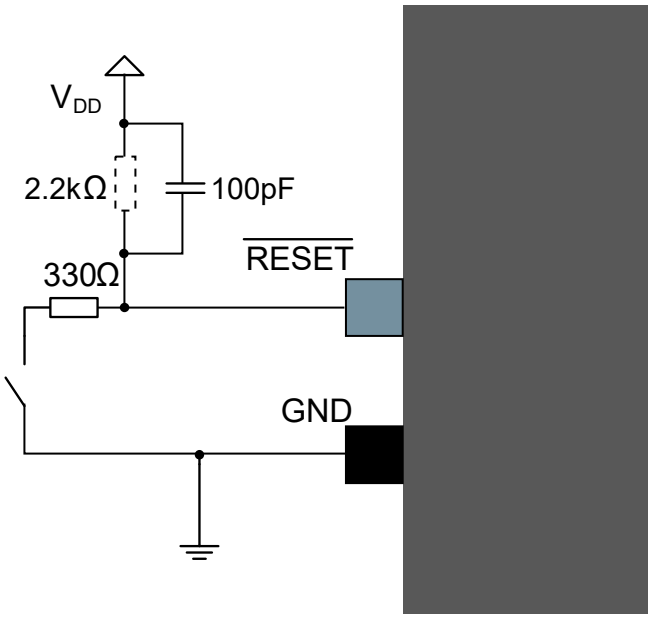
A pull-up resistor ensures that the reset does not go low and unintentionally cause a device reset. An additional resistor has been added in series with the switch to safely discharge the filtering capacitor, thereby preventing a current surge when shorting the filtering capacitor, which can cause a noise spike that may negatively affect the system.

Figure 40-5. External Reset Circuit Schematic (General Purpose)(2)



The following reset circuit is intended to improve EFT immunity but does not filter low-frequency glitches, which makes it unsuitable as an example for applications requiring debouncing on a reset button.

Figure 40-6. External Reset Circuit Schematic (EFT Immunity Enhancement) [2]



Notes:

1. The device features an internal pull-up resistor on the $\overline{\text{RESET}}$ pin; therefore, an external pull-up resistor is optional.
2. These values are provided only as a typical example. Refer to the *Power Supply Electrical Characteristics* to determine the values specified in the system parameters to meet reset timing requirements (TRST).

Note: Reset circuit recommendations to improve EFT/ESD/EMI immunity can be found in the "Basic 32-Bit MCU Design and Troubleshooting Checklist" guide on the Microchip website, www.microchip.com.

40.6. Unused or Unconnected Pins

For unused pins, the default state will result in the lowest current leakage. Therefore, there is no need to configure unused pins to reduce the power consumption. However, as a best practice, always configure unused pins as output-low.

40.7. Clocks and Crystal Oscillators

The device can operate using internal or external clock sources, or a combination of both. For example, the internal 48 MHz oscillator can be used as source for the system clock, while an external 32.768 kHz crystal can serve as the clock source for the Real-Time counter (RTC).

40.7.1. External Clock Source

Figure 40-7. External Clock Source Schematic

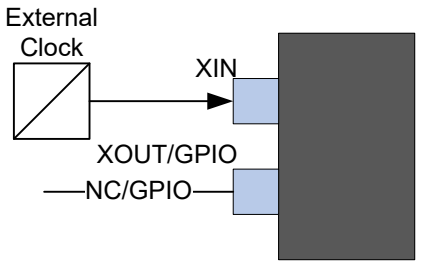
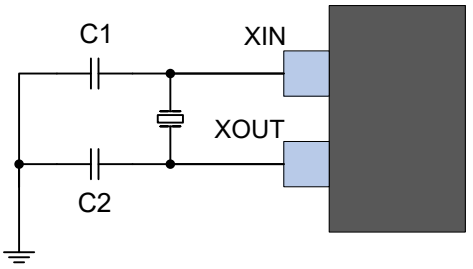


Table 40-2. External Clock Source Connections

Signal Name	Description	Recommended Pin Connection
XIN	Input for inverting oscillator pin: External Clock frequency range is specified in the XOSC Electrical Specifications (XOSC35 parameter)	Input for an external clock signal
XOUT/GPIO	Available for use as GPIO	NC/GPIO

40.7.2. Crystal Oscillator

Figure 40-8. Crystal Oscillator Schematic



The crystal must be located as close to the device as possible. Long signal lines may pose too high of a load for the crystal to operate properly and may cause crosstalk to other parts of the system.

Note: Crystal selection must be done using both the crystal data sheet and the crystal oscillator parameters given in the *XOSC Electrical Specifications* from the *Electrical Characteristics* chapter.

Table 40-3. Crystal Oscillator Checklist

Signal Name	Description	Recommended Pin Connection
XIN	External Crystal. Crystal frequency range is specified in the *XOSC Electrical Specifications* (XOSC_1 parameter).	C1 Load Capacitor (1,2)
XOUT		C2 Load Capacitor (1,2)

Notes:

1. The capacitors must be placed close to the device.
2. The calculation for crystal load capacitors is provided in the *XOSC Electrical Specifications*.

40.7.3. External Slow Clock Source

Figure 40-9. External Slow Clock Source Schematics

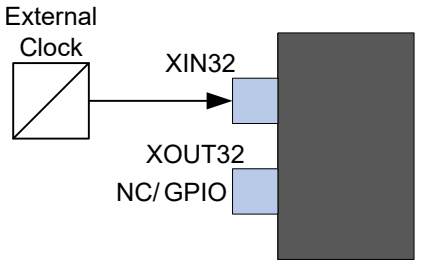


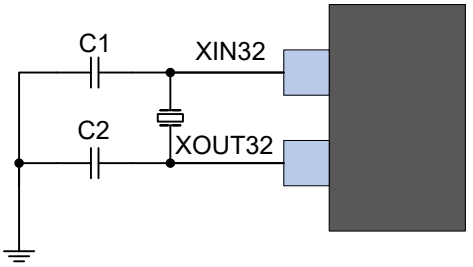
Table 40-4. External Slow Clock Source Connections

Signal Name	Description	Recommended Pin Connection
XIN32	Input for the inverting oscillator pin. External Clock frequency range is specified in the XOSC32K Electrical Specifications (XOSC32K_19 parameter).	XIN32 is used as an input for an external clock signal
XOUT32/GPIO	NC/GPIO	Available for use as a GPIO

40.7.4. 32.768 kHz Crystal Oscillator

The PIC32CM6408PL10 32.768 kHz crystal oscillator is optimized for very low power consumption; therefore, close attention must be paid when selecting crystals.

Figure 40-10. 32.768 kHz Crystal Oscillator Schematics



Note: 32.768 kHz crystal selection must be done using both the crystal data sheet and the crystal oscillator parameters given in the *XOSC32K Electrical Specifications* from the *Electrical Characteristics* chapter.

Table 40-5. 32.768 kHz Crystal Oscillator Checklist

Signal Name	Description	Recommended Pin Connection
XIN32	External Crystal. Crystal frequency range is specified in the *XOSC32K Electrical Characteristics* (XOSC32K_1 parameter).	C1 Load Capacitor (1,2)
XOUT32		C2 Load Capacitor (1,2)

Notes:

1. The capacitors must be placed close to the device.
2. The calculation for crystal load capacitors is provided in the *XOSC32K Electrical Characteristics*.

40.8. Programming and Debug Ports

For initial evaluation, the Curiosity evaluation board can be used. It supports programming and debugging through the onboard embedded debugger; therefore, an external programmer or debugger is not required.

For programming and debugging the PIC32CM6408PL10, the device must be connected using the Serial Wire Debug (SWD) interface. The SWD interface is supported by several Microchip and third-party programmers and debuggers.

Refer to the relevant Microchip user guides for details on debugging and programming connections and options. For connecting to any third-party programming or debugging tool, refer to the specific programmer or debugger user guide.

By default, the SWCLK pin is internally pulled up after reset.

An external pull-up resistor on the SWCLK pin is critical for reliable operation. A weak pull-up (resistor value in the range of tens to 100 kΩ) on the SWDIO pin is also considered best practice.

Figure 40-11. SWCLK Circuit Connections

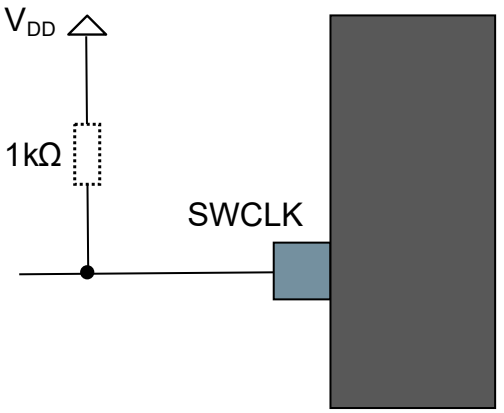


Table 40-6. SWCLK Circuit Connections

Pin Name	Description	Recommended Pin Connection
SWCLK	Serial wire clock pin	Pull-up resistor: 1 kΩ

40.8.1. Cortex Debug Connector (10-pin)

For debuggers and/or programmers that support the Cortex Debug Connector (10-pin) interface, the signals must be connected as shown below.

Figure 40-12. Cortex Debug Connector (10-pin)

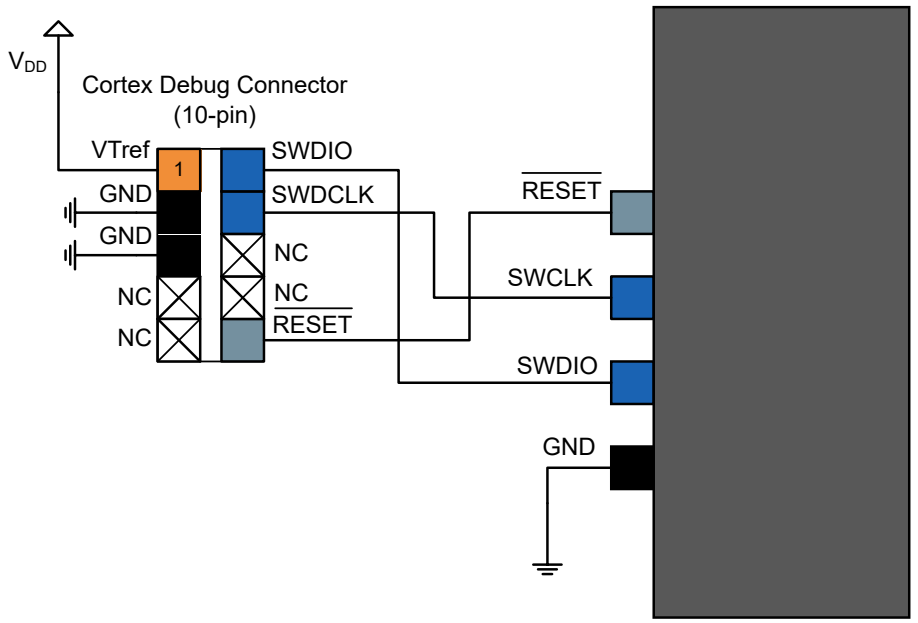


Table 40-7. Cortex Debug Connector (10-pin)

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, must be connected to the device V_{DD}
GND	Ground

Note: JTAG and Serial Wire Interface recommendations can be found in the “Basic 32-Bit MCU Design and Troubleshooting Checklist” guide on the Microchip website, www.microchip.com.

40.8.2. 20-pin IDC JTAG Connector

For debuggers and/or programmers that support the 20-pin IDC JTAG Connector, such as the SAM-ICE, the signals must be connected as shown below.

Figure 40-13. 20-pin IDC JTAG Connector

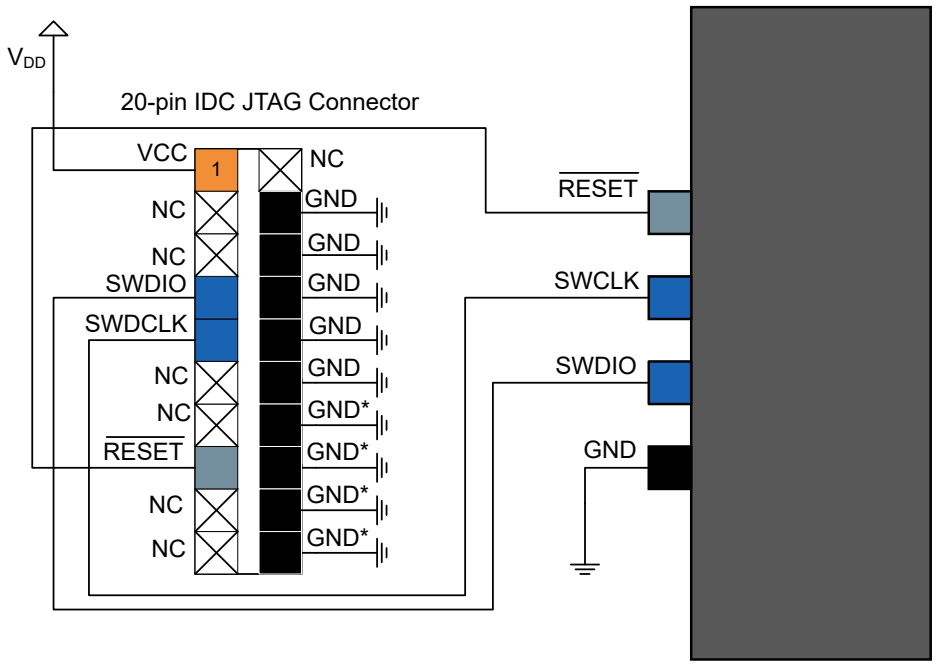


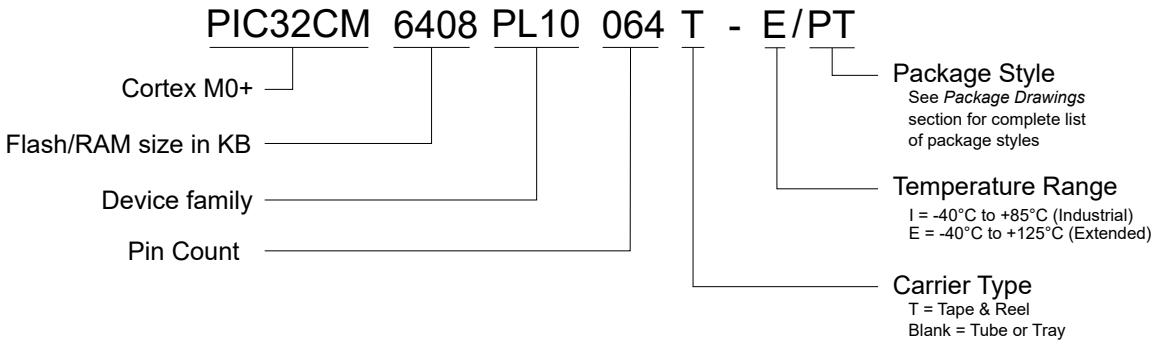
Table 40-8. 20-pin IDC JTAG Connector

Header Signal Name	Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
$\overline{\text{RESET}}$	Target device reset pin, active low
VCC	Target voltage sense, must be connected to the device V_{DD}
GND	Ground
GND*	These pins are reserved for firmware extension purposes. They can be left unconnected or connected to GND in a typical debug environment. They are not essential for SWD in general.

Note: JTAG and Serial Wire Interface recommendations can be found in the *Basic 32-Bit MCU Design and Troubleshooting Checklist* guide on the Microchip website, www.microchip.com.

41. Ordering Information

Figure 41-1. PIC32CM6408PL10 Family Ordering Information



Note: Automotive-grade ordering codes (VAO suffix) are set up upon request for devices that are labelled “Recommended for Automotive Designs” on the product page. To request VAO ordering codes not listed on the respective product page, contact your local Microchip sales representative.

CPN	Flash in KB	Pin Count	Temp. Range	Carrier Type	Package Type(1)	Package Style	Package Code
PIC32CM6408PL10028-E/3LW	64	28	E	Tube	VQFN WF	3LW	3LW
PIC32CM6408PL10028-E/SP	64	28	E	Tube	SPDIP	SP	M3X
PIC32CM6408PL10028-E/SS	64	28	E	Tube	SSOP	SS	N2X
PIC32CM6408PL10028-I/3LW	64	28	I	Tube	VQFN WF	3LW	3LW
PIC32CM6408PL10028-I/SP	64	28	I	Tube	SPDIP	SP	M3X
PIC32CM6408PL10028-I/SS	64	28	I	Tube	SSOP	SS	N2X
PIC32CM6408PL10028T-E/3LW	64	28	E	Tape and Reel	VQFN WF	3LW	3LW
PIC32CM6408PL10028T-E/SS	64	28	E	Tape and Reel	SSOP	SS	N2X
PIC32CM6408PL10028T-I/3LW	64	28	I	Tape and Reel	VQFN WF	3LW	3LW
PIC32CM6408PL10028T-I/SS	64	28	I	Tape and Reel	SSOP	SS	N2X
PIC32CM6408PL10032-E/PT	64	32	E	Tray	TQFP	PT	T5X
PIC32CM6408PL10032-E/QZB	64	32	E	Tray	VQFN WF	QZB	QZB
PIC32CM6408PL10032-I/PT	64	32	I	Tray	TQFP	PT	T5X
PIC32CM6408PL10032-I/QZB	64	32	I	Tray	VQFN WF	QZB	QZB
PIC32CM6408PL10032T-E/PT	64	32	E	Tape and Reel	TQFP	PT	T5X
PIC32CM6408PL10032T-E/QZB	64	32	E	Tape and Reel	VQFN WF	QZB	QZB
PIC32CM6408PL10032T-I/PT	64	32	I	Tape and Reel	TQFP	PT	T5X
PIC32CM6408PL10032T-I/QZB	64	32	I	Tape and Reel	VQFN WF	QZB	QZB
PIC32CM6408PL10048-E/6MX	64	48	E	Tray	VQFN WF	6MX	6MX
PIC32CM6408PL10048-E/PT	64	48	E	Tray	TQFP	PT	Y8X
PIC32CM6408PL10048-I/6MX	64	48	I	Tray	VQFN WF	6MX	6MX
PIC32CM6408PL10048-I/PT	64	48	I	Tray	TQFP	PT	Y8X
PIC32CM6408PL10048T-E/6MX	64	48	E	Tape and Reel	VQFN WF	6MX	6MX
PIC32CM6408PL10048T-E/PT	64	48	E	Tape and Reel	TQFP	PT	Y8X
PIC32CM6408PL10048T-I/6MX	64	48	I	Tape and Reel	VQFN WF	6MX	6MX
PIC32CM6408PL10048T-I/PT	64	48	I	Tape and Reel	TQFP	PT	Y8X
PIC32CM6408PL10064-E/5LX	64	64	E	Tray	VQFN WF	5LX	5LX
PIC32CM6408PL10064-E/PT	64	64	E	Tray	TQFP	PT	V2X

Ordering Information (continued)

CPN	Flash in KB	Pin Count	Temp. Range	Carrier Type	Package Type(1)	Package Style	Package Code
PIC32CM6408PL10064-I/5LX	64	64	I	Tray	VQFN WF	5LX	5LX
PIC32CM6408PL10064-I/PT	64	64	I	Tray	TQFP	PT	V2X
PIC32CM6408PL10064T-E/5LX	64	64	E	Tape and Reel	VQFN WF	5LX	5LX
PIC32CM6408PL10064T-E/PT	64	64	E	Tape and Reel	TQFP	PT	V2X
PIC32CM6408PL10064T-I/5LX	64	64	I	Tape and Reel	VQFN WF	5LX	5LX
PIC32CM6408PL10064T-I/PT	64	64	I	Tape and Reel	TQFP	PT	V2X

Note:

1. WF = Wettable Flanks

42. Data Sheet Revision History

Note: The data sheet revision is independent of the die revision and the device variant (last letter of the ordering number).

42.1. Rev.A - 12/2025

Initial release.

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 979-8-3371-2486-5

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Product Page Links

PIC32CM6408PL10028, PIC32CM6408PL10032, PIC32CM6408PL10048, PIC32CM6408PL10064