ATtiny3216/3217

tinyAVR® 1-series

Introduction

The ATtiny3216/3217 are members of the tinyAVR® 1-series of microcontrollers, using the AVR® processor with hardware multiplier, running at up to 20 MHz, with 32 KB Flash, 2 KB of SRAM, and 256 bytes of EEPROM in a 20- and 24-pin package. The tinyAVR® 1-series uses the latest technologies with a flexible, low-power architecture, including Event System, accurate analog features, and Core Independent Peripherals (CIPs). Capacitive touch interfaces with Driven Shield+ and Boost Mode technologies are supported with the integrated Peripheral Touch Controller (PTC).

Attention: Automotive products are documented in separate data sheets.

Features

- CPU
  - AVR® CPU
  - Running at up to 20 MHz
  - Single-cycle I/O access
  - Two-level interrupt controller
  - Two-cycle hardware multiplier
- Memories
  - 32 KB In-system self-programmable Flash memory
  - 256 bytes EEPROM
  - 2 KB SRAM
  - Write/erase endurance:
    - Flash 10,000 cycles
    - EEPROM 100,000 cycles
  - Data retention:
    - 40 years at 55°C
- System
  - Power-on Reset (POR)
  - Brown-out Detector (BOD)
  - Clock options:
    - 16/20 MHz low-power internal RC oscillator
    - 32.768 kHz Ultra Low-Power (ULP) internal RC oscillator
    - 32.768 kHz external crystal oscillator
    - External clock input
  - Single-pin Unified Program and Debug Interface (UPDI)
  - Three sleep modes:

    - Idle with all peripherals running for immediate wake-up
    - Standby
        - Configurable operation of selected peripherals
    - Power-Down with full data retention
- Peripherals
    - One 16-bit Timer/Counter type A (TCA) with a dedicated period register and three compare channels
    - Two 16-bit Timer/Counter type B (TCB) with input capture
    - One 12-bit Timer/Counter type D (TCD) optimized for control applications
    - One 16-bit Real-Time Counter (RTC) running from an external crystal, external clock, or internal RC oscillator
    - Watchdog Timer (WDT) with Window mode, with a separate on-chip oscillator
    - One USART with fractional baud rate generator, auto-baud, and start-of-frame detection
    - One master/slave Serial Peripheral Interface (SPI)
    - One Two-Wire Interface (TWI) with dual address match
        - Philips I^2C compatible
        - Standard mode (Sm, 100 kHz)
        - Fast mode (Fm, 400 kHz)
        - Fast mode plus (Fm+, 1 MHz)
    - Three Analog Comparators (AC) with a low propagation delay
    - Two 10-bit 115 ksps Analog-to-Digital Converters (ADCs)
    - Three 8-bit Digital-to-Analog Converters (DACs) with one external channel
    - Multiple voltage references (V_{REF}):
        - 0.55V
        - 1.1V
        - 1.5V
        - 2.5V
        - 4.3V
    - Event System (EVSYS) for CPU independent and predictable inter-peripheral signaling
    - Configurable Custom Logic (CCL) with two programmable look-up tables
    - Automated CRC memory scan
    - Peripheral Touch Controller (PTC)
        - Capacitive touch buttons, sliders, wheels and 2D surfaces
        - Wake-up on touch
        - Driven shield for improved moisture and noise handling performance
        - Up to 14 self-capacitance channels
        - Up to 49 mutual capacitance channels
    - External interrupt on all general purpose pins
- I/O and Packages:
    - Up to 22 programmable I/O lines
    - 20-pin SOIC300
    - 24-pin VQFN 4x4 mm
- Temperature Ranges:
    - -40°C to 105°C
    - -40°C to 125°C
- Speed Grades:
    - 0-5 MHz @ 1.8V – 5.5V
    - 0-10 MHz @ 2.7V – 5.5V
    - 0-20 MHz @ 4.5V – 5.5V

Table of Contents

1. Silicon Errata and Data Sheet Clarification Document

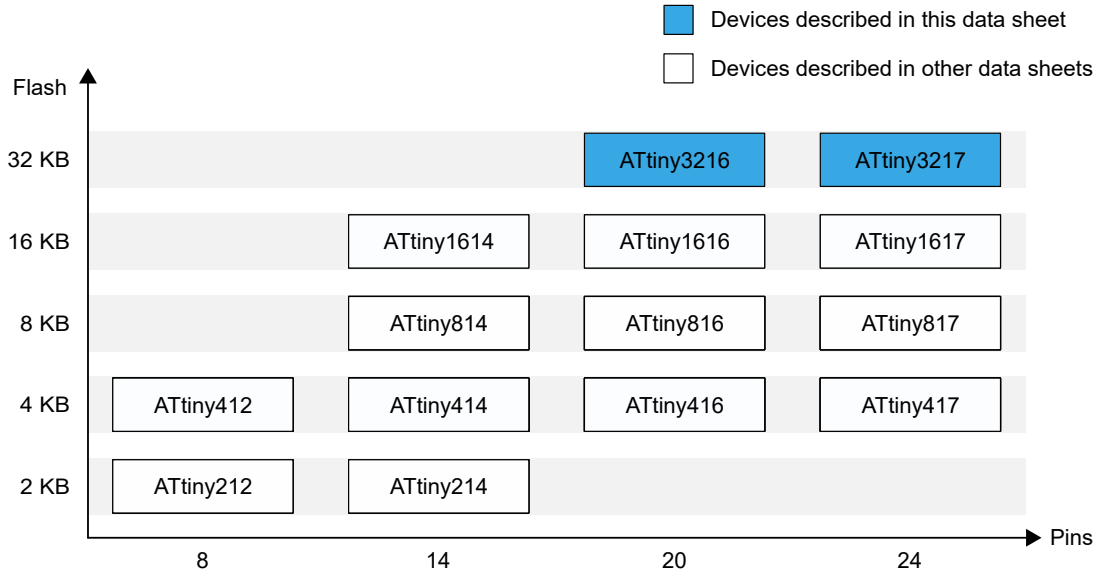
Microchip aims to provide its customers with the best documentation possible to ensure a successful use of Microchip products. Between data sheet updates, a Silicon errata and data sheet clarification document will contain the most recent information for the data sheet. The *ATtiny3216/3217 Silicon Errata and Data Sheet Clarification* (www.microchip.com/DS80000887) is available at the device product page on https://www.microchip.com.

2. tinyAVR® 1-series Overview

The following figure shows the tinyAVR 1-series devices, laying out pin count variants and memory sizes:

- Vertical migration upwards is possible without code modification, as these devices are pin-compatible and provide the same or more features. Downward migration may require code modification due to fewer available instances of some peripherals.
- Horizontal migration to the left reduces the pin count and, therefore, the available features

Figure 2-1. tinyAVR® 1-series Overview



Devices with different Flash memory sizes typically also have different SRAM and EEPROM.

2.1 Configuration Summary

2.1.1 Peripheral Summary

Table 2-1. Peripheral Summary

	ATtiny3216	ATtiny3217
Pins	20	24
SRAM	2 KB	2 KB
Flash	32 KB	32 KB
EEPROM	256B	256B
Max. frequency (MHz)	20	20
16-bit Timer/Counter type A (TCA)	1	1
16-bit Timer/Counter type B (TCB)	2	2
12-bit Timer/Counter type D (TCD)	1	1

.....continued

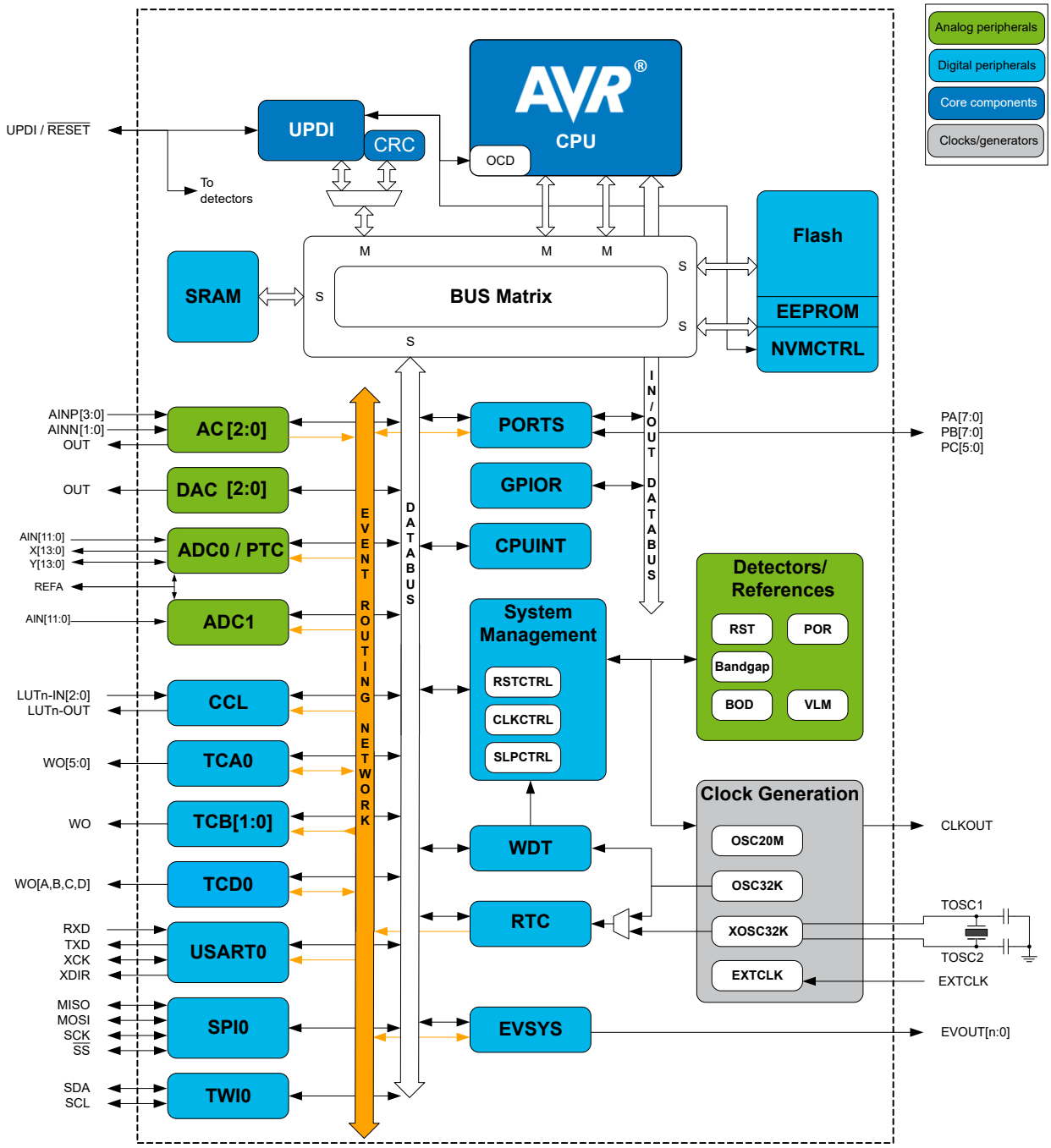
	ATtiny3216	ATtiny3217
Real-Time Counter (RTC)	1	1
USART	1	1
SPI	1	1
TWI (I^2C)	1	1
ADC	2	2
ADC channels	12+8	12+12
DAC	3	3
AC	3	3
AC inputs	3p/2n+ 4p/1n+ 3p/1n	4p/2n+ 4p/2n+ 4p/2n
Peripheral Touch Controller (PTC)(1)	1	1
PTC number of self-capacitance channels	12	14
PTC number of mutual capacitance channels	36	49
Configurable Custom Logic	1	1
Window Watchdog	1	1
Event System channels	6	6
General purpose I/O	18	22
External interrupts	18	22
CRCSCAN	1	1

Note:

1. The PTC takes control over the ADC0 while the PTC is used.

3. Block Diagram

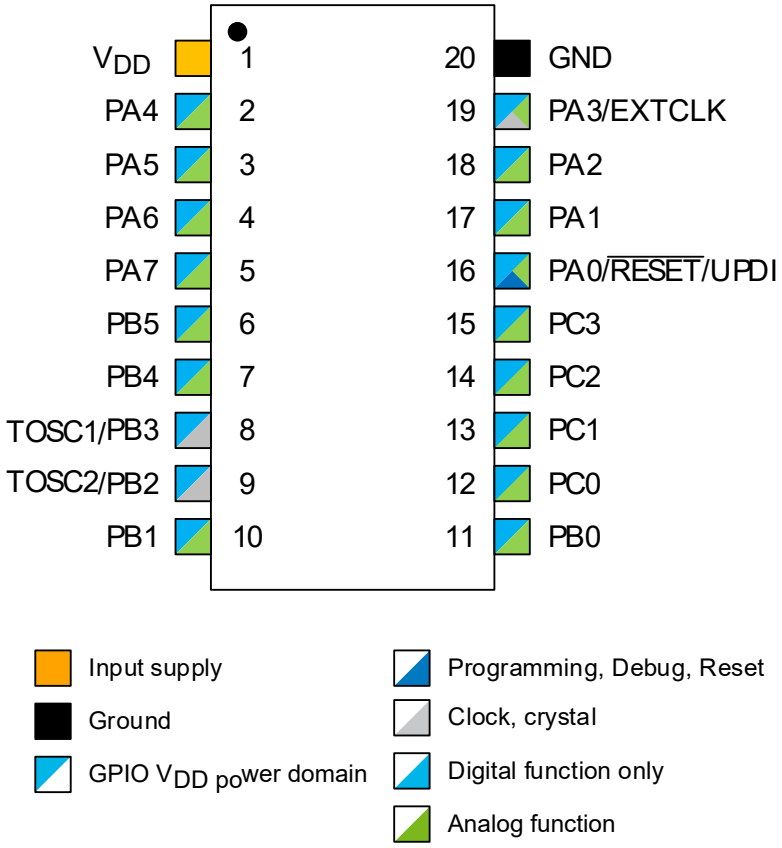
Figure 3-1. tinyAVR® 1-series Block Diagram



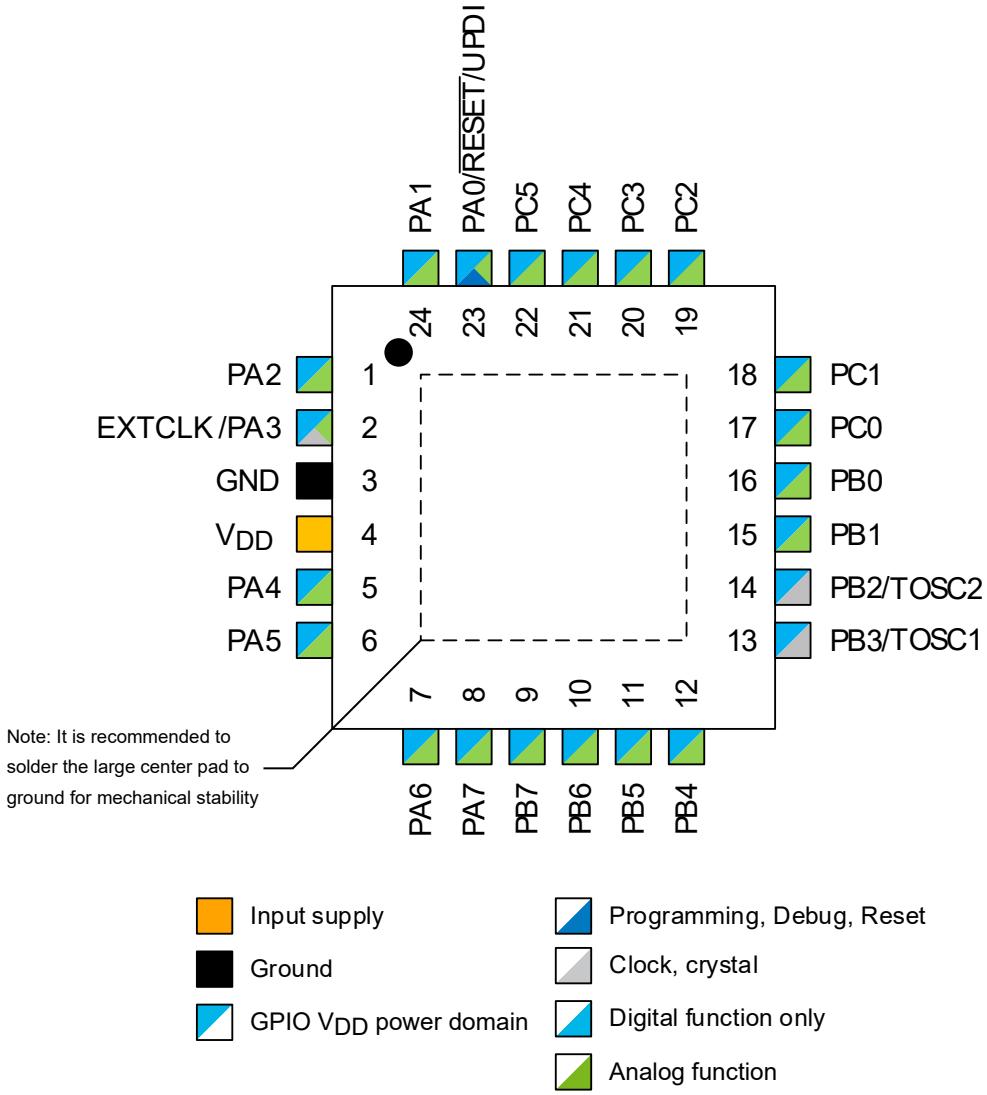
Note: The block diagram represents the largest device of the tinyAVR 1-series, both in terms of pin count and Flash size. See sections 2.1 Configuration Summary and 5.1 Multiplexed Signals for an overview of the features of the specific devices in this data sheet.

4. Pinout

4.1 20-Pin SOIC



4.2 24-Pin VQFN



5. I/O Multiplexing and Considerations

5.1 Multiplexed Signals

Table 5-1. PORT Function Multiplexing

VQFN 24-Pin	SOIC 20-Pin	Pin Name (1,2)	Other/Special	ADC0	ADC1	PTC(4)	AC0	AC1	AC2	DAC0	USART0	SPI0	TWI0	TCA0	TCBn	TCD0	CCL
23	16	PA0	$\overline{\text{RESET}}$/ UPDI	AIN0													LUT0-IN0
24	17	PA1		AIN1							TxD(3)	MOSI	SDA(3)				LUT0-IN1
1	18	PA2	EVOUT0	AIN2							RxD(3)	MISO	SCL(3)				LUT0-IN2
2	19	PA3	EXTCLK	AIN3							XCK(3)	SCK		WO3	TCB1 WO		
3	20	GND															
4	1	V_{DD}															
5	2	PA4		AIN4	AIN0	X0/Y0					XDIR(3)	$\overline{\text{SS}}$		WO4		WOA	LUT0-OUT
6	3	PA5	VREFA	AIN5	AIN1	X1/Y1	OUT	AINN0						WO5	TCB0 WO	WOB	
7	4	PA6		AIN6	AIN2	X2/Y2	AINN0	AINP1	AINP0	OUT							
8	5	PA7		AIN7	AIN3	X3/Y3	AINP0	AINP0	AINN0								LUT1-OUT
9		PB7			AIN4			AINN1	AINP3								
10		PB6			AIN5		AINP3		AINN1								
11	6	PB5	CLKOUT	AIN8		X12/Y12	AINP1		AINP2					WO2(3)			
12	7	PB4		AIN9		X13/Y13	AINN1	AINP3						WO1(3)			LUT0-OUT(3)
13	8	PB3	TOSC1					OUT			RxD			WO0(3)			
14	9	PB2	TOSC2, EVOUT1						OUT		TxD			WO2			
15	10	PB1		AIN10		X4/Y4	AINP2				XCK		SDA	WO1			
16	11	PB0		AIN11		X5/Y5		AINP2	AINP1		XDIR		SCL	WO0			
17	12	PC0			AIN6	X6/Y6						SCK(3)			TCB0 WO(3)	WOC	
18	13	PC1			AIN7	X7/Y7						MISO(3)				WOD	LUT1-OUT(3)
19	14	PC2	EVOUT2		AIN8	X8/Y8						MOSI(3)					
20	15	PC3			AIN9	X9/Y9						$\overline{\text{SS}}$(3)		WO3(3)			LUT1-IN0
21		PC4			AIN10	X10/Y10								WO4(3)	TCB1 WO(3)		LUT1-IN1
22		PC5			AIN11	X11/Y11								WO5(3)			LUT1-IN2

Note:

1. Pin names are of type P*xn*, with *x* being the PORT instance (A, B) and *n* the pin number. The notation for signals is PORT*x*_PIN*n*. All pins can be used as event input.
2. All pins can be used for external interrupt, where pins P*x*2 and P*x*6 of each port have full asynchronous detection.
3. Alternate pin positions. For selecting the alternate positions, refer to section 15. PORTMUX - Port Multiplexer.
4. Every PTC line can be configured as X- or Y-line.

6. Memories

6.1 Overview

The main memories are SRAM data memory, EEPROM data memory, and Flash program memory. Also, the peripheral registers are located in the I/O memory space.

Table 6-1. Physical Properties of Flash Memory

Property	
Size	32 KB
Page size	128B
Number of pages	256
Start address	0x8000

Table 6-2. Physical Properties of SRAM

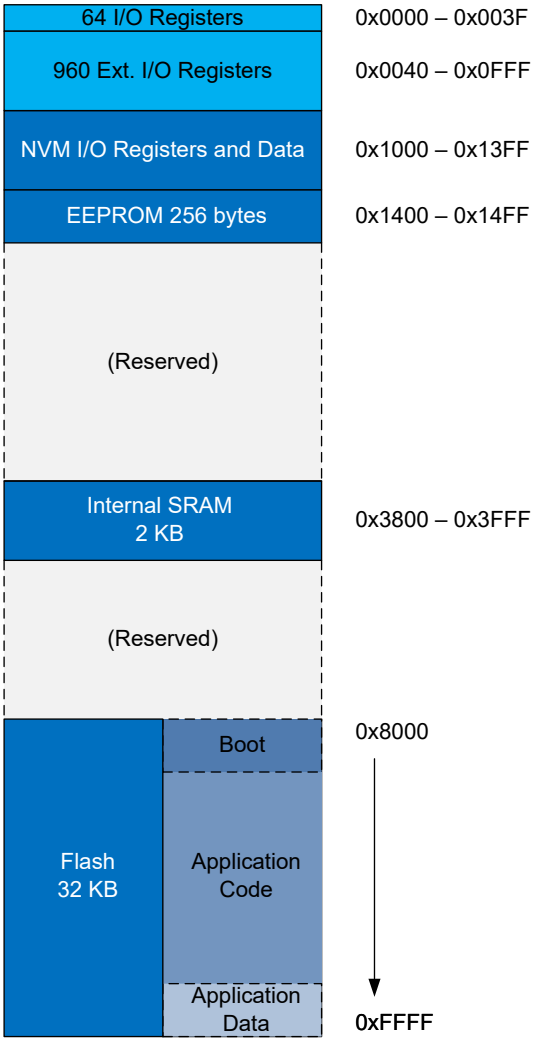
Property	
Size	2 KB
Start address	0x3800

Table 6-3. Physical Properties of EEPROM

Property	
Size	256B
Page size	64B
Number of pages	4
Start address	0x1400

6.2 Memory Map

Figure 6-1. Memory Map



6.3 In-System Reprogrammable Flash Program Memory

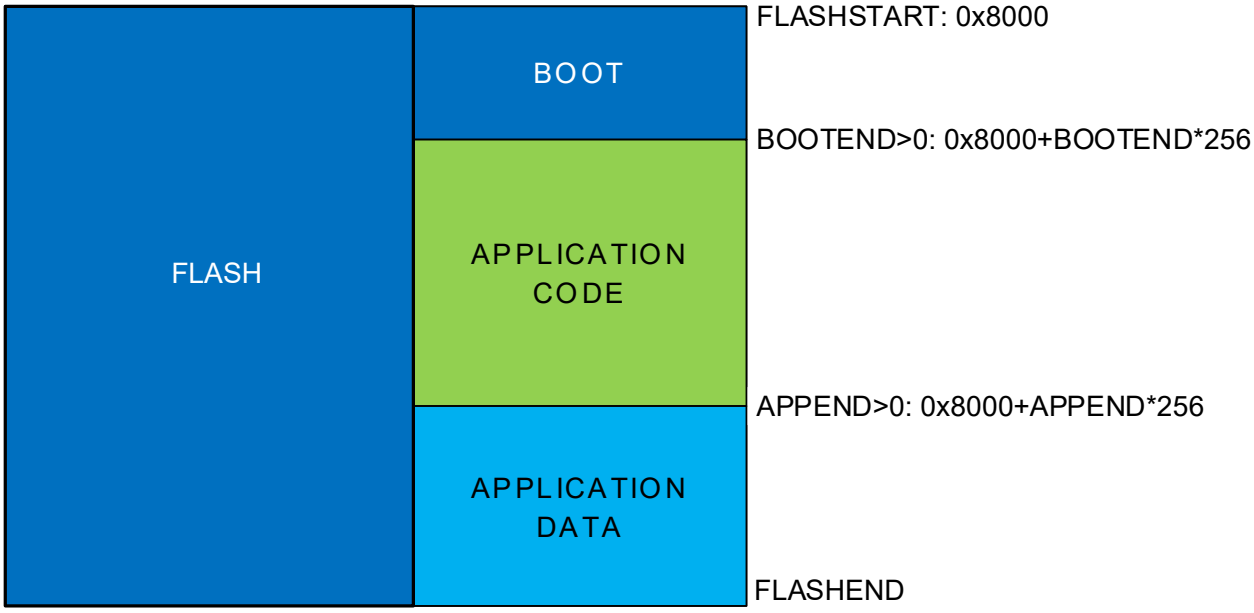
The ATtiny3216/3217 contains 32 KB on-chip in-system reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32-bit wide, the Flash is organized as 4K x 16. For write protection, the Flash program memory space can be divided into three sections (see the illustration below): Bootloader section, Application code section, and Application data section, with restricted access rights among them.

The Program Counter (PC) is 14-bit wide to address the whole program memory. The procedure for writing Flash memory is described in detail in the documentation of the Nonvolatile Memory Controller (NVMCTRL) peripheral.

The entire Flash memory is mapped in the memory space and is accessible with normal `LD`/`ST` instructions as well as the `LPM` instruction. For `LD`/`ST` instructions, the Flash is mapped from address 0x8000. For the `LPM` instruction, the Flash start address is 0x0000.

The ATtiny3216/3217 also has a CRC peripheral that is a master on the bus.

Figure 6-2. Flash and the Three Sections



6.4 SRAM Data Memory

The 2 KB SRAM is used for data storage and stack.

6.5 EEPROM Data Memory

The ATtiny3216/3217 has 256 bytes of EEPROM data memory, see section 6.2 Memory Map. The EEPROM memory supports single-byte read and write. The EEPROM is controlled by the Nonvolatile Memory Controller (NVMCTRL).

6.6 User Row

In addition to the EEPROM, the ATtiny3216/3217 has one extra page of EEPROM memory that can be used for firmware settings; the User Row (USERROW). This memory supports single-byte read and write as the normal EEPROM. The CPU can write and read this memory as normal EEPROM, and the UPDI can write and read it as a normal EEPROM memory if the part is unlocked. The User Row can be written by the UPDI when the part is locked. USERROW is not affected by a chip erase.

6.7 Signature Bytes

All tinyAVR® microcontrollers have a 3-byte signature code that identifies the device. The three bytes reside in a separate address space. For the device, the signature bytes are given in the following table.

Note: When the device is locked, only the System Information Block (SIB) can be accessed.

Table 6-4. Device ID

Device Name	Signature Bytes Address		
	0x00	0x01	0x02
ATtiny3216	0x1E	0x95	0x21
ATtiny3217	0x1E	0x95	0x22

6.8 I/O Memory

All ATtiny3216/3217 I/Os and peripherals are located in the I/O memory space. The I/O address range from 0x00 to 0x3F can be accessed in a single cycle using IN and OUT instructions. The extended I/O memory space from 0x0040 to 0x0FFF can be accessed by the LD/LDS/LDD and ST/STS/STD instructions, transferring data between the 32 general purpose working registers and the I/O memory space.

I/O registers within the address range 0x00-0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the Instruction Set section for more details.

For compatibility with future devices, reserved bits must be written to '0', if accessed. Reserved I/O memory addresses must never be written.

Some of the interrupt flags are cleared by writing a '1' to them. On ATtiny3216/3217 devices, the CBI and SBI instructions will only operate on the specified bit and can be used on registers containing such interrupt flags. The CBI and SBI instructions work with registers 0x00-0x1F only.

General Purpose I/O Registers

The ATtiny3216/3217 devices provide four general purpose I/O registers. These registers can be used for storing any information, and they are particularly useful for storing global variables and interrupt flags. General purpose I/O registers, which reside in the address range 0x1C-0x1F, are directly bit-accessible using the SBI, CBI, SBIS, and SBIC instructions.

6.8.1 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	GPIOR0	7:0	GPIOR[7:0]							
0x01	GPIOR1	7:0	GPIOR[7:0]							
0x02	GPIOR2	7:0	GPIOR[7:0]							
0x03	GPIOR3	7:0	GPIOR[7:0]							

6.8.2 Register Description

6.8.2.1 General Purpose I/O Register n

Name: GPIORn
Offset: 0x00 + n*0x01 [n=0..3]
Reset: 0x00
Property: -

These are general purpose registers that can be used to store data, such as global variables and flags, in the bit-accessible I/O memory space.

Bit	7	6	5	4	3	2	1	0
	GPIOR[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 – GPIOR[7:0] General Purpose I/O Register Byte

6.9 Memory Section Access from CPU and UPDI on Locked Device

The device can be locked so that the memories cannot be read using the UPDI. The locking protects both the Flash (all Boot, Application Code, and Application Date sections), SRAM, and the EEPROM including the FUSE data. This prevents successful reading of application data or code using the debugger interface. Regular memory access from within the application is still enabled.

The device is locked by writing a non-valid key to the LOCKBIT bit field in FUSE.LOCKBIT.

Table 6-5. Memory Access Unlocked (FUSE.LOCKBIT Valid Key)(1)

Memory Section	CPU Access		UPDI Access	
	Read	Write	Read	Write
SRAM	Yes	Yes	Yes	Yes
Registers	Yes	Yes	Yes	Yes
Flash	Yes	Yes	Yes	Yes
EEPROM	Yes	Yes	Yes	Yes
USERROW	Yes	Yes	Yes	Yes
SIGROW	Yes	No	Yes	No
Other fuses	Yes	No	Yes	Yes

Table 6-6. Memory Access Locked (FUSE.LOCKBIT Invalid Key)(1)

Memory Section	CPU Access		UPDI Access	
	Read	Write	Read	Write
SRAM	Yes	Yes	No	No
Registers	Yes	Yes	No	No
Flash	Yes	Yes	No	No
EEPROM	Yes	Yes	No	No
USERROW	Yes	Yes	No	Yes(2)
SIGROW	Yes	No	No	No
Other fuses	Yes	No	No	No

Note:

1. Read operations marked No in the tables may appear to be successful, but the data are not valid. Hence, any attempt of code validation through the UPDI will fail on these memory sections.
2. In the Locked mode, the USERROW can be written using the Fuse Write command, but the current USERROW values cannot be read out.

Important: The only way to unlock a device is through a CHIPERASE. No application data are retained.

6.10 Configuration and User Fuses (FUSE)

Fuses are part of the nonvolatile memory and hold the device configuration. The fuses are available from the device power-up. The fuses can be read by the CPU or the UPDI, but can only be programmed or cleared by the UPDI. The configuration values stored in the fuses are written to their respective target registers at the end of the start-up sequence.

The fuses for peripheral configuration (FUSE) are pre-programmed but can be altered by the user. Altered values in the configuration fuse will be effective only after a Reset.
Note: When writing the fuses, all reserved bits must be written to '1'.

6.10.1 Signature Row Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	DEVICEID0	7:0	DEVICEID[7:0]							
0x01	DEVICEID1	7:0	DEVICEID[7:0]							
0x02	DEVICEID2	7:0	DEVICEID[7:0]							
0x03	SERNUM0	7:0	SERNUM[7:0]							
0x04	SERNUM1	7:0	SERNUM[7:0]							
0x05	SERNUM2	7:0	SERNUM[7:0]							
0x06	SERNUM3	7:0	SERNUM[7:0]							
0x07	SERNUM4	7:0	SERNUM[7:0]							
0x08	SERNUM5	7:0	SERNUM[7:0]							
0x09	SERNUM6	7:0	SERNUM[7:0]							
0x0A	SERNUM7	7:0	SERNUM[7:0]							
0x0B	SERNUM8	7:0	SERNUM[7:0]							
0x0C	SERNUM9	7:0	SERNUM[7:0]							
0x0D ... 0x1F	Reserved									
0x20	TEMPSENSE0	7:0	TEMPSENSE[7:0]							
0x21	TEMPSENSE1	7:0	TEMPSENSE[7:0]							
0x22	OSC16ERR3V	7:0	OSC16ERR3V[7:0]							
0x23	OSC16ERR5V	7:0	OSC16ERR5V[7:0]							
0x24	OSC20ERR3V	7:0	OSC20ERR3V[7:0]							
0x25	OSC20ERR5V	7:0	OSC20ERR5V[7:0]							

6.10.2 Signature Row Description

6.10.2.1 Device ID n

Name: DEVICEIDn
Offset: 0x00 + n*0x01 [n=0..2]
Reset: [Device ID]
Property: -

Each device has a device ID identifying this device and its properties such as memory sizes, pin count, and die revision. This can be used to identify a device and hence, the available features by software. The Device ID consists of three bytes: SIGROW.DEVICEID[2:0].

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

Bits 7:0 – DEVICEID[7:0] Byte n of the Device ID

6.10.2.2 Serial Number Byte n

Name: SERNUMn
Offset: 0x03 + n*0x01 [n=0..9]
Reset: [device serial number]
Property: -

Each device has an individual serial number, representing a unique ID. This can be used to identify a specific device in the field. The serial number consists of ten bytes: SIGROW.SERNUM[9:0].

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

Bits 7:0 – SERNUM[7:0] Serial Number Byte n

6.10.2.3 Temperature Sensor Calibration n

Name: TEMPSENSEn
Offset: 0x20 + n*0x01 [n=0..1]
Reset: [Temperature sensor calibration value]
Property: -

The Temperature Sensor Calibration registers contain correction factors for temperature measurements from the on-chip sensor. The ADC.SIGROW.TEMPSENSE0 is a correction factor for the gain/slope (unsigned), and SIGROW.TEMPSENSE1 is a correction factor for the offset (signed).

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

Bits 7:0 – TEMPSENSE[7:0] Temperature Sensor Calibration Byte n
Refer to the ADC section for a description of how to use this register.

6.10.2.4 OSC16 Error at 3V

Name: OSC16ERR3V
Offset: 0x22
Reset: [Oscillator frequency error value]
Property: -

Bit	7	6	5	4	3	2	1	0
	OSC16ERR3V[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 7:0 – OSC16ERR3V[7:0] OSC16 Error at 3V
These registers contain the signed oscillator frequency error value relative to the nominal oscillator frequency when running at an internal 16 MHz at 3V, as measured during production.

6.10.2.5 OSC16 Error at 5V

Name: OSC16ERR5V
Offset: 0x23
Reset: [Oscillator frequency error value]
Property: -

Bit	7	6	5	4	3	2	1	0
	OSC16ERR5V[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 7:0 – OSC16ERR5V[7:0] OSC16 Error at 5V
These registers contain the signed oscillator frequency error value relative to the nominal oscillator frequency when running at an internal 16 MHz at 5V, as measured during production.

6.10.2.6 OSC20 Error at 3V

Name: OSC20ERR3V
Offset: 0x24
Reset: [Oscillator frequency error value]
Property: -

Bit	7	6	5	4	3	2	1	0
	OSC20ERR3V[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 7:0 – OSC20ERR3V[7:0] OSC20 Error at 3V
These registers contain the signed oscillator frequency error value relative to the nominal oscillator frequency when running at an internal 20 MHz at 3V, as measured during production.

6.10.2.7 OSC20 Error at 5V

Name: OSC20ERR5V
Offset: 0x25
Reset: [Oscillator frequency error value]
Property: -

Bit	7	6	5	4	3	2	1	0
	OSC20ERR5V[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bits 7:0 – OSC20ERR5V[7:0] OSC20 Error at 5V
These registers contain the signed oscillator frequency error value relative to the nominal oscillator frequency when running at an internal 20 MHz at 5V, as measured during production.

6.10.3 Fuse Summary - FUSE

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	WDTCFG	7:0	WINDOW[3:0]				PERIOD[3:0]			
0x01	BODCFG	7:0	LVL[2:0]			SAMPFREQ	ACTIVE[1:0]		SLEEP[1:0]	
0x02	OSCCFG	7:0	OSCLOCK						FREQSEL[1:0]	
0x03	Reserved									
0x04	TCD0CFG	7:0	CMPDEN	CMPCEN	CMPBEN	CMPAEN	CMPD	CMPC	CMPB	CMPA
0x05	SYSCFG0	7:0	CRCSRC[1:0]			TOUTDIS	RSTPINCFG[1:0]			EESAVE
0x06	SYSCFG1	7:0						SUT[2:0]		
0x07	APPEND	7:0	APPEND[7:0]							
0x08	BOOTEND	7:0	BOOTEND[7:0]							
0x09	Reserved									
0x0A	LOCKBIT	7:0	LOCKBIT[7:0]							

6.10.4 Fuse Description

6.10.4.1 Watchdog Configuration

Name: WDTCFG
Offset: 0x00
Reset: -
Property: -

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

Bits 7:4 – WINDOW[3:0] Watchdog Window Time-Out Period
This value is loaded into the WINDOW bit field of the Watchdog Control A (WDT.CTRLA) register during Reset.

Bits 3:0 – PERIOD[3:0] Watchdog Time-Out Period
This value is loaded into the PERIOD bit field of the Watchdog Control A (WDT.CTRLA) register during Reset.

6.10.4.2 BOD Configuration

Name: BODCFG
Offset: 0x01
Reset: -
Property: -

The bit values of this fuse register are written to the corresponding BOD configuration registers at start-up.

Bit	7	6	5	4	3	2	1	0
	LVL[2:0]			SAMPFREQ	ACTIVE[1:0]		SLEEP[1:0]	
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:5 – LVL[2:0] BOD Level
This value is loaded into the LVL bit field of the BOD Control B (BOD.CTRLB) register during Reset.

Value	Name	Description
0x0	BODLEVEL0	1.8V
0x2	BODLEVEL2	2.6V
0x7	BODLEVEL7	4.2V

Note:

- The values in the description are typical
- Refer to the *BOD and POR Characteristics* in *Electrical Characteristics* for maximum and minimum values

Bit 4 – SAMPFREQ BOD Sample Frequency
This value is loaded into the SAMPFREQ bit of the BOD Control A (BOD.CTRLA) register during Reset.

Value	Description
0x0	Sample frequency is 1 kHz
0x1	Sample frequency is 125 Hz

Bits 3:2 – ACTIVE[1:0] BOD Operation Mode in Active and Idle
This value is loaded into the ACTIVE bit field of the BOD Control A (BOD.CTRLA) register during Reset.

Value	Description
0x0	Disabled
0x1	Enabled
0x2	Sampled
0x3	Enabled with wake-up halted until BOD is ready

Bits 1:0 – SLEEP[1:0] BOD Operation Mode in Sleep
This value is loaded into the SLEEP bit field of the BOD Control A (BOD.CTRLA) register during Reset.

Value	Description
0x0	Disabled
0x1	Enabled
0x2	Sampled
0x3	Reserved

6.10.4.3 Oscillator Configuration

Name: OSCCFG
Offset: 0x02
Reset: -
Property: -

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

Bit 7 – OSCLOCK Oscillator Lock
This Fuse bit is loaded to LOCK in CLKCTRL.OSC20MCALIBB during Reset.

Value	Description
0	Calibration registers of the OSC20M oscillator are accessible
1	Calibration registers of the OSC20M oscillator are locked

Bits 1:0 – FREQSEL[1:0] Frequency Select
This bit field selects the operation frequency of the 16/20 MHz internal oscillator (OSC20M) and determines the respective factory calibration values to be written to CAL20M in CLKCTRL.OSC20MCALIBA and TEMPCAL20M in CLKCTRL.OSC20MCALIBB.

Value	Description
0x1	Run at 16 MHz with corresponding factory calibration
0x2	Run at 20 MHz with corresponding factory calibration
Other	Reserved

6.10.4.4 Timer Counter Type D Configuration

Name: TCD0CFG
Offset: 0x04
Reset: -
Property: -

The bit values of this fuse register are written to the corresponding bits in the TCD.FAULTCTRL register of TCD0 at start-up.

Bit	7	6	5	4	3	2	1	0
	CMPDEN	CMPCEN	CMPBEN	CMPAEN	CMPD	CMPC	CMPB	CMPA
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 4, 5, 6, 7 – CMPEN Compare x Enable

Value	Description
0	Compare x output on Pin is disabled
1	Compare x output on Pin is enabled

Bits 0, 1, 2, 3 – CMP Compare x
This bit selects the default state of Compare x after Reset, or when entering debug if FAULTDET is '1'.

Value	Description
0	Compare x default state is '0'
1	Compare x default state is '1'

6.10.4.5 System Configuration 0

Name: SYSCFG0
Offset: 0x05
Reset: 0xC4
Property: -

Bit	7	6	5	4	3	2	1	0
	CRCSRC[1:0]			TOUTDIS	RSTPINCFG[1:0]			EESAVE
Access	R	R		R	R	R		R
Reset	1	1		1	0	1		0

Bits 7:6 – CRCSRC[1:0] CRC Source
See the CRC description for more information about the functionality.

Value	Name	Description
0x0	FLASH	CRC of full Flash (boot, application code and application data)
0x1	BOOT	CRC of the boot section
0x2	BOOTAPP	CRC of application code and boot sections
0x3	NOCRC	No CRC

Bit 4 – TOUTDIS Time-Out Disable
This bit can disable the blocking of NVM writes after POR.
When the TOUTDIS bit in FUSE.SYSCFG0 is '0' and the RSTPINCFG bit field in FUSE.SYSCFG0 is configured to GPIO or RESET, there will be a time-out period after POR that blocks NVM writes.
The NVM write block will last for 768 OSC32K cycles after POR. The EEBUSY and FBUSY bits in the NVMCTRL.STATUS register must read '0' before the page buffer can be filled or NVM commands can be issued.

Value	Description
0	NVM write block is enabled
1	NVM write block is disabled

Bits 3:2 – RSTPINCFG[1:0] Reset Pin Configuration
This bit field selects the Reset/UPDI pin configuration.

Value	Description
0x0	GPIO
0x1	UPDI
0x2	RESET
Other	Reserved

Note: When configuring the RESET pin as GPIO, there is a potential conflict between the GPIO actively driving the output, and a high-voltage UPDI enable sequence initiation. To avoid this, the GPIO output driver is disabled for 768 OSC32K cycles after a System Reset. Enable any interrupts for this pin only after this period.

Bit 0 – EESAVE EEPROM Save During Chip Erase
Note: If the device is locked, the EEPROM is always erased by a chip erase, regardless of this bit.

Value	Description
0	EEPROM erased during chip erase
1	EEPROM not erased under chip erase

6.10.4.6 System Configuration 1

Name: SYSCFG1
Offset: 0x06
Reset: -
Property: -

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

Bits 2:0 – SUT[2:0] Start-Up Time Setting
This bit field selects the start-up time between power-on and code execution.

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

6.10.4.7 Application Code End

Name: APPEND
Offset: 0x07
Reset: -
Property: -

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

Bits 7:0 – APPEND[7:0] Application Code Section End
This bit field sets the end of the application code section in blocks of 256 bytes. The end of the application code section will be set as (BOOT size) + (application code size). The remaining Flash will be application data. A value of `0x00` defines the Flash from BOOTEND*256 to the end of Flash as the application code. When both FUSE.APPEND and FUSE.BOOTEND are `0x00`, the entire Flash is the BOOT section.

6.10.4.8 Boot End

Name: BOOTEND
Offset: 0x08
Reset: -
Property: -

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

Bits 7:0 – BOOTEND[7:0] Boot Section End
This bit field sets the end of the boot section in blocks of 256 bytes. A value of `0x00` defines the whole Flash as the BOOT section. When both FUSE.APPEND and FUSE.BOOTEND are `0x00`, the entire Flash is the BOOT section.

6.10.4.9 Lockbits

Name: LOCKBIT
Offset: 0x0A
Reset: -
Property: -

Bit	7	6	5	4	3	2	1	0
	LOCKBIT[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 – LOCKBIT[7:0] Lockbits
When the part is locked, UPDI cannot access the system bus, so it cannot read out anything but the System Information Block (SIB).

Value	Description
0xC5	Valid key - memory access is unlocked
other	Invalid key - memory access is locked

7. Peripherals and Architecture

7.1 Peripheral Address Map

The address map shows the base address for each peripheral. For complete register description and summary for each peripheral, refer to the respective sections.

Table 7-1. Peripheral Address Map

Base Address	Name	Description
0x0000	VPORTA	Virtual Port A
0x0004	VPORTB	Virtual Port B
0x0008	VPORTC	Virtual Port C
0x001C	GPIO	General Purpose I/O registers
0x0030	CPU	CPU
0x0040	RSTCTRL	Reset Controller
0x0050	SLPCTRL	Sleep Controller
0x0060	CLKCTRL	Clock Controller
0x0080	BOD	Brown-out Detector
0x00A0	VREF	Voltage Reference
0x0100	WDT	Watchdog Timer
0x0110	CPUINT	Interrupt Controller
0x0120	CRCSCAN	Cyclic Redundancy Check Memory Scan
0x0140	RTC	Real-Time Counter
0x0180	EVSYS	Event System
0x01C0	CCL	Configurable Custom Logic
0x0200	PORTMUX	Port Multiplexer
0x0400	PORTA	Port A Configuration
0x0420	PORTB	Port B Configuration
0x0440	PORTC	Port C Configuration
0x0600	ADC0	Analog-to-Digital Converter 0/Peripheral Touch Controller
0x0640	ADC1	Analog-to-Digital Converter 1
0x0680	AC0	Analog Comparator 0
0x0688	AC1	Analog Comparator 1
0x0690	AC2	Analog Comparator 2
0x06A0	DAC0	Digital-to-Analog Converter 0
0x06A8	DAC1	Digital-to-Analog Converter 1
0x06B0	DAC2	Digital-to-Analog Converter 2
0x0800	USART0	Universal Synchronous Asynchronous Receiver Transmitter 0

.....continued		
Base Address	**Name**	**Description**
0x0810	TWI0	Two-Wire Interface 0
0x0820	SPI0	Serial Peripheral Interface 0
0x0A00	TCA0	Timer/Counter Type A 0
0x0A40	TCB0	Timer/Counter Type B 0
0x0A50	TCB1	Timer/Counter Type B 1
0x0A80	TCD0	Timer/Counter Type D 0
0x0F00	SYSCFG	System Configuration
0x1000	NVMCTRL	Nonvolatile Memory Controller
0x1100	SIGROW	Signature Row
0x1280	FUSES	Device-specific fuses
0x1300	USERROW	User Row

7.2 Interrupt Vector Mapping

Each of the interrupt vectors is connected to one peripheral instance, as shown in the table below. A peripheral can have one or more interrupt sources, see the *Interrupt* section in the *Functional Description* of the respective peripheral for more details on the available interrupt sources.

When the Interrupt condition occurs, an Interrupt flag (*name*IF) is set in the Interrupt Flags register of the peripheral (*peripheral*.INTFLAGS).

An interrupt is enabled or disabled by writing to the corresponding Interrupt Enable (*name*IE) bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

The naming of the registers may vary slightly in some peripherals.

An interrupt request is generated when the corresponding interrupt is enabled, and the interrupt flag is set. The interrupt request remains Active until the Interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

Interrupts must be enabled globally for interrupt requests to be generated.

Table 7-2. Interrupt Vector Mapping

Vector Number	Program Address (word)	Peripheral Source	Description
0	0x00	RESET	RESET
1	0x02	CRCSCAN_NMI	NMI - Non-Maskable Interrupt from CRC
2	0x04	BOD_VLM	VLM - Voltage Level Monitor
3	0x06	PORTA_PORT	PORTA - Port A
4	0x08	PORTB_PORT	PORTB - Port B
5	0x0A	PORTC_PORT	PORTC - Port C
6	0x0C	RTC_CNT	RTC - Real-Time Counter
7	0x0E	RTC_PIT	PIT - Periodic Interrupt Timer (in RTC peripheral)
8	0x10	TCA0_LUNF/TCA0_OVF	TCA0 - Timer Counter Type A, LUNF/OVF

.....continued

Vector Number	Program Address (word)	Peripheral Source	Description
9	0x12	TCA0_HUNF	TCA0, HUNF
10	0x14	TCA0_LCMP0/ TCA0_CMP0	TCA0, LCMP0/CMP0
11	0x16	TCA0_LCMP1/ TCA0_CMP1	TCA0, LCMP1/CMP1
12	0x18	TCA0_CMP2/ TCA0_LCMP2	TCA0, LCMP2/CMP2
13	0x1A	TCB0_INT	TCB0 - Timer Counter Type B
14	0x1C	TCB1_INT	TCB1 - Timer Counter Type B
15	0x1E	TCD0_OVF	TCD0 - Timer Counter Type D, OVF
16	0x20	TCD0_TRIG	TCD0, TRIG
17	0x22	AC0_AC	AC0 – Analog Comparator
18	0x24	AC1_AC	AC1 – Analog Comparator
19	0x26	AC2_AC	AC2 – Analog Comparator
20	0x28	ADC0_RESRDY	ADC0 – Analog-to-Digital Converter, RESRDY
21	0x2A	ADC0_WCOMP	ADC0, WCOMP
22	0x2C	ADC1_RESRDY	ADC1 – Analog-to-Digital Converter, RESRDY
23	0x2E	ADC1_WCOMP	ADC1, WCOMP
24	0x30	TWI0_TWIS	TWI0 - Two-Wire Interface/I^2C, TWIS
25	0x32	TWI0_TWIM	TWI0, TWIM
26	0x34	SPI0_INT	SPI0 - Serial Peripheral Interface
27	0x36	USART0_RXC	USART0 - Universal Asynchronous Receiver-Transmitter, RXC
28	0x38	USART0_DRE	USART0, DRE
29	0x3A	USART0_TXC	USART0, TXC
30	0x3C	NVMCTRL_EE	NVM - Nonvolatile Memory

7.3 System Configuration (SYSCFG)

The system configuration contains the revision ID of the part. The revision ID is readable from the CPU, making it useful for implementing application changes between part revisions.

7.3.1 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	Reserved									
0x01	REVID	7:0	REVID[7:0]							

7.3.2 Register Description

7.3.2.1 Device Revision ID Register

Name: REVID
Offset: 0x01
Reset: [revision ID]
Property: -

This register is read-only and displays the device revision ID.

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

Bits 7:0 – REVID[7:0] Revision ID
This bit field contains the device revision. 0x00 = A, 0x01 = B, and so on.

8. AVR® CPU

8.1 Features

- 8-bit, High-Performance AVR RISC CPU:
  - 135 instructions
  - Hardware multiplier
- 32 8-bit Registers Directly Connected to the ALU
- Stack in RAM
- Stack Pointer Accessible in I/O Memory Space
- Direct Addressing of up to 64 KB of Unified Memory
- Efficient Support for 8-, 16-, and 32-bit Arithmetic
- Configuration Change Protection for System-Critical Features
- Native On-Chip Debugging (OCD) Support:
  - Two hardware breakpoints
  - Change of flow, interrupt, and software breakpoints
  - Run-time read-out of Stack Pointer (SP) register, Program Counter (PC), and Status Register (SREG)
  - Register file read- and writable in Stopped mode

8.2 Overview

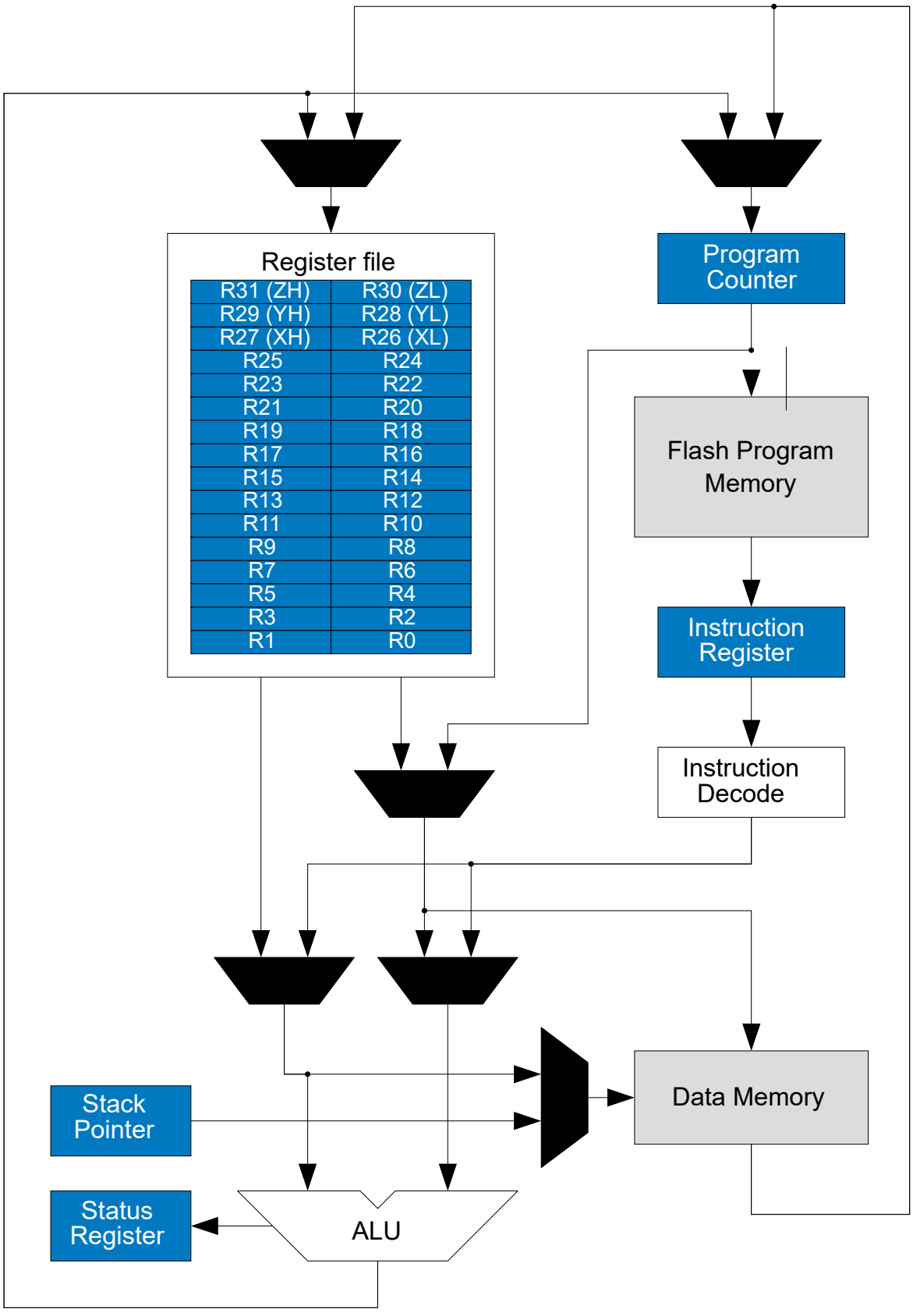
All AVR devices use the AVR 8-bit CPU. The CPU is able to access memories, perform calculations, control peripherals, and execute instructions in the program memory. Interrupt handling is described in a separate section.

8.3 Architecture

To maximize performance and parallelism, the AVR CPU uses a Harvard architecture with separate buses for program and data. Instructions in the program memory are executed with a single-level pipeline. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This enables instructions to be executed on every clock cycle.

Refer to the *Instruction Set Summary* section for a summary of all AVR instructions.

Figure 8-1. AVR® CPU Architecture



8.4 Arithmetic Logic Unit (ALU)

The Arithmetic Logic Unit (ALU) supports arithmetic and logic operations between working registers, or between a constant and a working register. Also, single-register operations can be executed.

The ALU operates in a direct connection with all the 32 general purpose working registers in the register file. Arithmetic operations between working registers or between a working register and an immediate operand are executed in a single clock cycle, and the result is stored in the register file. After an arithmetic or logic operation, the Status Register (CPU.SREG) is updated to reflect information about the result of the operation.

ALU operations are divided into three main categories – arithmetic, logical, and bit functions. Both 8- and 16-bit arithmetic are supported, and the instruction set allows for efficient implementation of the 32-bit arithmetic. The hardware multiplier supports signed and unsigned multiplication and fractional formats.

8.4.1 Hardware Multiplier

The multiplier is capable of multiplying two 8-bit numbers into a 16-bit result. The hardware multiplier supports different variations of signed and unsigned integer and fractional numbers:

- Multiplication of signed/unsigned integers
- Multiplication of signed/unsigned fractional numbers
- Multiplication of a signed integer with an unsigned integer
- Multiplication of a signed fractional number with an unsigned fractional number

A multiplication takes two CPU clock cycles.

8.5 Functional Description

8.5.1 Program Flow

After being reset, the CPU will execute instructions from the lowest address in the Flash program memory, 0x0000. The Program Counter (PC) addresses the next instruction to be fetched.

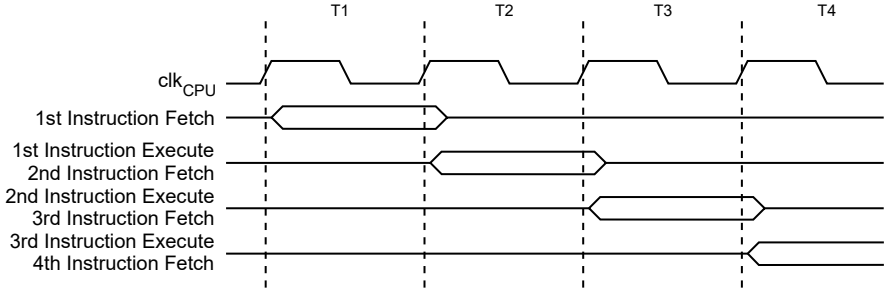
The program flow is supported by conditional and unconditional change of flow instructions, capable of addressing the whole address space directly. Most AVR instructions use a 16-bit word format, and a limited number use a 32-bit format.

During interrupts and subroutine calls, the return address PC is stored on the stack as a word pointer. The stack is allocated in the general data SRAM, and consequently, the stack size is only limited by the total SRAM size and the usage of the SRAM. After the Stack Pointer (SP) is reset, it points to the highest address in the internal SRAM. The SP is read/write accessible in the I/O memory space, enabling easy implementation of multiple stacks or stack areas. The data SRAM can easily be accessed through the five different Addressing modes supported by the AVR CPU.

8.5.2 Instruction Execution Timing

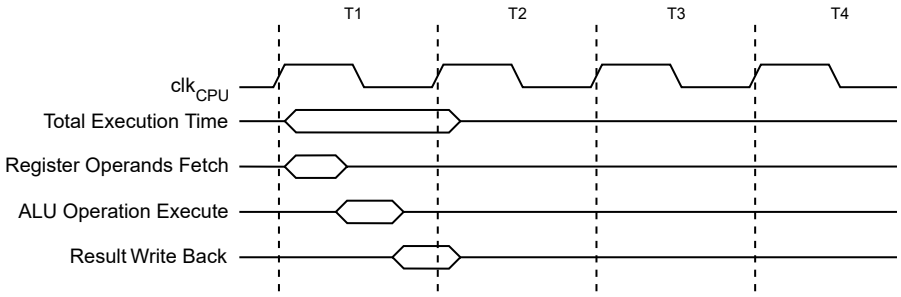
The AVR CPU is clocked by the CPU clock, CLK_CPU. No internal clock division is applied. The figure below shows the parallel instruction fetches and executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept enabling up to 1 MIPS/MHz performance with high efficiency.

Figure 8-2. The Parallel Instruction Fetches and Executions



The following figure shows the internal timing concept for the register file. In a single clock cycle, an ALU operation using two register operands is executed, and the result is stored in the destination register.

Figure 8-3. Single Cycle ALU Operation



8.5.3 Status Register

The Status Register (CPU.SREG) contains information about the result of the most recently executed arithmetic or logic instructions. This information can be used for altering the program flow to perform conditional operations.

CPU.SREG is updated after all ALU operations, as specified in the *Instruction Set Summary* section. This will, in many cases, remove the need for using the dedicated compare instructions, resulting in a faster and more compact code. CPU.SREG is not automatically stored or restored when entering or returning from an Interrupt Service Routine (ISR). Therefore, maintaining the Status Register between context switches must be handled by user-defined software. CPU.SREG is accessible in the I/O memory space.

8.5.4 Stack and Stack Pointer

The stack is used for storing return addresses after interrupts and subroutine calls. Also, it can be used for storing temporary data. The Stack Pointer (SP) always points to the top of the stack. The SP is defined by the Stack Pointer bits in the Stack Pointer register (CPU.SP). The CPU.SP is implemented as two 8-bit registers that are accessible in the I/O memory space.

Data are pushed and popped from the stack using the `PUSH` and `POP` instructions. The stack grows from higher to lower memory locations. This means that pushing data onto the stack decreases the SP, and popping data off the stack increases the SP. The SP is automatically set to the highest address of the internal SRAM after being reset. If the stack is changed, it must be set to point above the SRAM start address (see the SRAM Data Memory section in the Memories chapter for the SRAM start address), and it must be defined before any subroutine calls are executed and before interrupts are enabled. See the table below for SP details.

Table 8-1. Stack Pointer Instructions

Instruction	Stack Pointer	Description
`PUSH`	Decremented by 1	Data are pushed onto the stack

`CALL`
`ICALL`
`RCALL`
	Decremented by 2	A return address is pushed onto the stack with a subroutine call or interrupt

`POP`	Incremented by 1	Data are popped from the stack
`RET` `RETI`	Incremented by 2	A return address is popped from the stack with a return from subroutine or return from interrupt

During interrupts or subroutine calls, the return address is automatically pushed on the stack as a word pointer, and the SP is decremented by two. The return address consists of two bytes and the Least Significant Byte (LSB) is pushed on the stack first (at the higher address). As an example, a byte pointer return address of 0x0006 is saved on the stack as 0x0003 (shifted one bit to the right), pointing to the fourth 16-bit instruction word in the program memory. The return address is popped off the stack with `RETI` (when returning from interrupts) and `RET` (when returning from subroutine calls), and the SP is incremented by two.

The SP is decremented by '`1`' when data are pushed on the stack with the `PUSH` instruction, and incremented by '`1`' when data are popped off the stack using the `POP` instruction.

To prevent corruption when updating the SP from software, a write to SPL will automatically disable interrupts for up to four instructions or until the next I/O memory write, whichever comes first.

8.5.5 Register File

The register file consists of 32 8-bit general purpose working registers used by the CPU. The register file is located in a separate address space from the data memory.

All CPU instructions that operate on working registers have direct and single-cycle access to the register file. Some limitations apply to which working registers can be accessed by an instruction, like the constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI ORI, and LDI. These instructions apply to the second half of the working registers in the register file, R16 to R31. See the *AVR Instruction Set Manual* for further details.

Figure 8-4. AVR® CPU General Purpose Working Registers

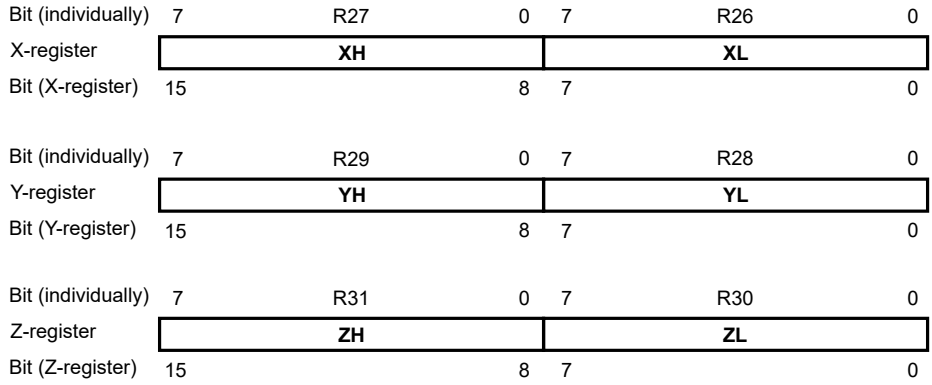
7 0	Addr.	
R0	0x00	
R1	0x01	
R2	0x02	
...		
R13	0x0D	
R14	0x0E	
R15	0x0F	
R16	0x10	
R17	0x11	
...		
R26	0x1A	X-register Low Byte
R27	0x1B	X-register High Byte
R28	0x1C	Y-register Low Byte
R29	0x1D	Y-register High Byte
R30	0x1E	Z-register Low Byte
R31	0x1F	Z-register High Byte

8.5.5.1 The X-, Y-, and Z-Registers

Working registers R26...R31 have added functions besides their general purpose usage.

These registers can form 16-bit Address Pointers for indirect addressing of data memory. These three address registers are called the X-register, Y-register, and Z-register. The Z-register can also be used as Address Pointer for program memory.

Figure 8-5. The X-, Y-, and Z-Registers

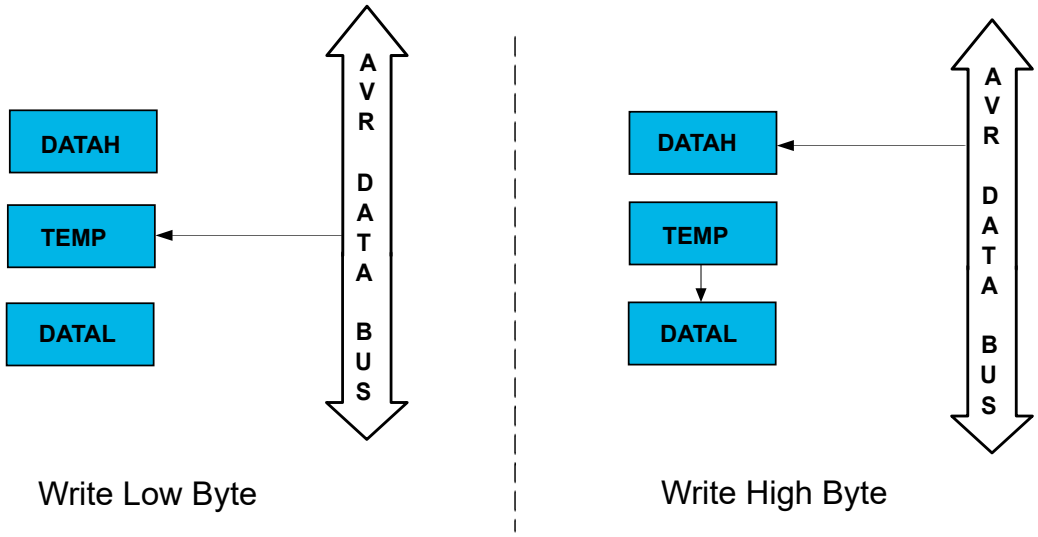


The lowest register address holds the Least Significant Byte (LSB), and the highest register address holds the Most Significant Byte (MSB). These address registers can function as fixed displacement, automatic increment, and automatic decrement, with different `LD*`/`ST*` instructions. See the *Instruction Set Summary* section for details.

8.5.6 Accessing 16-bit Registers

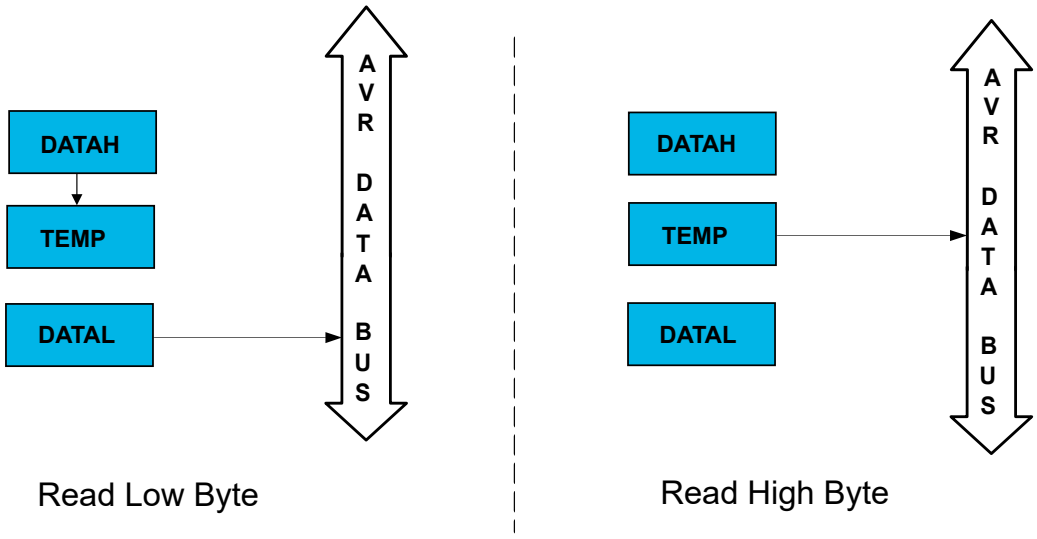
Most of the registers for the ATtiny3216/3217 devices are 8-bit registers, but the devices also features a few 16-bit registers. As the AVR data bus has a width of 8 bits, accessing the 16-bit requires two read or write operations. All the 16-bit registers of the ATtiny3216/3217 devices are connected to the 8-bit bus through a temporary (TEMP) register.

Figure 8-6. 16-Bit Register Write Operation



For a 16-bit write operation, the low byte register (e.g. DATAL) of the 16-bit register must be written before the high byte register (e.g. DATAH). Writing the low byte register will result in a write to the temporary (TEMP) register instead of the low byte register, as shown in the left side of Figure 8-6. When the high byte register of the 16-bit register is written, TEMP will be copied into the low byte of the 16-bit register in the same clock cycle, as shown in the right side of Figure 8-6.

Figure 8-7. 16-Bit Register Read Operation



For a 16-bit read operation, the low byte register (e.g. DATAL) of the 16-bit register must be read before the high byte register (e.g. DATAH). When the low byte register is read, the high byte register of the 16-bit register is copied into

the temporary (TEMP) register in the same clock cycle, as show in the left side of Figure 8-7. Reading the high byte register will result in a read from TEMP instead of the high byte register, as shown in right side of Figure 8-7.

The described mechanism ensures that the low and high bytes of 16-bit registers are always accessed simultaneously when reading or writing the registers.

Interrupts can corrupt the timed sequence if an interrupt is triggered during a 16-bit read/write operation and a 16-bit register within the same peripheral is accessed in the interrupt service routine. To prevent this, interrupts should be disabled when writing or reading 16-bit registers. Alternatively, the temporary register can be read before and restored after the 16-bit access in the interrupt service routine.

8.5.6.1 Accessing 24-Bit Registers

For 24-bit registers, the read and write access is done in the same way as described for 16-bit registers, except there are two temporary registers for 24-bit registers. The Least Significant Byte must be written first when doing a write, and read first when doing a read.

8.5.7 Configuration Change Protection (CCP)

System critical I/O register settings are protected from accidental modification. Flash self-programming (via store to NVM controller) is protected from accidental execution. This is handled globally by the Configuration Change Protection (CCP) register.

Changes to the protected I/O registers or bits, or execution of protected instructions, are only possible after the CPU writes a signature to the CCP register. The different signatures are listed in the description of the CCP register (CPU.CCP).

There are two modes of operation: One for protected I/O registers, and one for protected self-programming.

8.5.7.1 Sequence for Write Operation to Configuration Change Protected I/O Registers

In order to write to registers protected by CCP, these steps are required:

1. The software writes the signature that enables change of protected I/O registers to the CCP bit field in the CPU.CCP register.
2. Within four instructions, the software must write the appropriate data to the protected register.
   Most protected registers also contain a Write Enable/Change Enable/Lock bit. This bit must be written to '1' in the same operation as the data are written.

   The protected change is immediately disabled if the CPU performs write operations to the I/O register or data memory, if load or store accesses to Flash, NVMCTRL, or EEPROM are conducted, or if the SLEEP instruction is executed.

8.5.7.2 Sequence for Execution of Self-Programming

In order to execute self-programming (the execution of writes to the NVM controller's command register), the following steps are required:

1. The software temporarily enables self-programming by writing the SPM signature to the CCP register (CPU.CCP).
2. Within four instructions, the software must execute the appropriate instruction. The protected change is immediately disabled if the CPU performs accesses to the Flash, NVMCTRL, or EEPROM, or if the SLEEP instruction is executed.

Once the correct signature is written by the CPU, interrupts will be ignored for the duration of the configuration change enable period. Any interrupt request (including non-maskable interrupts) during the CCP period will set the corresponding Interrupt flag as normal, and the request is kept pending. After the CCP period is completed, any pending interrupts are executed according to their level and priority.

8.5.8 On-Chip Debug Capabilities

The AVR CPU includes native On-Chip Debug (OCD) support. It includes some powerful debug capabilities to enable profiling and detailed information about the CPU state. It is possible to alter the CPU state and resume code execution. Also, normal debug capabilities like hardware Program Counter breakpoints, breakpoints on change of flow instructions, breakpoints on interrupts, and software breakpoints (BREAK instruction) are present. Refer to the *Unified Program and Debug Interface* section for details about OCD.

8.6 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	CCP	7:0	CCP[7:0]							
0x05 ... 0x0C	Reserved									
0x0D	SP	7:0	SP[7:0]							
		15:8	SP[15:8]							
0x0F	SREG	7:0	I	T	H	S	V	N	Z	C

8.7 Register Description

8.7.1 Configuration Change Protection

Name: CCP
Offset: 0x04
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	CCP[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 – CCP[7:0] Configuration Change Protection
Writing the correct signature to this bit field allows changing protected I/O registers or executing protected instructions within the next four CPU instructions executed.
All interrupts are ignored during these cycles. After these cycles are completed, the interrupts will automatically be handled again by the CPU, and any pending interrupts will be executed according to their level and priority.
When the protected I/O register signature is written, CCP[0] will read as '1' as long as the CCP feature is enabled.
When the protected self-programming signature is written, CCP[1] will read as '1' as long as the CCP feature is enabled.
CCP[7:2] will always read as '0'.

Value	Name	Description
0x9D	SPM	Allow Self-Programming
0xD8	IOREG	Unlock protected I/O registers

8.7.2 Stack Pointer

Name: SP
Offset: 0x0D
Reset: Top of stack
Property: -

The CPU.SP register holds the Stack Pointer (SP) that points to the top of the stack. After being reset, the SP points to the highest internal SRAM address.

Only the number of bits required to address the available data memory, including external memory (up to 64 KB), is implemented for each device. Unused bits will always read as '0'.

The CPU.SPL and CPU.SPH register pair represents the 16-bit value, CPU.SP. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + 0x01.

To prevent corruption when updating the SP from software, a write to CPU.SPL will automatically disable interrupts for the next four instructions or until the next I/O memory write, whichever comes first.

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

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

Bits 15:8 – SP[15:8] Stack Pointer High Byte
These bits hold the MSB of the 16-bit register.

Bits 7:0 – SP[7:0] Stack Pointer Low Byte
These bits hold the LSB of the 16-bit register.

8.7.3 Status Register

Name: SREG
Offset: 0x0F
Reset: 0x00
Property: -

The Status Register contains information about the result of the most recently executed arithmetic or logic instructions. For details about the bits in this register and how they are influenced by different instructions, see the *Instruction Set Summary* section.

Bit	7	6	5	4	3	2	1	0
	I	T	H	S	V	N	Z	C
Access	R/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 – I Global Interrupt Enable Bit
Writing a '1' to this bit enables interrupts on the device.
Writing a '0' to this bit disables interrupts on the device, independent of the individual interrupt enable settings of the peripherals.
This bit is not cleared by hardware while entering an Interrupt Service Routine (ISR) or set when the RETI instruction is executed.
This bit can be set and cleared by software with the SEI and CLI instructions.
Changing the I bit through the I/O register results in a one-cycle Wait state on the access.

Bit 6 – T Transfer Bit
The bit copy instructions, Bit Load (BLD) and Bit Store (BST), use the T bit as source or destination for the operated bit.

Bit 5 – H Half Carry Flag
This flag is set when there is a half carry in arithmetic operations that support this, and is cleared otherwise. Half carry is useful in BCD arithmetic.

Bit 4 – S Sign Flag
This flag is always an Exclusive Or (*XOR*) between the Negative flag (N) and the Two's Complement Overflow flag (V).

Bit 3 – V Two's Complement Overflow Flag
This flag is set when there is an overflow in arithmetic operations that support this, and is cleared otherwise.

Bit 2 – N Negative Flag
This flag is set when there is a negative result in an arithmetic or logic operation, and is cleared otherwise.

Bit 1 – Z Zero Flag
This flag is set when there is a zero result in an arithmetic or logic operation, and is cleared otherwise.

Bit 0 – C Carry Flag
This flag is set when there is a carry in an arithmetic or logic operation, and is cleared otherwise.

9. NVMCTRL - Nonvolatile Memory Controller

9.1 Features

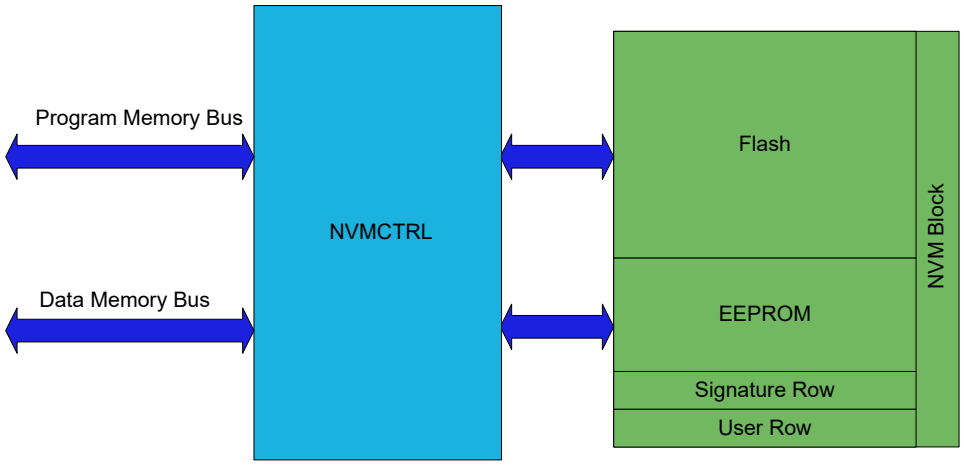
- Unified Memory
- In-System Programmable
- Self-Programming and Boot Loader Support
- Configurable Sections for Write Protection:
  - Boot section for boot loader code or application code
  - Application code section for application code
  - Application data section for application code or data storage
- Signature Row for Factory-Programmed Data:
  - ID for each device type
  - Serial number for each device
  - Calibration bytes for factory-calibrated peripherals
- User Row for Application Data:
  - Can be read and written from software
  - Can be written from UPDI on locked device
  - Content is kept after chip erase

9.2 Overview

The NVM Controller (NVMCTRL) is the interface between the CPU and Nonvolatile Memories (Flash, EEPROM, Signature Row, User Row and fuses). These are reprogrammable memory blocks that retain their values even when they are not powered. The Flash is mainly used for program storage and can also be used for data storage. The EEPROM is used for data storage and can be programmed while the CPU is running the program from the Flash.

9.2.1 Block Diagram

Figure 9-1. NVMCTRL Block Diagram



9.3 Functional Description

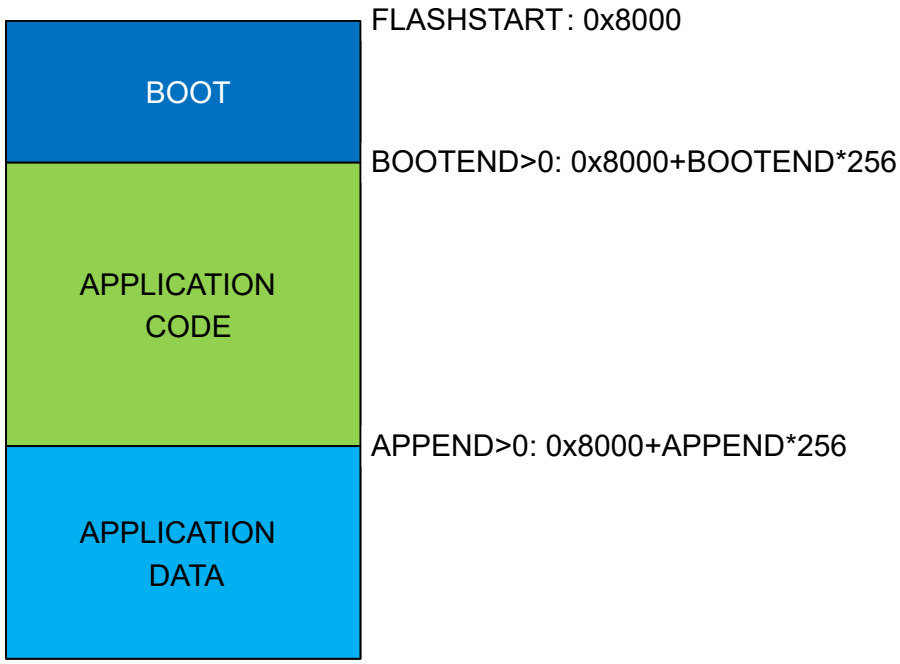
9.3.1 Memory Organization

9.3.1.1 Flash

The Flash is divided into a set of pages. A page is the basic unit addressed when programming the Flash. It is only possible to write or erase a whole page at a time. One page consists of several words.

The Flash can be divided into three sections in blocks of 256 bytes for different security. The three different sections are BOOT, Application Code (APPCODE), and Application Data (APPDATA).

Figure 9-2. Flash Sections



Section Sizes

The sizes of these sections are set by the Boot Section End fuse (FUSE.BOOTEND) and the Application Code Section End fuse (FUSE.APPEND).

The fuses select the section sizes in blocks of 256 bytes. The BOOT section stretches from the start of the Flash until BOOTEND. The APPCODE section runs from BOOTEND until APPEND. The remaining area is the APPDATA section. If APPEND is written to '0', the APPCODE section runs from BOOTEND to the end of Flash (removing the APPDATA section). If BOOTEND and APPEND are written to '0', the entire Flash is regarded as the BOOT section. APPEND may either be set to '0' or a value greater than or equal to BOOTEND.

Table 9-1. Setting Up Flash Sections

BOOTEND	APPEND	BOOT Section	APPCODE Section	APPDATA Section
0	0	0 to FLASHEND	—	—
> 0	0	0 to 256*BOOTEND	256*BOOTEND to FLASHEND	—
> 0	== BOOTEND	0 to 256*BOOTEND	—	256*BOOTEND to FLASHEND
> 0	> BOOTEND	0 to 256*BOOTEND	256*BOOTEND to 256*APPEND	256*APPEND to FLASHEND

Note:

1. See also the BOOTEND and APPEND descriptions.
2. Interrupt vectors are by default located after the BOOT section. This can be changed in the interrupt controller.

If FUSE.BOOTEND is written to 0x04 and FUSE.APPEND is written to 0x08, the first 4*256 bytes will be BOOT, the next 4*256 bytes will be APPCODE, and the remaining Flash will be APPDATA.

Inter-Section Write Protection

Between the three Flash sections, directional write protection is implemented:

- The code in the BOOT section can write to APPCODE and APPDATA
- The code in APPCODE can write to APPDATA
- The code in APPDATA cannot write to Flash or EEPROM

Boot Section Lock and Application Code Section Write Protection

The two Lock bits (APCWP and BOOTLOCK in NVMCTRL.CTRLB) can be set to lock further updates of the respective APPCODE or BOOT section until the next Reset.

The CPU can never write to the BOOT section. NVMCTRL_CTRLB.BOOTLOCK prevents reads and execution of code from the BOOT section.

9.3.1.2 EEPROM

The EEPROM is divided into a set of pages where one page consists of multiple bytes. The EEPROM has byte granularity on erase/write. Within one page, only the bytes marked to be updated will be erased/written. The byte is marked by writing a new value to the page buffer for that address location.

9.3.1.3 User Row

The User Row is one extra page of EEPROM. This page can be used to store various data, such as calibration/ configuration data and serial numbers. This page is not erased by a chip erase. The User Row is written as normal EEPROM, but in addition, it can be written through UPDI on a locked device.

9.3.2 Memory Access

9.3.2.1 Read

Reading of the Flash and EEPROM is done by using load instructions with an address according to the memory map. Reading any of the arrays while a write or erase is in progress will result in a bus wait, and the instruction will be suspended until the ongoing operation is complete.

9.3.2.2 Page Buffer Load

The page buffer is loaded by writing directly to the memories as defined in the memory map. Flash, EEPROM, and User Row share the same page buffer so only one section can be programmed at a time. The Least Significant bits (LSb) of the address are used to select where in the page buffer the data is written. The resulting data will be a binary AND operation between the new and the previous content of the page buffer. The page buffer will automatically be erased (all bits set) after:

- A device Reset
- Any page write or erase operation
- A Clear Page Buffer command
- A device wake-up from any Sleep mode

9.3.2.3 Programming

For page programming, filling the page buffer and writing the page buffer into Flash, User Row, and EEPROM are two separate operations.

Before programming a Flash page with the data in the page buffer, the Flash page must be erased. The page buffer is also erased when the device enters a Sleep mode. Programming an unerased Flash page will corrupt its content.

The Flash can either be written with the erase and write separately, or one command handling both:

Alternative 1:

1. Fill the page buffer.
2. Write the page buffer to Flash with the Erase and Write Page (ERWP) command.

Alternative 2:

1. Write to a location in the page to set up the address.
2. Perform an Erase Page (ER) command.
3. Fill the page buffer.
4. Perform a Write Page (WP) command.

The NVM command set supports both a single erase and write operation, and split Erase Page (ER) and Write Page (WP) commands. This split commands enable shorter programming time for each command, and the erase operations can be done during non-time-critical programming execution.

The EEPROM programming is similar, but only the bytes updated in the page buffer will be written or erased in the EEPROM.

9.3.2.4 Commands

Reading the Flash/EEPROM and writing the page buffer is handled with normal load/store instructions. Other operations, such as writing and erasing the memory arrays, are handled by commands in the NVM.

To execute a command in the NVM:

1. Confirm that any previous operation is completed by reading the Busy Flags (EEBUSY and FBUSY) in the NVMCTRL.STATUS register.
2. Write the NVM command unlock to the Configuration Change Protection register in the CPU (CPU.CCP).
3. Write the desired command value to the CMD bits in the Control A register (NVMCTRL.CTRLA) within the next four instructions.

9.3.2.4.1 Write Command

The Write Page (WP) command of the Flash controller writes the content of the page buffer to the Flash or EEPROM.

If the write is to the Flash, the CPU will stop executing code as long as the Flash is busy with the write operation. If the write is to the EEPROM, the CPU can continue executing code while the operation is ongoing.

The page buffer will automatically be cleared after the operation is finished.

9.3.2.4.2 Erase Command

The Erase Page (ER) command erases the current page. There must be one byte written in the page buffer for the Erase Page (ER) command to take effect.

For erasing the Flash, first, write to one address in the desired page, then execute the command. The whole page in the Flash will then be erased. The CPU will be halted while the erase is ongoing.

For the EEPROM, only the bytes written in the page buffer will be erased when the command is executed. To erase a specific byte, write to its corresponding address before executing the command. To erase a whole page, all the bytes in the page buffer have to be updated before executing the command. The CPU can continue running code while the operation is ongoing.

The page buffer will automatically be cleared after the operation is finished.

9.3.2.4.3 Erase/Write Operation

The Erase and Write Page (ERWP) command is a combination of the Erase Page and Write Page commands, but without clearing the page buffer after the Erase Page command: The erase/write operation first erases the selected page, then it writes the content of the page buffer to the same page.

When executed on the Flash, the CPU will be halted when the operations are ongoing. When executed on EEPROM, the CPU can continue executing code.

The page buffer will automatically be cleared after the operation is finished.

9.3.2.4.4 Page Buffer Clear Command

The Page Buffer Clear (PBC) command clears the page buffer. The contents of the page buffer will be all '1's after the operation. The CPU will be halted when the operation executes (seven CPU cycles).

9.3.2.4.5 Chip Erase Command

The Chip Erase (CHER) command erases the Flash and the EEPROM. The EEPROM is unaltered if the EEPROM Save During Chip Erase (EESAVE) fuse in FUSE.SYSCFG0 is set. The Flash will not be protected by Boot Section Lock (BOOTLOCK) or Application Code Section Write Protection (APCWP) in NVMCTRL.CTRLB. The memory will be all '1's after the operation.

9.3.2.4.6 EEPROM Erase Command

The EEPROM Erase (EEER) command erases the EEPROM. The EEPROM will be all '1's after the operation. The CPU will be halted while the EEPROM is being erased.

9.3.2.4.7 Write Fuse Command

The Write Fuse (WFU) command writes the fuses. It can only be used by the UPDI; the CPU cannot start this command.

Follow this procedure to use the Write Fuse command:

1. Write the address of the fuse to the Address register (NVMCTRL.ADDR).
2. Write the data to be written to the fuse to the Data register (NVMCTRL.DATA).
3. Execute the Write Fuse command.
4. After the fuse is written, a Reset is required for the updated value to take effect.

For reading fuses, use a regular read on the memory location.

9.3.2.5 Write Access after Reset

After a Power-on Reset (POR), the NVMCTRL rejects any write attempts to the NVM for a certain time. During this period, the Flash Busy (FBUSY) and the EEPROM Busy (EBUSY) bits in the STATUS register will read '1'. EEBUSY and FBUSY must read '0' before the page buffer can be filled, or NVM commands can be issued.

This time-out period is disabled either by writing the Time-Out Disable bit (TOUTDIS) in the System Configuration 0 Fuse (FUSE.SYSCFG0) to '0' or by configuring the RSTPINCFG bit field in FUSE.SYSCFG0 to UPDI.

9.3.3 Preventing Flash/EEPROM Corruption

During periods of low V_{DD}, the Flash program or EEPROM data can be corrupted if the supply voltage is too low for the CPU and the Flash/EEPROM to operate properly. These issues are the same on-board level systems using Flash/EEPROM, and the same design solutions may be applied.

A Flash/EEPROM corruption can be caused by two situations 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 itself can execute instructions incorrectly when the supply voltage is too low.

See the *Electrical Characteristics* chapter for Maximum Frequency vs. V_{DD}.

Attention: Flash/EEPROM corruption can be avoided by taking these measures:

1. Keep the device in Reset during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-Out Detector (BOD).
2. The voltage level monitor in the BOD can be used to prevent starting a write to the EEPROM close to the BOD level.
3. If the detection levels of the internal BOD do not match the required detection level, an external low V_{DD} Reset protection circuit can be used. If a Reset occurs while a write operation is ongoing, the write operation will be aborted.

9.3.4 Interrupts

Table 9-2. Available Interrupt Vectors and Sources

Offset	Name	Vector Description	Conditions
0x00	EEREADY	NVM	The EEPROM is ready for new write/erase operations.

When an interrupt condition occurs, the corresponding interrupt flag is set in the Interrupt Flags (NVMCTRL.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding bit in the Interrupt Control (NVMCTRL.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the NVMCTRL.INTFLAGS register for details on how to clear interrupt flags.

9.3.5 Sleep Mode Operation

If there is no ongoing write operation, the NVMCTRL will enter a sleep mode when the system enters a sleep mode.

If a write operation is ongoing when the system enters a sleep mode, the NVM block, the NVM Controller, and the system clock will remain ON until the write is finished. This is valid for all sleep modes, including Power-Down sleep mode.

The EEPROM Ready interrupt will wake up the device only from Idle sleep mode.

The page buffer is cleared when waking up from sleep.

9.3.6 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 9-3. NVMCTRL - Registers under Configuration Change Protection

Register	Key
NVMCTRL.CTRLA	SPM

9.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0						CMD[2:0]		
0x01	CTRLB	7:0							BOOTLOCK	APCWP
0x02	STATUS	7:0						WRERROR	EEBUSY	FBUSY
0x03	INTCTRL	7:0								EEREADY
0x04	INTFLAGS	7:0								EEREADY
0x05	Reserved									
0x06	DATA	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
0x08	ADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							

9.5 Register Description

9.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: Configuration Change Protection

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

Bits 2:0 – CMD[2:0] Command
Write this bit field to issue a command. The Configuration Change Protection key for self-programming (SPM) has to be written within four instructions before this write.

Value	Name	Description
0x0	-	No command
0x1	WP	Write page buffer to memory (NVMCTRL.ADDR selects which memory)
0x2	ER	Erase page (NVMCTRL.ADDR selects which memory)
0x3	ERWP	Erase and write page (NVMCTRL.ADDR selects which memory)
0x4	PBC	Page buffer clear
0x5	CHER	Chip erase: Erase Flash and EEPROM (unless EESAVE in FUSE.SYSCFG is '1')
0x6	EEER	EEPROM Erase
0x7	WFU	Write fuse (only accessible through UPDI)

9.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

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

Bit 1 – BOOTLOCK Boot Section Lock
Writing a '1' to this bit locks the boot section from read and instruction fetch.
If this bit is '1', a read from the boot section will return '0'. A fetch from the boot section will also return '0' as instruction.
This bit can be written from the boot section only. It can only be cleared to '0' by a Reset.
This bit will take effect only when the boot section is left the first time after the bit is written.

Bit 0 – APCWP Application Code Section Write Protection
Writing a '1' to this bit protects the application code section from further writes.
This bit can only be written to '1'. It is cleared to '0' only by Reset.

9.5.3 Status

Name: STATUS
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
						WRERROR	EEBUSY	FBUSY
Access						R	R	R
Reset						0	0	0

Bit 2 – WRERROR Write Error
This bit will read '1' when a write error has happened. A write error could be writing to different sections before doing a page write or writing to a protected area. This bit is valid for the last operation.

Bit 1 – EEBUSY EEPROM Busy
This bit will read '1' when the EEPROM is busy with a command.

Bit 0 – FBUSY Flash Busy
This bit will read '1' when the Flash is busy with a command.

9.5.4 Interrupt Control

Name: INTCTRL
Offset: 0x03
Reset: 0x00
Property: -

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

Bit 0 – EEREADY EEPROM Ready Interrupt
Writing a '1' to this bit enables the interrupt, which indicates that the EEPROM is ready for new write/erase operations.
This is a level interrupt that will be triggered only when the EEREADY flag in the INTFLAGS register is set to '0'. Thus, the interrupt must not be enabled before triggering an NVM command, as the EEREADY flag will not be set before the NVM command issued. The interrupt may be disabled in the interrupt handler.

9.5.5 Interrupt Flags

Name: INTFLAGS
Offset: 0x04
Reset: 0x00
Property: -

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

Bit 0 – EEREADY EEREADY Interrupt Flag
This flag is set continuously as long as the EEPROM is not busy. This flag is cleared by writing a '1' to it.

9.5.6 Data

Name: DATA
Offset: 0x06
Reset: 0x00
Property: -

The NVMCTRL.DATAL and NVMCTRL.DATAH register pair represents the 16-bit value, NVMCTRL.DATA. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

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

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

Bits 15:0 – DATA[15:0] Data Register
This register is used by the UPDI for fuse write operations.

9.5.7 Address

Name: ADDR
Offset: 0x08
Reset: 0x00
Property: -

The NVMCTRL.ADDRL and NVMCTRL.ADDRH register pair represents the 16-bit value, NVMCTRL.ADDR. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

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 15:0 – ADDR[15:0] Address
The Address register contains the address to the last memory location that has been updated.

10. CLKCTRL - Clock Controller

10.1 Features

- All Clocks and Clock Sources are Automatically Enabled When Requested by Peripherals
- Internal Oscillators:
  - 16/20 MHz Oscillator (OSC20M)
  - 32.768 kHz Ultra Low-Power Oscillator (OSCULP32K)
- External Clock Options:
  - 32.768 kHz Crystal Oscillator (XOSC32K)
  - External clock
- Main Clock Features:
  - Safe run-time switching
  - Prescaler with 1x to 64x division in 12 different settings

10.2 Overview

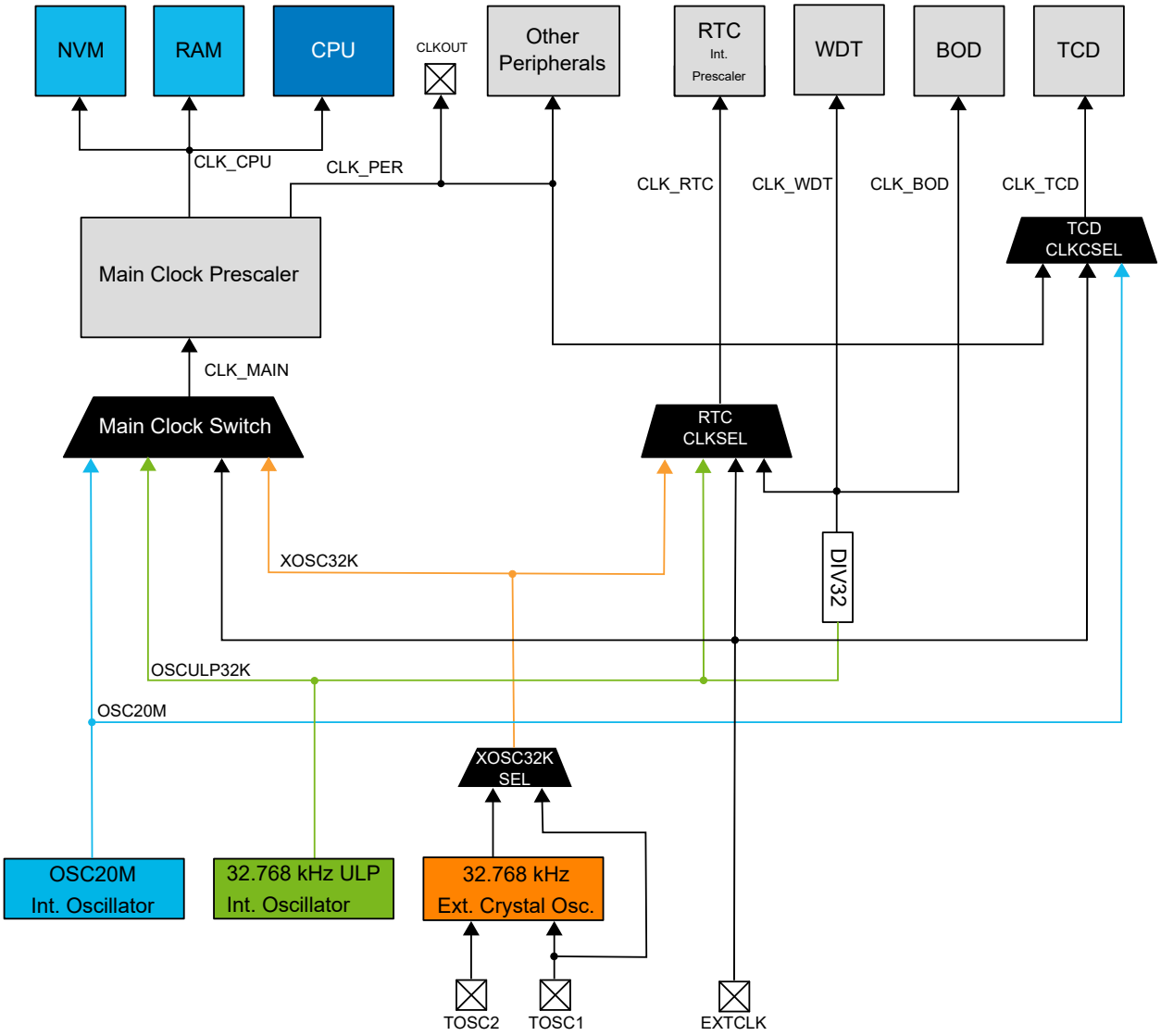
The Clock Controller (CLKCTRL) peripheral controls, distributes and prescales the clock signals from the available oscillators. The CLKCTRL supports internal and external clock sources.

The CLKCTRL is based on an automatic clock request system, implemented in all peripherals on the device. The peripherals will automatically request the clocks needed. If multiple clock sources are available, the request is routed to the correct clock source.

The Main Clock (CLK_MAIN) is used by the CPU, RAM, and the I/O bus. The main clock source can be selected and prescaled. Some peripherals can share the same clock source as the main clock, or run asynchronously to the main clock domain.

10.2.1 Block Diagram - CLKCTRL

Figure 10-1. CLKCTRL Block Diagram



The clock system consists of the Main Clock and other asynchronous clocks:

- Main Clock
  This clock is used by the CPU, RAM, Flash, the I/O bus and all peripherals connected to the I/O bus. It is always running in Active and Idle sleep modes and can be running in Standby sleep mode if requested.

  The main clock CLK_MAIN is prescaled and distributed by the Clock Controller:
  - CLK_CPU is used by the CPU, SRAM and the NVMCTRL peripheral to access the non-volatile memory
  - CLK_PER is used by all peripherals that are not listed under asynchronous clocks
- Clocks running asynchronously to the Main Clock domain:
  - CLK_RTC is used by the RTC/PIT. It will be requested when the RTC/PIT is enabled. The clock source for CLK_RTC must only be changed if the peripheral is disabled.
  - CLK_WDT is used by the WDT. It will be requested when the WDT is enabled.
  - CLK_BOD is used by the BOD. It will be requested when the BOD is enabled in Sampled Mode.
  - CLK_TCD is used by the TCD. It will be requested when the TCD is enabled. The clock source can only be changed if the peripheral is disabled.

The clock source for the Main Clock domain is configured by writing to the Clock Select (CLKSEL) bits in the Main Clock Control A (CLKCTRL.MCLKCTRLA) register. The asynchronous clock sources are configured by registers in the respective peripheral.

10.2.2 Signal Description

Signal	Type	Description
CLKOUT	Digital output	CLK_PER output

10.3 Functional Description

10.3.1 Sleep Mode Operation

When a clock source is not used/requested, it will turn off. It is possible to request a clock source directly by writing a '1' to the Run in Standby (RUNSTDBY) bit in the respective oscillator's Control A (CLKCTRL.[osc]CTRLA) register. This will cause the oscillator to run constantly, except for Power-Down sleep mode. Additionally, when this bit is written to '1', the oscillator start-up time is eliminated when the clock source is requested by a peripheral.

The main clock will always run in Active and Idle sleep mode. In Standby sleep mode, the main clock will only run if any peripheral is requesting it, or the Run in Standby (RUNSTDBY) bit in the respective oscillator's Control A (CLKCTRL.[osc]CTRLA) register is written to '1'.

In Power-Down sleep mode, the main clock will stop after all NVM operations are completed.

10.3.2 Main Clock Selection and Prescaler

All internal oscillators can be used as the main clock source for CLK_MAIN. The main clock source is selectable from software and can be safely changed during normal operation.

Built-in hardware protection prevents unsafe clock switching.

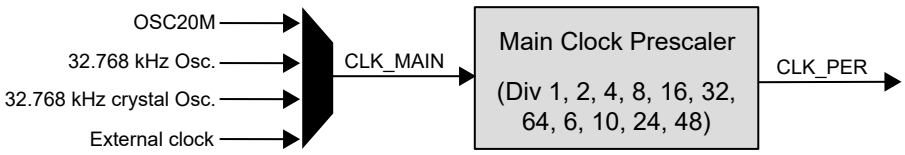
Upon selection of an external clock source, a switch to the chosen clock source will only occur if edges are detected, indicating it is stable. Until a sufficient number of clock edges are detected, the switch will not occur, and it will not be possible to change to another clock source again without executing a Reset.

An ongoing clock source switch is indicated by the System Oscillator Changing (SOSC) flag in the Main Clock Status (CLKCTRL.MCLKSTATUS) register. The stability of the external clock sources is indicated by the respective status (EXTS and XOSC32KS in CLKCTRL.MCLKSTATUS) flags.

⚠ CAUTION If an external clock source fails while used as CLK_MAIN source, only the WDT can provide a mechanism to switch back via System Reset.

CLK_MAIN is fed into a prescaler before it is used by the peripherals (CLK_PER) in the device. The prescaler divides CLK_MAIN by a factor from 1 to 64.

Figure 10-2. Main Clock and Prescaler



The Main Clock and Prescaler configuration (CLKCTRL.MCLKCTRLA, CLKCTRL.MCLKCTRLB) registers are protected by the Configuration Change Protection Mechanism, employing a timed write procedure for changing these registers.

10.3.3 Main Clock After Reset

After any Reset, CLK_MAIN is provided by the 16/20 MHz Oscillator (OSC20M) and with a prescaler division factor of 6. Since the actual frequency of the OSC20M is determined by the Frequency Select (FREQSEL) bits of the Oscillator Configuration (FUSE.OSCCFG) fuse, these frequencies are possible after Reset:

Table 10-1. Peripheral Clock Frequencies After Reset

CLK_MAIN as Per FREQSEL in FUSE.OSCCFG	Resulting CLK_PER
16 MHz	2.66 MHz
20 MHz	3.3 MHz

See the OSC20M description for further details.

10.3.4 Clock Sources

All internal clock sources are enabled automatically when they are requested by a peripheral. The crystal oscillator, based on an external crystal, must be enabled by writing a '1' to the ENABLE bit in the 32.768 kHz Crystal Oscillator Control A (CLKCTRL.XOSC32KCTRLA) register before it can serve as a clock source.

The respective Oscillator Status bits in the Main Clock Status (CLKCTRL.MCLKSTATUS) register indicate whether the clock source is running and stable.

10.3.4.1 Internal Oscillators

The internal oscillators do not require any external components to run.

10.3.4.1.1 16/20 MHz Oscillator (OSC20M)

This oscillator can operate at multiple frequencies, selected by the value of the Frequency Select (FREQSEL) bits in the Oscillator Configuration (FUSE.OSCCFG) fuse. The center frequencies are:

- 16 MHz
- 20 MHz

After a System Reset, FUSE.OSCCFG determines the initial frequency of CLK_MAIN.

During Reset, the calibration values for the OSC20M are loaded from fuses. There are two different Calibration bit fields:

- The Calibration (CAL20M) bit field in the Calibration A (CLKCTRL.OSC20MCALIBA) register enables calibration around the current center frequency
- The Oscillator Temperature Coefficient Calibration (TEMPCAL20M) bit field in the Calibration B (CLKCTRL.OSC20MCALIBB) register enables adjustment of the slope of the temperature drift compensation

For applications requiring a more fine-tuned frequency setting than the oscillator calibration provides, factory-stored frequency error after calibrations are available.

The oscillator calibration can be locked by the Oscillator Lock (OSCLOCK) Fuse (FUSE.OSCCFG). When this fuse is '1', it is not possible to change the calibration. The calibration is locked if this oscillator is used as the main clock source and the Lock Enable (LOCKEN) bit in the Control B (CLKCTRL.OSC20MCALIBB) register is '1'.

The Calibration bits are protected by the Configuration Change Protection Mechanism, requiring a timed write procedure for changing the main clock and prescaler settings.

The start-up time of this oscillator is the analog start-up time plus four oscillator cycles. Refer to the Electrical Characteristics section for the start-up time.

When changing the oscillator calibration value, the frequency may overshoot. If the oscillator is used as the main clock (CLK_MAIN), it is recommended to change the main clock prescaler so that the main clock frequency does not exceed ¼ of the maximum operation main clock frequency as described in the General Operating Ratings section. The system clock prescaler can be changed back after the oscillator calibration value has been updated.

OSC20M Stored Frequency Error Compensation

This oscillator can operate at multiple frequencies, selected by the value of the Frequency Select (FREQSEL) bits in the Oscillator Configuration (FUSE.OSCCFG) fuse at Reset. As previously mentioned, appropriate calibration values are loaded to adjust to center frequency (OSC20M) and temperature drift compensation (TEMPCAL20M), meeting

the specifications defined in the internal oscillator characteristics. For applications requiring a wider operating range, the relative factory stored frequency error after calibrations can be used. The four errors are measured with different settings and are available in the signature row as signed byte values.

- SIGROW.OSC16ERR3V is the frequency error from 16 MHz measured at 3V
- SIGROW.OSC16ERR5V is the frequency error from 16 MHz measured at 5V
- SIGROW.OSC20ERR3V is the frequency error from 20 MHz measured at 3V
- SIGROW.OSC20ERR5V is the frequency error from 20 MHz measured at 5V

The error is stored as a compressed **Q1.10** fixed point 8-bit value, not to lose resolution, where the MSb is the sign bit, and the seven LSbs are the lower bits of the **Q1.10**.

$$\mathrm{BAUD}_{actual} = \left(\mathrm{BAUD}_{ideal} + \frac{\mathrm{BAUD}_{ideal} * SigRowError}{1024}\right)$$

The minimum legal BAUD register value is `0x40`, the target BAUD register value must, therefore, not be lower than `0x4A` to ensure that the compensated BAUD value stays within the legal range, even for parts with negative compensation values. The example code below demonstrates how to apply this value for more accurate USART baud rate:

```
  #include <assert.h>
  /* Baud rate compensated with factory stored frequency error */
  /* Asynchronous communication without Auto-baud (Sync Field) */
  /* 16MHz Clock, 3V and 600 BAUD              */

  int8_t  sigrow_val    = SIGROW.OSC16ERR3V;   // read signed error
  int32_t baud_reg_val  = 600;                 // ideal BAUD register value

  assert (baud_reg_val >= 0x4A);                        // Verify legal min BAUD register
value with max neg comp
  baud_reg_val *= (1024 + sigrow_val);       // sum resolution + error
  baud_reg_val /=  1024;                       // divide by resolution
  USART0.BAUD = (int16_t) baud_reg_val;        // set adjusted baud rate
```

10.3.4.1.2 32.768 kHz Oscillator (OSCULP32K)

The 32.768 kHz oscillator is optimized for Ultra Low-Power (ULP) operation. Power consumption is decreased at the cost of decreased accuracy compared to an external crystal oscillator.

This oscillator provides the 1.024 kHz signal for the Real-Time Counter (RTC), the Watchdog Timer (WDT), and the Brown-out Detector (BOD).

The start-up time of this oscillator is the oscillator start-up time plus four oscillator cycles. Refer to the Electrical Characteristics section for the start-up time.

10.3.4.2 External Clock Sources

These external clock sources are available:

- External Clock from pin EXTCLK
- 32.768 kHz Crystal Oscillator on pins TOSC1 and TOSC2
- 32.768 kHz External Clock on pin TOSC1

10.3.4.2.1 External Clock (EXTCLK)

The EXTCLK is taken directly from the pin. This GPIO pin is automatically configured for EXTCLK if any peripheral is requesting this clock.

This clock source has a start-up time of two cycles when first requested.

10.3.4.2.2 32.768 kHz Crystal Oscillator (XOSC32K)

This oscillator supports two input options: Either a crystal is connected to the pins TOSC1 and TOSC2, or an external clock running at 32.768 kHz is connected to TOSC1. The input option must be configured by writing the Source Select (SEL) bit in the XOSC32K Control A (CLKCTRL.XOSC32KCTRLA) register.

The XOSC32K is enabled by writing a '`1`' to its ENABLE bit in CLKCTRL.XOSC32KCTRLA. When enabled, the configuration of the GPIO pins used by the XOSC32K is overridden as TOSC1, TOSC2 pins. The ENABLE bit needs to be set for the oscillator to start running when requested.

The start-up time of a given crystal oscillator can be accommodated by writing to the Crystal Start-up Time (CSUT) bits in CLKCTRL.XOSC32KCTRLA.

When XOSC32K is configured to use an external clock on TOSC1, the start-up time is fixed to two cycles.

10.3.5 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 10-2. CLKCTRL - Registers Under Configuration Change Protection

Register	Key
CLKCTRL.MCLKCTRLB	IOREG
CLKCTRL.MCLKLOCK	IOREG
CLKCTRL.XOSC32KCTRLA	IOREG
CLKCTRL.MCLKCTRLA	IOREG
CLKCTRL.OSC20MCTRLA	IOREG
CLKCTRL.OSC20MCALIBA	IOREG
CLKCTRL.OSC20MCALIBB	IOREG
CLKCTRL.OSC32KCTRLA	IOREG

10.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	MCLKCTRLA	7:0	CLKOUT						CLKSEL[1:0]	
0x01	MCLKCTRLB	7:0				PDIV[3:0]				PEN
0x02	MCLKLOCK	7:0								LOCKEN
0x03	MCLKSTATUS	7:0	EXTS	XOSC32KS	OSC32KS	OSC20MS				SOSC
0x04 ... 0x0F	Reserved									
0x10	OSC20MCTRLA	7:0							RUNSTDBY	
0x11	OSC20MCALIBA	7:0			CAL20M[5:0]					
0x12	OSC20MCALIBB	7:0	LOCK				TEMPCAL20M[3:0]			
0x13 ... 0x17	Reserved									
0x18	OSC32KCTRLA	7:0							RUNSTDBY	
0x19 ... 0x1B	Reserved									
0x1C	XOSC32KCTRLA	7:0			CSUT[1:0]			SEL	RUNSTDBY	ENABLE

10.5 Register Description

10.5.1 Main Clock Control A

Name: MCLKCTRLA
Offset: 0x00
Reset: 0x00
Property: Configuration Change Protection

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

Bit 7 – CLKOUT System Clock Out
When this bit is written to '1', the system clock is output to the CLKOUT pin.
When the device is in a sleep mode, there is no clock output unless a peripheral is using the system clock.

Bits 1:0 – CLKSEL[1:0] Clock Select
This bit field selects the source for the Main Clock (CLK_MAIN).

Value	Name	Description
0x0	OSC20M	16/20 MHz internal oscillator
0x1	OSCULP32K	32.768 kHz internal ultra low-power oscillator
0x2	XOSC32K	32.768 kHz external crystal oscillator
0x3	EXTCLK	External clock

10.5.2 Main Clock Control B

Name: MCLKCTRLB
Offset: 0x01
Reset: 0x11
Property: Configuration Change Protection

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

Bits 4:1 – PDIV[3:0] Prescaler Division
If the Prescaler Enable (PEN) bit is written to '1', this bit field defines the division ratio of the main clock prescaler. This bit field can be written during run-time to vary the clock frequency of the system to suit the application requirements.
The user software must ensure a correct configuration of input frequency (CLK_MAIN) and prescaler settings, such that the resulting frequency of CLK_PER never exceeds the allowed maximum (see Electrical Characteristics).

Value	Description
Value	Division
0x0	2
0x1	4
0x2	8
0x3	16
0x4	32
0x5	64
0x8	6
0x9	10
0xA	12
0xB	24
0xC	48
other	Reserved

Bit 0 – PEN Prescaler Enable
This bit must be written '1' to enable the prescaler. When enabled, the division ratio is selected by the PDIV bit field. When this bit is written to '0', the main clock will pass through undivided (CLK_PER=CLK_MAIN), regardless of the value of PDIV.

10.5.3 Main Clock Lock

Name: MCLKLOCK
Offset: 0x02
Reset: Based on OSCLOCK in FUSE.OSCCFG
Property: Configuration Change Protection

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

Bit 0 – LOCKEN Lock Enable
Writing this bit to '1' will lock the CLKCTRL.MCLKCTRLA and CLKCTRL.MCLKCTRLB registers, and, if applicable, the calibration settings for the current main clock source from further software updates. Once locked, the CLKCTRL.MCLKLOCK registers cannot be accessed until the next hardware Reset.
This protects the CLKCTRL.MCLKCTRLA and CLKCTRL.MCLKCTRLB registers and calibration settings for the main clock source from unintentional modification by software.
At Reset, the LOCKEN bit is loaded based on the OSCLOCK bit in FUSE.OSCCFG.

10.5.4 Main Clock Status

Name: MCLKSTATUS
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	EXTS	XOSC32KS	OSC32KS	OSC20MS				SOSC
Access	R	R	R	R				R
Reset	0	0	0	0				0

Bit 7 – EXTS External Clock Status

Value	Description
0	EXTCLK has not started
1	EXTCLK has started

Bit 6 – XOSC32KS XOSC32K Status
The Status bit will only be available if the source is requested as the main clock or by another module. If the oscillator RUNSTDBY bit is set, but the oscillator is unused/not requested, this bit will be '0'.

Value	Description
0	XOSC32K is not stable
1	XOSC32K is stable

Bit 5 – OSC32KS OSCULP32K Status
The Status bit will only be available if the source is requested as the main clock or by another module. If the oscillator RUNSTDBY bit is set, but the oscillator is unused/not requested, this bit will be '0'.

Value	Description
0	OSCULP32K is not stable
1	OSCULP32K is stable

Bit 4 – OSC20MS OSC20M Status
The Status bit will only be available if the source is requested as the main clock or by another module. If the oscillator RUNSTDBY bit is set, but the oscillator is unused/not requested, this bit will be '0'.

Value	Description
0	OSC20M is not stable
1	OSC20M is stable

Bit 0 – SOSC Main Clock Oscillator Changing

Value	Description
0	The clock source for CLK_MAIN is not undergoing a switch
1	The clock source for CLK_MAIN is undergoing a switch and will change as soon as the new source is stable

10.5.5 16/20 MHz Oscillator Control A

Name: OSC20MCTRLA
Offset: 0x10
Reset: 0x00
Property: Configuration Change Protection

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

Bit 1 – RUNSTDBY Run in Standby
This bit forces the oscillator ON in all modes, even when unused by the system. In Standby sleep mode, this can be used to ensure immediate wake-up and not waiting for the oscillator start-up time.
When not requested by peripherals, no oscillator output is provided.
It takes four oscillator cycles to open the clock gate after a request, but the oscillator analog start-up time will be removed when this bit is set.

10.5.6 16/20 MHz Oscillator Calibration A

Name: OSC20MCALIBA
Offset: 0x11
Reset: Based on FREQSEL in FUSE.OSCCFG
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
			CAL20M[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			x	x	x	x	x	x

Bits 5:0 – CAL20M[5:0] Calibration
This bit field changes the frequency around the current center frequency of the OSC20M for fine-tuning.
At Reset, the factory calibrated values are loaded based on the FREQSEL bits in FUSE.OSCCFG.

10.5.7 16/20 MHz Oscillator Calibration B

Name: OSC20MCALIBB
Offset: 0x12
Reset: Based on FUSE.OSCCFG
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
	LOCK				TEMPCAL20M[3:0]			
Access	R				R/W	R/W	R/W	R/W
Reset	x				x	x	x	x

Bit 7 – LOCK Oscillator Calibration Locked by Fuse
When this bit is set, the calibration settings in CLKCTRL.OSC20MCALIBA and CLKCTRL.OSC20MCALIBB cannot be changed.
At Reset, the value is loaded from the OSCLOCK bit in the Oscillator Configuration (FUSE.OSCCFG) fuse.

Bits 3:0 – TEMPCAL20M[3:0] Oscillator Temperature Coefficient Calibration
This bit field tunes the slope of the temperature compensation.
At Reset, the factory calibrated values are loaded based on the FREQSEL bits in FUSE.OSCCFG.

10.5.8 32.768 kHz Oscillator Control A

Name: OSC32KCTRLA
Offset: 0x18
Reset: 0x00
Property: Configuration Change Protection

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

Bit 1 – RUNSTDBY Run in Standby
This bit forces the oscillator ON in all modes, even when unused by the system. In Standby sleep mode, this can be used to ensure immediate wake-up and not waiting for the oscillator start-up time.
When not requested by peripherals, no oscillator output is provided.
It takes four oscillator cycles to open the clock gate after a request, but the oscillator analog start-up time will be removed when this bit is set.

10.5.9 32.768 kHz Crystal Oscillator Control A

Name: XOSC32KCTRLA
Offset: 0x1C
Reset: 0x00
Property: Configuration Change Protection

The SEL and CSUT bits cannot be changed as long as the ENABLE bit is set, or the XOSC32K Stable (XOSC32KS) bit in CLKCTRL.MCLKSTATUS is high.
To change settings safely, write a '0' to the ENABLE bit and wait until XOSC32KS is '0' before re-enabling the XOSC32K with new settings.

Bit	7	6	5	4	3	2	1	0
			CSUT[1:0]			SEL	RUNSTDBY	ENABLE
Access			R/W	R/W		R/W	R/W	R/W
Reset			0	0		0	0	0

Bits 5:4 – CSUT[1:0] Crystal Start-Up Time
This bit field selects the start-up time for the XOSC32K. It is write-protected when the oscillator is enabled (ENABLE=1).
If SEL=1, the start-up time will not be applied.

Value	Name	Description
0x0	1K	1k cycles
0x1	16K	16k cycles
0x2	32K	32k cycles
0x3	64K	64k cycles

Bit 2 – SEL Source Select
This bit selects the external source type. It is write-protected when the oscillator is enabled (ENABLE=1).

Value	Description
0	External crystal
1	External clock on TOSC1 pin

Bit 1 – RUNSTDBY Run in Standby
Writing this bit to '1' starts the crystal oscillator and forces the oscillator ON in all modes, even when unused by the system if the ENABLE bit is set. In Standby sleep mode, this can be used to ensure immediate wake-up and not waiting for oscillator start-up time. When this bit is '0', the crystal oscillator is only running when requested, and the ENABLE bit is set.
The output of XOSC32K is not sent to other peripherals unless it is requested by one or more peripherals.
When the RUNSTDBY bit is set, there will only be a delay of two to three crystal oscillator cycles after a request until the oscillator output is received, if the initial crystal start-up time has already completed.
Depending on the RUNSTBY bit, the oscillator will be turned ON all the time if the device is in Active, Idle, or Standby sleep mode, or only be enabled when requested.
This bit is I/O protected to prevent any unintentional enabling of the oscillator.

Bit 0 – ENABLE Enable
When this bit is written to '1', the configuration of the respective input pins is overridden to TOSC1 and TOSC2. Also, the Source Select (SEL) and Crystal Start-Up Time (CSUT) bits become read-only.
This bit is I/O protected to prevent any unintentional enabling of the oscillator.

11. SLPCTRL - Sleep Controller

11.1 Features

- Power Management for Adjusting Power Consumption and Functions
- Three Sleep Modes:
  - Idle
  - Standby
  - Power-Down
- Configurable Standby Mode where Peripherals Can Be Configured as ON or OFF

11.2 Overview

Sleep modes are used to shut down peripherals and clock domains in the device in order to save power. The Sleep Controller (SLPCTRL) controls and handles the transitions between Active and sleep modes.

There are four modes available: One Active mode in which software is executed, and three sleep modes. The available sleep modes are Idle, Standby and Power-Down.

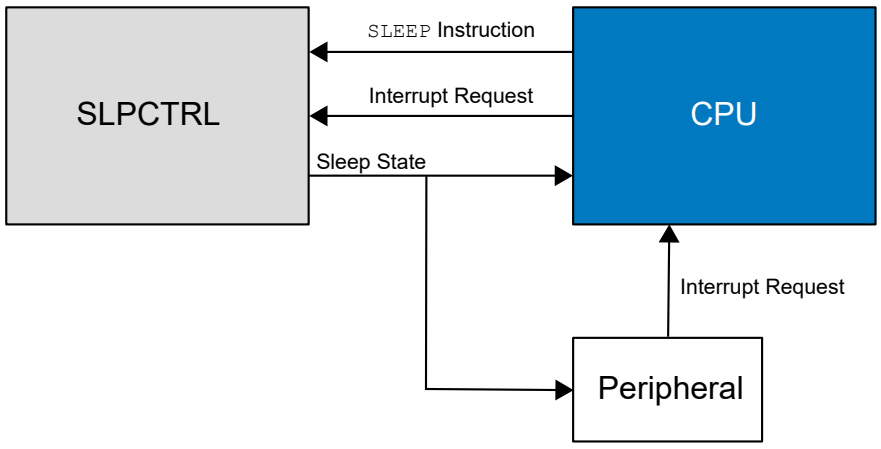
All sleep modes are available and can be entered from the Active mode. In Active mode, the CPU is executing application code. When the device enters sleep mode, the program execution is stopped. The application code decides which sleep mode to enter and when.

Interrupts are used to wake the device from sleep. The available interrupt wake-up sources depend on the configured sleep mode. When an interrupt occurs, the device will wake up and execute the Interrupt Service Routine before continuing normal program execution from the first instruction after the `SLEEP` instruction. Any Reset will take the device out of sleep mode.

The content of the register file, SRAM and registers, is kept during sleep. If a Reset occurs during sleep, the device will reset, start and execute from the Reset vector.

11.2.1 Block Diagram

Figure 11-1. Sleep Controller in the System



11.3 Functional Description

11.3.1 Initialization

To put the device into a sleep mode, follow these steps:

1. Configure and enable the interrupts that are able to wake the device from sleep.
   Also, enable global interrupts.

   WARNING If there are no interrupts enabled when going to sleep, the device cannot wake up again. Only a Reset will allow the device to continue operation.

2. Select which sleep mode to enter and enable the Sleep Controller by writing to the Sleep Mode (SMODE) bit field and the Enable (SEN) bit in the Control A (SLPCTRL.CTRLA) register.
   The `SLEEP` instruction must be executed to make the device go to sleep.

11.3.2 Operation

11.3.2.1 Sleep Modes

In addition to Active mode, there are three different sleep modes with decreasing power consumption and functionality.

Idle The CPU stops executing code. No peripherals are disabled, and all interrupt sources can wake the device.

Standby The user can configure peripherals to be enabled or not, using the respective RUNSTBY bit. This means that the power consumption is highly dependent on what functionality is enabled, and thus may vary between the Idle and Power-Down levels.
SleepWalking is available for the ADC module.

Power-Down BOD, WDT, and PIT (a component of the RTC) are active.
The only wake-up sources are the pin change interrupt, PIT, VLM, TWI address match, and CCL.

Table 11-1. Sleep Mode Activity Overview

Group	Peripheral		Active in Sleep Mode		
		Clock	Idle	Standby	Power-Down
Active Clock Domain	CPU	CLK_CPU			
	RTC	CLK_RTC	X	X(1)	X(2)
	ADCn/PTC	CLK_PER	X	X(1)	
	TCBn	CLK_PER	X	X(1)	
	BOD	CLK_BOR	X	X	X
	WDT	CLK_WDT	X	X	X
	All other peripherals	CLK_PER	X		
Oscillators	Main clock source		X	X(1)	
	RTC clock source		X	X(1)	X(2)
	WDT oscillator		X	X	X

.....continued					
Group	Peripheral		Active in Sleep Mode		
		Clock	Idle	Standby	Power-Down
Wake-Up Sources	PORT Pin interrupt		X	X	X
	TWI Address Match interrupt		X	X	X
	USART Start-of-Frame interrupts		X	X(1)	
	ADC/PTC interrupts		X	X(1)	
	RTC interrupts		X	X(1)	X(2)
	TCBn Capture interrupt		X	X(1)	
	All other interrupts		X		

Note:

1. The RUNSTBY bit of the corresponding peripheral must be set to enter the Active state.
2. Only the PIT is available in the Power-Down sleep mode.

11.3.2.2 Wake-up Time

The normal wake-up time for the device is six main clock cycles (CLK_PER), plus the time it takes to start the main clock source:

- In Idle sleep mode, the main clock source is kept running to eliminate additional wake-up time.
- In Standby sleep mode, the main clock might be running depending on the peripheral configuration.
- In Power-Down sleep mode, only the ULP 32.768 kHz oscillator and the RTC clock may be running if it is used by the BOD or WDT. All other clock sources will be OFF.

Table 11-2. Sleep Modes and Start-up Time

Sleep Mode	Start-up Time
IDLE	6 CLK
Standby	6 CLK + OSC start-up
Power-Down	6 CLK + OSC start-up

The start-up time for the different clock sources is described in the Clock Controller (CLKCTRL) section.

In addition to the normal wake-up time, it is possible to make the device wait until the BOD is ready before executing code. This is done by writing 0x3 to the BOD Operation mode in Active and Idle bits (ACTIVE) in the BOD Configuration fuse (FUSE.BODCFG). If the BOD is ready before the normal wake-up time, the total wake-up time will be the same. If the BOD takes longer than the normal wake-up time, the wake-up time will be extended until the BOD is ready. This ensures correct supply voltage whenever code is executed.

11.3.3 Debug Operation

During run-time debugging, this peripheral will continue normal operation. The SLPCTRL is only affected by a break in the debug operation: If the SLPCTRL is in a sleep mode when a break occurs, the device will wake up, and the SLPCTRL will go to Active mode, even if there are no pending interrupt requests.

If the peripheral is configured to require periodic service by the CPU through interrupts or similar, improper operation or data loss may result during halted debugging.

11.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0						SMODE[1:0]		SEN

11.5 Register Description

11.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

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

Bits 2:1 – SMODE[1:0] Sleep Mode
Writing these bits selects which sleep mode to enter when the Sleep Enable (SEN) bit is written to '1' and the SLEEP instruction is executed.

Value	Name	Description
0x0	IDLE	Idle sleep mode enabled
0x1	STANDBY	Standby sleep mode enabled
0x2	PDOWN	Power-Down sleep mode enabled
other	-	Reserved

Bit 0 – SEN Sleep Enable
This bit must be written to '1' before the SLEEP instruction is executed to make the MCU enter the selected Sleep mode.

12. RSTCTRL - Reset Controller

12.1 Features

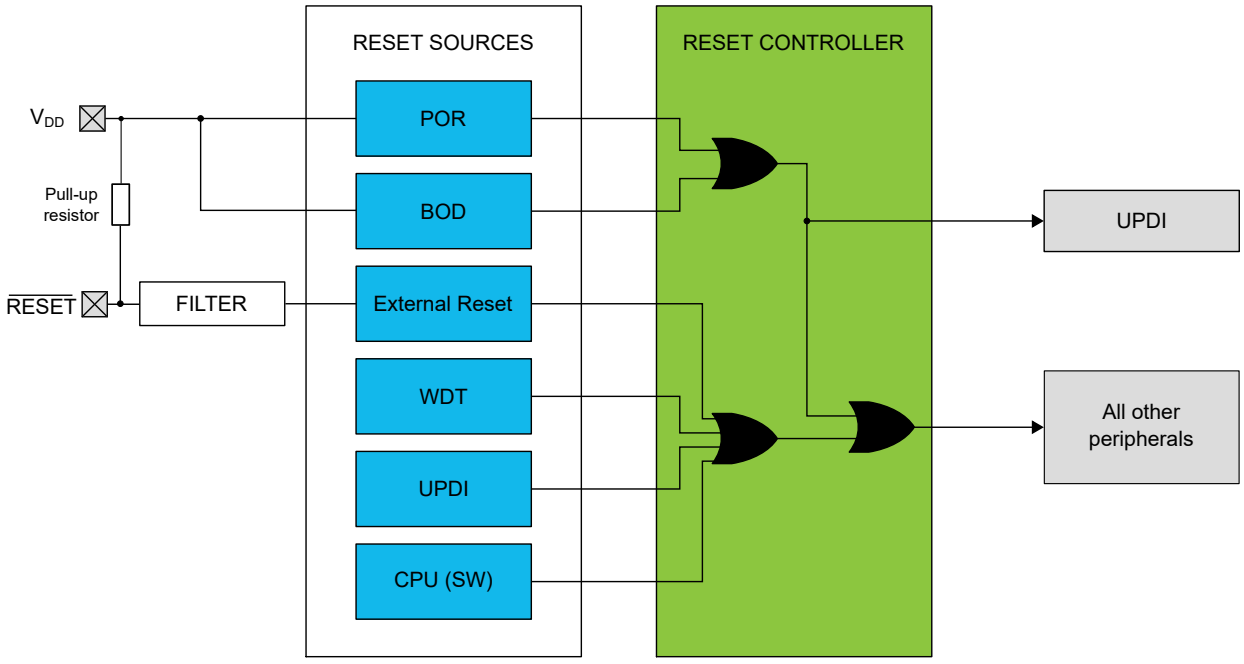
- Returns the Device to an Initial State after a Reset
- Identifies the Previous Reset Source
- Power Supply Reset Sources:
  - Power-on Reset (POR)
  - Brown-out Detector (BOD) Reset
- User Reset Sources:
  - External Reset ($\overline{\text{RESET}}$)
  - Watchdog Timer (WDT) Reset
  - Software Reset (SWRST)
  - Universal Program Debug Interface (UPDI) Reset

12.2 Overview

The Reset Controller (RSTCTRL) manages the Reset of the device. It issues a device Reset, sets the device to its initial state, and allows the Reset source to be identified by software.

12.2.1 Block Diagram

Figure 12-1. Reset System Overview



12.2.2 Signal Description

Signal	Description	Type
$\overline{\text{RESET}}$	External Reset (active-low)	Digital input

12.3 Functional Description

12.3.1 Initialization

The RSTCTRL is always enabled, but some of the Reset sources must be enabled individually (either by Fuses or by software) before they can request a Reset.

After a Reset from any source, the registers in the device with automatic loading from the Fuses or from the Signature Row are updated.

12.3.2 Operation

12.3.2.1 Reset Sources

After any Reset, the source that caused the Reset is found in the Reset Flag (RSTCTRL.RSTFR) register. The user can identify the previous Reset source by reading this register in the software application.

There are two types of Resets based on the source:

- Power Supply Reset Sources:
  - Power-on Reset (POR)
  - Brown-out Detector (BOD) Reset
- User Reset Sources:
  - External Reset ($\overline{\text{RESET}}$)
  - Watchdog Timer (WDT) Reset
  - Software Reset (SWRST)
  - Universal Program Debug Interface (UPDI) Reset

12.3.2.1.1 Power-on Reset (POR)

The purpose of the Power-on Reset (POR) is to ensure a safe start-up of logic and memories. The POR will keep the device in Reset until the voltage level is high enough. The POR is generated by an on-chip detection circuit. The POR is always enabled and activated when V_{DD} is below the POR threshold voltage.

Figure 12-2. MCU Start-Up, $\overline{\text{RESET}}$ Tied to V_{DD}

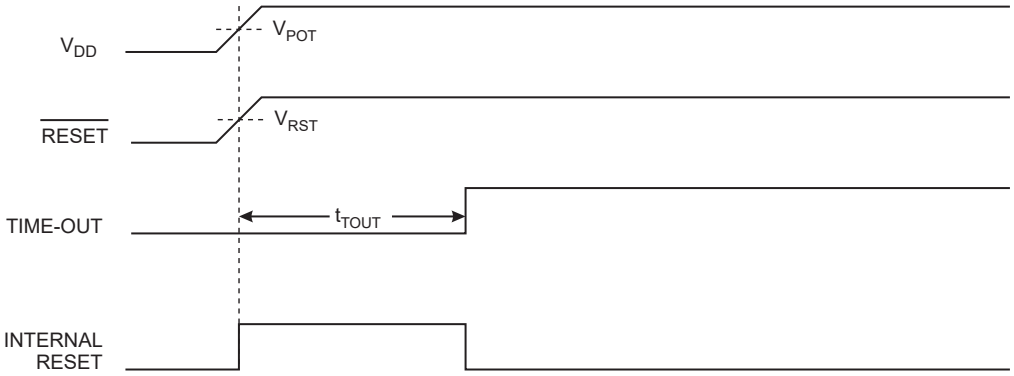
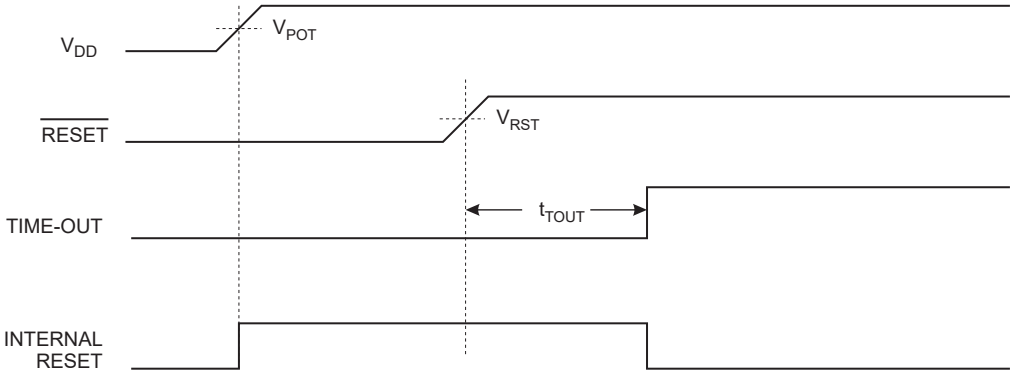


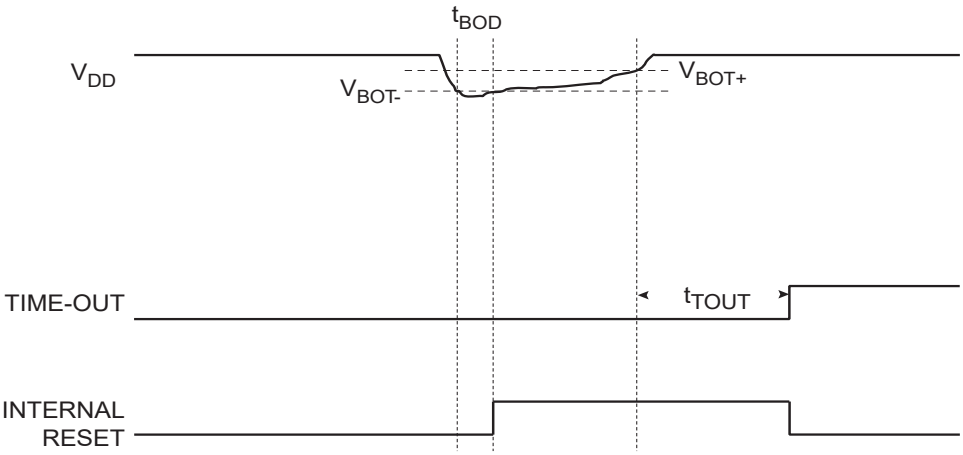
Figure 12-3. MCU Start-Up, $\overline{\text{RESET}}$ Extended Externally



12.3.2.1.2 Brown-out Detector (BOD) Reset

The on-chip Brown-out Detection (BOD) circuit will monitor the V_{DD} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by fuses. If BOD is unused in the application, it is forced to a minimum level in order to ensure a safe operation during internal Reset and chip erase.

Figure 12-4. Brown-out Detection Reset



12.3.2.1.3 Software Reset

The software Reset makes it possible to issue a system Reset from software. The Reset is generated by writing a '1' to the Software Reset Enable bit (SWRE) in the Software Reset register (RSTCTRL.SWRR).

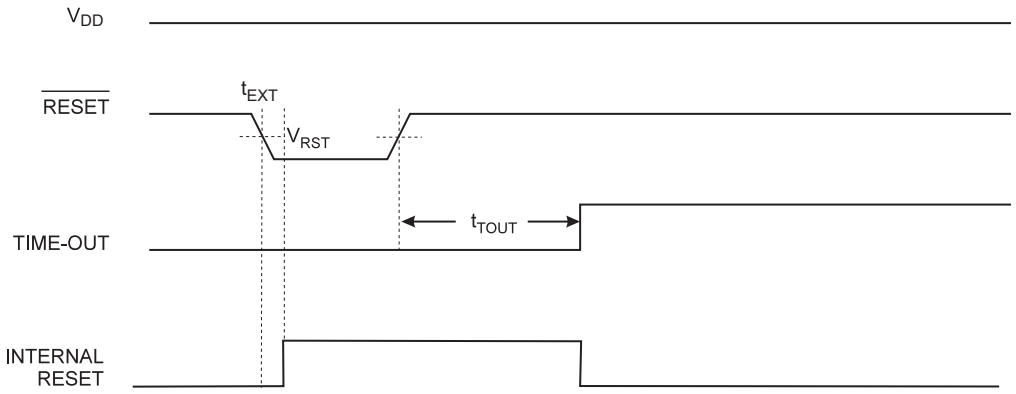
The Reset will take place immediately after the bit is written, and the device will be kept in Reset until the Reset sequence is completed.

12.3.2.1.4 External Reset

The external Reset is enabled by a fuse, see the RSTPINCFG field in FUSE.SYSCFG0.

When enabled, the external Reset requests a Reset as long as the $\overline{\text{RESET}}$ pin is low. The device will stay in Reset until $\overline{\text{RESET}}$ is high again.

Figure 12-5. External Reset Characteristics



12.3.2.1.5 Watchdog Reset

The Watchdog Timer (WDT) is a system function for monitoring correct program operation. If the WDT is not reset from software according to the programmed time-out period, a Watchdog Reset will be issued. See the WDT documentation for further details.

12.3.2.1.6 Universal Program Debug Interface (UPDI) Reset

The Universal Program Debug Interface (UPDI) contains a separate Reset source used to reset the device during external programming and debugging. The Reset source is accessible only from external debuggers and programmers. More details can be found in the UPDI section.

12.3.2.1.7 Domains Affected By Reset

The following logic domains are affected by the various Resets:

Table 12-1. Logic Domains Affected by Various Resets

Reset Type	Fuses are Reloaded	TCD Pin Override Functionality Available	Reset of TCD Pin Override Settings	Reset of BOD Configuration	Reset of UPDI	Reset of Other Volatile Logic
POR	X		X	X	X	X
BOD	X	X			X	X
Software Reset	X	X				X
External Reset	X	X				X
Watchdog Reset	X	X				X
UPDI Reset	X	X				X

12.3.2.2 Reset Time

The Reset time can be split into two parts.

The first part is when any of the Reset sources are active. This part depends on the input to the Reset sources. The external Reset is active as long as the $\overline{\text{RESET}}$ pin is low. The Power-on Reset (POR) and the Brown-out Detector (BOD) are active as long as the supply voltage is below the Reset source threshold.

The second part is when all the Reset sources are released, and an internal Reset initialization of the device is done. This time will be increased with the start-up time given by the Start-Up Time Setting (SUT) bit field in the System Configuration 1 (FUSE.SYSCFG1) fuse. The internal Reset initialization time will also increase if the Cyclic Redundancy Check Memory Scan (CRCSCAN) is configured to run at start-up. This configuration can be changed in the CRC Source (CRCSRC) bit field in the System Configuration 0 (FUSE.SYSCFG0) fuse.

12.3.3 Sleep Mode Operation

The RSTCTRL operates in Active mode and in all sleep modes.

12.3.4 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 12-2. RSTCTRL - Registers Under Configuration Change Protection

Register	Key
RSTCTRL.SWRR	IOREG

12.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	RSTFR	7:0			UPDIRF	SWRF	WDRF	EXTRF	BORF	PORF
0x01	SWRR	7:0								SWRE

12.5 Register Description

12.5.1 Reset Flag Register

Name: RSTFR
Offset: 0x00
Reset: 0xXX
Property: -

All flags are cleared by writing a '1' to them. They are also cleared by a Power-on Reset (POR), with the exception of the Power-on Reset Flag (PORF).

Bit	7	6	5	4	3	2	1	0
			UPDIRF	SWRF	WDRF	EXTRF	BORF	PORF
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			x	x	x	x	x	x

Bit 5 – UPDIRF UPDI Reset Flag
This bit is set if a UPDI Reset occurs.

Bit 4 – SWRF Software Reset Flag
This bit is set if a Software Reset occurs.

Bit 3 – WDRF Watchdog Reset Flag
This bit is set if a Watchdog Reset occurs.

Bit 2 – EXTRF External Reset Flag
This bit is set if an External Reset occurs.

Bit 1 – BORF Brown-out Reset Flag
This bit is set if a Brown-out Reset occurs.

Bit 0 – PORF Power-on Reset Flag
This bit is set if a POR occurs.
After a POR, only the POR flag is set and all the other flags are cleared. No other flags can be set before a full system boot is run after the POR.

12.5.2 Software Reset Register

Name: SWRR
Offset: 0x01
Reset: 0x00
Property: Configuration Change Protection

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

Bit 0 – SWRE Software Reset Enable
When this bit is written to '1', a software Reset will occur.
This bit will always read as '0'.

13. CPUINT - CPU Interrupt Controller

13.1 Features

- Short and Predictable Interrupt Response Time
- Separate Interrupt Configuration and Vector Address for Each Interrupt
- Interrupt Prioritizing by Level and Vector Address
- Non-Maskable Interrupts (NMI) for Critical Functions
- Two Interrupt Priority Levels: 0 (Normal) and 1 (High):
  - One of the interrupt requests can optionally be assigned as a priority level 1 interrupt
  - Optional round robin priority scheme for priority level 0 interrupts
- Interrupt Vectors Optionally Placed in the Application Section or the Boot Loader Section
- Selectable Compact Vector Table (CVT)

13.2 Overview

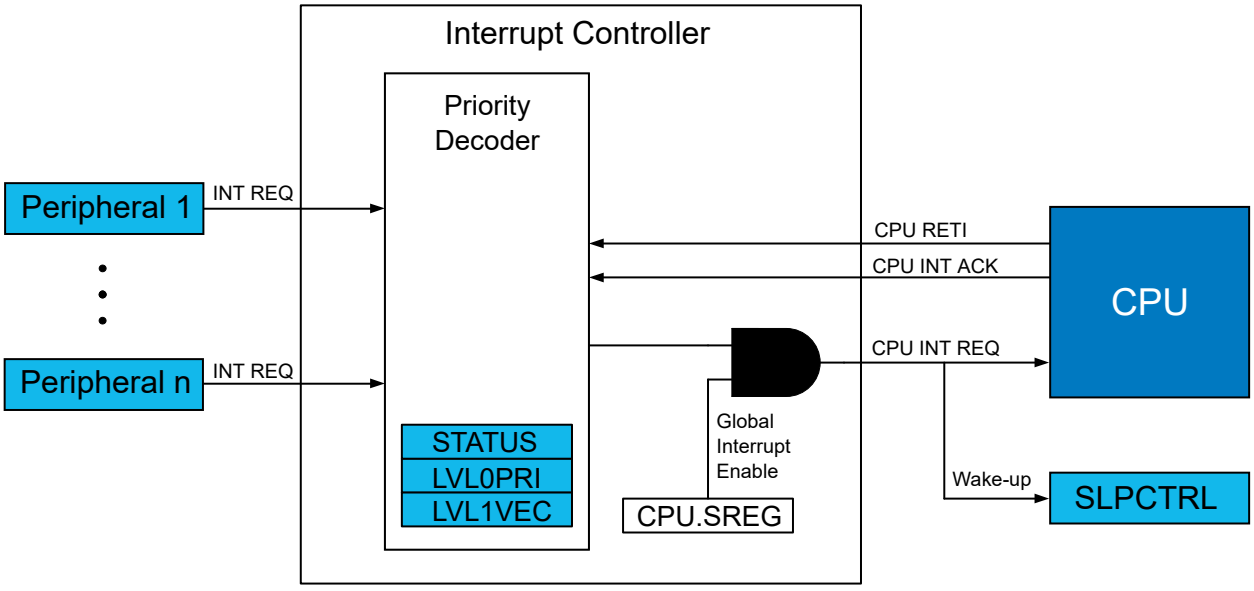
An interrupt request signals a change of state inside a peripheral and can be used to alter the program execution. The peripherals can have one or more interrupts. All interrupts are individually enabled and configured. When an interrupt is enabled and configured, it will generate an interrupt request when the interrupt condition occurs.

The CPU Interrupt Controller (CPUINT) handles and prioritizes the interrupt requests. When an interrupt is enabled and the interrupt condition occurs, the CPUINT will receive the interrupt request. Based on the interrupt's priority level and the priority level of any ongoing interrupt, the interrupt request is either acknowledged or kept pending until it has priority. After returning from the interrupt handler, the program execution continues from where it was before the interrupt occurred, and any pending interrupts are served after one instruction is executed.

The CPUINT offers NMI for critical functions, one selectable high-priority interrupt and an optional round robin scheduling scheme for normal-priority interrupts. The round robin scheduling ensures that all interrupts are serviced within a certain amount of time.

13.2.1 Block Diagram

Figure 13-1. CPUINT Block Diagram



13.3 Functional Description

13.3.1 Initialization

An interrupt must be initialized in the following order:

1. Configure the CPUINT if the default configuration is not adequate (optional):
   - Vector handling is configured by writing to the respective bits (IVSEL and CVT) in the Control A (CPUINT.CTRLA) register.
   - Vector prioritizing by round robin is enabled by writing a '1' to the Round Robin Priority Enable (LVL0RR) bit in CPUINT.CTRLA.
   - Select the Priority Level 1 vector by writing the interrupt vector number to the Interrupt Vector with Priority Level 1 (CPUINT.LVL1VEC) register.
2. Configure the interrupt conditions within the peripheral and enable the peripheral's interrupt.
3. Enable interrupts globally by writing a '1' to the Global Interrupt Enable (I) bit in the CPU Status (CPU.SREG) register.

13.3.2 Operation

13.3.2.1 Enabling, Disabling and Resetting

The global enabling of interrupts is done by writing a '1' to the Global Interrupt Enable (I) bit in the CPU Status (CPU.SREG) register. To disable interrupts globally, write a '0' to the I bit in CPU.SREG.

The desired interrupt lines must also be enabled in the respective peripheral by writing to the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

The interrupt flags are not automatically cleared after the interrupt is executed. The respective INTFLAGS register descriptions provide information on how to clear specific flags.

13.3.2.2 Interrupt Vector Locations

The interrupt vector placement is dependent on the value of the Interrupt Vector Select (IVSEL) bit in the Control A (CPUINT.CTRLA) register. Refer to the IVSEL description in CPUINT.CTRLA for the possible locations.

If the program never enables an interrupt source, the interrupt vectors are not used, and the regular program code can be placed at these locations.

13.3.2.3 Interrupt Response Time

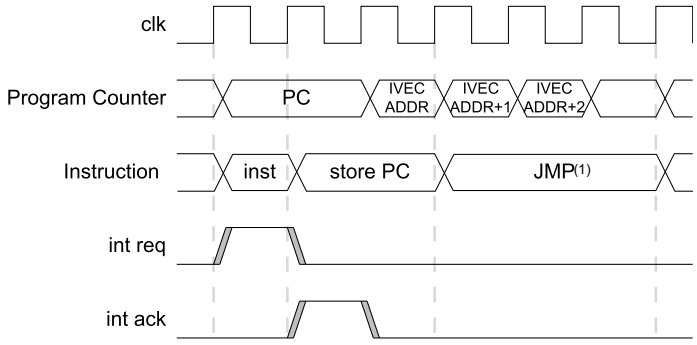
The minimum interrupt response time is represented in the following table.

Table 13-1. Minimum Interrupt Response Time

	Flash Size > 8 KB	Flash Size ≤ 8 KB
Finish ongoing instruction	One cycle	One cycle
Store PC to stack	Two cycles	Two cycles
Jump to interrupt handler	Three cycles (jmp)	Two cycles (rjmp)

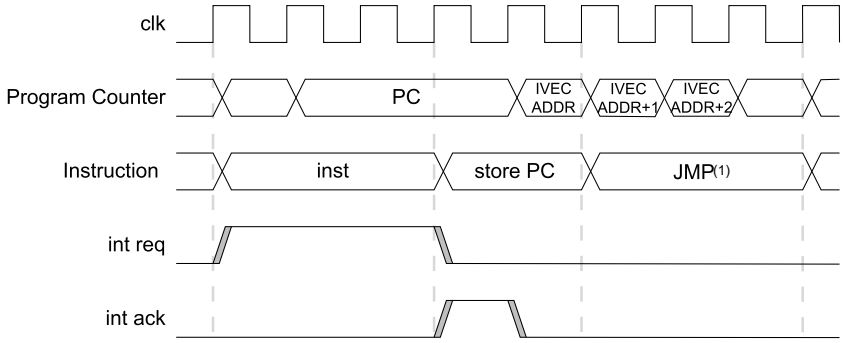
After the Program Counter is pushed on the stack, the program vector for the interrupt is executed. See the following figure.

Figure 13-2. Interrupt Execution of Single-Cycle Instruction



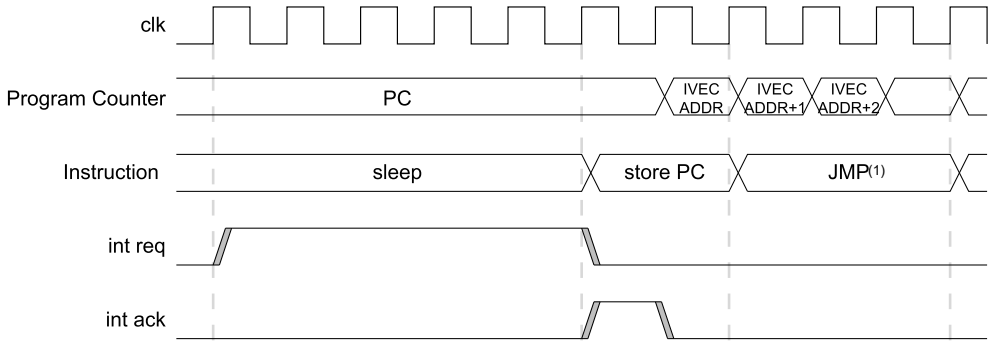
If an interrupt occurs during the execution of a multi-cycle instruction, the instruction is completed before the interrupt is served, as shown in the following figure.

Figure 13-3. Interrupt Execution of Multi-Cycle Instruction



If an interrupt occurs when the device is in a sleep mode, the interrupt execution response time is increased by five clock cycles, as shown in the figure below. Also, the response time is increased by the start-up time from the selected sleep mode.

Figure 13-4. Interrupt Execution From Sleep



A return from an interrupt handling routine takes four to five clock cycles, depending on the size of the Program Counter. During these clock cycles, the Program Counter is popped from the stack, and the Stack Pointer is incremented.

Note:

1. Devices with 8 KB of Flash or less use `RJMP` instead of `JMP`, which takes only two clock cycles.

13.3.2.4 Interrupt Priority

All interrupt vectors are assigned to one of three possible priority levels, as shown in the table below. An interrupt request from a high-priority source will interrupt any ongoing interrupt handler from a normal-priority source. When returning from the high-priority interrupt handler, the execution of the normal-priority interrupt handler will resume.

Table 13-2. Interrupt Priority Levels

Priority	Level	Source
Highest	Non-Maskable Interrupt	Device-dependent and statically assigned
...	Level 1 (high priority)	One vector is optionally user selectable as level 1
Lowest	Level 0 (normal priority)	The remaining interrupt vectors

13.3.2.4.1 Non-Maskable Interrupts

A Non-Maskable Interrupt (NMI) will be executed regardless of the setting of the I bit in CPU.SREG. An NMI will never change the I bit. No other interrupt can interrupt an NMI handler. If more than one NMI is requested at the same time, the priority is static according to the interrupt vector address, where the lowest address has the highest priority.

Which interrupts are non-maskable is device-dependent and not subject to configuration. Non-maskable interrupts must be enabled before they can be used. Refer to the interrupt vector mapping of the device for available NMI lines.

13.3.2.4.2 High-Priority Interrupt

It is possible to assign one interrupt request to level 1 (high priority) by writing its interrupt vector number to the CPUINT.LVL1VEC register. This interrupt request will have a higher priority than the other (normal priority) interrupt requests. The priority level 1 interrupts will interrupt the level 0 interrupt handlers.

13.3.2.4.3 Normal-Priority Interrupts

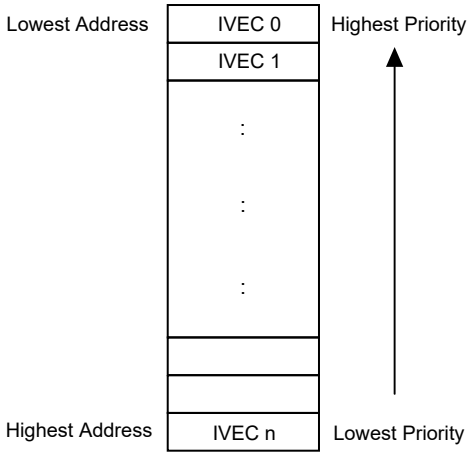
All interrupt vectors other than NMI are assigned to priority level 0 (normal) by default. The user may override this by assigning one of these vectors as a high-priority vector. The device will have many normal-priority vectors, and some of these may be pending at the same time. Two different scheduling schemes are available to choose which of the pending normal-priority interrupts to service first: Static or round robin.

IVEC is the interrupt vector mapping, as listed in the *Peripherals and Architecture* chapter. The following sections use IVEC to explain the scheduling schemes. IVEC0 is the Reset vector, IVEC1 is the NMI vector, and so on. In a vector table with n+1 elements, the vector with the highest vector number is denoted IVECn. Reset, non-maskable interrupts and high-level interrupts are included in the IVEC map, but will always be prioritized over the normal-priority interrupts.

Static Scheduling

If several level 0 interrupt requests are pending at the same time, the one with the highest priority is scheduled for execution first. The following figure illustrates the default configuration, where the interrupt vector with the lowest address has the highest priority.

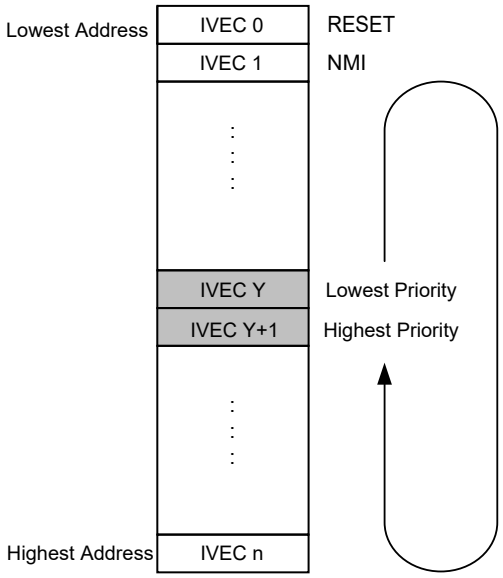
Figure 13-5. Default Static Scheduling



Modified Static Scheduling

The default priority can be changed by writing a vector number to the CPUINT.LVL0PRI register. This vector number will be assigned the lowest priority. The next interrupt vector in the IVEC will have the highest priority among the LVL0 interrupts, as shown in the following figure.

Figure 13-6. Static Scheduling when CPUINT.LVL0PRI is Different From Zero



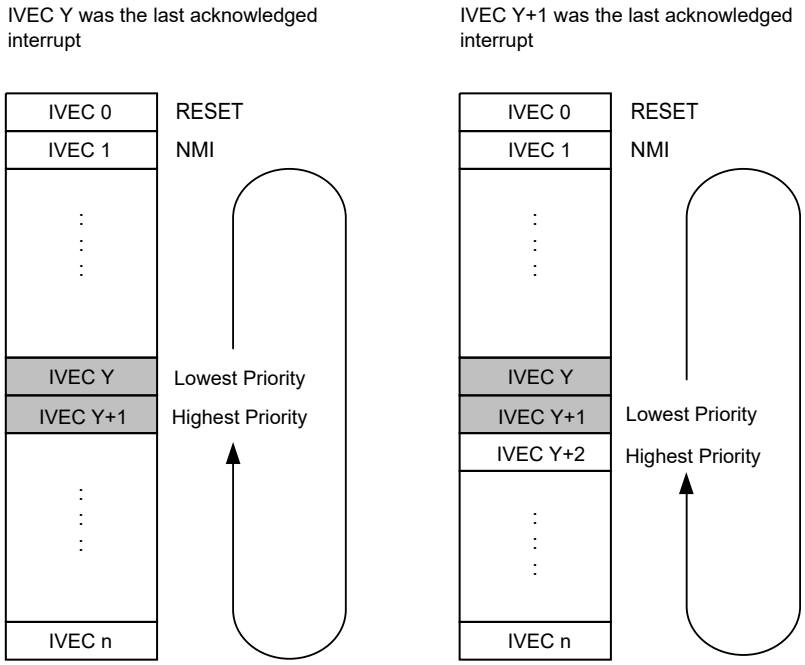
Here, value Y has been written to CPUINT.LVL0PRI, so that interrupt vector Y+1 has the highest priority. Note that, in this case, the priorities will wrap so that the lowest address no longer has the highest priority. This does not include RESET and NMI, which will always have the highest priority.

Refer to the interrupt vector mapping of the device for available interrupt requests and their interrupt vector number.

Round Robin Scheduling

The static scheduling may prevent some interrupt requests from being serviced. To avoid this, the CPUINT offers round robin scheduling for normal-priority (LVL0) interrupts. In the round robin scheduling, the CPUINT.LVL0PRI register stores the last acknowledged interrupt vector number. This register ensures that the last acknowledged interrupt vector gets the lowest priority and is automatically updated by the hardware. The following figure illustrates the priority order after acknowledging IVEC Y and after acknowledging IVEC Y+1.

Figure 13-7. Round Robin Scheduling



The round robin scheduling for LVL0 interrupt requests is enabled by writing a '1' to the Round Robin Priority Enable (LVL0RR) bit in the Control A (CPUINT.CTRLA) register.

13.3.2.5 Compact Vector Table

The Compact Vector Table (CVT) is a feature to allow writing of compact code by having all level 0 interrupts share the same interrupt vector number. Thus, the interrupts share the same Interrupt Service Routine (ISR). This reduces the number of interrupt handlers and thereby frees up memory that can be used for the application code.

When CVT is enabled by writing a '1' to the CVT bit in the Control A (CPUINT.CTRLA) register, the vector table contains these three interrupt vectors:

1. The non-maskable interrupts (NMI) at vector address 1.
2. The Priority Level 1 (LVL1) interrupt at vector address 2.
3. All priority level 0 (LVL0) interrupts at vector address 3.

This feature is most suitable for devices with limited memory and applications using a small number of interrupt generators.

13.3.3 Debug Operation

When using a level 1 priority interrupt, it is important to make sure the Interrupt Service Routine is configured correctly as it may cause the application to be stuck in an interrupt loop with level 1 priority.

By reading the CPUINT STATUS (CPUINT.STATUS) register, it is possible to see if the application has executed the correct `RETI` (interrupt return) instruction. The CPUINT.STATUS register contains state information, which ensures that the CPUINT returns to the correct interrupt level when the `RETI` instruction is executed at the end of an interrupt handler. Returning from an interrupt will return the CPUINT to the state it had before entering the interrupt.

13.3.4 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 13-3. CPUINT - Registers under Configuration Change Protection

Register	Key
IVSEL in CPUINT.CTRLA	IOREG
CVT in CPUINT.CTRLA	IOREG

13.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		IVSEL	CVT					LVL0RR
0x01	STATUS	7:0	NMIEX						LVL1EX	LVL0EX
0x02	LVL0PRI	7:0	LVL0PRI[7:0]							
0x03	LVL1VEC	7:0	LVL1VEC[7:0]							

13.5 Register Description

13.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
		IVSEL	CVT					LVL0RR
Access		R/W	R/W					R/W
Reset		0	0					0

Bit 6 – IVSEL Interrupt Vector Select
This bit is protected by the Configuration Change Protection mechanism.

Value	Description
0	Interrupt vectors are placed at the start of the application section of the Flash
1	Interrupt vectors are placed at the start of the boot section of the Flash

Bit 5 – CVT Compact Vector Table
This bit is protected by the Configuration Change Protection mechanism.

Value	Description
0	Compact Vector Table function is disabled
1	Compact Vector Table function is enabled

Bit 0 – LVL0RR Round Robin Priority Enable
This bit is not protected by the Configuration Change Protection mechanism.

Value	Description
0	Priority is fixed for priority level 0 interrupt requests: The lowest interrupt vector address has the highest priority
1	The round robin priority scheme is enabled for priority level 0 interrupt requests

13.5.2 Status

Name: STATUS
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	NMIEX						LVL1EX	LVL0EX
Access	R						R	R
Reset	0						0	0

Bit 7 – NMIEX Non-Maskable Interrupt Executing
This flag is set if a non-maskable interrupt is executing. The flag is cleared when returning (RETI) from the interrupt handler.

Bit 1 – LVL1EX Level 1 Interrupt Executing
This flag is set when a priority level 1 interrupt is executing, or when the interrupt handler has been interrupted by an NMI. The flag is cleared when returning (RETI) from the interrupt handler.

Bit 0 – LVL0EX Level 0 Interrupt Executing
This flag is set when a priority level 0 interrupt is executing, or when the interrupt handler has been interrupted by a priority level 1 interrupt or an NMI. The flag is cleared when returning (RETI) from the interrupt handler.

13.5.3 Interrupt Priority Level 0

Name: LVL0PRI
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	LVL0PRI[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 – LVL0PRI[7:0] Interrupt Priority Level 0
This register is used to modify the priority of the LVL0 interrupts. See the section Normal-Priority Interrupts for more information.

13.5.4 Interrupt Vector with Priority Level 1

Name: LVL1VEC
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	LVL1VEC[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 – LVL1VEC[7:0] Interrupt Vector with Priority Level 1
This bit field contains the number of the single vector with increased priority level 1 (LVL1). If this bit field has the value `0x00`, no vector has LVL1. Consequently, the LVL1 interrupt is disabled.

14. EVSYS - Event System

14.1 Features

- System for Direct Peripheral-to-Peripheral Signaling
- Peripherals Can Directly Produce, Use and React to Peripheral Events
- Short Response Time
- Up to Four Parallel Asynchronous Event Channels Available
- Up to Two Parallel Synchronous Event Channels Available
- Channels Can Be Configured to Have One Triggering Peripheral Action and Multiple Peripheral Users
- Peripherals Can Directly Trigger and React to Events from Other Peripherals
- Events Can Be Sent and/or Received by Most Peripherals, and by Software
- Works in Active Mode and Standby Sleep Mode

14.2 Overview

The Event System (EVSYS) enables direct peripheral-to-peripheral signaling. It allows a change in one peripheral (the event generator) to trigger actions in other peripherals (the event users) through event channels, without using the CPU. It is designed to provide short and predictable response times between peripherals, allowing for autonomous peripheral control and interaction, and also for the synchronized timing of actions in several peripheral modules. It is thus a powerful tool for reducing the complexity, size, and the execution time of the software.

A change of the event generator's state is referred to as an event and usually corresponds to one of the peripheral's interrupt conditions. Events can be directly forwarded to other peripherals using the dedicated event routing network. The routing of each channel is configured in software, including event generation and use.

Only one trigger from an event generator peripheral can be routed on each channel, but multiple channels can use the same generator source. Multiple peripherals can use events from the same channel.

A channel path can be either asynchronous or synchronous to the main clock. The mode must be selected based on the requirements of the application.

The Event System can directly connect analog and digital converters, analog comparators, I/O port pins, the real-time counter, timer/counters, and the configurable custom logic peripheral. Events can also be generated from software and the peripheral clock.

14.2.1 Block Diagram

Figure 14-1. Block Diagram

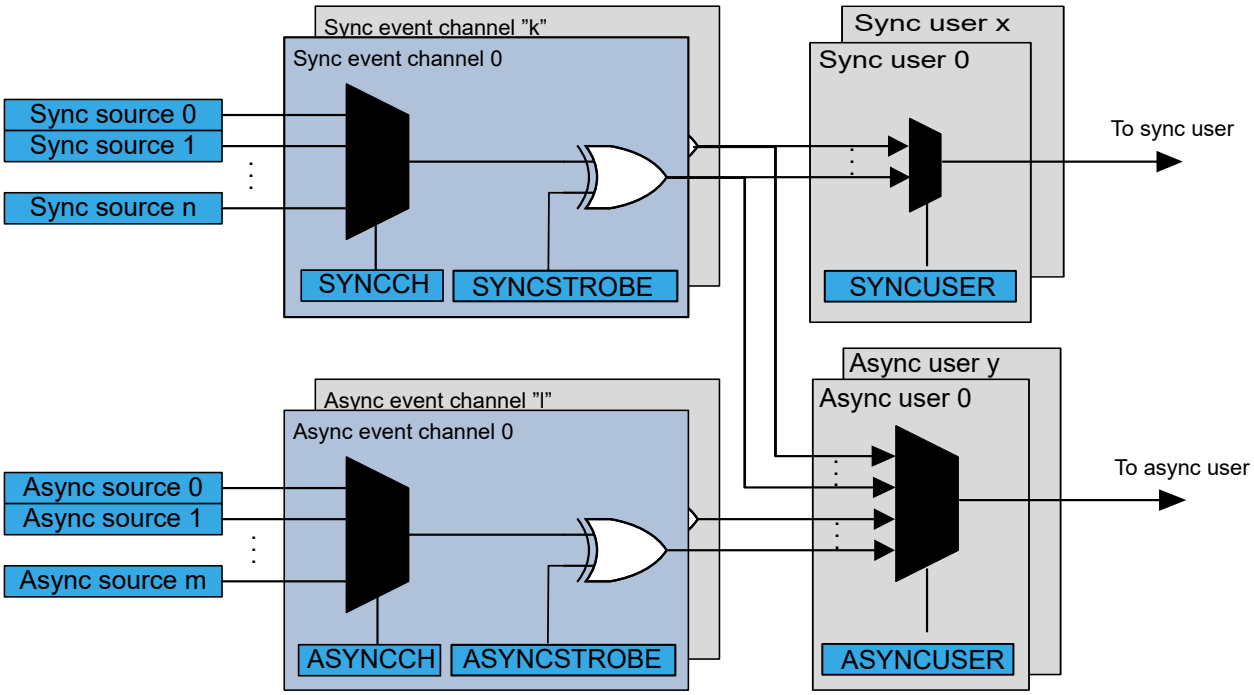
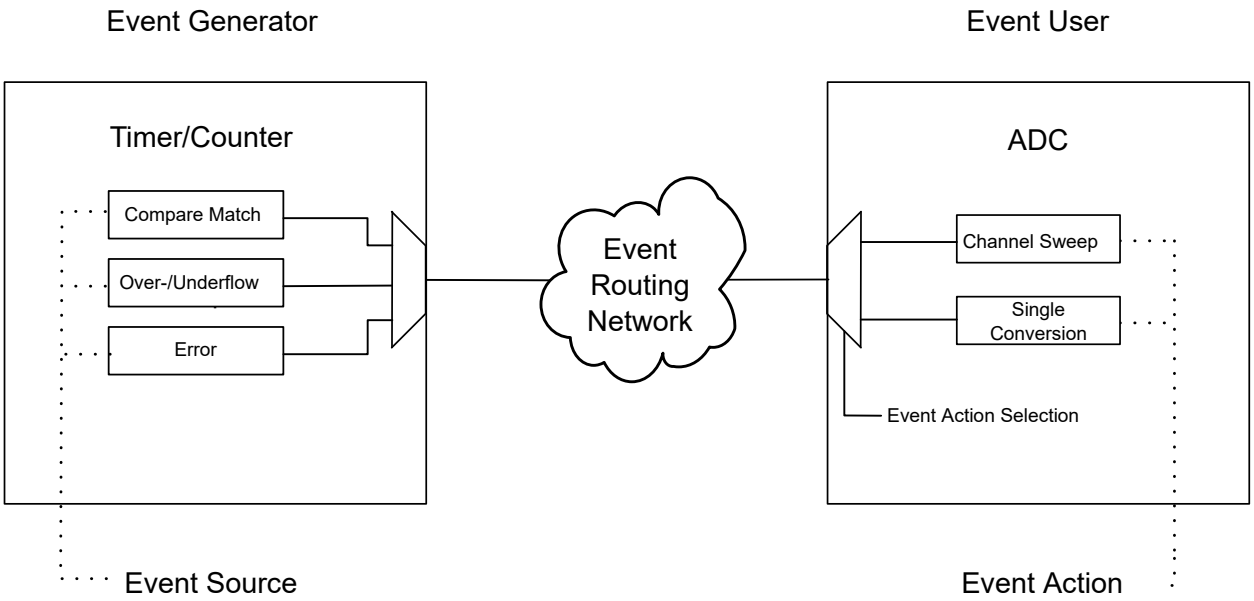


Figure 14-2. Example of Event Source, Generator, User and Action



Note:

1. For an overview of peripherals supporting events, refer to the block diagram of the device.
2. For a list of event generators, refer to the Channel n Generator Selection (EVSYS.SYNCCH and EVSYS.ASYNCCH) registers.
3. For a list of event users, refer to the User Channel n Input Selection (EVSYS.SYNCUSER and EVSYS.ASYNCUSER) registers.

14.2.2 Signal Description

Internal Event Signaling
The event signaling can happen either synchronously or asynchronously to the main clock (CLK_MAIN).

Depending on the underlying event, the event signal can be a pulse with a duration of one clock cycle, or a level signal (similar to a status flag).

Event Output to Pin

Signal	Type	Description
EVOUT[2:0]	Digital Output	Event Output

14.2.3 System Dependencies

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

Table 14-1. EVSYS System Dependencies

Dependency	Applicable	Peripheral
Clocks	Yes	CLKCTRL
I/O Lines and Connections	Yes	PORTMUX
Interrupts	No	-
Events	Yes	EVSYS
Debug	Yes	UPDI

14.2.3.1 Clocks

The EVSYS uses the peripheral clock for I/O registers and software events. When correctly set up, the routing network can also be used in sleep modes without any clock. Software events will not work in sleep modes where the peripheral clock is halted.

14.2.3.2 I/O Lines

The EVSYS can output three event channels asynchronously on pins. The output signals are called EVOUT[2:0].

1. Configure which event channel (one of SYNCCH[1:0] or ASYNCCH[3:0]) is output on which EVOUTn bit by writing to EVSYS.ASYNCUSER10, EVSYS.ASYNCUSER9, or EVSYS.ASYNCUSER8, respectively.
2. Optional: Configure the pin properties using the port peripheral.
3. Enable the pin output by writing '1' to the respective EVOUTn bit in the Control A (PORTMUX.CTRLA) register of the PORTMUX peripheral.

14.3 Functional Description

14.3.1 Initialization

Before enabling events within the device, the event users multiplexer and event channels must be configured.

14.3.2 Operation

14.3.2.1 Event User Multiplexer Setup

The event user multiplexer selects the channel for an event user. Each event user has one dedicated event user multiplexer. Each multiplexer is connected to the supported event channel outputs and can be configured to select one of these channels.

Event users, which support asynchronous events, also support synchronous events. There are also event users that support only synchronous events.

The event user multiplexers are configured by writing to the corresponding registers:

- Event users supporting both synchronous and asynchronous events are configured by writing to the respective asynchronous User Channel Input Selection n (EVSYS.ASYNCUSERn) register
- The users of synchronous-only events are configured by writing to the respective Synchronous User Channel Input Selection n (EVSYS.SYNCUSERn) register

The default setup of all user multiplexers is OFF.

14.3.2.2 Event System Channel

An event channel can be connected to one of the event generators. Event channels support either asynchronous generators or synchronous generators.

The source for each asynchronous event channel is configured by writing to the respective Asynchronous Channel n Input Selection (EVSYS.ASYNCCHn) register.

The source for each synchronous event channel is configured by writing to the respective Synchronous Channel n Input Selection (EVSYS.SYNCCHn) register.

14.3.2.3 Event Generators

Each event channel can receive the events from several event generators. For details on event generation, refer to the documentation of the corresponding peripheral.

For each event channel, there are several possible event generators, only one of which can be selected at a time. The event generator trigger is selected for each channel by writing to the respective channel registers (EVSYS.ASYNCCHn, EVSYS.SYNCCHn). By default, the channels are not connected to any event generator.

14.3.2.4 Software Event

In a software event, the CPU will “strobe” an event channel by inverting the current value for one system clock cycle.

A software event is triggered on a channel by writing a ‘1’ to the respective Strobe bit in the appropriate Channel Strobe register:

- Software events on asynchronous channel *l* are initiated by writing a ‘1’ to the ASYNCSTROBE[*l*] bit in the Asynchronous Channel Strobe (EVSYS.ASYNCSTROBE) register
- Software events on synchronous channel *k* are initiated by writing a ‘1’ to the SYNCSTROBE[*k*] bit in the Synchronous Channel Strobe (EVSYS.SYNCSTROBE) register

Software events are no different to those produced by event generator peripherals with respect to event users: When the bit is written to ‘1’, an event will be generated on the respective channel, and received and processed by the event user.

14.3.3 Interrupts

Not applicable.

14.3.4 Sleep Mode Operation

When configured, the Event System will work in all sleep modes. One exception is software events that require a system clock.

14.3.5 Debug Operation

This peripheral is unaffected by entering Debug mode.

14.3.6 Synchronization

Asynchronous events are synchronized and handled by compatible event users. Event user peripherals not compatible with asynchronous events can only be configured to listen to synchronous event channels.

14.3.7 Configuration Change Protection

Not applicable.

14.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	ASYNCSTROBE	7:0	ASYNCSTROBE[7:0]							
0x01	SYNCSTROBE	7:0	SYNCSTROBE[7:0]							
0x02	ASYNCCH0	7:0	ASYNCCH[7:0]							
0x03	ASYNCCH1	7:0	ASYNCCH[7:0]							
0x04	ASYNCCH2	7:0	ASYNCCH[7:0]							
0x05	ASYNCCH3	7:0	ASYNCCH[7:0]							
0x06 ... 0x09	Reserved									
0x0A	SYNCCH0	7:0	SYNCCH[7:0]							
0x0B	SYNCCH1	7:0	SYNCCH[7:0]							
0x0C ... 0x11	Reserved									
0x12	ASYNCUSER0	7:0	ASYNCUSER[7:0]							
...										
0x1E	ASYNCUSER12	7:0	ASYNCUSER[7:0]							
0x1F ... 0x21	Reserved									
0x22	SYNCUSER0	7:0	SYNCUSER[7:0]							
0x23	SYNCUSER1	7:0	SYNCUSER[7:0]							

14.5 Register Description

14.5.1 Asynchronous Channel Strobe

Name: ASYNCSTROBE
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ASYNCSTROBE[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 – ASYNCSTROBE[7:0] Asynchronous Channel Strobe
If the Strobe register location is written, each event channel will be inverted for one system clock cycle (i.e., a single event is generated).

14.5.2 Synchronous Channel Strobe

Name: SYNCSTROBE
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	SYNCSTROBE[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 – SYNCSTROBE[7:0] Synchronous Channel Strobe
If the Strobe register location is written, each event channel will be inverted for one system clock cycle (i.e., a single event is generated).

14.5.3 Asynchronous Channel n Generator Selection

Name: ASYNCCHn
Offset: 0x02 + n*0x01 [n=0..3]
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ASYNCCH[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 – ASYNCCH[7:0] Asynchronous Channel Generator Selection

Value	ASYNCCH0	ASYNCCH1	ASYNCCH2	ASYNCCH3
0x00	OFF	OFF	OFF	OFF
0x01	CCL_LUT0			
0x02	CCL_LUT1			
0x03	AC0_OUT			
0x04	TCD0_CMPBCLR			
0x05	TCD0_CMPASET			
0x06	TCD0_CMPBSET			
0x07	TCD0_PROGEV			
0x08	RTC_OVF			
0x09	RTC_CMP			
0x0A	PORTA_PIN0	PORTB_PIN0	PORTC_PIN0	PIT_DIV8192
0x0B	PORTA_PIN1	PORTB_PIN1	PORTC_PIN1	PIT_DIV4096
0x0C	PORTA_PIN2	PORTB_PIN2	PORTC_PIN2	PIT_DIV2048
0x0D	PORTA_PIN3	PORTB_PIN3	PORTC_PIN3	PIT_DIV1024
0x0E	PORTA_PIN4	PORTB_PIN4	PORTC_PIN4	PIT_DIV512
0x0F	PORTA_PIN5	PORTB_PIN5	PORTC_PIN5	PIT_DIV256
0x10	PORTA_PIN6	PORTB_PIN6	AC1_OUT	PIT_DIV128
0x11	PORTA_PIN7	PORTB_PIN7	AC2_OUT	PIT_DIV64
0x12	UPDI	AC1_OUT		AC1_OUT
0x13	AC1_OUT	AC2_OUT	-	AC2_OUT
0x14	AC2_OUT	-	-	-
Other	-	-	-	-

Note: Not all pins of a port are available on devices with low pin counts. Check the Pinout Diagram and/or the I/O Multiplexing table for details.

14.5.4 Synchronous Channel n Generator Selection

Name: SYNCCHn
Offset: 0x0A + n*0x01 [n=0..1]
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	SYNCCH[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 – SYNCCH[7:0] Synchronous Channel Generator Selection

Value	SYNCCH0	SYNCCH1
0x00	OFF	
0x01	TCB0	
0x02	TCA0_OVF_LUNF	
0x03	TCA0_HUNF	
0x04	TCA0_CMP0	
0x05	TCA0_CMP1	
0x06	TCA0_CMP2	
0x07	PORTC_PIN0	-
0x08	PORTC_PIN1	PORTB_PIN0
0x09	PORTC_PIN2	PORTB_PIN1
0x0A	PORTC_PIN3	PORTB_PIN2
0x0B	PORTC_PIN4	PORTB_PIN3
0x0C	PORTC_PIN5	PORTB_PIN4
0x0D	PORTA_PIN0	PORTB_PIN5
0x0E	PORTA_PIN1	PORTB_PIN6
0x0F	PORTA_PIN2	PORTB_PIN7
0x10	PORTA_PIN3	TCB1
0x11	PORTA_PIN4	-
0x12	PORTA_PIN5	-
0x13	PORTA_PIN6	-
0x14	PORTA_PIN7	-
0x15	TCB1	-
Other	-	-

Note: Not all pins of a port are available on devices with low pin counts. Check the Pinout Diagram and/or the I/O Multiplexing table for details.

14.5.5 Asynchronous User Channel n Input Selection

Name: ASYNCUSERn
Offset: 0x12 + n*0x01 [n=0..12]
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ASYNCUSER[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 – ASYNCUSER[7:0] Asynchronous User Channel Selection

ASYNCUSERn	User Multiplexer	Description
0	TCB0	Timer/Counter B 0
1	ADC0	ADC 0
2	CCL_LUT0EV0	CCL LUT0 Event 0
3	CCL_LUT1EV0	CCL LUT1 Event 0
4	CCL_LUT0EV1	CCL LUT0 Event 1
5	CCL_LUT1EV1	CCL LUT1 Event 1
6	TCD0_EV0	Timer Counter D 0 Event 0
7	TCD0_EV1	Timer Counter D 0 Event 1
8	EVOUT0	Event OUT 0
9	EVOUT1	Event OUT 1
10	EVOUT2	Event OUT 2
11	TCB1	Timer/Counter B 1
12	ADC1	ADC 1

Value	Name
0x0	OFF
0x1	SYNCCH0
0x2	SYNCCH1
0x3	ASYNCCH0
0x4	ASYNCCH1
0x5	ASYNCCH2
0x6	ASYNCCH3
Other	-

14.5.6 Synchronous User Channel n Input Selection

Name: SYNCUSERn
Offset: 0x22 + n*0x01 [n=0..1]
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	SYNCUSER[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 – SYNCUSER[7:0] Synchronous User Channel Selection

SYNCUSERn	User Multiplexer	Description
0	TCA0	Timer/Counter A
1	USART0	USART

Value	Name
0x0	OFF
0x1	SYNCCH0
0x2	SYNCCH1
Other	-

15. PORTMUX - Port Multiplexer

15.1 Overview

The Port Multiplexer (PORTMUX) can either enable or disable the functionality of pins, or change between default and alternative pin positions. This depends on the actual pin and property and is described in detail in the PORTMUX register map.

For available pins and functionalities, refer to Table 5-1.

15.2 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0			LUT1	LUT0		EVOUT2	EVOUT1	EVOUT0
0x01	CTRLB	7:0				TWI0		SPI0		USART0
0x02	CTRLC	7:0			TCA05	TCA04	TCA03	TCA02	TCA01	TCA00
0x03	CTRLD	7:0							TCB1	TCB0

15.3 Register Description

15.3.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
			LUT1	LUT0		EVOUT2	EVOUT1	EVOUT0
Access			R/W	R/W		R/W	R/W	R/W
Reset			0	0		0	0	0

Bit 5 – LUT1 CCL LUT 1 Output
Write this bit to '1' to select the alternative pin location for CCL LUT 1.

Bit 4 – LUT0 CCL LUT 0 Output
Write this bit to '1' to select the alternative pin location for CCL LUT 0.

Bit 2 – EVOUT2 Event Output 2
Write this bit to '1' to enable event output 2.

Bit 1 – EVOUT1 Event Output 1
Write this bit to '1' to enable event output 1.

Bit 0 – EVOUT0 Event Output 0
Write this bit to '1' to enable event output 0.

15.3.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
				TWI0		SPI0		USART0
Access				R/W		R/W		R/W
Reset				0		0		0

Bit 4 – TWI0 TWI 0 Communication
Write this bit to '1' to select alternative communication pins for TWI 0.

Bit 2 – SPI0 SPI 0 Communication
Write this bit to '1' to select alternative communication pins for SPI 0.

Bit 0 – USART0 USART 0 Communication
Write this bit to '1' to select alternative communication pins for USART 0.

15.3.3 Control C

Name: CTRLC
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
			TCA05	TCA04	TCA03	TCA02	TCA01	TCA00
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TCA05 TCA0 Waveform Output 5
Write this bit to '1' to select the alternative output pin for TCA0 waveform output 5 in Split mode.
Not applicable when TCA in Normal mode.

Bit 4 – TCA04 TCA0 Waveform Output 4
Write this bit to '1' to select the alternative output pin for TCA0 waveform output 4 in Split mode.
Not applicable when TCA in Normal mode.

Bit 3 – TCA03 TCA0 Waveform Output 3
Write this bit to '1' to select the alternative output pin for TCA0 waveform output 3 in Split mode.
Not applicable when TCA in Normal mode.

Bit 2 – TCA02 TCA0 Waveform Output 2
Write this bit to '1' to select the alternative output pin for TCA0 waveform output 2.
In Split Mode, this bit controls output from low byte compare channel 2.

Bit 1 – TCA01 TCA0 Waveform Output 1
Write this bit to '1' to select the alternative output pin for TCA0 waveform output 1.
In Split mode, this bit controls output from low byte compare channel 1.

Bit 0 – TCA00 TCA0 Waveform Output 0
Write this bit to '1' to select the alternative output pin for TCA0 waveform output 0.
In Split mode, this bit controls output from low byte compare channel 0.

15.3.4 Control D

Name: CTRLD
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
							TCB1	TCB0
Access							R/W	R/W
Reset							0	0

Bit 1 – TCB1 TCB1 Output
Write this bit to '1' to select the alternative output pin for 16-bit timer/counter B 1.

Bit 0 – TCB0 TCB0 Output
Write this bit to '1' to select the alternative output pin for 16-bit timer/counter B 0.

16. PORT - I/O Pin Configuration

16.1 Features

- General Purpose Input and Output Pins with Individual Configuration:
  - Pull-up
  - Inverted I/O
- Interrupts and Events:
  - Sense both edges
  - Sense rising edges
  - Sense falling edges
  - Sense low level
- Asynchronous Pin Change Sensing that Can Wake the Device From all Sleep Modes
- Efficient and Safe Access to Port Pins
  - Hardware Read-Modify-Write (RMW) through dedicated toggle/clear/set registers
  - Mapping of often-used PORT registers into bit-accessible I/O memory space (virtual ports)

16.2 Overview

The I/O pins of the device are controlled by instances of the PORT peripheral registers. Each PORT instance has up to eight I/O pins. The PORTs are named PORTA, PORTB, PORTC, etc. Refer to the *I/O Multiplexing and Considerations* section to see which pins are controlled by what instance of PORT. The base addresses of the PORT instances and the corresponding Virtual PORT instances are listed in the *Peripherals and Architecture* section.

Each PORT pin has a corresponding bit in the Data Direction (PORTx.DIR) and Data Output Value (PORTx.OUT) registers to enable that pin as an output and to define the output state. For example, pin PA3 is controlled by DIR[3] and OUT[3] of the PORTA instance.

The input value of a PORT pin is synchronized to the Peripheral Clock (CLK_PER) and then made accessible as the data input value (PORTx.IN). The value of the pin can be read whether the pin is configured as input or output.

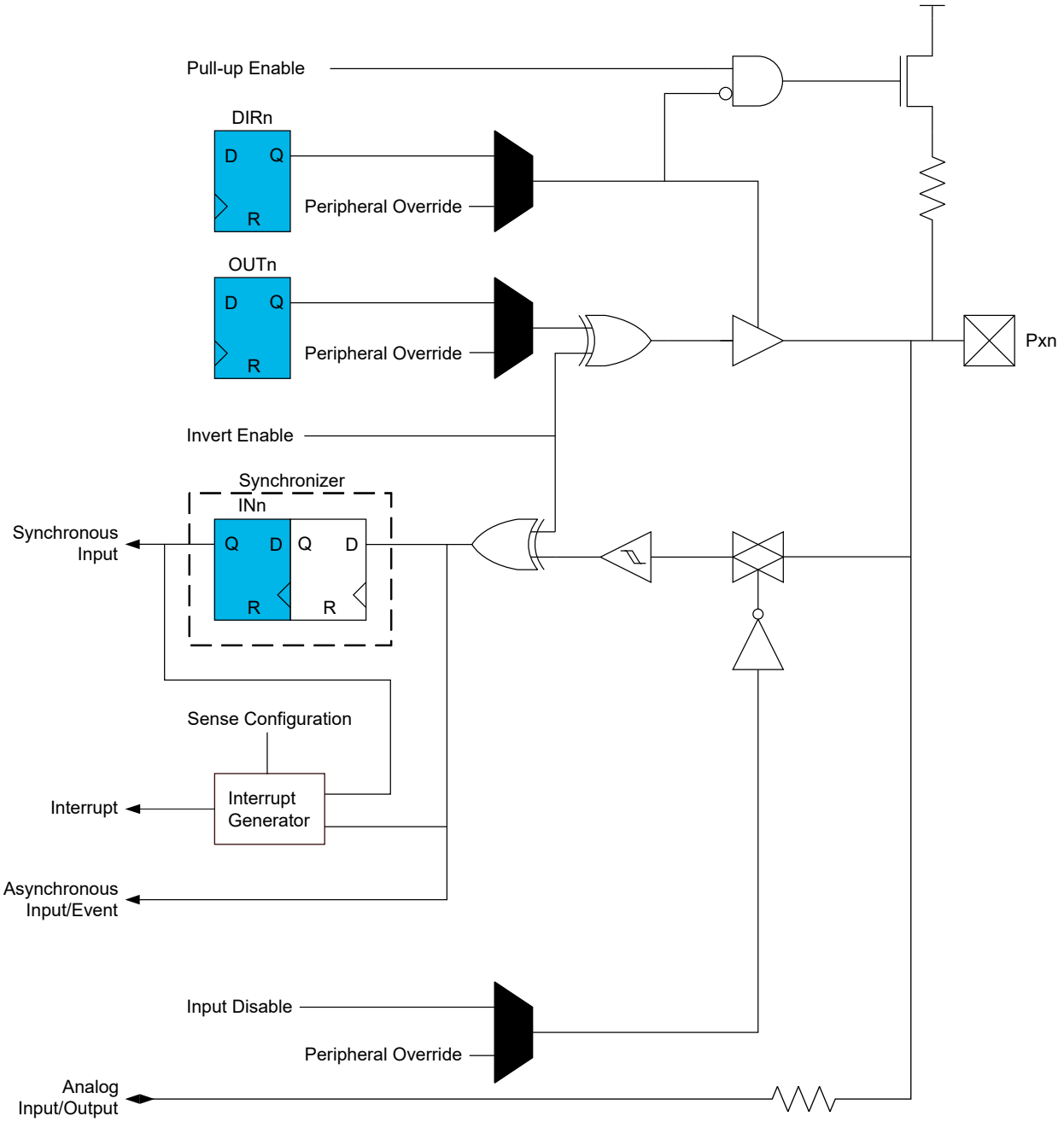
The PORT also supports asynchronous input sensing with interrupts and events for selectable pin change conditions. Asynchronous pin change sensing means that a pin change can trigger an interrupt and wake the device from sleep, including sleep modes where CLK_PER is stopped.

All pin functions are individually configurable per pin. The pins have hardware Read-Modify-Write functionality for a safe and correct change of the drive values and/or input and sense configuration.

The PORT pin configuration controls input and output selection of other device functions.

16.2.1 Block Diagram

Figure 16-1. PORT Block Diagram



16.2.2 Signal Description

Signal	Type	Description
Pxn	I/O pin	I/O pin n on PORTx

16.3 Functional Description

16.3.1 Initialization

After Reset, all outputs are tri-stated, and digital input buffers enabled even if there is no clock running.

The following steps are all optional when initializing PORT operation:

- Enable or disable the output driver for pin Pxn by respectively writing '1' to bit n in the PORTx.DIRSET or PORTx.DIRCLR register
- Set the output driver for pin Pxn to high or low level respectively by writing '1' to bit n in the PORTx.OUTSET or PORTx.OUTCLR register
- Read the input of pin Pxn by reading bit n in the PORTx.IN register
- Configure the individual pin configurations and interrupt control for pin Pxn in PORTx.PINnCTRL

Important: For lowest power consumption, disable the digital input buffer of unused pins and pins that are used as analog inputs or outputs.

Specific pins, such as those used to connect a debugger, may be configured differently, as required by their special function.

16.3.2 Operation

16.3.2.1 Basic Functions

Each pin group x has its own set of PORT registers. I/O pin Pxn can be controlled by the registers in PORTx.

To use pin number n as an output, write bit n of the PORTx.DIR register to '1'. This can be done by writing bit n in the PORTx.DIRSET register to '1', which will avoid disturbing the configuration of other pins in that group. The n^{th} bit in the PORTx.OUT register must be written to the desired output value.

Similarly, writing a PORTx.OUTSET bit to '1' will set the corresponding bit in the PORTx.OUT register to '1'. Writing a bit in PORTx.OUTCLR to '1' will clear that bit in PORTx.OUT to '0'. Writing a bit in PORTx.OUTTGL or PORTx.IN to '1' will toggle that bit in PORTx.OUT.

To use pin n as an input, bit n in the PORTx.DIR register must be written to '0' to disable the output driver. This can be done by writing bit n in the PORTx.DIRCLR register to '1', which will avoid disturbing the configuration of other pins in that group. The input value can be read from bit n in the PORTx.IN register as long as the ISC bit is not set to INPUT_DISABLE.

Writing a bit to '1' in PORTx.DIRTGL will toggle that bit in PORTx.DIR and toggle the direction of the corresponding pin.

16.3.2.2 Pin Configuration

The Pin n Control (PORTx.PINnCTRL) register is used to configure inverted I/O, pull-up, and input sensing of a pin. The control register for pin n is at the byte address $\mathrm{PORTx} + 0\mathrm{x}10 + n$.

All input and output on the respective pin n can be inverted by writing a '1' to the Inverted I/O Enable (INVEN) bit in PORTx.PINnCTRL. When INVEN is '1', the PORTx.IN/OUT/OUTSET/OUTTGL registers will have an inverted operation for this pin.

Toggling the INVEN bit causes an edge on the pin, which can be detected by all peripherals using this pin, and is seen by interrupts or events if enabled.

The input pull-up of pin n is enabled by writing a '1' to the Pull-up Enable (PULLUPEN) bit in PORTx.PINnCTRL. The pull-up is disconnected when the pin is configured as an output, even if PULLUPEN is '1'.

Pin interrupts can be enabled for pin n by writing to the Input/Sense Configuration (ISC) bit field in PORTx.PINnCTRL. Refer to 16.3.3 Interrupts for further details.

The digital input buffer for pin n can be disabled by writing the INPUT_DISABLE setting to ISC. This can reduce power consumption and may reduce noise if the pin is used as analog input. While configured to INPUT_DISABLE, bit n in PORTx.IN will not change since the input synchronizer is disabled.

16.3.2.3 Virtual Ports

The Virtual PORT registers map the most frequently used regular PORT registers into the I/O Register space with single-cycle bit access. Access to the Virtual PORT registers has the same outcome as access to the regular registers but allows for memory specific instructions, such as bit manipulation instructions, which cannot be used in the extended I/O Register space where the regular PORT registers reside. The following table shows the mapping between the PORT and VPORT registers.

Table 16-1. Virtual Port Mapping

Regular PORT Register	Mapped to Virtual PORT Register
PORTx.DIR	VPORTx.DIR
PORTx.OUT	VPORTx.OUT
PORTx.IN	VPORTx.IN
PORTx.INTFLAGS	VPORTx.INTFLAGS

16.3.2.4 Peripheral Override

Peripherals such as USARTs, ADCs and timers may be connected to I/O pins. Such peripherals will usually have a primary and, optionally, one or more alternate I/O pin connections, selectable by PORTMUX or a multiplexer inside the peripheral. By configuring and enabling such peripherals, the general purpose I/O pin behavior normally controlled by PORT will be overridden in a peripheral dependent way. Some peripherals may not override all the PORT registers, leaving the PORT module to control some aspects of the I/O pin operation.

Refer to the description of each peripheral for information on the peripheral override. Any pin in a PORT that is not overridden by a peripheral will continue to operate as a general purpose I/O pin.

16.3.3 Interrupts

Table 16-2. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
PORTx	PORT interrupt	INTn in PORTx.INTFLAGS is raised as configured by the Input/Sense Configuration (ISC) bit in PORTx.PINnCTRL

Each PORT pin n can be configured as an interrupt source. Each interrupt can be individually enabled or disabled by writing to ISC in PORTx.PINnCTRL.

When an interrupt condition occurs, the corresponding interrupt flag is set in the Interrupt Flags register of the peripheral (*peripheral*.INTFLAGS).

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

When setting or changing interrupt settings, take these points into account:

- If an Inverted I/O Enable (INVEN) bit is toggled in the same cycle as ISC is changed, the edge caused by the inversion toggling may not cause an interrupt request
- If an input is disabled by writing to ISC while synchronizing an interrupt, that interrupt may be requested on re-enabling the input, even if it is re-enabled with a different interrupt setting
- If the interrupt setting is changed by writing to ISC while synchronizing an interrupt, that interrupt may not be requested

16.3.3.1 Asynchronous Sensing Pin Properties

All PORT pins support asynchronous input sensing with interrupts for selectable pin change conditions. Fully asynchronous pin change sensing can trigger an interrupt and wake the device from all sleep modes, including modes where the Peripheral Clock (CLK_PER) is stopped, while partially asynchronous pin change sensing is limited

as per the table below. See the *I/O Multiplexing and Considerations* section for further details on which pins support fully asynchronous pin change sensing.

Table 16-3. Behavior Comparison of Sense Pins

Property	Partially Asynchronous Pins	Fully Asynchronous Pins
Waking the device from sleep modes with CLK_PER running	From all interrupt sense configurations	From all interrupt sense configurations
Waking the device from sleep modes with CLK_PER stopped	Only from BOTHEDGES or LEVEL interrupt sense configurations	
Minimum pulse-width to trigger an interrupt with CLK_PER running	Minimum one CLK_PER cycle	Less than one CLK_PER cycle
Minimum pulse-width to trigger an interrupt with CLK_PER stopped	The pin value must be kept until CLK_PER has restarted(1)	
Interrupt “dead-time”	No new interrupt for three CLK_PER cycles after the previous	

Note:

1. If a partially asynchronous input pin is used for wake-up from sleep with CLK_PER stopped, the required level must be held long enough for the MCU to complete the wake-up to trigger the interrupt. If the level disappears, the MCU can wake up without any interrupt generated.

16.3.4 Events

PORT can generate the following events:

Table 16-4. Event Generators in PORTx

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
PORTx	PINn	Pin level	Level	Asynchronous	Given by pin level

All PORT pins are asynchronous event system generators. PORT has as many event generators as there are PORT pins in the device. Each event system output from PORT is the value present on the corresponding pin if the digital input buffer is enabled. If a pin input buffer is disabled, the corresponding event system output is zero.

PORT has no event inputs. Refer to the *Event System (EVSYS)* section for more details regarding event types and Event System configuration.

16.3.5 Sleep Mode Operation

Except for interrupts and input synchronization, all pin configurations are independent of sleep modes. All pins can wake the device from sleep, see the PORT Interrupt section for further details.

Peripherals connected to the PORTs can be affected by sleep modes, described in the respective peripherals’ data sheet section.

Important: The PORTs will always use the Peripheral Clock (CLK_PER). Input synchronization will halt when this clock stops.

16.3.6 Debug Operation

When the CPU is halted in Debug mode, the PORT continues normal operation. If the PORT is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

16.4 Register Summary - PORTx

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	DIR	7:0	DIR[7:0]							
0x01	DIRSET	7:0	DIRSET[7:0]							
0x02	DIRCLR	7:0	DIRCLR[7:0]							
0x03	DIRTGL	7:0	DIRTGL[7:0]							
0x04	OUT	7:0	OUT[7:0]							
0x05	OUTSET	7:0	OUTSET[7:0]							
0x06	OUTCLR	7:0	OUTCLR[7:0]							
0x07	OUTTGL	7:0	OUTTGL[7:0]							
0x08	IN	7:0	IN[7:0]							
0x09	INTFLAGS	7:0	INT[7:0]							
0x0A ... 0x0F	Reserved									
0x10	PIN0CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x11	PIN1CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x12	PIN2CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x13	PIN3CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x14	PIN4CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x15	PIN5CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x16	PIN6CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x17	PIN7CTRL	7:0	INVEN				PULLUPEN	ISC[2:0]		

16.5 Register Description - PORTx

16.5.1 Data Direction

Name: DIR
Offset: 0x00
Reset: 0x00
Property: -

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

Bits 7:0 – DIR[7:0] Data Direction
This bit field controls the output driver for each PORTx pin.
This bit field does not control the digital input buffer. The digital input buffer for pin n (Pxn) can be configured in the Input/Sense Configuration (ISC) bit field in the Pin n Control (PORTx.PINnCTRL) register.
The available configuration for each bit n in this bit field is shown in the table below.

Value	Description
0	Pxn is configured as an input-only pin, and the output driver is disabled
1	Pxn is configured as an output pin, and the output driver is enabled

16.5.2 Data Direction Set

Name: DIRSET
Offset: 0x01
Reset: 0x00
Property: -

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

Bits 7:0 – DIRSET[7:0] Data Direction Set
This bit field controls the output driver for each PORTx pin, without using a read-modify-write operation.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will set the corresponding bit in PORTx.DIR, which will configure pin n (Pxn) as an output pin and enable the output driver.
Reading this bit field will return the value of PORTx.DIR.

16.5.3 Data Direction Clear

Name: DIRCLR
Offset: 0x02
Reset: 0x00
Property: -

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

Bits 7:0 – DIRCLR[7:0] Data Direction Clear
This bit field controls the output driver for each PORTx pin, without using a read-modify-write operation.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will clear the corresponding bit in PORTx.DIR, which will configure pin n (Pxn) as an input-only pin and disable the output driver.
Reading this bit field will return the value of PORTx.DIR.

16.5.4 Data Direction Toggle

Name: DIRTGL
Offset: 0x03
Reset: 0x00
Property: -

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

Bits 7:0 – DIRTGL[7:0] Data Direction Toggle
This bit field controls the output driver for each PORTx pin, without using a read-modify-write operation.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will toggle the corresponding bit in PORTx.DIR.
Reading this bit field will return the value of PORTx.DIR.

16.5.5 Output Value

Name: OUT
Offset: 0x04
Reset: 0x00
Property: -

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

Bits 7:0 – OUT[7:0] Output Value
This bit field controls the output driver level for each PORTx pin.
This configuration only has an effect when the output driver (PORTx.DIR) is enabled for the corresponding pin.
The available configuration for each bit n in this bit field is shown in the table below.

Value	Description
0	The pin n (Pxn) output is driven low
1	The Pxn output is driven high

16.5.6 Output Value Set

Name: OUTSET
Offset: 0x05
Reset: 0x00
Property: -

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

Bits 7:0 – OUTSET[7:0] Output Value Set
This bit field controls the output driver level for each PORTx pin, without using a read-modify-write operation.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will set the corresponding bit in PORTx.OUT, which will configure the output for pin n (Pxn) to be driven high.
Reading this bit field will return the value of PORTx.OUT.

16.5.7 Output Value Clear

Name: OUTCLR
Offset: 0x06
Reset: 0x00
Property: -

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

Bits 7:0 – OUTCLR[7:0] Output Value Clear
This bit field controls the output driver level for each PORTx pin, without using a read-modify-write operation.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will clear the corresponding bit in PORTx.OUT, which will configure the output for pin n (Pxn) to be driven low.
Reading this bit field will return the value of PORTx.OUT.

16.5.8 Output Value Toggle

Name: OUTTGL
Offset: 0x07
Reset: 0x00
Property: -

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

Bits 7:0 – OUTTGL[7:0] Output Value Toggle
This bit field controls the output driver level for each PORTx pin, without using a read-modify-write operation.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will toggle the corresponding bit in PORTx.OUT.
Reading this bit field will return the value of PORTx.OUT.

16.5.9 Input Value

Name: IN
Offset: 0x08
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	IN[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 – IN[7:0] Input Value
This bit field shows the state of the PORTx pins when the digital input buffer is enabled.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will toggle the corresponding bit in PORTx.OUT.
If the digital input buffer is disabled, the input is not sampled, and the bit value will not change. The digital input buffer for pin n (Pxn) can be configured in the Input/Sense Configuration (ISC) bit field in the Pin n Control (PORTx.PINnCTRL) register.
The available states of each bit n in this bit field is shown in the table below.

Value	Description
0	The voltage level on Pxn is low
1	The voltage level on Pxn is high

16.5.10 Interrupt Flags

Name: INTFLAGS
Offset: 0x09
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	INT[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 – INT[7:0] Pin Interrupt Flag
Pin interrupt flag n is cleared by writing a '1' to it.
Pin interrupt flag n is set when the change or state of pin n (Pxn) matches the pin's Input/Sense Configuration (ISC) in PORTx.PINnCTRL.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will clear Pin interrupt flag n.

16.5.11 Pin n Control

Name: PINnCTRL
Offset: 0x10 + n*0x01 [n=0..7]
Reset: 0x00
Property: -

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

Bit 7 – INVEN Inverted I/O Enable
This bit controls whether the input and output for pin n are inverted or not.

Value	Description
0	Input and output values are not inverted
1	Input and output values are inverted

Bit 3 – PULLUPEN Pull-up Enable
This bit controls whether the internal pull-up of pin n is enabled or not when the pin is configured as input-only.

Value	Description
0	Pull-up disabled
1	Pull-up enabled

Bits 2:0 – ISC[2:0] Input/Sense Configuration
This bit field controls the input and sense configuration of pin n. The sense configuration determines how a port interrupt can be triggered.

Value	Name	Description
0x0	INTDISABLE	Interrupt disabled but input buffer enabled
0x1	BOTHEDGES	Interrupt enabled with sense on both edges
0x2	RISING	Interrupt enabled with sense on rising edge
0x3	FALLING	Interrupt enabled with sense on falling edge
0x4	INPUT_DISABLE	Interrupt and digital input buffer disabled(1)
0x5	LEVEL	Interrupt enabled with sense on low level
other	—	Reserved

Note:
1. If the digital input buffer for pin n is disabled, bit n in the Input Value (PORTx.IN) register will not be updated.

16.6 Register Summary - VPORTx

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	DIR	7:0	DIR[7:0]							
0x01	OUT	7:0	OUT[7:0]							
0x02	IN	7:0	IN[7:0]							
0x03	INTFLAGS	7:0	INT[7:0]							

16.7 Register Description - VPORTx

16.7.1 Data Direction

Name: DIR
Offset: 0x00
Reset: 0x00
Property: -

Access to the Virtual PORT registers has the same outcome as access to the regular registers but allows for memory specific instructions, such as bit manipulation instructions, which cannot be used in the extended I/O Register space where the regular PORT registers reside.

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

Bits 7:0 – DIR[7:0] Data Direction
This bit field controls the output driver for each PORTx pin.
This bit field does not control the digital input buffer. The digital input buffer for pin n (Pxn) can be configured in the Input/Sense Configuration (ISC) bit field in the Pin n Control (PORTx.PINnCTRL) register.
The available configuration for each bit n in this bit field is shown in the table below.

Value	Description
0	Pxn is configured as an input-only pin, and the output driver is disabled
1	Pxn is configured as an output pin, and the output driver is enabled

16.7.2 Output Value

Name: OUT
Offset: 0x01
Reset: 0x00
Property: -

Access to the Virtual PORT registers has the same outcome as access to the regular registers but allows for memory specific instructions, such as bit manipulation instructions, which cannot be used in the extended I/O Register space where the regular PORT registers reside.

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

Bits 7:0 – OUT[7:0] Output Value
This bit field controls the output driver level for each PORTx pin.
This configuration only has an effect when the output driver (PORTx.DIR) is enabled for the corresponding pin.
The available configuration for each bit n in this bit field is shown in the table below.

Value	Description
0	The pin n (Pxn) output is driven low
1	The Pxn output is driven high

16.7.3 Input Value

Name: IN
Offset: 0x02
Reset: 0x00
Property: -

Access to the Virtual PORT registers has the same outcome as access to the regular registers but allows for memory specific instructions, such as bit manipulation instructions, which cannot be used in the extended I/O Register space where the regular PORT registers reside.

Bit	7	6	5	4	3	2	1	0
	IN[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 – IN[7:0] Input Value
This bit field shows the state of the PORTx pins when the digital input buffer is enabled.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will toggle the corresponding bit in PORTx.OUT.
If the digital input buffer is disabled, the input is not sampled, and the bit value will not change. The digital input buffer for pin n (Pxn) can be configured in the Input/Sense Configuration (ISC) bit field in the Pin n Control (PORTx.PINnCTRL) register.
The available states of each bit n in this bit field is shown in the table below.

Value	Description
0	The voltage level on Pxn is low
1	The voltage level on Pxn is high

16.7.4 Interrupt Flags

Name: INTFLAGS
Offset: 0x03
Reset: 0x00
Property: -

Access to the Virtual PORT registers has the same outcome as access to the regular registers but allows for memory specific instructions, such as bit manipulation instructions, which cannot be used in the extended I/O Register space where the regular PORT registers reside.

Bit	7	6	5	4	3	2	1	0
	INT[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 – INT[7:0] Pin Interrupt Flag
Pin interrupt flag n is cleared by writing a '1' to it.
Pin interrupt flag n is set when the change or state of pin n (Pxn) matches the pin's Input/Sense Configuration (ISC) in PORTx.PINnCTRL.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will clear Pin interrupt flag n.

17. BOD - Brown-out Detector

17.1 Features

- Brown-out Detector Monitors the Power Supply to Avoid Operation Below a Programmable Level
- Three Available Modes:
  - Enabled mode (continuously active)
  - Sampled mode
  - Disabled
- Separate Selection of Mode for Active and Sleep Modes
- Voltage Level Monitor (VLM) with Interrupt
- Programmable VLM Level Relative to the BOD Level

17.2 Overview

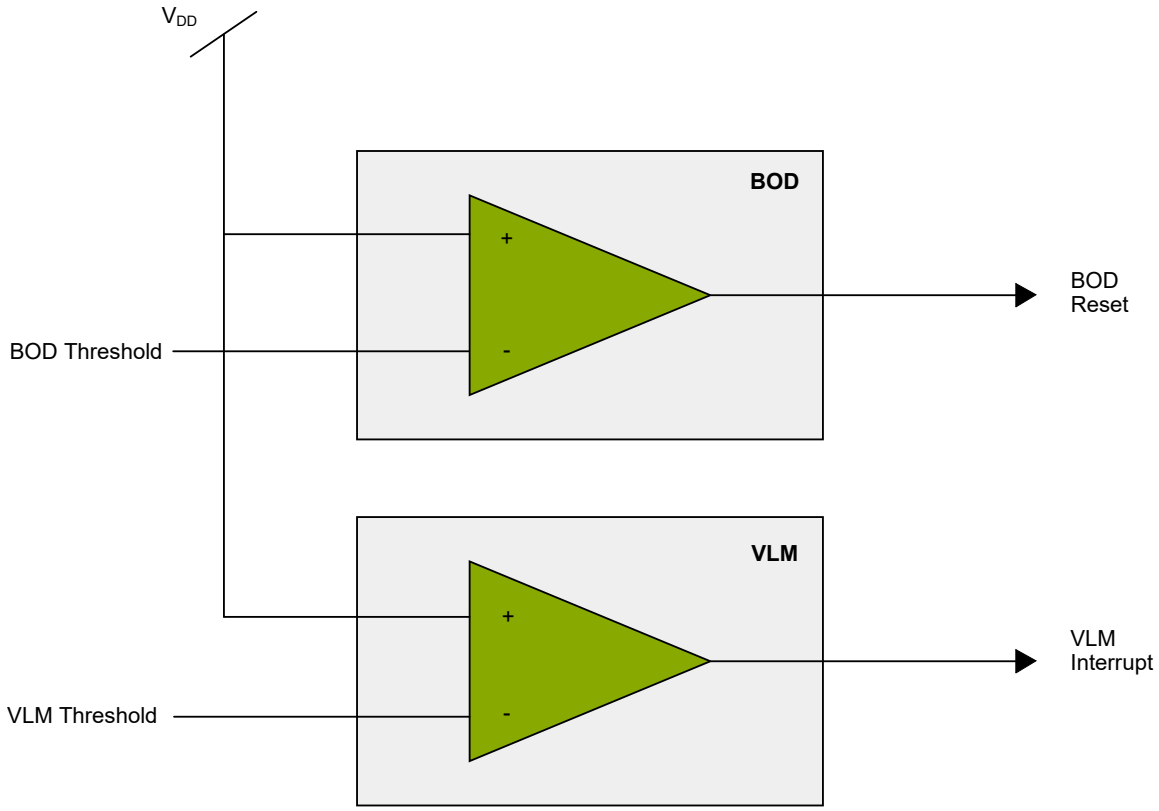
The Brown-out Detector (BOD) monitors the power supply and compares the supply voltage with the programmable brown-out threshold level. The brown-out threshold level defines when to generate a System Reset. The Voltage Level Monitor (VLM) monitors the power supply and compares it to a threshold higher than the BOD threshold. The VLM can then generate an interrupt as an “early warning” when the supply voltage is approaching the BOD threshold. The VLM threshold level is expressed as a percentage above the BOD threshold level.

The BOD is controlled mainly by fuses and has to be enabled by the user. The mode used in Standby sleep mode and Power-Down sleep mode can be altered in normal program execution. The VLM is controlled by I/O registers as well.

When activated, the BOD can operate in Enabled mode, where the BOD is continuously active, or in Sampled mode, where the BOD is activated briefly at a given period to check the supply voltage level.

17.2.1 Block Diagram

Figure 17-1. BOD Block Diagram



17.3 Functional Description

17.3.1 Initialization

The BOD settings are loaded from fuses during Reset. The BOD level and operating mode in Active and Idle sleep mode are set by fuses and cannot be changed by software. The operating mode in Standby and Power-Down sleep mode is loaded from fuses and can be changed by software.

The Voltage Level Monitor function can be enabled by writing a '1' to the VLM Interrupt Enable (VLMIE) bit in the Interrupt Control (BOD.INTCTRL) register. The VLM interrupt is configured by writing the VLM Configuration (VLMCFG) bits in BOD.INTCTRL. An interrupt is requested when the supply voltage crosses the VLM threshold either from above, below, or any direction.

The VLM functionality will follow the BOD mode. If the BOD is disabled, the VLM will not be enabled, even if the VLMIE is '1'. If the BOD is using Sampled mode, the VLM will also be sampled. When the VLM interrupt is enabled, the interrupt flag will be set according to VLMCFG when the voltage level is crossing the VLM level.

The VLM threshold is defined by writing the VLM Level (VLMLVL) bits in the Control A (BOD.VLMCTRLA) register.

17.3.2 Interrupts

Table 17-1. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
VLM	Voltage Level Monitor	Supply voltage crossing the VLM threshold as configured by the VLM Configuration (VLMCFG) bit field in the Interrupt Control (BOD.INTCTRL) register

The VLM interrupt will not be executed if the CPU is halted in Debug mode.

When an interrupt condition occurs, the corresponding interrupt flag is set in the peripheral's Interrupt Flags (*peripheral*.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

17.3.3 Sleep Mode Operation

The BOD configuration in the different sleep modes is defined by fuses. The mode used in Active mode and Idle sleep mode is defined by the ACTIVE fuses in FUSE.BODCFG, which is loaded into the ACTIVE bit field in the Control A (BOD.CTRLA) register. The mode used in Standby sleep mode and Power-Down sleep mode is defined by SLEEP in FUSE.BODCFG, which is loaded into the SLEEP bit field in the Control A (BOD.CTRLA) register.

The operating mode in Active mode and Idle sleep mode (i.e., ACTIVE in BOD.CTRLA) cannot be altered by software. The operating mode in Standby sleep mode and Power-Down sleep mode can be altered by writing to the SLEEP bit field in the Control A (BOD.CTRLA) register.

When the device is going into Standby or Power-Down sleep mode, the BOD will change the operation mode as defined by SLEEP in BOD.CTRLA. When the device is waking up from Standby or Power-Down sleep mode, the BOD will operate in the mode defined by the ACTIVE bit field in the Control A (BOD.CTRLA) register.

17.3.4 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 17-2. Registers Under Configuration Change Protection

Register	Key
SLEEP in BOD.CTRLA	IOREG

17.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0				SAMPFREQ	ACTIVE[1:0]		SLEEP[1:0]	
0x01	CTRLB	7:0						LVL[2:0]		
0x02 ... 0x07	Reserved									
0x08	VLMCTRLA	7:0							VLMLVL[1:0]	
0x09	INTCTRL	7:0						VLMCFG[1:0]		VLMIE
0x0A	INTFLAGS	7:0								VLMIF
0x0B	STATUS	7:0								VLMS

17.5 Register Description

17.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: Loaded from fuse
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
				SAMPFREQ	ACTIVE[1:0]		SLEEP[1:0]	
Access				R	R	R	R/W	R/W
Reset				x	x	x	x	x

Bit 4 – SAMPFREQ Sample Frequency
This bit controls the BOD sample frequency.
The Reset value is loaded from the SAMPFREQ bit in FUSE.BODCFG.
This bit is not under Configuration Change Protection (CCP).

Value	Description
0x0	Sample frequency is 1 kHz
0x1	Sample frequency is 125 Hz

Bits 3:2 – ACTIVE[1:0] Active
These bits select the BOD operation mode when the device is in Active or Idle mode.
The Reset value is loaded from the ACTIVE bit field in FUSE.BODCFG.
This bit field is not under Configuration Change Protection (CCP).

Value	Name	Description
0x0	DIS	Disabled
0x1	ENABLED	Enabled in continuous mode
0x2	SAMPLED	Enabled in sampled mode
0x3	ENWAKE	Enabled in continuous mode. Execution is halted at wake-up until BOD is running

Bits 1:0 – SLEEP[1:0] Sleep
These bits select the BOD operation mode when the device is in Standby or Power-Down sleep mode. The Reset value is loaded from the SLEEP bit field in FUSE.BODCFG.

Value	Name	Description
0x0	DIS	Disabled
0x1	ENABLED	Enabled in continuous mode
0x2	SAMPLED	Enabled in sampled mode
0x3	-	Reserved

17.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: Loaded from fuse
Property: -

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

Bits 2:0 – LVL[2:0] BOD Level
This bit field controls the BOD threshold level.
The Reset value is loaded from the BOD Level (LVL) bits in the BOD Configuration Fuse (FUSE.BODCFG).

Value	Name	Description
0x0	BODLEVEL0	1.8V
0x2	BODLEVEL2	2.6V
0x7	BODLEVEL7	4.2V

Note:

- Refer to the *BOD and POR Characteristics* in *Electrical Characteristics* for further details
- Values in the description are typical values

17.5.3 VLM Control A

Name: VLMCTRLA
Offset: 0x08
Reset: 0x00
Property: -

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

Bits 1:0 – VLMLVL[1:0] VLM Level
These bits select the VLM threshold relative to the BOD threshold (LVL in BOD.CTRLB).

Value	Description
0x0	VLM threshold 5% above BOD threshold
0x1	VLM threshold 15% above BOD threshold
0x2	VLM threshold 25% above BOD threshold
other	Reserved

17.5.4 Interrupt Control

Name: INTCTRL
Offset: 0x09
Reset: 0x00
Property: -

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

Bits 2:1 – VLMCFG[1:0] VLM Configuration
These bits select which incidents will trigger a VLM interrupt.

Value	Name	Description
0x0	BELOW	V_{DD} falls below VLM threshold
0x1	ABOVE	V_{DD} rises above VLM threshold
0x2	CROSS	V_{DD} crosses VLM threshold
Other	-	Reserved

Bit 0 – VLMIE VLM Interrupt Enable
Writing a '1' to this bit enables the VLM interrupt.

17.5.5 VLM Interrupt Flags

Name: INTFLAGS
Offset: 0x0A
Reset: 0x00
Property: -

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

Bit 0 – VLMIF VLM Interrupt Flag
This flag is set when a trigger from the VLM is given, as configured by the VLMCFG bit in the BOD.INTCTRL register. The flag is only updated when the BOD is enabled.

17.5.6 VLM Status

Name: STATUS
Offset: 0x0B
Reset: 0x00
Property: -

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

Bit 0 – VLMS VLM Status
This bit is only valid when the BOD is enabled.

Value	Description
0	The voltage is above the VLM threshold level
1	The voltage is below the VLM threshold level

18. VREF - Voltage Reference

18.1 Features

- Programmable Voltage Reference Sources:
  - One for each ADC peripheral
  - One for each AC and DAC peripheral
- Each Reference Source Supports Five Different Voltages:
  - 0.55V
  - 1.1V
  - 1.5V
  - 2.5V
  - 4.3V

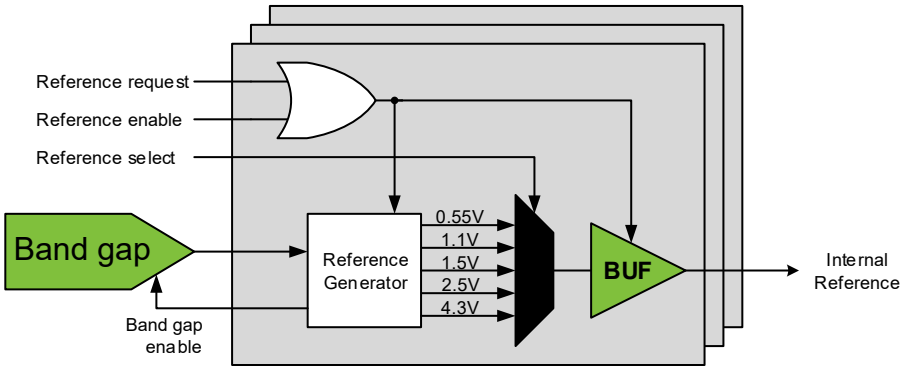
18.2 Overview

The Voltage Reference (VREF) peripheral provides control registers for the voltage reference sources used by several peripherals. The user can select the reference voltages for the ADCn by writing to the ADCn Reference Select (ADCnREFSEL) bit field in the Control x (VREF.CTRLx) registers, and for both ACn and DACn by writing to the DACn Reference Select (DACnREFSEL) bit field in the Control x (VREF.CTRLx) registers.

A voltage reference source is enabled automatically when requested by a peripheral. The user can enable the reference voltage sources (and thus, override the automatic disabling of unused sources) by writing to the respective Force Enable (ADCnREFEN, DACnREFEN) bit in the Control B (VREF.CTRLB) register. This may be done to decrease the start-up time, at the cost of increased power consumption.

18.2.1 Block Diagram

Figure 18-1. VREF Block Diagram



18.3 Functional Description

18.3.1 Initialization

The default configuration will enable the respective source when the ADCn, ACn, or DACn is requesting a reference voltage. The default reference voltages are 0.55V but can be configured by writing to the respective Reference Select (ADCnREFSEL, DACnREFSEL) bit field in the Control (VREF.CTRLx) registers.

18.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		ADC0REFSEL[2:0]				DAC0REFSEL[2:0]		
0x01	CTRLB	7:0			DAC2REFEN	ADC1REFEN	DAC1REFEN		ADC0REFEN	DAC0REFEN
0x02	CTRLC	7:0		ADC1REFSEL[2:0]				DAC1REFSEL[2:0]		
0x03	CTRLD	7:0						DAC2REFSEL[2:0]		

18.5 Register Description

18.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		ADC0REFSEL[2:0]				DAC0REFSEL[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 6:4 – ADC0REFSEL[2:0] ADC0 Reference Select
This bit field selects the reference voltage for the ADC0.

Value	Description
0x0	0.55V
0x1	1.1V
0x2	2.5V
0x3	4.3V
0x4	1.5V
other	Reserved

Bits 2:0 – DAC0REFSEL[2:0] DAC0 and AC0 Reference Select
This bit field selects the reference voltage for the DAC0 and AC0.

Value	Description
0x0	0.55V
0x1	1.1V
0x2	2.5V
0x3	4.3V
0x4	1.5V
other	Reserved

18.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
			DAC2REFEN	ADC1REFEN	DAC1REFEN		ADC0REFEN	DAC0REFEN
Access			R/W	R/W	R/W		R/W	R/W
Reset			0	0	0		0	0

Bit 5 – DAC2REFEN DAC2 and AC2 Reference Force Enable
Writing a '1' to this bit forces the voltage reference for the DAC2 and AC2 to be running, even if it is not requested.
Writing a '0' to this bit allows to automatic enable/disable the reference source when not requested.

Bit 4 – ADC1REFEN ADC1 Reference Force Enable
Writing a '1' to this bit forces the voltage reference for the ADC1 to be running, even if it is not requested.
Writing a '0' to this bit allows to automatic enable/disable the reference source when not requested.
Note: Do not force the internal reference enabled (ADCnREFEN=1 in VREF.CTRLB) when the ADC is using the external reference (the REFSEL bit field in ADC.CTRLC).

Bit 3 – DAC1REFEN DAC1 and AC1 Reference Force Enable
Writing a '1' to this bit forces the voltage reference for the DAC1 and AC1 to be running, even if it is not requested.
Writing a '0' to this bit allows to automatic enable/disable the reference source when not requested.

Bit 1 – ADC0REFEN ADC0 Reference Force Enable
Writing a '1' to this bit forces the voltage reference for the ADC0 to be running, even if it is not requested.
Writing a '0' to this bit allows automatic enable/disable of the reference source by the peripheral.
Note: Do not force the internal reference enabled (ADCnREFEN=1 in VREF.CTRLB) when the ADC is using the external reference (the REFSEL bit field in ADC.CTRLC).

Bit 0 – DAC0REFEN DAC0 and AC0 Reference Force Enable
Writing a '1' to this bit forces the voltage reference for the DAC0 and AC0 to be running, even if it is not requested.
Writing a '0' to this bit allows automatic enable/disable of the reference source by the peripheral.

18.5.3 Control C

Name: CTRLC
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		ADC1REFSEL[2:0]				DAC1REFSEL[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 6:4 – ADC1REFSEL[2:0] ADC1 Reference Select
This bit field selects the reference voltage for the ADC1.

Value	Description
0x0	0.55V
0x1	1.1V
0x2	2.5V
0x3	4.3V
0x4	1.5V
other	Reserved

Bits 2:0 – DAC1REFSEL[2:0] DAC1 and AC1 Reference Select
This bit field selects the reference voltage for the DAC1 and AC1.

Value	Description
0x0	0.55V
0x1	1.1V
0x2	2.5V
0x3	4.3V
0x4	1.5V
other	Reserved

18.5.4 Control D

Name: CTRLD
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
						DAC2REFSEL[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – DAC2REFSEL[2:0] DAC2 and AC2 Reference Select
This bit field selects the reference voltage for the DAC2 and AC2.

Value	Description
0x0	0.55V
0x1	1.1V
0x2	2.5V
0x3	4.3V
0x4	1.5V
other	Reserved

19. WDT - Watchdog Timer

19.1 Features

- Issues a System Reset if the Watchdog Timer is not Cleared Before its Time-out Period
- Operating Asynchronously from System Clock Using an Independent Oscillator
- Using the 1.024 kHz Output of the 32.768 kHz Ultra Low-Power Oscillator (OSCULP32K)
- 11 Selectable Time-out Periods, from 8 ms to 8s
- Two Operation Modes:
  - Normal mode
  - Window mode
- Configuration Lock to Prevent Unwanted Changes
- Closed Period Timer Activation After First WDT Instruction for Easy Setup

19.2 Overview

The Watchdog Timer (WDT) is a system function for monitoring correct program operation. It allows the system to recover from situations such as runaway or deadlocked code by issuing a Reset. When enabled, the WDT is a constantly running timer configured to a predefined time-out period. If the WDT is not reset within the time-out period, it will issue a system Reset. The WDT is reset by executing the Watchdog Timer Reset (`WDR`) instruction from software.

The WDT has two modes of operation: Normal mode and Window mode. The settings in the Control A (WDT.CTRLA) register determine the mode of operation.

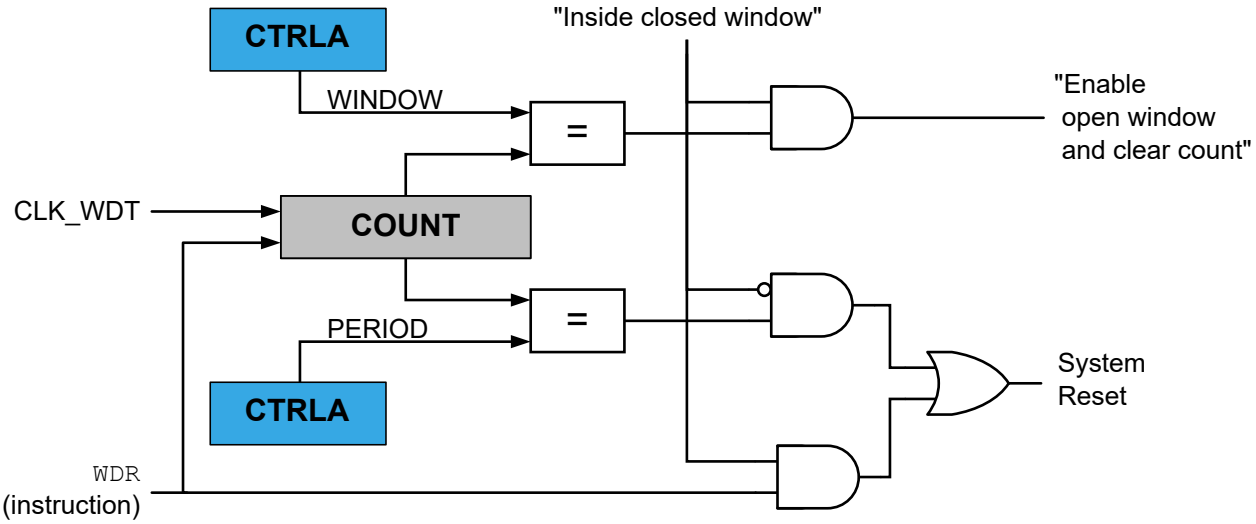
A Window mode defines a time slot or "window" inside the time-out period during which the WDT must be reset. If the WDT is reset outside this window, either too early or too late, a system Reset will be issued. Compared to the Normal mode, the Window mode can catch situations where a code error causes constant `WDR` execution.

When enabled, the WDT will run in Active mode and all sleep modes. It is asynchronous (i.e., running from a CPU independent clock source). For this reason, it will continue to operate and be able to issue a system Reset even if the main clock fails.

The CCP mechanism ensures that the WDT settings cannot be changed by accident. For increased safety, a configuration for locking the WDT settings is available.

19.2.1 Block Diagram

Figure 19-1. WDT Block Diagram



19.2.2 Signal Description

Not applicable.

19.3 Functional Description

19.3.1 Initialization

- The WDT is enabled when a non-zero value is written to the Period (PERIOD) bits in the Control A (WDT.CTRLA) register
- Optional: Write a non-zero value to the Window (WINDOW) bits in WDT.CTRLA to enable the Window mode operation.

All bits in the Control A register and the Lock (LOCK) bit in the STATUS (WDT.STATUS) register are write-protected by the Configuration Change Protection mechanism.

The Reset value of WDT.CTRLA is defined by a fuse (FUSE.WDTCFG), so the WDT can be enabled at boot time. If this is the case, the LOCK bit in WDT.STATUS is set at boot time.

19.3.2 Clocks

A 1.024 kHz Oscillator Clock (CLK_WDT_OSC) is sourced from the internal Ultra Low-Power Oscillator, OSCULP32K. Due to the ultra low-power design, the oscillator is not very accurate, and so the exact time-out period may vary from device to device. This variation must be kept in mind when designing software that uses the WDT to ensure that the time-out periods used are valid for all devices.

The 1.024 kHz Oscillator Clock, CLK_WDT_OSC, is asynchronous to the system clock. Due to this asynchronicity, writing to the WDT Control register will require synchronization between the clock domains.

19.3.3 Operation

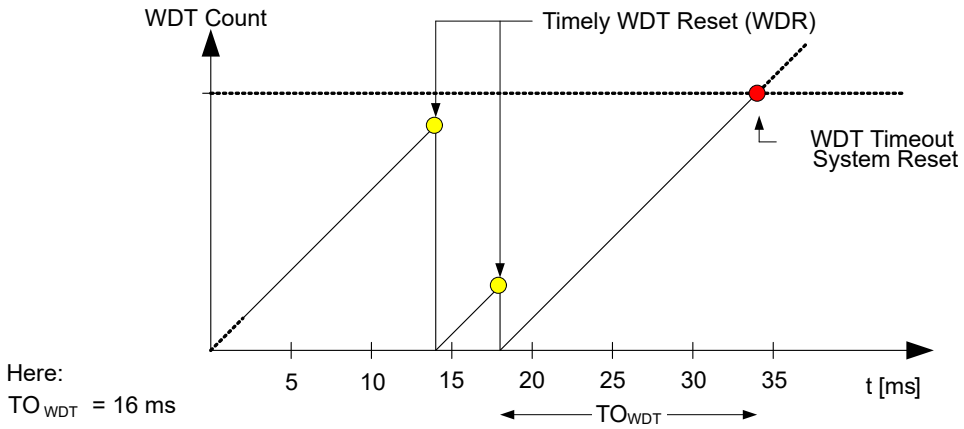
19.3.3.1 Normal Mode

In Normal mode operation, a single time-out period is set for the WDT. If the WDT is not reset from software using the `WDR` any time before the time-out occurs, the WDT will issue a system Reset.

A new WDT time-out period will be started each time the WDT is reset by `WDR`.

There are 11 possible WDT time-out periods (TO_{WDT}), selectable from 8 ms to 8s by writing to the Period (PERIOD) bit field in the Control A (WDT.CTRLA) register.

Figure 19-2. Normal Mode Operation



Normal mode is enabled as long as the WINDOW bit field in the Control A (WDT.CTRLA) register is `0x0`.

19.3.3.2 Window Mode

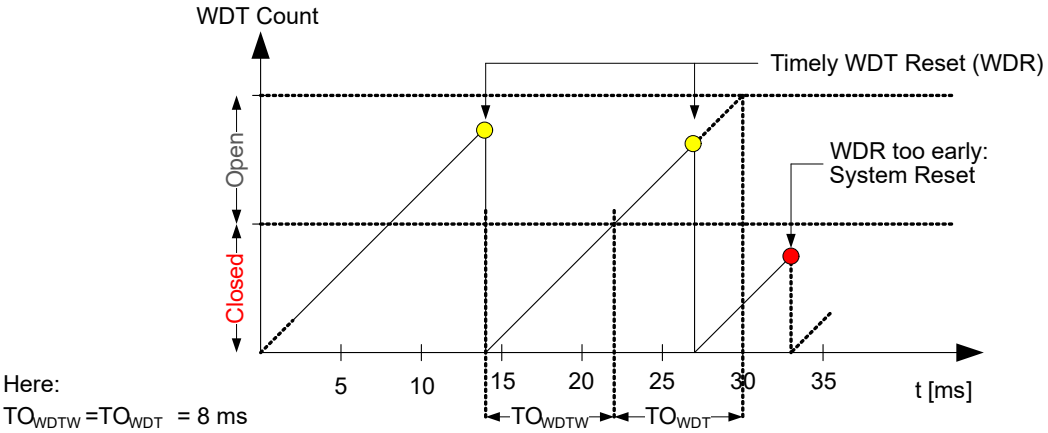
In Window mode operation, the WDT uses two different time-out periods: A closed Window Time-out period (TO_{WDTW}) and the normal time-out period (TO_{WDT}):

- The closed window time-out period defines a duration from 8 ms to 8s, where the WDT cannot be reset. If the WDT is reset during this period, the WDT will issue a system Reset.
- The normal WDT time-out period, which is also 8 ms to 8s, defines the duration of the open period during which the WDT can (and needs to) be reset. The open period will always follow the closed period, so the total duration of the time-out period is the sum of the closed window and the open window time-out periods.

When enabling Window mode or when going out of Debug mode, the first closed period is activated after the first `WDR` instruction.

If a second `WDR` is issued while a previous `WDR` is being synchronized, the second one will be ignored.

Figure 19-3. Window Mode Operation



The Window mode is enabled by writing a non-zero value to the WINDOW bit field in the Control A (WDT.CTRLA) register, and disabled by writing it to `0x0`.

19.3.3.3 Configuration Protection and Lock

The WDT provides two security mechanisms to avoid unintentional changes to the WDT settings.

The first mechanism is the Configuration Change Protection mechanism, employing a timed-write procedure for changing the WDT control registers.

The second mechanism locks the configuration by writing a '`1`' to the LOCK bit in the STATUS (WDT.STATUS) register. When this bit is '`1`', the Control A (WDT.CTRLA) register cannot be changed. Consequently, the WDT cannot be disabled from the software.

LOCK in WDT.STATUS can only be written to '1'. It can only be cleared in Debug mode.

If the WDT configuration is loaded from fuses, LOCK is automatically set in WDT.STATUS.

19.3.4 Sleep Mode Operation

The WDT will continue to operate in any sleep mode where the source clock is Active.

19.3.5 Debug Operation

When run-time debugging, this peripheral will continue normal operation. Halting the CPU in Debugging mode will halt the normal operation of the peripheral.

When halting the CPU in Debug mode, the WDT counter is reset.

When starting the CPU again and the WDT is operating in Window mode, the first closed window time-out period will be disabled, and a Normal mode time-out period is executed.

19.3.6 Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domain, the Control A (WDT.CTRLA) register is synchronized when written. The Synchronization Busy (SYNCBUSY) flag in the STATUS (WDT.STATUS) register indicates if there is an ongoing synchronization.

Writing to WDT.CTRLA while SYNCBUSY=1 is not allowed.

The following registers are synchronized when written:

- PERIOD bits in Control A (WDT.CTRLA) register
- Window Period (WINDOW) bits in WDT.CTRLA

The WDR instruction will need two to three cycles of the WDT clock to be synchronized. Issuing a new WDR instruction while a WDR instruction is being synchronized will be ignored.

19.3.7 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 19-1. WDT - Registers Under Configuration Change Protection

Register	Key
WDT.CTRLA	IOREG
LOCK bit in WDT.STATUS	IOREG

List of bits/registers protected by CCP:

- Period bits in Control A (CTRLA.PERIOD) register
- Window Period bits in Control A (CTRLA.WINDOW) register
- LOCK bit in STATUS (STATUS.LOCK) register

19.4 Register Summary - WDT

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	WINDOW[3:0]				PERIOD[3:0]			
0x01	STATUS	7:0	LOCK							SYNCBUSY

19.5 Register Description

19.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: From FUSE.WDTCFG
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
	WINDOW[3:0]				PERIOD[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
Writing a non-zero value to these bits enables the Window mode, and selects the duration of the closed period accordingly.
The bits are optionally lock-protected:

- If LOCK bit in WDT.STATUS is '1', all bits are change-protected (Access = R)
- If LOCK bit in WDT.STATUS is '0', all bits can be changed (Access = R/W)

Value	Name	Description
0x0	OFF	-
0x1	8CLK	0.008s
0x2	16CLK	0.016s
0x3	32CLK	0.031s
0x4	64CLK	0.063s
0x5	128CLK	0.125s
0x6	256CLK	0.25s
0x7	512CLK	0.5s
0x8	1KCLK	1s
0x9	2KCLK	2s
0xA	4KCLK	4s
0xB	8KCLK	8s
other	-	Reserved

Bits 3:0 – PERIOD[3:0] Period
Writing a non-zero value to this bit enables the WDT, and selects the time-out period in Normal mode accordingly. In Window mode, these bits select the duration of the open window.
The bits are optionally lock-protected:

- If LOCK in WDT.STATUS is '1', all bits are change-protected (Access = R)
- If LOCK in WDT.STATUS is '0', all bits can be changed (Access = R/W)

Value	Name	Description
0x0	OFF	-
0x1	8CLK	0.008s
0x2	16CLK	0.016s
0x3	32CLK	0.031s
0x4	64CLK	0.063s
0x5	128CLK	0.125s
0x6	256CLK	0.25s
0x7	512CLK	0.5s
0x8	1KCLK	1s
0x9	2KCLK	2s
0xA	4KCLK	4s
0xB	8KCLK	8s
other	-	Reserved

19.5.2 Status

Name: STATUS
Offset: 0x01
Reset: 0x00
Property: Configuration Change Protection

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

Bit 7 – LOCK Lock
Writing this bit to '1' write-protects the WDT.CTRLA register.
It is only possible to write this bit to '1'. This bit can be cleared in Debug mode only.
If the PERIOD bits in WDT.CTRLA are different from zero after boot code, the lock will be automatically set.
This bit is under CCP.

Bit 0 – SYNCBUSY Synchronization Busy
This bit is set after writing to the WDT.CTRLA register, while the data is being synchronized from the system clock domain to the WDT clock domain.
This bit is cleared by the system after the synchronization is finished.
This bit is not under CCP.

20. TCA - 16-bit Timer/Counter Type A

20.1 Features

- 16-Bit Timer/Counter
- Three Compare Channels
- Double-Buffered Timer Period Setting
- Double-Buffered Compare Channels
- Waveform Generation:
  - Frequency generation
  - Single-slope PWM (Pulse-Width Modulation)
  - Dual-slope PWM
- Count on Event
- Timer Overflow Interrupts/Events
- One Compare Match per Compare Channel
- Two 8-Bit Timer/Counters in Split Mode

20.2 Overview

The flexible 16-bit PWM Timer/Counter type A (TCA) provides accurate program execution timing, frequency and waveform generation, and command execution.

A TCA consists of a base counter and a set of compare channels. The base counter can be used to count clock cycles or events, or let events control how it counts clock cycles. It has direction control and period setting that can be used for timing. The compare channels can be used together with the base counter to do compare match control, frequency generation, and pulse-width waveform modulation.

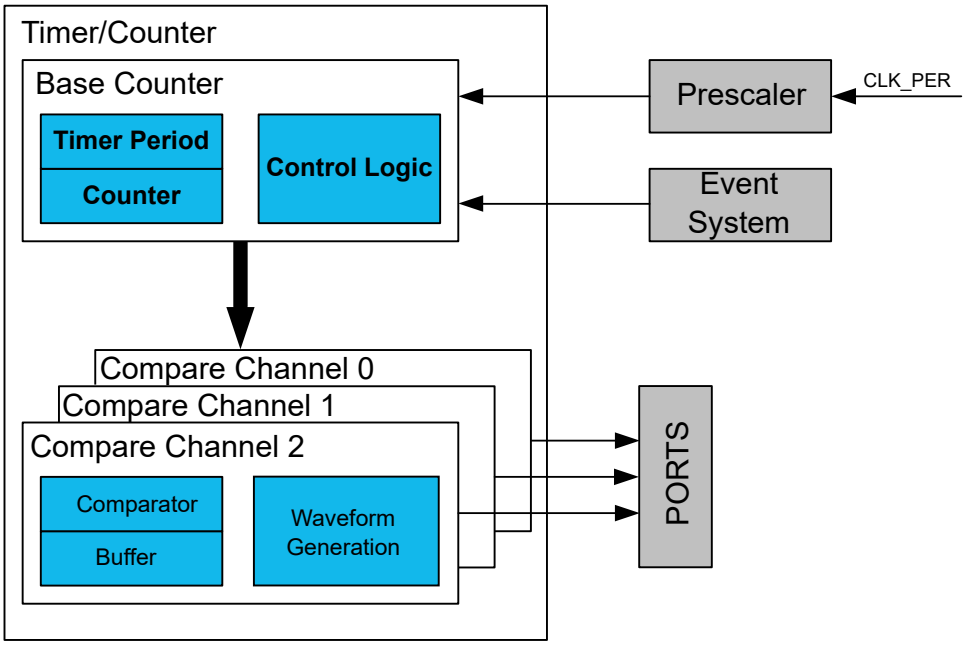
Depending on the mode of operation, the counter is cleared, reloaded, incremented, or decremented at each timer/counter clock or event input.

A timer/counter can be clocked and timed from the peripheral clock, with optional prescaling, or from the Event System. The Event System can also be used for direction control or to synchronize operations.

By default, the TCA is a 16-bit timer/counter. The timer/counter has a Split mode feature that splits it into two 8-bit timer/counters with three compare channels each.

A block diagram of the 16-bit timer/counter with closely related peripheral modules (in grey) is shown in the figure below.

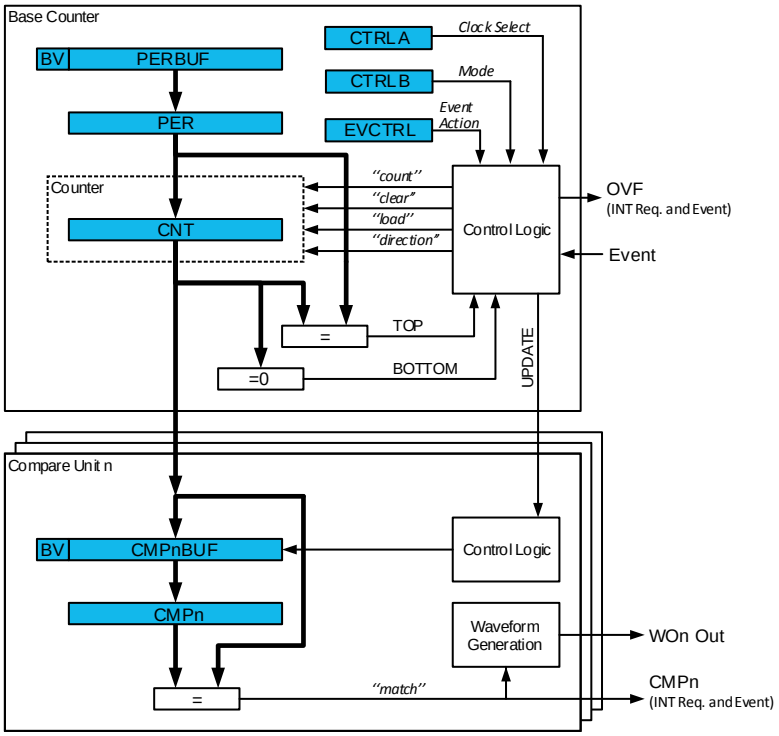
Figure 20-1. 16-bit Timer/Counter and Closely Related Peripherals



20.2.1 Block Diagram

The figure below shows a detailed block diagram of the timer/counter.

Figure 20-2. Timer/Counter Block Diagram



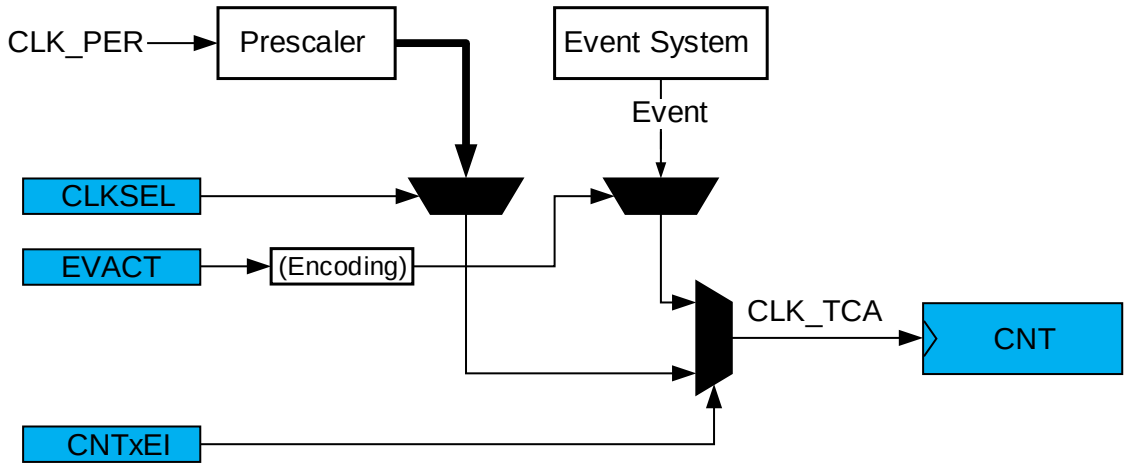
The Counter (TCAn.CNT) register, Period and Compare (TCAn.PER and TCAn.CMPn) registers, and their corresponding buffer registers (TCAn.PERBUF and TCAn.CMPnBUF) are 16-bit registers. All buffer registers have a Buffer Valid (BV) flag that indicates when the buffer contains a new value.

During normal operation, the counter value is continuously compared to zero and the period (PER) value to determine whether the counter has reached TOP or BOTTOM. The counter value can also be compared to the TCAn.CMPn registers.

The timer/counter can generate interrupt requests, events, or change the waveform output after being triggered by the Counter (TCAn.CNT) register reaching TOP, BOTTOM, or CMPn. The interrupt requests, events, or waveform output changes will occur on the next CLK_TCA cycle after the triggering.

CLK_TCA is either the prescaled peripheral clock or events from the Event System, as shown in the figure below.

Figure 20-3. Timer/Counter Clock Logic



20.2.2 Signal Description

Signal	Description	Type
WOn	Digital output	Waveform output

20.3 Functional Description

20.3.1 Definitions

The following definitions are used throughout the documentation:

Table 20-1. Timer/Counter Definitions

Name	Description
BOTTOM	The counter reaches BOTTOM when it becomes 0x0000.
MAX	The counter reaches MAXimum when it becomes all ones.
TOP	The counter reaches TOP when it becomes equal to the highest value in the count sequence.
UPDATE	The update condition is met when the timer/counter reaches BOTTOM or TOP, depending on the Waveform Generator mode. Buffered registers with valid buffer values will be updated unless the Lock Update (LUPD) bit in the TCAn.CTRLE register has been set.

.....continued

Name	Description
CNT	Counter register value.
CMP	Compare register value.
PER	Period register value.

In general, the term timer is used when the timer/counter is counting periodic clock ticks. The term counter is used when the input signal has sporadic or irregular ticks. The latter can be the case when counting events.

20.3.2 Initialization

To start using the timer/counter in a basic mode, follow these steps:

1. Write a TOP value to the Period (TCAn.PER) register.
2. Enable the peripheral by writing a '1' to the ENABLE bit in the Control A (TCAn.CTRLA) register.
   The counter will start counting clock ticks according to the prescaler setting in the Clock Select (CLKSEL) bit field in the TCAn.CTRLA register.
3. Optional: By writing a '1' to the Enable Count on Event Input (CNTEI) bit in the Event Control (TCAn.EVCTRL) register, events are counted instead of clock ticks.
4. The counter value can be read from the Counter (CNT) bit field in the Counter (TCAn.CNT) register.

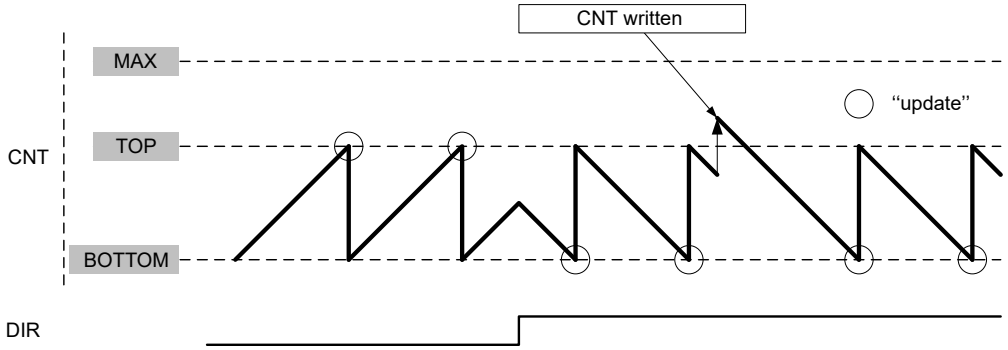
20.3.3 Operation

20.3.3.1 Normal Operation

In normal operation, the counter is counting clock ticks in the direction selected by the Direction (DIR) bit in the Control E (TCAn.CTRLE) register, until it reaches TOP or BOTTOM. The clock ticks are given by the peripheral clock (CLK_PER), prescaled according to the Clock Select (CLKSEL) bit field in the Control A (TCAn.CTRLA) register.

When TOP is reached while the counter is counting up, the counter will wrap to '0' at the next clock tick. When counting down, the counter is reloaded with the Period (TCAn.PER) register value when BOTTOM is reached.

Figure 20-4. Normal Operation



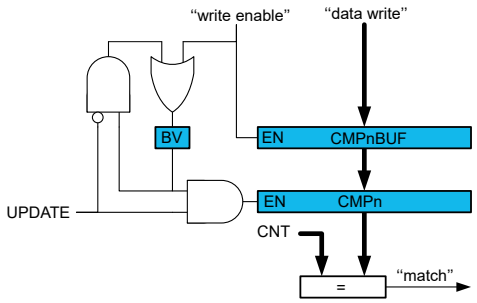
It is possible to change the counter value in the Counter (TCAn.CNT) register when the counter is running. The write access to TCAn.CNT has higher priority than count, clear or reload, and will be immediate. The direction of the counter can also be changed during normal operation by writing to DIR in TCAn.CTRLE.

20.3.3.2 Double Buffering

The Period (TCAn.PER) register value and the Compare n (TCAn.CMPn) register values are all double-buffered (TCAn.PERBUF and TCAn.CMPnBUF).

Each buffer register has a Buffer Valid (BV) flag (PERBV, CMPnBV) in the Control F (TCAn.CTRLF) register, which indicates that the buffer register contains a valid (new) value that can be copied into the corresponding Period or Compare register. When the Period register and Compare n registers are used for a compare operation, the BV flag is set when data are written to the buffer register and cleared on an UPDATE condition. This is shown for a Compare (CMPn) register in the figure below.

Figure 20-5. Period and Compare Double Buffering



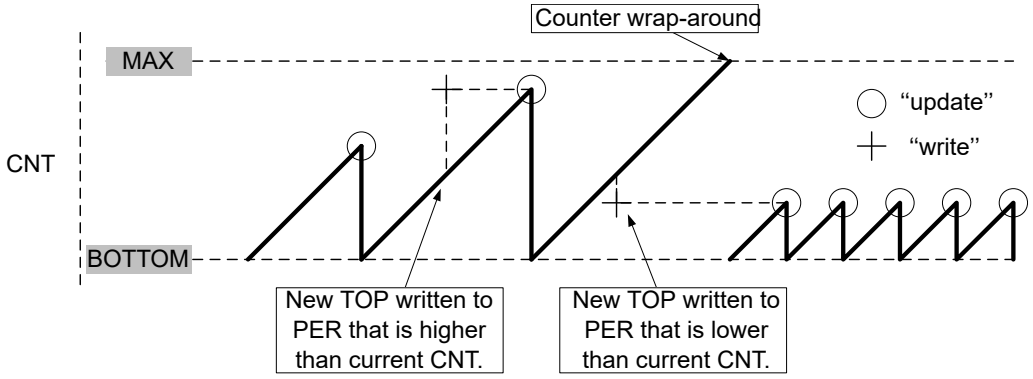
Both the TCAn.CMPn and TCAn.CMPnBUF registers are available as I/O registers. This allows initialization and bypassing of the buffer register and the double-buffering function.

20.3.3.3 Changing the Period

The Counter period is changed by writing a new TOP value to the Period (TCAn.PER) register.

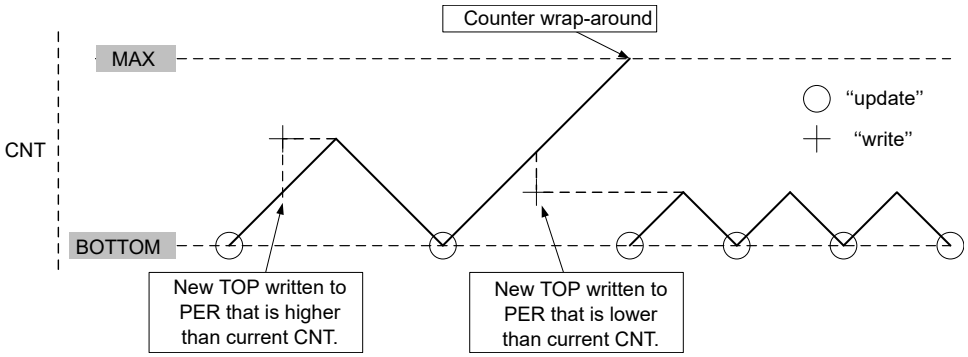
No Buffering: If double-buffering is not used, any period update is immediate.

Figure 20-6. Changing the Period Without Buffering



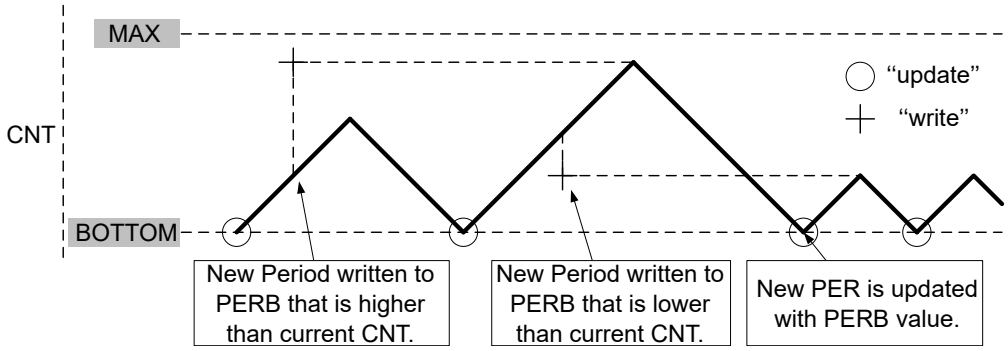
A counter wrap-around can occur in any mode of operation when counting up without buffering, as the TCAn.CNT and TCAn.PER registers are continuously compared. If a new TOP value is written to TCAn.PER that is lower than the current TCAn.CNT, the counter will wrap first, before a compare match occurs.

Figure 20-7. Unbuffered Dual-Slope Operation



With Buffering: When double-buffering is used, the buffer can be written at any time and still maintain correct operation. The TCAn.PER is always updated on the UPDATE condition, as shown for dual-slope operation in the figure below. This prevents wrap-around and the generation of odd waveforms.

Figure 20-8. Changing the Period Using Buffering



Note: Buffering is used in figures illustrating TCA operation if not otherwise specified.

20.3.3.4 Compare Channel

Each Compare Channel n continuously compares the counter value (TCAn.CNT) with the Compare n (TCAn.CMPn) register. If TCAn.CNT equals TCAn.CMPn, the Comparator n signals a match. The match will set the Compare Channel's interrupt flag at the next timer clock cycle, and the optional interrupt is generated.

The Compare n Buffer (TCAn.CMPnBUF) register provides double-buffer capability equivalent to that for the period buffer. The double-buffering synchronizes the update of the TCAn.CMPn register with the buffer value to either the TOP or BOTTOM of the counting sequence, according to the UPDATE condition. The synchronization prevents the occurrence of odd-length, non-symmetrical pulses for glitch-free output.

20.3.3.4.1 Waveform Generation

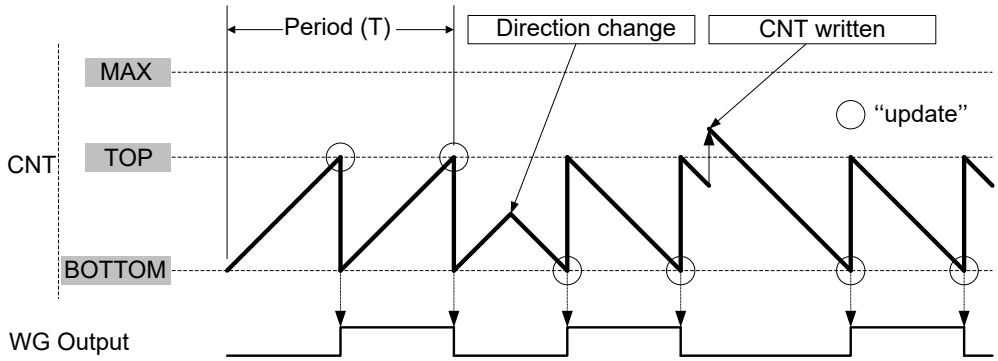
The compare channels can be used for waveform generation on the corresponding port pins. The following requirements must be met to make the waveform visible on the connected port pin:

1. A Waveform Generation mode must be selected by writing the WGMODE bit field in TCAn.CTRLB.
2. The compare channels used must be enabled (CMPnEN = `1` in TCAn.CTRLB). This will override the output value for the corresponding pin. An alternative pin can be selected by configuring the Port Multiplexer (PORTMUX). Refer to the *PORTMUX* chapter for details.
3. The direction for the associated port pin n must be configured as an output (PORTx.DIR[n] = `1`).
4. Optional: Enable the inverted waveform output for the associated port pin n (INVEN = `1` in PORTx.PINnCTRL).

20.3.3.4.2 Frequency (FRQ) Waveform Generation

For frequency generation, the period time (T) is controlled by the TCAn.CMP0 register instead of the Period (TCAn.PER) register. The corresponding waveform generator output is toggled on each compare match between the TCAn.CNT and TCAn.CMPn registers.

Figure 20-9. Frequency Waveform Generation



The waveform frequency (f_{FRQ}) is defined by the following equation:

$$f_{FRQ} = \frac{f_{CLK\_PER}}{2N(CMPn+1)}$$

where *N* represents the prescaler divider used (see the CLKSEL bit field in the TCAn.CTRLA register), and f_{CLK_PER} is the peripheral clock frequency.

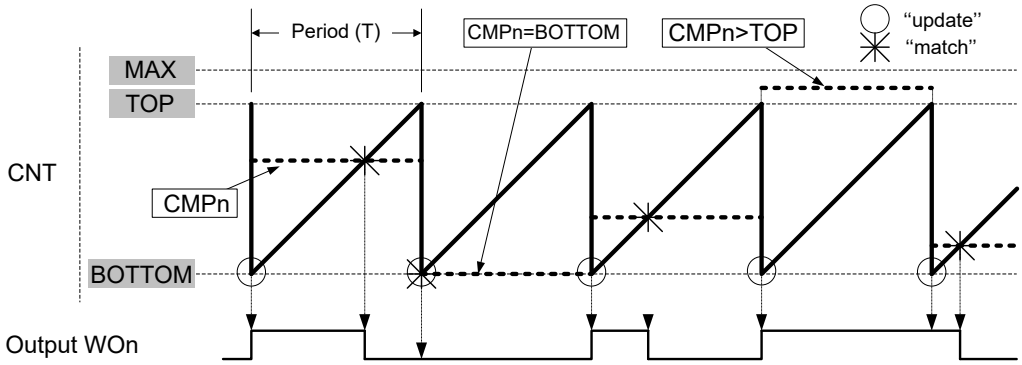
The maximum frequency of the waveform generated is half of the peripheral clock frequency ($f_{CLK\_PER}/2$) when TCAn.CMP0 is written to 0x0000 and no prescaling is used (*N* = 1, CLKSEL = 0x0 in TCAn.CTRLA).

20.3.3.4.3 Single-Slope PWM Generation

For single-slope Pulse-Width Modulation (PWM) generation the period (T) is controlled by the TCAn.PER register, while the values of the TCAn.CMPn registers control the duty cycles of the generated waveforms. The figure below shows how the counter counts from BOTTOM to TOP and then restarts from BOTTOM. The waveform generator output is set at BOTTOM and cleared on the compare match between the TCAn.CNT and TCAn.CMPn registers.

CMPn = BOTTOM will produce a static low signal on WOn while CMPn > TOP will produce a static high signal on WOn.

Figure 20-10. Single-Slope Pulse-Width Modulation



The TCAn.PER register defines the PWM resolution. The minimum resolution is 2 bits (TCAn.PER = 0x0002), and the maximum resolution is 16 bits (TCAn.PER = MAX-1).

The following equation calculates the exact resolution in bits for single-slope PWM (R_{PWM_SS}):

$$R_{PWM\_SS} = \frac{\log(PER+2)}{\log(2)}$$

The single-slope PWM frequency (f_{PWM_SS}) depends on the period setting (TCAn.PER), the peripheral clock frequency f_{CLK_PER} and the TCA prescaler (the CLKSEL bit field in the TCAn.CTRLA register). It is calculated by the following equation where *N* represents the prescaler divider used:

$$f_{PWM\_SS} = \frac{f_{CLK\_PER}}{N(PER+1)}$$

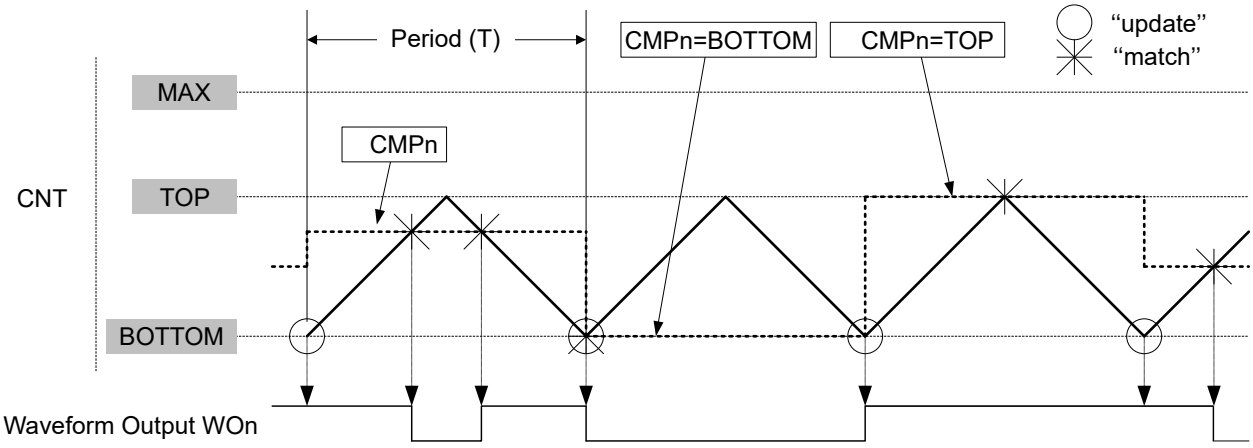
20.3.3.4.4 Dual-Slope PWM

For dual-slope PWM generation, the period (T) is controlled by TCAn.PER, while the values of TCAn.CMPn control the duty cycle of the WG output.

The figure below shows how, for dual-slope PWM, the counter counts repeatedly from BOTTOM to TOP and then from TOP to BOTTOM. The waveform generator output is set at BOTTOM, cleared on compare match when up-counting and set on compare match when down-counting.

CMPn = BOTTOM will produce a static low signal on WOn, while CMPn = TOP will produce a static high signal on WOn.

Figure 20-11. Dual-Slope Pulse-Width Modulation



Using dual-slope PWM results in half the maximum operation frequency compared to single-slope PWM operation, due to twice the number of timer increments per period.

The Period (TCAn.PER) register defines the PWM resolution. The minimum resolution is 2 bits (TCAn.PER = 0x0003), and the maximum resolution is 16 bits (TCAn.PER = MAX).

The following equation calculates the exact resolution in bits for dual-slope PWM (R_{PWM_DS}):

$$R_{\mathrm{PWM\_DS}} = \frac{\log(\mathrm{PER}+1)}{\log(2)}$$

The PWM frequency depends on the period setting in the TCAn.PER register, the peripheral clock frequency (f_{CLK_PER}) and the prescaler divider selected in the CLKSEL bit field in the TCAn.CTRLA register. It is calculated by the following equation:

$$f_{\mathrm{PWM\_DS}} = \frac{f_{\mathrm{CLK\_PER}}}{2N \cdot \mathrm{PER}}$$

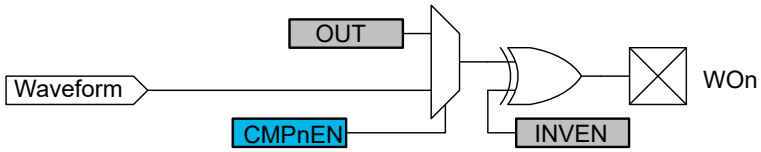
N represents the prescaler divider used.

20.3.3.4.5 Port Override for Waveform Generation

To make the waveform generation available on the port pins, the corresponding port pin direction must be set as output (PORTx.DIR[n] = `1`). The TCA will override the port pin values when the compare channel is enabled (CMPnEN = `1` in TCAn.CTRLB) and a Waveform Generation mode is selected.

The figure below shows the port override for TCA. The timer/counter compare channel will override the port pin output value (OUT) on the corresponding port pin. Enabling inverted I/O on the port pin (INVEN = `1` in PORT.PINn) inverts the corresponding WG output.

Figure 20-12. Port Override for Timer/Counter Type A



20.3.3.5 Timer/Counter Commands

A set of commands can be issued by software to immediately change the state of the peripheral. These commands give direct control of the UPDATE, RESTART and RESET signals. A command is issued by writing the respective value to the Command (CMD) bit field in the Control E (TCAn.CTRLESET) register.

An UPDATE command has the same effect as when an UPDATE condition occurs, except that the UPDATE command is not affected by the state of the Lock Update (LUPD) bit in the Control E (TCAn.CTRLE) register.

The software can force a restart of the current waveform period by issuing a RESTART command. In this case, the counter, direction, and all compare outputs are set to '`0`'.

A RESET command will set all timer/counter registers to their initial values. A RESET command can be issued only when the timer/counter is not running (ENABLE = 0 in the TCAn.CTRLA register).

20.3.3.6 Split Mode - Two 8-Bit Timer/Counters

Split Mode Overview

To double the number of timers and PWM channels in the TCA, a Split mode is provided. In this Split mode, the 16-bit timer/counter acts as two separate 8-bit timers, which each have three compare channels for PWM generation. The Split mode will only work with single-slope down-count. Event controlled operation is not supported in Split mode.

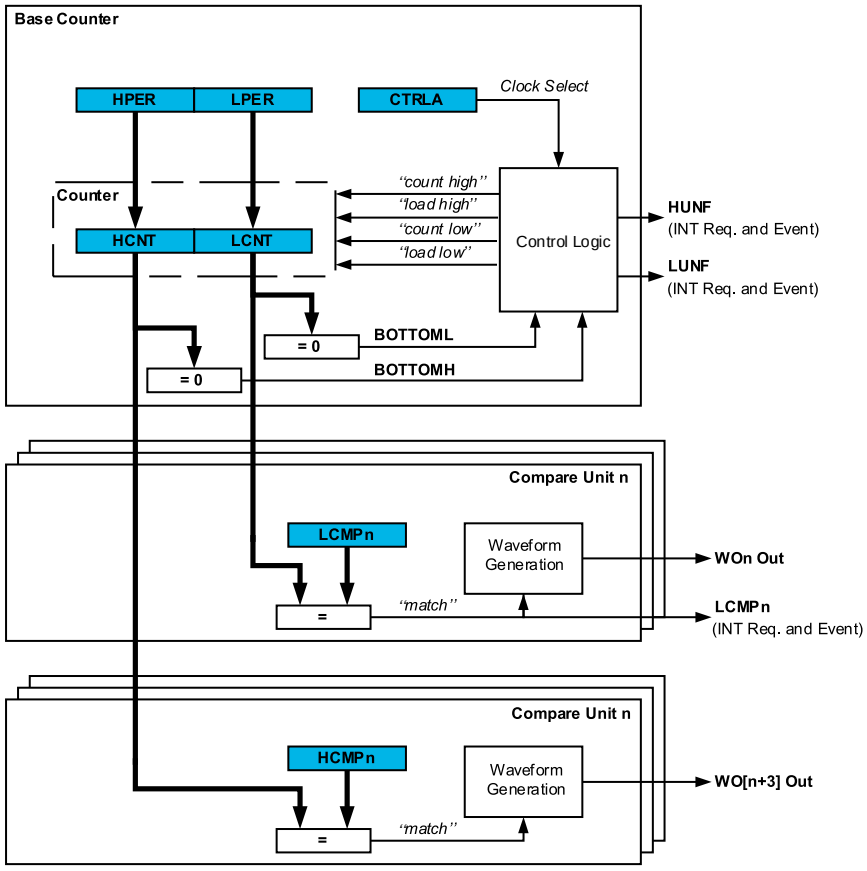
Activating Split mode results in changes to the functionality of some registers and register bits. The modifications are described in a separate register map (see 20.6 Register Summary - Split Mode).

Split Mode Differences Compared to Normal Mode

- Count:
  - Down-count only
  - Low Byte Timer Counter (TCAn.LCNT) register and High Byte Timer Counter (TCAn.HCNT) register are independent
- Waveform Generation:
  - Single-slope PWM only (WGMODE = SINGLESLOPE in TCAn.CTRLB)
- Interrupt:
  - No change for Low Byte Timer Counter (TCAn.LCNT) register
  - Underflow interrupt for High Byte Timer Counter (TCAn.HCNT) register
  - No compare interrupt or flag for High Byte Compare n (TCAn.HCMPn) register
- Event Actions: Not Compatible
- Buffer Registers and Buffer Valid Flags: Unused
- Register Access: Byte Access to All Registers

Block Diagram

Figure 20-13. Timer/Counter Block Diagram Split Mode



Split Mode Initialization

When shifting between Normal mode and Split mode, the functionality of some registers and bits changes, but their values do not. For this reason, disabling the peripheral (ENABLE = 0 in TCAn.CTRLA) and doing a hard Reset (CMD = RESET in TCAn.CTRLESET) is recommended when changing the mode to avoid unexpected behavior.

To start using the timer/counter in basic Split mode after a hard Reset, follow these steps:

1. Enable Split mode by writing a '1' to the Split mode enable (SPLITM) bit in the Control D (TCAn.CTRLD) register.
2. Write a TOP value to the Period (TCAn.PER) registers.
3. Enable the peripheral by writing a '1' to the Enable (ENABLE) bit in the Control A (TCAn.CTRLA) register. The counter will start counting clock ticks according to the prescaler setting in the Clock Select (CLKSEL) bit field in the TCAn.CTRLA register.
4. The counter values can be read from the Counter bit field in the Counter (TCAn.CNT) registers.

20.3.4 Events

The TCA can generate the events described in the table below. All event generators except TCAn_HUNF are shared between Normal mode and Split mode operation.

Table 20-2. Event Generators in TCA

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCAn	OVF_LUNF	Normal mode: Overflow Split mode: Low byte timer underflow	Pulse	CLK_PER	One CLK_PER period
	HUNF	Normal mode: Not available Split mode: High byte timer underflow	Pulse	CLK_PER	One CLK_PER period
	CMP0	Normal mode: Compare Channel 0 match Split mode: Low byte timer Compare Channel 0 match	Pulse	CLK_PER	One CLK_PER period
	CMP1	Normal mode: Compare Channel 1 match Split mode: Low byte timer Compare Channel 1 match	Pulse	CLK_PER	One CLK_PER period
	CMP2	Normal mode: Compare Channel 2 match Split mode: Low byte timer Compare Channel 2 match	Pulse	CLK_PER	One CLK_PER period

Note: The conditions for generating an event are identical to those that will raise the corresponding interrupt flag in the TCAn.INTFLAGS register for both Normal mode and Split mode.

The TCA has one event user for detecting and acting upon input events. The table below describes the event user and the associated functionality.

Table 20-3. Event User in TCA

User Name	Description	Input Detection	Async/Sync
TCAn	Count on a positive event edge	Edge	Sync
	Count on any event edge	Edge	Sync
	Count while the event signal is high	Level	Sync
	The event level controls count direction, up when low and down when high	Level	Sync

The specific actions described in the table above are selected by writing to the Event Action Event Action (EVACT) bits in the Event Control (TCAn.EVCTRL) register. Input events are enabled by writing a '1' to the Enable Count on Event Input (CNTEI) bit in the Event Control (TCAn.EVCTRL) register.

Event inputs are not used in Split mode.

Refer to the *Event System (EVSYS)* chapter for more details regarding event types and Event System configuration.

20.3.5 Interrupts

Table 20-4. Available Interrupt Vectors and Sources in Normal Mode

Name	Vector Description	Conditions
OVF	Overflow or underflow interrupt	The counter has reached TOP or BOTTOM
CMP0	Compare Channel 0 interrupt	Match between the counter value and the Compare 0 register
CMP1	Compare Channel 1 interrupt	Match between the counter value and the Compare 1 register
CMP2	Compare Channel 2 interrupt	Match between the counter value and the Compare 2 register

Table 20-5. Available Interrupt Vectors and Sources in Split Mode

Name	Vector Description	Conditions
LUNF	Low-byte Underflow interrupt	Low byte timer reaches BOTTOM
HUNF	High-byte Underflow interrupt	High byte timer reaches BOTTOM
LCMP0	Compare Channel 0 interrupt	Match between the counter value and the low byte of the Compare 0 register
LCMP1	Compare Channel 1 interrupt	Match between the counter value and the low byte of the Compare 1 register
LCMP2	Compare Channel 2 interrupt	Match between the counter value and the low byte of the Compare 2 register

When an interrupt condition occurs, the corresponding interrupt flag is set in the peripheral's Interrupt Flags (*peripheral*.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

20.3.6 Sleep Mode Operation

The timer/counter will continue operation in Idle sleep mode.

20.4 Register Summary - Normal Mode

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0					CLKSEL[2:0]			ENABLE
0x01	CTRLB	7:0		CMP2EN	CMP1EN	CMP0EN	ALUPD	WGMODE[2:0]		
0x02	CTRLC	7:0						CMP2OV	CMP1OV	CMP0OV
0x03	CTRLD	7:0								SPLITM
0x04	CTRLECLR	7:0					CMD[1:0]		LUPD	DIR
0x05	CTRLESET	7:0					CMD[1:0]		LUPD	DIR
0x06	CTRLFCLR	7:0					CMP2BV	CMP1BV	CMP0BV	PERBV
0x07	CTRLFSET	7:0					CMP2BV	CMP1BV	CMP0BV	PERBV
0x08	Reserved									
0x09	EVCTRL	7:0					EVACT[2:0]			CNTEI
0x0A	INTCTRL	7:0		CMP2	CMP1	CMP0				OVF
0x0B	INTFLAGS	7:0		CMP2	CMP1	CMP0				OVF
0x0C ... 0x0D	Reserved									
0x0E	DBGCTRL	7:0								DBGRUN
0x0F	TEMP	7:0	TEMP[7:0]							
0x10 ... 0x1F	Reserved									
0x20	CNT	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
0x22 ... 0x25	Reserved									
0x26	PER	7:0	PER[7:0]							
		15:8	PER[15:8]							
0x28	CMP0	7:0	CMP[7:0]							
		15:8	CMP[15:8]							
0x2A	CMP1	7:0	CMP[7:0]							
		15:8	CMP[15:8]							
0x2C	CMP2	7:0	CMP[7:0]							
		15:8	CMP[15:8]							
0x2E ... 0x35	Reserved									
0x36	PERBUF	7:0	PERBUF[7:0]							
		15:8	PERBUF[15:8]							
0x38	CMP0BUF	7:0	CMPBUF[7:0]							
		15:8	CMPBUF[15:8]							
0x3A	CMP1BUF	7:0	CMPBUF[7:0]							
		15:8	CMPBUF[15:8]							
0x3C	CMP2BUF	7:0	CMPBUF[7:0]							
		15:8	CMPBUF[15:8]							

20.5 Register Description - Normal Mode

20.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

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

Bits 3:1 – CLKSEL[2:0] Clock Select
These bits select the clock frequency for the timer/counter.

Value	Name	Description
0x0	DIV1	$f_{TCA} = f_{CLK\_PER}$
0x1	DIV2	$f_{TCA} = f_{CLK\_PER}/2$
0x2	DIV4	$f_{TCA} = f_{CLK\_PER}/4$
0x3	DIV8	$f_{TCA} = f_{CLK\_PER}/8$
0x4	DIV16	$f_{TCA} = f_{CLK\_PER}/16$
0x5	DIV64	$f_{TCA} = f_{CLK\_PER}/64$
0x6	DIV256	$f_{TCA} = f_{CLK\_PER}/256$
0x7	DIV1024	$f_{TCA} = f_{CLK\_PER}/1024$

Bit 0 – ENABLE Enable

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

20.5.2 Control B - Normal Mode

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		CMP2EN	CMP1EN	CMP0EN	ALUPD	WGMODE[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 4, 5, 6 – CMPEN Compare n Enable
In the FRQ and PWM Waveform Generation modes the Compare n Enable (CMPnEN) bits will make the waveform output available on the pin corresponding to WOn, overriding the value in the corresponding PORT output register. The corresponding pin direction must be configured as an output in the PORT peripheral.

Value	Description
0	Waveform output WOn will not be available on the corresponding pin
1	Waveform output WOn will override the output value of the corresponding pin

Bit 3 – ALUPD Auto-Lock Update
The Auto-Lock Update bit controls the Lock Update (LUPD) bit in the TCAn.CTRLE register. When ALUPD is written to '1', LUPD will be set to '1' until the Buffer Valid (CMPnBV) bits of all enabled compare channels are '1'. This condition will clear LUPD.
It will remain cleared until the next UPDATE condition, where the buffer values will be transferred to the CMPn registers and LUPD will be set to '1' again. This makes sure that the CMPnBUF register values are not transferred to the CMPn registers until all enabled compare buffers are written.

Value	Description
0	LUPD in TCA.CTRLE is not altered by the system
1	LUPD in TCA.CTRLE is set and cleared automatically

Bits 2:0 – WGMODE[2:0] Waveform Generation Mode
These bits select the Waveform Generation mode and control the counting sequence of the counter, TOP value, UPDATE condition, Interrupt condition, and the type of waveform generated.
No waveform generation is performed in the Normal mode of operation. For all other modes, the waveform generator output will only be directed to the port pins if the corresponding CMPnEN bit has been set. The port pin direction must be set as output.

Table 20-6. Timer Waveform Generation Mode

Value	Group Configuration	Mode of Operation	TOP	UPDATE	OVF
0x0	NORMAL	Normal	PER	TOP(1)	TOP(1)
0x1	FRQ	Frequency	CMP0	TOP(1)	TOP(1)
0x2	-	Reserved	-	-	-
0x3	SINGLESLOPE	Single-slope PWM	PER	BOTTOM	BOTTOM
0x4	-	Reserved	-	-	-
0x5	DSTOP	Dual-slope PWM	PER	BOTTOM	TOP
0x6	DSBOTH	Dual-slope PWM	PER	BOTTOM	TOP and BOTTOM
0x7	DSBOTTOM	Dual-slope PWM	PER	BOTTOM	BOTTOM

Note:
1. When counting up.

20.5.3 Control C - Normal Mode

Name: CTRLC
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
						CMP2OV	CMP1OV	CMP0OV
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 2 – CMP2OV Compare Output Value 2
See CMP0OV.

Bit 1 – CMP1OV Compare Output Value 1
See CMP0OV.

Bit 0 – CMP0OV Compare Output Value 0
The CMPnOV bits allow direct access to the waveform generator's output compare value when the timer/counter is not enabled. This is used to set or clear the WG output value when the timer/counter is not running.

20.5.4 Control D

Name: CTRLD
Offset: 0x03
Reset: 0x00
Property: -

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

Bit 0 – SPLITM Enable Split Mode
This bit sets the timer/counter in Split mode operation. It will then work as two 8-bit timer/counters. The register map will change compared to normal 16-bit mode.

20.5.5 Control Register E Clear - Normal Mode

Name: CTRLECLR
Offset: 0x04
Reset: 0x00
Property: -

This register can be used instead of a Read-Modify-Write (RMW) to clear individual bits by writing a '1' to its bit location.

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

Bits 3:2 – CMD[1:0] Command
These bits are used for software control of update, restart and Reset of the timer/counter. The command bits are always read as '0'.

Value	Name	Description
0x0	NONE	No command
0x1	UPDATE	Force update
0x2	RESTART	Force restart
0x3	RESET	Force hard Reset (ignored if the timer/counter is enabled)

Bit 1 – LUPD Lock Update
Lock update can be used to ensure that all buffers are valid before an update is performed.

Value	Description
0	The buffered registers are updated as soon as an UPDATE condition has occurred
1	No update of the buffered registers is performed, even though an UPDATE condition has occurred

Bit 0 – DIR Counter Direction
Normally this bit is controlled in hardware by the Waveform Generation mode or by event actions, but it can also be changed from software.

Value	Description
0	The counter is counting up (incrementing)
1	The counter is counting down (decrementing)

20.5.6 Control Register E Set - Normal Mode

Name: CTRLESET
Offset: 0x05
Reset: 0x00
Property: -

This register can be used instead of a Read-Modify-Write (RMW) to set individual bits by writing a '1' to its bit location.

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

Bits 3:2 – CMD[1:0] Command
These bits are used for software control of update, restart and Reset the timer/counter. The command bits are always read as '0'.

Value	Name	Description
0x0	NONE	No command
0x1	UPDATE	Force update
0x2	RESTART	Force restart
0x3	RESET	Force hard Reset (ignored if the timer/counter is enabled)

Bit 1 – LUPD Lock Update
Locking the update ensures that all buffers are valid before an update is performed.

Value	Description
0	The buffered registers are updated as soon as an UPDATE condition has occurred
1	No update of the buffered registers is performed, even though an UPDATE condition has occurred

Bit 0 – DIR Counter Direction
Normally this bit is controlled in hardware by the Waveform Generation mode or by event actions, but it can also be changed from software.

Value	Description
0	The counter is counting up (incrementing)
1	The counter is counting down (decrementing)

20.5.7 Control Register F Clear

Name: CTRLFCLR
Offset: 0x06
Reset: 0x00
Property: -

This register can be used instead of a Read-Modify-Write (RMW) to clear individual bits by writing a '1' to its bit location.

Bit	7	6	5	4	3	2	1	0
					CMP2BV	CMP1BV	CMP0BV	PERBV
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – CMP2BV Compare 2 Buffer Valid
See CMP0BV.

Bit 2 – CMP1BV Compare 1 Buffer Valid
See CMP0BV.

Bit 1 – CMP0BV Compare 0 Buffer Valid
The CMPnBV bits are set when a new value is written to the corresponding TCAn.CMPnBUF register. These bits are automatically cleared on an UPDATE condition.

Bit 0 – PERBV Period Buffer Valid
This bit is set when a new value is written to the TCAn.PERBUF register. This bit is automatically cleared on an UPDATE condition.

20.5.8 Control Register F Set

Name: CTRLFSET
Offset: 0x07
Reset: 0x00
Property: -

This register can be used instead of a Read-Modify-Write (RMW) to set individual bits by writing a '1' to its bit location.

Bit	7	6	5	4	3	2	1	0
					CMP2BV	CMP1BV	CMP0BV	PERBV
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – CMP2BV Compare 2 Buffer Valid
See CMP0BV.

Bit 2 – CMP1BV Compare 1 Buffer Valid
See CMP0BV.

Bit 1 – CMP0BV Compare 0 Buffer Valid
The CMPnBV bits are set when a new value is written to the corresponding TCAn.CMPnBUF register. These bits are automatically cleared on an UPDATE condition.

Bit 0 – PERBV Period Buffer Valid
This bit is set when a new value is written to the TCAn.PERBUF register. This bit is automatically cleared on an UPDATE condition.

20.5.9 Event Control

Name: EVCTRL
Offset: 0x09
Reset: 0x00
Property: -

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

Bits 3:1 – EVACT[2:0] Event Action
These bits define what action the counter will take upon certain event conditions.

Value	Name	Description
0x0	EVACT_POSEDGE	Count on positive event edge
0x1	EVACT_ANYEDGE	Count on any event edge
0x2	EVACT_HIGHLVL	Count prescaled clock cycles while the event signal is high
0x3	EVACT_UPDOWN	Count prescaled clock cycles. The event signal controls the count direction, up when low and down when high.
Other		Reserved

Bit 0 – CNTEI Enable Count on Event Input

Value	Description
0	Count on Event input is disabled
1	Count on Event input is enabled according to EVACT bit field

20.5.10 Interrupt Control Register - Normal Mode

Name: INTCTRL
Offset: 0x0A
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		CMP2	CMP1	CMP0				OVF
Access		R/W	R/W	R/W				R/W
Reset		0	0	0				0

Bit 6 – CMP2 Compare Channel 2 Interrupt Enable
See CMP0.

Bit 5 – CMP1 Compare Channel 1 Interrupt Enable
See CMP0.

Bit 4 – CMP0 Compare Channel 0 Interrupt Enable
Writing the CMPn bit to '1' enables the interrupt from Compare Channel n.

Bit 0 – OVF Timer Overflow/Underflow Interrupt Enable
Writing the OVF bit to '1' enables the overflow/underflow interrupt.

20.5.11 Interrupt Flag Register - Normal Mode

Name: INTFLAGS
Offset: 0x0B
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		CMP2	CMP1	CMP0				OVF
Access		R/W	R/W	R/W				R/W
Reset		0	0	0				0

Bit 6 – CMP2 Compare Channel 2 Interrupt Flag
See the CMP0 flag description.

Bit 5 – CMP1 Compare Channel 1 Interrupt Flag
See the CMP0 flag description.

Bit 4 – CMP0 Compare Channel 0 Interrupt Flag
The Compare Interrupt (CMPn) flag is set on a compare match on the corresponding compare channel.
For all modes of operation, the CMPn flag will be set when a compare match occurs between the Count (CNT) register and the corresponding Compare n (CMPn) register. The CMPn flag is not cleared automatically. It will be cleared only by writing a '1' to its bit location.

Bit 0 – OVF Overflow/Underflow Interrupt Flag
This flag is set either on a TOP (overflow) or BOTTOM (underflow) condition, depending on the WGMODE setting.
The OVF flag is not cleared automatically. It will be cleared only by writing a '1' to its bit location.

20.5.12 Debug Control Register

Name: DBGCTRL
Offset: 0x0E
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Run in Debug

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

20.5.13 Temporary Bits for 16-Bit Access

Name: TEMP
Offset: 0x0F
Reset: 0x00
Property: -

The Temporary register is used by the CPU for 16-bit single-cycle access to the 16-bit registers of this peripheral. The register is common for all the 16-bit registers of this peripheral and can be read and written by software. For more details on reading and writing 16-bit registers, refer to *Accessing 16-Bit Registers* in the *AVR CPU* section.

Bit	7	6	5	4	3	2	1	0
	TEMP[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 – TEMP[7:0] Temporary Bits for 16-bit Access

20.5.14 Counter Register - Normal Mode

Name: CNT
Offset: 0x20
Reset: 0x00
Property: -

The TCAn.CNTL and TCAn.CNTH register pair represents the 16-bit value, TCAn.CNT. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

CPU and UPDI write access has priority over internal updates of the register.

Bit	15	14	13	12	11	10	9	8
	CNT[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
	CNT[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 – CNT[15:8] Counter High Byte
These bits hold the MSB of the 16-bit Counter register.

Bits 7:0 – CNT[7:0] Counter Low Byte
These bits hold the LSB of the 16-bit Counter register.

20.5.15 Period Register - Normal Mode

Name: PER
Offset: 0x26
Reset: 0xFFFF
Property: -

TCAn.PER contains the 16-bit TOP value in the timer/counter in all modes of operation, except Frequency Waveform Generation (FRQ).

The TCAn.PERL and TCAn.PERH register pair represents the 16-bit value, TCAn.PER. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

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 15:8 – PER[15:8] Periodic High Byte
These bits hold the MSB of the 16-bit Period register.

Bits 7:0 – PER[7:0] Periodic Low Byte
These bits hold the LSB of the 16-bit Period register.

20.5.16 Compare n Register - Normal Mode

Name: CMPn
Offset: 0x28 + n*0x02 [n=0..2]
Reset: 0x00
Property: -

This register is continuously compared to the counter value. Normally, the outputs from the comparators are used to generate waveforms.

TCAn.CMPn registers are updated with the buffer value from their corresponding TCAn.CMPnBUF register when an UPDATE condition occurs.

The TCAn.CMPnL and TCAn.CMPnH register pair represents the 16-bit value, TCAn.CMPn. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

Bit	15	14	13	12	11	10	9	8
	CMP[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
	CMP[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 – CMP[15:8] Compare High Byte
These bits hold the MSB of the 16-bit Compare register.

Bits 7:0 – CMP[7:0] Compare Low Byte
These bits hold the LSB of the 16-bit Compare register.

20.5.17 Period Buffer Register

Name: PERBUF
Offset: 0x36
Reset: 0xFFFF
Property: -

This register serves as the buffer for the Period (TCAn.PER) register. Writing to this register from the CPU or UPDI will set the Period Buffer Valid (PERBV) bit in the TCAn.CTRLF register.

The TCAn.PERBUFL and TCAn.PERBUFH register pair represents the 16-bit value, TCAn.PERBUF. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

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 15:8 – PERBUF[15:8] Period Buffer High Byte
These bits hold the MSB of the 16-bit Period Buffer register.

Bits 7:0 – PERBUF[7:0] Period Buffer Low Byte
These bits hold the LSB of the 16-bit Period Buffer register.

20.5.18 Compare n Buffer Register

Name: CMPnBUF
Offset: 0x38 + n*0x02 [n=0..2]
Reset: 0x00
Property: -

This register serves as the buffer for the associated Compare n (TCAn.CMPn) register. Writing to this register from the CPU or UPDI will set the Compare Buffer valid (CMPnBV) bit in the TCAn.CTRLF register.

The TCAn.CMPnBUFL and TCAn.CMPnBUFH register pair represents the 16-bit value, TCAn.CMPnBUF. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

Bit	15	14	13	12	11	10	9	8
	CMPBUF[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
	CMPBUF[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 – CMPBUF[15:8] Compare High Byte
These bits hold the MSB of the 16-bit Compare Buffer register.

Bits 7:0 – CMPBUF[7:0] Compare Low Byte
These bits hold the LSB of the 16-bit Compare Buffer register.

20.6 Register Summary - Split Mode

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0					CLKSEL[2:0]			ENABLE
0x01	CTRLB	7:0		HCMP2EN	HCMP1EN	HCMP0EN		LCMP2EN	LCMP1EN	LCMP0EN
0x02	CTRLC	7:0		HCMP2OV	HCMP1OV	HCMP0OV		LCMP2OV	LCMP1OV	LCMP0OV
0x03	CTRLD	7:0								SPLITM
0x04	CTRLECLR	7:0					CMD[1:0]		CMDEN[1:0]	
0x05	CTRLESET	7:0					CMD[1:0]		CMDEN[1:0]	
0x06 ... 0x09	Reserved									
0x0A	INTCTRL	7:0		LCMP2	LCMP1	LCMP0			HUNF	LUNF
0x0B	INTFLAGS	7:0		LCMP2	LCMP1	LCMP0			HUNF	LUNF
0x0C ... 0x0D	Reserved									
0x0E	DBGCTRL	7:0								DBGRUN
0x0F ... 0x1F	Reserved									
0x20	LCNT	7:0	LCNT[7:0]							
0x21	HCNT	7:0	HCNT[7:0]							
0x22 ... 0x25	Reserved									
0x26	LPER	7:0	LPER[7:0]							
0x27	HPER	7:0	HPER[7:0]							
0x28	LCMP0	7:0	LCMP[7:0]							
0x29	HCMP0	7:0	HCMP[7:0]							
0x2A	LCMP1	7:0	LCMP[7:0]							
0x2B	HCMP1	7:0	HCMP[7:0]							
0x2C	LCMP2	7:0	LCMP[7:0]							
0x2D	HCMP2	7:0	HCMP[7:0]							

20.7 Register Description - Split Mode

20.7.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

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

Bits 3:1 – CLKSEL[2:0] Clock Select
These bits select the clock frequency for the timer/counter.

Value	Name	Description
0x0	DIV1	$f_{TCA} = f_{CLK\_PER}$
0x1	DIV2	$f_{TCA} = f_{CLK\_PER}/2$
0x2	DIV4	$f_{TCA} = f_{CLK\_PER}/4$
0x3	DIV8	$f_{TCA} = f_{CLK\_PER}/8$
0x4	DIV16	$f_{TCA} = f_{CLK\_PER}/16$
0x5	DIV64	$f_{TCA} = f_{CLK\_PER}/64$
0x6	DIV256	$f_{TCA} = f_{CLK\_PER}/256$
0x7	DIV1024	$f_{TCA} = f_{CLK\_PER}/1024$

Bit 0 – ENABLE Enable

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

20.7.2 Control B - Split Mode

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		HCMP2EN	HCMP1EN	HCMP0EN		LCMP2EN	LCMP1EN	LCMP0EN
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 6 – HCMP2EN High byte Compare 2 Enable
See HCMP0EN.

Bit 5 – HCMP1EN High byte Compare 1 Enable
See HCMP0EN.

Bit 4 – HCMP0EN High byte Compare 0 Enable
Setting the HCMPnEN bit in the FRQ or PWM Waveform Generation mode of operation will override the port output register for the corresponding WO[n+3] pin.

Bit 2 – LCMP2EN Low byte Compare 2 Enable
See LCMP0EN.

Bit 1 – LCMP1EN Low byte Compare 1 Enable
See LCMP0EN.

Bit 0 – LCMP0EN Low byte Compare 0 Enable
Setting the LCMPnEN bit in the FRQ or PWM Waveform Generation mode of operation will override the port output register for the corresponding WOn pin.

20.7.3 Control C - Split Mode

Name: CTRLC
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		HCMP2OV	HCMP1OV	HCMP0OV		LCMP2OV	LCMP1OV	LCMP0OV
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 6 – HCMP2OV High byte Compare 2 Output Value
See HCMP0OV.

Bit 5 – HCMP1OV High byte Compare 1 Output Value
See HCMP0OV.

Bit 4 – HCMP0OV High byte Compare 0 Output Value
The HCMPnOV bit allows direct access to the output compare value of the waveform generator when the timer/counter is not enabled. This is used to set or clear the WO[n+3] output value when the timer/counter is not running.

Bit 2 – LCMP2OV Low byte Compare 2 Output Value
See LCMP0OV.

Bit 1 – LCMP1OV Low byte Compare 1 Output Value
See LCMP0OV.

Bit 0 – LCMP0OV Low byte Compare 0 Output Value
The LCMPnOV bit allows direct access to the output compare value of the waveform generator when the timer/counter is not enabled. This is used to set or clear the WOn output value when the timer/counter is not running.

20.7.4 Control D

Name: CTRLD
Offset: 0x03
Reset: 0x00
Property: -

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

Bit 0 – SPLITM Enable Split Mode
This bit sets the timer/counter in Split mode operation. It will then work as two 8-bit timer/counters. The register map will change compared to normal 16-bit mode.

20.7.5 Control Register E Clear - Split Mode

Name: CTRLECLR
Offset: 0x04
Reset: 0x00
Property: -

This register can be used instead of a Read-Modify-Write (RMW) to clear individual bits by writing a '1' to its bit location.

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

Bits 3:2 – CMD[1:0] Command
These bits are used for software control of restart and reset of the timer/counter. The command bits are always read as '0'.

Value	Name	Description
0x0	NONE	No command
0x1	-	Reserved
0x2	RESTART	Force restart
0x3	RESET	Force hard Reset (ignored if the timer/counter is enabled)

Bits 1:0 – CMDEN[1:0] Command Enable
These bits configure what timer/counters the command given by the CMD-bits will be applied to.

Value	Name	Description
0x0	NONE	None
0x1	-	Reserved
0x2	-	Reserved
0x3	BOTH	Command (CMD) will be applied to both low byte and high byte timer/counter

20.7.6 Control Register E Set - Split Mode

Name: CTRLESET
Offset: 0x05
Reset: 0x00
Property: -

This register can be used instead of a Read-Modify-Write (RMW) to set individual bits by writing a '1' to its bit location.

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

Bits 3:2 – CMD[1:0] Command
This bit field used for software control of restart and reset of the timer/counter. The command bits are always read as '0'. The CMD bit field must be used together with the Command Enable (CMDEN) bits. Using the RESET command requires that both low byte and high byte timer/counter are selected with CMDEN.

Value	Name	Description
0x0	NONE	No command
0x1	-	Reserved
0x2	RESTART	Force restart
0x3	RESET	Force hard Reset (ignored if the timer/counter is enabled)

Bits 1:0 – CMDEN[1:0] Command Enable
These bits configure what timer/counters the command given by the CMD-bits will be applied to.

Value	Name	Description
0x0	NONE	None
0x1	-	Reserved
0x2	-	Reserved
0x3	BOTH	Command (CMD) will be applied to both low byte and high byte timer/counter

20.7.7 Interrupt Control Register - Split Mode

Name: INTCTRL
Offset: 0x0A
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		LCMP2	LCMP1	LCMP0			HUNF	LUNF
Access		R/W	R/W	R/W			R/W	R/W
Reset		0	0	0			0	0

Bit 6 – LCMP2 Low byte Compare Channel 2 Interrupt Enable
See LCMP0.

Bit 5 – LCMP1 Low byte Compare Channel 1 Interrupt Enable
See LCMP0.

Bit 4 – LCMP0 Low byte Compare Channel 0 Interrupt Enable
Writing the LCMPn bit to '1' enables the low byte Compare Channel n interrupt.

Bit 1 – HUNF High byte Underflow Interrupt Enable
Writing the HUNF bit to '1' enables the high byte underflow interrupt.

Bit 0 – LUNF Low byte Underflow Interrupt Enable
Writing the LUNF bit to '1' enables the low byte underflow interrupt.

20.7.8 Interrupt Flag Register - Split Mode

Name: INTFLAGS
Offset: 0x0B
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		LCMP2	LCMP1	LCMP0			HUNF	LUNF
Access		R/W	R/W	R/W			R/W	R/W
Reset		0	0	0			0	0

Bit 6 – LCMP2 Low byte Compare Channel 2 Interrupt Flag
See LCMP0 flag description.

Bit 5 – LCMP1 Low byte Compare Channel 1 Interrupt Flag
See LCMP0 flag description.

Bit 4 – LCMP0 Low byte Compare Channel 0 Interrupt Flag
The Low byte Compare Interrupt (LCMPn) flag is set on a compare match on the corresponding compare channel in the low byte timer.
For all modes of operation, the LCMPn flag will be set when a compare match occurs between the Low Byte Timer Counter (TCAn.LCNT) register and the corresponding Compare n (TCAn.LCMPn) register. The LCMPn flag will not be cleared automatically and has to be cleared by software. This is done by writing a '1' to its bit location.

Bit 1 – HUNF High byte Underflow Interrupt Flag
This flag is set on a high byte timer BOTTOM (underflow) condition. HUNF is not automatically cleared and needs to be cleared by software. This is done by writing a '1' to its bit location.

Bit 0 – LUNF Low byte Underflow Interrupt Flag
This flag is set on a low byte timer BOTTOM (underflow) condition. LUNF is not automatically cleared and needs to be cleared by software. This is done by writing a '1' to its bit location.

20.7.9 Debug Control Register

Name: DBGCTRL
Offset: 0x0E
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Run in Debug

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

20.7.10 Low Byte Timer Counter Register - Split Mode

Name: LCNT
Offset: 0x20
Reset: 0x00
Property: -

TCAn.LCNT contains the counter value for the low byte timer. CPU and UPDI write access has priority over count, clear or reload of the counter.

Bit	7	6	5	4	3	2	1	0
	LCNT[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 – LCNT[7:0] Counter Value for Low Byte Timer
These bits define the counter value of the low byte timer.

20.7.11 High Byte Timer Counter Register - Split Mode

Name: HCNT
Offset: 0x21
Reset: 0x00
Property: -

TCAn.HCNT contains the counter value for the high byte timer. CPU and UPDI write access has priority over count, clear or reload of the counter.

Bit	7	6	5	4	3	2	1	0
	HCNT[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 – HCNT[7:0] Counter Value for High Byte Timer
These bits define the counter value in high byte timer.

20.7.12 Low Byte Timer Period Register - Split Mode

Name: LPER
Offset: 0x26
Reset: 0xFF
Property: -

The TCAn.LPER register contains the TOP value for the low byte timer.

Bit	7	6	5	4	3	2	1	0
	LPER[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 – LPER[7:0] Period Value Low Byte Timer
These bits hold the TOP value for the low byte timer.

20.7.13 High Byte Period Register - Split Mode

Name: HPER
Offset: 0x27
Reset: 0xFF
Property: -

The TCAn.HPER register contains the TOP value for the high byte timer.

Bit	7	6	5	4	3	2	1	0
	HPER[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 – HPER[7:0] Period Value High Byte Timer
These bits hold the TOP value for the high byte timer.

20.7.14 Compare Register n For Low Byte Timer - Split Mode

Name: LCMPn
Offset: 0x28 + n*0x02 [n=0..2]
Reset: 0x00
Property: -

The TCAn.LCMPn register represents the compare value of Compare Channel n for the low byte timer. This register is continuously compared to the counter value of the low byte timer, TCAn.LCNT. Normally, the outputs from the comparators are then used to generate waveforms.

Bit	7	6	5	4	3	2	1	0
	LCMP[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 – LCMP[7:0] Compare Value of Channel n
These bits hold the compare value of channel n that is compared to TCAn.LCNT.

20.7.15 High Byte Compare Register n - Split Mode

Name: HCMPn
Offset: 0x29 + n*0x02 [n=0..2]
Reset: 0x00
Property: -

The TCAn.HCMPn register represents the compare value of Compare Channel n for the high byte timer. This register is continuously compared to the counter value of the high byte timer, TCAn.HCNT. Normally, the outputs from the comparators are then used to generate waveforms.

Bit	7	6	5	4	3	2	1	0
	HCMP[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 – HCMP[7:0] Compare Value of Channel n
These bits hold the compare value of channel n that is compared to TCAn.HCNT.

21. TCB - 16-bit Timer/Counter Type B

21.1 Features

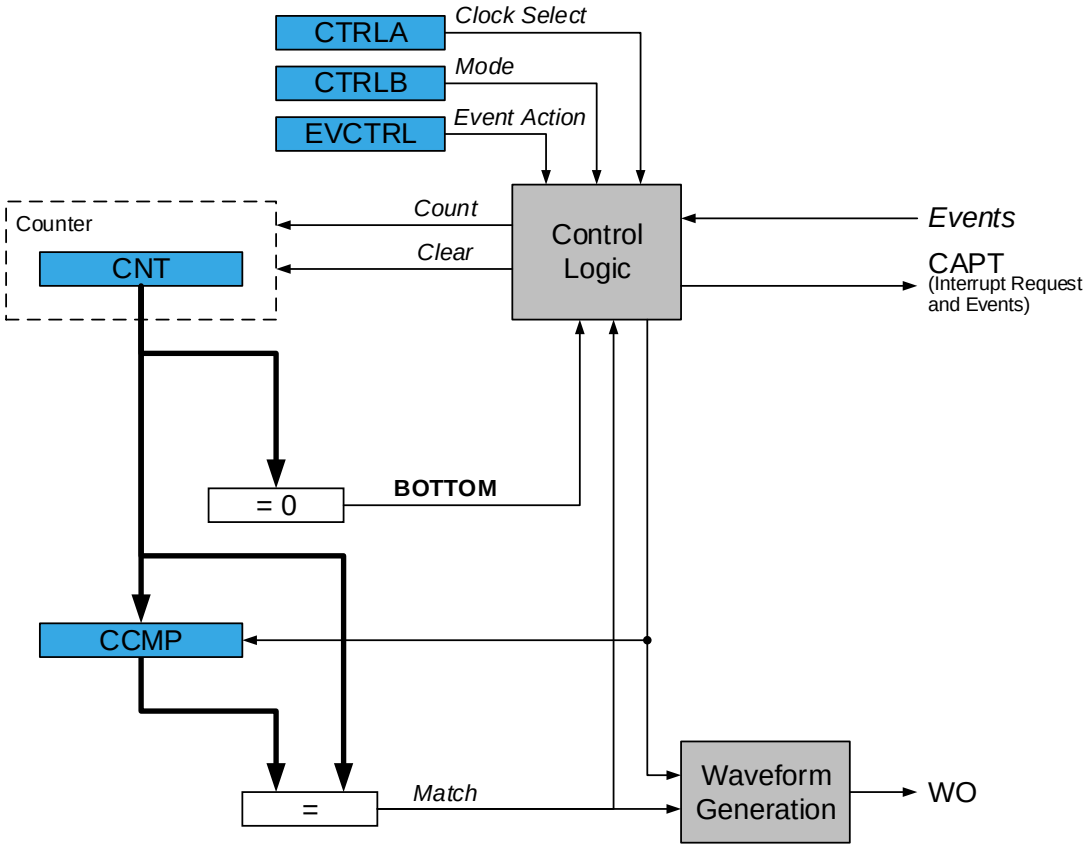
- 16-bit Counter Operation Modes:
  - Periodic interrupt
  - Time-out check
  - Input capture
    - On event
    - Frequency measurement
    - Pulse-width measurement
    - Frequency and pulse-width measurement
  - Single-shot
  - 8-bit Pulse-Width Modulation (PWM)
- Noise Canceler on Event Input
- Synchronize Operation with TCAn

21.2 Overview

The capabilities of the 16-bit Timer/Counter type B (TCB) include frequency and waveform generation, and input capture on event with time and frequency measurement of digital signals. The TCB consists of a base counter and control logic that can be set in one of eight different modes, each mode providing unique functionality. The base counter is clocked by the peripheral clock with optional prescaling.

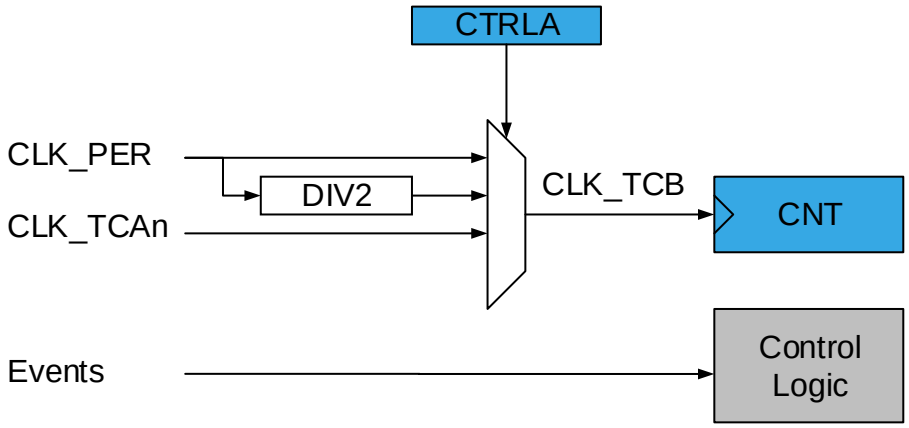
21.2.1 Block Diagram

Figure 21-1. Timer/Counter Type B Block



The timer/counter can be clocked from the Peripheral Clock (CLK_PER), or a 16-bit Timer/Counter type A (CLK_TCAn).

Figure 21-2. Timer/Counter Clock Logic



The Clock Select (CLKSEL) bit field in the Control A (TCBn.CTRLA) register selects one of the prescaler outputs directly as the clock (CLK_TCB) input.

Setting the timer/counter to use the clock from a TCAn allows the timer/counter to run in sync with that TCAn.

By using the EVSYS, any event source, such as an external clock signal on any I/O pin, may be used as a control logic input. When an event action controlled operation is used, the clock selection must be set to use an event channel as the counter input.

21.2.2 Signal Description

Signal	Description	Type
WO	Digital Asynchronous Output	Waveform Output

21.3 Functional Description

21.3.1 Definitions

The following definitions are used throughout the documentation:

Table 21-1. Timer/Counter Definitions

Name	Description
BOTTOM	The counter reaches BOTTOM when it becomes 0x0000
MAX	The counter reaches maximum when it becomes 0xFFFF
TOP	The counter reaches TOP when it becomes equal to the highest value in the count sequence
CNT	Counter register value
CCMP	Capture/Compare register value

Note: In general, the term ‘timer’ is used when the timer/counter is counting periodic clock ticks. The term ‘counter’ is used when the input signal has sporadic or irregular ticks.

21.3.2 Initialization

By default, the TCB is in Periodic Interrupt mode. Follow these steps to start using it:

1. Write a TOP value to the Compare/Capture (TCBn.CCMP) register.
2. Optional: Write the Compare/Capture Output Enable (CCMPEN) bit in the Control B (TCBn.CTRLB) register to‘1’. This will make the waveform output available on the corresponding pin, overriding the value in the corresponding PORT output register.
3. Enable the counter by writing a ‘1’ to the ENABLE bit in the Control A (TCBn.CTRLA) register.
   The counter will start counting clock ticks according to the prescaler setting in the Clock Select (CLKSEL) bit field in the Control A (TCBn.CTRLA) register.
4. The counter value can be read from the Count (TCBn.CNT) register. The peripheral will generate a CAPT interrupt and event when the CNT value reaches TOP.
   4.1. If the Compare/Capture register is modified to a value lower than the current Count register, the peripheral will count to MAX and wrap around.

21.3.3 Operation

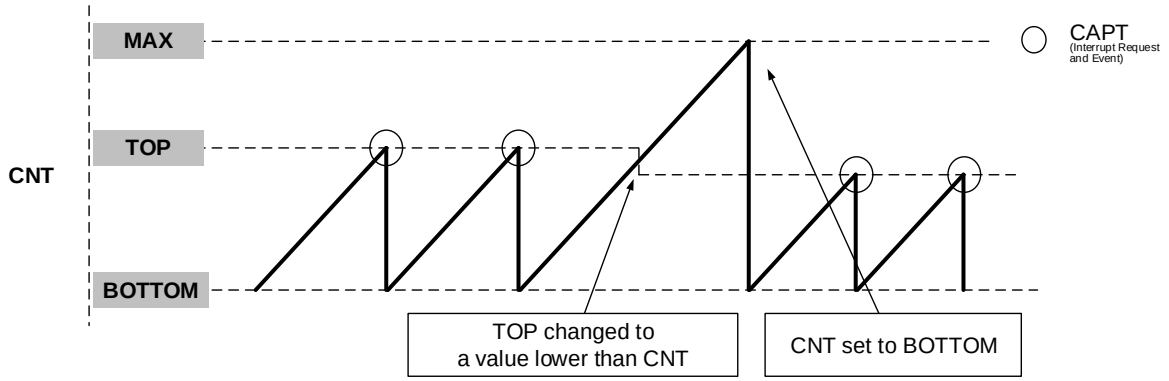
21.3.3.1 Modes

The timer can be configured to run in one of the eight different modes described in the sections below. The event pulse needs to be longer than one system clock cycle in order to ensure edge detection.

21.3.3.1.1 Periodic Interrupt Mode

In the Periodic Interrupt mode, the counter counts to the capture value and restarts from BOTTOM. A CAPT interrupt and event is generated when the counter is equal to TOP. If TOP is updated to a value lower than count upon reaching MAX the counter restarts from BOTTOM.

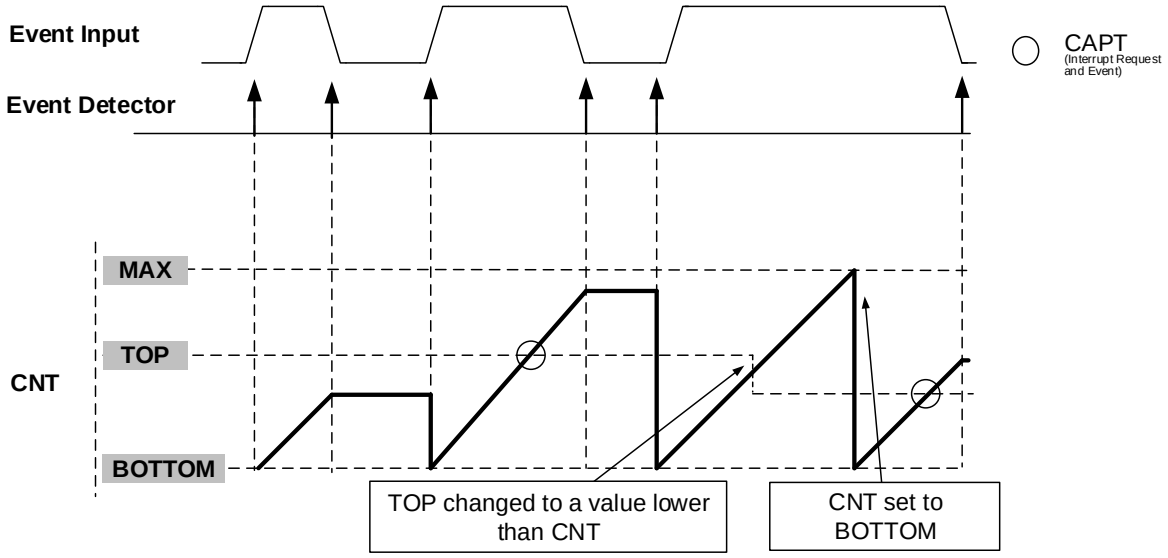
Figure 21-3. Periodic Interrupt Mode



21.3.3.1.2 Time-Out Check Mode

In the Time-Out Check mode, the peripheral starts counting on the first signal edge and stops on the next signal edge detected on the event input channel. Start or Stop edge is determined by the Event Edge (EDGE) bit in the Event Control (TCBn.EVCTRL) register. If the Count (TCBn.CNT) register reaches TOP before the second edge, a CAPT interrupt and event will be generated. In Freeze state, after a Stop edge is detected, the counter will restart on a new Start edge. If TOP is updated to a value lower than the Count (TCBn.CNT) register upon reaching MAX the counter restarts from BOTTOM. Reading the Count (TCBn.CNT) register or Compare/Capture (TCBn.CCMP) register, or writing the Run (RUN) bit in the Status (TCBn.STATUS) register in Freeze state will have no effect.

Figure 21-4. Time-Out Check Mode

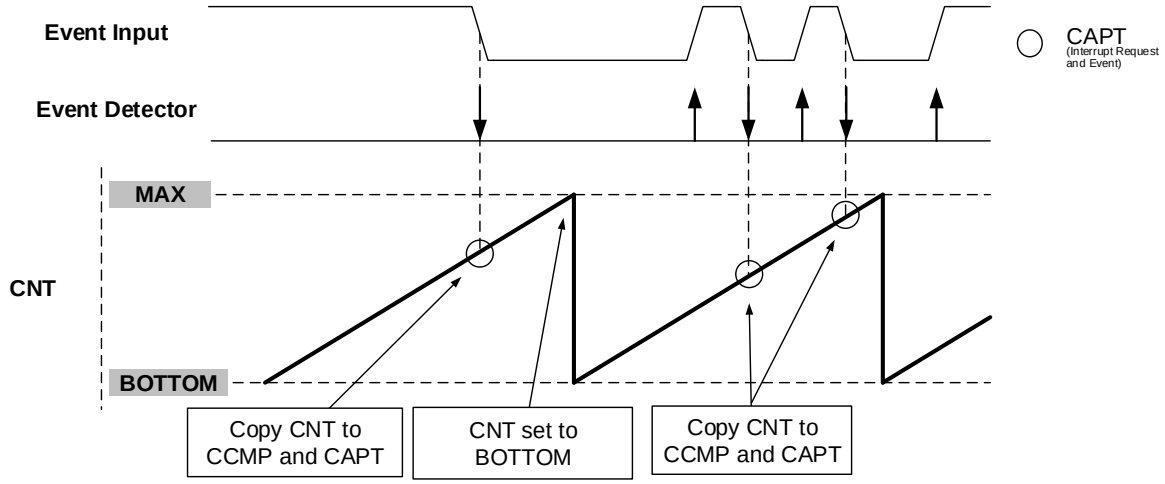


21.3.3.1.3 Input Capture on Event Mode

In the Input Capture on Event mode, the counter will count from BOTTOM to MAX continuously. When an event is detected the Count (TCBn.CNT) register value is transferred to the Compare/Capture (TCBn.CCMP) register and a CAPT interrupt and event is generated. The Event edge detector that can be configured to trigger a capture on either rising or falling edges.

The figure below shows the input capture unit configured to capture on the falling edge of the event input signal. The CAPT Interrupt flag is automatically cleared after the low byte of the Compare/Capture (TCBn.CCMP) register has been read.

Figure 21-5. Input Capture on Event



It is recommended to write zero to the TCBn.CNT register when entering this mode from any other mode.

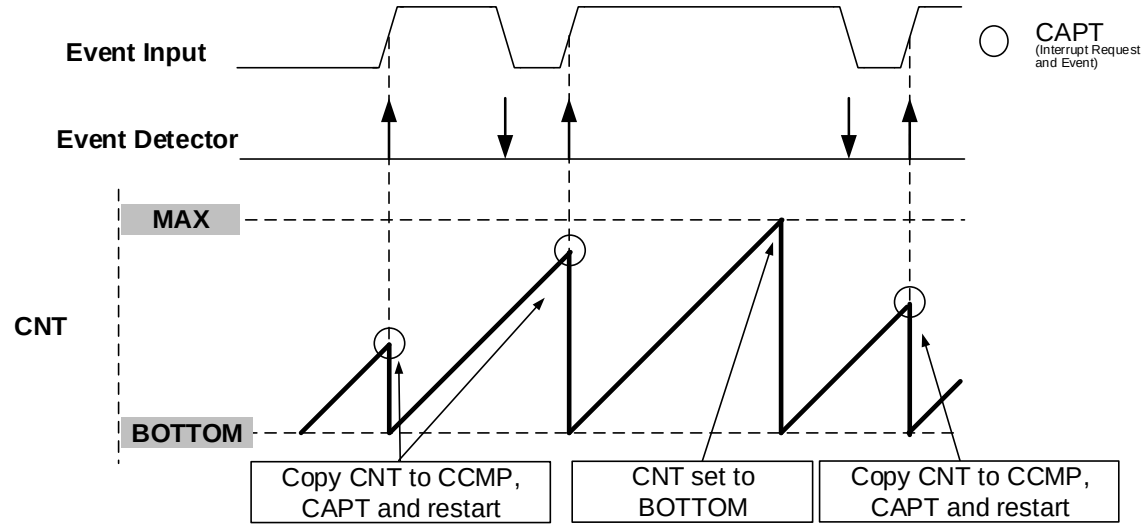
21.3.3.1.4 Input Capture Frequency Measurement Mode

In the Input Capture Frequency Measurement mode, the TCB captures the counter value and restarts on either a positive or negative edge of the event input signal.

The CAPT Interrupt flag is automatically cleared after the low byte of the Compare/Capture (TCBn.CCMP) register has been read.

The figure below illustrates this mode when configured to act on rising edge.

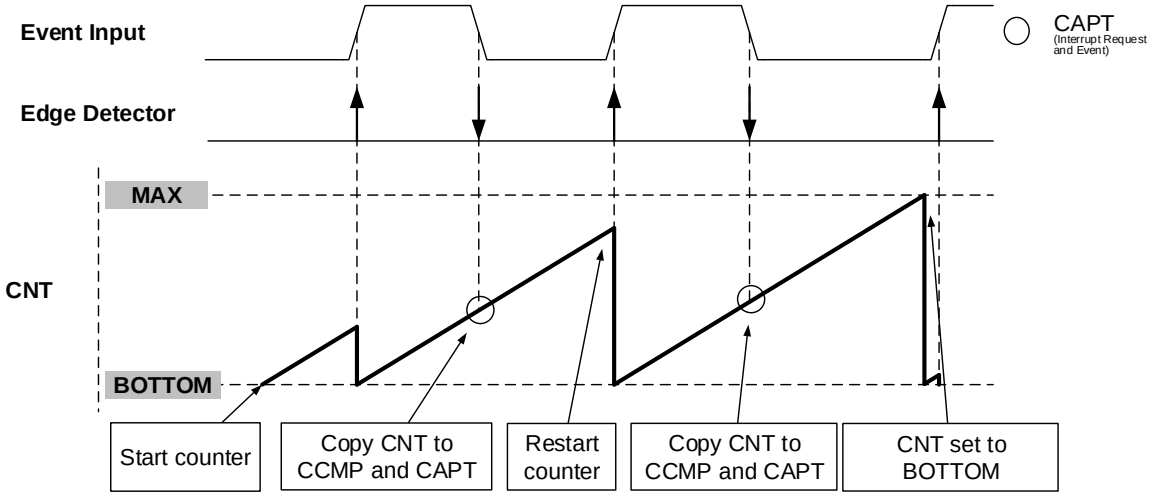
Figure 21-6. Input Capture Frequency Measurement



21.3.3.1.5 Input Capture Pulse-Width Measurement Mode

In the Input Capture Pulse-Width Measurement mode, the input capture pulse-width measurement will restart the counter on a positive edge, and capture on the next falling edge before an interrupt request is generated. The CAPT Interrupt flag is automatically cleared after the low byte of the Compare/Capture (TCBn.CCMP) register has been read. The timer will automatically switch between rising and falling edge detection, but a minimum edge separation of two clock cycles is required for correct behavior.

Figure 21-7. Input Capture Pulse-Width Measurement

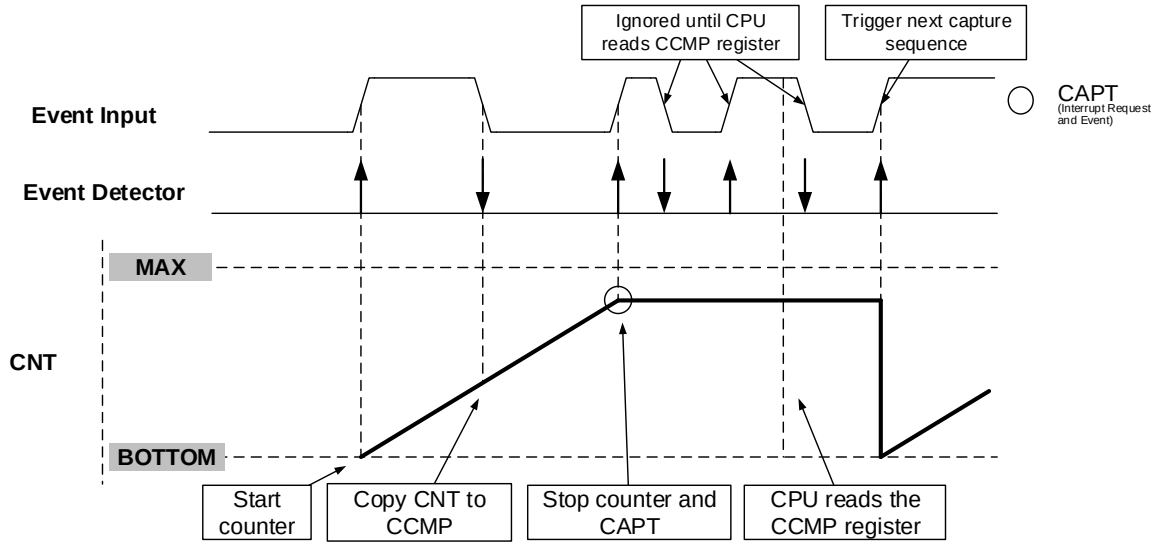


21.3.3.1.6 Input Capture Frequency and Pulse-Width Measurement Mode

In the Input Capture Frequency and Pulse-Width Measurement mode, the timer will start counting when a positive edge is detected on the event input signal. The count value is captured on the following falling edge. The counter stops when the second rising edge of the event input signal is detected. This will set the interrupt flag.

The CAPT Interrupt flag is automatically cleared after the low byte of the Compare/Capture (TCBn.CCMP) register has been read, and the timer/counter is ready for a new capture sequence. Therefore, the Count (TCBn.CNT) register must be read before the Compare/Capture (TCBn.CCMP) register, since it is reset to BOTTOM at the next positive edge of the event input signal.

Figure 21-8. Input Capture Frequency and Pulse-Width Measurement



21.3.3.1.7 Single-Shot Mode

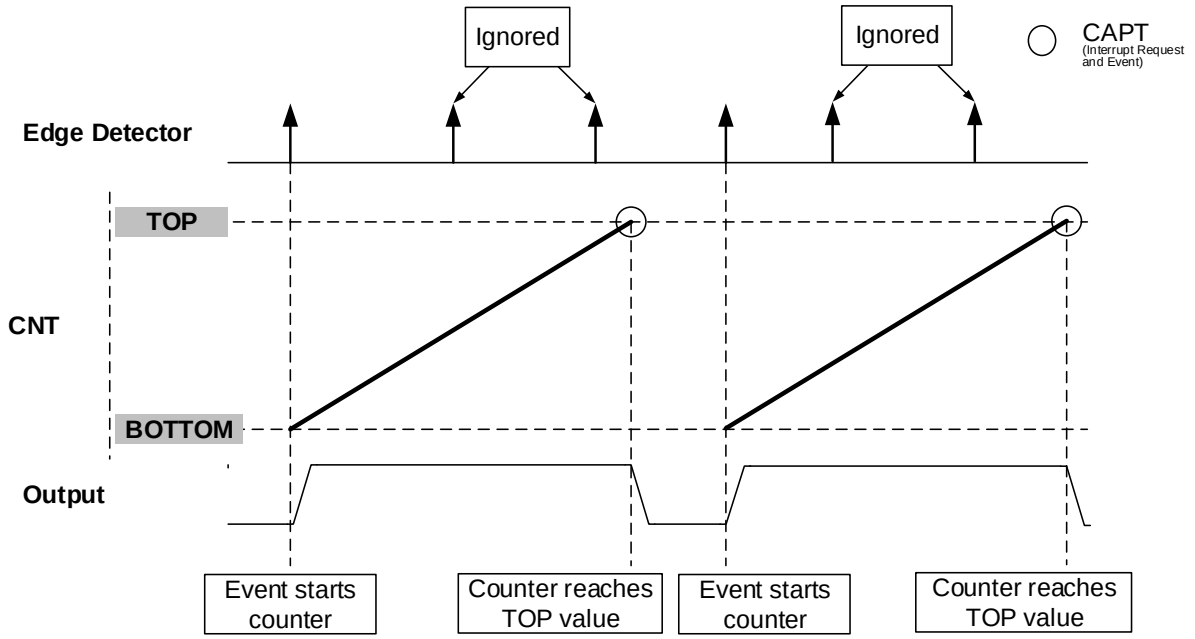
The Single-Shot mode can be used to generate a pulse with a duration defined by the Compare (TCBn.CCMP) register, every time a rising or falling edge is observed on a connected event channel.

When the counter is stopped, the output pin is driven to low. If an event is detected on the connected event channel, the timer will reset and start counting from BOTTOM to TOP while driving its output high. The RUN bit in the Status (TCBn.STATUS) register can be read to see if the counter is counting or not. When the Counter register reaches the CCMP register value, the counter will stop, and the output pin will go low for at least one prescaler cycle. A new event arriving during this time will be ignored. There is a two clock-cycle delay from when the event is received until the output is set high.

The counter will start counting as soon as the module is enabled, even without triggering an event. This is prevented by writing TOP to the Counter register. Similar behavior is seen if the Event Edge (EDGE) bit in the Event Control (TCBn.EVCTRL) register is '1' while the module is enabled. Writing TOP to the Counter register prevents this as well.

If the Event Asynchronous (ASYNC) bit in the Control B (TCBn.CTRLB) register is written to '1' the timer will react asynchronously to an incoming event. An edge on the event will immediately cause the output signal to be set. The counter will still start counting two clock cycles after the event is received.

Figure 21-9. Single-Shot Mode

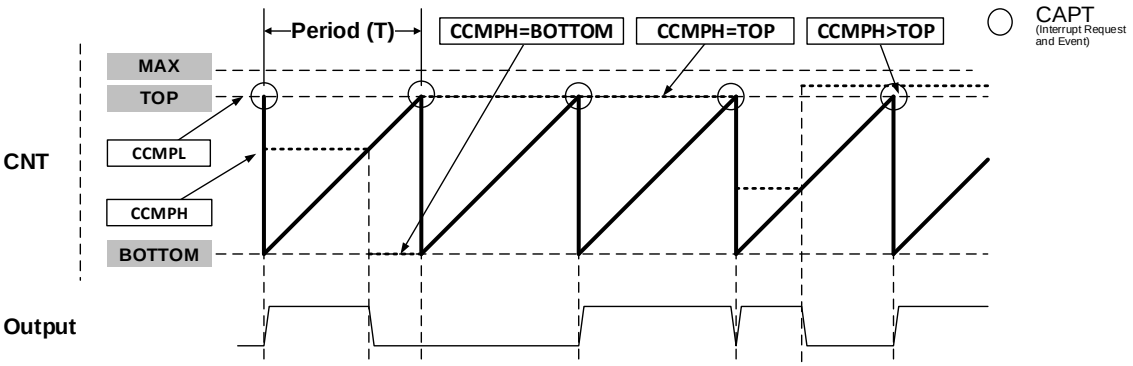


21.3.3.1.8 8-Bit PWM Mode

The TCB can be configured to run in 8-bit PWM mode, where each of the register pairs in the 16-bit Compare/ Capture (TCBn.CCMPH and TCBn.CCMPL) register are used as individual Compare registers. The period (T) is controlled by CCMPL, while CCMPH controls the duty cycle of the waveform. The counter will continuously count from BOTTOM to CCMPL, and the output will be set at BOTTOM and cleared when the counter reaches CCMPH.

CCMPH is the number of cycles for which the output will be driven high. CCMPL+1 is the period of the output pulse.

Figure 21-10. 8-Bit PWM Mode



21.3.3.2 Output

Timer synchronization and output logic level are dependent on the selected Timer Mode (CNTMODE) bit field in Control B (TCBn.CTRLB) register. In Single-Shot mode the timer/counter can be configured so that the signal generation happens asynchronously to an incoming event (ASYNC = 1 in TCBn.CTRLB). The output signal is then set immediately at the incoming event instead of being synchronized to the TCB clock. Even though the output is set immediately, it will take two to three CLK_TCB cycles before the counter starts counting.

Writing the Compare/Capture Output Enable (CCMPEN) bit in TCBn.CTRLB to '1' enables the waveform output. This will make the waveform output available on the corresponding pin, overriding the value in the corresponding PORT output register.

The different configurations and their impact on the output are listed in the table below.

Table 21-2. Output Configuration

CCMPEN	CNTMODE	ASYNC	Output
1	Single-Shot mode	0	The output is high when the counter starts and the output is low when the counter stops
		1	The output is high when the event arrives and the output is low when the counter stops
	8-bit PWM mode	Not applicable	8-bit PWM mode
	Other modes	Not applicable	The output initial level sets the CCMPINIT bit in the TCBn.CTRLB register
0	Not applicable	Not applicable	No output

It is not recommended to change modes while the peripheral is enabled as this can produce an unpredictable output. There is a possibility that an interrupt flag is set during the timer configuration. It is recommended to clear the Timer/Counter Interrupt Flags (TCBn.INTFLAGS) register after configuring the peripheral.

21.3.3.3 Noise Canceler

The Noise Canceler improves the noise immunity by using a simple digital filter scheme. When the Noise Filter (FILTER) bit in the Event Control (TCBn.EVCTRL) register is enabled, the peripheral monitors the event channel and keeps a record of the last four observed samples. If four consecutive samples are equal, the input is considered to be stable and the signal is fed to the edge detector.

When enabled the Noise Canceler introduces an additional delay of four system clock cycles between a change applied to the input and the update of the Input Compare register.

The Noise Canceler uses the system clock and is, therefore, not affected by the prescaler.

21.3.3.4 Synchronized with Timer/Counter Type A

The TCB can be configured to use the clock (CLK_TCA) of a Timer/Counter type A (TCAn) by writing to the Clock Select bit field (CLKSEL) in the Control A register (TCBn.CTRLA). In this setting, the TCB will count on the exact same clock source as selected in TCAn.

When the Synchronize Update (SYNCUPD) bit in the Control A (TCBn.CTRLA) register is written to '1', the TCB counter will restart when the TCAn counter restarts.

21.3.4 Events

The TCB can generate the events described in the following table:

Table 21-3. Event Generators in TCB

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCBn	CAPT	CAPT flag set	Pulse	CLK_PER	One CLK_PER period

The conditions for generating the CAPT event is identical to those that will raise the corresponding interrupt flag in the Timer/Counter Interrupt Flags (TCBn.INTFLAGS) register. Refer to the *Event System* section for more details regarding event users and Event System configuration.

The TCB can receive the events described in the following table:

Table 21-4. Event Users and Available Event Actions in TCB

User Name		Description	Input Detection	Async/Sync
Peripheral	Input			
TCBn	CAPT	Time-Out Check Count mode	Edge	Sync
		Input Capture on Event Count mode		
		Input Capture Frequency Measurement Count mode		
		Input Capture Pulse-Width Measurement Count mode		
		Input Capture Frequency and Pulse-Width Measurement Count mode		
		Single-Shot Count mode		Both

If the Capture Event Input Enable (CAPTEI) bit in the Event Control (TCBn.EVCTRL) register is written to '1', incoming events will result in an event action as defined by the Event Edge (EDGE) bit in Event Control (TCBn.EVCTRL) register and the Timer Mode (CNTMODE) bit field in Control B (TCBn.CTRLB) register. The event needs to last for at least one CLK_PER cycle to be recognized.

If the Asynchronous mode is enabled for Single-Shot mode, the event is edge-triggered and will capture changes on the event input shorter than one system clock cycle.

21.3.5 Interrupts

Table 21-5. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
CAPT	TCB interrupt	Depending on the operating mode. See the description of the CAPT bit in the TCBn.INTFLAG register.

When an interrupt condition occurs, the corresponding interrupt flag is set in the peripheral's Interrupt Flags (*peripheral*.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

21.3.6 Sleep Mode Operation

TCBn is by default disabled in Standby sleep mode. It will be halted as soon as the sleep mode is entered.

The module can stay fully operational in the Standby sleep mode if the Run Standby (RUNSTDBY) bit in the TCBn.CTRLA register is written to '1'.

All operations are halted in Power-Down sleep mode.

21.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		RUNSTDBY		SYNCUPD		CLKSEL[1:0]		ENABLE
0x01	CTRLB	7:0		ASYNC	CCMPINIT	CCMPEN		CNTMODE[2:0]		
0x02 ... 0x03	Reserved									
0x04	EVCTRL	7:0		FILTER		EDGE				CAPTEI
0x05	INTCTRL	7:0								CAPT
0x06	INTFLAGS	7:0								CAPT
0x07	STATUS	7:0								RUN
0x08	DBGCTRL	7:0								DBGRUN
0x09	TEMP	7:0	TEMP[7:0]							
0x0A	CNT	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
0x0C	CCMP	7:0	CCMP[7:0]							
		15:8	CCMP[15:8]							

21.5 Register Description

21.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		RUNSTDBY		SYNCUPD		CLKSEL[1:0]		ENABLE
Access		R/W		R/W		R/W	R/W	R/W
Reset		0		0		0	0	0

Bit 6 – RUNSTDBY Run in Standby
Writing a '1' to this bit will enable the peripheral to run in Standby Sleep mode. Not applicable when CLKSEL is set to 0x2 (CLK_TCA).

Bit 4 – SYNCUPD Synchronize Update
When this bit is written to '1', the TCB will restart whenever TCA0 is restarted or overflows. This can be used to synchronize capture with the PWM period.

Bits 2:1 – CLKSEL[1:0] Clock Select
Writing these bits selects the clock source for this peripheral.

Value	Name	Description
0x0	CLKDIV1	CLK_PER
0x1	CLKDIV2	CLK_PER/DIV2
0x3	CLKTCA	Use TCA_CLK from TCA0
0x4	-	Reserved

Bit 0 – ENABLE Enable
Writing this bit to '1' enables the Timer/Counter type B peripheral.

21.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		ASYNC	CCMPINIT	CCMPEN		CNTMODE[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 6 – ASYNC Asynchronous Enable
Writing this bit to '1' will allow asynchronous updates of the TCB output signal in Single-Shot mode.

Value	Description
0	The output will go HIGH when the counter starts after synchronization
1	The output will go HIGH when an event arrives

Bit 5 – CCMPINIT Compare/Capture Pin Initial Value
This bit is used to set the initial output value of the pin when a pin output is used. This bit has no effect in 8-bit PWM mode and Single-Shot mode.

Value	Description
0	Initial pin state is LOW
1	Initial pin state is HIGH

Bit 4 – CCMPEN Compare/Capture Output Enable
Writing this bit to '1' enables the waveform output. This will make the waveform output available on the corresponding pin, overriding the value in the corresponding PORT output register. The corresponding pin direction must be configured as an output in the PORT peripheral.

Value	Description
0	Waveform output is not enabled on the corresponding pin
1	Waveform output will override the output value of the corresponding pin

Bits 2:0 – CNTMODE[2:0] Timer Mode
Writing these bits selects the Timer mode.

Value	Name	Description
0x0	INT	Periodic Interrupt mode
0x1	TIMEOUT	Time-out Check mode
0x2	CAPT	Input Capture on Event mode
0x3	FRQ	Input Capture Frequency Measurement mode
0x4	PW	Input Capture Pulse-Width Measurement mode
0x5	FRQPW	Input Capture Frequency and Pulse-Width Measurement mode
0x6	SINGLE	Single-Shot mode
0x7	PWM8	8-Bit PWM mode

21.5.3 Event Control

Name: EVCTRL
Offset: 0x04
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		FILTER		EDGE				CAPTEI
Access		R/W		R/W				R/W
Reset		0		0				0

Bit 6 – FILTER Input Capture Noise Cancellation Filter
Writing this bit to '1' enables the Input Capture Noise Cancellation unit.

Bit 4 – EDGE Event Edge
This bit is used to select the event edge. The effect of this bit is dependent on the selected Count Mode (CNTMODE) bit field in TCBn.CTRLB. "—" means that an event or edge has no effect in this mode.

Count Mode	EDGE	Positive Edge	Negative Edge
Periodic Interrupt mode	0	—	—
	1	—	—
Timeout Check mode	0	Start counter	Stop counter
	1	Stop counter	Start counter
Input Capture on Event mode	0	Input Capture, interrupt	—
	1	—	Input Capture, interrupt
Input Capture Frequency Measurement mode	0	Input Capture, clear and restart counter, interrupt	—
	1	—	Input Capture, clear and restart counter, interrupt
Input Capture Pulse-Width Measurement mode	0	Clear and restart counter	Input Capture, interrupt
	1	Input Capture, interrupt	Clear and restart counter
Input Capture Frequency and Pulse-Width Measurement mode	0	• On the 1st Positive: Clear and restart counter • On the following Negative: Input Capture • On the 2nd Positive: Stop counter, interrupt	
	1	• On the 1st Negative: Clear and restart counter • On the following Positive: Input Capture • On the 2nd Negative: Stop counter, interrupt	
Single-Shot mode	0	Start counter	—
	1	—	Start counter
8-Bit PWM mode	0	—	—
	1	—	—

Bit 0 – CAPTEI Capture Event Input Enable
Writing this bit to '1' enables the input capture event.

21.5.4 Interrupt Control

Name: INTCTRL
Offset: 0x05
Reset: 0x00
Property: -

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

Bit 0 – CAPT Capture Interrupt Enable
Writing this bit to '1' enables interrupt on capture.

21.5.5 Interrupt Flags

Name: INTFLAGS
Offset: 0x06
Reset: 0x00
Property: -

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

Bit 0 – CAPT Capture Interrupt Flag
This bit is set when a capture interrupt occurs. The interrupt conditions are dependent on the Counter Mode (CNTMODE) bit field in the Control B (TCBn.CTRLB) register.
This bit is cleared by writing a '1' to it or when the Capture register is read in Capture mode.

Table 21-6. Interrupt Sources Set Conditions by Counter Mode

Counter Mode	Interrupt Set Condition	TOP Value	CAPT
Periodic Interrupt mode	Set when the counter reaches TOP	CCMP	CNT == TOP
Timeout Check mode	Set when the counter reaches TOP		
Single-Shot mode	Set when the counter reaches TOP		
Input Capture Frequency Measurement mode	Set on edge when the Capture register is loaded and the counter restarts; the flag clears when the capture is read	--	On Event, copy CNT to CCMP, and restart counting (CNT == BOTTOM)
Input Capture on Event mode	Set when an event occurs and the Capture register is loaded; the flag clears when the capture is read		On Event, copy CNT to CCMP, and continue counting
Input Capture Pulse-Width Measurement mode	Set on edge when the Capture register is loaded; the previous edge initialized the count; the flag clears when the capture is read		
Input Capture Frequency and Pulse-Width Measurement mode	Set on the second edge (positive or negative) when the counter is stopped; the flag clears when the capture is read		
8-Bit PWM mode	Set when the counter reaches CCML	CCML	CNT == CCML

21.5.6 Status

Name: STATUS
Offset: 0x07
Reset: 0x00
Property: -

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

Bit 0 – RUN Run
When the counter is running, this bit is set to '1'. When the counter is stopped, this bit is cleared to '0'.
The bit is read-only and cannot be set by UPDI.

21.5.7 Debug Control

Name: DBGCTRL
Offset: 0x08
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

21.5.8 Temporary Value

Name: TEMP
Offset: 0x09
Reset: 0x00
Property: -

The Temporary register is used by the CPU for 16-bit single-cycle access to the 16-bit registers of this peripheral. The register is common for all the 16-bit registers of this peripheral and can be read and written by software. For more details on reading and writing 16-bit registers, refer to *Accessing 16-Bit Registers* in the *AVR CPU* section.

Bit	7	6	5	4	3	2	1	0
	TEMP[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 – TEMP[7:0] Temporary Value

21.5.9 Count

Name: CNT
Offset: 0x0A
Reset: 0x00
Property: -

The TCBn.CNTL and TCBn.CNTH register pair represents the 16-bit value TCBn.CNT. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

CPU and UPDI write access has priority over internal updates of the register.

Bit	15	14	13	12	11	10	9	8
	CNT[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
	CNT[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 – CNT[15:8] Count Value High
These bits hold the MSB of the 16-bit Counter register.

Bits 7:0 – CNT[7:0] Count Value Low
These bits hold the LSB of the 16-bit Counter register.

21.5.10 Capture/Compare

Name: CCMP
Offset: 0x0C
Reset: 0x00
Property: -

The TCBn.CCMPL and TCBn.CCMPH register pair represents the 16-bit value TCBn.CCMP. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

This register has different functions depending on the mode of operation:

- For Capture operation, these registers contain the captured value of the counter at the time the capture occurs
- In Periodic Interrupt/Time-Out and Single-Shot mode, this register acts as the TOP value
- In 8-bit PWM mode, TCBn.CCMPL and TCBn.CCMPH act as two independent registers: The period of the waveform is controlled by CCMPL, while CCMPH controls the duty cycle.

Bit	15	14	13	12	11	10	9	8
	CCMP[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
	CCMP[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 – CCMP[15:8] Capture/Compare Value High Byte
These bits hold the MSB of the 16-bit compare, capture, and top value.

Bits 7:0 – CCMP[7:0] Capture/Compare Value Low Byte
These bits hold the LSB of the 16-bit compare, capture, and top value.

22. TCD - 12-Bit Timer/Counter Type D

22.1 Features

- 12-bit Timer/Counter
- Programmable Prescaler
- Double-Buffered Compare Registers
- Waveform Generation:
  - One Ramp mode
  - Two Ramp mode
  - Four Ramp mode
  - Dual Slope mode
- Two Separate Input Channels
- Software and Input Based Capture
- Programmable Filter for Input Events
- Conditional Waveform Generation on External Events:
  - Fault handling
  - Input blanking
  - Overload protection
  - Fast emergency stop by hardware
- Half-Bridge and Full-Bridge Output Support

22.2 Overview

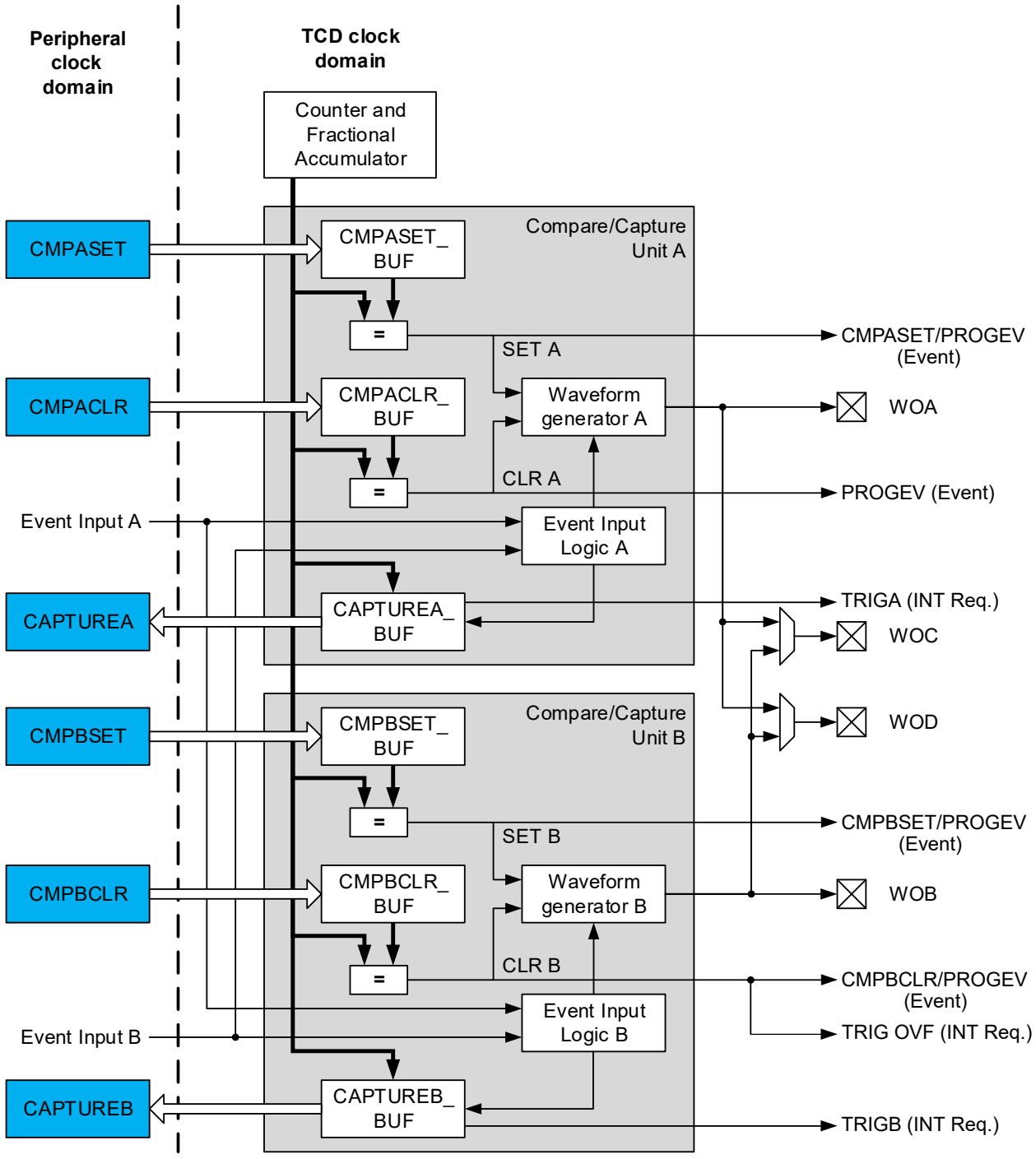
The Timer/Counter type D (TCD) is a high-performance waveform generator that consists of an asynchronous counter, a prescaler, and compare, capture and control logic.

The TCD contains a counter that can run on a clock which is asynchronous to the peripheral clock. It contains compare logic that generates two independent outputs with optional dead time. It is connected to the Event System for capture and deterministic Fault control. The timer/counter can generate interrupts and events on compare match and overflow.

This device provides one instance of the TCD peripheral, TCD0.

22.2.1 Block Diagram

Figure 22-1. Timer/Counter Block Diagram



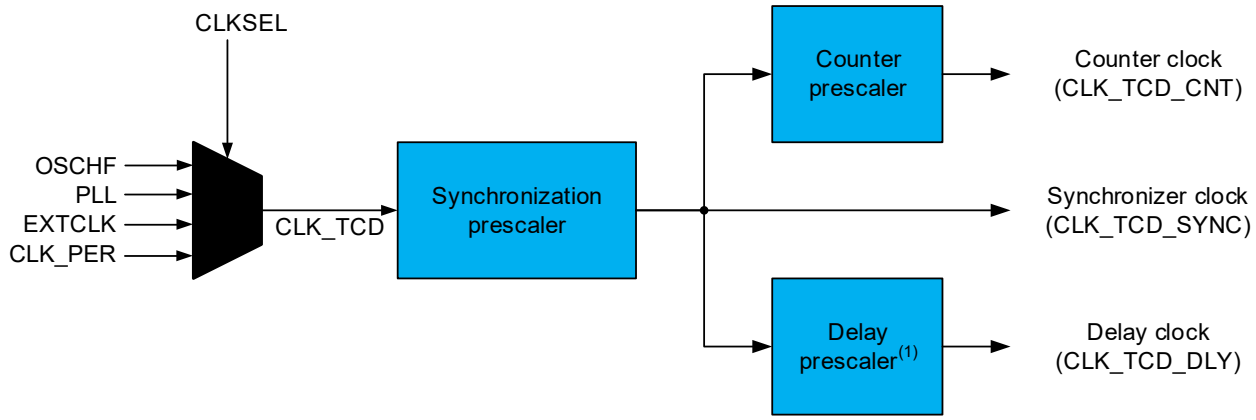
The TCD core is asynchronous to the peripheral clock. The timer/counter consists of two compare/capture units, each with a separate waveform output. There are also two extra waveform outputs which can be equal to the output from one of the units. For each compare/capture unit, there is a pair of compare registers which are stored in the respective peripheral registers (TCDn.CMPASET, TCDn.CMPACLR, TCDn.CMPBSET, TCDn.CMPBCLR).

During normal operation, the counter value is continuously compared to the compare registers. This is used to generate both interrupts and events.

The TCD can use the input events in ten different input modes, selected separately for the two input events. The input mode defines how the input events will affect the outputs, and where in the TCD cycle the counter must go when an event occurs.

The TCD can select between four different clock sources that can be prescaled. There are three different prescalers with separate controls, as shown below.

Figure 22-2. Clock Selection and Prescalers Overview



1. Used by input blanking/delay event out.

The TCD synchronizer clock is separate from the other module clocks, enabling faster synchronization between the TCD domain and the I/O domain.

The total prescaling for the counter is:

$$\text{SYNCPRESC\_division\_factor} \times \text{CNTPRESC\_division\_factor}$$

The delay prescaler is used to prescale the clock used for the input blanking/delayed event output functionality. The prescaler can be configured independently allowing separate range and accuracy settings from the counter functionality. The synchronization prescaler and counter prescaler can be configured from the Control A (TCDn.CTRLA) register, while the delay prescaler can be configured from the Delay Control (TCDn.DLYCTRL) register.

22.2.2 Signal Description

Signal	Description	Type
WOA	TCD waveform output A	Digital output
WOB	TCD waveform output B	Digital output
WOC	TCD waveform output C	Digital output
WOD	TCD waveform output D	Digital output

22.3 Functional Description

22.3.1 Definitions

The following definitions are used throughout the documentation:

Table 22-1. Timer/Counter Definitions

Name	Description
TCD cycle	The sequence of four states that the counter needs to go through before it has returned to the same position.

.....continued	
Name	**Description**
Input blanking	The functionality to ignore an event input for a programmable time in a selectable part of the TCD cycle.
Asynchronous output control	Allows the event to override the output instantly when an event occurs. It is used for handling non-recoverable Faults.
One ramp	The counter is reset to zero once during a TCD cycle.
Two ramp	The counter is reset to zero two times during a TCD cycle.
Four ramp	The counter is reset to zero four times during a TCD cycle.
Dual ramp	The counter counts both up and down between zero and a selected top value during a TCD cycle.
Input mode	A predefined setting that changes the output characteristics, based on the given input events.

22.3.2 Initialization

To initialize the TCD:

1. Select the clock source and the prescaler from the Control A (TCDn.CTRLA) register.
2. Select the Waveform Generation Mode from the Control B (TCDn.CTRLB) register.
3. Optional: Configure the other static registers to the desired functionality.
4. Write the initial values in the Compare (TCDn.CMPxSET/CLR) registers.
5. Optional: Write the desired values to the other double-buffered registers.
6. Ensure that the Enable Ready (ENRDY) bit in the Status (TCDn.STATUS) register is set to '1'.
7. Enable the TCD by writing a '1' to the ENABLE bit in the Control A (TCDn.CTRLA) register.

22.3.3 Operation

22.3.3.1 Register Synchronization Categories

Most of the I/O registers need to be synchronized to the TCD core clock domain. This is done differently for different register categories.

Table 22-2. Categorization of Registers

Enable and Command Registers	Double-Buffered Registers	Static Registers	Read-Only Registers	Normal I/O Registers
TCDn.CTRLA (ENABLE bit)	TCDn.DLYCTRL	TCDn.CTRLA[1] (All bits except ENABLE bit)	TCDn.STATUS	TCDn.INTCTRL
TCDn.CTRLE	TCDn.DLYVAL	TCDn.CTRLB	TCDn.CAPTUREA	TCDn.INTFLAGS
	TCDn.DITCTRL	TCDn.CTRLC	TCDn.CAPTUREB	
	TCDn.DITVAL	TCDn.CTRLD		
	TCDn.DBGCTRL	TCDn.EVCTRLA		
	TCDn.CMPASET	TCDn.EVCTRLB		
	TCDn.CMPACLR	TCDn.INPUTCTRLA		
	TCDn.CMPBSET	TCDn.INPUTCTRLB		
	TCDn.CMPBCLR	TCDn.FAULTCTRL[2]		

Note:

1. The bits in the Control A (TCDn.CTRLA) register are enable-protected, except the ENABLE bit. They can only be written when ENABLE is written to '0' first.
2. This register is protected by the Configuration Change Protection Mechanism, requiring a timed write procedure for changing its value settings.

Enable and Command Registers

Because of the synchronization between the clock domains, it is only possible to change the ENABLE bit in the Control A (TCDn.CTRLA) register, while the Enable Ready (ENRDY) bit in the Status (TCDn.STATUS) register is '1'.

The Control E (TCDn.CTRLE) register is automatically synchronized to the TCD core domain when the TCD is enabled and as long as no synchronization is ongoing already. Check if the Command Ready (CCMDRDY) bit in TCDn.STATUS is '1' to ensure that it is possible to issue a new command. TCDn.CTRLE is a strobe register that will clear itself when the command is sent.

Double-Buffered Registers

The double-buffered registers can be updated in normal I/O writes, while TCD is enabled and no synchronization between the two clock domains is ongoing. Check that the CMDRDY bit in TCDn.STATUS is '1' to ensure that it is possible to update the double-buffered registers. The values will be synchronized to the TCD core domain when a synchronization command is sent or when TCD is enabled.

Table 22-3. Issuing Synchronization Command

Synchronization Issuing Bit	Double Register Update
CTRLC.AUPDATE	Every time the CMPBCLRH register is written, the synchronization occurs at the end of the TCD cycle.
CTRLE.SYNC (1)	Occurs once, as soon as the SYNC bit is synchronized with the TDC domain.
CTRLE.SYNCEOC (1)	Occurs once at the end of the next TCD cycle.

Note:

1. If synchronization is already ongoing, the action has no effect.

Static Registers

Static registers cannot be updated while TCD is enabled. Therefore, these registers must be configured before enabling TCD. To see if TCD is enabled, check if ENABLE in TCDn.CTRLA is read as '1'.

Normal I/O and Read-Only Registers

Normal I/O and read-only registers are not constrained by any synchronization between the domains. The read-only registers inform about synchronization status and values synchronized from the core domain.

22.3.3.2 Waveform Generation Modes

The TCD provides four different Waveform Generation modes controlled by the Waveform Generation Mode (WGMODE) bit field in the Control B (TCDn.CTRLB) register. The Waveform Generation modes are:

- One Ramp mode
- Two Ramp mode
- Four Ramp mode
- Dual Slope mode

The Waveform Generation modes determine how the counter is counting during a TCD cycle and how the compare values influence the waveform. A TCD cycle is split into these states:

- Dead time WOA (DTA)
- On time WOA (OTA)
- Dead time WOB (DTB)

- On time WOB (OTB)

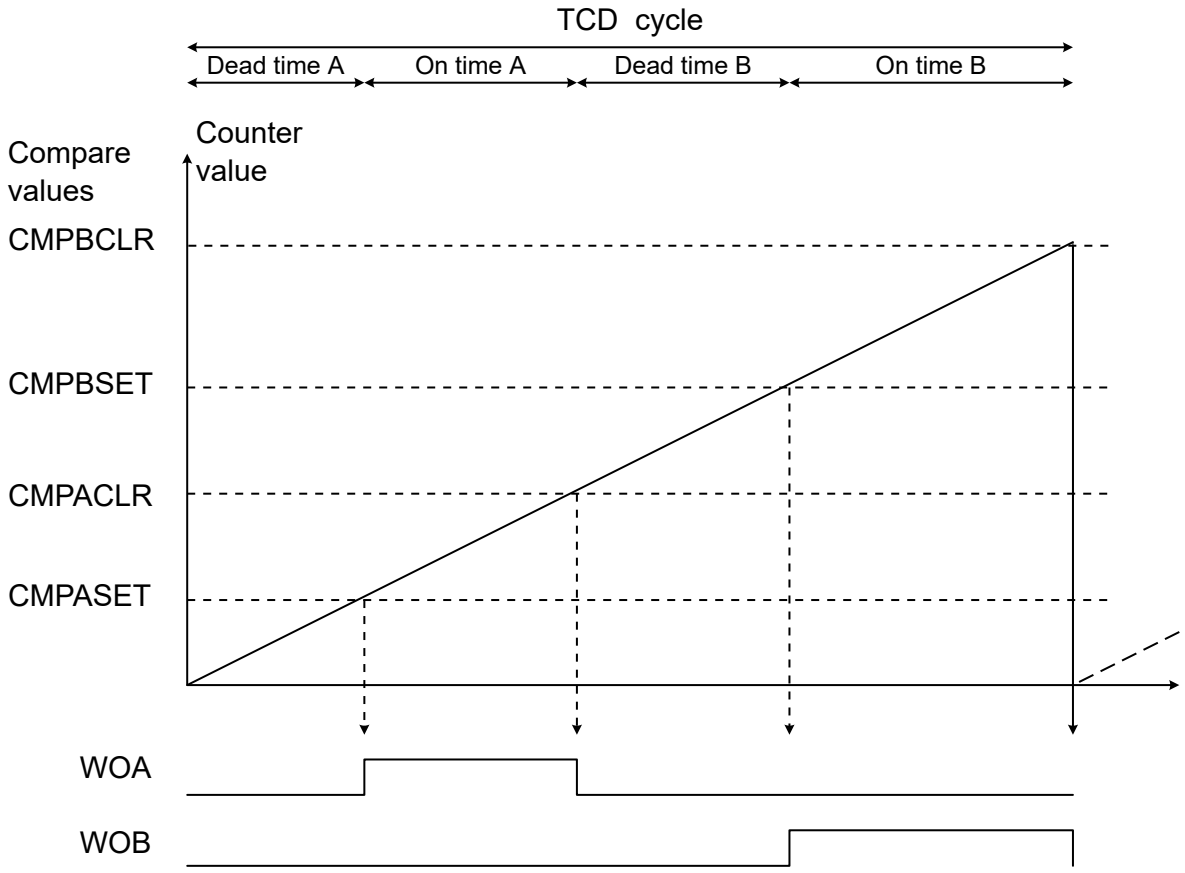
The Compare A Set (CMPASET), Compare A Clear (CMPACLR), Compare B Set (CMPBSET) and Compare B Clear (CMPBCLR) compare values define when each state ends and the next begins.

22.3.3.2.1 One Ramp Mode

In One Ramp mode, the TCD counter counts up until it reaches the CMPBCLR value. Then, the TCD cycle is completed, and the counter restarts from `0x000`, beginning a new TCD cycle. The TCD cycle period is:

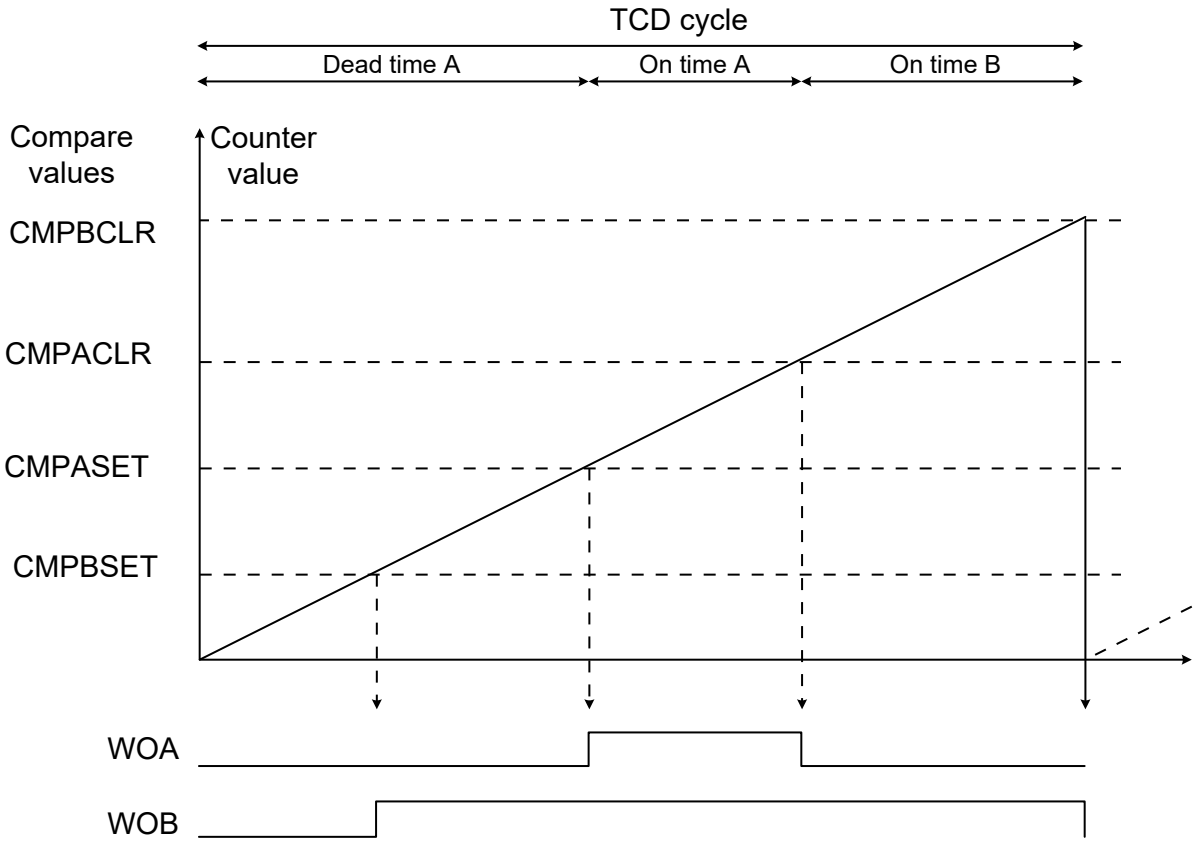
$$T_{\text{TCD\_cycle}} = \frac{(\text{CMPBCLR} + 1)}{f_{\text{CLK\_TCD\_CNT}}}$$

Figure 22-3. One Ramp Mode



In the figure above, *CMPASET* < *CMPACLR* < *CMPBSET* < *CMPBCLR*. In One Ramp mode, this is required to avoid overlapping outputs during the on time. The figure below is an example where *CMPBSET* < *CMPASET* < *CMPACLR* < *CMPBCLR*, which has overlapping outputs during the on time.

Figure 22-4. One Ramp Mode with *CMPBSET* < *CMPASET*



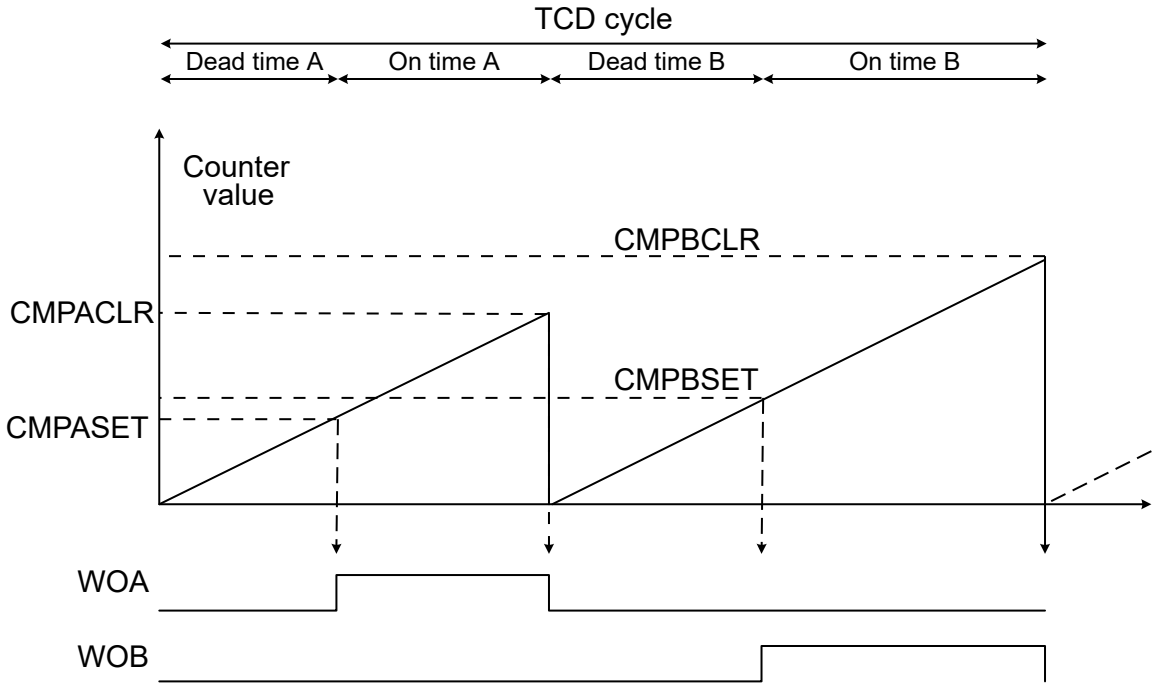
A match with CMPBCLR will always result in all outputs being cleared. If any of the other compare values are bigger than CMPBCLR, their associated effect will never occur. If the CMPACLR is smaller than the CMPASET value, the clear value will not have any effect.

22.3.3.2.2 Two Ramp Mode

In Two Ramp mode, the TCD counter counts up until it reaches the CMPACLR value, then it resets and counts up until it reaches the CMPBCLR value. Then, the TCD cycle is completed, and the counter restarts from `0x000`, beginning a new TCD cycle. The TCD cycle period is given by:

$$T_{\mathrm{TCD\_cycle}} = \frac{(\mathrm{CMPACLR} + 1 + \mathrm{CMPBCLR} + 1)}{f_{\mathrm{CLK\_TCD\_CNT}}}$$

Figure 22-5. Two Ramp Mode



In the figure above, *CMPASET* < *CMPACLR* and *CMPBSET* < *CMPBCLR*. This causes the outputs to go high. There are no restrictions on the CMPASET and CMPACLR compared to the CMPBSET and CMPBCLR values.

In Two Ramp mode, it is not possible to get overlapping outputs without using the override feature. Even if *CMPASET/CMPBSET* > *CMPACLR/CMPBCLR*, the counter resets at CMPACLR/CMPBCLR and will never reach CMPASET/CMPBSET.

22.3.3.2.3 Four Ramp Mode

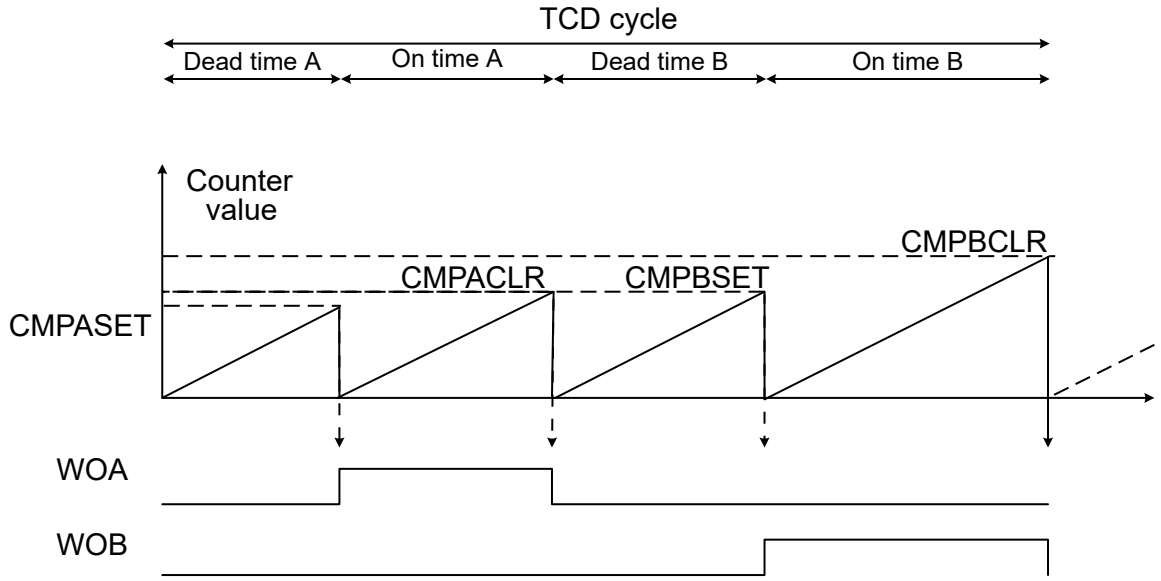
In Four Ramp mode, the TCD cycle follows this pattern:

1. A TCD cycle begins with the TCD counter counting up from zero until it reaches the CMPASET value, and resets to zero.
2. The counter counts up until it reaches the CMPACLR value, and resets to zero.
3. The counter counts up until it reaches the CMPBSET value, and resets to zero.
4. The counter counts up until it reaches the CMPBCLR value, and ends the TCD cycle by resetting to zero.

The TCD cycle period is given by:

$$T_{\mathrm{TCD\_cycle}} = \frac{(\mathrm{CMPASET}+1)+(\mathrm{CMPACLR}+1)+(\mathrm{CMPBSET}+1)+(\mathrm{CMPBCLR}+1)}{f_{\mathrm{CLK\_TCD\_CNT}}}$$

Figure 22-6. Four Ramp Mode



There are no restrictions regarding the compare values, because there are no dependencies between them.

In Four Ramp mode, it is not possible to get overlapping outputs without using the override feature.

22.3.3.2.4 Dual Slope Mode

In Dual Slope mode, a TCD cycle consists of the TCD counter counting down from CMPBCLR value to zero, and up again to the CMPBCLR value. This gives a TCD cycle period:

$$T_{\mathrm{TCD\_cycle}} = \frac{2 \times (\mathrm{CMPBCLR} + 1)}{f_{\mathrm{CLK\_TCD\_CNT}}}$$

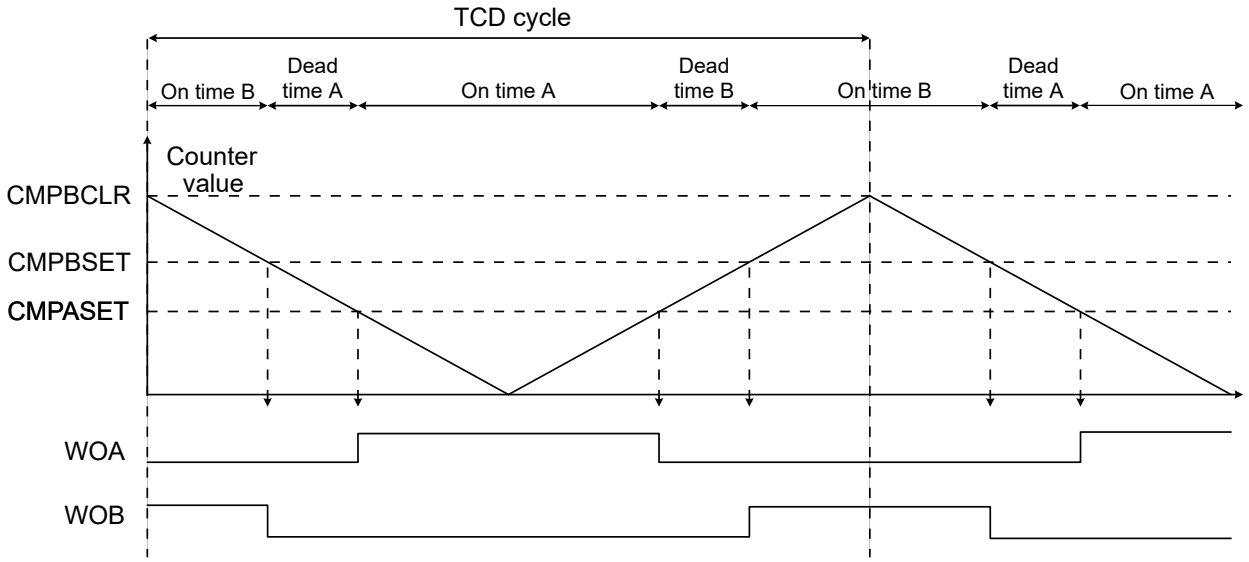
The WOA output is set when the TCD counter counts down and matches the CMPASET value. WOA is cleared when the TCD counter counts up and matches the CMPASET value.

The WOB output is set when the TCD counter counts up and matches the CMPBSET value. WOB is cleared when the TCD counter counts down and matches the CMPBSET value.

The outputs will overlap if *CMPASET* > *CMPBSET*.

CMPACLR is not used in Dual Slope mode. Writing a value to CMPACLR has no effect.

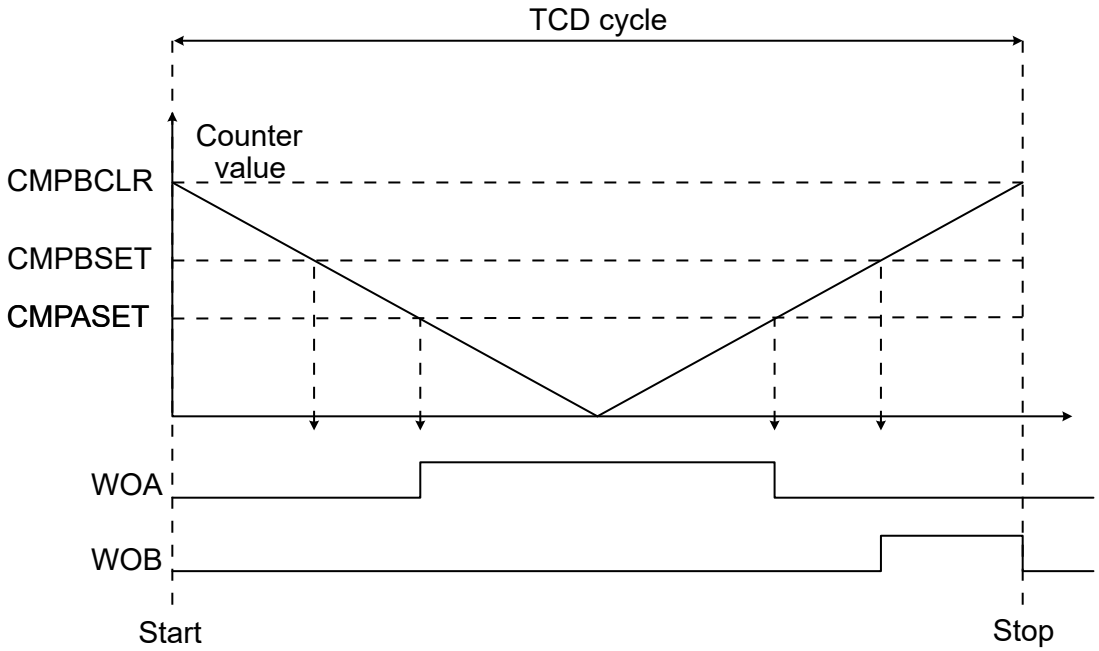
Figure 22-7. Dual Slope Mode



When starting the TCD in Dual Slope mode, the TCD counter starts at the CMPBCLR value and counts down. In the first cycle, the WOB will not be set until the TCD counter matches the CMPBSET value when counting up.

When the Disable at End of Cycle Strobe (DISEOC) bit in the Control E (TCDn.CTRLE) register is set, the TCD will automatically be disabled at the end of the TCD cycle.

Figure 22-8. Dual Slope Mode Starting and Stopping



22.3.3.3 Disabling TCD

Disabling the TCD can be done in two different ways:

1. By writing a '0' to the ENABLE bit in the Control A (TCDn.CTRLA) register. This disables the TCD instantly when synchronized to the TCD core domain.
2. By writing a '1' to the Disable at End of Cycle Strobe (DISEOC) bit in the Control E (TCDn.CTRLE) register. This disables the TCD at the end of the TCD cycle.

22.3.3.4 TCD Inputs

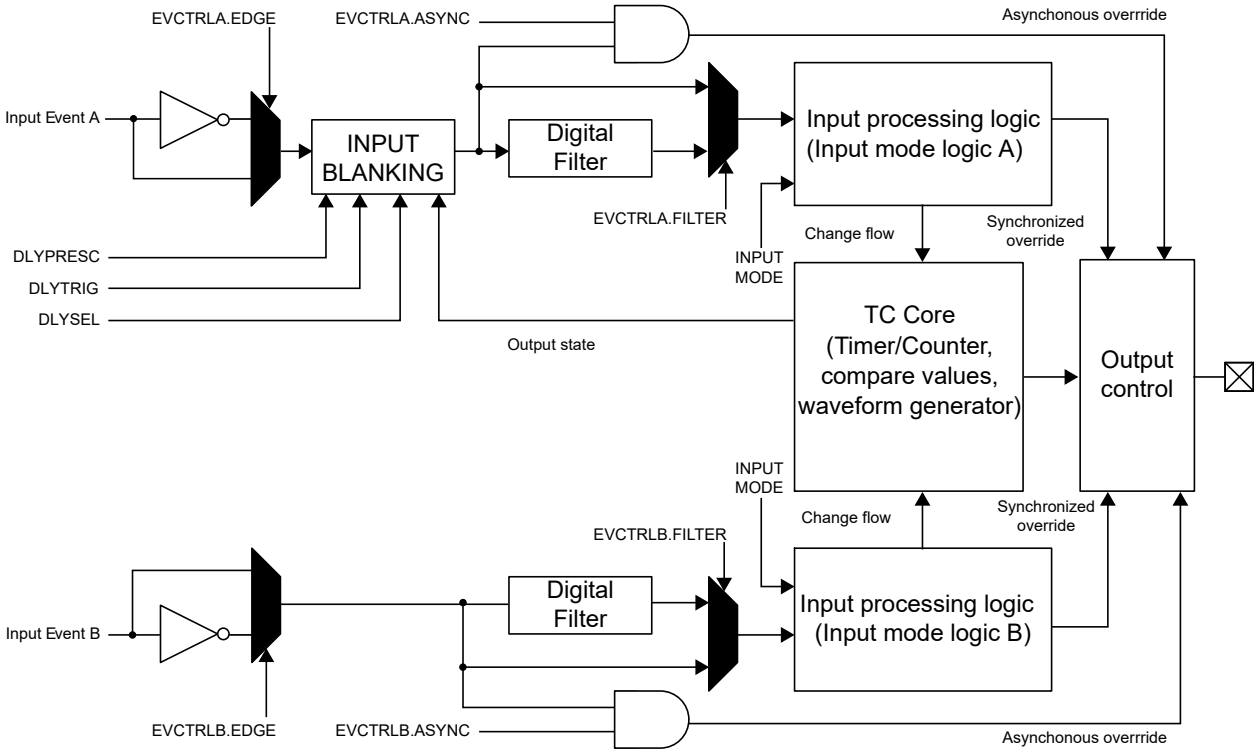
The TCD has two inputs connected to the Event System: input A and input B. Each input has a functionality connected to the corresponding output (WOA and WOB). This functionality is controlled by the Event Control (TCDn.EVCTRLA and TCDn.EVCTRLB) registers and the Input Control (TCDn.INPUTCTRLA and TCDn.INPUTCTRLB) registers.

To enable the input events, write a '1' to the Trigger Event Input Enable (TRIGEI) bit in the corresponding Event Control (TCDn.EVCTRLA or TCDn.EVCTRLB) register. The inputs will be used as a Fault detect by default, but they can also be used as a capture trigger. To enable a capture trigger, write a '1' to the ACTION bit in the corresponding Event Control (TCDn.EVCTRLA or TCDn.EVCTRLB) register. To disable Fault detect, the INPUTMODE bit field in the corresponding Input Control (TCDn.INPUTCTRLA or TCDn.INPUTCTRLB) register must be written to '0'.

There are ten different input modes for the Fault detection. The two inputs have the same functionality, except for input blanking which is only supported by input A. Input blanking is configured by the Delay Control (TCDn.DLYCTRL) register and the Delay Value (TCDn.DLYVAL) register.

The inputs are connected to the Event System. The connections between the event source and the TCD input must be configured in the Event System.

Figure 22-9. TCD Input Overview



There is a delay of two/three clock cycles on the TCD synchronizer clock between receiving the input event, processing it, and overriding the outputs. If using the asynchronous event detection, the outputs will override instantly outside the input processing.

22.3.3.4.1 Input Blanking

Input blanking functionality masks out the input events for a programmable time in a selectable part of the TCD cycle. Input blanking can be used to mask out 'false' input events triggered right after changes on the outputs occur.

Input blanking can be enabled by configuring the Delay Select (DLYSEL) bit field in the Delay Control (TCDn.DLYCTRL) register. The trigger source is selected by the Delay Trigger (DLYTRIG) bit field in TCDn.DLYCTRL.

Input blanking uses the delay clock. After a trigger, a counter counts up until the Delay Value (DLYVAL) bit field in the Delay Value (TCDn.DLYVAL) register is reached. Afterward, input blanking is turned off. The TCD delay clock is a prescaled version of the synchronizer clock (CLK_TCD_SYNC). The division factor is set by the Delay Prescaler (DLYPRESC) bit field in the Delay Control (TCDn.DLYCTRL) register. The duration of the input blanking is given by:

$$t_{\mathrm{BLANK}} = \frac{\mathrm{DLYPRESC\_division\_factor} \times \mathrm{DLYVAL}}{f_{\mathrm{CLK\_TCD\_SYNC}}}$$

Input blanking uses the same logic as the programmable output event. For this reason, it is not possible to use both at the same time.

22.3.3.4.2 Digital Filter

The digital filter for event input x is enabled by writing a '1' to the FILTER bit in the corresponding Event Control (TCDn.EVCTRLA or TCDn.EVCTRLB) register. When the digital filter is enabled, any pulse lasting less than four counter clock cycles will be filtered out. Any change on the incoming event will, therefore, take four counter clock cycles before it affects the input processing logic.

22.3.3.4.3 Asynchronous Event Detection

To enable asynchronous event detection on an input event, the Event Configuration (CFG) bit field in the corresponding Event Control (TCDn.EVCTRLA or TCDn.EVCTRLB) register must be configured accordingly.

The asynchronous event detection makes it possible to asynchronously override the output when the input event occurs. What the input event will do depends on the input mode. The outputs have direct override while the counter flow will be changed when the event is synchronized to the synchronizer clock (CLK_TCD_SYNC).

It is not possible to use asynchronous event detection and digital filter at the same time.

22.3.3.4.4 Software Commands

The following table displays the commands for the TCD module.

Table 22-4. Software Commands

Trigger	Software Command
The SYNCEOC bit in the TCDn.CTRLE register	Update the double-buffered registers at the end of the TCD cycle
The SYNC bit in the TCDn.CTRLE register	Update the double-buffered registers
The RESTART bit in the TCDn.CTRLE register	Restart the TCD counter
The SCAPTUREA bit in the TCDn.CTRLE register	Capture to Capture A (TCDn.CAPTUREAL/H) register
The SCAPTUREB bit in the TCDn.CTRLE register	Capture to Capture B (TCDn.CAPTUREBL/H) register

22.3.3.4.5 Input Modes

The user can select between ten input modes. The selection is done by writing to the Input Mode (INPUTMODE) bit field in the Input Control (TCDn.INPUTCTRLA and TCDn.INPUTCTRLB) registers.

Input Modes Validity

Not all input modes work in all Waveform Generation modes. The table below shows the Waveform Generation modes in which the different input modes are valid.

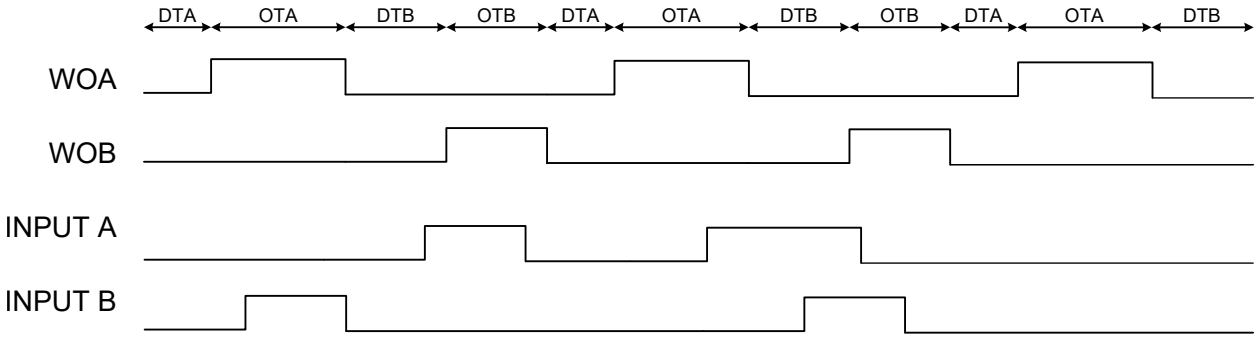
Table 22-5. Input Modes Validity

INPUTMODE	One Ramp Mode	Two Ramp Mode	Four Ramp Mode	Dual Slope Mode
0	Valid	Valid	Valid	Valid
1	Valid	Valid	Valid	Do not use
2	Do not use	Valid	Valid	Do not use
3	Do not use	Valid	Valid	Do not use
4	Valid	Valid	Valid	Valid
5	Do not use	Valid	Valid	Do not use
6	Do not use	Valid	Valid	Do not use
7	Valid	Valid	Valid	Valid
8	Valid	Valid	Valid	Do not use
9	Valid	Valid	Valid	Do not use
10	Valid	Valid	Valid	Do not use

Input Mode 0: Input Has No Action

In Input mode 0, the inputs do not affect the outputs, but they can still trigger captures and interrupts if enabled.

Figure 22-10. Input Mode 0

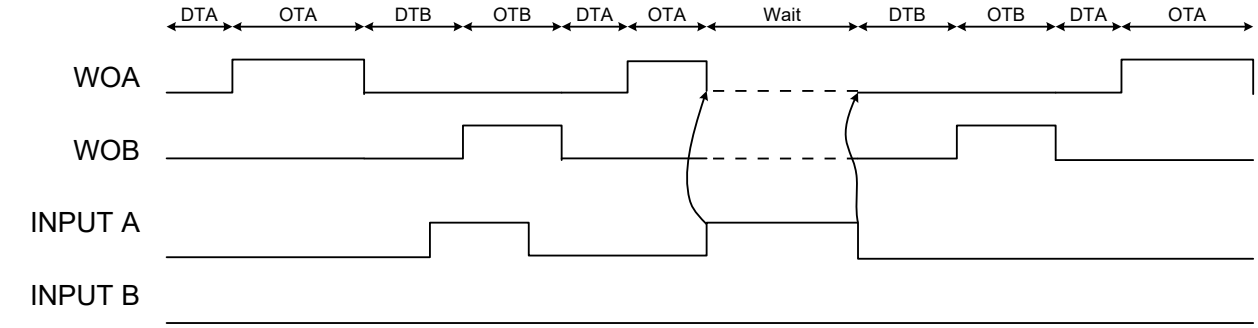


Input Mode 1: Stop Output, Jump to Opposite Compare Cycle, and Wait

An input event in Input mode 1 will stop the output signal, jump to the opposite dead time, and wait until the input event goes low before the TCD counter continues.

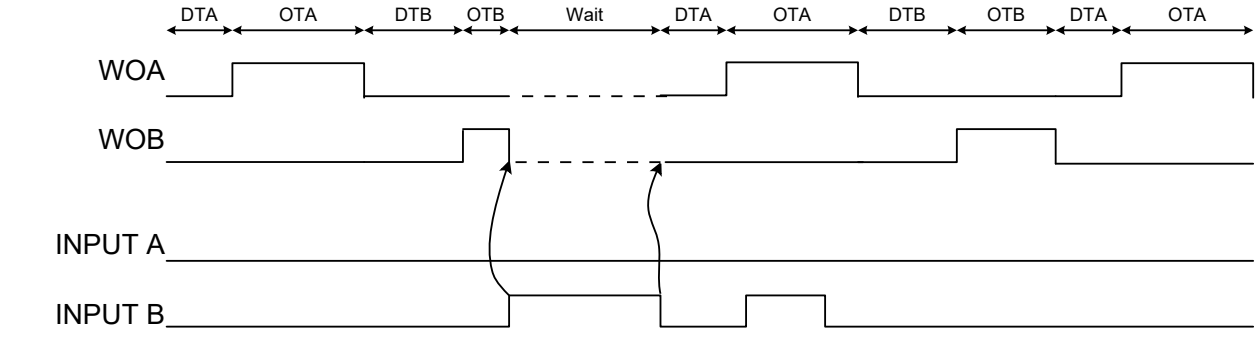
If Input mode 1 is used on input A, an event will only have an effect if the TCD is in dead time A or on time A, and it will only affect the WOA output. When the event is done, the TCD counter starts at dead time B.

Figure 22-11. Input Mode 1 on Input A



If Input mode 1 is used on input B, an event will only have an effect if the TCD is in dead time B or on time B, and it will only affect the WOB output. When the event is done, the TCD counter starts at dead time A.

Figure 22-12. Input Mode 1 on Input B

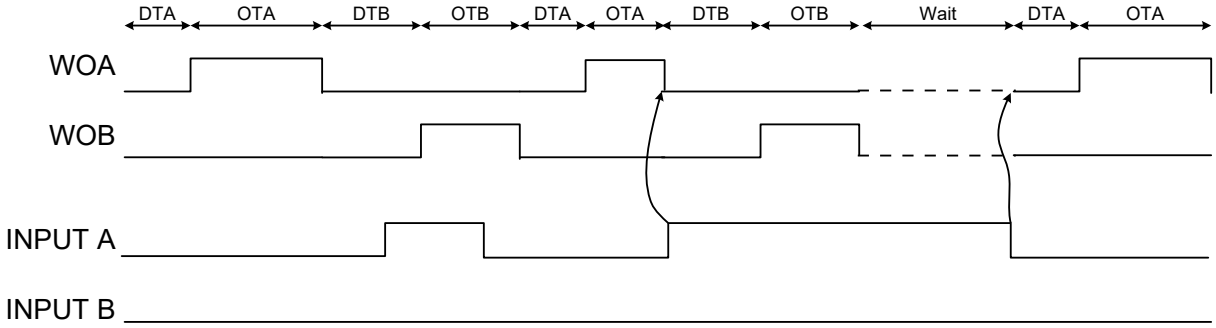


Input Mode 2: Stop Output, Execute Opposite Compare Cycle, and Wait

An input event in Input mode 2 will stop the output signal, execute to the opposite dead time and on time, and then wait until the input event goes low before the TCD counter continues. If the input is done before the opposite dead time and on time have finished, there will be no waiting, but the opposite dead time and on time will continue.

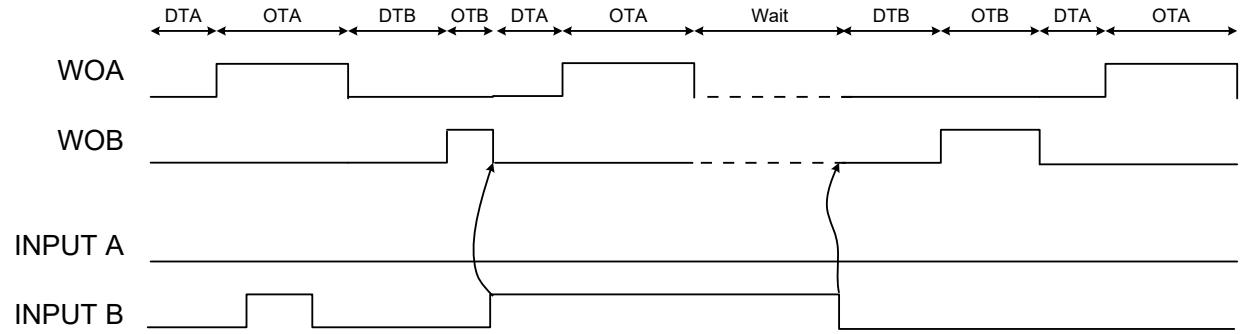
If Input mode 2 is used on input A, an event will only have an effect if the TCD is in dead time A or on time A, and will only affect the WOA output.

Figure 22-13. Input Mode 2 on Input A



If Input mode 2 is used on input B, an event will only have an effect if the TCD is in dead time B or on time B, and it will only affect the WOB output.

Figure 22-14. Input Mode 2 on Input B

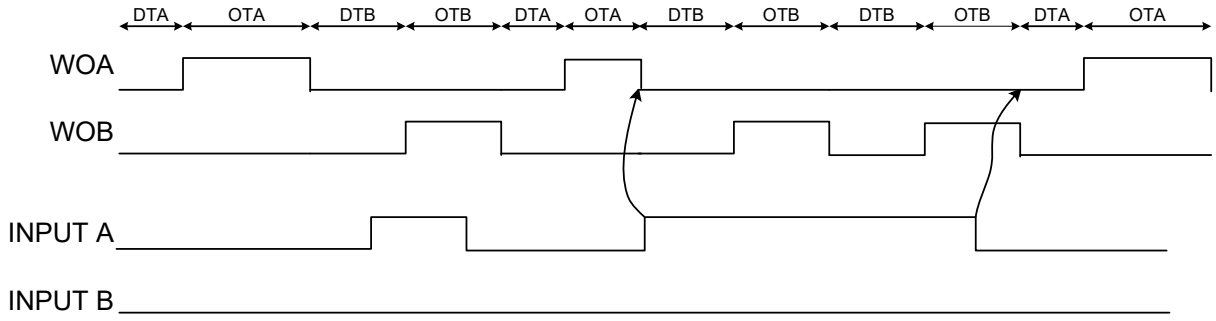


Input Mode 3: Stop Output, Execute Opposite Compare Cycle while Fault Active

An input event in Input mode 3 will stop the output signal and start executing the opposite dead time and on time repetitively, as long as the Fault/input is active. When the input is released, the ongoing dead time and/or on time will finish, and then the normal flow will start.

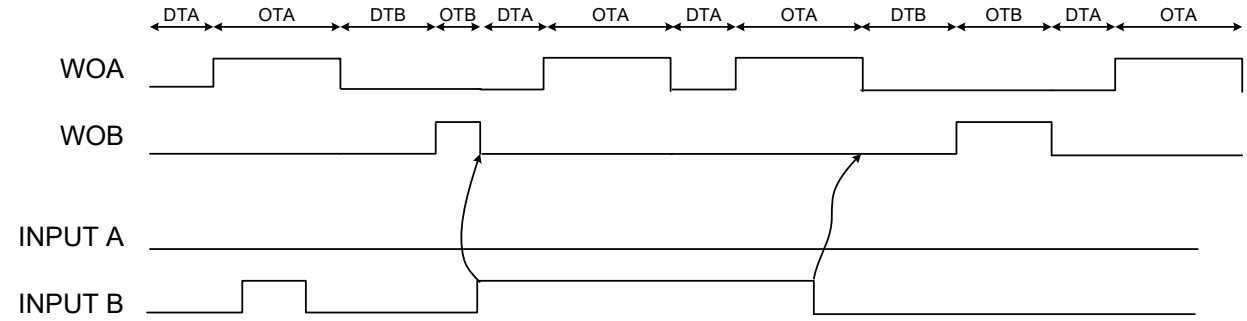
If Input mode 3 is used on input A, an event will only have an effect if the TCD is in dead time A or on time A.

Figure 22-15. Input Mode 3 on Input A



If Input mode 3 is used on input B, an event will only have an effect if the TCD is in dead time B or on time B.

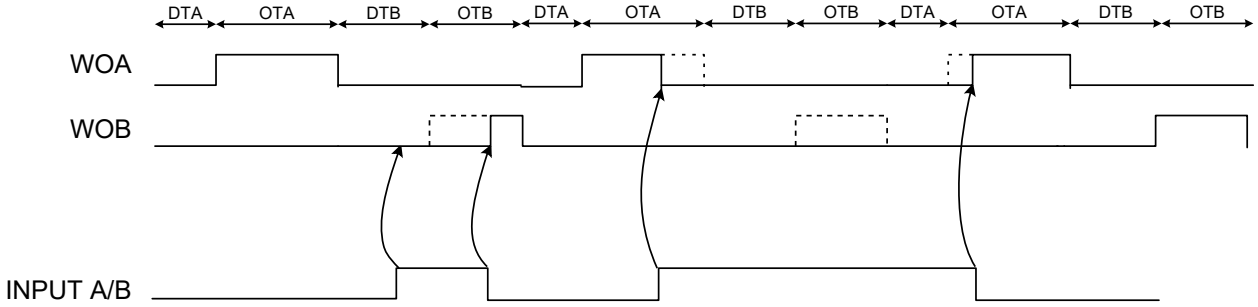
Figure 22-16. Input Mode 3 on Input B



Input Mode 4: Stop all Outputs, Maintain Frequency

When Input mode 4 is used, both input A and input B will give the same functionality. An input event will deactivate the outputs as long as the event is active. The TCD counter will not be affected by events in this input mode.

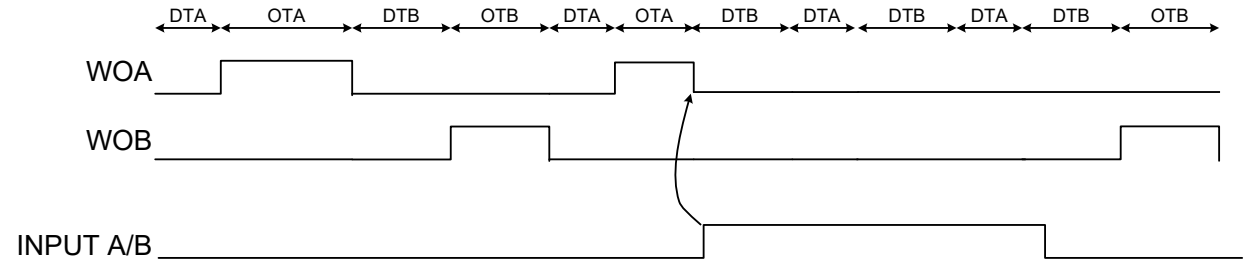
Figure 22-17. Input Mode 4



Input Mode 5: Stop all Outputs, Execute Dead Time while Fault Active

When Input mode 5 is used, both input A and input B give the same functionality. The input event stops the outputs and starts on the opposite dead time if it occurs during an on time. If the event occurs during dead time, the dead time will continue until the next on time is scheduled to start. Though, if the input is still active, the cycle will continue with the other dead time. As long as the input event is active, alternating dead times will occur. When the input event stops, the ongoing dead time will finish, and the next on time will continue in the normal flow.

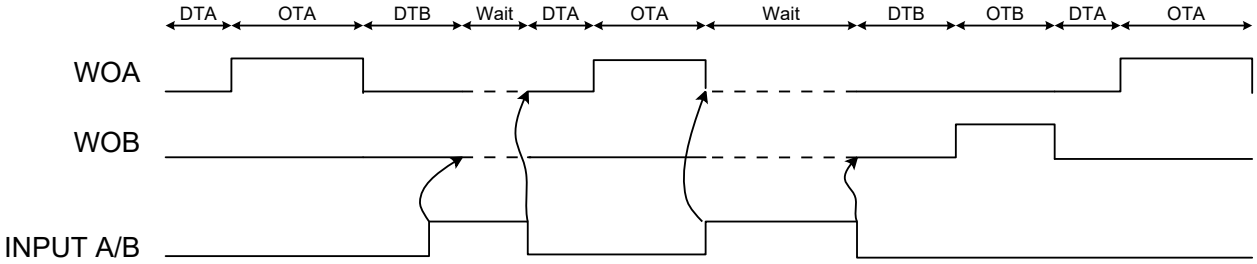
Figure 22-18. Input Mode 5



Input Mode 6: Stop All Outputs, Jump to Next Compare Cycle, and Wait

When Input mode 6 is used, both input A and input B will give the same functionality. The input event stops the outputs and jumps to the opposite dead time if it occurs during an on time. If the event occurs during dead time, the dead time will continue until the next on time is scheduled to start. As long as the input event is active, the TCD counter will wait. When the input event stops, the next dead time will start, and normal flow will continue.

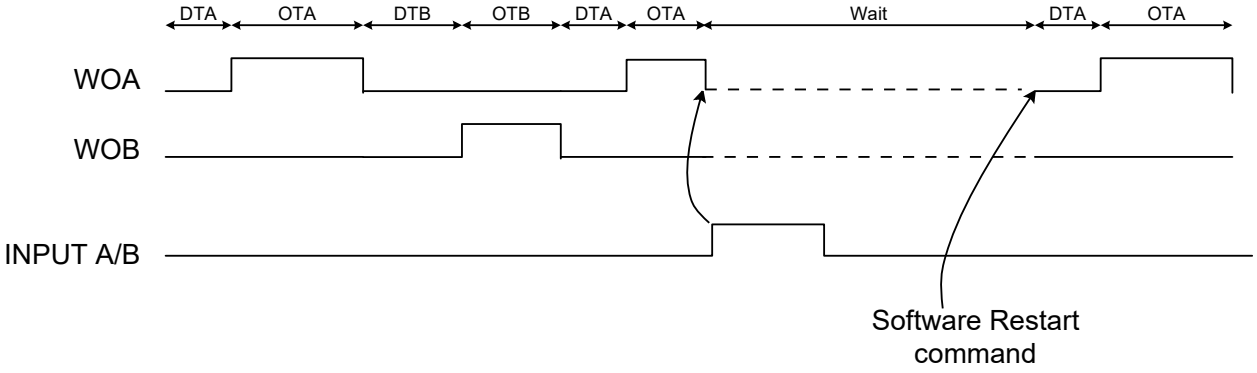
Figure 22-19. Input Mode 6



Input Mode 7: Stop all Outputs, Wait for Software Action

When Input mode 7 is used, both input A and input B will give the same functionality. The input events stop the outputs and the TCD counter. It will be stopped until a Restart command is given. If the input event is still high when the Restart command (RESTART bit in TCDn.CTRLE register) is given, it will stop again. When the TCD counter restarts, it will always start on dead time A.

Figure 22-20. Input Mode 7

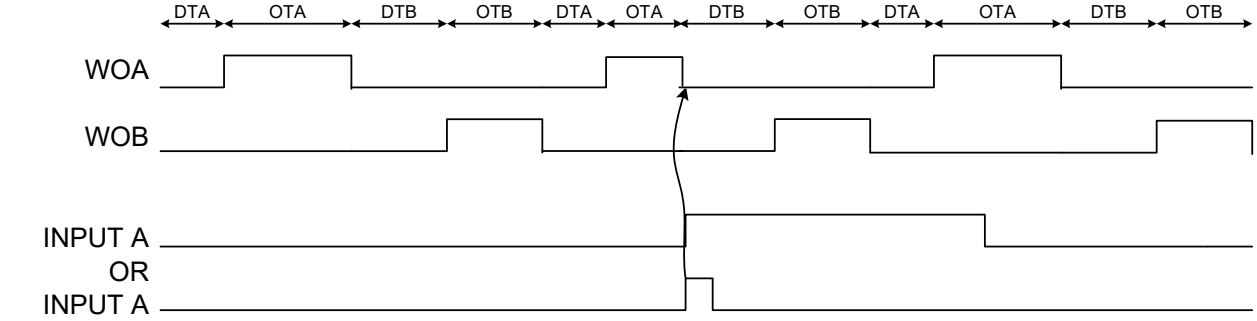


Input Mode 8: Stop Output on Edge, Jump to Next Compare Cycle

In Input mode 8, a positive edge on the input event while the corresponding output is ON will cause the output to stop and the TCD counter to jump to the opposite dead time.

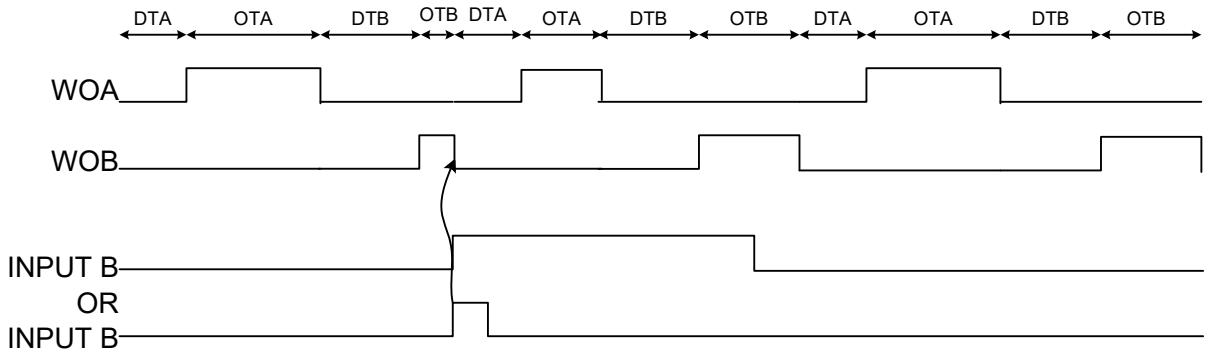
If Input mode 8 is used on input A and a positive edge on the input event occurs while in on time A, the TCD counter jumps to dead time B.

Figure 22-21. Input Mode 8 on Input A



If Input mode 8 is used on input B and a positive edge on the input event occurs while in on time B, the TCD counter jumps to dead time A.

Figure 22-22. Input Mode 8 on Input B

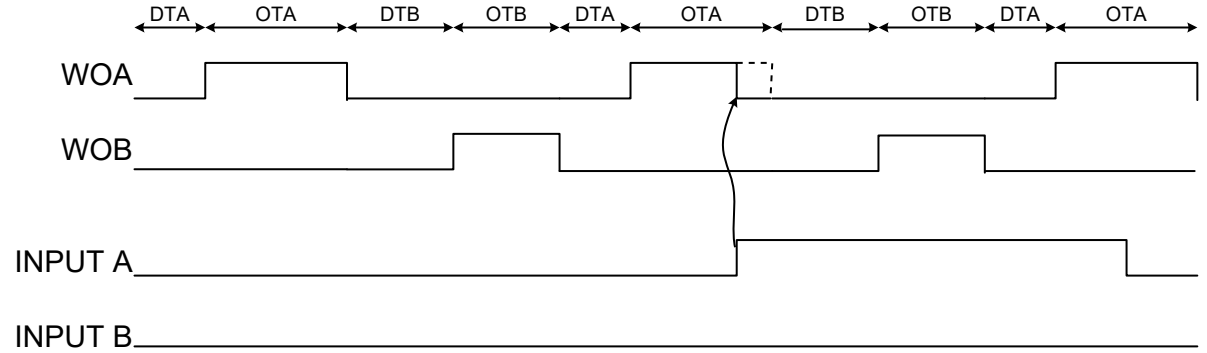


Input Mode 9: Stop Output on Edge, Maintain Frequency

In Input mode 9, a positive edge on the input event while the corresponding output is ON will cause the output to stop during the rest of the on time. The TCD counter will not be affected by the event, only the output.

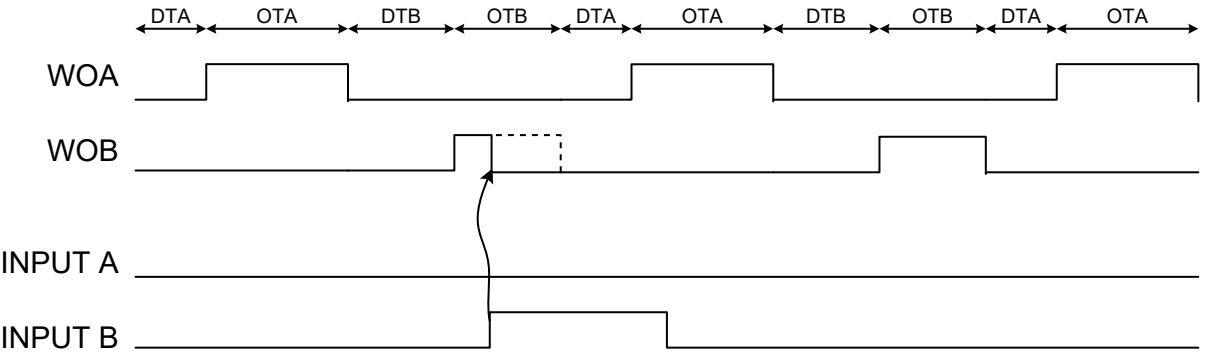
If Input mode 9 is used on input A and a positive edge on the input event occurs while in on time A, the output will be OFF for the rest of the on time.

Figure 22-23. Input Mode 9 on Input A



If Input mode 9 is used on input B and a positive edge on the input event occurs while in on time B, the output will be OFF for the rest of the on time.

Figure 22-24. Input Mode 9 on Input B

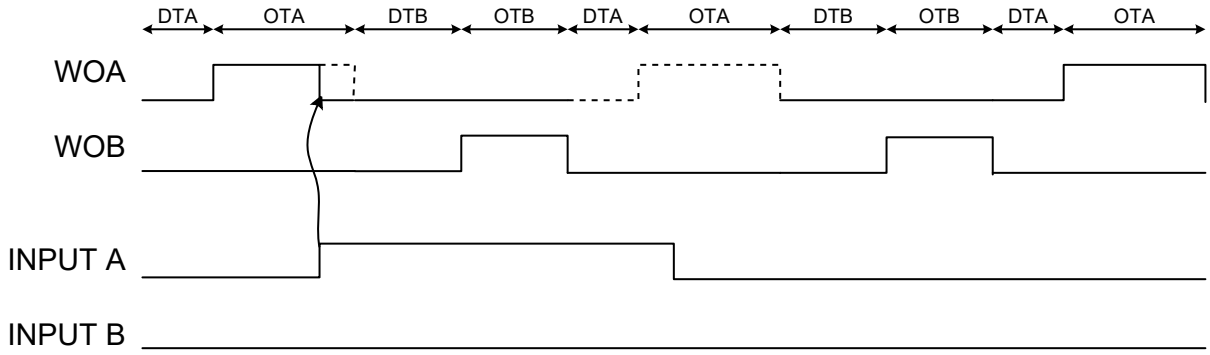


Input Mode 10: Stop Output at Level, Maintain Frequency

In Input mode 10, the input event will cause the corresponding output to stop, as long as the input is active. If the input goes low while there must have been an on time on the corresponding output, the output will be deactivated for the rest of the on time. The TCD counter is not affected by the event, only the output.

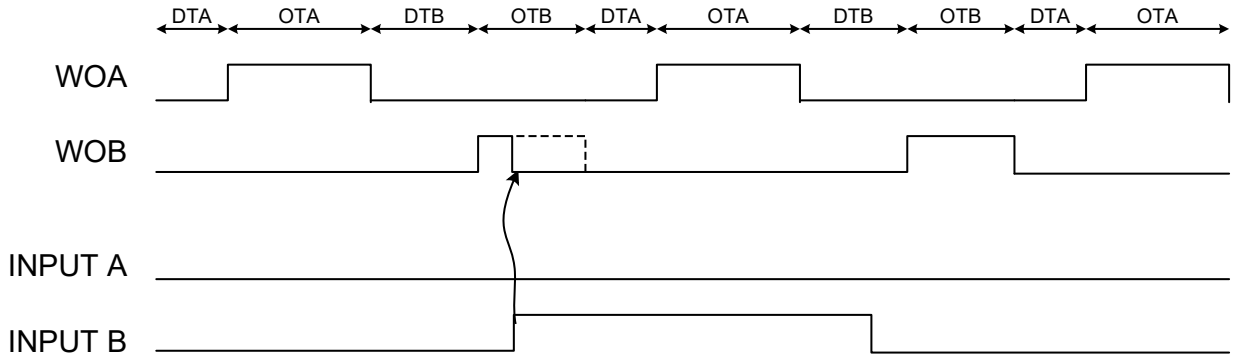
If Input mode 10 is used on input A and an input event occurs, the WOA will be OFF as long as the event lasts. If released during an on time, it will be OFF for the rest of the on time.

Figure 22-25. Input Mode 10 on Input A



If Input mode 10 is used on input B and an input event occurs, the WOB will be OFF as long as the event lasts. If released during an on time, it will be OFF for the rest of the on time.

Figure 22-26. Input Mode 10 on Input B



Input Mode Summary

Table 22-6 summarizes the conditions, as illustrated in the timing diagrams of the preceding sections.

Table 22-6. Input Mode Summary

INPUTMODE	Trigger → Output Affected	Fault On/Active	Fault Release/Inactive
0	-	No action.	No action.
1	Input A→WOA	End the current on time and wait.	Start with dead time for the other compare.
	Input B→WOB		
2	Input A→WOA	End the current on time, execute the other compare cycle and wait.	Start with dead time for the current compare.
	Input B→WOB		
3	Input A→WOA	Execute the current on time, then execute the other compare cycle repetitively.	Re-enable the current compare cycle.
	Input B→WOB		
4	Input A→{WOA, WOB}	Deactivate the outputs.	
	Input B→{WOA, WOB}		
5	Input A→{WOA, WOB}	Execute dead time only.	
	Input B→{WOA, WOB}		
6	Input A→{WOA, WOB}	End on time and wait.	Start with dead time for the other compare.
	Input B→{WOA, WOB}		

.....continued

INPUTMODE	Trigger → Output Affected	Fault On/Active	Fault Release/Inactive
7	Input A→{WOA, WOB}	End on time and wait for software action.	Start with dead time for the current compare.
	Input B→{WOA, WOB}		
8	Input A→WOA	End the current on time and continue with the other off time.	
	Input B→WOB		
9	Input A→WOA	Block the current on time and continue the sequence.	
	Input B→WOB		
10	Input A→WOA	Deactivate on time until the end of the sequence while the trigger is active.	
	Input B→WOB		
other	-	-	-

Note: When using different modes on each event input, take into consideration possible conflicts, keeping in mind that TCD has a single counter, to avoid unexpected results.

22.3.3.5 Dithering

If it is not possible to achieve the desired frequency because of the prescaler/period selection limitations, dithering can be used to approximate the desired frequency and reduce the waveform drift.

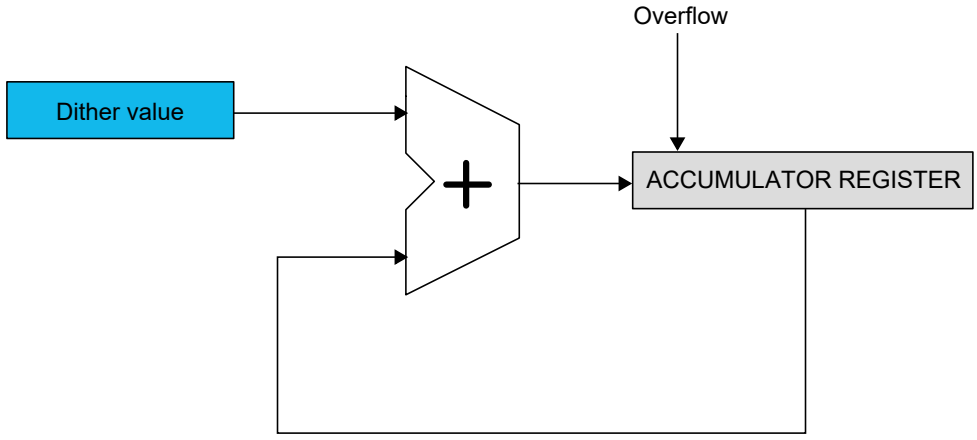
The dither accumulates the fractional error of the counter clock for each cycle. When the fractional error overflows, an additional clock cycle is added to the selected part of the TCD cycle.

Example 22-1. Generate 75 kHz from a 10 MHz Clock

If the timer clock frequency is 10 MHz, it will give the timer a resolution of 100 ns. The desired output frequency is 75 kHz, which means a period of 13333 ns. This period cannot be achieved with a 100 ns resolution as it would require 133.33 cycles. The output period can be set to either 133 cycles (75.188 kHz) or 134 cycles (74.626 kHz).

It is possible to change the period between the two frequencies manually in the firmware to get an average output frequency of 75 kHz (change every third period to 134 cycles). The dither can do this automatically by accumulating the error (0.33 cycles). The accumulator calculates when the accumulated error is larger than one clock cycle. When that happens, an additional cycle is added to the timer period.

Figure 22-27. Dither Logic



The user can select where in the TCD cycle the dither will be added by writing to the Dither Selection (DITHERSEL) bits in the Dither Control (TCDn.DITCTRL) register:

- On time B
- On time A and B
- Dead time B
- Dead time A and B

How much the dithering will affect the TCD cycle time depends on what Waveform Generation mode is used (see Table 22-7). Dithering is not supported in Dual Slope mode.

Table 22-7. Mode-Dependent Dithering Additions to TCD Cycle

WAVEGEN	DITHERSEL in TCDn.DITCTRL	Additional TCD Clock Cycles to TCD Cycle
One Ramp mode	On time B	1
	On time A and B	1
	Dead time B	0
	Dead time A and B	0
Two Ramp mode	On time B	1
	On time A and B	2
	Dead time B	0
	Dead time A and B	0
Four Ramp mode	On time B	1
	On time A and B	2
	Dead time B	1
	Dead time A and B	2
Dual Slope mode	On time B	Not supported
	On time A and B	Not supported
	Dead time B	Not supported
	Dead time A and B	Not supported

The differences in the number of TCD clock cycles added to the TCD cycle are caused by the different number of compare values used by the TCD cycle. For example, in One Ramp mode, only CMPBCLR affects the TCD cycle time.

For DITHERSEL configurations where no extra cycles are added to the TCD cycles, compensation is reached by shortening the following output state.

Example 22-2. DITHERSEL in One Ramp Mode

In One Ramp mode with DITHERSEL selecting dead time B, the dead time B will be increased by one cycle when dither overflow occurs, reducing on time B by one cycle.

22.3.3.6 TCD Counter Capture

The TCD counter is asynchronous to the peripheral clock, so it is not possible to read out the counter value directly. It is possible to capture the TCD counter value, synchronized to the I/O clock domain, in two ways:

- Capture value on input events
- Software capture

The capture logic contains two separate capture blocks, CAPTUREA and CAPTUREB, that can capture and synchronize the TCD counter value to the I/O clock domain. CAPTUREA/B can be triggered by input event A/B or by software.

The capture values can be obtained by reading first TCDn.CAPTUREAL/TCDn.CAPTUREBL and then TCDn.CAPTUREAH/TCDn.CAPTUREBH registers.

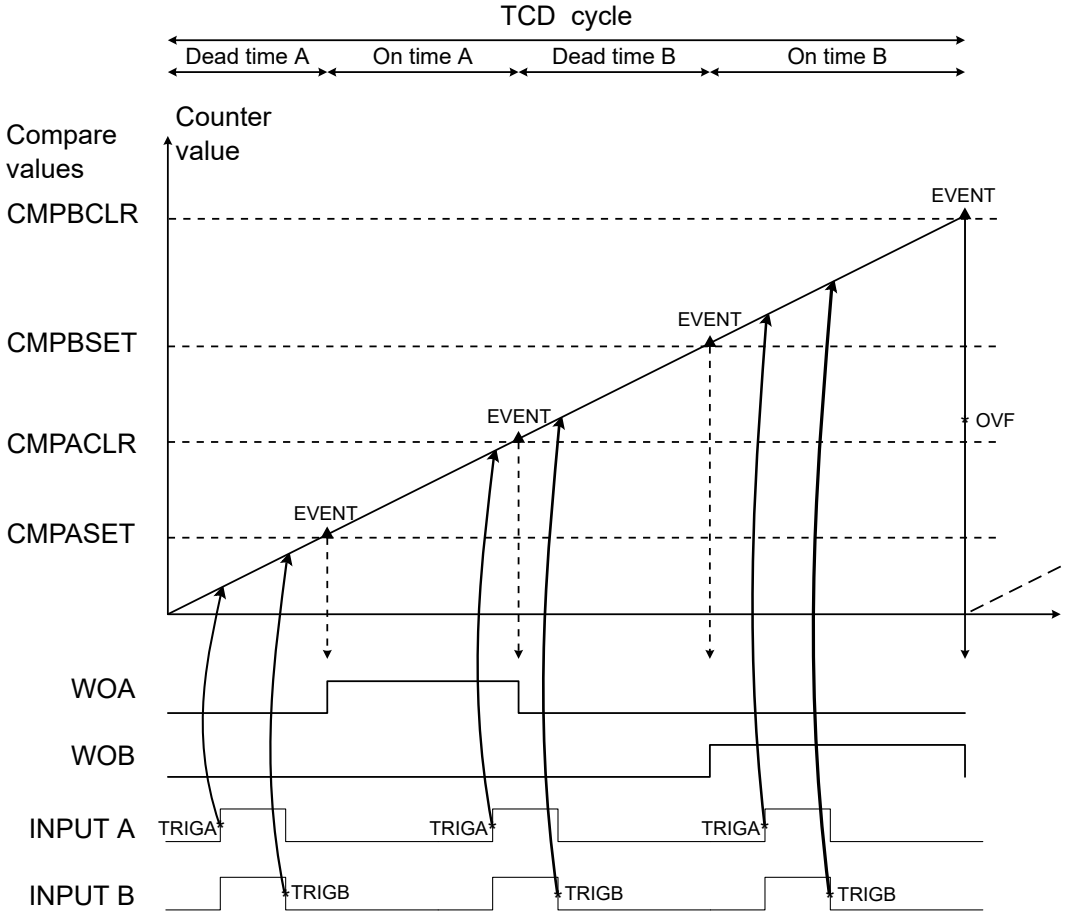
Captures Triggered by Input Events

To enable the capture on an input event, write a '1' to the ACTION bit in the respective Event Control (TCDn.EVCTRLA or TCDn.EVCTRLB) register when configuring an event input.

When a capture has occurred, the TRIGA/B flag is raised in the Interrupt Flags (TCDn.INTFLAGS) register. The corresponding TRIGA/B interrupt can be enabled by writing a '1' to the respective Trigger Interrupt Enable (TRIGA or TRIGB) bit in the Interrupt Control (TCDn.INTCTRL) register. By polling TRIGA or TRIGB in TCDn.INTFLAGS, the user knows that a CAPTURE value is available, and can read out the value by reading first the TCDn.CAPTUREAL or TCDn.CAPTUREBL register and then the TCDn.CAPTUREAH or TCDn.CAPTUREBH register.

Example 22-3. PWM Capture

To perform a PWM capture, connect both event A and event B to the same asynchronous event channel that contains the PWM signal. To get information on the PWM signal, configure one event input to capture the rising edge of the signal. Configure the other event input to capture the falling edge of the signal.



Note:

▲ Event trigger

* Interrupt trigger

Capture Triggered by Software

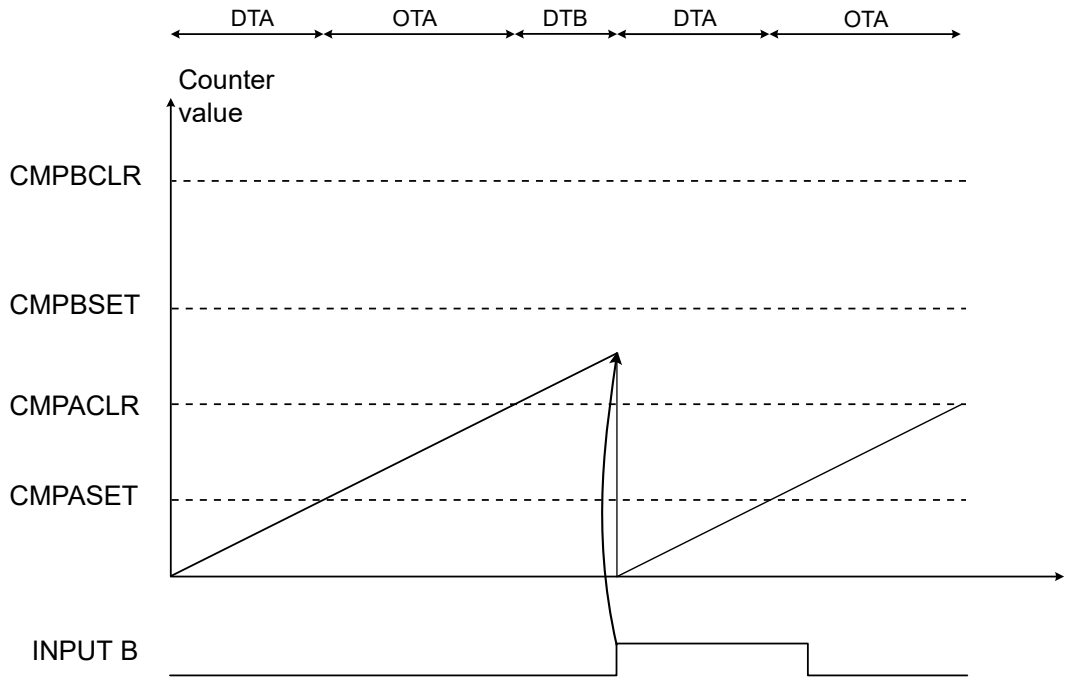
The software can capture the TCD value by writing a '1' to the respective Software Capture A/B Strobe (SCAPTUREx) bit in the Control E (TCDn.CTRLE) register. When this command is executed and the Command Ready (CMDRDY) bit in the Status (TCDn.STATUS) register reads '1' again, the CAPTUREA/B value is available. It can now be read by reading first the TCDn.CAPTUREAL or TCDn.CAPTUREBL register and then the TCDn.CAPTUREAH or TCDn.CAPTUREBH register.

Using Capture Together with Input Modes

The capture functionality can be used together with input modes. The same event will then both capture the counter value and trigger a change in the counter flow, depending on the input mode selected.

Example 22-4. Reset One Ramp Mode by Input Event Capture

In One Ramp mode, the counter can be reset by an input event capture. To achieve this, use input event B and write 0x08 to the INPUTMODE bit field in the Input Control B (TCDn.INPUTCTRLB) register.



22.3.3.7 Output Control

The outputs are configured by writing to the Fault Control (TCDn.FAULTCTRL) register. TCDn.FAULTCTRL is only reset to '0' after a POR reset. During the reset sequence after any Reset, TCDn.FAULTCTRL will get its values from the TCD Fuse (FUSE.TCDCFG).

The Compare x Enable (CMPxEN) bits in TCDn.FAULTCTRL enable the different outputs. The CMPx bits in TCDn.FAULTCTRL set the output values when a Fault is triggered.

The TCD itself generates two different outputs, WOA and WOB. The two additional outputs, WOC and WOD, can be configured by software to be connected to either WOA or WOB by writing the Compare C/D Output Select (CMPCSEL and CMPDSEL) bits in the Control C (TCDn.CTRLC) register.

The user can override the outputs based on the TCD counter state by writing a '1' to the Compare Output Value Override (CMPOVR) bit in the Control C (TCDn.CTRLC) register. The user can then select the output values in the different dead and on times by writing to the Compare Value (CMPAVAL and CMPBVAL) bit fields in the Control D (TCDn.CTRLD) register.

When used in One Ramp mode, WOA will only use the setup for dead time A (DTA) and on time A (OTA) to set the output. WOB will only use dead time B (DTB) and on time B (OTB) values to set the output.

When using the override feature together with Faults detection (input modes), the CMPA (and CMPC/D if WOC/D equals WOA) bit in TCDn.FAULTCTRL must be equal to CMPAVAL[0] and [2] in CTRL. If not, the first cycle after a Fault is detected can have the wrong polarity on the outputs. The same applies to CMPB in the TCDn.FAULTCTRL (and CMPC/D if WOC/D equals WOB) bit, which must be equal to CMPBVAL[0] and [2] in TCDn.CTRLD.

Due to the asynchronous nature of the TCD and that input events can immediately affect the output signal, there is a risk of nanosecond spikes occurring on the output without any load on the pin. The case occurs in any input mode different from '0' and when an input event is triggering. The spike value will always be in the direction of the CMPx values given by the TCDn.FAULTCTRL register.

22.3.4 Events

The TCD can generate the events described in the following table:

Table 22-8. Event Generators in TCD

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCDn	CMPBCLR	The counter matches CMPBCLR	Pulse	CLK_TCD	One CLK_TCD_CNT period
	CMPASET	The counter matches CMPASET			
	CMPBSET	The counter matches CMPBSET			
	PROGEV	Programmable event output(1)			One CLK_TCD_SYNC period

Note:

1. The user can select the trigger and all the compare matches (including CMPACLR). Also, it is possible to delay the output event from 0 to 255 TCD delay cycles.

The three events based on the counter match directly generate event strobes that last for one clock cycle on the TCD counter clock. The programmable output event generates an event strobe that lasts for one clock cycle on the TCD synchronizer clock.

The TCD can receive the events described in the following table:

Table 22-9. Event Users and Available Event Actions in TCD

User Name		Description	Input Detection	Async/Sync
Peripheral	Input			
TCDn	Input A/ Input B	Stop the output, jump to the opposite compare cycle and wait.	Level	Both
		Stop the output, execute the opposite compare cycle and wait.		
		Stop the output, execute the opposite compare cycle while the Fault is active.		
		Stop all outputs, maintain the frequency.		
		Stop all outputs, execute dead time while the Fault is active.		
		Stop all outputs, jump to the next compare cycle and wait.		
		Stop all outputs, wait for software action.		
		Stop the output on the edge, jump to the next compare cycle.	Edge	
		Stop the output on the edge, maintain the frequency.		
		Stop the output at level, maintain the frequency.	Level	

Input A and Input B are TCD event users that detect and act upon the input events. Additional information about input events and how to configure them can be found in the 22.3.3.4 TCD Inputs section. Refer to the Event System (EVSYS) section for more details regarding event types and Event System configuration.

22.3.4.1 Programmable Output Events

The Programmable Output Event (PROGEV) uses the same logic as the input blanking for trigger selection and delay. Therefore, it is not possible to configure the functionalities independently. If the input blanking functionality is used, the output event cannot be delayed, and the trigger used for input blanking will also be used for the output event.

PROGEV is configured in the TCDn.DLYCTRL and TCDn.DLYVAL registers. It is possible to delay the output event by 0 to 255 TCD delay clock cycles. The delayed output event functionality uses the TCD delay clock and counts until the DLYVAL value is reached before the trigger is sent out as an event. The TCD delay clock is a prescaled version of the TCD synchronizer clock (CLK_TCD_SYNC), and the division factor is set by the DLYPRESC bits in the TCDn.DLYCTRL register. The output event will be delayed by the TCD clock period x DLYPRESC division factor x DLYVAL.

22.3.5 Interrupts

Table 22-10. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
OVF	Overflow interrupt	The TCD finishes one TCD cycle.
TRIG	Trigger interrupt	• TRIGA: On event input A • TRIGB: On event input B

When an interrupt condition occurs, the corresponding interrupt flag is set in the Interrupt Flags (TCDn.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the Interrupt Control (TCDn.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

When several interrupt request conditions are supported by an interrupt vector, the interrupt requests are ORed together into one combined interrupt request to the interrupt controller. The user must read the peripheral's INTFLAGS register to determine which of the interrupt conditions are present.

22.3.6 Sleep Mode Operation

The TCD operates in Idle sleep mode and is stopped when entering Standby and Power-Down sleep modes.

22.3.7 Debug Operation

Halting the CPU in Debugging mode will halt the normal operation of the peripheral. This peripheral can be forced to operate with the CPU halted by writing a '1' to the Debug Run (DBGRUN) bit in the Debug Control (TCDn.DBGCTRL) register.

When the Fault Detection (FAULTDET) bit in TCDn.DBGCTRL is written to '1', and the CPU is halted in Debug mode, an event/Fault is created on both input event channels. These events/Faults last as long as the break and can serve as a safeguard in Debug mode, for example, by forcing external components off.

If the peripheral is configured to require periodic service by the CPU through interrupts or similar, improper operation or data loss may result during halted debugging.

22.3.8 Configuration Change Protection

This peripheral has registers that are under Configuration Change Protection (CCP). To write to these registers, a certain key must first be written to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to a protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged.

The following registers are under CCP:

Table 22-11. Registers under Configuration Change Protection in TCD

Register	Key
TCDn.FAULTCTRL	IOREG

22.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		CLKSEL[1:0]		CNTPRES[1:0]		SYNCPRES[1:0]		ENABLE
0x01	CTRLB	7:0							WGMODE[1:0]	
0x02	CTRLC	7:0	CMPDSEL	CMPCSEL			FIFTY		AUPDATE	CMPOVR
0x03	CTRLD	7:0	CMPBVAL[3:0]				CMPAVAL[3:0]			
0x04	CTRLE	7:0	DISEOC			SCAPTUREB	SCAPTUREA	RESTART	SYNC	SYNCEOC
0x05 ... 0x07	Reserved									
0x08	EVCTRLA	7:0	CFG[1:0]			EDGE		ACTION		TRIGEI
0x09	EVCTRLB	7:0	CFG[1:0]			EDGE		ACTION		TRIGEI
0x0A ... 0x0B	Reserved									
0x0C	INTCTRL	7:0					TRIGB	TRIGA		OVF
0x0D	INTFLAGS	7:0					TRIGB	TRIGA		OVF
0x0E	STATUS	7:0	PWMACTB	PWMACTA					CMDRDY	ENRDY
0x0F	Reserved									
0x10	INPUTCTRLA	7:0					INPUTMODE[3:0]			
0x11	INPUTCTRLB	7:0					INPUTMODE[3:0]			
0x12	FAULTCTRL	7:0	CMPDEN	CMPCEN	CMPBEN	CMPAEN	CMPD	CMPC	CMPB	CMPA
0x13	Reserved									
0x14	DLYCTRL	7:0			DLYPRESC[1:0]		DLYTRIG[1:0]		DLYSEL[1:0]	
0x15	DLYVAL	7:0	DLYVAL[7:0]							
0x16 ... 0x17	Reserved									
0x18	DITCTRL	7:0							DITHERSEL[1:0]	
0x19	DITVAL	7:0					DITHER[3:0]			
0x1A ... 0x1D	Reserved									
0x1E	DBGCTRL	7:0						FAULTDET		DBGRUN
0x1F ... 0x21	Reserved									
0x22	CAPTUREA	7:0	CAPTUREA[7:0]							
		15:8					CAPTUREA[11:8]			
0x24	CAPTUREB	7:0	CAPTUREB[7:0]							
		15:8					CAPTUREB[11:8]			
0x26 ... 0x27	Reserved									
0x28	CMPASET	7:0	CMPASET[7:0]							
		15:8					CMPASET[11:8]			
0x2A	CMPACLR	7:0	CMPACLR[7:0]							
		15:8					CMPACLR[11:8]			
0x2C	CMPBSET	7:0	CMPBSET[7:0]							
		15:8					CMPBSET[11:8]			
0x2E	CMPBCLR	7:0	CMPBCLR[7:0]							
		15:8					CMPBCLR[11:8]			

22.5 Register Description

22.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: Enable-protected

Bit	7	6	5	4	3	2	1	0
		CLKSEL[1:0]		CNTPRES[1:0]		SYNCPRES[1:0]		ENABLE
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 6:5 – CLKSEL[1:0] Clock Select
The Clock Select bits select the clock source of the TCD clock.

Value	Name	Description
0x0	20MHZ	Internal 16/20 MHz Oscillator (OSC20M)
0x1	-	Reserved
0x2	EXTCLK	External Clock
0x3	SYSCLK	System Clock

Bits 4:3 – CNTPRES[1:0] Counter Prescaler
The Counter Prescaler bits select the division factor of the TCD counter clock.

Value	Name	Description
0x0	DIV1	Division factor 1
0x1	DIV4	Division factor 4
0x2	DIV32	Division factor 32
0x3	-	Reserved

Bits 2:1 – SYNCPRES[1:0] Synchronization Prescaler
The Synchronization Prescaler bits select the division factor of the TCD clock.

Value	Name	Description
0x0	DIV1	Division factor 1
0x1	DIV2	Division factor 2
0x2	DIV4	Division factor 4
0x3	DIV8	Division factor 8

Bit 0 – ENABLE Enable
When writing to this bit, it will automatically be synchronized to the TCD clock domain.
This bit can be changed as long as the synchronization of this bit is not ongoing. See the Enable Ready (ENRDY) bit in the Status (TCDn.STATUS) register.
This bit is not enable-protected.

Value	Name	Description
0	NO	The TCD is disabled.
1	YES	The TCD is enabled and running.

22.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

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

Bits 1:0 – WGMODE[1:0] Waveform Generation Mode
These bits select the waveform generation.

Value	Name	Description
0x0	ONERAMP	One Ramp mode
0x1	TWORAMP	Two Ramp mode
0x2	FOURRAMP	Four Ramp mode
0x3	DS	Dual Slope mode

22.5.3 Control C

Name: CTRLC
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	CMPDSEL	CMPCSEL			FIFTY		AUPDATE	CMPOVR
Access	R/W	R/W			R/W		R/W	R/W
Reset	0	0			0		0	0

Bit 7 – CMPDSEL Compare D Output Select
This bit selects which waveform will be connected to output D.

Value	Name	Description
0	PWMA	Waveform A
1	PWMB	Waveform B

Bit 6 – CMPCSEL Compare C Output Select
This bit selects which waveform will be connected to output C.

Value	Name	Description
0	PWMA	Waveform A
1	PWMB	Waveform B

Bit 3 – FIFTY Fifty Percent Waveform
If the two waveforms have identical characteristics, this bit can be written to '1'. This will cause any values written to the TCDn.CMPBSET/TCDn.CLR register to also be written to the TCDn.CMPASET/TCDn.CLR register.

Bit 1 – AUPDATE Automatically Update
If this bit is written to '1', synchronization at the end of the TCD cycle is automatically requested after the Compare B Clear High (TCDn.CMPBCLRH) register is written.
If the fifty percent waveform is enabled by setting the FIFTY bit in this register, writing the Compare A Clear High register will also request a synchronization at the end of the TCD cycle if the AUPDATE bit is set.

Bit 0 – CMPOVR Compare Output Value Override
When this bit is written to '1', default values of the Waveform Outputs A and B are overridden by the values written in the Compare x Value in Active state bit fields in the Control D register. See the 22.5.4 CTRLD register description for more details.

22.5.4 Control D

Name: CTRLD
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	CMPBVAL[3:0]				CMPAVAL[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 0:3, 4:7 – CMPVAL Compare x Value (in Active state)
These bits set the logical value of the PWMx signal for the corresponding states in the TCD cycle.
These settings are valid only if the Compare Output Value Override (CMPOVR) bit in the Control C (TCDn.CTRLC) register is written to '1'.

Table 22-12. Two and Four Ramp Mode

CMPxVAL	DTA	OTA	DTB	OTB
PWMA	CMPAVAL[0]	CMPAVAL[1]	CMPAVAL[2]	CMPAVAL[3]
PWMB	CMPBVAL[0]	CMPBVAL[1]	CMPBVAL[2]	CMPBVAL[3]

When used in One Ramp mode, WOA will only use the setup for dead time A (DTA) and on time A (OTA) to set the output. WOB will only use dead time B (DTB) and on time B (OTB) values to set the output.

Table 22-13. One Ramp Mode

CMPxVAL	DTA	OTA	DTB	OTB
PWMA	CMPAVAL[1]	CMPAVAL[0]	-	-
PWMB	-	-	CMPBVAL[3]	CMPBVAL[2]

22.5.5 Control E

Name: CTRLE
Offset: 0x04
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	DISEOC			SCAPTUREB	SCAPTUREA	RESTART	SYNC	SYNCEOC
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit 7 – DISEOC Disable at End of TCD Cycle Strobe
When this bit is written to '1', the TCD will automatically disable at the end of the TCD cycle.
Note that ENRDY in TCDn.STATUS will stay low until the TCD is disabled.
Writing to this bit has effect only if there is no ongoing synchronization of the ENABLE value in TCDn.CTRLA with the TCD domain. See also the ENRDY bit in TCDn.STATUS.

Bit 4 – SCAPTUREB Software Capture B Strobe
When this bit is written to '1', a software capture to the Capture B (TCDn.CAPTUREBL/H) register is triggered as soon as synchronization to the TCD clock domain occurs.
Writing to this bit has effect only if there is no ongoing synchronization of a command. See also the CMDRDY bit in TCDn.STATUS.

Bit 3 – SCAPTUREA Software Capture A Strobe
When this bit is written to '1', a software capture to the Capture A (TCDn.CAPTUREAL/H) register is triggered as soon as synchronization to the TCD clock domain occurs.
Writing to this bit has effect only if there is no ongoing synchronization of a command. See also the CMDRDY bit in TCDn.STATUS.

Bit 2 – RESTART Restart Strobe
When this bit is written to '1', a restart of the TCD counter is executed as soon as this bit is synchronized to the TCD domain.
Writing to this bit has effect only if there is no ongoing synchronization of a command. See also the CMDRDY bit in TCDn.STATUS.

Bit 1 – SYNC Synchronize Strobe
When this bit is written to '1', the double-buffered registers will be loaded to the TCD domain as soon as this bit is synchronized to the TCD domain.
Writing to this bit has effect only if there is no ongoing synchronization of a command. See also the CMDRDY bit in TCDn.STATUS.

Bit 0 – SYNCEOC Synchronize End of TCD Cycle Strobe
When this bit is written to '1', the double-buffered registers will be loaded to the TCD domain at the end of the next TCD cycle.
Writing to this bit has effect only if there is no ongoing synchronization of a command. See also the CMDRDY bit in TCDn.STATUS.

22.5.6 Event Control A

Name: EVCTRLA
Offset: 0x08
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	CFG[1:0]			EDGE		ACTION		TRIGEI
Access	R/W	R/W		R/W		R/W		R/W
Reset	0	0		0		0		0

Bits 7:6 – CFG[1:0] Event Configuration
When the input capture noise canceler is activated (FILTERON), the event input is filtered. The filter function requires four successive equal valued samples of the trigger pin to change its output. The input capture is, therefore, delayed by four clock cycles when the noise canceler is enabled (FILTERON).
When the Asynchronous Event is enabled (ASYNCON), the event input will affect the output directly.

Value	Name	Description
0x0	NEITHER	Neither filter nor asynchronous event is enabled.
0x1	FILTERON	Input capture noise cancellation filter enabled.
0x2	ASYNCON	Asynchronous event output qualification enabled.
other	-	Reserved.

Bit 4 – EDGE Edge Selection
This bit is used to select the active edge or level for the event input.

Value	Name	Description
0	FALL_LOW	The falling edge or low level of the event input triggers a Capture or Fault action.
1	RISE_HIGH	The rising edge or high level of the event input triggers a Capture or Fault action.

Bit 2 – ACTION Event Action
This bit enables capturing on the event input. By default, the input will trigger a Fault, depending on the Input Control register's Input mode. It is also possible to trigger a capture on the event input.

Value	Name	Description
0	FAULT	Event triggers a Fault.
1	CAPTURE	Event triggers a Fault and capture.

Bit 0 – TRIGEI Trigger Event Input Enable
Writing this bit to '1' enables event as the trigger for input A.

22.5.7 Event Control B

Name: EVCTRLB
Offset: 0x09
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	CFG[1:0]			EDGE		ACTION		TRIGEI
Access	R/W	R/W		R/W		R/W		R/W
Reset	0	0		0		0		0

Bits 7:6 – CFG[1:0] Event Configuration
When the input capture noise canceler is activated (FILTERON), the event input is filtered. The filter function requires four successive equal valued samples of the trigger pin to change its output. The input capture is, therefore, delayed by four clock cycles when the noise canceler is enabled (FILTERON).
When the Asynchronous Event is enabled (ASYNCON), the event input will affect the output directly.

Value	Name	Description
0x0	NEITHER	Neither filter nor asynchronous event is enabled.
0x1	FILTERON	Input capture noise cancellation filter enabled.
0x2	ASYNCON	Asynchronous event output qualification enabled.
other	-	Reserved.

Bit 4 – EDGE Edge Selection
This bit is used to select the active edge or level for the event input.

Value	Name	Description
0	FALL_LOW	The falling edge or low level of the event input triggers a Capture or Fault action.
1	RISE_HIGH	The rising edge or high level of the event input triggers a Capture or Fault action.

Bit 2 – ACTION Event Action
This bit enables capturing on the event input. By default, the input will trigger a Fault, depending on the Input Control register's Input mode. It is also possible to trigger a capture on the event input.

Value	Name	Description
0	FAULT	Event triggers a Fault.
1	CAPTURE	Event triggers a Fault and capture.

Bit 0 – TRIGEI Trigger Event Input Enable
Writing this bit to '1' enables event as a trigger for input B.

22.5.8 Interrupt Control

Name: INTCTRL
Offset: 0x0C
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
					TRIGB	TRIGA		OVF
Access					R/W	R/W		R/W
Reset					0	0		0

Bit 3 – TRIGB Trigger B Interrupt Enable
Writing this bit to '1' enables the interrupt when trigger input B is received.

Bit 2 – TRIGA Trigger A Interrupt Enable
Writing this bit to '1' enables the interrupt when trigger input A is received.

Bit 0 – OVF Counter Overflow
Writing this bit to '1' enables the restart-of-sequence interrupt or overflow interrupt.

22.5.9 Interrupt Flags

Name: INTFLAGS
Offset: 0x0D
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
					TRIGB	TRIGA		OVF
Access					R/W	R/W		R/W
Reset					0	0		0

Bit 3 – TRIGB Trigger B Interrupt Flag
The Trigger B Interrupt (TRIGB) flag is set on a Trigger B or Capture B condition. The flag is cleared by writing a '1' to its bit location.

Bit 2 – TRIGA Trigger A Interrupt Flag
The Trigger A Interrupt (TRIGA) flag is set on a Trigger A or Capture A condition. The flag is cleared by writing a '1' to its bit location.

Bit 0 – OVF Overflow Interrupt Flag
The Overflow Flag (OVF) is set at the end of a TCD cycle. The flag is cleared by writing a '1' to its bit location.

22.5.10 Status

Name: STATUS
Offset: 0x0E
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	PWMACTB	PWMACTA					CMDRDY	ENRDY
Access	R/W	R/W					R	R
Reset	0	0					0	0

Bit 7 – PWMACTB PWM Activity on B
This bit is set by hardware each time the WOB output toggles from '0' to '1' or from '1' to '0'.
This status bit must be cleared by software by writing a '1' to it before new PWM activity can be detected.

Bit 6 – PWMACTA PWM Activity on A
This bit is set by hardware each time the WOA output toggles from '0' to '1' or from '1' to '0'.
This status bit must be cleared by software by writing a '1' to it before new PWM activity can be detected.

Bit 1 – CMDRDY Command Ready
This status bit tells when a command is synced to the TCD domain and the system is ready to receive new commands.
The following actions clear the CMDRDY bit:

1. TCDn.CTRLE SYNCEOC strobe.
2. TCDn.CTRLE SYNC strobe.
3. TCDn.CTRLE RESTART strobe.
4. TCDn.CTRLE SCAPTUREA Capture A strobe.
5. TCDn.CTRLE SCAPTUREB Capture B strobe.
6. TCDn.CTRLC AUPDATE written to '1' and writing to the TCDn.CMPBCLRH register.

Bit 0 – ENRDY Enable Ready
This status bit tells when the ENABLE value in TCDn.CTRLA is synced to the TCD domain and is ready to be written to again.
The following actions clear the ENRDY bit:

1. Writing to the ENABLE bit in TCDn.CTRLA.
2. TCDn.CTRLE DISEOC strobe.
3. Going into BREAK in an On-Chip Debugging (OCD) session while the Debug Run (DBGCTRL) bit in TCDn.DBGCTRL is '0'.

22.5.11 Input Control A

Name: INPUTCTRLA
Offset: 0x10
Reset: 0x00
Property: -

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

Bits 3:0 – INPUTMODE[3:0] Input Mode

Value	Name	Description
0x0	NONE	The input has no action.
0x1	JMPWAIT	Stop the output, jump to the opposite compare cycle, and wait.
0x2	EXECWAIT	Stop the output, execute the opposite compare cycle, and wait.
0x3	EXECFAULT	Stop the output, execute the opposite compare cycle while the Fault is active.
0x4	FREQ	Stop all outputs, maintain the frequency.
0x5	EXECDT	Stop all outputs, execute dead time while the Fault is active.
0x6	WAIT	Stop all outputs, jump to the next compare cycle, and wait.
0x7	WAITSW	Stop all outputs, wait for software action.
0x8	EDGETRIG	Stop the output on the edge, jump to the next compare cycle.
0x9	EDGETRIGFREQ	Stop the output on the edge, maintain the frequency.
0xA	LVLTRIGFREQ	Stop the output at level, maintain the frequency.

22.5.12 Input Control B

Name: INPUTCTRLB
Offset: 0x11
Reset: 0x00
Property: -

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

Bits 3:0 – INPUTMODE[3:0] Input Mode

Value	Name	Description
0x0	NONE	The input has no action.
0x1	JMPWAIT	Stop the output, jump to the opposite compare cycle, and wait.
0x2	EXECWAIT	Stop the output, execute the opposite compare cycle, and wait.
0x3	EXECFAULT	Stop the output, execute the opposite compare cycle while the Fault is active.
0x4	FREQ	Stop all outputs, maintain the frequency.
0x5	EXECDT	Stop all outputs, execute dead time while the Fault is active.
0x6	WAIT	Stop all outputs, jump to the next compare cycle, and wait.
0x7	WAITSW	Stop all outputs, wait for software action.
0x8	EDGETRIG	Stop the output on the edge, jump to the next compare cycle.
0x9	EDGETRIGFREQ	Stop the output on the edge, maintain the frequency.
0xA	LVLTRIGFREQ	Stop the output at level, maintain the frequency.

22.5.13 Fault Control

Name: FAULTCTRL
Offset: 0x12
Reset: 0x00
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
	CMPDEN	CMPCEN	CMPBEN	CMPAEN	CMPD	CMPC	CMPB	CMPA
Access	R/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 4, 5, 6, 7 – CMPEN Compare x Enable
This bit field enable compare as output on the pin. This bit field is reset to '0' after a Power-On Reset. At any other reset, the content is kept but during the reset sequence loaded from the TCD Configuration Fuse (FUSE.TCDCFG)

Bits 0, 1, 2, 3 – CMP Compare x Value
This bit field set the default state from Reset, or when an input event triggers a fault causing changes to the output. This bit field is reset to '0' after a Power-On Reset. At any other reset, the content is kept but during the reset sequence loaded from the TCD Configuration Fuse (FUSE.TCDCFG).

22.5.14 Delay Control

Name: DLYCTRL
Offset: 0x14
Reset: 0x00
Property: -

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

Bits 5:4 – DLYPRESC[1:0] Delay Prescaler
These bits control the prescaler settings for the blanking or output event delay.

Value	Name	Description
0x0	DIV1	Prescaler division factor 1
0x1	DIV2	Prescaler division factor 2
0x2	DIV4	Prescaler division factor 4
0x3	DIV8	Prescaler division factor 8

Bits 3:2 – DLYTRIG[1:0] Delay Trigger
These bits control the trigger of the blanking or output event delay.

Value	Name	Description
0x0	CMPASET	CMPASET triggers delay
0x1	CMPACLR	CMPACLR triggers delay
0x2	CMPBSET	CMPBSET triggers delay
0x3	CMPBCLR	CMPASET triggers delay (end of cycle)

Bits 1:0 – DLYSEL[1:0] Delay Select
These bits control what function must be used by the delay trigger, the blanking or output event delay.

Value	Name	Description
0x0	OFF	Delay functionality not used
0x1	INBLANK	Input blanking enabled
0x2	EVENT	Event delay enabled
0x3	-	Reserved

22.5.15 Delay Value

Name: DLYVAL
Offset: 0x15
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	DLYVAL[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 – DLYVAL[7:0] Delay Value
These bits configure the blanking/output event delay time or event output synchronization delay in a number of prescaled TCD cycles.

22.5.16 Dither Control

Name: DITCTRL
Offset: 0x18
Reset: 0x00
Property: -

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

Bits 1:0 – DITHERSEL[1:0] Dither Select
This bit field selects which state of the TCD cycle will benefit from the dither function. See the 22.3.3.5 Dithering section.

Value	Name	Description
0x0	ONTIMEB	On time ramp B
0x1	ONTIMEAB	On time ramp A and B
0x2	DEADTIMEB	Dead time ramp B
0x3	DEADTIMEAB	Dead time ramp A and B

22.5.17 Dither Value

Name: DITVAL
Offset: 0x19
Reset: 0x00
Property: -

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

Bits 3:0 – DITHER[3:0] Dither Value
These bits configure the fractional adjustment of the on time or off time, according to the Dither Selection (DITHERSEL) bits in the Dither Control (TCDn.DITCTRL) register. The DITHER value is added to a 4-bit accumulator at the end of each TCD cycle. When the accumulator overflows, the frequency adjustment will occur.
The DITHER bits are double-buffered, so the new value is copied when an update condition occurs.

22.5.18 Debug Control

Name: DBGCTRL
Offset: 0x1E
Reset: 0x00
Property: -

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

Bit 2 – FAULTDET Fault Detection
This bit defines how the peripheral behaves when stopped in Debug mode.

Value	Name	Description
0	NONE	No Fault is generated if TCD is stopped in Debug mode.
1	FAULT	A Fault is generated, and both trigger flags are set, if TCD is halted in Debug mode.

Bit 0 – DBGRUN Debug Run
When written to '1', the peripheral will continue operating in Debug mode when the CPU is halted.

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

22.5.19 Capture A

Name: CAPTUREA
Offset: 0x22
Reset: 0x00
Property: -

The TCDn.CAPTUREAL and TCDn.CAPTUREAH register pair represents the 12-bit TCDn.CAPTUREA value.

For capture operation, these registers constitute the second buffer level and access point for the CPU. The TCDn.CAPTUREA registers are updated with the buffer value when an update condition occurs. The CAPTURE A register contains the TCD counter value when a trigger A or software capture A occurs.

The TCD counter value is synchronized to CAPTUREA by either software or an event.

The capture register is blocked for an update of new capture data until the higher byte of this register is read.

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

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

Bits 11:0 – CAPTUREA[11:0] Capture A Byte

22.5.20 Capture B

Name: CAPTUREB
Offset: 0x24
Reset: 0x00
Property: -

The TCDn.CAPTUREBL and TCDn.CAPTUREBH register pair represents the 12-bit TCDn.CAPTUREB value.

For capture operation, these registers constitute the second buffer level and access point for the CPU. The TCDn.CAPTUREB registers are updated with the buffer value when an update condition occurs. The CAPTURE B register contains the TCD counter value when a trigger B or software capture B occurs.

The TCD counter value is synchronized to CAPTUREB by either software or an event.

The capture register is blocked for an update of new capture data until the higher byte of this register is read.

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

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

Bits 11:0 – CAPTUREB[11:0] Capture B Byte

22.5.21 Compare Set A

Name: CMPASET
Offset: 0x28
Reset: 0x00
Property: -

The TCDn.CMPASETL and TCDn.CMPASETH register pair represents the 12-bit TCDn.CMPASET value. This register is continuously compared to the counter value. Then, the outputs from the comparators are used for generating waveforms.

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

Bit	7	6	5	4	3	2	1	0
	CMPASET[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 11:0 – CMPASET[11:0] Compare A Set
These bits hold the value of the compare register.

22.5.22 Compare Set B

Name: CMPBSET
Offset: 0x2C
Reset: 0x00
Property: -

The TCDn.CMPBSETL and TCDn.CMPBSETH register pair represents the 12-bit TCDn.CMPBSET value. This register is continuously compared to the counter value. Then, the outputs from the comparators are used for generating waveforms.

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

Bit	7	6	5	4	3	2	1	0
	CMPBSET[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 11:0 – CMPBSET[11:0] Compare B Set
These bits hold the value of the compare register.

22.5.23 Compare Clear A

Name: CMPACLR
Offset: 0x2A
Reset: 0x00
Property: -

The TCDn.CMPACLRL and TCDn.CMPACLRH register pair represents the 12-bit TCDn.CMPACLR value. This register is continuously compared to the counter value. Then, the outputs from the comparators are used for generating waveforms.

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

Bit	7	6	5	4	3	2	1	0
	CMPACLR[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 11:0 – CMPACLR[11:0] Compare A Clear
These bits hold the value of the compare register.

22.5.24 Compare Clear B

Name: CMPBCLR
Offset: 0x2E
Reset: 0x00
Property: -

The TCDn.CMPBCLRL and TCDn.CMPBCLRH register pair represents the 12-bit TCDn.CMPBCLR value. This register is continuously compared to the counter value. Then, the outputs from the comparators are used for generating waveforms.

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

Bit	7	6	5	4	3	2	1	0
	CMPBCLR[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 11:0 – CMPBCLR[11:0] Compare B Clear
These bits hold the value of the compare register.

23. RTC - Real-Time Counter

23.1 Features

- 16-bit Resolution
- Selectable Clock Sources
- Programmable 15-bit Clock Prescaling
- One Compare Register
- One Period Register
- Clear Timer on Period Overflow
- Optional Interrupt/Event on Overflow and Compare Match
- Periodic Interrupt and Event

23.2 Overview

The RTC peripheral offers two timing functions: the Real-Time Counter (RTC) and a Periodic Interrupt Timer (PIT).

The PIT functionality can be enabled independently of the RTC functionality.

RTC - Real-Time Counter

The RTC counts (prescaled) clock cycles in a Counter register and compares the content of the Counter register to a Period register and a Compare register.

The RTC can generate both interrupts and events on compare match or overflow. It will generate a compare interrupt and/or event at the first count after the counter equals the Compare register value, and an overflow interrupt and/or event at the first count after the counter value equals the Period register value. The overflow will reset the counter value to zero.

The RTC peripheral typically runs continuously, including in Low-Power Sleep modes, to keep track of time. It can wake up the device from sleep modes and/or interrupt the device at regular intervals.

The reference clock is typically the 32.768 kHz output from an external crystal. The RTC can also be clocked from an external clock signal, the 32.768 kHz Internal Ultra Low-Power Oscillator (OSCULP32K), or the OSCULP32K divided by 32.

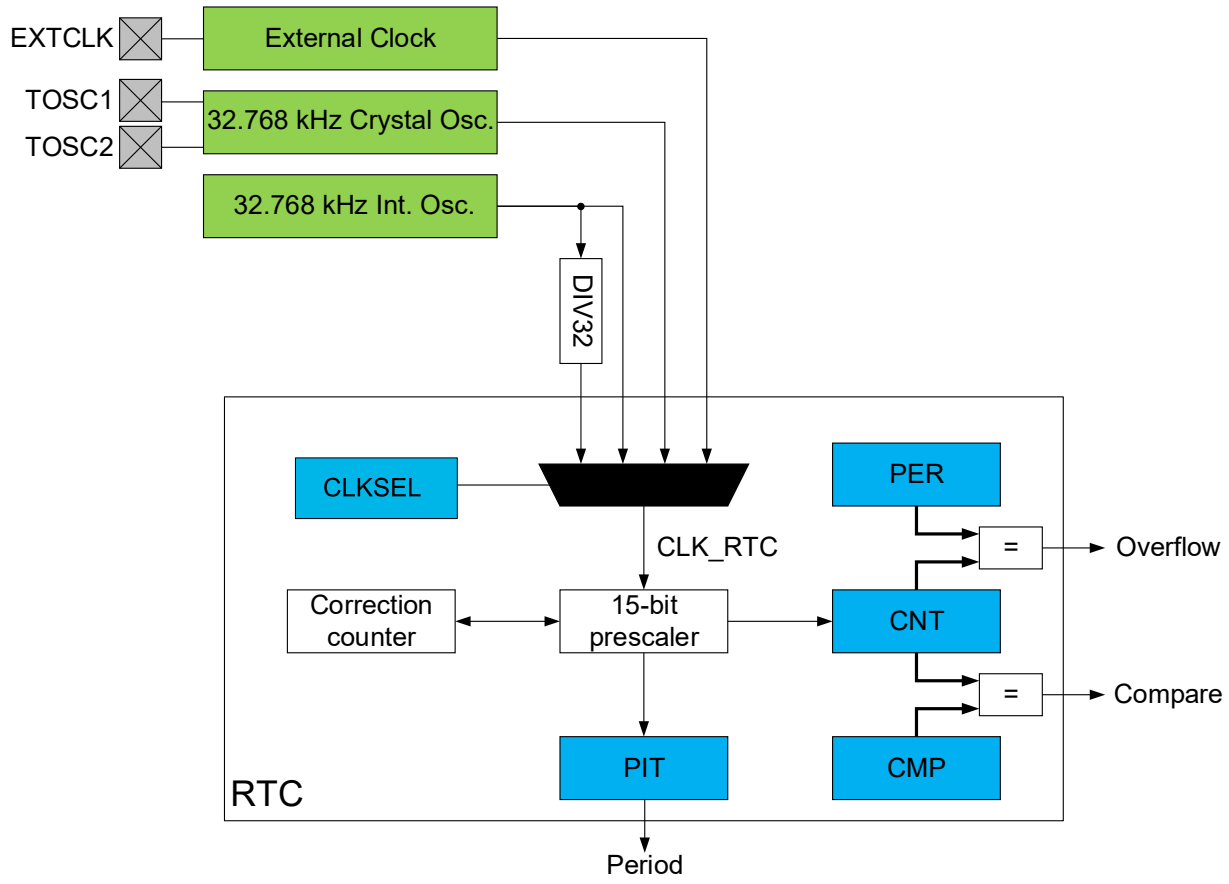
The RTC peripheral includes a 15-bit programmable prescaler that can scale down the reference clock before it reaches the counter. A wide range of resolutions and time-out periods can be configured for the RTC. With a 32.768 kHz clock source, the maximum resolution is 30.5 µs, and time-out periods can be up to two seconds. With a resolution of 1s, the maximum time-out period is more than 18 hours (65536 seconds).

PIT - Periodic Interrupt Timer

The PIT uses the same clock source (CLK_RTC) as the RTC function and can generate an interrupt request or a level event on every n^{th} clock period. The n can be selected from {4, 8, 16,... 32768} for interrupts and from {64, 128, 256,... 8192} for events.

23.2.1 Block Diagram

Figure 23-1. RTC Block Diagram



23.3 Clocks

The peripheral clock (CLK_PER) is required to be at least four times faster than the RTC clock (CLK_RTC) for reading the counter value, regardless of the prescaler setting.

A 32.768 kHz crystal can be connected to the TOSC1 or TOSC2 pins, along with any required load capacitors. Alternatively, an external digital clock can be connected to the TOSC1 pin.

23.4 RTC Functional Description

The RTC peripheral offers two timing functions: the Real-Time Counter (RTC) and a Periodic Interrupt Timer (PIT). This subsection describes the RTC.

23.4.1 Initialization

Before enabling the RTC peripheral and the desired actions (interrupt requests and output events), the source clock for the RTC counter must be configured to operate the RTC.

23.4.1.1 Configure the Clock CLK_RTC

To configure the CLK_RTC, follow these steps:

1. Configure the desired oscillator to operate as required, in the Clock Controller (CLKCTRL) peripheral.
2. Write the Clock Select (CLKSEL) bit field in the Clock Selection (RTC.CLKSEL) register accordingly.

The CLK_RTC clock configuration is used by both RTC and PIT functionality.

23.4.1.2 Configure RTC

To operate the RTC, follow these steps:

1. Set the compare value in the Compare (RTC.CMP) register, and/or the overflow value in the Period (RTC.PER) register.
2. Enable the desired interrupts by writing to the respective interrupt enable bits (CMP, OVF) in the Interrupt Control (RTC.INTCTRL) register.
3. Configure the RTC internal prescaler by writing the desired value to the Prescaler (PRESCALER) bit field in the Control A (RTC.CTRLA) register.
4. Enable the RTC by writing a '1' to the RTC Peripheral Enable (RTCEN) bit in the RTC.CTRLA register.

Note: The RTC peripheral is used internally during device start-up. Always check the Synchronization Busy bits in the Status (RTC.STATUS) and Periodic Interrupt Timer Status (RTC.PITSTATUS) registers, and on the initial configuration.

23.4.2 Operation - RTC

23.4.2.1 Enabling and Disabling

The RTC is enabled by writing the RTC Peripheral Enable (RTCEN) bit in the Control A (RTC.CTRLA) register to '1'. The RTC is disabled by writing the RTC Peripheral Enable (RTCEN) bit in RTC.CTRLA to '0'.

23.5 PIT Functional Description

The RTC peripheral offers two timing functions: the Real-Time Counter (RTC) and a Periodic Interrupt Timer (PIT). This subsection describes the PIT.

23.5.1 Initialization

To operate the PIT, follow these steps:

1. Configure the RTC clock CLK_RTC as described in section 23.4.1.1 Configure the Clock CLK_RTC.
2. Enable the interrupt by writing a '1' to the Periodic Interrupt (PI) bit in the PIT Interrupt Control (RTC.PITINTCTRL) register.
3. Select the period for the interrupt by writing the desired value to the Period (PERIOD) bit field in the Periodic Interrupt Timer Control A (RTC.PITCTRLA) register.
4. Enable the PIT by writing a '1' to the Periodic Interrupt Timer Enable (PITEN) bit in the RTC.PITCTRLA register.

Note: The RTC peripheral is used internally during device start-up. Always check the Synchronization Busy bits in the RTC.STATUS and RTC.PITSTATUS registers, and on the initial configuration.

23.5.2 Operation - PIT

23.5.2.1 Enabling and Disabling

The PIT is enabled by writing the Periodic Interrupt Timer Enable (PITEN) bit in the Periodic Interrupt Timer Control A (RTC.PITCTRLA) register to '1'. The PIT is disabled by writing the Periodic Interrupt Timer Enable (PITEN) bit in RTC.PITCTRLA to '0'.

23.5.2.2 PIT Interrupt Timing

Timing of the First Interrupt

The PIT function and the RTC function are running from the same counter inside the prescaler and can be configured as described below:

- The RTC interrupt period is configured by writing the Period (RTC.PER) register
- The PIT interrupt period is configured by writing the Period (PERIOD) bit field in Periodic Interrupt Timer Control A (RTC.PITCTRLA) register

The prescaler is OFF when both functions are OFF (RTC Peripheral Enable (RTCEN) bit in RTC.CTRLA and the Periodic Interrupt Timer Enable (PITEN) bit in RTC.PITCTRLA are '0'), but it is running (that is, its internal counter is counting) when either function is enabled. For this reason, the timing of the first PIT interrupt and the first RTC count tick will be unknown (anytime between enabling and a full period).

Continuous Operation

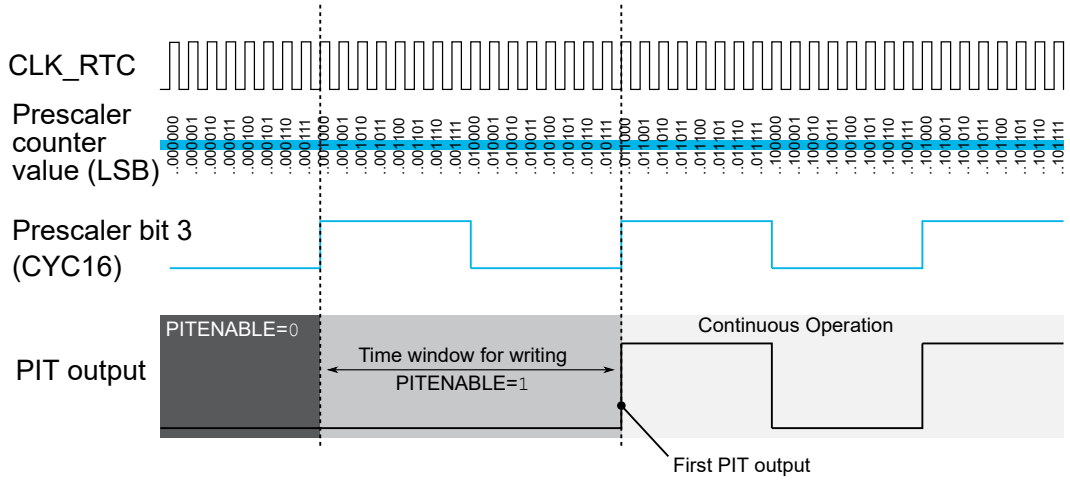
After the first interrupt, the PIT will continue toggling every ½ PIT period resulting in a full PIT period signal.

Example 23-1. PIT Timing Diagram for PERIOD=CYC16

For PERIOD=CYC16 in RTC.PITCTRLA, the PIT output effectively follows the state of the prescaler counter bit 3, so the resulting interrupt output has a period of 16 CLK_RTC cycles.

The time between writing PITEN to '1' and the first PIT interrupt can vary between virtually zero and a full PIT period of 16 CLK_RTC cycles. The precise delay between enabling the PIT and its first output depends on the prescaler's counting phase: the first interrupt shown below is produced by writing PITEN to '1' at any time inside the leading time window.

Figure 23-2. Timing Between PIT Enable and First Interrupt



23.6 Events

The RTC can generate the events described in the following table:

Table 23-1. RTC Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of the Event
Module	Event				
RTC	OVF	Overflow	Pulse	CLK_RTC	One CLK_RTC period
	CMP	Compare Match			One CLK_RTC period
	PIT_DIV8192	Prescaled RTC clock divided by 8192	Level		Given by prescaled RTC clock divided by 8192
	PIT_DIV4096	Prescaled RTC clock divided by 4096			Given by prescaled RTC clock divided by 4096
	PIT_DIV2048	Prescaled RTC clock divided by 2048			Given by prescaled RTC clock divided by 2048
	PIT_DIV1024	Prescaled RTC clock divided by 1024			Given by prescaled RTC clock divided by 1024
	PIT_DIV512	Prescaled RTC clock divided by 512			Given by prescaled RTC clock divided by 512
	PIT_DIV256	Prescaled RTC clock divided by 256			Given by prescaled RTC clock divided by 256
	PIT_DIV128	Prescaled RTC clock divided by 128			Given by prescaled RTC clock divided by 128
	PIT_DIV64	Prescaled RTC clock divided by 64			Given by prescaled RTC clock divided by 64

The conditions for generating the OVF and CMP events are identical to those that will raise the corresponding interrupt flags in the RTC.INTFLAGS register.

Refer to the *(EVSYS) Event System* section for more details regarding event users and Event System configuration.

23.7 Interrupts

Table 23-2. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
RTC	Real-Time Counter overflow and compare match interrupt	• Overflow (OVF): The counter has reached the value from the RTC.PER register and wrapped to zero. • Compare (CMP): Match between the value from the Counter (RTC.CNT) register and the value from the Compare (RTC.CMP) register.
PIT	Periodic Interrupt Timer interrupt	A time period has passed, as configured by the PERIOD bit field in RTC.PITCTRLA.

When an interrupt condition occurs, the corresponding interrupt flag is set in the peripheral's Interrupt Flags (*peripheral*.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

Note that:

- The RTC has two INTFLAGS registers: RTC.INTFLAGS and RTC.PITINTFLAGS.
- The RTC has two INTCTRL registers: RTC.INTCTRL and RTC.PITINTCTRL.

23.8 Sleep Mode Operation

The RTC will continue to operate in Idle Sleep mode. It will run in Standby Sleep mode if the Run in Standby (RUNSTDBY) bit in RTC.CTRLA is set.

The PIT will continue to operate in any sleep mode.

23.9 Synchronization

Both the RTC and the PIT are asynchronous, operating from a different clock source (CLK_RTC) independently of the peripheral clock (CLK_PER). For Control and Count register updates, it will take some RTC and/or peripheral clock cycles before an updated register value is available in a register or until a configuration change affects the RTC or PIT, respectively. This synchronization time is described for each register in the *Register Description* section.

For some RTC registers, a Synchronization Busy flag is available (CMPBUSY, PERBUSY, CNTBUSY, CTRLABUSY) in the Status (RTC.STATUS) register.

For the RTC.PITCTRLA register, a Synchronization Busy flag is available (CTRLBUSY) in the Periodic Interrupt Timer Status (RTC.PITSTATUS) register.

Check these flags before writing to the mentioned registers.

23.10 Debug Operation

If the Debug Run (DBGRUN) bit in the Debug Control (RTC.DBGCTRL) register is '1', the RTC will continue normal operation. If DBGRUN is '0' and the CPU is halted, the RTC will halt the operation and ignore any incoming events.

If the Debug Run (DBGRUN) bit in the Periodic Interrupt Timer Debug Control (RTC.PITDBGCTRL) register is '1', the PIT will continue normal operation. If DBGRUN is '0' in the Debug mode and the CPU is halted, the PIT output will be low. When the PIT output is high at the time, a new positive edge occurs to set the interrupt flag when restarting from a break. The result is an additional PIT interrupt that would not happen during normal operation. If the PIT output is low at the break, the PIT will resume low without additional interrupt.

23.11 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY	PRESCALER[3:0]						RTCEN
0x01	STATUS	7:0					CMPBUSY	PERBUSY	CNTBUSY	CTRLABUSY
0x02	INTCTRL	7:0							CMP	OVF
0x03	INTFLAGS	7:0							CMP	OVF
0x04	TEMP	7:0	TEMP[7:0]							
0x05	DBGCTRL	7:0								DBGRUN
0x06	Reserved									
0x07	CLKSEL	7:0							CLKSEL[1:0]	
0x08	CNT	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
0x0A	PER	7:0	PER[7:0]							
		15:8	PER[15:8]							
0x0C	CMP	7:0	CMP[7:0]							
		15:8	CMP[15:8]							
0x0E ... 0x0F	Reserved									
0x10	PITCTRLA	7:0		PERIOD[3:0]						PITEN
0x11	PITSTATUS	7:0								CTRLBUSY
0x12	PITINTCTRL	7:0								PI
0x13	PITINTFLAGS	7:0								PI
0x14	Reserved									
0x15	PITDBGCTRL	7:0								DBGRUN

23.12 Register Description

23.12.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

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

Bit 7 – RUNSTDBY Run in Standby

Value	Description
0	RTC disabled in Standby Sleep mode
1	RTC enabled in Standby Sleep mode

Bits 6:3 – PRESCALER[3:0] Prescaler
These bits define the prescaling of the CLK_RTC clock signal. Due to synchronization between the RTC clock and the peripheral clock, there is a latency of two RTC clock cycles from updating the register until this has an effect. Application software needs to check that the CTRLABUSY flag in RTC.STATUS register is cleared before writing to this register.

Value	Name	Description
0x0	DIV1	RTC clock/1 (no prescaling)
0x1	DIV2	RTC clock/2
0x2	DIV4	RTC clock/4
0x3	DIV8	RTC clock/8
0x4	DIV16	RTC clock/16
0x5	DIV32	RTC clock/32
0x6	DIV64	RTC clock/64
0x7	DIV128	RTC clock/128
0x8	DIV256	RTC clock/256
0x9	DIV512	RTC clock/512
0xA	DIV1024	RTC clock/1024
0xB	DIV2048	RTC clock/2048
0xC	DIV4096	RTC clock/4096
0xD	DIV8192	RTC clock/8192
0xE	DIV16384	RTC clock/16384
0xF	DIV32768	RTC clock/32768

Bit 0 – RTCEN RTC Peripheral Enable

Value	Description
0	RTC peripheral is disabled
1	RTC peripheral is enabled

23.12.2 Status

Name: STATUS
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
					CMPBUSY	PERBUSY	CNTBUSY	CTRLABUSY
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – CMPBUSY Compare Synchronization Busy
This bit is '1' when the RTC is busy synchronizing the Compare (RTC.CMP) register in the RTC clock domain.

Bit 2 – PERBUSY Period Synchronization Busy
This bit is '1' when the RTC is busy synchronizing the Period (RTC.PER) register in the RTC clock domain.

Bit 1 – CNTBUSY Counter Synchronization Busy
This bit is '1' when the RTC is busy synchronizing the Count (RTC.CNT) register in the RTC clock domain.

Bit 0 – CTRLABUSY Control A Synchronization Busy
This bit is '1' when the RTC is busy synchronizing the Control A (RTC.CTRLA) register in the RTC clock domain.

23.12.3 Interrupt Control

Name: INTCTRL
Offset: 0x02
Reset: 0x00
Property: -

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

Bit 1 – CMP Compare Match Interrupt Enable
Enable interrupt-on-compare match (that is, when the value from the Count (RTC.CNT) register matches the value from the Compare (RTC.CMP) register).

Value	Description
0	The compare match interrupt is disabled
1	The compare match interrupt is enabled

Bit 0 – OVF Overflow Interrupt Enable
Enable interrupt-on-counter overflow (that is, when the value from the Count (RTC.CNT) register matched the value from the Period (RTC.PER) register and wraps around to zero).

Value	Description
0	The overflow interrupt is disabled
1	The overflow interrupt is enabled

23.12.4 Interrupt Flag

Name: INTFLAGS
Offset: 0x03
Reset: 0x00
Property: -

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

Bit 1 – CMP Compare Match Interrupt Flag
This flag is set when the value from the Count (RTC.CNT) register matches the value from the Compare (RTC.CMP) register.
Writing a '1' to this bit clears the flag.

Bit 0 – OVF Overflow Interrupt Flag
This flag is set when the value from the Count (RTC.CNT) register has reached the value from the Period (RTC.PER) register and wrapped to zero.
Writing a '1' to this bit clears the flag.

23.12.5 Temporary

Name: TEMP
Offset: 0x4
Reset: 0x00
Property: -

The Temporary register is used by the CPU for 16-bit single-cycle access to the 16-bit registers of this peripheral. The register is common for all the 16-bit registers of this peripheral and can be read and written by software. For more details on reading and writing 16-bit registers, refer to *Accessing 16-Bit Registers* in the *AVR CPU* section.

Bit	7	6	5	4	3	2	1	0
	TEMP[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 – TEMP[7:0] Temporary
Temporary register for read/write operations in 16-bit registers.

23.12.6 Debug Control

Name: DBGCTRL
Offset: 0x05
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

23.12.7 Clock Selection

Name: CLKSEL
Offset: 0x07
Reset: 0x00
Property: -

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

Bits 1:0 – CLKSEL[1:0] Clock Select
Writing these bits select the source for the RTC clock (CLK_RTC).

Value	Name	Description
0x0	INT32K	32.768 kHz from OSCULP32K
0x1	INT1K	1.024 kHz from OSCULP32K
0x2	TOSC32K	32.768 kHz from XOSC32K or external clock from TOSC1
0x3	EXTCLK	External clock from EXTCLK pin

23.12.8 Count

Name: CNT
Offset: 0x08
Reset: 0x0000
Property: -

The RTC.CNTL and RTC.CNTH register pair represents the 16-bit value, RTC.CNT. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

Due to the synchronization between the RTC clock and main clock domains, there is a latency of two RTC clock cycles from updating the register until this has an effect. The application software needs to check that the CNTBUSY flag in RTC.STATUS is cleared before writing to this register.

Bit	15	14	13	12	11	10	9	8
	CNT[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
	CNT[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 – CNT[15:8] Counter High Byte
These bits hold the MSB of the 16-bit Counter register.

Bits 7:0 – CNT[7:0] Counter Low Byte
These bits hold the LSB of the 16-bit Counter register.

23.12.9 Period

Name: PER
Offset: 0x0A
Reset: 0xFFFF
Property: -

The RTC.PERL and RTC.PERH register pair represents the 16-bit value, RTC.PER. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

Due to the synchronization between the RTC clock and main clock domains, there is a latency of two RTC clock cycles from updating the register until this has an effect. The application software needs to check that the PERBUSY flag in RTC.STATUS is cleared before writing to this register.

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 15:8 – PER[15:8] Period High Byte
These bits hold the MSB of the 16-bit Period register.

Bits 7:0 – PER[7:0] Period Low Byte
These bits hold the LSB of the 16-bit Period register.

23.12.10 Compare

Name: CMP
Offset: 0x0C
Reset: 0x0000
Property: -

The RTC.CMPL and RTC.CMPH register pair represents the 16-bit value, RTC.CMP. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

Bit	15	14	13	12	11	10	9	8
	CMP[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
	CMP[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 – CMP[15:8] Compare High Byte
These bits hold the MSB of the 16-bit Compare register.

Bits 7:0 – CMP[7:0] Compare Low Byte
These bits hold the LSB of the 16-bit Compare register.

23.12.11 Periodic Interrupt Timer Control A

Name: PITCTRLA
Offset: 0x10
Reset: 0x00
Property: -

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

Bits 6:3 – PERIOD[3:0] Period
Writing this bit field selects the number of RTC clock cycles between each interrupt.

Value	Name	Description
0x0	OFF	No interrupt
0x1	CYC4	4 cycles
0x2	CYC8	8 cycles
0x3	CYC16	16 cycles
0x4	CYC32	32 cycles
0x5	CYC64	64 cycles
0x6	CYC128	128 cycles
0x7	CYC256	256 cycles
0x8	CYC512	512 cycles
0x9	CYC1024	1024 cycles
0xA	CYC2048	2048 cycles
0xB	CYC4096	4096 cycles
0xC	CYC8192	8192 cycles
0xD	CYC16384	16384 cycles
0xE	CYC32768	32768 cycles
0xF	-	Reserved

Bit 0 – PITEN Periodic Interrupt Timer Enable

Value	Description
0	Periodic Interrupt Timer disabled
1	Periodic Interrupt Timer enabled

23.12.12 Periodic Interrupt Timer Status

Name: PITSTATUS
Offset: 0x11
Reset: 0x00
Property: -

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

Bit 0 – CTRLBUSY PITCTRLA Synchronization Busy
This bit is '1' when the RTC is busy synchronizing the Periodic Interrupt Timer Control A (RTC.PITCTRLA) register in the RTC clock domain.

23.12.13 PIT Interrupt Control

Name: PITINTCTRL
Offset: 0x12
Reset: 0x00
Property: -

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

Bit 0 – PI Periodic Interrupt

Value	Description
0	The periodic interrupt is disabled
1	The periodic interrupt is enabled

23.12.14 PIT Interrupt Flag

Name: PITINTFLAGS
Offset: 0x13
Reset: 0x00
Property: -

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

Bit 0 – PI Periodic Interrupt Flag
This flag is set when a periodic interrupt is issued.
Writing a '1' clears the flag.

23.12.15 Periodic Interrupt Timer Debug Control

Name: PITDBGCTRL
Offset: 0x15
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

24. USART - Universal Synchronous and Asynchronous Receiver and Transmitter

24.1 Features

- Full-Duplex Operation
- Half-Duplex Operation:
  - One-Wire mode
  - RS-485 mode
- Asynchronous or Synchronous Operation
- Supports Serial Frames with Five, Six, Seven, Eight or Nine Data Bits and One or Two Stop Bits
- Fractional Baud Rate Generator:
  - Can generate the desired baud rate from any peripheral clock frequency
  - No need for an external oscillator
- Built-In Error Detection and Correction Schemes:
  - Odd or even parity generation and parity check
  - Buffer overflow and frame error detection
  - Noise filtering including false Start bit detection and digital low-pass filter
- Separate Interrupts for:
  - Transmit complete
  - Transmit Data register empty
  - Receive complete
- Master SPI Mode
- Multiprocessor Communication Mode
- Start-of-Frame Detection
- IRCOM Module for IrDA® Compliant Pulse Modulation/Demodulation
- LIN Slave Support

24.2 Overview

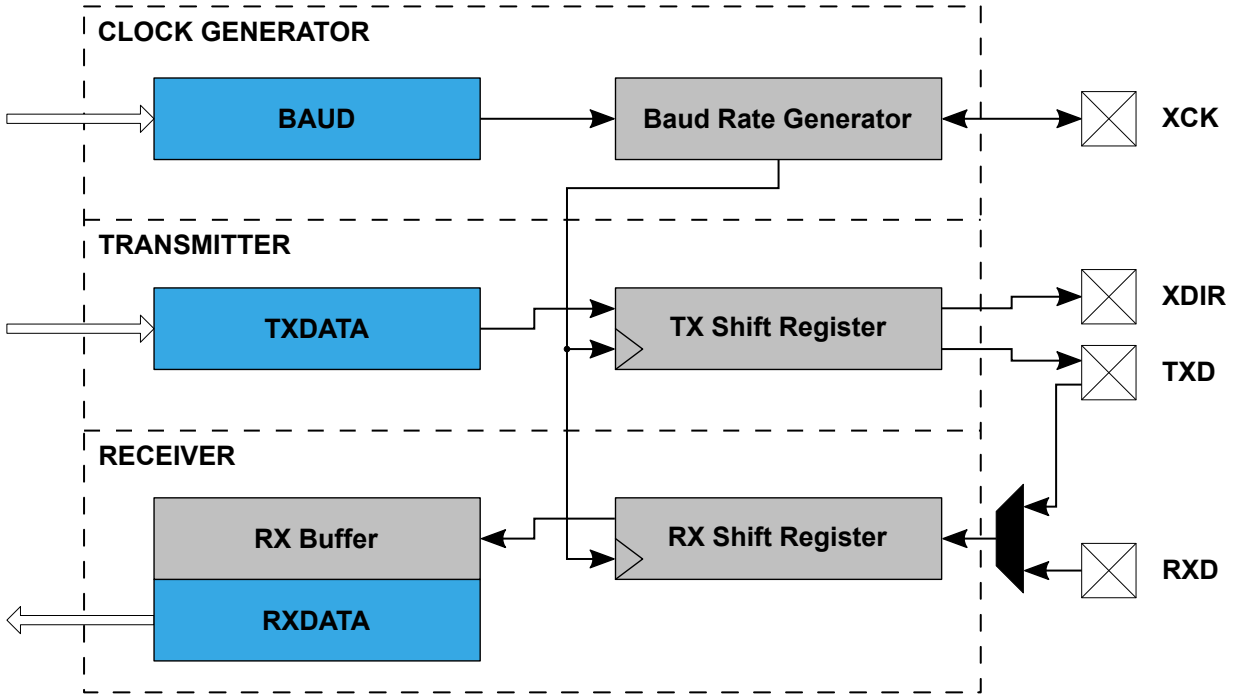
The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a fast and flexible serial communication peripheral. The USART supports a number of different modes of operation that can accommodate multiple types of applications and communication devices. For example, the One-Wire Half-Duplex mode is useful when low pin count applications are desired. The communication is frame-based, and the frame format can be customized to support a wide range of standards.

The USART is buffered in both directions, enabling continued data transmission without any delay between frames. Separate interrupts for receive and transmit completion allow fully interrupt-driven communication.

The transmitter consists of a single-write buffer, a Shift register, and control logic for different frame formats. The receiver consists of a two-level receive buffer and a Shift register. The status information of the received data is available for error checking. Data and clock recovery units ensure robust synchronization and noise filtering during asynchronous data reception.

24.2.1 Block Diagram

Figure 24-1. USART Block Diagram



24.2.2 Signal Description

Signal	Type	Description
XCK	Output/input	Clock for synchronous operation
XDIR	Output	Transmit enable for RS-485
TxD	Output/input	Transmitting line (and receiving line in One-Wire mode)
RxD	Input	Receiving line

24.3 Functional Description

24.3.1 Initialization

Full Duplex Mode:

1. Set the baud rate (USARTn.BAUD).
2. Set the frame format and mode of operation (USARTn.CTRLC).
3. Configure the TXD pin as an output.
4. Enable the transmitter and the receiver (USARTn.CTRLB).

Note:

- For interrupt-driven USART operation, global interrupts must be disabled during the initialization
- Before doing a reinitialization with a changed baud rate or frame format, be sure that there are no ongoing transmissions while the registers are changed

One-Wire Half Duplex Mode:

1. Internally connect the TXD to the USART receiver (the LBME bit in the USARTn.CTRLA register).
2. Enable internal pull-up for the RX/TX pin (the PULLUPEN bit in the PORTx.PINnCTRL register).

3. Enable Open-Drain mode (the ODME bit in the USARTn.CTRLB register).
4. Set the baud rate (USARTn.BAUD).
5. Set the frame format and mode of operation (USARTn.CTRLC).
6. Enable the transmitter and the receiver (USARTn.CTRLB).

Note:

- When Open-Drain mode is enabled, the TXD pin is automatically set to output by hardware
- For interrupt-driven USART operation, global interrupts must be disabled during the initialization
- Before doing a reinitialization with a changed baud rate or frame format, be sure that there are no ongoing transmissions while the registers are changed

24.3.2 Operation

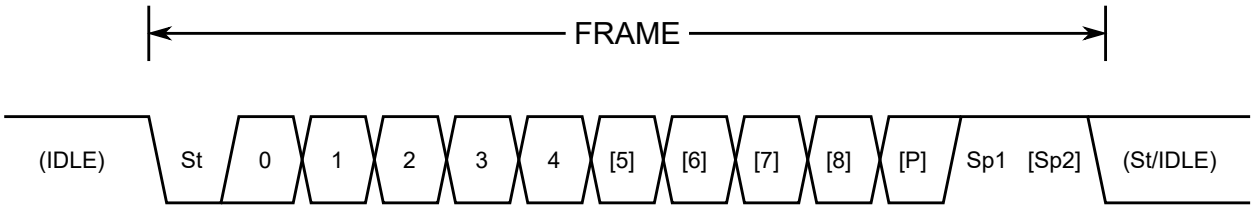
24.3.2.1 Frame Formats

The USART data transfer is frame-based. A frame starts with a 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, either the next frame can follow immediately, or the communication line can return to the Idle (high) state. The USART accepts all combinations of the following as valid frame formats:

- 1 Start bit
- 5, 6, 7, 8, or 9 data bits
- No, even, or odd Parity bit
- 1 or 2 Stop bits

The figure below illustrates the possible combinations of frame formats. Bits inside brackets are optional.

Figure 24-2. Frame Formats



St Start bit, always low

(n) Data bits (0 to 8)

P Parity bit, may be odd or even

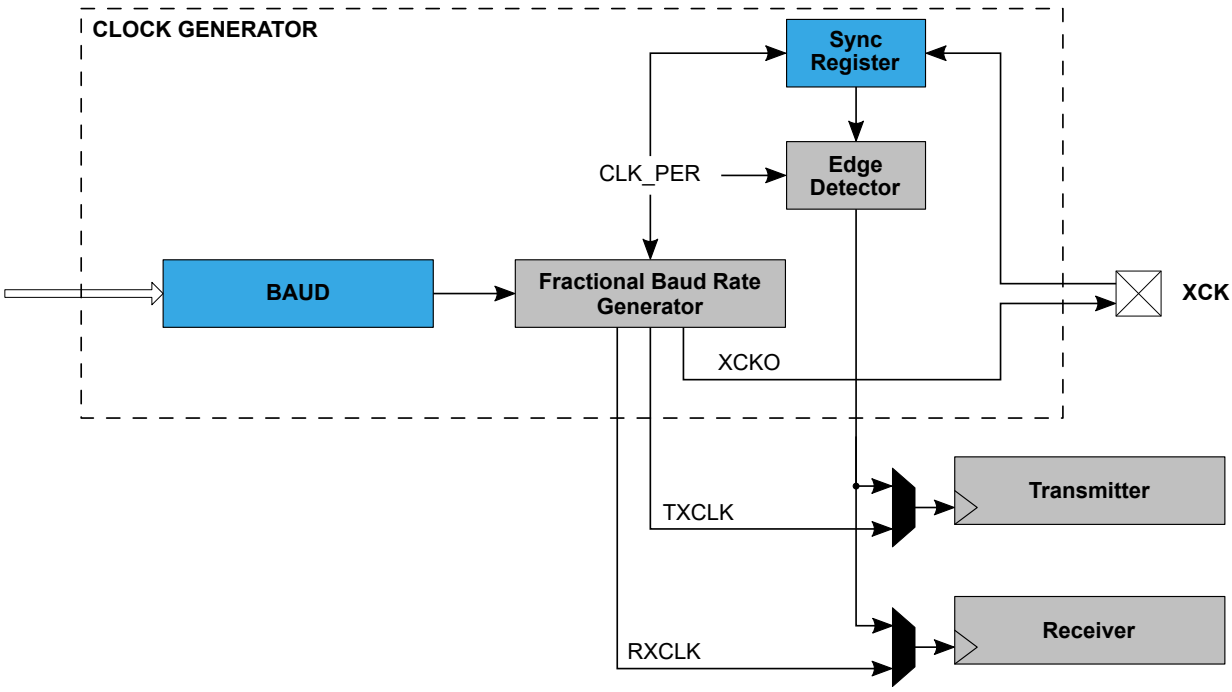
Sp Stop bit, always high

IDLE No transfer on the communication line (RxD or TxD). The Idle state is always high.

24.3.2.2 Clock Generation

The clock used for shifting and sampling data bits is generated internally by the fractional baud rate generator or externally from the Transfer Clock (XCK) pin.

Figure 24-3. Clock Generation Logic Block Diagram



24.3.2.2.1 The Fractional Baud Rate Generator

In modes where the USART is not using the XCK input as a clock source, the fractional Baud Rate Generator is used to generate the clock. Baud rate is given in terms of bits per second (bps) and is configured by writing the USARTn.BAUD register. The baud rate (f_{BAUD}) is generated by dividing the peripheral clock (f_{CLK_PER}) by a division factor decided by the BAUD register.

The fractional Baud Rate Generator features hardware that accommodates cases where f_{CLK_PER} is not divisible by f_{BAUD}. Usually, this situation would lead to a rounding error. The fractional Baud Rate Generator expects the BAUD register to contain the desired division factor left shifted by six bits, as implemented by the equations in Table 24-1. The six LSbs will then hold the fractional part of the desired divisor. The fractional part of the BAUD register is used to dynamically adjust f_{BAUD} to achieve a closer approximation to the desired baud rate.

Since the baud rate cannot be higher than f_{CLK_PER}, the integer part of the BAUD register needs to be at least 1. Since the result is left shifted by six bits, the corresponding minimum value of the BAUD register is 64. The valid range is, therefore, 64 to 65535.

In Synchronous mode, only the 10-bit integer part of the BAUD register (BAUD[15:6]) determines the baud rate, and the fractional part (BAUD[5:0]) must, therefore, be written to zero.

The table below lists equations for translating baud rates into input values for the BAUD register. The equations take fractional interpretation into consideration, so the BAUD values calculated with these equations can be written directly to USARTn.BAUD without any additional scaling.

Table 24-1. Equations for Calculating Baud Rate Register Setting

Operating Mode	Conditions	Baud Rate (Bits Per Seconds)	USART.BAUD Register Value Calculation
Asynchronous	$f_{BAUD} \leq \frac{f_{CLK\_PER}}{S}$ $USART.BAUD \geq 64$	$f_{BAUD} = \frac{64 \times f_{CLK\_PER}}{S \times BAUD}$	$BAUD = \frac{64 \times f_{CLK\_PER}}{S \times f_{BAUD}}$
Synchronous Master	$f_{BAUD} \leq \frac{f_{CLK\_PER}}{S}$ $USART.BAUD \geq 64$	$f_{BAUD} = \frac{f_{CLK\_PER}}{S \times BAUD[15:6]}$	$BAUD[15:6] = \frac{f_{CLK\_PER}}{S \times f_{BAUD}}$

S is the number of samples per bit

- Asynchronous Normal mode: S = 16
- Asynchronous Double-Speed mode: S = 8
- Synchronous mode: S = 2

24.3.2.3 Data Transmission

The USART transmitter sends data by periodically driving the transmission line low. The data transmission is initiated by loading the transmit buffer (USARTn.TXDATA) with the data to be sent. The data in the transmit buffer is moved to the Shift register once it 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 frame in the Shift register has been shifted out, and there are no new data present in the transmit buffer, the Transmit Complete Interrupt Flag (the TXCIF bit in the USARTn.STATUS register) is set, and the interrupt is generated if it is enabled.

TXDATA can only be written when the Data Register Empty Interrupt Flag (the DREIF bit in the USARTn.STATUS register) is set, indicating that the register is empty and ready for new data.

When using frames with fewer than eight bits, the Most Significant bits (MSb) written to TXDATA are ignored. If 9-bit characters are used, the DATA[8] bit in the USARTn.TXDATAH register has to be written before the DATA[7:0] bits in the USARTn.TXDATAL register.

24.3.2.3.1 Disabling the Transmitter

When disabling the transmitter, the operation will not become effective until ongoing and pending transmissions are completed (that is, when the Transmit Shift register and Transmit Buffer register do not contain data to be transmitted). When the transmitter is disabled, it will no longer override the TXD pin, and the PORT module regains control of the pin. The pin is automatically configured as an input by hardware regardless of its previous setting. The pin can now be used as a normal I/O pin with no port override from the USART.

24.3.2.4 Data Reception

The USART receiver samples the reception line to detect and interpret the received data. The direction of the pin must, therefore, be configured as an input by writing a '0' to the corresponding bit in the Direction register (PORTx.DIRn).

The receiver accepts data when a valid Start bit is detected. Each bit that follows the Start bit will be sampled at the baud rate or XCK clock and shifted into the Receive Shift register until the first Stop bit of a frame is received. A 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 receive buffer. The Receive Complete Interrupt Flag (the RXCIF bit in the USARTn.STATUS register) is set, and the interrupt is generated if enabled.

The RXDATA register is the part of the RX buffer that can be read by the application software when RXCIF is set. When using frames with fewer than eight bits, the unused Most Significant bits (MSb) are read as zero. If 9-bit characters are used, the DATA[8] bit in the USARTn.RXDATAH register must be read before the DATA[7:0] bits in the USARTn.RXDATAL register.

24.3.2.4.1 Receiver Error Flags

The USART receiver features error detection mechanisms that uncover corruption of the transmission. These mechanisms include the following:

- Frame Error detection - controls whether the received frame is valid
- Buffer Overflow detection - indicates data loss due to the receiver buffer being full and overwritten by the new data
- Parity Error detection - checks the validity of the incoming frame by calculating its parity and comparing it to the Parity bit

Each error detection mechanism controls one error flag that can be read in the RXDATAH register:

- Frame Error (FERR)
- Buffer Overflow (BUFOVF)
- Parity Error (PERR)

The error flags are located in the RX buffer together with their corresponding frame. The RXDATAH register that contains the error flags must be read before the RXDATAL register, since reading the RXDATAL register will trigger the RX buffer to shift out the RXDATA bytes.

Note: If the Character Size bit field (the CHSIZE bits in the USARTn.CTRLC register) is set to nine bits, low byte first (9BITL), the RXDATAH register will, instead of the RXDATAL register, trigger the RX buffer to shift out the RXDATA bytes. The RXDATAL register must, in that case, be read before the RXDATAH register.

24.3.2.4.2 Disabling the Receiver

When disabling the receiver, the operation is immediate. The receiver buffer will be flushed, and data from ongoing receptions will be lost.

24.3.2.4.3 Flushing the Receive Buffer

If the RX buffer has to be flushed during normal operation, repeatedly read the DATA location (USARTn.RXDATAH and USARTn.RXDATAL registers) until the Receive Complete Interrupt Flag (the RXCIF bit in the USARTn.RXDATAH register) is cleared.

24.3.3 Communication Modes

The USART is a flexible peripheral that supports multiple different communication protocols. The available modes of operation can be split into two groups: Synchronous and asynchronous communication.

The synchronous communication relies on one device on the bus to be the master, providing the rest of the devices with a clock signal through the XCK pin. All the devices use this common clock signal for both transmission and reception, requiring no additional synchronization mechanism.

The device can be configured to run either as a master or a slave on the synchronous bus.

The asynchronous communication does not use a common clock signal. Instead, it relies on the communicating devices to be configured with the same baud rate. When receiving a transmission, the hardware synchronization mechanisms are used to align the incoming transmission with the receiving device peripheral clock.

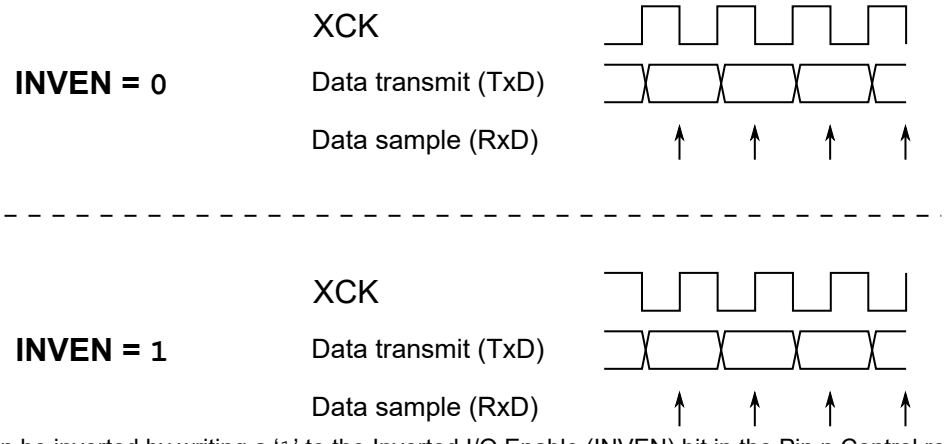
Four different modes of reception are available when communicating asynchronously. One of these modes can receive transmissions at twice the normal speed, sampling only eight times per bit instead of the normal 16. The other three operating modes use variations of synchronization logic, all receiving at normal speed.

24.3.3.1 Synchronous Operation

24.3.3.1.1 Clock Operation

The XCK pin direction controls whether the transmission clock is an input (Slave mode) or an output (Master mode). The corresponding port pin direction must be set to output for Master mode or to input for Slave mode (PORTx.DIRn). The data input (on RXD) is sampled at the XCK clock edge which is opposite the edge where data are transmitted (on TXD) as shown in the figure below.

Figure 24-4. Synchronous Mode XCK Timing



The I/O pin can be inverted by writing a '1' to the Inverted I/O Enable (INVEN) bit in the Pin n Control register of the port peripheral (PORTx.PINnCTRL). Using the inverted I/O setting for the corresponding XCK port pin, the XCK clock edges used for sampling RxD and transmitting on TxD can be selected. If the inverted I/O is disabled (INVEN = 0), the rising XCK clock edge represents the start of a new data bit, and the received data will be sampled at the falling

XCK clock edge. If inverted I/O is enabled (INVEN = 1), the falling XCK clock edge represents the start of a new data bit, and the received data will be sampled at the rising XCK clock edge.

24.3.3.1.2 External Clock Limitations

When the USART is configured in Synchronous Slave mode, the XCK signal must be provided externally by the master device. Since the clock is provided externally, configuring the BAUD register will have no impact on the transfer speed. Successful clock recovery requires the clock signal to be sampled at least twice for each rising and falling edge. The maximum XCK speed in Synchronous Operation mode, f_{Slave_XCK}, is therefore limited by:

$$f_{Slave\_XCK} < \frac{f_{CLK\_PER}}{4}$$

If the XCK clock has jitter, or if the high/low period duty cycle is not 50/50, the maximum XCK clock speed must be reduced accordingly to ensure that XCK is sampled a minimum of two times for each edge.

24.3.3.1.3 USART in Master SPI Mode

The USART may be configured to function with multiple different communication interfaces, and one of these is the Serial Peripheral Interface (SPI) where it can function as the master device. The SPI is a four-wire interface that enables a master device to communicate with one or multiple slaves.

Frame Formats

The serial frame for the USART in Master SPI mode always contains eight Data bits. The Data bits can be configured to be transmitted with either the LSb or MSb first, by writing to the Data Order bit (UDORD) in the Control C register (USARTn.CTRLC).

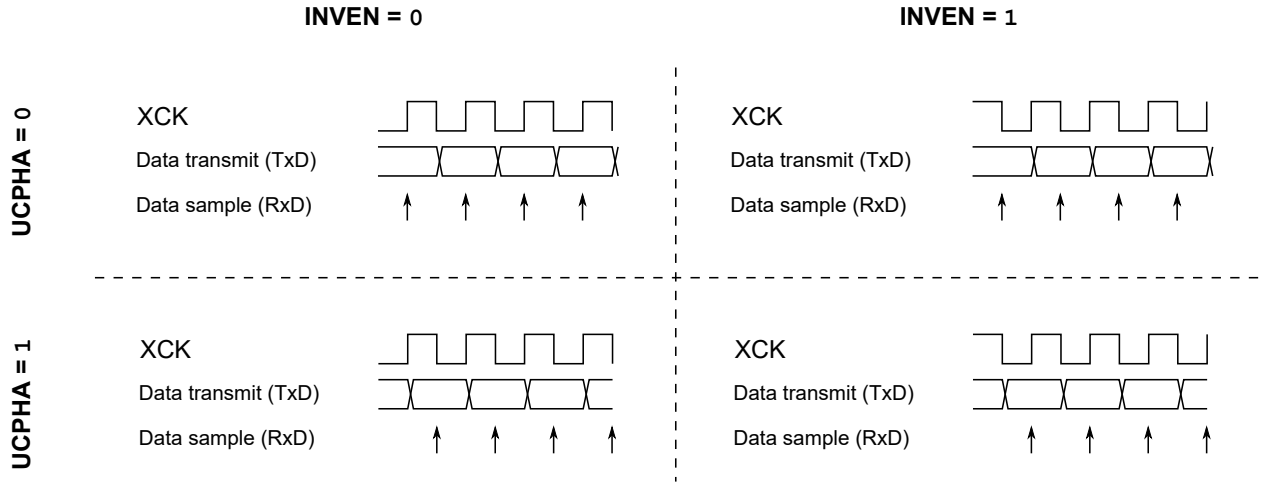
SPI does not use Start, Stop, or Parity bits, so the transmission frame can only consist of the Data bits.

Clock Generation

Being a master device in a synchronous communication interface, the USART in Master SPI mode must generate the interface clock to be shared with the slave devices. The interface clock is generated using the fractional Baud Rate Generator, which is described in 24.3.2.2.1 The Fractional Baud Rate Generator.

Each Data bit is transmitted by pulling the data line high or low for one full clock period. The receiver will sample bits in the middle of the transmitter hold period as shown in the figure below. It also shows how the timing scheme can be configured using the Inverted I/O Enable (INVEN) bit in the PORTx.PINnCTRL register and the USART Clock Phase (UCPHA) bit in the USARTn.CTRLC register.

Figure 24-5. Data Transfer Timing Diagrams



The table below further explains the figure above.

Table 24-2. Functionality of INVEN and UCPHA Bits

INVEN	UCPHA	Leading Edge (1)	Trailing Edge (1)
0	0	Rising, sample	Falling, transmit

.....continued			
INVEN	**UCPHA**	**Leading Edge (1)**	**Trailing Edge (1)**
0	1	Rising, transmit	Falling, sample
1	0	Falling, sample	Rising, transmit
1	1	Falling, transmit	Rising, sample

Note:

1. The leading edge is the first clock edge of a clock cycle. The trailing edge is the last clock edge of a clock cycle.

Data Transmission

Data transmission in Master SPI mode is functionally identical to general USART operation as described in the *Operation* section. The transmitter interrupt flags and corresponding USART interrupts are also identical. See 24.3.2.3 Data Transmission for further description.

Data Reception

Data reception in Master SPI mode is identical in function to general USART operation as described in the *Operation* section. The receiver interrupt flags and the corresponding USART interrupts are also identical, aside from the receiver error flags that are not in use and always read as '0'. See 24.3.2.4 Data Reception for further description.

USART in Master SPI Mode vs. SPI

The USART in Master SPI mode is fully compatible with a stand-alone SPI peripheral. Their data frame and timing configurations are identical. Some SPI specific special features are, however, not supported with the USART in Master SPI mode:

- Write Collision Flag Protection
- Double-Speed mode
- Multi-Master support

A comparison of the pins used with USART in Master SPI mode and with SPI is shown in the table below.

Table 24-3. Comparison of USART in Master SPI Mode and SPI Pins

USART	SPI	Comment
TXD	MOSI	Master out
RXD	MISO	Master in
XCK	SCK	Functionally identical
-	$\overline{SS}$	Not supported by USART in Master SPI mode(1)

Note:

1. For the stand-alone SPI peripheral, this pin is used with the Multi-Master function or as a dedicated Slave Select pin. The Multi-Master function is not available with the USART in Master SPI mode, and no dedicated Slave Select pin is available.

24.3.3.2 Asynchronous Operation

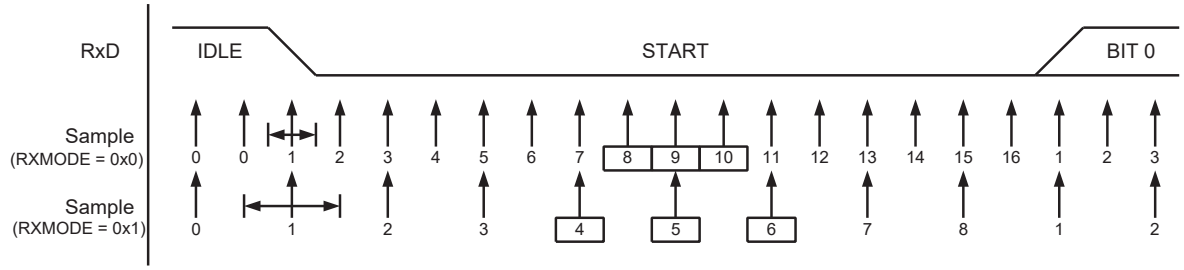
24.3.3.2.1 Clock Recovery

Since there is no common clock signal when using Asynchronous mode, each communicating device generates separate clock signals. These clock signals must be configured to run at the same baud rate for the communication to take place. The devices, therefore, run at the same speed, but their timing is skewed in relation to each other. To accommodate this, the USART features a hardware clock recovery unit which synchronizes the incoming asynchronous serial frames with the internally generated baud rate clock.

The figure below illustrates the sampling process for the Start bit of an incoming frame. It shows the timing scheme for both Normal and Double-Speed mode (the RXMODE bits in the USARTn.CTRLB register configured respectively to 0x00 and 0x01). The sample rate for Normal mode is 16 times the baud rate, while the sample rate for Double-Speed mode is eight times the baud rate (see 24.3.3.2.4 Double-Speed Operation for more details). The horizontal

arrows show the maximum synchronization error. Note that the maximum synchronization error is larger in Double-Speed mode.

Figure 24-6. Start Bit Sampling

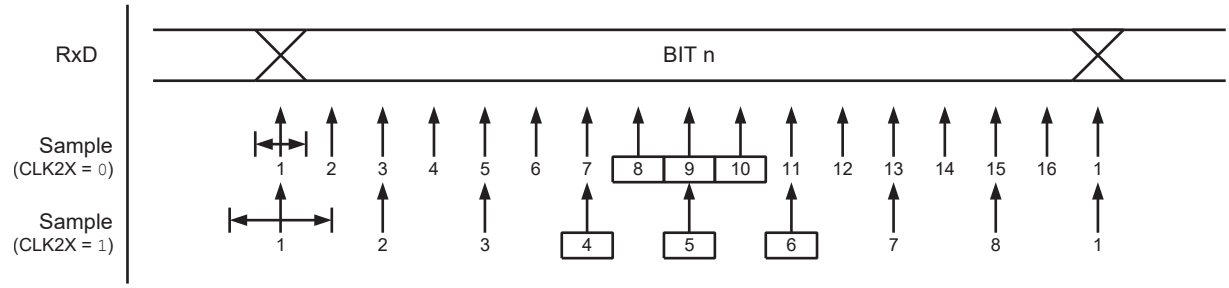


When the clock recovery logic detects a falling edge from Idle (high) state to the Start bit (low), the Start bit detection sequence is initiated. In the figure above, sample 1 denotes the first sample reading '0'. The clock recovery logic then uses three subsequent samples (samples 8, 9, and 10 in Normal mode, and samples 4, 5, 6 in Double-Speed mode) to decide if a valid Start bit is received. If two or three samples read '0', the Start bit is accepted. The clock recovery unit is synchronized, and the data recovery can begin. If less than two samples read '0', the Start bit is rejected. This process is repeated for each Start bit.

24.3.3.2.2 Data Recovery

As with clock recovery, the data recovery unit samples at a rate 8 or 16 times faster than the baud rate depending on whether it is running in Double-Speed or Normal mode, respectively. The figure below shows the sampling process for reading a bit in a received frame.

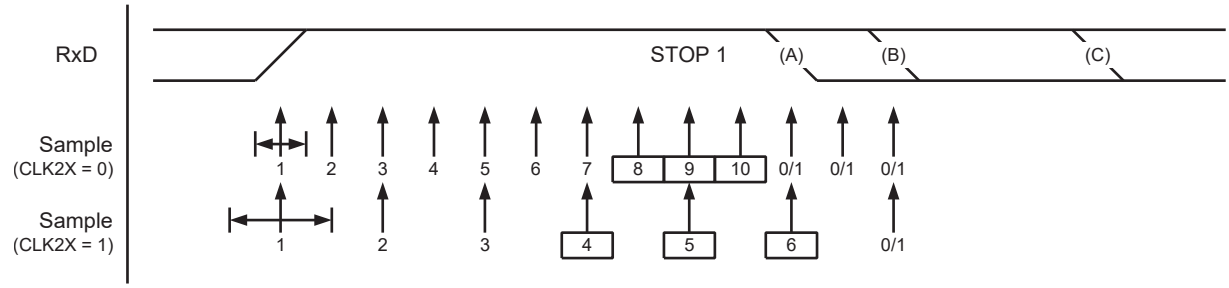
Figure 24-7. Sampling of Data and Parity Bits



A majority voting technique is, like with clock recovery, used on the three center samples for deciding the logic level of the received bit. The process is repeated for each bit until a complete frame is received.

The data recovery unit will only receive the first Stop bit while ignoring the rest if there are more. If the sampled Stop bit is read '0', the Frame Error flag will be set. The figure below shows the sampling of a Stop bit. It also shows the earliest possible beginning of the next frame's Start bit.

Figure 24-8. Stop Bit and Next Start Bit Sampling



A new high-to-low transition indicating the Start bit of a new frame can come right after the last of the bits used for majority voting. For Normal-Speed mode, the first low-level sample can be at the point marked (A) in the figure above. For Double-Speed mode the first low level must be delayed to point (B), being the first sample after the majority vote samples. Point (C) marks a Stop bit of full length at the nominal baud rate.

24.3.3.2.3 Error Tolerance

The speed of the internally generated baud rate and the externally received data rate should ideally be identical, but due to natural clock source error, this is normally not the case. The USART is tolerant of such error, and the limits of this tolerance make up what is sometimes known as the Operational Range.

The following tables list the operational range of the USART, being the maximum receiver baud rate error that can be tolerated. Note that Normal-Speed mode has higher toleration of baud rate variations than Double-Speed mode.

Table 24-4. Recommended Maximum Receiver Baud Rate Error for Normal Speed Mode

D	R_{slow} [%]	R_{fast} [%]	Maximum Total Error [%]	Recommended Max. Receiver Error [%]
5	93.20	106.67	-6.80/+6.67	±3.0
6	94.12	105.79	-5.88/+5.79	±2.5
7	94.81	105.11	-5.19/+5.11	±2.0
8	95.36	104.58	-4.54/+4.58	±2.0
9	95.81	104.14	-4.19/+4.14	±1.5
10	96.17	103.78	-3.83/+3.78	±1.5

Note:

- D: The sum of character size and parity size (D = 5 to 10 bits)
- R_{SLOW}: The ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate
- R_{FAST}: The ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate

Table 24-5. Recommended Maximum Receiver Baud Rate Error for Double Speed Mode

D	R_{slow} [%]	R_{fast} [%]	Maximum Total Error [%]	Recommended Max. Receiver Error [%]
5	94.12	105.66	-5.88/+5.66	±2.5
6	94.92	104.92	-5.08/+4.92	±2.0
7	95.52	104.35	-4.48/+4.35	±1.5
8	96.00	103.90	-4.00/+3.90	±1.5
9	96.39	103.53	-3.61/+3.53	±1.5
10	96.70	103.23	-3.30/+3.23	±1.0

Note:

- D: The sum of character size and parity size (D = 5 to 10 bits)
- R_{SLOW}: The ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate
- R_{FAST}: The ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate

The recommendations of the maximum receiver baud rate error were made under the assumption that the receiver and transmitter equally divide the maximum total error.

The following equations are used to calculate the maximum ratio of the incoming data rate and the internal receiver baud rate.

$R_{SLOW} = \frac{S(D+1)}{S(D+1)+S_F-1}$	$R_{FAST} = \frac{S(D+2)}{S(D+1)+S_M}$

- D: The sum of character size and parity size (D = 5 to 10 bits)
- S: Samples per bit. S = 16 for Normal Speed mode and S = 8 for Double-Speed mode.
- S_F: First sample number used for majority voting. SF = 8 for Normal-Speed mode and SF = 4 for Double-Speed mode.

- S_M: Middle sample number used for majority voting. SM = 9 for Normal-Speed mode and SM = 5 for Double-Speed mode.
- R_{SLOW}: The ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate
- R_{FAST}: The ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate

24.3.3.2.4 Double-Speed Operation

Double-speed operation allows for higher baud rates under asynchronous operation with lower peripheral clock frequencies. This operation mode is enabled by writing the RXMODE bits in the Control B (USARTn.CTRLB) register to 0x01.

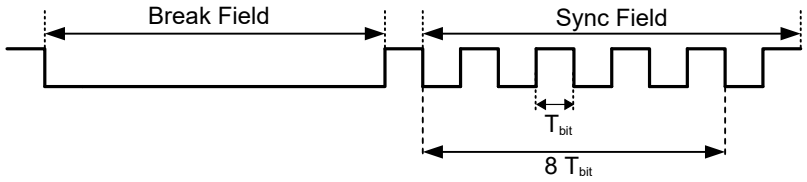
When enabled, the baud rate for a given asynchronous baud rate setting will be doubled. This is shown in the equations in 24.3.2.2.1 The Fractional Baud Rate Generator. In this mode, the receiver will use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery. This requires a more accurate baud rate setting and peripheral clock. See 24.3.3.2.3 Error Tolerance for more details.

24.3.3.2.5 Auto-Baud

The auto-baud feature lets the USART configure its BAUD register based on input from a communication device. This allows the device to communicate autonomously with multiple devices communicating with different baud rates. The USART peripheral features two auto-baud modes: Generic Auto-Baud mode and LIN Constrained Auto-Baud mode.

Both auto-baud modes must receive an auto-baud frame as seen in the figure below.

Figure 24-9. Auto-Baud Timing



The break field is detected when 12 or more consecutive low cycles are sampled and notifies the USART that it is about to receive the synchronization field. After the break field, when the Start bit of the synchronization field is detected, a counter running at the peripheral clock speed is started. The counter is then incremented for the next eight T_{bit} of the synchronization field. When all eight bits are sampled, the counter is stopped. The resulting counter value is in effect the new BAUD register value.

When the USART Receive mode is set to GENAUTO (the RXMODE bits in the USARTn.CTRLB register), the Generic Auto-Baud mode is enabled. In this mode, one can set the Wait For Break (WFB) bit in the USARTn.STATUS register to enable detection of a break field of any length (that is, also shorter than 12 cycles). This makes it possible to set an arbitrary new baud rate without knowing the current baud rate. If the measured sync field results in a valid BAUD value (0x0064 - 0xFFFF), the BAUD register is updated.

When USART Receive mode is set to LINAUTO mode (the RXMODE bits in the USARTn.CTRLB register), it follows the LIN format. The WFB functionality of the Generic Auto-Baud mode is not compatible with the LIN Constrained Auto-Baud mode. This means that the received signal must be low for 12 peripheral clock cycles or more for a break field to be valid. When a break field has been detected, the USART expects the following synchronization field character to be 0x55. If the received synchronization field character is not 0x55, the Inconsistent Sync Field Error Flag (the ISFIF bit in the USARTn.STATUS register) is set, and the baud rate is unchanged.

24.3.3.2.6 Half Duplex Operation

Half duplex is a type of communication where two or more devices may communicate with each other, but only one at a time. The USART can be configured to operate in the following half duplex modes:

- One-Wire mode
- RS-485 mode

One-Wire Mode

One-Wire mode is enabled by setting the Loop-Back Mode Enable (LBME) bit in the USARTn.CTRLA register. This will enable an internal connection between the TXD pin and the USART receiver, making the TXD pin a combined TxD/RxD line. The RXD pin will be disconnected from the USART receiver and may be controlled by a different peripheral.

In One-Wire mode, multiple devices are able to manipulate the TxD/RxD line at the same time. In the case where one device drives the pin to a logical high level (V_{CC}), and another device pulls the line low (GND), a short will occur. To accommodate this, the USART features an Open-Drain mode (the ODME bit in the USARTn.CTRLB register) which prevents the transmitter from driving a pin to a logical high level, thereby constraining it to only be able to pull it low. Combining this function with the internal pull-up feature (the PULLUPEN bit in the PORTx.PINnCTRL register) will let the line be held high through a pull-up resistor, allowing any device to pull it low. When the line is pulled low the current from V_{CC} to GND will be limited by the pull-up resistor. The TXD pin is automatically set to output by hardware when the Open-Drain mode is enabled.

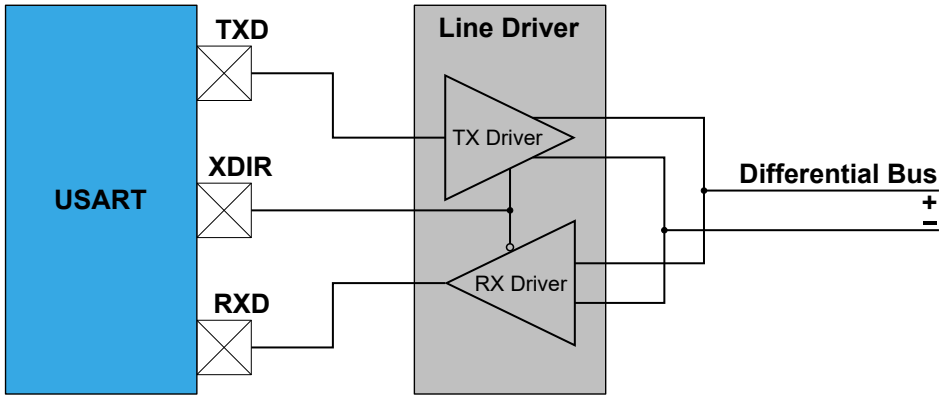
When the USART is transmitting to the TxD/RxD line, it will also receive its own transmission. This can be used to check for overlapping transmissions by checking if the received data are the same as the transmitted data as it should be.

RS-485 Mode

RS-485 is a communication standard supported by the USART peripheral. It is a physical interface that defines the setup of a communication circuit. Data are transmitted using differential signaling, making communication robust against noise. RS-485 is enabled by writing to the RS485 bit field (USARTn.CTRLA).

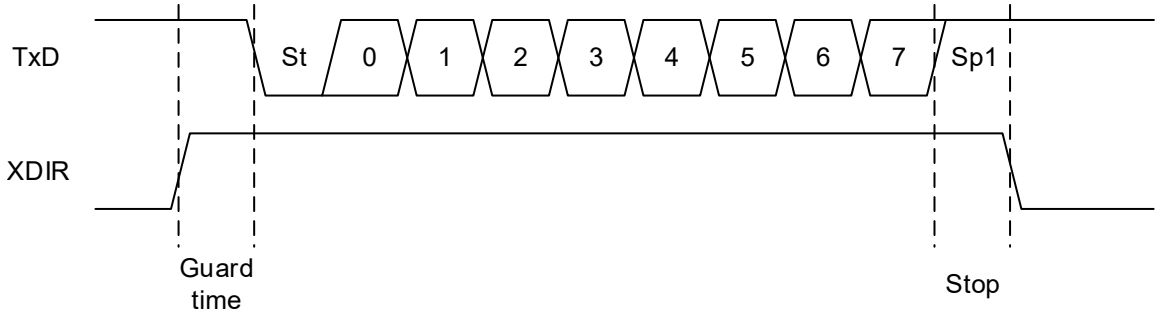
The RS-485 mode supports external line driver devices that convert a single USART transmission into corresponding differential pair signals. Writing RS485[0] to '1' enables the automatic control of the XDIR pin that can be used to enable transmission or reception for the line driver device. The USART automatically drives the XDIR pin high while the USART is transmitting and pulls it low when the transmission is complete. An example of such a circuit is shown in the figure below.

Figure 24-10. RS-485 Bus Connection



The XDIR pin goes high one baud clock cycle in advance of data being shifted out to allow some guard time to enable the external line driver. The XDIR pin will remain high for the complete frame including Stop bit(s).

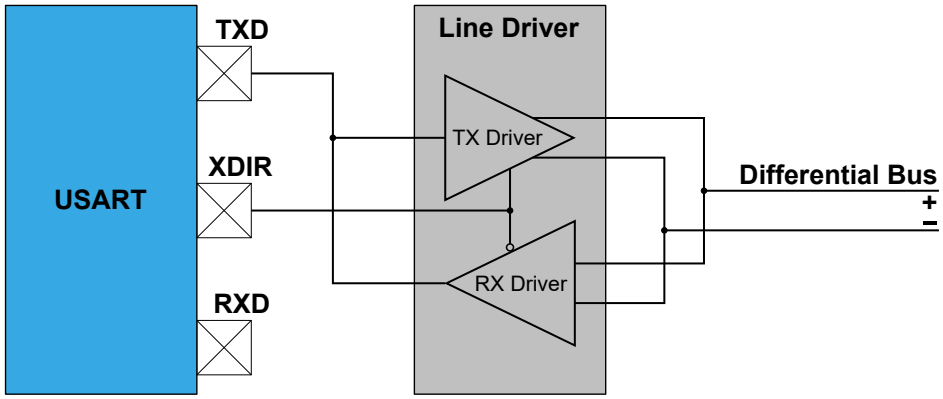
Figure 24-11. XDIR Drive Timing



Writing RS485[1] to '1' enables the RS-485 mode which automatically sets the TXD pin to output one clock cycle before starting transmission and sets it back to input when the transmission is complete.

RS-485 mode is compatible with One-Wire mode. One-Wire mode enables an internal connection between the TXD pin and the USART receiver, making the TXD pin a combined TxD/RxD line. The RXD pin will be disconnected from the USART receiver and may be controlled by a different peripheral. An example of such a circuit is shown in the figure below.

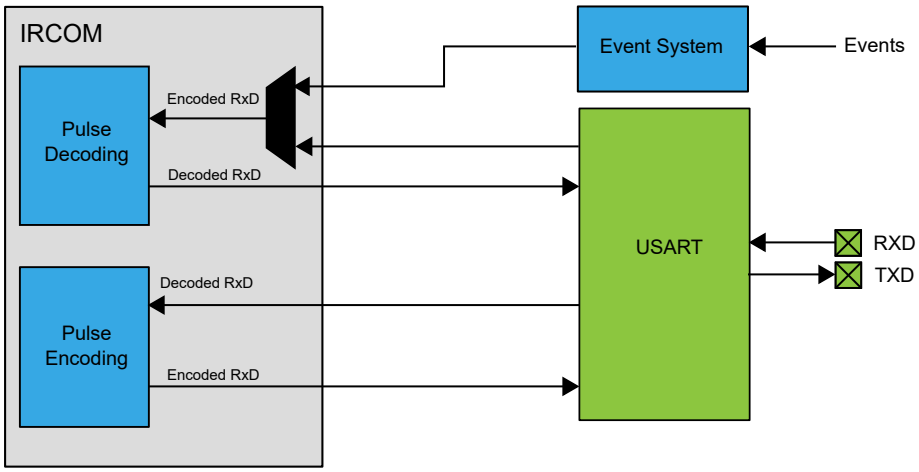
Figure 24-12. RS-485 with Loop-Back Mode Connection



24.3.3.2.7 IRCOM Mode of Operation

The USART peripheral can be configured in Infrared Communication mode (IRCOM) which is IrDA® 1.4 compatible with baud rates up to 115.2 kbps. When enabled, the IRCOM mode enables infrared pulse encoding/decoding for the USART.

Figure 24-13. Block Diagram



The USART is set in IRCOM mode by writing 0x02 to the CMODE bits in the USARTn.CTRLC register. The data on the TXD/RXD pins are the inverted values of the transmitted/received infrared pulse. It is also possible to select an event channel from the Event System as an input for the IRCOM receiver. This enables the IRCOM to receive input from the I/O pins or sources other than the corresponding RXD pin. This will disable the RxD input from the USART pin.

For transmission, three pulse modulation schemes are available:

- 3/16 of the baud rate period
- Fixed programmable pulse time based on the peripheral clock frequency
- Pulse modulation disabled

For the reception, a fixed programmable minimum high-level pulse-width for the pulse to be decoded as a logical '0' is used. Shorter pulses will then be discarded, and the bit will be decoded to logical '1' as if no pulse was received.

When IRCOM mode is enabled, Double-Speed mode cannot be used for the USART.

24.3.4 Additional Features

24.3.4.1 Parity

Parity bits can be used by the USART to check the validity of a data frame. The Parity bit is set by the transmitter based on the number of bits with the value of '1' in a transmission and controlled by the receiver upon reception. If

the Parity bit is inconsistent with the transmission frame, the receiver may assume that the data frame has been corrupted.

Even or odd parity can be selected for error checking by writing the Parity Mode (PMODE) bits in the USARTn.CTRLC register. If even parity is selected, the Parity bit is set to '1' if the number of Data bits with value '1' is odd (making the total number of bits with value '1' even). If odd parity is selected, the Parity bit is set to '1' if the number of data bits with value '1' is even (making the total number of bits with value '1' odd).

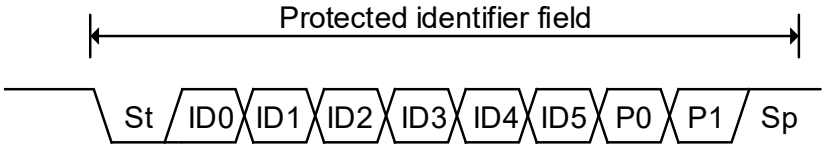
When enabled, the parity checker calculates the parity of the data bits in incoming frames and compares the result with the Parity bit of the corresponding frame. If a parity error is detected, the Parity Error flag (the PERR bit in the USARTn.RXDATAH register) is set.

If LIN Constrained Auto-Baud mode is enabled (RXMODE = 0x03 in the USARTn.CTRLB register), a parity check is only performed on the protected identifier field. A parity error is detected if one of the equations below is not true, which sets the Parity Error flag.

$$P0 = ID0 \text{ XOR } ID1 \text{ XOR } ID2 \text{ XOR } ID4$$

$$P1 = \text{ NOT } (ID1 \text{ XOR } ID3 \text{ XOR } ID4 \text{ XOR } ID5)$$

Figure 24-14. Protected Identifier Field and Mapping of Identifier and Parity Bits



24.3.4.2 Start-of-Frame Detection

The Start-of-Frame Detection feature enables the USART to wake up from Standby Sleep mode upon data reception.

When a high-to-low transition is detected on the RXD pin, the oscillator is powered up, and the USART peripheral clock is enabled. After start-up, the rest of the data frame can be received, provided that the baud rate is slow enough in relation to the oscillator start-up time. The start-up time of the oscillators varies with supply voltage and temperature. For details on oscillator start-up time characteristics, refer to the *Electrical Characteristics* section.

If a false Start bit is detected, the device will, if another wake-up source has not been triggered, go back into the Standby Sleep mode.

The Start-of-Frame detection works in Asynchronous mode only. It is enabled by writing the Start-of-Frame Detection Enable (SFDEN) bit in the USARTn.CTRLB register. If a Start bit is detected while the device is in Standby Sleep mode, the USART Receive Start Interrupt Flag (RXSIF) bit is set.

The USART Receive Complete Interrupt Flag (RXCIF) bit and the RXSIF bit share the same interrupt line, but each has its dedicated interrupt settings. The table below shows the USART Start Frame Detection modes, depending on the interrupt setting.

Table 24-6. USART Start Frame Detection Modes

SFDEN	RXSIF Interrupt	RXCIF Interrupt	Comment
0	x	x	Standard mode.
1	Disabled	Disabled	Only the oscillator is powered during the frame reception. If the interrupts are disabled and buffer overflow is ignored, all incoming frames will be lost.
1	Disabled	Enabled	System/all clocks are awakened on Receive Complete interrupt.
1	Enabled	x	System/all clocks are awakened when a Start bit is detected.

Note: The SLEEP instruction will not shut down the oscillator if there is ongoing communication.

24.3.4.3 Multiprocessor Communication

The Multiprocessor Communication mode (MPCM) effectively reduces the number of incoming frames that have to be handled by the receiver in a system with multiple microcontrollers communicating via the same serial bus. This

mode is enabled by writing a '1' to the MPCM bit in the Control B register (USARTn.CTRLB). In this mode, a dedicated bit in the frames is used to indicate whether the frame is an address or data frame type.

If the receiver is set up to receive frames that contain five to eight data bits, the first Stop bit is used to indicate the frame type. If the receiver is set up for frames with nine data bits, the ninth bit is used to indicate frame type. When the frame type bit is '1', the frame contains an address. When the frame type bit is '0', the frame is a data frame. If 5- to 8-bit character frames are used, the transmitter must be set to use two Stop bits, since the first Stop bit is used for indicating the frame type.

If a particular slave MCU has been addressed, it will receive the following data frames as usual, while the other slave MCUs will ignore the frames until another address frame is received.

24.3.4.3.1 Using Multiprocessor Communication

The following procedure should be used to exchange data in Multiprocessor Communication mode (MPCM):

1. All slave MCUs are in Multiprocessor Communication mode.
2. The master MCU sends an address frame, and all slaves receive and read this frame.
3. Each slave MCU determines if it has been selected.
4. The addressed MCU will disable MPCM and receive all data frames. The other slave MCUs will ignore the data frames.
5. When the addressed MCU has received the last data frame, it must enable MPCM again and wait for a new address frame from the master.

The process then repeats from step 2.

24.3.5 Events

The USART can generate the events described in the table below.

Table 24-7. Event Generators in USART

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
USARTn	XCK	The clock signal in SPI Master mode and Synchronous USART Master mode	Pulse	CLK_PER	One XCK period

The table below describes the event user and its associated functionality.

Table 24-8. Event Users in USART

User Name		Description	Input Detection	Async/Sync
Peripheral	Input			
USARTn	IREI	USARTn IrDA event input	Pulse	Sync

24.3.6 Interrupts

Table 24-9. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
RXC	Receive Complete interrupt	• There is unread data in the receive buffer (RXCIE) • Receive of Start-of-Frame detected (RXSIE) • Auto-Baud Error/ISFIF flag set (ABEIE)
DRE	Data Register Empty interrupt	The transmit buffer is empty/ready to receive new data (DREIE)
TXC	Transmit Complete interrupt	The entire frame in the Transmit Shift register has been shifted out and there are no new data in the transmit buffer (TXCIE)

When an Interrupt condition occurs, the corresponding Interrupt flag is set in the STATUS register (USARTn.STATUS).

An interrupt source is enabled or disabled by writing to the corresponding bit in the Control A register (USARTn.CTRLA).

An interrupt request is generated when the corresponding interrupt source is enabled, and the Interrupt flag is set. The interrupt request remains active until the Interrupt flag is cleared. See the USARTn.STATUS register for details on how to clear Interrupt flags.

24.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	RXDATAL	7:0	DATA[7:0]							
0x01	RXDATAH	7:0	RXCIF	BUFOVF				FERR	PERR	DATA[8]
0x02	TXDATAL	7:0	DATA[7:0]							
0x03	TXDATAH	7:0								DATA[8]
0x04	STATUS	7:0	RXCIF	TXCIF	DREIF	RXSIF	ISFIF		BDF	WFB
0x05	CTRLA	7:0	RXCIE	TXCIE	DREIE	RXSIE	LBME	ABEIE	RS485[1:0]	
0x06	CTRLB	7:0	RXEN	TXEN		SFDEN	ODME	RXMODE[1:0]		MPCM
0x07	CTRLC	7:0	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		
0x07	CTRLC	7:0	CMODE[1:0]					UDORD	UCPHA	
0x08	BAUD	7:0	BAUD[7:0]							
		15:8	BAUD[15:8]							
0x0A	CTRLD	7:0	ABW[1:0]							
0x0B	DBGCTRL	7:0								DBGRUN
0x0C	EVCTRL	7:0								IREI
0x0D	TXPLCTRL	7:0	TXPL[7:0]							
0x0E	RXPLCTRL	7:0		RXPL[6:0]						

24.5 Register Description

24.5.1 Receiver Data Register Low Byte

Name: RXDATAL
Offset: 0x00
Reset: 0x00
Property: -

Reading the USARTn.RXDATAL register will return the contents of the eight least significant RXDATA bits.
The receive buffer consists of a two-level buffer. The data buffer and the corresponding flags in the high byte of RXDATA will change state whenever the receive buffer is accessed (read). If the CHSIZE bits in the USARTn.CTRLC register are set to 9BIT Low byte first, read the USARTn.RXDATAL register before the USARTn.RXDATAH register. Otherwise, always read the USARTn.RXDATAH register before the USARTn.RXDATAL register in order to get the correct flags.

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

Bits 7:0 – DATA[7:0] Receiver Data Register

24.5.2 Receiver Data Register High Byte

Name: RXDATAH
Offset: 0x01
Reset: 0x00
Property: -

Reading the USARTn.RXDATAH register location will return the contents of the ninth RXDATA bit plus Status bits.

The receive buffer consists of a two-level buffer. The data buffer and the corresponding flags in the high byte of USARTn.RXDATAH will change state whenever the receive buffer is accessed (read). If the CHSIZE bits in the USARTn.CTRLC register are set to 9BIT Low byte first, read the USARTn.RXDATAL register before the USARTn.RXDATAH register. Otherwise, always read the USARTn.RXDATAH register before the USARTn.RXDATAL register in order to get the correct flags.

Bit	7	6	5	4	3	2	1	0
	RXCIF	BUFOVF				FERR	PERR	DATA[8]
Access	R	R				R	R	R
Reset	0	0				0	0	0

Bit 7 – RXCIF USART Receive Complete Interrupt Flag
This flag is set when there are unread data in the receive buffer and cleared when the receive buffer is empty (that is, does not contain any unread data). When the receiver is disabled the receive buffer will be flushed and, consequently, the RXCIF bit will become '0'.

Bit 6 – BUFOVF Buffer Overflow
The BUFOVF flag indicates data loss due to a "receiver buffer full" condition. This flag is set if a Buffer Overflow condition is detected. A buffer overflow occurs when the receive buffer is full (two characters), it is a new character waiting in the Receive Shift register, and a new Start bit is detected. This flag is valid until the receive buffer (USARTn.RXDATAL) is read.
This flag is not used in Master SPI mode of operation.

Bit 2 – FERR Frame Error
The FERR flag indicates the state of the first Stop bit of the next readable frame stored in the receive buffer. This bit is set if the received character had a frame error, that is, when the first Stop bit was '0' and cleared when the Stop bit of the received data is '1'. This bit is valid until the receive buffer (USARTn.RXDATAL) is read. The FERR bit is not affected by the SBMODE bit in the USARTn.CTRLC register since the receiver ignores all, except for the first Stop bit.
This flag is not used in Master SPI mode of operation.

Bit 1 – PERR Parity Error
If parity checking is enabled and the next character in the receive buffer has a parity error, this flag is set. If parity check is not enabled the PERR bit will always be read as '0'. This bit is valid until the receive buffer (USARTn.RXDATAL) is read. For details on parity calculation refer to 24.3.4.1 Parity. If USART is set to LINAUTO mode, this bit will be a parity check of the protected identifier field and will be valid when the DATA[8] bit in the USARTn.RXDATAH register reads low.
This flag is not used in Master SPI mode of operation.

Bit 0 – DATA[8] Receiver Data Register
When the USART receiver is configured to LINAUTO mode, this bit indicates if the received data are within the response space of a LIN frame. If the received data are in the protected identifier field, this bit will be read as '0'. Otherwise, the bit will be read as '1'. For a receiver mode other than LINAUTO mode, the DATA[8] bit holds the ninth data bit in the received character when operating with serial frames with nine data bits.

24.5.3 Transmit Data Register Low Byte

Name: TXDATAL
Offset: 0x02
Reset: 0x00
Property: -

The Transmit Data Buffer (TXB) register will be the destination for data written to the USARTn.TXDATAL register location.

For 5-, 6-, or 7-bit characters the upper, unused bits will be ignored by the transmitter and set to zero by the receiver.

The transmit buffer can only be written when the DREIF flag in the USARTn.STATUS register is set. Data written to the DATA bits when the DREIF flag is not set will be ignored by the USART transmitter. When data are written to the transmit buffer, and the transmitter is enabled, the transmitter will load the data into the Transmit Shift register when the Shift register is empty. The data are then transmitted on the TXD pin.

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] Transmit Data Register

24.5.4 Transmit Data Register High Byte

Name: TXDATAH
Offset: 0x03
Reset: 0x00
Property: -

The USARTn.TXDATAH register holds the ninth data bit in the character to be transmitted when operating with serial frames with nine data bits. When used, this bit must be written before writing to the USARTn.TXDATAL register except if the CHSIZE bits in the USARTn.CTRLC register are is set to 9BIT low byte first, where the USARTn.TXDATAL register should be written first.
This bit is unused in Master SPI mode of operation.

Bit	7	6	5	4	3	2	1	0
								DATA[8]
Access								R/W
Reset								0

Bit 0 – DATA[8] Transmit Data Register
This bit is used when CHSIZE=9BIT in the USARTn.CTRLC register.

24.5.5 USART Status Register

Name: STATUS
Offset: 0x04
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RXCIF	TXCIF	DREIF	RXSIF	ISFIF		BDF	WFB
Access	R	R/W	R	R/W	R/W		R/W	R/W
Reset	0	0	1	0	0		0	0

Bit 7 – RXCIF USART Receive Complete Interrupt Flag
This flag is set to '1' when there are unread data in the receive buffer and cleared when the receive buffer is empty (that is, does not contain any unread data). When the receiver is disabled the receive buffer will be flushed and, consequently, the RXCIF bit will become '0'.
When interrupt-driven data reception is used, the receive complete interrupt routine must read the received data from RXDATA in order to clear the RXCIF. If not, a new interrupt will occur directly after the return from the current interrupt.

Bit 6 – TXCIF USART Transmit Complete Interrupt Flag
This flag is set when the entire frame in the Transmit Shift register has been shifted out, and there are no new data in the transmit buffer (TXDATA).
Writing a '1' to this bit will clear the flag.

Bit 5 – DREIF USART Data Register Empty Flag
This flag indicates if the transmit buffer (TXDATA) is ready to receive new data. The flag is set to '1' when the transmit buffer is empty and is '0' when the transmit buffer contains data to be transmitted but has not yet been moved into the Shift register. The DREIF bit is set after a Reset to indicate that the transmitter is ready. Always write this bit to '0' when writing the STATUS register.
DREIF is cleared to '0' by writing TXDATAL. When interrupt-driven data transmission is used, the Data Register Empty interrupt routine must either write new data to TXDATA in order to clear DREIF or disable the Data Register Empty interrupt. If not, a new interrupt will occur directly after the return from the current interrupt.

Bit 4 – RXSIF USART Receive Start Interrupt Flag
This flag indicates a valid Start condition on the RxD line. The flag is set when the system is in Standby Sleep mode and a high (IDLE) to low (START) valid transition is detected on the RxD line. If the start detection is not enabled, the RXSIF bit will always read '0'. This flag can only be cleared by writing a '1' to its bit location. This flag is not used in the Master SPI mode operation.

Bit 3 – ISFIF Inconsistent Sync Field Interrupt Flag
This flag is set when the auto-baud is enabled and the Sync Field bit time is too fast or too slow to give a valid baud setting. It will also be set when USART is set to LINAUTO mode, and the SYNC character differ from data value 0x55.
Writing a '1' to this bit will clear the flag and bring the USART back to Idle state.

Bit 1 – BDF Break Detected Flag
This flag is intended for USART configured to LINAUTO Receive mode. The break detector has a fixed threshold of 11 bits low for a break to be detected. The BDF bit is set after a valid break and sync character is detected. The bit is automatically cleared when the next data are received. The bit will behave identically when the USART is set to GENAUTO mode. In NORMAL or CLK2X Receive mode, the BDF bit is unused.
This bit is cleared by writing a '1' to it.

Bit 0 – WFB Wait For Break
Writing this bit to '1' will register the next low and high transition on the RxD line as a break character. This can be used to wait for a break character of arbitrary width. Combined with USART set to GENAUTO mode, this allows the

user to set any BAUD rate through BREAK and SYNC as long as it falls within the valid range of the USARTn.BAUD register. This bit will always read '0'.

24.5.6 Control A

Name: CTRLA
Offset: 0x05
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	DREIE	RXSIE	LBME	ABEIE	RS485[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 – RXCIE Receive Complete Interrupt Enable
This bit enables the Receive Complete interrupt (interrupt vector RXC). The enabled interrupt will be triggered when the RXCIF bit in the USARTn.STATUS register is set.

Bit 6 – TXCIE Transmit Complete Interrupt Enable
This bit enables the Transmit Complete interrupt (interrupt vector TXC). The enabled interrupt will be triggered when the TXCIF bit in the USARTn.STATUS register is set.

Bit 5 – DREIE Data Register Empty Interrupt Enable
This bit enables the Data Register Empty interrupt (interrupt vector DRE). The enabled interrupt will be triggered when the DREIF bit in the USART.STATUS register is set.

Bit 4 – RXSIE Receiver Start Frame Interrupt Enable
Writing a '1' to this bit enables the Start Frame Detector to generate an interrupt on interrupt vector RXC when a Start-of-Frame condition is detected.

Bit 3 – LBME Loop-back Mode Enable
Writing a '1' to this bit enables an internal connection between the TXD pin and the USART receiver and disables input from the RXD pin to the USART receiver.

Bit 2 – ABEIE Auto-baud Error Interrupt Enable
Writing a '1' to this bit enables the auto-baud error interrupt on interrupt vector RXC. The enabled interrupt will trigger for conditions where the ISFIF flag is set.

Bits 1:0 – RS485[1:0] RS-485 Mode
These bits enable the RS-485 and select the operation mode. Writing RS485[0] to '1' enables the RS-485 mode which automatically drives the XDIR pin high one clock cycle before starting transmission and pulls it low again when the transmission is complete. Writing RS485[1] to '1' enables the RS-485 mode which automatically sets the TXD pin to output one clock cycle before starting transmission and sets it back to input when the transmission is complete.

24.5.7 Control B

Name: CTRLB
Offset: 0x06
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RXEN	TXEN		SFDEN	ODME	RXMODE[1:0]		MPCM
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 – RXEN Receiver Enable
Writing this bit to '1' enables the USART receiver. Disabling the receiver will flush the receive buffer invalidating the FERR, BUFOVF, and PERR flags. In GENAUTO and LINAUTO mode, disabling the receiver will reset the auto-baud detection logic.

Bit 6 – TXEN Transmitter Enable
Writing this bit to '1' enables the USART transmitter. The transmitter will override normal port operation for the TXD pin when enabled. Disabling the transmitter (writing the TXEN bit to '0') will not become effective until ongoing and pending transmissions are completed (that is, when the Transmit Shift register and Transmit Buffer register does not contain data to be transmitted). When the transmitter is disabled, it will no longer override the TXD pin, and the pin direction is automatically set as input by hardware, even if it was configured as output by the user.

Bit 4 – SFDEN Start-of-Frame Detection Enable
Writing this bit to '1' enables the USART Start-of-Frame Detection mode. The Start-of-Frame detector is able to wake up the system from Idle or Standby Sleep modes when a high (IDLE) to low (START) transition is detected on the RxD line.

Bit 3 – ODME Open Drain Mode Enable
Writing this bit to '1' gives the TXD pin open-drain functionality. Internal Pull-up should be enabled for the TXD pin (the PULLUPEN bit in the PORTx.PINnCTRL register) to prevent the line from floating when a logic '1' is output to the TXD pin.

Bits 2:1 – RXMODE[1:0] Receiver Mode
Writing these bits select the receiver mode of the USART. In the CLK2X mode, the divisor of the baud rate divider will be reduced from 16 to 8 effectively doubling the transfer rate for Asynchronous Communication modes. For synchronous operation, the CLK2X mode has no effect, and the RXMODE bits should always be written to 0x00. RXMODE must be 0x00 when the USART Communication mode is configured to IRCOM. Setting RXMODE to GENAUTO enables generic auto-baud where the SYNC character is valid when eight bits alternating between '0' and '1' have been registered. In this mode, any SYNC character that gives a valid BAUD rate will be accepted. In LINAUTO mode the SYNC character is constrained and found valid if every two bits falls within 32 ±6 baud samples of the internal baud rate and match data value 0x55. The GENAUTO and LINAUTO modes are only supported for USART operated in Asynchronous Slave mode.

Value	Name	Description
0x00	NORMAL	Normal USART mode, standard transmission speed
0x01	CLK2X	Normal USART mode, double transmission speed
0x02	GENAUTO	Generic Auto-Baud mode
0x03	LINAUTO	LIN Constrained Auto-Baud mode

Bit 0 – MPCM Multi-Processor Communication Mode
Writing a '1' to this bit enables the Multi-Processor Communication mode: The USART receiver ignores all incoming frames that do not contain address information. The transmitter is unaffected by the MPCM setting. For more information see 24.3.4.3 Multiprocessor Communication.

24.5.8 Control C - Asynchronous Mode

Name: CTRLC
Offset: 0x07
Reset: 0x03
Property: -

This register description is valid for all modes except the Master SPI mode. When the USART Communication Mode bits (CMODE) in this register are written to 'MSPI', see CTRLC - Master SPI mode for the correct description.

Bit	7	6	5	4	3	2	1	0
	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[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	1	1

Bits 7:6 – CMODE[1:0] USART Communication Mode
Writing these bits select the Communication mode of the USART.
Writing a 0x03 to these bits alters the available bit fields in this register, see CTRLC - Master SPI mode.

Value	Name	Description
0x00	ASYNCHRONOUS	Asynchronous USART
0x01	SYNCHRONOUS	Synchronous USART
0x02	IRCOM	Infrared Communication
0x03	MSPI	Master SPI

Bits 5:4 – PMODE[1:0] Parity Mode
Writing these bits enable and select the type of parity generation.
When 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, compare it to the PMODE setting, and set the Parity Error (PERR) flag in the STATUS (USARTn.STATUS) register if a mismatch is detected.

Value	Name	Description
0x0	DISABLED	Disabled
0x1	-	Reserved
0x2	EVEN	Enabled, even parity
0x3	ODD	Enabled, odd parity

Bit 3 – SBMODE Stop Bit Mode
Writing this bit selects the number of Stop bits to be inserted by the transmitter.
The receiver ignores this setting.

Value	Description
0	1 Stop bit
1	2 Stop bits

Bits 2:0 – CHSIZE[2:0] Character Size
Writing these bits select the number of data bits in a frame. The receiver and transmitter use the same setting. For 9BIT character size, the order of which byte to read or write first, low or high byte of RXDATA or TXDATA, is selectable.

Value	Name	Description
0x00	5BIT	5-bit
0x01	6BIT	6-bit
0x02	7BIT	7-bit
0x03	8BIT	8-bit
0x04	-	Reserved
0x05	-	Reserved
0x06	9BITL	9-bit (Low byte first)
0x07	9BITH	9-bit (High byte first)

24.5.9 Control C - Master SPI Mode

Name: CTRLC
Offset: 0x07
Reset: 0x00
Property: -

This register description is valid only when the USART is in Master SPI mode (CMODE written to MSPI). For other CMODE values, see CTRLC - Asynchronous mode.

See 24.3.3.1.3 USART in Master SPI Mode for a full description of the Master SPI mode operation.

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

Bits 7:6 – CMODE[1:0] USART Communication Mode
Writing these bits select the communication mode of the USART.
Writing a value different than `0x03` to these bits alters the available bit fields in this register, see CTRLC - Asynchronous mode.

Value	Name	Description
0x00	ASYNCHRONOUS	Asynchronous USART
0x01	SYNCHRONOUS	Synchronous USART
0x02	IRCOM	Infrared Communication
0x03	MSPI	Master SPI

Bit 2 – UDORD USART Data Order
Writing this bit selects the frame format.
The receiver and transmitter use the same setting. Changing the setting of the UDORD bit will corrupt all ongoing communication for both the receiver and the transmitter.

Value	Description
0	MSb of the data word is transmitted first
1	LSb of the data word is transmitted first

Bit 1 – UCPHA USART Clock Phase
The UCPHA bit setting determines if data are sampled on the leading (first) edge or tailing (last) edge of XCKn. Refer to Clock Generation for details.

24.5.10 Baud Register

Name: BAUD
Offset: 0x08
Reset: 0x00
Property: -

The USARTn.BAUDL and USARTn.BAUDH register pair represents the 16-bit value, USARTn.BAUD. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

Ongoing transmissions of the transmitter and receiver will be corrupted if the baud rate is changed. Writing to this register will trigger an immediate update of the baud rate prescaler. For more information on how to set the baud rate, see Table 24-1, Equations for Calculating Baud Rate Register Setting.

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 15:8 – BAUD[15:8] USART Baud Rate High Byte
These bits hold the MSB of the 16-bit Baud register.

Bits 7:0 – BAUD[7:0] USART Baud Rate Low Byte
These bits hold the LSB of the 16-bit Baud register.

24.5.11 Control D

Name: CTRLD
Offset: 0x0a
Reset: 0x00
Property: -

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

Bits 7:6 – ABW[1:0] Auto-baud Window Size
These bits set the window size for which the SYNC character bits are validated.

Value	Name	Description
0x00	WDW0	18% tolerance
0x01	WDW1	15% tolerance
0x02	WDW2	21% tolerance
0x03	WDW3	25% tolerance

24.5.12 Debug Control Register

Name: DBGCTRL
Offset: 0x0B
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

24.5.13 IrDA Control Register

Name: EVCTRL
Offset: 0x0C
Reset: 0x00
Property: -

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

Bit 0 – IREI IrDA Event Input Enable
This bit enables the event source for the IRCOM Receiver. If event input is selected for the IRCOM receiver, the input from the USART's RXD pin is automatically disabled.

24.5.14 IRCOM Transmitter Pulse Length Control Register

Name: TXPLCTRL
Offset: 0x0D
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	TXPL[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 – TXPL[7:0] Transmitter Pulse Length
This 8-bit value sets the pulse modulation scheme for the transmitter. Setting this register will have effect only if IRCOM mode is selected by the USART, and it must be configured before the USART transmitter is enabled (TXEN).

Value	Description
0x00	3/16 of the baud rate period pulse modulation is used.
0x01-0xFE	Fixed pulse length coding is used. The 8-bit value sets the number of peripheral clock periods for the pulse. The start of the pulse will be synchronized with the rising edge of the baud rate clock.
0xFF	Pulse coding disabled. RX and TX signals pass through the IRCOM module unaltered. This enables other features through the IRCOM module, such as half-duplex USART, loop-back testing, and USART RX input from an event channel.

24.5.15 IRCOM Receiver Pulse Length Control Register

Name: RXPLCTRL
Offset: 0x0E
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		RXPL[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 6:0 – RXPL[6:0] Receiver Pulse Length
This 7-bit value sets the filter coefficient for the IRCOM transceiver. Setting this register will only have effect if IRCOM mode is selected by a USART, and it must be configured before the USART receiver is enabled (RXEN).

Value	Description
0x00	Filtering disabled.
0x01-0x7F	Filtering enabled. The value of RXPL+1 represents the number of samples required for a received pulse to be accepted.

25. SPI - Serial Peripheral Interface

25.1 Features

- Full Duplex, Three-Wire Synchronous Data Transfer
- Master or Slave Operation
- LSb First or MSb First Data Transfer
- Seven Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wake-up from Idle Mode
- Double-Speed (CK/2) Master SPI Mode

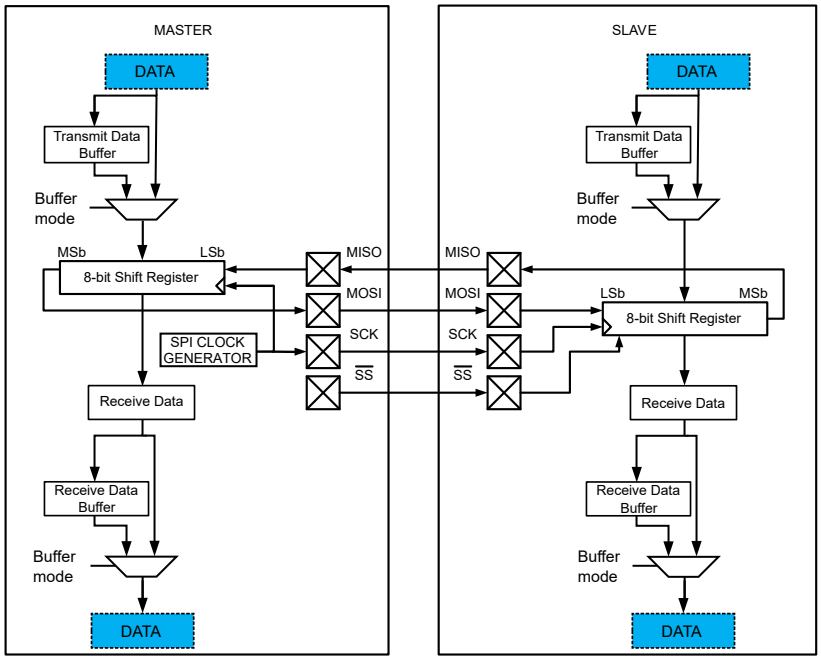
25.2 Overview

The Serial Peripheral Interface (SPI) is a high-speed synchronous data transfer interface using three or four pins. It allows full duplex communication between an AVR® device and peripheral devices, or between several microcontrollers. The SPI peripheral can be configured as either master or slave. The master initiates and controls all data transactions.

The interconnection between master and slave devices with SPI is shown in the block diagram. The system consists of two shift registers and a master clock generator. The SPI master initiates the communication cycle by pulling the desired slave's Slave Select ($\overline{SS}$) signal low. The master and slave prepare the data to be sent to their respective shift registers, and the master generates the required clock pulses on the SCK line to exchange data. Data are always shifted from master to slave on the master output, slave input (MOSI) line, and from slave to master on the master input, slave output (MISO) line.

25.2.1 Block Diagram

Figure 25-1. SPI Block Diagram



The SPI is built around an 8-bit shift register that will shift data out and in at the same time. The Transmit Data register and the Receive Data register are not physical registers but are mapped to other registers when written or

read: Writing the Transmit Data register (SPIn.DATA) will write the shift register in Normal mode and the Transmit Buffer register in Buffer mode. Reading the Receive Data register (SPIn.DATA) will read the Receive Data register in Normal mode and the Receive Data Buffer in Buffer mode.

In Master mode, the SPI has a clock generator to generate the SCK clock. In Slave mode, the received SCK clock is synchronized and sampled to trigger the shifting of data in the shift register.

25.2.2 Signal Description

Table 25-1. Signals in Master and Slave Mode

Signal	Description	Pin Configuration	
		Master Mode	Slave Mode
MOSI	Master Out Slave In	User defined(1)	Input
MISO	Master In Slave Out	Input	User defined(1,2)
SCK	Serial Clock	User defined(1)	Input
$\overline{SS}$	Slave Select	User defined(1)	Input

Note:

1. If the pin data direction is configured as output, the pin level is controlled by the SPI.
2. If the SPI is in Slave mode and the MISO pin data direction is configured as output, the $\overline{SS}$ pin controls the MISO pin output in the following way:
   - If the $\overline{SS}$ pin is driven low, the MISO pin is controlled by the SPI.
   - If the $\overline{SS}$ pin is driven high, the MISO pin is tri-stated.

When the SPI module is enabled, the pin data direction for the signals marked with “Input” in Table 25-1 is overridden.

25.3 Functional Description

25.3.1 Initialization

Initialize the SPI to a basic functional state by following these steps:

1. Configure the $\overline{SS}$ pin in the port peripheral.
2. Select SPI master/slave operation by writing the Master/Slave Select bit (MASTER) in the Control A register (SPIn.CTRLA).
3. In Master mode, select the clock speed by writing the Prescaler bits (PRESC) and the Clock Double bit (CLK2X) in SPIn.CTRLA.
4. Optional: Select the Data Transfer mode by writing to the MODE bits in the Control B register (SPIn.CTRLB).
5. Optional: Write the Data Order bit (DORD) in SPIn.CTRLA.
6. Optional: Setup Buffer mode by writing BUFEN and BUFWR bits in the Control B register (SPIn.CTRLB).
7. Optional: To disable the multi-master support in Master mode, write ‘1’ to the Slave Select Disable bit (SSD) in SPIn.CTRLB.
8. Enable the SPI by writing a ‘1’ to the ENABLE bit in SPIn.CTRLA.

25.3.2 Operation

25.3.2.1 Master Mode Operation

When the SPI is configured in Master mode, a write to the SPIn.DATA register will start a new transfer. The SPI master can operate in two modes, Normal and Buffer, as explained below.

25.3.2.1.1 Normal Mode

In Normal mode, the system is single-buffered in the transmit direction and double-buffered in the receive direction. This influences the data handling in the following ways:

1. New bytes to be sent cannot be written to the DATA register (SPIn.DATA) before the entire transfer has completed. A premature write will cause corruption of the transmitted data, and the Write Collision flag (WRCOL in SPIn.INTFLAGS) will be set.
2. Received bytes are written to the Receive Data Buffer register immediately after the transmission is completed.
3. The Receive Data Buffer register has to be read before the next transmission is completed or data will be lost. This register is read by reading SPIn.DATA.
4. The Transmit Data Buffer and Receive Data Buffer registers are not used in Normal mode.

After a transfer has completed, the Interrupt Flag will be set in the Interrupt Flags register (IF in SPIn.INTFLAGS). This will cause the corresponding interrupt to be executed if this interrupt and the global interrupts are enabled. Setting the Interrupt Enable (IE) bit in the Interrupt Control register (SPIn.INTCTRL) will enable the interrupt.

25.3.2.1.2 Buffer Mode

The Buffer mode is enabled by writing the BUFEN bit in the SPIn.CTRLB register to '1'. The BUFWR bit in SPIn.CTRLB has no effect in Master mode. In Buffer mode, the system is double-buffered in the transmit direction and triple-buffered in the receive direction. This influences the data handling the following ways:

1. New bytes can be written to the DATA register (SPIn.DATA) as long as the Data Register Empty Interrupt Flag (DREIF) in the Interrupt Flag Register (SPIn.INTFLAGS) is set. The first write will be transmitted right away, and the following write will go to the Transmit Data Buffer register.
2. A received byte is placed in a two-entry Receive First-In, First-Out (RX FIFO) queue comprised of the Receive Data register and Receive Data Buffer immediately after the transmission is completed.
3. The DATA register is used to read from the RX FIFO. The RX FIFO must be read at least every second transfer to avoid any loss of data.

When both the shift register and the Transmit Data Buffer register become empty, the Transfer Complete Interrupt Flag (TXCIF) in the Interrupt Flags register (SPIn.INTFLAGS) will be set. This will cause the corresponding interrupt to be executed if this interrupt and the global interrupts are enabled. Setting the Transfer Complete Interrupt Enable (TXCIE) in the Interrupt Control register (SPIn.INTCTRL) enables the Transfer Complete Interrupt.

25.3.2.1.3 $\overline{SS}$ Pin Functionality in Master Mode - Multi-Master Support

In Master mode, the Slave Select Disable bit in Control Register B (SSD bit in SPIn.CTRLB) controls how the SPI uses the $\overline{SS}$ pin.

- If SSD in SPIn.CTRLB is '0', the SPI can use the $\overline{SS}$ pin to transition from Master to Slave mode. This allows multiple SPI masters on the same SPI bus.
- If SSD in SPIn.CTRLB is '0', and the $\overline{SS}$ pin is configured as an output pin, it can be used as a regular I/O pin or by other peripheral modules, and will not affect the SPI system.
- If SSD in SPIn.CTRLB is '1', the SPI does not use the $\overline{SS}$ pin, and it can be used as a regular I/O pin, or by other peripheral modules.

If the SSD bit in SPIn.CTRLB is '0', and the $\overline{SS}$ is configured as an input pin, the $\overline{SS}$ pin must be held high to ensure master SPI operation. A low level will be interpreted as another master is trying to take control of the bus. This will switch the SPI into Slave mode, and the hardware of the SPI will perform the following actions:

1. The master bit in the SPI Control A Register (MASTER in SPIn.CTRLA) is cleared, and the SPI system becomes a slave. The direction of the SPI pins will be switched when conditions in Table 25-2 are met.
2. The Interrupt Flag in the Interrupt Flags register (IF in SPIn.INTFLAGS) will be set. If the interrupt is enabled and the global interrupts are enabled, the interrupt routine will be executed.

Table 25-2. Overview of the $\overline{SS}$ Pin Functionality when the SSD Bit in SPIn.CTRLB is '0'

$\overline{SS}$ Configuration	$\overline{SS}$ Pin-Level	Description
Input	High	Master activated (selected)
	Low	Master deactivated, switched to Slave mode
Output	High	Master activated (selected)
	Low	

Note: If the device is in Master mode and it cannot be ensured that the $\overline{SS}$ pin will stay high between two transmissions, the status of the Master bit (the MASTER bit in SPIn.CTRLA) has to be checked before a new byte is written. After the Master bit has been cleared by a low level on the $\overline{SS}$ line, it must be set by the application to re-enable the SPI Master mode.

25.3.2.2 Slave Mode

In Slave mode, the SPI peripheral receives SPI clock and Slave Select from a Master. Slave mode supports three operational modes: One Normal mode and two configurations for the Buffered mode. In Slave mode, the control logic will sample the incoming signal on the SCK pin. To ensure correct sampling of this clock signal, the minimum low and high periods must each be longer than two peripheral clock cycles.

25.3.2.2.1 Normal Mode

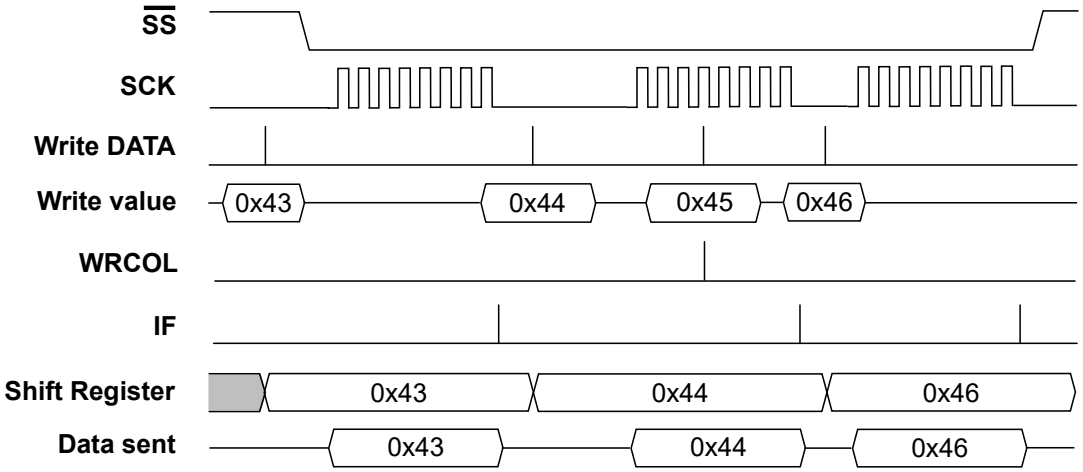
In Normal mode, the SPI peripheral will remain idle as long as the $\overline{SS}$ pin is driven high. In this state, the software may update the contents of the DATA register, but the data will not be shifted out by incoming clock pulses on the SCK pin until the $\overline{SS}$ pin is driven low. If the $\overline{SS}$ pin is driven low, the slave will start to shift out data on the first SCK clock pulse. When one byte has been completely shifted, the SPI Interrupt Flag (IF) in SPIn.INTFLAGS is set.

The user application may continue placing new data to be sent into the DATA register before reading the incoming data. New bytes to be sent cannot be written to the DATA register before the entire transfer has completed. A premature write will be ignored and the hardware will set the Write Collision flag (WRCOL in SPIn.INTFLAGS).

When the $\overline{SS}$ pin is driven high, the SPI logic is halted and the SPI slave will not receive any new data. Any partially received packet in the shift register will be lost.

Figure 25-2 shows a transmission sequence in Normal mode. Notice how the value 0x45 is written to the DATA register but never transmitted.

Figure 25-2. SPI Timing Diagram in Normal Mode (Buffer Mode Not Enabled)



The figure above shows three transfers and one write to the DATA register while the SPI is busy with a transfer. This write will be ignored and the Write Collision flag (WRCOL in SPIn.INTFLAGS) is set.

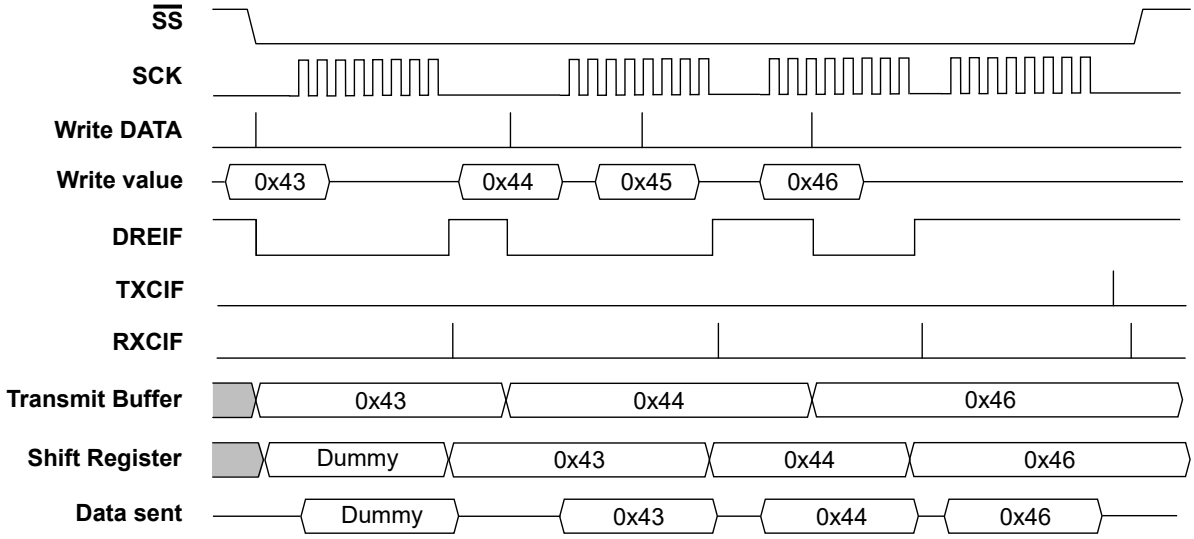
25.3.2.2.2 Buffer Mode

To avoid data collisions, the SPI peripheral can be configured in Buffered mode by writing a '1' to the Buffer Mode Enable bit in the Control B register (BUFEN in SPIn.CTRLB). In this mode, the SPI has additional interrupt flags and extra buffers. The extra buffers are shown in Figure 25-1. There are two different modes for the Buffer mode, selected with the Buffer mode Wait for Receive bit (BUFWR). The two different modes are described below with timing diagrams.

Slave Buffer Mode with Wait for Receive Bit Written to '0'

In Slave mode, if the Wait for Receive bit (BUFWR in SPIn.CRTLB) is written to '0', a dummy byte will be sent before the transmission of user data starts. Figure 25-3 shows a transmission sequence with this configuration. Notice how the value 0x45 is written to the Data register (SPIn.DATA) but never transmitted.

Figure 25-3. SPI Timing Diagram in Buffer Mode with BUFWR in SPIn.CTRLB Written to '0'



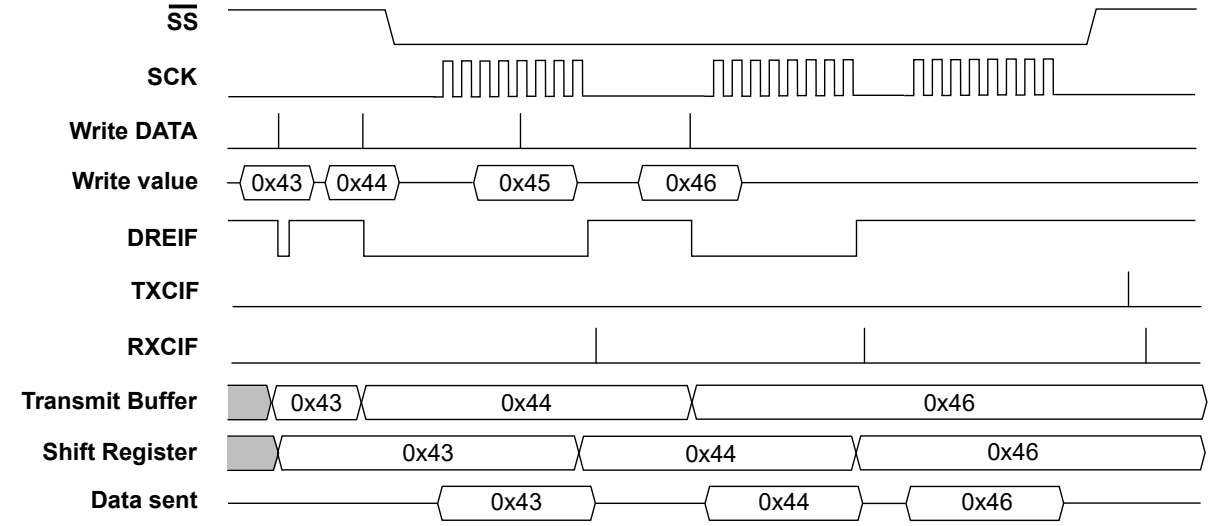
When the Wait for Receive bit (BUFWR in SPIn.CTRLB) is written to '0', all writes to the Data register (SPIn.DATA) goes to the Transmit Data Buffer register. The figure above shows that the value 0x43 is written to the Data register (SPIn.DATA), but it is not immediately transferred to the shift register so the first byte sent will be a dummy byte. The value of the dummy byte equals the values that was in the shift register at the time. After the first dummy transfer is completed the value 0x43 is transferred to the shift register. Then 0x44 is written to the Data register (SPIn.DATA) and goes to the Transmit Data Buffer register. A new transfer is started, and 0x43 will be sent. The value 0x45 is written to the Data register (SPIn.DATA), but the Transmit Data Buffer register is not updated since it is already full containing 0x44 and the Data Register Empty Interrupt Flag (DREIF in SPIn.INTFLAGS) is low. The value 0x45 will be lost. After the transfer, the value 0x44 is moved to the shift register. During the next transfer, 0x46 is written to the Data register (SPIn.DATA), and 0x44 is sent out. After the transfer is complete, 0x46 is copied into the shift register and sent out in the next transfer.

The DREIF goes low every time the Transmit Data Buffer register is written, and goes high after a transfer when the previous value in the Transmit Data Buffer register is copied into the shift register. The Receive Complete Interrupt Flag (RXCIF in SPIn.INTFLAGS) is set one cycle after the DREIF goes high. The Transfer Complete Interrupt Flag is set one cycle after the Receive Complete Interrupt Flag is set when both the value in the shift register and the Transmit Data Buffer register have been sent.

Slave Buffer Mode with Wait for Receive Bit Written to '1'

In Slave mode, if the Wait for Receive bit (BUFWR in SPIn.CRTLB) is written to '1', the transmission of user data starts as soon as the $\overline{SS}$ pin is driven low. Figure 25-4 shows a transmission sequence with this configuration. Notice how the value 0x45 is written to the Data register (SPIn.DATA) but never transmitted.

Figure 25-4. SPI Timing Diagram in Buffer Mode with CTRLB.BUFWR Written to '1'



All writes to the Data register (SPIn.DATA) go to the Transmit Data Buffer register. The figure above shows that the value 0x43 is written to the Data register (SPIn.DATA) and since the $\overline{SS}$ pin is high it is copied to the shift register in the next cycle. Then the next write (0x44) will go to the Transmit Data Buffer register. During the first transfer the value 0x43 will be shifted out. In the figure above, the value 0x45 is written to the Data register (SPIn.DATA), but the Transmit Data Buffer register is not updated since the DREIF is low. After the transfer is completed, the value 0x44 from the Transmit Data Buffer register is copied to the shift register. The value 0x46 is written to the Transmit Data Buffer register. During the next two transfers, 0x44 and 0x46 are shifted out. The flags behave identical to Buffer Mode Wait for Receive Bit (BUFWR in SPIn.CTRLB) set to '0'.

25.3.2.2.3 $\overline{SS}$ Pin Functionality in Slave Mode

The Slave Select ($\overline{SS}$) pin plays a central role in the operation of the SPI. Depending on the mode the SPI is in, and the configuration of this pin, it can be used to activate or deactivate devices. The $\overline{SS}$ pin is used as a Chip Select pin.

In Slave mode, $\overline{SS}$, MOSI, and SCK are always inputs. The behavior of the MISO pin depends on the configured data direction of the pin in the port peripheral and the value of $\overline{SS}$. When the $\overline{SS}$ pin is driven low, the SPI is activated and will respond to received SCK pulses by clocking data out on MISO, if the user has configured the data direction of the MISO pin as output. When the $\overline{SS}$ pin is driven high, the SPI is deactivated, meaning that it will not receive incoming data. If the MISO pin data direction is configured as output, the MISO pin will be tri-stated. Table 25-3 shows an overview of the $\overline{SS}$ pin functionality.

Table 25-3. Overview of the $\overline{SS}$ Pin Functionality

$\overline{SS}$ Configuration	$\overline{SS}$ Pin-Level	Description	MISO Pin Mode	
			Port Direction = Output	Port Direction = Input
Always Input	High	Slave deactivated (deselected)	Tri-stated	Input
	Low	Slave activated (selected)	Output	Input

Note: In Slave mode, the SPI state machine will be reset when the $\overline{SS}$ pin is driven high. If the $\overline{SS}$ pin is driven high during a transmission, the SPI will stop sending and receiving data immediately and both data received and data sent must be considered lost. As the $\overline{SS}$ pin is used to signal the start and end of a transfer, it is useful for achieving packet/byte synchronization and keeping the Slave bit counter synchronized with the master clock generator.

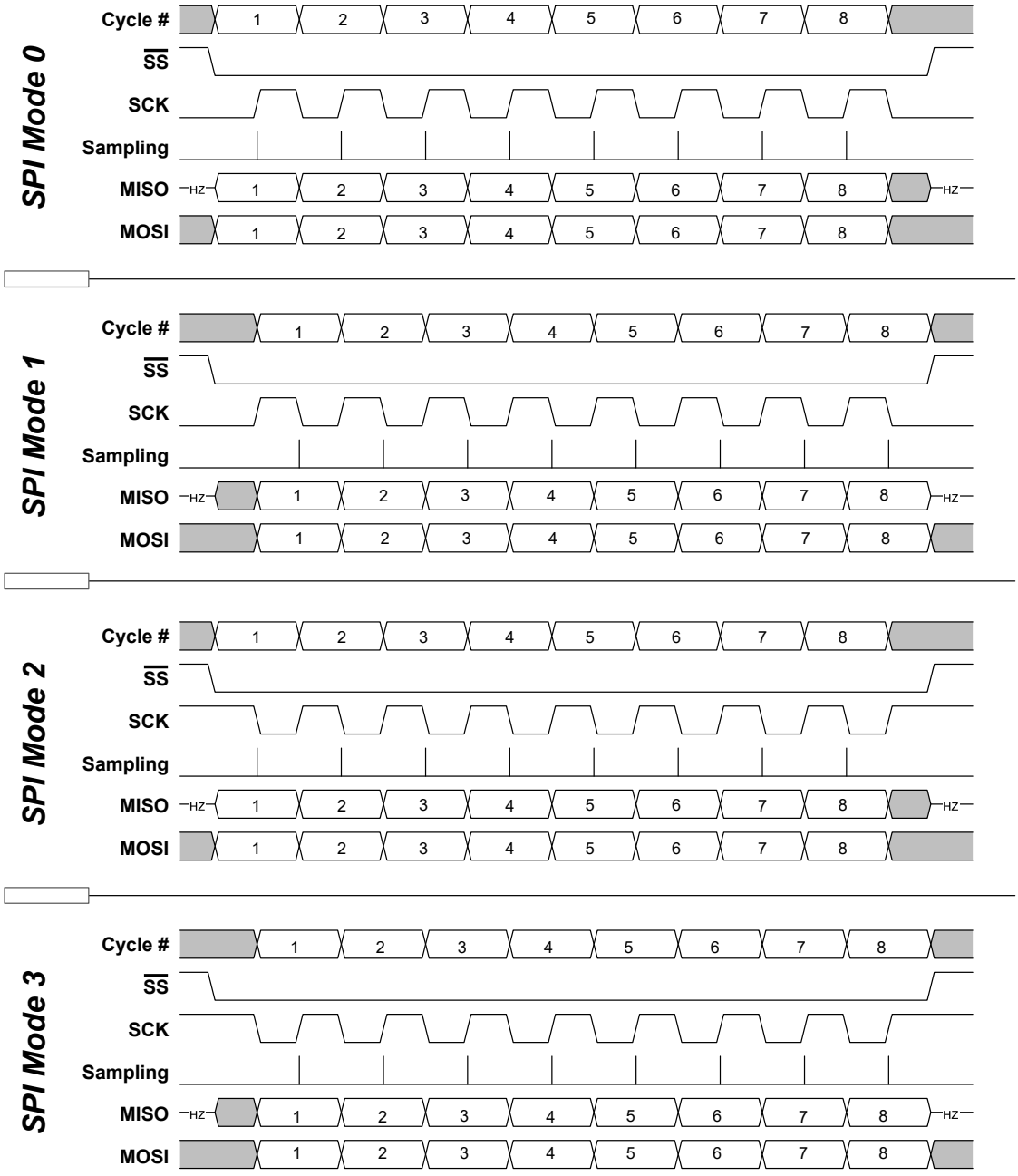
25.3.2.3 Data Modes

There are four combinations of SCK phase and polarity with respect to serial data. The desired combination is selected by writing to the MODE bits in the Control B register (SPIn.CTRLB).

The SPI data transfer formats are shown below. Data bits are shifted out and latched in on opposite edges of the SCK signal, ensuring sufficient time for data signals to stabilize.

The leading edge is the first clock edge of a clock cycle. The trailing edge is the last clock edge of a clock cycle.

Figure 25-5. SPI Data Transfer Modes



25.3.2.4 Events

The SPI can generate the following events:

Table 25-4. Event Generators in SPI

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Module	Event				
SPIn	SCK	SPI Master clock	Level	CLK_PER	Minimum two CLK_PER periods

The SPI has no event users.

Refer to the *Event System* chapter for more details regarding event types and Event System configuration.

25.3.2.5 Interrupts

Table 25-5. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions	
		Normal Mode	Buffer Mode
SPIn	SPI interrupt	• IF: Interrupt Flag interrupt • WRCOL: Write Collision interrupt	• SSI: Slave Select Trigger Interrupt • DRE: Data Register Empty interrupt • TXC: Transfer Complete interrupt • RXC: Receive Complete interrupt

When an interrupt condition occurs, the corresponding interrupt flag is set in the peripheral's Interrupt Flags (*peripheral*.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

25.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		DORD	MASTER	CLK2X		PRESC[1:0]		ENABLE
0x01	CTRLB	7:0	BUFEN	BUFWR				SSD	MODE[1:0]	
0x02	INTCTRL	7:0	RXCIE	TXCIE	DREIE	SSIE				IE
0x03	INTFLAGS	7:0	IF	WRCOL						
0x03	INTFLAGS	7:0	RXCIF	TXCIF	DREIF	SSIF				BUFOVF
0x04	DATA	7:0	DATA[7:0]							

25.5 Register Description

25.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		DORD	MASTER	CLK2X		PRESC[1:0]		ENABLE
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 6 – DORD Data Order

Value	Description
0	The MSb of the data word is transmitted first
1	The LSb of the data word is transmitted first

Bit 5 – MASTER Master/Slave Select
This bit selects the desired SPI mode.
If $\overline{SS}$ is configured as input and driven low while this bit is '1', then this bit is cleared and the IF in SPIn.INTFLAGS is set. The user has to write MASTER = 1 again to re-enable SPI Master mode.
This behavior is controlled by the Slave Select Disable (SSD) bit in SPIn.CTRLB.

Value	Description
0	SPI Slave mode selected
1	SPI Master mode selected

Bit 4 – CLK2X Clock Double
When this bit is written to '1' the SPI speed (SCK frequency, after internal prescaler) is doubled in Master mode.

Value	Description
0	SPI speed (SCK frequency) is not doubled
1	SPI speed (SCK frequency) is doubled in Master mode

Bits 2:1 – PRESC[1:0] Prescaler
This bit field controls the SPI clock rate configured in Master mode. These bits have no effect in Slave mode. The relationship between SCK and the peripheral clock frequency (f_{CLK_PER}) is shown below.
The output of the SPI prescaler can be doubled by writing the CLK2X bit to '1'.

Value	Name	Description
0x0	DIV4	CLK_PER/4
0x1	DIV16	CLK_PER/16
0x2	DIV64	CLK_PER/64
0x3	DIV128	CLK_PER/128

Bit 0 – ENABLE SPI Enable

Value	Description
0	SPI is disabled
1	SPI is enabled

25.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	BUFEN	BUFWR				SSD	MODE[1:0]	
Access	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

Bit 7 – BUFEN Buffer Mode Enable
Writing this bit to '1' enables Buffer mode. This will enable two receive buffers and one transmit buffer. Both will have separate interrupt flags, transmit complete and receive complete.

Bit 6 – BUFWR Buffer Mode Wait for Receive
When writing this bit to '0' the first data transferred will be a dummy sample.

Value	Description
0	One SPI transfer must be completed before the data are copied into the shift register.
1	If writing to the Data register when the SPI is enabled and $\overline{SS}$ is high, the first write will go directly to the shift register.

Bit 2 – SSD Slave Select Disable
If this bit is set when operating as SPI Master (MASTER = 1 in SPIn.CTRLA), $\overline{SS}$ does not disable Master mode.

Value	Description
0	Enable the Slave Select line when operating as SPI master
1	Disable the Slave Select line when operating as SPI master

Bits 1:0 – MODE[1:0] Mode
These bits select the Transfer mode. The four combinations of SCK phase and polarity with respect to the serial data are shown below. These bits decide whether the first edge of a clock cycle (leading edge) is rising or falling and whether data setup and sample occur on the leading or trailing edge. When the leading edge is rising, the SCK signal is low when idle, and when the leading edge is falling, the SCK signal is high when idle.

Value	Name	Description
0x0	0	Leading edge: Rising, sample Trailing edge: Falling, setup
0x1	1	Leading edge: Rising, setup Trailing edge: Falling, sample
0x2	2	Leading edge: Falling, sample Trailing edge: Rising, setup
0x3	3	Leading edge: Falling, setup Trailing edge: Rising, sample

25.5.3 Interrupt Control

Name: INTCTRL
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	DREIE	SSIE				IE
Access	R/W	R/W	R/W	R/W				R/W
Reset	0	0	0	0				0

Bit 7 – RXCIE Receive Complete Interrupt Enable
In Buffer mode, this bit enables the Receive Complete interrupt. The enabled interrupt will be triggered when the RXCIF in the SPIn.INTFLAGS register is set. In the Non-Buffer mode, this bit is '0'.

Bit 6 – TXCIE Transfer Complete Interrupt Enable
In Buffer mode, this bit enables the Transfer Complete interrupt. The enabled interrupt will be triggered when the TXCIF in the SPIn.INTFLAGS register is set. In the Non-Buffer mode, this bit is '0'.

Bit 5 – DREIE Data Register Empty Interrupt Enable
In Buffer mode, this bit enables the Data Register Empty interrupt. The enabled interrupt will be triggered when the DREIF in the SPIn.INTFLAGS register is set. In the Non-Buffer mode, this bit is '0'.

Bit 4 – SSIE Slave Select Trigger Interrupt Enable
In Buffer mode, this bit enables the Slave Select interrupt. The enabled interrupt will be triggered when the SSIF in the SPIn.INTFLAGS register is set. In the Non-Buffer mode, this bit is '0'.

Bit 0 – IE Interrupt Enable
This bit enables the SPI interrupt when the SPI is not in Buffer mode. The enabled interrupt will be triggered when RXCIF/IF is set in the SPIn.INTFLAGS register.

25.5.4 Interrupt Flags - Normal Mode

Name: INTFLAGS
Offset: 0x03
Reset: 0x00
Property: -

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

Bit 7 – IF Interrupt Flag
This flag is set when a serial transfer is complete, and one byte is completely shifted in/out of the SPIn.DATA register. If $\overline{SS}$ is configured as input and is driven low when the SPI is in Master mode, this will also set this flag. The IF is cleared by hardware when executing the corresponding interrupt vector. Alternatively, the IF can be cleared by first reading the SPIn.INTFLAGS register when IF is set, and then accessing the SPIn.DATA register.

Bit 6 – WRCOL Write Collision
The WRCOL flag is set if the SPIn.DATA register is written before a complete byte has been shifted out. This flag is cleared by first reading the SPIn.INTFLAGS register when WRCOL is set, and then accessing the SPIn.DATA register.

25.5.5 Interrupt Flags - Buffer Mode

Name: INTFLAGS
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RXCIF	TXCIF	DREIF	SSIF				BUFOVF
Access	R/W	R/W	R/W	R/W				R/W
Reset	0	0	0	0				0

Bit 7 – RXCIF Receive Complete Interrupt Flag
This flag is set when there are unread data in the Receive Data Buffer register and cleared when the Receive Data Buffer register is empty (that is, it does not contain any unread data).
When interrupt-driven data reception is used, the Receive Complete Interrupt routine must read the received data from the DATA register in order to clear RXCIF. If not, a new interrupt will occur directly after the return from the current interrupt. This flag can also be cleared by writing a '1' to its bit location.

Bit 6 – TXCIF Transfer Complete Interrupt Flag
This flag is set when all the data in the Transmit shift register has been shifted out, and there is no new data in the transmit buffer (SPIn.DATA). The flag is cleared by writing a '1' to its bit location.

Bit 5 – DREIF Data Register Empty Interrupt Flag
This flag indicates whether the Transmit Data Buffer register is ready to receive new data. The flag is '1' when the transmit buffer is empty and '0' when the transmit buffer contains data to be transmitted that has not yet been moved into the shift register. The DREIF is cleared after a Reset to indicate that the transmitter is ready.
The DREIF is cleared by writing to DATA. When interrupt-driven data transmission is used, the Data Register Empty Interrupt routine must either write new data to DATA in order to clear DREIF or disable the Data Register Empty interrupt. If not, a new interrupt will occur directly after the return from the current interrupt.

Bit 4 – SSIF Slave Select Trigger Interrupt Flag
This flag indicates that the SPI has been in Master mode and the $\overline{SS}$ pin has been pulled low externally, so the SPI is now working in Slave mode. The flag will only be set if the Slave Select Disable (SSD) bit is not '1'. The flag is cleared by writing a '1' to its bit location.

Bit 0 – BUFOVF Buffer Overflow
This flag indicates data loss due to a Receive Data Buffer full condition. This flag is set if a Buffer Overflow condition is detected. A Buffer Overflow occurs when the receive buffer is full (two bytes), and a third byte has been received in the shift register. If there is no transmit data, the Buffer Overflow will not be set before the start of a new serial transfer. This flag is cleared when the DATA register is read, or by writing a '1' to its bit location.

25.5.6 Data

Name: DATA
Offset: 0x04
Reset: 0x00
Property: -

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] SPI Data
The DATA register is used for sending and receiving data. Writing to the register initiates the data transmission when in Master mode, while preparing data for sending in Slave mode. The byte written to the register shifts out on the SPI output line when a transaction is initiated.
The SPIn.DATA register is not a physical register. Depending on what mode is configured, it is mapped to other registers as described below.

- Normal mode:
  - Writing the DATA register will write the shift register
  - Reading from DATA will read from the Receive Data register
- Buffer mode:
  - Writing the DATA register will write to the Transmit Data Buffer register.
  - Reading from DATA will read from the Receive Data Buffer register. The contents of the Receive Data register will then be moved to the Receive Data Buffer register.

26. TWI - Two-Wire Interface

26.1 Features

- Two-Wire Communication Interface
- Philips I^2C Compatible
  - Standard mode
  - Fast mode
  - Fast mode Plus
- System Management Bus (SMBus) 2.0 Compatible
  - Support arbitration between Start/repeated Start and data bit
  - Slave arbitration allows support for the Address Resolution Protocol (ARP)
  - Configurable SMBus Layer 1 time-outs in hardware
- Independent Master and Slave Operation
  - Combined (same pins)
  - Single or multi-master bus operation with full arbitration support
- Hardware Support for Slave Address Match
  - Operates in all Sleep modes
  - 7-bit address recognition
  - General call address recognition
  - Support for address range masking or secondary address match
- Input Filter for Bus Noise Suppression
- Smart Mode Support

26.2 Overview

The Two-Wire Interface (TWI) is a bidirectional, two-wire communication interface (bus) with a Serial Data Line (SDA) and a Serial Clock Line (SCL).

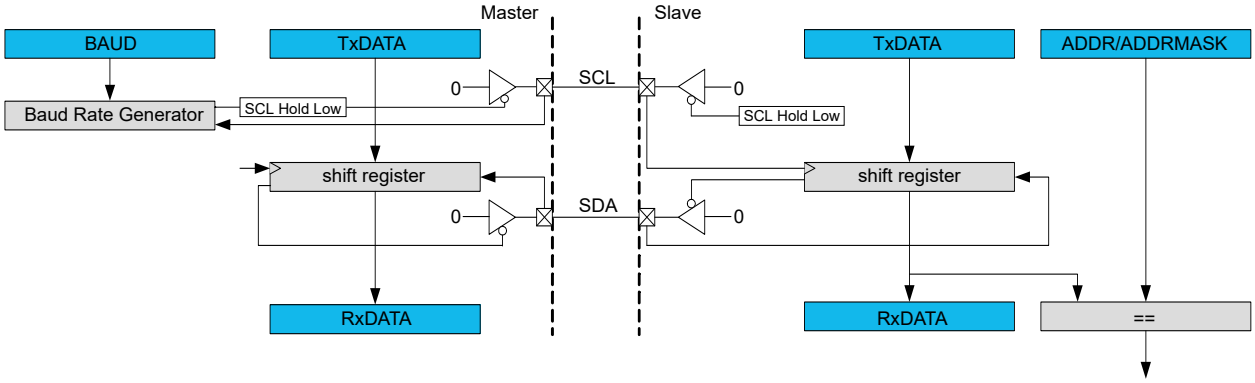
The TWI bus connects one or several slave devices to one or several master devices. Any device connected to the bus can act as a master, a slave, or both. The master generates the SCL by using a Baud Rate Generator (BRG) and initiates data transactions by addressing one slave and telling whether it wants to transmit or receive data. The BRG is capable of generating the Standard mode (Sm) and Fast mode (Fm, Fm+) bus frequencies from 100 kHz up to 1 MHz.

The TWI will detect Start and Stop conditions, bus collisions and bus errors. Arbitration lost, errors, collision, and clock hold are also detected and indicated in separate status flags available in both Master and Slave modes.

The TWI supports multi-master bus operation and arbitration. An arbitration scheme handles the case where more than one master tries to transmit data at the same time. The TWI also supports Smart mode, which can auto-trigger operations and thus reduce software complexity. The TWI supports Quick Command mode where the master can address a slave without exchanging data.

26.2.1 Block Diagram

Figure 26-1. TWI Block Diagram



26.2.2 Signal Description

Signal	Description	Type
SCL	Serial Clock Line	Digital I/O
SDA	Serial Data Line	Digital I/O

26.3 Functional Description

26.3.1 General TWI Bus Concepts

The TWI provides a simple, bidirectional, two-wire communication bus consisting of:

- Serial Data Line (SDA) for packet transfer
- Serial Clock Line (SCL) for the bus clock

The two lines are open-collector lines (wired-AND).

The TWI bus topology is a simple and efficient method of interconnecting multiple devices on a serial bus. A device connected to the bus can be a master or a slave. Only master devices can control the bus and the bus communication.

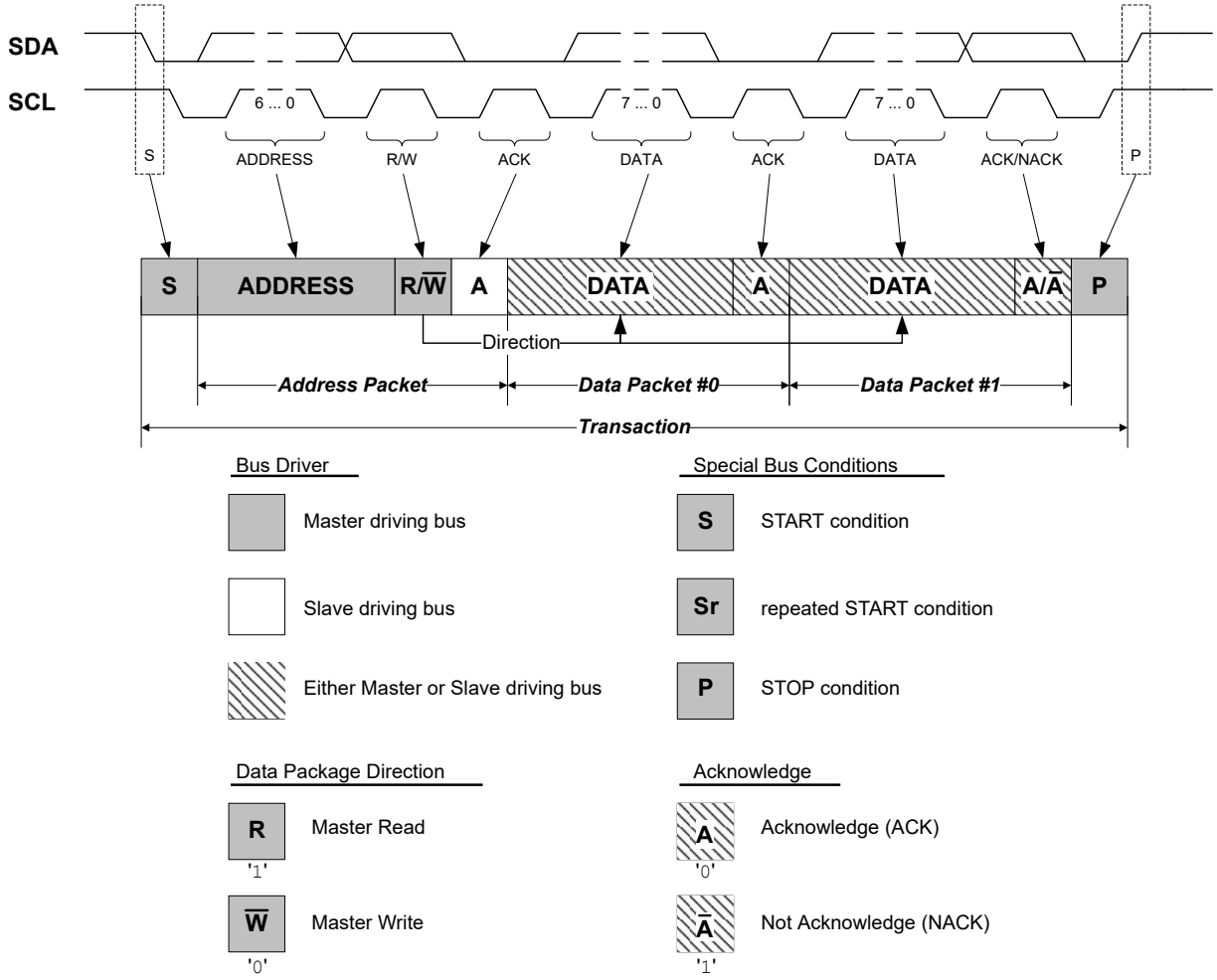
A unique address is assigned to each slave device connected to the bus, and the master will use it to control the slave and initiate a transaction. Several masters can be connected to the same bus. This is called a multi-master environment. An arbitration mechanism is provided for resolving bus ownership among masters, since only one master device may own the bus at any given time.

A master indicates the start of a transaction by issuing a Start condition (S) on the bus. The master provides the clock signal for the transaction. An address packet with a 7-bit slave address (ADDRESS) and a direction bit, representing whether the master wishes to read or write data (R/$\overline{W}$), are then sent.

The addressed I^2C slave 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 1-bit reply indicating whether the data was acknowledged or not by the receiver.

After all the data packets (DATA) are transferred, the master issues a Stop condition (P) on the bus to end the transaction.

Figure 26-2. Basic TWI Transaction Diagram Topology for a 7-bit Address Bus



26.3.2 TWI Basic Operation

26.3.2.1 Initialization

If used, the following bits must be configured before enabling the TWI device:

- The SDA Setup Time (SDASETUP) bit from the Control A (TWIn.CTRLA) register
- The SDA Hold Time (SDAHOLD) bit field from the Control A (TWIn.CTRLA) register
- The FM Plus Enable (FMPEN) bit from the Control A (TWIn.CTRLA) register

26.3.2.1.1 Master Initialization

The Master Baud Rate (TWIn.MBAUD) register must be written to a value that will result in a valid TWI bus clock frequency. Writing a '1' to the Enable TWI Master (ENABLE) bit in the Master Control A (TWIn.MCTRLA) register will start the TWI master. The Bus State (BUSSTATE) bit field from the Master Status (TWIn.MSTATUS) register must be set to 0x1, to force the bus state to Idle.

26.3.2.1.2 Slave Initialization

The address of the slave must be written in the Slave Address (TWIn.SADDR) register. Writing a '1' to the Enable TWI Slave (ENABLE) bit in the Slave Control A (TWIn.SCTRLA) register will start the TWI slave. The slave device will wait for a master device to issue a Start condition and the matching slave address.

26.3.2.2 TWI Master Operation

The TWI master is byte-oriented, with an optional interrupt after each byte. There are separate interrupt flags for the master write and read operation. Interrupt flags can also be used for polled operation. There are dedicated status flags for indicating ACK/NACK received, bus error, arbitration lost, clock hold, and bus state.

When an interrupt flag is set to '1', the SCL is forced low. This will give the master time to respond or handle any data, and will, in most cases, require software interaction. Clearing the interrupt flags releases the SCL. The number of interrupts generated is kept to a minimum by an automatic handling of most conditions.

26.3.2.2.1 Clock Generation

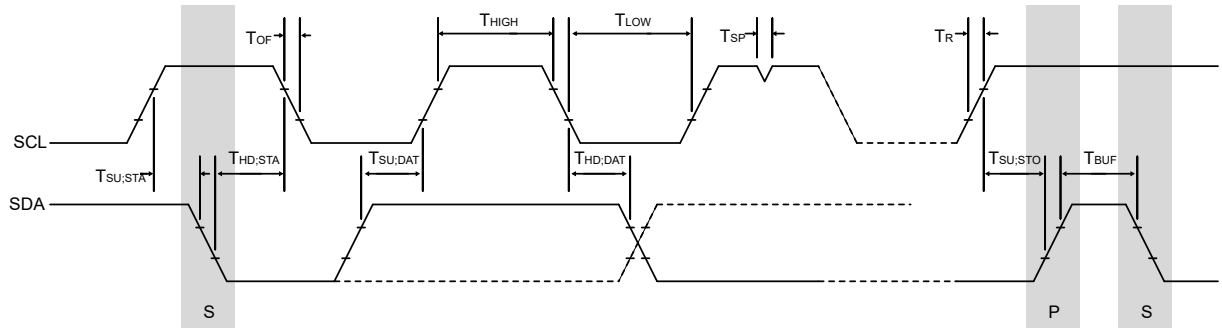
The TWI supports several transmission modes with different frequency limitations:

- Standard mode (Sm) up to 100 kHz
- Fast mode (Fm) up to 400 kHz
- Fast mode Plus (Fm+) up to 1 MHz

The Master Baud Rate (TWIn.MBAUD) register must be written to a value that will result in a TWI bus clock frequency equal or less than those frequency limits, depending on the transmission mode.

The low (T_{LOW}) and high (T_{HIGH}) times are determined by the Master Baud Rate (TWIn.MBAUD) register, while the rise (T_R) and fall (T_{OF}) times are determined by the bus topology.

Figure 26-3. SCL Timing



- T_{LOW} is the low period of SCL clock
- T_{HIGH} is the high period of SCL clock
- T_R is determined by the bus impedance; for internal pull-ups. Refer to *Electrical Characteristics* for details.
- T_{OF} is determined by the open-drain current limit and bus impedance. Refer to *Electrical Characteristics* for details.

Properties of the SCL Clock

The SCL frequency is given by:

$$f_{\mathrm{SCL}} = \frac{1}{T_{\mathrm{LOW}} + T_{\mathrm{HIGH}} + T_{\mathrm{OF}} + T_{\mathrm{R}}}[\mathrm{Hz}]$$

The SCL clock is designed to have a 50/50 duty cycle, where T_{OF} is considered a part of T_{LOW}. T_{HIGH} will not start until a high state of SCL has been detected.The BAUD bit field in the TWIn.MBAUD register and the SCL frequency are related by the following formula:

$$f_{SCL} = \frac{f_{CLK\_PER}}{10 + 2 \times BAUD + f_{CLK\_PER} \times T_R} \qquad (1)$$

Equation 1 can be transformed to express BAUD:

$$BAUD = \frac{f_{CLK\_PER}}{2 \times f_{SCL}} - \left(5 + \frac{f_{CLK\_PER} \times T_R}{2}\right) \qquad (2)$$

Calculation of the BAUD Value

To ensure operation within the specifications of the desired speed mode (Sm, Fm, Fm+), follow these steps:

1. Calculate a value for the BAUD bit field using equation 2
2. Calculate T_{LOW} using the BAUD value from step 1:

$$T_{LOW} = \frac{BAUD + 5}{f_{CLK\_PER}} - T_{OF} \qquad (3)$$

3. Check if your T_{LOW} from equation 3 is above the specified minimum of the desired mode (T_{LOW_Sm}= 4700 ns, T_{LOW_Fm}= 1300 ns, T_{LOW_Fm+}= 500 ns)
   - If the calculated T_{LOW} is above the limit, use the BAUD value from equation 2
   - If the limit is not met, calculate a new BAUD value using equation 4 below, where T_{LOW_mode} is either T_{LOW_Sm}, T_{LOW_Fm}, or T_{LOW_Fm+} from the mode specifications:

$$BAUD = f_{CLK\_PER} \times (T_{LOW\_mode} + T_{OF}) - 5 \qquad (4)$$

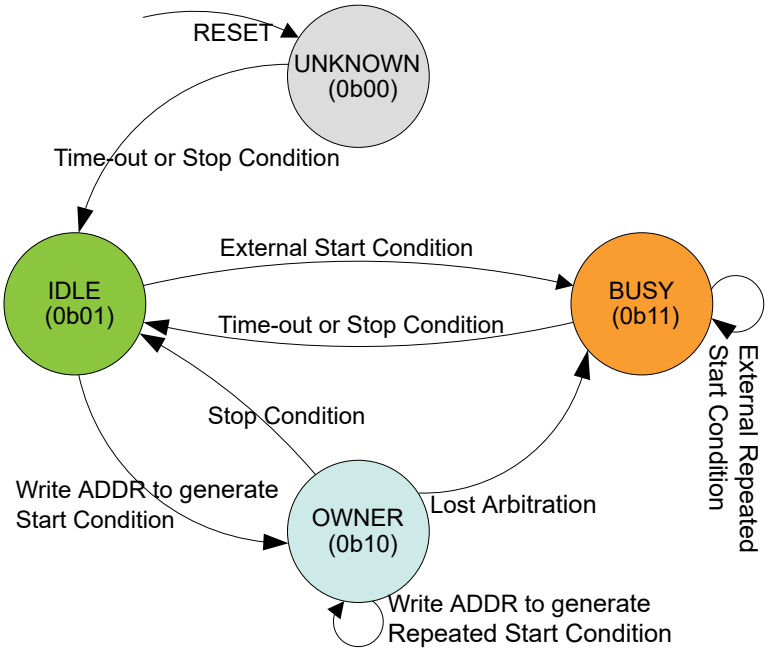
26.3.2.2.2 TWI Bus State Logic

The bus state logic continuously monitors the activity on the TWI bus when the master is enabled. It continues to operate in all Sleep modes, including Power-Down.

The bus state logic includes Start and Stop condition detectors, collision detection, inactive bus time-out detection, and a bit counter. These are used to determine the bus state. The software can get the current bus state by reading the Bus State (BUSSTATE) bit field in the Master Status (TWIn.MSTATUS) register.

The bus state can be Unknown, Idle, Busy or Owner, and it is determined according to the state diagram shown below.

Figure 26-4. Bus State Diagram



1. **Unknown**: The bus state machine is active when the TWI master is enabled. After the TWI master has been enabled, the bus state is Unknown. The bus state will also be set to Unknown after a System Reset is performed or after the TWI master is disabled.
2. **Idle**: The bus state machine can be forced to enter the Idle state by writing `0x1` to the Bus State (BUSSTATE) bit field. The bus state logic cannot be forced into any other state. If no state is set by the application software, the bus state will become Idle when the first Stop condition is detected. If the Inactive Bus Time-Out (TIMEOUT) bit field from the Master Control A (TWIn.MCTRLA) register is configured to a nonzero value, the bus state will change to Idle on the occurrence of a time-out. When the bus is Idle, it is ready for a new transaction.
3. **Busy**: If a Start condition, generated externally, is detected when the bus is Idle, the bus state becomes Busy. The bus state changes back to Idle when a Stop condition is detected or when a time-out, if configured, is set.
4. **Owner**: If a Start condition is generated internally when the bus is Idle, the bus state becomes Owner. If the complete transaction is performed without interference, the master issues a Stop condition and the bus state

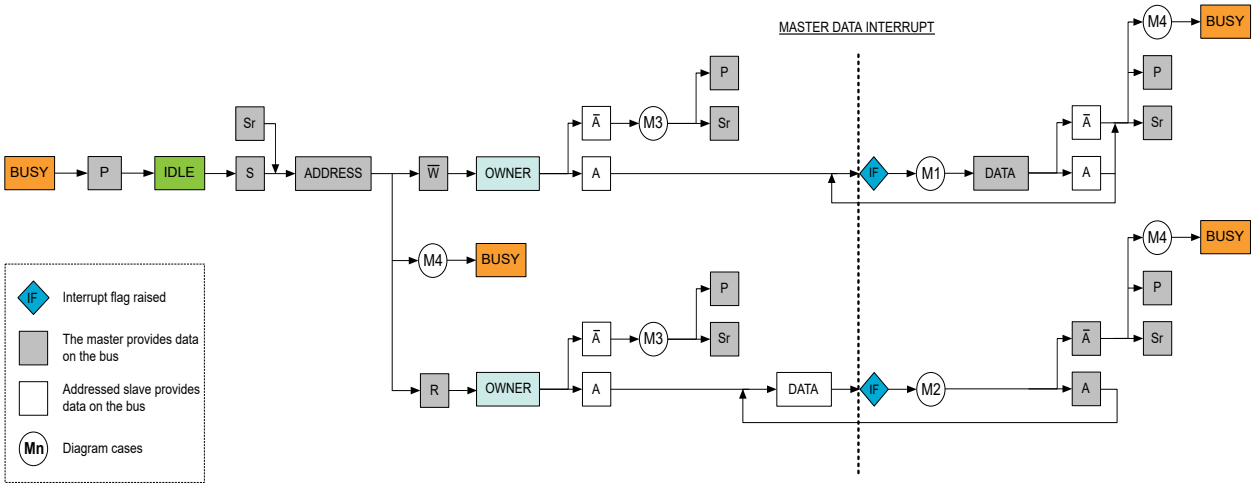
changes back to Idle. If a collision is detected, the arbitration is lost and the bus state becomes Busy until a Stop condition is detected.

26.3.2.2.3 Transmitting Address Packets

The master starts performing a bus transaction when the Master Address (TWIn.MADDR) register is written with the slave address and the R/$\overline{W}$ direction bit. The value of the MADDR register is then copied in the Master Data (TWIn.MDATA) register. If the bus state is Busy, the TWI master will wait until the bus state becomes Idle before issuing the Start condition. The TWI will issue a Start condition, and the shift register performs a byte transmit operation on the bus.

Depending on the arbitration and the R/$\overline{W}$ direction bit, one of four cases (M1 to M4) arises after the transmission of the address packet.

Figure 26-5. TWI Master Operation



Case M1: Address Packet Transmit Complete - Direction Bit Set to '0'

If a slave device responds to the address packet with an ACK, the Write Interrupt Flag (WIF) is set to '1', the Received Acknowledge (RXACK) flag is set to '0', and the Clock Hold (CLKHOLD) flag is set to '1'. The WIF, RXACK and CLKHOLD flags are located in the Master Status (TWIn.MSTATUS) register.

The clock hold is active at this point, forcing the SCL low. This will stretch the low period of the clock to slow down the overall clock frequency, forcing delays required to process the data and preventing further activity on the bus.

The software can prepare to:

- Transmit data packets to the slave

Case M2: Address Packet Transmit Complete - Direction Bit Set to '1'

If a slave device responds to the address packet with an ACK, the RXACK flag is set to '0', and the slave can start sending data to the master without any delays because the slave owns the bus at this moment. The clock hold is active at this point, forcing the SCL low.

The software can prepare to:

- Read the received data packet from the slave

Case M3: Address Packet Transmit Complete - Address not Acknowledged by Slave

If no slave device responds to the address packet, the WIF and the RXACK flags will be set to '1'. The clock hold is active at this point, forcing the SCL low.

The missing ACK response can indicate that the I^2C slave is busy with other tasks, or it is in a Sleep mode, and it is not able to respond.

The software can prepare to take one of the following actions:

- Retransmit the address packet
- Complete the transaction by issuing a Stop condition in the Command (MCMD) bit field from the Master Control B (TWIn.MCTRLB) register, which is the recommended action

Case M4: Arbitration Lost or Bus Error

If arbitration is lost, both the WIF and the Arbitration Lost (ARBLOST) flags in the Master Status (TWIn.MSTATUS) register are set to '1'. The SDA is disabled and the SCL is released. The bus state changes to Busy, and the master is no longer allowed to perform any operation on the bus until the bus state is changed back to Idle.

A bus error will behave similarly to the arbitration lost condition. In this case, the Bus Error (BUSERR) flag in the Master Status (TWIn.MSTATUS) register is set to '1', in addition to the WIF and ARBLOST flags.

The software can prepare to:

- Abort the operation and wait until the bus state changes to Idle by reading the Bus State (BUSSTATE) bit field in the Master Status (TWIn.MSTATUS) register

26.3.2.2.4 Transmitting Data Packets

Assuming the above M1 case, the TWI master can start transmitting data by writing to the Master Data (TWIn.MDATA) register, which will also clear the Write Interrupt Flag (WIF). During the data transfer, the master is continuously monitoring the bus for collisions and errors. The WIF flag will be set to '1' after the data packet transfer has been completed.

If the transmission is successful and the master receives an ACK bit from the slave, the Received Acknowledge (RXACK) flag will be set to '0', meaning that the slave is ready to receive new data packets.

The software can prepare to take one of the following actions:

- Transmit a new data packet
- Transmit a new address packet
- Complete the transaction by issuing a Stop condition in the Command (MCMD) bit field from the Master Control B (TWIn.MCTRLB) register

If the transmission is successful and the master receives a NACK bit from the slave, the RXACK flag will be set to '1', meaning that the slave is not able to or does not need to receive more data.

The software can prepare to take one of the following actions:

- Transmit a new address packet
- Complete the transaction by issuing a Stop condition in the Command (MCMD) bit field from the Master Control B (TWIn.MCTRLB) register

The RXACK status is valid only if the WIF flag is set to '1' and the Arbitration Lost (ARBLOST) and Bus Error (BUSERR) flags are set to '0'.

The transmission can be unsuccessful if a collision is detected. Then, the master will lose arbitration, the Arbitration Lost (ARBLOST) flag will be set to '1', and the bus state changes to Busy. An arbitration lost during the sending of the data packet is treated the same way as the above M4 case.

The WIF, ARBLOST, BUSERR and RXACK flags are all located in the Master Status (TWIn.MSTATUS) register.

26.3.2.2.5 Receiving Data Packets

Assuming the M2 case above, the clock is released for one byte, allowing the slave to put one byte of data on the bus. The master will receive one byte of data from the slave, and the Read Interrupt Flag (RIF) will be set to '1' together with the Clock Hold (CLKHOLD) flag. The action selected by the Acknowledge Action (ACKACT) bit in the Master Control B (TWIn.MCTRLB) register is automatically sent on the bus when a command is written to the Command (MCMD) bit field in the TWIn.MCTRLB register.

The software can prepare to take one of the following actions:

- Respond with an ACK by writing '0' to the ACKACT bit in the TWIn.MCTRLB register and prepare to receive a new data packet
- Respond with a NACK by writing '1' to the ACKACT bit and then transmit a new address packet
- Respond with a NACK by writing '1' to the ACKACT bit and then complete the transaction by issuing a Stop condition in the MCMD bit field from the TWIn.MCTRLB register

A NACK response might not be successfully executed, as arbitration can be lost during the transmission. If a collision is detected, the master loses arbitration, and the Arbitration Lost (ARBLOST) flag is set to '1' and the bus state changes to Busy. The Master Write Interrupt Flag (WIF) is set if the arbitration was lost when sending a NACK or a

bus error occurred during the procedure. An arbitration lost during the sending of the data packet is treated in the same way as the above M4 case.

The RIF, CLKHOLD, ARBLOST and WIF flags are all located in the Master Status (TWIn.MSTATUS) register.

Note: The RIF and WIF flags are mutually exclusive and cannot be set simultaneously.

26.3.2.3 TWI Slave Operation

The TWI slave is byte-oriented with optional interrupts after each byte. There are separate interrupt flags for the slave data and for address/Stop recognition. Interrupt flags can also be used for polled operation. There are dedicated status flags for indicating ACK/NACK received, clock hold, collision, bus error, and R/$\overline{W}$ direction bit.

When an interrupt flag is set to '1', the SCL is forced low. This will give the slave time to respond or handle any data, and will, in most cases, require software interaction. The number of interrupts generated is kept to a minimum by automatic handling of most conditions.

The Address Recognition Mode (PMEN) bit in the Slave Control A (TWIn.SCTRLA) register can be configured to allow the slave to respond to all received addresses.

26.3.2.3.1 Receiving Address Packets

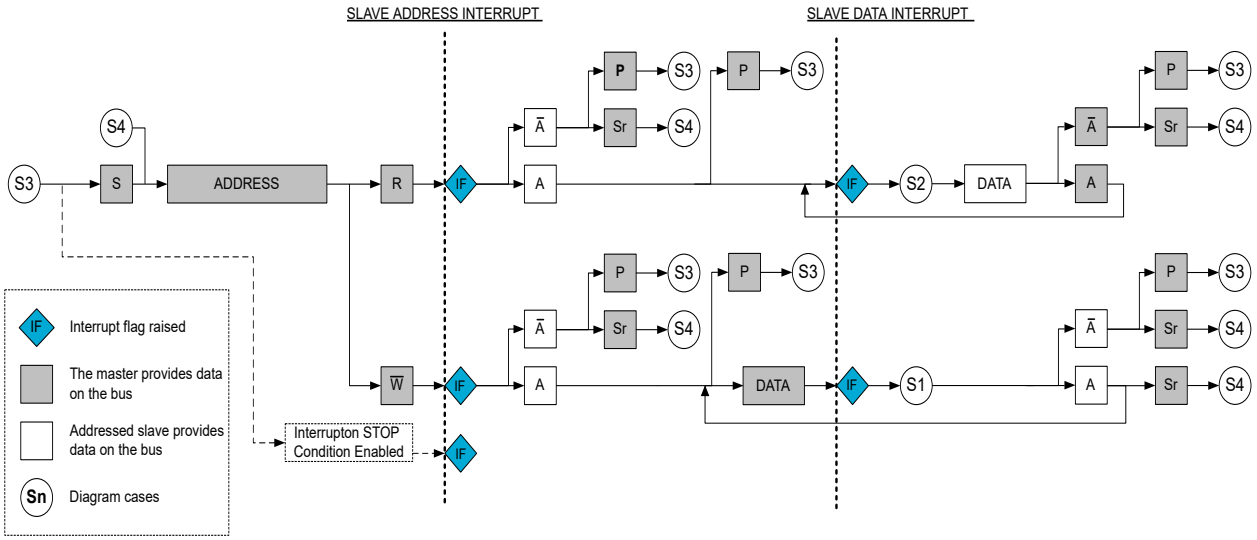
When the TWI is configured as a slave, it will wait for a Start condition to be detected. When this happens, the successive address packet will be received and checked by the address match logic. The slave will ACK a correct address and store the address in the Slave Data (TWIn.SDATA) register. If the received address is not a match, the slave will not acknowledge or store the address, but wait for a new Start condition.

The Address or Stop Interrupt Flag (APIF) in the Slave Status (TWIn.SSTATUS) register is set to '1' when a Start condition is succeeded by one of the following:

- A valid address match with the address stored in the Address (ADDR[7:1]) bit field in the Slave Address (TWIn.SADDR) register
- The General Call Address 0x00 and the Address (ADDR[0]) bit in the Slave Address (TWIn.SADDR) register are set to '1'
- A valid address match with the secondary address stored in the Address Mask (ADDRMASK) bit field and the Address Mask Enable (ADDREN) bit is set to '1' in the Slave Address Mask (TWIn.SADDRMASK) register
- Any address if the Address Recognition Mode (PMEN) bit in the Slave Control A (TWIn.SCTRLA) register is set to '1'

Depending on the Read/Write Direction (DIR) bit in the Slave Status (TWIn.SSTATUS) register and the bus condition, one of four distinct cases (S1 to S4) arises after the reception of the address packet.

Figure 26-6. TWI Slave Operation



Case S1: Address Packet Accepted - Direction Bit Set to '0'
If an ACK is sent by the slave after the address packet is received and the Read/Write Direction (DIR) bit in the Slave Status (TWIn.SSTATUS) register is set to '0', the master indicates a write operation.

The clock hold is active at this point, forcing the SCL low. This will stretch the low period of the clock to slow down the overall clock frequency, forcing delays required to process the data and preventing further activity on the bus.

The software can prepare to:

- Read the received data packet from the master

Case S2: Address Packet Accepted - Direction Bit Set to '1'
If an ACK is sent by the slave after the address packet is received and the DIR bit is set to '1', the master indicates a read operation, and the Data Interrupt Flag (DIF) in the Slave Status (TWIn.SSTATUS) register will be set to '1'.

The clock hold is active at this point, forcing the SCL low.

The software can prepare to:

- Transmit data packets to the master

Case S3: Stop Condition Received
When the Stop condition is received, the Address or Stop (AP) flag will be set to '0', indicating that a Stop condition, and not an address match, activated the Address or Stop Interrupt Flag (APIF).

The AP and APIF flags are located in the Slave Status (TWIn.SSTATUS) register.

The software can prepare to:

- Wait until a new address packet will be addressed to it

Case S4: Collision
If the slave is not able to send a high-level data bit or a NACK, the Collision (COLL) bit in the Slave Status (TWIn.SSTATUS) register is set to '1'. The slave will commence its operation as normal, except no low values will be shifted out on the SDA. The data and acknowledge output from the slave logic will be disabled. The clock hold is released. A Start or repeated Start condition will be accepted.

The COLL bit is intended for systems where the Address Resolution Protocol (ARP) is employed. A detected collision in non-ARP situations indicates that there has been a protocol violation and must be treated as a bus error.

26.3.2.3.2 Receiving Data Packets

Assuming the above S1 case, the slave must be ready to receive data. When a data packet is received, the Data Interrupt Flag (DIF) in the Slave Status (TWIn.SSTATUS) register is set to '1'. The action selected by the Acknowledge Action (ACKACT) bit in the Slave Control B (TWIn.SCTRLB) register is automatically sent on the bus when a command is written to the Command (SCMD) bit field in the TWIn.SCTRLB register.

The software can prepare to take one of the following actions:

- Respond with an ACK by writing '0' to the ACKACT bit in the TWIn.SCTRLB register, indicating that the slave is ready to receive more data
- Respond with a NACK by writing '1' to the ACKACT bit, indicating that the slave cannot receive any more data and the master must issue a Stop or repeated Start condition

26.3.2.3.3 Transmitting Data Packets

Assuming the above S2 case, the slave can start transmitting data by writing to the Slave Data (TWIn.SDATA) register. When a data packet transmission is completed, the Data Interrupt Flag (DIF) in the Slave Status (TWIn.SSTATUS) register is set to '1'.

The software can prepare to take one of the following actions:

- Check if the master responded with an ACK by reading the Received Acknowledge (RXACK) bit from the Slave Status (TWIn.SSTATUS) register and start transmitting new data packets
- Check if the master responded with a NACK by reading the RXACK and stop transmitting data packets. The master must send a Stop or repeated Start condition after the NACK.

26.3.3 Additional Features

26.3.3.1 SMBus

If the TWI is used in an SMBus environment, the Inactive Bus Time-Out (TIMEOUT) bit field from the Master Control A (TWIn.MCTRLA) register must be configured. It is recommended to write to the Master Baud Rate (TWIn.MBAUD) register before setting the time-out because it is dependent on the baud rate setting.

A frequency of 100 kHz can be used for the SMBus environment. For the Standard mode (Sm) and Fast mode (Fm), the operating frequency has slew rate limited output, while for the Fast mode Plus (Fm+), it has x10 output drive strength.

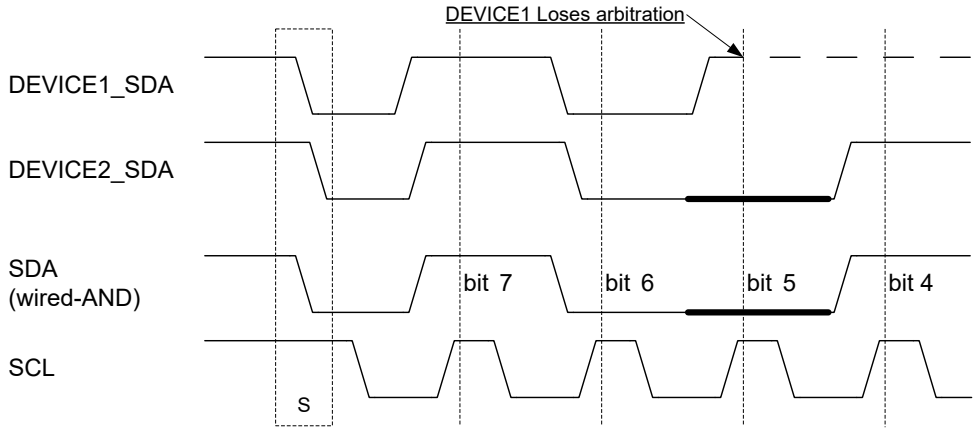
The TWI also allows for an SMBus compatible SDA hold time configured in the SDA Hold Time (SDAHOLD) bit field from the Control A (TWIn.CTRLA) register.

26.3.3.2 Multi Master

A master can start a bus transaction only if it has detected that the bus is in the Idle state. As the TWI bus is a multi-master bus, more devices may try to initiate a transaction at the same time. This results in multiple masters owning the bus simultaneously. The TWI solves this problem by using an arbitration scheme where the master loses control of the bus if it is not able to transmit a high-level data bit on the SDA and the Bus State (BUSSTATE) bit field from the Master Status (TWIn.MSTATUS) register will be changed to Busy. The masters that lose the arbitration must wait until the bus becomes Idle before attempting to reacquire the bus ownership.

Both devices can issue a Start condition, but DEVICE1 loses arbitration when attempting to transmit a high level (bit 5) while DEVICE2 is transmitting a low level.

Figure 26-7. TWI Arbitration



26.3.3.3 Smart Mode

The TWI interface has a Smart mode that simplifies the application code and minimizes the user interaction needed to adhere to the I^2C protocol.

For the TWI Master, the Smart mode will automatically send the ACK action as soon as the Master Data (TWIn.MDATA) register is read. This feature is only active when the Acknowledge Action (ACKACT) bit in the Master Control B (TWIn.MCTRLB) register is set to ACK. If the ACKACT bit is set to NACK, the TWI Master will not generate a NACK after the MDATA register is read. This feature is enabled when the Smart Mode Enable (SMEN) bit in the Master Control A (TWIn.MCTRLA) register is set to '1'.

For the TWI Slave, the Smart mode will automatically send the ACK action as soon as the Slave Data (TWIn.SDATA) register is read. The Smart mode will automatically set the Data Interrupt Flag (DIF) to '0' in the Slave Status (TWIn.SSTATUS) register if the TWIn.SDATA register is read or written. This feature is enabled when the Smart Mode Enable (SMEN) bit in the Slave Control A (TWIn.SCTRLA) register is set to '1'.

26.3.3.4 Quick Command Mode

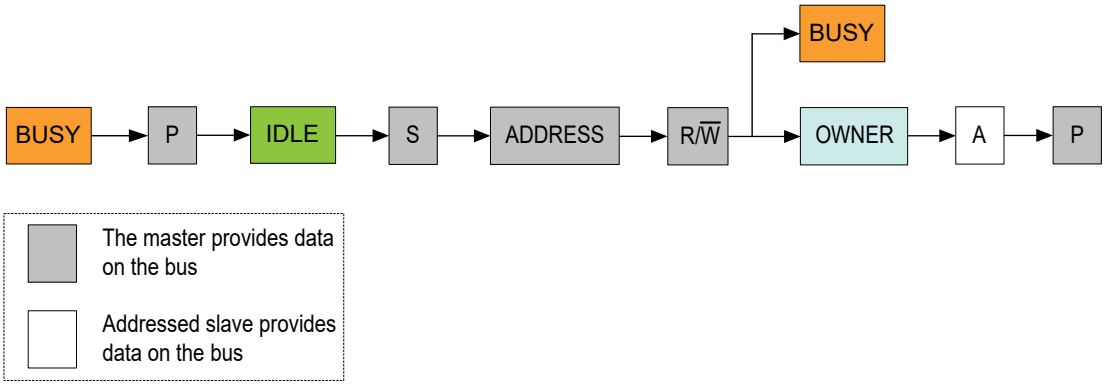
With Quick Command mode, the R/$\overline{W}$ bit from the address packet denotes the command. This mode is enabled by writing '1' to the Quick Command Enable (QCEN) bit in the Master Control A (TWIn.MCTRLA) register. There are no data sent or received.

The Quick Command mode is SMBus specific, where the R/$\overline{W}$ bit can be used to turn a device function on/off or to enable/disable a low-power Standby mode. This mode can be enabled to auto-trigger operations and reduce the software complexity.

After the master receives an ACK from the slave, either the Read Interrupt Flag (RIF) or Write Interrupt Flag (WIF) will be set, depending on the value of the R/$\overline{W}$ bit. When either the RIF or WIF flag is set after issuing a Quick Command, the TWI will accept a Stop command by writing the Command (MCMD) bit field in the Master Control B (TWIn.MCTRLB) register.

The RIF and WIF flags, together with the value of the last Received Acknowledge (RXACK) flag are all located in the Master Status (TWIn.MSTATUS) register.

Figure 26-8. Quick Command Frame Format



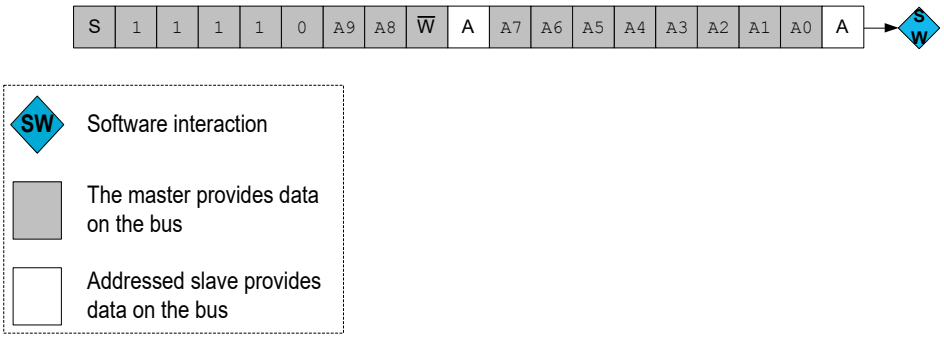
26.3.3.5 10-bit Address

Regardless of whether the transaction is a read or write, the master must start by sending the 10-bit address with the R/$\overline{W}$ direction bit set to '0'.

The slave address match logic supports recognition of 7-bit addresses and general call address. The Slave Address (TWIn.SADDR) register is used by the slave address match logic to determine if a master device has addressed the TWI slave.

The TWI slave address match logic only supports recognition of the first byte of a 10-bit address and the second byte must be handled in software. The first byte of the 10-bit address will be recognized if the upper five bits of the Slave Address (TWIn.SADDR) register are 0b11110. Thus, the first byte will consist of five indication bits, the two Most Significant bits (MSb) of the 10-bits address and the R/$\overline{W}$ direction bit. The Least Significant Byte (LSB) of the address that follows from the master will come in the form of a data packet.

Figure 26-9. 10-bit Address Transmission



26.3.4 Interrupts

Table 26-1. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
Slave	TWI Slave interrupt	• DIF: Data Interrupt Flag in TWIn.SSTATUS is set to '1' • APIF: Address or Stop Interrupt Flag in TWIn.SSTATUS is set to '1'
Master	TWI Master interrupt	• RIF: Read Interrupt Flag in TWIn.MSTATUS is set to '1' • WIF: Write Interrupt Flag in TWIn.MSTATUS is set to '1'

When an interrupt condition occurs, the corresponding interrupt flag is set in the Master Status (TWIn.MSTATUS) register or the Slave Status (TWIn.SSTATUS) register.
When several interrupt request conditions are supported by an interrupt vector, the interrupt requests are ORed together into one combined interrupt request to the interrupt controller. The user must read the Interrupt flags from the TWIn.MSTATUS register or the TWIn.SSTATUS register, to determine which of the interrupt conditions are present.

26.3.5 Sleep Mode Operation

The bus state logic and the address recognition hardware continue to operate in all sleep modes. If a slave device is in sleep mode and a Start condition followed by the address of the slave is detected, clock stretching is active during the wake-up period until the main clock is available. The master will stop operation in all sleep modes.

26.3.6 Debug Operation

During run-time debugging, the TWI will continue normal operation. Halting the CPU in Debugging mode will halt the normal operation of the TWI. The TWI can be forced to operate with halted CPU by writing a '1' to the Debug Run (DBGRUN) bit in the Debug Control (TWIn.DBGCTRL) register. When the CPU is halted in Debug mode and the DBGRUN bit is '1', reading or writing the Master Data (TWIn.MDATA) register or the Slave Data (TWIn.SDATA) register will neither trigger a bus operation, nor cause transmit and clear flags. If the TWI is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during halted debugging.

26.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0				SDASETUP	SDAHOLD[1:0]		FMPEN	
0x01	Reserved									
0x02	DBGCTRL	7:0								DBGRUN
0x03	MCTRLA	7:0	RIEN	WIEN		QCEN	TIMEOUT[1:0]		SMEN	ENABLE
0x04	MCTRLB	7:0					FLUSH	ACKACT	MCMD[1:0]	
0x05	MSTATUS	7:0	RIF	WIF	CLKHOLD	RXACK	ARBLOST	BUSERR	BUSSTATE[1:0]	
0x06	MBAUD	7:0	BAUD[7:0]							
0x07	MADDR	7:0	ADDR[7:0]							
0x08	MDATA	7:0	DATA[7:0]							
0x09	SCTRLA	7:0	DIEN	APIEN	PIEN			PMEN	SMEN	ENABLE
0x0A	SCTRLB	7:0						ACKACT	SCMD[1:0]	
0x0B	SSTATUS	7:0	DIF	APIF	CLKHOLD	RXACK	COLL	BUSERR	DIR	AP
0x0C	SADDR	7:0	ADDR[7:0]							
0x0D	SDATA	7:0	DATA[7:0]							
0x0E	SADDRMASK	7:0	ADDRMASK[6:0]							ADDREN

26.5 Register Description

26.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

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

Bit 4 – SDASETUP SDA Setup Time
By default, there are four clock cycles of setup time on the SDA out signal while reading from the slave part of the TWI module.

Value	Name	Description
0	4CYC	SDA setup time is four clock cycles
1	8CYC	SDA setup time is eight clock cycles

Bits 3:2 – SDAHOLD[1:0] SDA Hold Time
This bit field selects the SDA hold time for the TWI. See the *Electrical Characteristics* section for details.

Value	Name	Description
0x0	OFF	Hold time OFF
0x1	50NS	Short hold time
0x1	300NS	Meets the SMBus 2.0 specifications under typical conditions
0x3	500NS	Meets the SMBus 2.0 specifications across all corners

Bit 1 – FMPEN FM Plus Enable
Writing a '1' to this bit selects the 1 MHz bus speed for the TWI in the default configuration.

Value	Name	Description
0	OFF	Operating in Standard mode or Fast mode
1	ON	Operating in Fast mode Plus

26.5.2 Debug Control

Name: DBGCTRL
Offset: 0x02
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run
See the 26.3.6 Debug Operation section for more details.

Value	Description
0	The TWI is halted in Break Debug mode and ignores events
1	The TWI will continue to run in Break Debug mode when the CPU is halted

26.5.3 Master Control A

Name: MCTRLA
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RIEN	WIEN		QCEN	TIMEOUT[1:0]		SMEN	ENABLE
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 – RIEN Read Interrupt Enable
A TWI master read interrupt will be generated only if this bit and the Global Interrupt Enable (I) bit in the Status (CPU.SREG) register are set to '1'.
Writing a '1' to this bit enables the interrupt on the Read Interrupt Flag (RIF) in the Master Status (TWIn.MSTATUS) register. When the master read interrupt occurs, the RIF flag is set to '1'.

Bit 6 – WIEN Write Interrupt Enable
A TWI master write interrupt will be generated only if this bit and the Global Interrupt Enable (I) bit in the Status (CPU.SREG) register are set to '1'.
Writing a '1' to this bit enables the interrupt on the Write Interrupt Flag (WIF) in the Master Status (TWIn.MSTATUS) register. When the master write interrupt occurs, the WIF flag is set to '1'.

Bit 4 – QCEN Quick Command Enable
Writing a '1' to this bit enables the Quick Command mode. If the Quick Command mode is enabled and a slave acknowledges the address, the corresponding Read Interrupt Flag (RIF) or Write Interrupt Flag (WIF) will be set depending on the value of R/$\overline{W}$ bit.
The software must issue a Stop command by writing to the Command (MCMD) bit field in the Master Control B (TWIn.MCTRLB) register.

Bits 3:2 – TIMEOUT[1:0] Inactive Bus Time-Out
Setting this bit field to a nonzero value will enable the inactive bus time-out supervisor. If the bus is inactive for longer than the TIMEOUT setting, the bus state logic will enter the Idle state.

Value	Name	Description
0x0	DISABLED	Bus time-out disabled - I^2C
0x1	50US	50 µs - SMBus (assume the baud rate is set to 100 kHz)
0x2	100US	100 µs (assume the baud rate is set to 100 kHz)
0x3	200US	200 µs (assume the baud rate is set to 100 kHz)

Bit 1 – SMEN Smart Mode Enable
Writing a '1' to this bit enables the Master Smart mode. When the Smart mode is enabled, the existing value in the Acknowledge Action (ACKACT) bit from the Master Control B (TWIn.MCTRLB) register is sent immediately after reading the Master Data (TWIn.MDATA) register.

Bit 0 – ENABLE Enable TWI Master
Writing a '1' to this bit enables the TWI as master.

26.5.4 Master Control B

Name: MCTRLB
Offset: 0x04
Reset: 0x00
Property: -

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

Bit 3 – FLUSH Flush
This bit clears the internal state of the master and the bus states changes to Idle. The TWI will transmit invalid data if the Master Data (TWIn.MDATA) register is written before the Master Address (TWIn.MADDR) register.
Writing a '1' to this bit generates a strobe for one clock cycle, disabling the master, and then re-enabling the master.
Writing a '0' to this bit has no effect.

Bit 2 – ACKACT Acknowledge Action
The ACKACT(1) bit represents the behavior in the Master mode under certain conditions defined by the bus state and the software interaction. If the Smart Mode Enable (SMEN) bit in the Master Control A (TWIn.MCTRLA) register is set to '1', the acknowledge action is performed when the Master Data (TWIn.MDATA) register is read, else a command must be written to the Command (MCDM) bit field in the Master Control B (TWIn.MCTRLB) register.
The acknowledge action is not performed when the Master Data (TWIn.MDATA) register is written, since the master is sending data.

Value	Name	Description
0	ACK	Send ACK
1	NACK	Send NACK

Bits 1:0 – MCMD[1:0] Command
The MCMD(1) bit field is a strobe. This bit field is always read as '0'.
Writing to this bit field triggers a master operation as defined by the table below.

Table 26-2. Command Settings

MCMD[1:0]	Group Configuration	DIR	Description
0x0	NOACT	X	Reserved
0x1	REPSTART	X	Execute Acknowledge Action followed by repeated Start condition
0x2	RECVTRANS	$\overline{W}$	Execute Acknowledge Action (no action) followed by a byte write operation(2)
		R	Execute Acknowledge Action followed by a byte read operation
0x3	STOP	X	Execute Acknowledge Action followed by issuing a Stop condition

Note:

1. The ACKACT bit and the MCMD bit field can be written at the same time.
2. For a master write operation, the TWI will wait for new data to be written to the Master Data (TWIn.MDATA) register.

26.5.5 Master Status

Name: MSTATUS
Offset: 0x05
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RIF	WIF	CLKHOLD	RXACK	ARBLOST	BUSERR	BUSSTATE[1:0]	
Access	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – RIF Read Interrupt Flag
This flag is set to '1' when the master byte read operation is successfully completed.
The RIF flag can be used for a master read interrupt. More information can be found in the Read Interrupt Enable (RIEN) bit from the Master Control A (TWIn.MCTRLA) register.
This flag is automatically cleared when accessing several other TWI registers. The RIF flag can be cleared by choosing one of the following methods:

1. Writing a '1' to it.
2. Writing to the Master Address (TWIn.MADDR) register.
3. Writing/Reading the Master Data (TWIn.MDATA) register.
4. Writing to the Command (MCMD) bit field from the Master Control B (TWIn.MCTRLB) register.

Bit 6 – WIF Write Interrupt Flag
This flag is set to '1' when a master transmit address or byte write operation is completed, regardless of the occurrence of a bus error or arbitration lost condition.
The WIF flag can be used for a master write interrupt. More information can be found from the Write Interrupt Enable (WIEN) bit in the Master Control A (TWIn.MCTRLA) register.
This flag can be cleared by choosing one of the methods described for the RIF flag.

Bit 5 – CLKHOLD Clock Hold
When this bit is read as '1', it indicates that the master is currently holding the SCL low, stretching the TWI clock period.
This bit can be cleared by choosing one of the methods described for the RIF flag.

Bit 4 – RXACK Received Acknowledge
When this flag is read as '0', it indicates that the most recent Acknowledge bit from the slave was ACK and the slave is ready for more data.
When this flag is read as '1', it indicates that the most recent Acknowledge bit from the slave was NACK and the slave is not able to or does not need to receive more data.

Bit 3 – ARBLOST Arbitration Lost
When this bit is read as '1', it indicates that the master has lost arbitration. This can happen in one of the following cases:

1. While transmitting a high data bit.
2. While transmitting a NACK bit.
3. While issuing a Start condition (S).
4. While issuing a repeated Start (Sr).

This flag can be cleared by choosing one of the methods described for the RIF flag.

Bit 2 – BUSERR Bus Error
The BUSERR flag indicates that an illegal bus operation has occurred. An illegal bus operation is detected if a protocol violating the Start (S), repeated Start (Sr), or Stop (P) conditions is detected on the TWI bus lines. A Start condition directly followed by a Stop condition is one example of a protocol violation.

The BUSERR flag can be cleared by choosing one of the following methods:

1. Writing a '1' to it.
2. Writing to the Master Address (TWIn.MADDR) register.

The TWI bus error detector is part of the TWI Master circuitry. For the bus errors to be detected, the TWI Master must be enabled (ENABLE bit in TWIn.MCTRLA is '1'), and the main clock frequency must be at least four times the SCL frequency.

Bits 1:0 – BUSSTATE[1:0] Bus State
This bit field indicates the current TWI bus state.

Value	Name	Description
0x0	UNKNOWN	Unknown bus state
0x1	IDLE	Idle bus state
0x2	OWNER	This TWI controls the bus
0x3	BUSY	Busy bus state

26.5.6 Master Baud Rate

Name: MBAUD
Offset: 0x06
Reset: 0x00
Property: -

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 Rate
This bit field is used to derive the SCL high and low time. It must be written while the master is disabled. The master can be disabled by writing '0' to the Enable TWI Master (ENABLE) bit from the Master Control A (TWIn.MCTRLA) register.
Refer to the 26.3.2.2.1 Clock Generation section for more information on how to calculate the frequency of the SCL.

26.5.7 Master Address

Name: MADDR
Offset: 0x07
Reset: 0x00
Property: -

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 7:0 – ADDR[7:0] Address
This register contains the address of the external slave device. When this bit field is written, the TWI will issue a Start condition, and the shift register performs a byte transmit operation on the bus depending on the bus state.
This register can be read at any time without interfering with the ongoing bus activity since a read access does not trigger the master logic to perform any bus protocol related operations.
The master control logic uses the bit 0 of this register as the R/$\overline{W}$ direction bit.

26.5.8 Master Data

Name: MDATA
Offset: 0x08
Reset: 0x00
Property: -

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
This bit field provides direct access to the master's physical shift register, which is used to shift out data on the bus (transmit) and to shift in data received from the bus (receive). The direct access implies that the MDATA register cannot be accessed during byte transmissions.
Reading valid data or writing data to be transmitted can only be successful when the CLKHOLD bit is read as '1' or when an interrupt occurs.
A write access to the MDATA register will command the master to perform a byte transmit operation on the bus, directly followed by receiving the Acknowledge bit from the slave. This is independent of the Acknowledge Action (ACKACT) bit from the Master Control B (TWIn.MCTRLB) register. The write operation is performed regardless of winning or losing arbitration before the Write Interrupt Flag (WIF) is set to '1'.
If the Smart Mode Enable (SMEN) bit in the Master Control A (TWIn.MCTRLA) register is set to '1', a read access to the MDATA register will command the master to perform an acknowledge action. This is dependent on the setting of the Acknowledge Action (ACKACT) bit from the Master Control B (TWIn.MCTRLB) register.
Note:

1. The WIF and RIF interrupt flags are cleared automatically if the MDATA register is read while ACKACT is set to '1'.
2. The ARBLOST and BUSEER flags are left unchanged.
3. The WIF, RIF, ARBLOST, and BUSERR flags together with the Clock Hold (CLKHOLD) bit are all located in the Master Status (TWIn.MSTATUS) register.

26.5.9 Slave Control A

Name: SCTRLA
Offset: 0x09
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	DIEN	APIEN	PIEN			PMEN	SMEN	ENABLE
Access	R/W	R/W	R/W			R/W	R/W	R/W
Reset	0	0	0			0	0	0

Bit 7 – DIEN Data Interrupt Enable
Writing this bit to '1' enables an interrupt on the Data Interrupt Flag (DIF) from the Slave Status (TWIn.SSTATUS) register.
A TWI slave data interrupt will be generated only if this bit, the DIF flag, and the Global Interrupt Enable (I) bit in Status (CPU.SREG) register are all '1'.

Bit 6 – APIEN Address or Stop Interrupt Enable
Writing this bit to '1' enables an interrupt on the Address or Stop Interrupt Flag (APIF) from the Slave Status (TWIn.SSTATUS) register.
A TWI slave address or stop interrupt will be generated only if this bit, the APIF flag, and the Global Interrupt Enable (I) bit in the Status (CPU.SREG) register are all '1'.
Note:

1. The slave stop interrupt shares the interrupt flag and vector with the slave address interrupt.
2. The Stop Interrupt Enable (PIEN) bit in the Slave Control A (TWIn.SCTRLA) register must be written to '1' for the APIF to be set on a Stop condition.
3. When the interrupt occurs, the Address or Stop (AP) bit in the Slave Status (TWIn.SSTATUS) register will determine whether an address match or a Stop condition caused the interrupt.

Bit 5 – PIEN Stop Interrupt Enable
Writing this bit to '1' allows the Address or Stop Interrupt Flag (APIF) in the Slave Status (TWIn.SSTATUS) register to be set when a Stop condition occurs. To use this feature, the main clock frequency must be at least four times the SCL frequency.

Bit 2 – PMEN Address Recognition Mode
If this bit is written to '1', the slave address match logic responds to all received addresses.
If this bit is written to '0', the address match logic uses the Slave Address (TWIn.SADDR) register to determine which address to recognize as the slave's address.

Bit 1 – SMEN Smart Mode Enable
Writing this bit to '1' enables the slave Smart mode. When the Smart mode is enabled, issuing a command by writing to the Command (SCMD) bit field in the Slave Control B (TWIn.SCTRLB) register or accessing the Slave Data (TWIn.SDATA) register resets the interrupt, and the operation continues. If the Smart mode is disabled, the slave always waits for a new slave command before continuing.

Bit 0 – ENABLE Enable TWI Slave
Writing this bit to '1' enables the TWI slave.

26.5.10 Slave Control B

Name: SCTRLB
Offset: 0x0A
Reset: 0x00
Property: -

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

Bit 2 – ACKACT Acknowledge Action
The ACKACT[1] bit represents the behavior of the slave device under certain conditions defined by the bus protocol state and the software interaction. If the Smart Mode Enable (SMEN) bit in the Slave Control A (TWIn.SCTRLA) register is set to '1', the acknowledge action is performed when the Slave Data (TWIn.SDATA) register is read, else a command must be written to the Command (SCMD) bit field in the Slave Control B (TWIn.SCTRLB) register.
The acknowledge action is not performed when the Slave Data (TWIn.SDATA) register is written, since the slave is sending data.

Value	Name	Description
0	ACK	Send ACK
1	NACK	Send NACK

Bits 1:0 – SCMD[1:0] Command
The SCMD[1] bit field is a strobe. This bit field is always read as '0'.
Writing to this bit field triggers a slave operation as defined by the table below.

Table 26-3. Command Settings

SCMD[1:0]	Group Configuration	DIR	Description
0x0	NOACT	X	No action
0x1	—	X	Reserved
0x2	COMPTRANS	Used to complete a transaction	
		$\overline{W}$	Execute Acknowledge Action succeeded by waiting for any Start (S/Sr) condition
		R	Wait for any Start (S/Sr) condition
0x3	RESPONSE	Used in response to an address interrupt (APIF)	
		$\overline{W}$	Execute Acknowledge Action succeeded by reception of next byte
		R	Execute Acknowledge Action succeeded by slave data interrupt
		Used in response to a data interrupt (DIF)	
		$\overline{W}$	Execute Acknowledge Action succeeded by reception of next byte
		R	Execute a byte read operation followed by Acknowledge Action

Note: 1. The ACKACT bit and the SCMD bit field can be written at the same time. The ACKACT will be updated before the command is triggered.

26.5.11 Slave Status

Name: SSTATUS
Offset: 0x0B
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	DIF	APIF	CLKHOLD	RXACK	COLL	BUSERR	DIR	AP
Access	R/W	R/W	R	R	R/W	R/W	R	R
Reset	0	0	0	0	0	0	0	0

Bit 7 – DIF Data Interrupt Flag
This flag is set to '1' when the slave byte transmit or receive operation is completed without any bus errors. This flag can be set to '1' with an unsuccessful transaction in case of collision detection. More information can be found in the Collision (COLL) bit description.
The DIF flag can generate a slave data interrupt. More information can be found in Data Interrupt Enable (DIEN) bit from the Slave Control A (TWIn.SCTRLA) register.
This flag is automatically cleared when accessing several other TWI registers. The DIF flag can be cleared by choosing one of the following methods:

1. Writing/Reading the Slave Data (TWIn.SDATA) register.
2. Writing to the Command (SCMD) bit field from the Slave Control B (TWIn.SCTRLB) register.

Bit 6 – APIF Address or Stop Interrupt Flag
This flag is set to '1' when the slave address has been received or by a Stop condition.
The APIF flag can generate a slave address or stop interrupt. More information can be found in the Address or Stop Interrupt Enable (APIEN) bit from the Slave Control A (TWIn.SCTRLA) register.
This flag can be cleared by choosing one of the methods described for the DIF flag.

Bit 5 – CLKHOLD Clock Hold
When this bit is read as '1', it indicates that the slave is currently holding the SCL low, stretching the TWI clock period.
This bit is set to '1' when an address or data interrupt occurs. Resetting the corresponding interrupt will indirectly set this bit to '0'.

Bit 4 – RXACK Received Acknowledge
When this flag is read as '0', it indicates that the most recent Acknowledge bit from the master was ACK.
When this flag is read as '1', it indicates that the most recent Acknowledge bit from the master was NACK.

Bit 3 – COLL Collision
When this bit is read as '1', it indicates that the slave has not been able to do one of the following:

1. Transmit high bits on the SDA. The Data Interrupt Flag (DIF) will be set to '1' at the end as a result of the internal completion of an unsuccessful transaction.
2. Transmit the NACK bit. The collision occurs because the slave address match already took place, and the APIF flag is set to '1' as a result.

Writing a '1' to this bit will clear the COLL flag. The flag is automatically cleared if any Start condition (S/Sr) is detected.
Note: The APIF and DIF flags can only generate interrupts whose handlers can be used to check for the collision.

Bit 2 – BUSERR Bus Error
The BUSERR flag indicates that an illegal bus operation has occurred. Illegal bus operation is detected if a protocol violating the Start (S), repeated Start (Sr), or Stop (P) conditions are detected on the TWI bus lines. A Start condition directly followed by a Stop condition is one example of a protocol violation.
Writing a '1' to this bit will clear the BUSERR flag.

The TWI bus error detector is part of the TWI Master circuitry. For the bus errors to be detected by the slave, the TWI Master must be enabled, and the main clock frequency must be at least four times the SCL frequency. The TWI Master can be enabled by writing a '1' to the ENABLE bit in the TWIn.MCTRLA register.

Bit 1 – DIR Read/Write Direction
This bit indicates the current TWI bus direction. The DIR bit reflects the direction bit value from the last address packet received from a master TWI device.
When this bit is read as '1', it indicates that a master read operation is in progress.
When this bit is read as '0', it indicates that a master write operation is in progress.

Bit 0 – AP Address or Stop
When the TWI slave Address or Stop Interrupt Flag (APIF) is set to '1', this bit determines whether the interrupt is due to an address detection or a Stop condition.

Value	Name	Description
0	STOP	A Stop condition generated the interrupt on the APIF flag
1	ADR	Address detection generated the interrupt on the APIF flag

26.5.12 Slave Address

Name: SADDR
Offset: 0x0C
Reset: 0x00
Property: -

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 7:0 – ADDR[7:0] Address
The Slave Address (TWIn.SADDR) register is used by the slave address match logic to determine if a master device has addressed the TWI slave. The Address or Stop Interrupt Flag (APIF) and the Address or Stop (AP) bit in the Slave Status (TWIn.SSTATUS) register are set to '1' if an address packet is received.
The upper seven bits (ADDR[7:1]) of the SADDR register represent the main slave address.
The Least Significant bit (ADDR[0]) of the SADDR register is used for the recognition of the General Call Address (0x00) of the I^2C protocol. This feature is enabled when this bit is set to '1'.

26.5.13 Slave Data

Name: SDATA
Offset: 0x0D
Reset: 0x00
Property: -

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
This bit field provides access to the slave data register.
Reading valid data or writing data to be transmitted can only be successfully achieved when the SCL is held low by the slave (i.e., when the slave CLKHOLD bit is set to '1'). It is not necessary to check the Clock Hold (CLKHOLD) bit from the Slave Status (TWIn.SSTATUS) register in software before accessing the SDATA register if the software keeps track of the present protocol state by using interrupts or observing the interrupt flags.
If the Smart Mode Enable (SMEN) bit in the Slave Control A (TWIn.SCTRLA) register is set to '1', a read access to the SDATA register, when the clock hold is active, auto-triggers bus operations and will command the slave to perform an acknowledge action. This is dependent on the setting of the Acknowledge Action (ACKACT) bit from the Slave Control B (TWIn.SCTRLB) register.

26.5.14 Slave Address Mask

Name: SADDRMASK
Offset: 0x0E
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ADDRMASK[6:0]							ADDREN
Access	R/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 7:1 – ADDRMASK[6:0] Address Mask
The ADDRMASK bit field acts as a second address match or an address mask register depending on the ADDREN bit.
If the ADDREN bit is written to '0', the ADDRMASK bit field can be loaded with a 7-bit Slave Address mask. Each of the bits in the Slave Address Mask (TWIn.SADDRMASK) register can mask (disable) the corresponding address bits in the TWI Slave Address (TWIn.SADDR) register. When a bit from the mask is written to '1', the address match logic ignores the comparison between the incoming address bit and the corresponding bit in the Slave Address (TWIn.SADDR) register. In other words, masked bits will always match, making it possible to recognize ranges of addresses.
If the ADDREN bit is written to '1', the Slave Address Mask (TWIn.SADDRMASK) register can be loaded with a second slave address in addition to the Slave Address (TWIn.SADDR) register. In this mode, the slave will have two unique addresses, one in the Slave Address (TWIn.SADDR) register and the other one in the Slave Address Mask (TWIn.SADDRMASK) register.

Bit 0 – ADDREN Address Mask Enable
If this bit is written to '0', the TWIn.SADDRMASK register acts as a mask to the TWIn.SADDR register.
If this bit is written to '1', the slave address match logic responds to the two unique addresses in slave TWIn.SADDR and TWIn.SADDRMASK.

27. CRCSCAN - Cyclic Redundancy Check Memory Scan

27.1 Features

- CRC-16-CCITT
- Check of the Entire Flash Section, Application Code, and/or Boot Section
- Selectable NMI Trigger on Failure
- User-Configurable Check During Internal Reset Initialization

27.2 Overview

A Cyclic Redundancy Check (CRC) takes a data stream of bytes from the NVM (either the entire Flash, only the Boot section, or both the Boot section and the application code section) and generates a checksum. The CRC peripheral (CRCSCAN) can be used to detect errors in the program memory.

The last location in the section to check has to contain the correct pre-calculated 16-bit checksum for comparison. If the checksum calculated by the CRCSCAN and the pre-calculated checksums match, a status bit is set. If they do not match, the Status register (CRCSCAN.STATUS) will indicate that it failed. The user can choose to let the CRCSCAN generate a Non-Maskable Interrupt (NMI) if the checksums do not match.

An *n*-bit CRC applied to a data block of arbitrary length will detect any single alteration (error burst) up to *n* bits in length. For longer error bursts a fraction $1-2^{-n}$ will be detected.

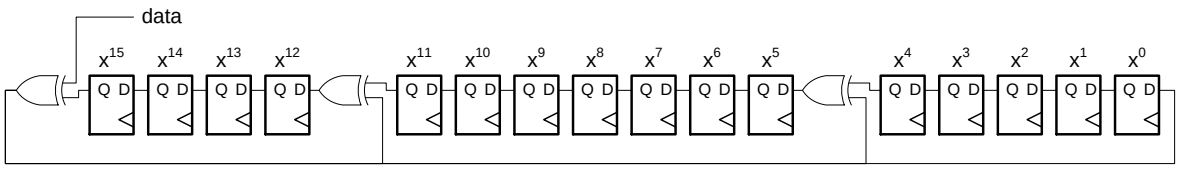
The CRC generator supports CRC-16-CCITT.

Polynomial:

- CRC-16-CCITT: $x^{16} + x^{12} + x^5 + 1$

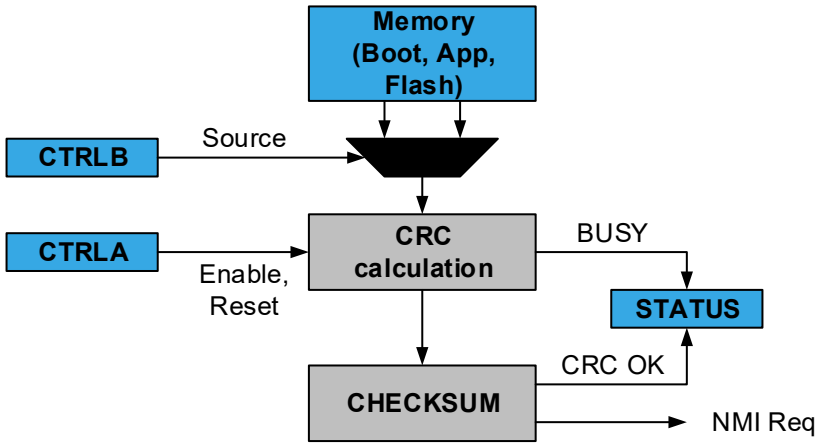
The CRC reads byte-by-byte the content of the section(s) it is set up to check, starting with byte 0, and generates a new checksum per byte. The byte is sent through a shift register as depicted below, starting with the Most Significant bit. If the last bytes in the section contain the correct checksum, the CRC will pass. See 27.3.2.1 Checksum for how to place the checksum. The initial value of the Checksum register is 0xFFFF.

Figure 27-1. CRC Implementation Description



27.2.1 Block Diagram

Figure 27-2. Cyclic Redundancy Check Block Diagram



27.3 Functional Description

27.3.1 Initialization

To enable a CRC in software (or via the debugger):

1. Write the Source (SRC) bit field of the Control B register (CRCSCAN.CTRLB) to select the desired mode and source settings.
2. Enable the CRCSCAN by writing a '1' to the ENABLE bit in the Control A register (CRCSCAN.CTRLA).
3. The CRC will start after three cycles. The CPU will continue executing during these three cycles.

The CRCSCAN can be configured to perform a code memory scan before the device leaves Reset. If this check fails, the CPU is not allowed to start normal code execution. This feature is enabled and controlled by the CRCSRC field in FUSE.SYSCFG0, see the *Fuses* chapter for more information.

If this feature is enabled, a successful CRC check will have the following outcome:

- Normal code execution starts
- The ENABLE bit in CRCSCAN.CTRLA will be '1'
- The SRC bit field in CRCSCAN.CTRLB will reflect the checked section(s)
- The OK flag in CRCSCAN.STATUS will be '1'

If this feature is enabled, a non-successful CRC check will have the following outcome:

- Normal code execution does not start, the CPU will hang executing no code
- The ENABLE bit in CRCSCAN.CTRLA will be '1'
- The SRC bit field in CRCSCAN.CTRLB will reflect the checked section(s)
- The OK flag in CRCSCAN.STATUS will be '0'
- This condition may be observed using the debug interface

27.3.2 Operation

When the CRC is operating in Priority mode, the CRC peripheral has priority access to the Flash and will stall the CPU until completed.

In Priority mode the CRC fetches a new word (16-bit) on every third main clock cycle, or when the CRC peripheral is configured to do a scan from start-up.

27.3.2.1 Checksum

The pre-calculated checksum must be present in the last location of the section to be checked. If the BOOT section is to be checked, the checksum must be saved in the last bytes of the BOOT section, and similarly for APPLICATION and the entire Flash. Table 27-1 shows explicitly how the checksum must be stored for the different sections. Also,

see the CRCSCAN.CTRLB register description for how to configure which section to check and the device fuse description for how to configure the BOOTEND and APPEND fuses.

Table 27-1. Placement of the Pre-Calculated Checksum in Flash

Section to Check	CHECKSUM[15:8]	CHECKSUM[7:0]
BOOT	FUSE_BOOTEND*256-2	FUSE_BOOTEND*256-1
BOOT and APPLICATION	FUSE_APPEND*256-2	FUSE_APPEND*256-1
Full Flash	FLASHEND-1	FLASHEND

27.3.3 Interrupts

Table 27-2. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
NMI	Non-Maskable Interrupt	CRC failure

When the interrupt condition occurs the OK flag in the Status (CRCSCAN.STATUS) register is cleared to '0'.

A Non-Maskable Interrupt (NMI) is enabled by writing a '1' to the respective Enable (NMIEN) bit in the Control A (CRCSCAN.CTRLA) register, but can only be disabled with a System Reset. An NMI is generated when the OK flag in the CRCSCAN.STATUS register is cleared, and the NMIEN bit is '1'. The NMI request remains active until a System Reset and cannot be disabled.

An NMI can be triggered even if interrupts are not globally enabled.

27.3.4 Sleep Mode Operation

CRCSCAN is halted in all Sleep modes. In all CPU Sleep modes, the CRCSCAN peripheral is halted and will resume operation when the CPU wakes up.

The CRCSCAN starts operation three cycles after writing the EN bit in CRCSCAN.CTRLA. During these three cycles, it is possible to enter Sleep mode. In this case:

1. The CRCSCAN will not start until the CPU is woken up.
2. Any interrupt handler will execute after CRCSCAN has finished.

27.3.5 Debug Operation

Whenever the debugger reads or writes a peripheral or memory location, the CRCSCAN will be disabled.

If the CRCSCAN is busy when the debugger accesses the device, the CRCSCAN will restart the ongoing operation when the debugger accesses an internal register or when the debugger disconnects.

The BUSY bit in the Status (CRCSCAN.STATUS) register will read '1' if the CRCSCAN was busy when the debugger caused it to disable, but it will not actively check any section as long as the debugger keeps it disabled. There are synchronized CRC status bits in the debugger's internal register space, which can be read by the debugger without disabling the CRCSCAN. Reading the debugger's internal CRC status bits will make sure that the CRCSCAN is enabled.

It is possible to write the CRCSCAN.STATUS register directly from the debugger:

- BUSY bit in CRCSCAN.STATUS:
  - Writing the BUSY bit to '0' will stop the ongoing CRC operation (so that the CRCSCAN does not restart its operation when the debugger allows it).
  - Writing the BUSY bit to '1' will make the CRC start a single check with the settings in the Control B (CRCSCAN.CTRLB) register, but not until the debugger allows it.

  As long as the BUSY bit in CRCSCAN.STATUS is '1', CRCSCAN.CTRLB and the Non-Maskable Interrupt Enable (NMIEN) bit in the Control A (CRCSCAN.CTRLA) register cannot be altered.
- OK bit in CRCSCAN.STATUS:

- Writing the OK bit to '0' can trigger a Non-Maskable Interrupt (NMI) if the NMIEN bit in CRCSCAN.CTRLA is '1'. If an NMI has been triggered, no writes to the CRCSCAN are allowed.
- Writing the OK bit to '1' will make the OK bit read as '1' when the BUSY bit in CRCSCAN.STATUS is '0'.

Writes to CRCSCAN.CTRLA and CRCSCAN.CTRLB from the debugger are treated in the same way as writes from the CPU.

27.4 Register Summary - CRCSCAN

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RESET						NMIEN	ENABLE
0x01	CTRLB	7:0			MODE[1:0]				SRC[1:0]	
0x02	STATUS	7:0							OK	BUSY

27.5 Register Description

27.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

If an NMI has been triggered this register is not writable.

Bit	7	6	5	4	3	2	1	0
	RESET						NMIEN	ENABLE
Access	R/W						R/W	R/W
Reset	0						0	0

Bit 7 – RESET Reset CRCSCAN
Writing this bit to '1' resets the CRCSCAN peripheral. The CRCSCAN Control registers and Status register (CRCSCAN.CTRLA, CRCSCAN.CTRLB, CRCSCAN.STATUS) will be cleared one clock cycle after the RESET bit is written to '1'.
If NMIEN is '0', this bit is writable both when the CRCSCAN is busy (the BUSY bit in CRCSCAN.STATUS is '1') and not busy (the BUSY bit is '0'), and will take effect immediately.
If NMIEN is '1', this bit is only writable when the CRCSCAN is not busy (the BUSY bit in CRCSCAN.STATUS is '0').
The RESET bit is a strobe bit.

Bit 1 – NMIEN Enable NMI Trigger
When this bit is written to '1', any CRC failure will trigger an NMI.
This bit can only be cleared by a system Reset - it is not cleared by a write to the RESET bit.
This bit can only be written to '1' when the CRCSCAN is not busy (the BUSY bit in CRCSCAN.STATUS is '0').

Bit 0 – ENABLE Enable CRCSCAN
Writing this bit to '1' enables the CRCSCAN peripheral with the current settings. It will stay '1' even after a CRC check has completed, but writing it to '1' again will start a new check.
Writing the bit to '0' has no effect
The CRCSCAN can be configured to run a scan during the MCU start-up sequence to verify the Flash sections before letting the CPU start normal code execution (see the 27.3.1 Initialization section). If this feature is enabled, the ENABLE bit will read as '1' when normal code execution starts.
To see whether the CRCSCAN peripheral is busy with an ongoing check, poll the BUSY bit in the Status register (CRCSCAN.STATUS).

27.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

The CRCSCAN.CTRLB register contains the mode and source settings for the CRC. It is not writable when the CRC is busy, or when an NMI has been triggered.

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

Bits 5:4 – MODE[1:0] CRC Flash Access Mode
The CRC can be enabled during internal Reset initialization to verify Flash sections before letting the CPU start (see the device data sheet fuse description). If the CRC is enabled during internal Reset initialization, the MODE bit field will read out non-zero when normal code execution starts. To ensure proper operation of the CRC under code execution, write the MODE bit to 0x0 again.

Value	Name	Description
0x0	PRIORITY	The CRC module runs a single check with priority to Flash. The CPU is halted until the CRC completes.
other	-	Reserved

Bits 1:0 – SRC[1:0] CRC Source
The SRC bit field selects which section of the Flash the CRC module will check. To set up section sizes, refer to the fuse description.
The CRC can be enabled during internal Reset initialization to verify Flash sections before letting the CPU start (see the *Fuses* chapter). If the CRC is enabled during internal Reset initialization, the SRC bit field will read out as FLASH, BOOTAPP, or BOOT when normal code execution starts (depending on the configuration).

Value	Name	Description
0x0	FLASH	The CRC is performed on the entire Flash (boot, application code, and application data sections).
0x1	BOOTAPP	The CRC is performed on the boot and application code sections of Flash.
0x2	BOOT	The CRC is performed on the boot section of Flash.
0x3	-	Reserved.

27.5.3 Status

Name: STATUS
Offset: 0x02
Reset: 0x02
Property: -

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

Bit 1 – OK CRC OK
When this bit is read as '1', the previous CRC completed successfully. The bit is set to '1' by default before a CRC scan is run. The bit is not valid unless BUSY is '0'.

Bit 0 – BUSY CRC Busy
When this bit is read as '1', the CRCSCAN is busy. As long as the module is busy, the access to the control registers is limited.

28. CCL - Configurable Custom Logic

28.1 Features

- Glue Logic for General Purpose PCB Design
- Up to Two Programmable Look-up Tables LUT[1:0]
- Combinatorial Logic Functions: Any Logic Expression That Is a Function of up to Three Inputs
- Sequential Logic Functions:
  - Gated D Flip-Flop
  - JK Flip-Flop
  - Gated D Latch
  - RS Latch
- Flexible Look-up Table Inputs Selection:
  - I/Os
  - Events
  - Subsequent LUT output
  - Internal peripherals
    - Analog comparator
    - Timer/counters
    - USART
    - SPI
- Clocked by System Clock or Other Peripherals
- The Output Can Be Connected to I/O Pins or Event System
- Optional Synchronizer, Filter, or Edge Detector Available on Each LUT Output

28.2 Overview

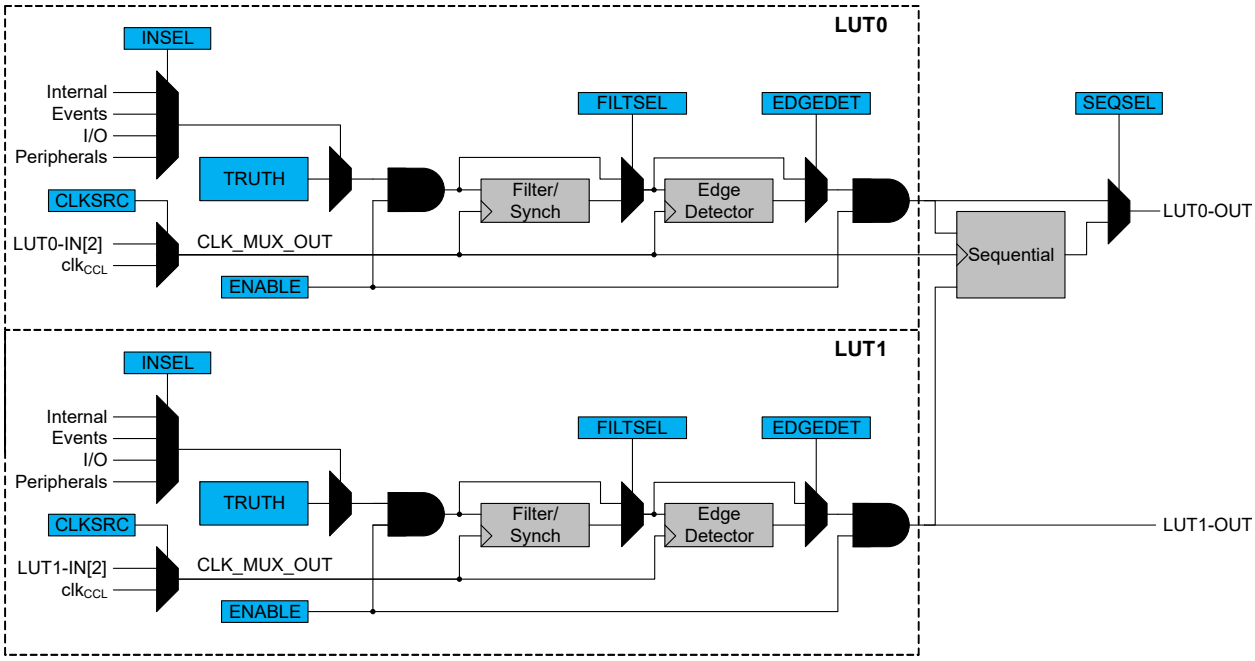
The Configurable Custom Logic (CCL) is a programmable logic peripheral that can be connected to the device pins, events or other internal peripherals. The CCL can serve as “glue logic” between the device peripherals and external devices. The CCL can eliminate the need for external logic components, and can also help the designer to overcome real-time constraints by combining core independent peripherals to handle the most time-critical parts of the application independent of the CPU.

The CCL peripheral has a number of Look-up Tables (LUT). Each LUT consists of three inputs, a truth table, and a filter/edge detector. Each LUT can generate an output as a user-programmable logic expression with three inputs. The inputs can be individually masked.

The output can be generated from the inputs combinatorially and be filtered to remove spikes. An optional sequential module can be enabled. The inputs to the sequential module are individually controlled by two independent, adjacent LUT (LUT0/LUT1) outputs, enabling complex waveform generation.

28.2.1 Block Diagram

Figure 28-1. Configurable Custom Logic



28.2.2 Signal Description

Pin Name	Type	Description
LUTn-OUT	Digital output	Output from LUT
LUTn-IN[2:0]	Digital input	Input to LUT

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

28.2.3 System Dependencies

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

Table 28-1. CCL System Dependencies

Dependency	Applicable	Peripheral
Clocks	Yes	CLKCTRL
I/O Lines and Connections	Yes	PORT
Interrupts	No	-
Events	Yes	EVSYS
Debug	Yes	UPDI

28.2.3.1 Clocks

By default, the CCL is using the peripheral clock of the device (CLK_PER).

Alternatively, the CCL can be clocked by a peripheral input that is available on LUT n input line 2 (LUTn_IN[2]). This is configured by writing a '1' to the Clock Source Selection (CLKSRC) bit in the LUT n Control A (CCL.LUTnCTRLA) register. The Sequential block and the even LUT in the LUT pair (SEQn.clk = LUT2n.clk) are clocked by the same clock. It is advised to disable the peripheral by writing a '0' to the Enable (ENABLE) bit in the Control A (CCL.CTRLA) register before configuring the CLKSRC bit in CCL.LUTnCTRLA.

Alternatively, the input line 2 (IN[2]) of an LUT can be used to clock the LUT and the corresponding Sequential block. This is enabled by writing a '1' to the CLKSRC bit in the CCL.LUTnCTRLA register.

The CCL must be disabled before changing the LUT clock source: Write a '0' to the ENABLE bit in CCL.CTRLA.

28.2.3.2 I/O Lines

The CCL can take inputs and generate output through I/O pins. For this to function properly, the I/O pins must be configured to be used by a Look-up Table (LUT).

28.2.3.3 Interrupts

Not applicable.

28.2.3.4 Debug Operation

When the CPU is halted in Debug mode, the CCL continues normal operation. However, the CCL cannot be halted when the CPU is halted in Debug mode. If the CCL is configured in a way that requires it to be periodically serviced by the CPU, improper operation or data loss may result during debugging.

28.3 Functional Description

28.3.1 Initialization

The following bits are enable-protected, meaning that they can only be written when the corresponding even LUT is disabled (ENABLE = '0' in CCL.LUT0CTRLA):

- Sequential Selection (SEQSEL) in Sequential Control 0 (CCL.SEQCTRL0) register

The following registers are enable-protected, meaning that they can only be written when the corresponding LUT is disabled (ENABLE = '0' in CCL.LUT0CTRLA):

- LUT n Control x (CCL.LUTnCTRLx) register, except the ENABLE bit

Enable-protected bits in the CCL.LUTnCTRLx registers can be written at the same time as the ENABLE bit in CCL.LUTnCTRLx is written to '1', but not at the same time as the ENABLE bit is written to '0'.

The enable-protection is denoted by the enable-protected property in the register description.

28.3.2 Operation

28.3.2.1 Enabling, Disabling and Resetting

The CCL is enabled by writing a '1' to the ENABLE bit in the Control A (CCL.CTRLA) register. The CCL is disabled by writing a '0' to that ENABLE bit.

Each LUT is enabled by writing a '1' to the LUT Enable (ENABLE) bit in the LUT n Control A (CCL.LUTnCTRLA) register. Each LUT is disabled by writing a '0' to the ENABLE bit in CCL.LUTnCTRLA.

28.3.2.2 Look-up Table Logic

The Look-up Table in each LUT unit can generate a combinational logic output as a function of up to three inputs IN[2:0]. The unused inputs can be masked (tied low). The truth table for the combinational logic expression is defined by the bits in the CCL.TRUTHn registers. Each combination of the input (IN[2:0]) bits corresponds to one bit in the TRUTHn register, as shown in the table below.

Table 28-2. 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]

.....continued			
IN[2]	IN[1]	IN[0]	OUT
1	0	0	TRUTH[4]
1	0	1	TRUTH[5]
1	1	0	TRUTH[6]
1	1	1	TRUTH[7]

28.3.2.3 Truth Table Inputs Selection

Input Overview
The inputs can be individually:

- Masked
- Driven by Peripherals:
  - Analog Comparator (AC) output
  - Timer/Counters (TC) waveform outputs
- Driven by Internal Events from Event System
- Driven by Other CCL Submodules

The input selection for each input y of LUT n is configured by writing the input y source selection bit in the LUT n Control x=[B,C] registers:

- INSEL0 in CCL.LUTnCTRLB
- INSEL1 in CCL.LUTnCTRLB
- INSEL2 in CCL.LUTnCTRLC

Internal Feedback Inputs (FEEDBACK)
When selected (INSELy=FEEDBACK in CCL.LUTnCTRLx), the Sequential (SEQ) output is used as input for the corresponding LUT.

The output from an internal sequential module can be used as an input source for the LUT. See the figure below for an example for LUT0 and LUT1. The sequential selection for each LUT follows the formula:

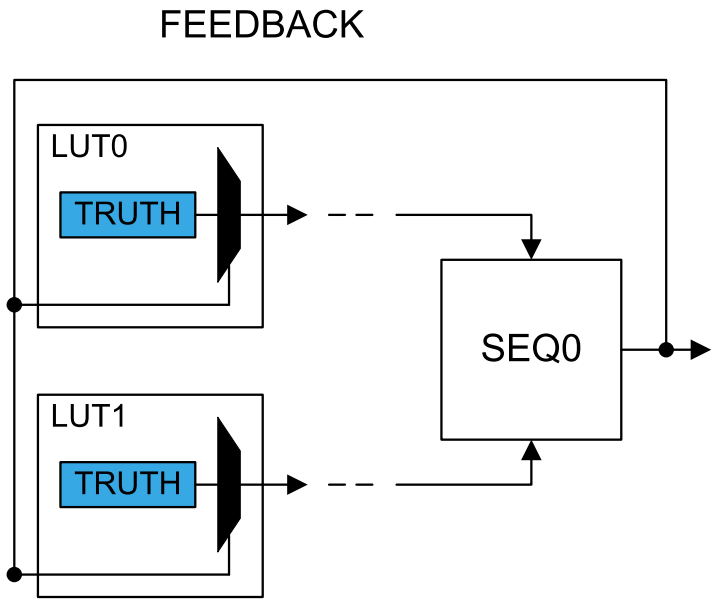
$\mathrm{IN}[2\mathrm{N}][i] = \mathrm{SEQ}[N]$

$\mathrm{IN}[2\mathrm{N}+1][i] = \mathrm{SEQ}[N]$

With *N* representing the sequencer number, and *i* = 0, 1, 2 representing the LUT input index.

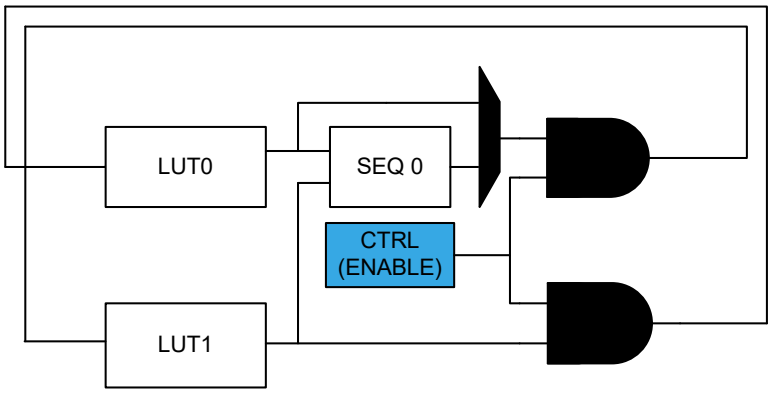
For details, refer to 28.3.2.6 Sequential Logic.

Figure 28-2. Feedback Input Selection



Linked LUT (LINK)
When selecting the LINK input option, the next LUT's direct output is used as the LUT input. In general, LUT[n+1] is linked to the input of LUT[n]. As an example, LUT1 is the input for LUT0. LUT0 is linked to the input of the last LUT.

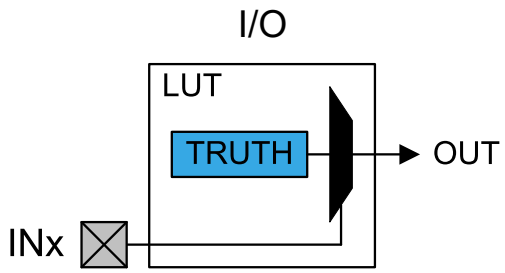
Figure 28-3. Linked LUT Input Selection



Internal Events Inputs Selection (EVENT)
Asynchronous events from the Event System can be used as input to the LUT. Two event input lines (EVENT0 and EVENT1) are available and can be selected as LUT input. Before selecting the EVENT input option by writing to the LUT Control B or C (CCL.LUTnCTRLB or LUTnCTRLC) register, the Event System must be configured.

I/O Pin Inputs (I/O)
When selecting the I/O option, the LUT input will be connected to its corresponding I/O pin. Refer to the *I/O Multiplexing* section for more details about where the LUTnINy is located.

Figure 28-4. I/O Pin Input Selection



Peripherals

The different peripherals on the three input lines of each LUT are selected by writing to the respective LUT n Input y bit fields in the LUT n Control B and C registers:

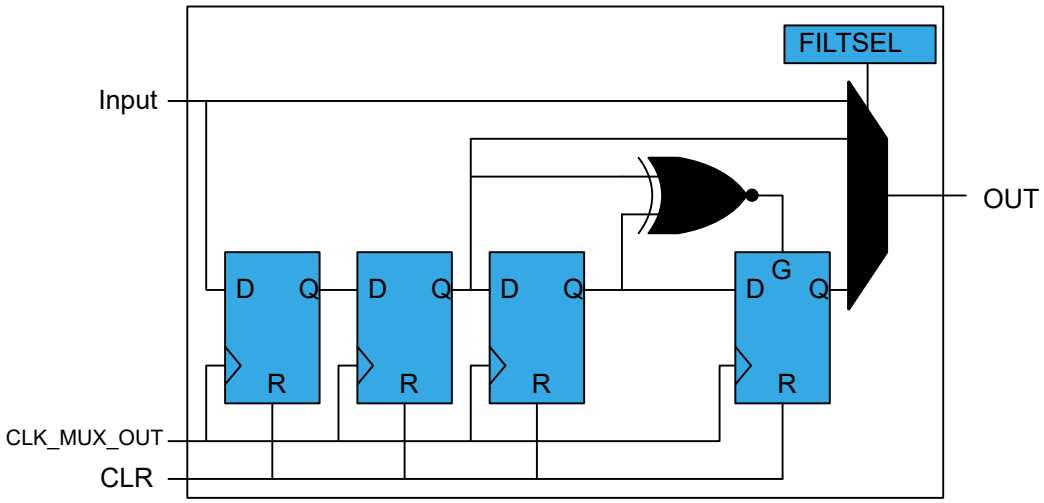
- INSEL0 in CCL.LUTnCTRLB
- INSEL1 in CCL.LUTnCTRLB
- INSEL2 in CCL.LUTnCTRLC

28.3.2.4 Filter

By default, the LUT output is a combinational function of the LUT inputs. This may cause some short glitches when the inputs change the value. These glitches can be removed by clocking through filters if demanded by application needs.

The Filter Selection (FILTSEL) bits in the LUT Control A (CCL.LUTnCTRLA) registers define the digital Filter options. When a filter is enabled, the output will be delayed by two to five CLK cycles (a peripheral clock or alternative clock). All internal Filter logic is cleared one clock cycle after the corresponding LUT is disabled.

Figure 28-5. Filter



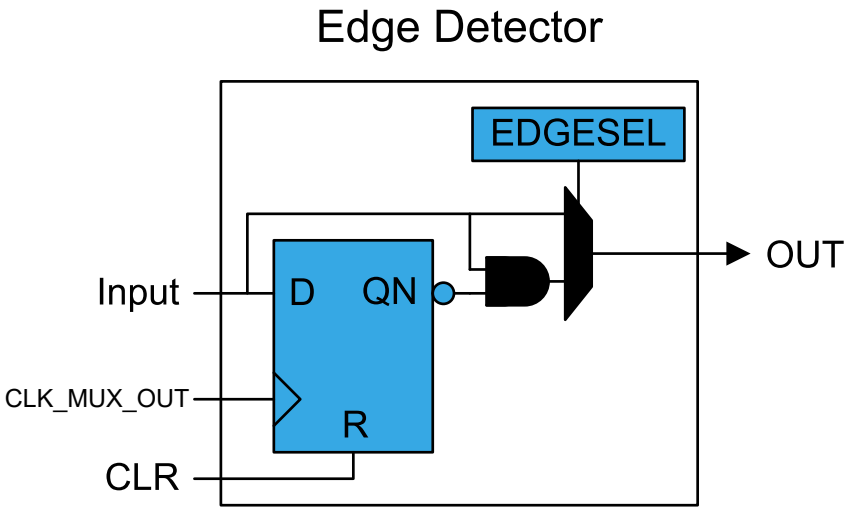
28.3.2.5 Edge Detector

The edge detector can be used to generate a pulse when detecting a rising edge on its input. To detect a falling edge, the truth table can be programmed to provide an inverted output.

The edge detector is enabled by writing a '1' to the Edge Selection (EDGEDET) bit in the LUT n Control A (CCL.LUTnCTRLA) register. To avoid unpredictable behavior, a valid filter option must be enabled as well.

The edge detection is disabled by writing a '0' to EDGEDET in CCL.LUTnCTRLA. After disabling an LUT, the corresponding internal edge detector logic is cleared one clock cycle later.

Figure 28-6. Edge Detector



28.3.2.6 Sequential Logic

Each LUT pair can be connected to an internal Sequential block. A Sequential block can function as either D flip-flop, JK flip-flop, gated D latch, or RS latch. The function is selected by writing the Sequential Selection (SEQSEL) bits in the Sequential Control (CCL.SEQCTRLn) register.

The Sequential block receives its input from either LUT, filter, or edge detector, depending on the configuration.

The Sequential block is clocked by the same clock as the corresponding LUT, which is either the peripheral clock or input line 2 (IN[2]). This is configured by the Clock Source (CLKSRC) bit in the LUT n Control A (CCL.LUTnCTRLA) register.

When the even LUT (LUT2n) is disabled, the latch is asynchronously cleared, during which the flip-flop Reset signal (R) is kept enabled for one clock cycle.

Gated D Flip-Flop (DFF)

The D-input is driven by the even LUT output (LUT2n), and the G-input is driven by the odd LUT output (LUT2n+1).

Figure 28-7. D Flip-Flop

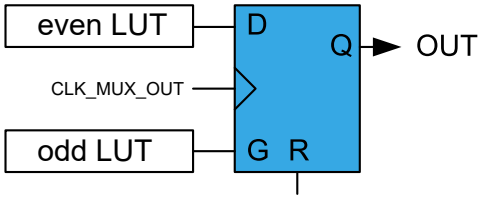


Table 28-3. DFF Characteristics

R	G	D	OUT
1	X	X	Clear
0	1	1	Set
		0	Clear
	0	X	Hold state (no change)

JK Flip-Flop (JK)

The J-input is driven by the even LUT output (LUT2n), and the K-input is driven by the odd LUT output (LUT2n+1).

Figure 28-8. JK Flip-Flop

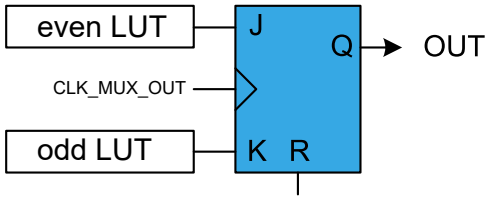


Table 28-4. JK Characteristics

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)

The D-input is driven by the even LUT output (LUT2n), and the G-input is driven by the odd LUT output (LUT2n+1).

Figure 28-9. D Latch

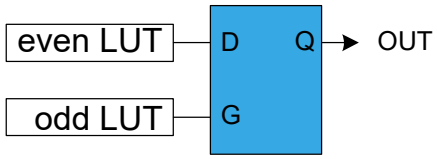


Table 28-5. D Latch Characteristics

G	D	OUT
0	X	Hold state (no change)
1	0	Clear
1	1	Set

RS Latch (RS)

The S-input is driven by the even LUT output (LUT2n), and the R-input is driven by the odd LUT output (LUT2n+1).

Figure 28-10. RS Latch

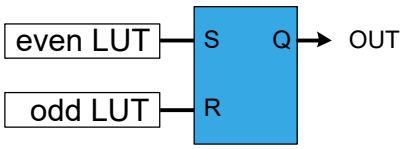


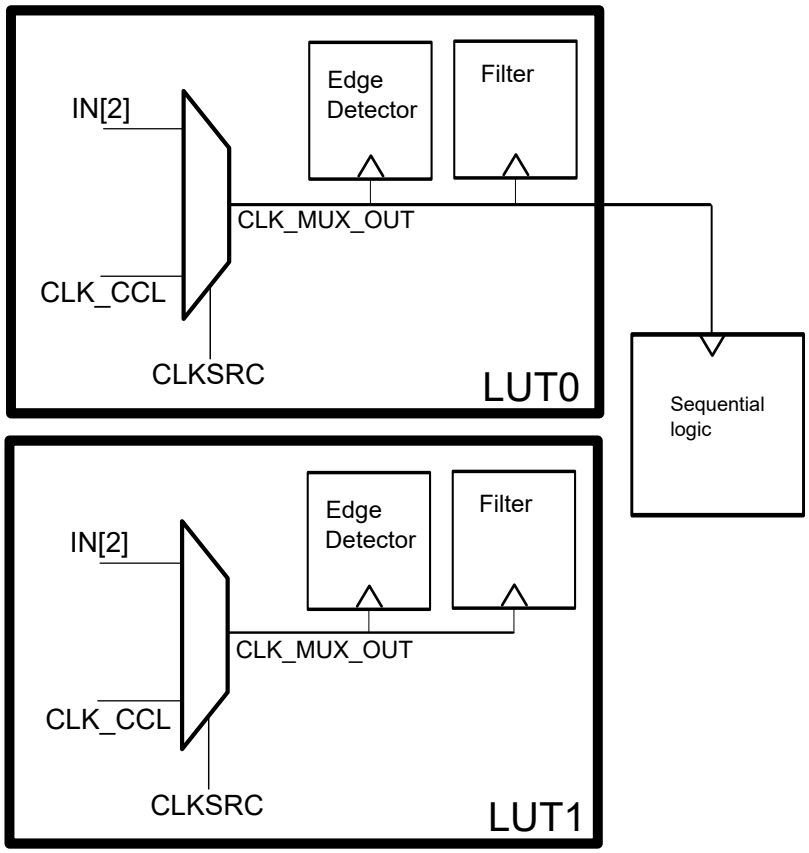
Table 28-6. RS Latch Characteristics

S	R	OUT
0	0	Hold state (no change)
0	1	Clear
1	0	Set
1	1	Forbidden state

28.3.2.7 Clock Source Settings

The Filter, edge detector, and Sequential logic are by default clocked by the system clock (CLK_PER). It is also possible to use the LUT input 2 (IN[2]) to clock these blocks (CLK_MUX_OUT in the figure below). This is configured by writing the Clock Source (CLKSRC) bit in the LUT Control A (CCL.LUTnCTRLA) register to '1'.

Figure 28-11. Clock Source Settings



When the Clock Source (CLKSRC) bit is '1', IN[2] is used to clock the corresponding filter and edge detector (CLK_MUX_OUT). The Sequential logic is clocked by CLK_MUX_OUT of the even LUT in the pair. When the CLKSRC bit is '1', IN[2] is treated as MASKed (low) in the truth table.

The CCL peripheral must be disabled while changing the clock source to avoid undetermined outputs from the peripheral.

28.3.3 Events

The CCL can generate the following output event:

- LUTnOUT: Look-up Table Output Value

The CCL can take the following actions on an input event:

- INx: The event is used as input for the truth table

28.3.4 Sleep Mode Operation

Writing the Run In Standby (RUNSTDBY) bit in the Control A (CCL.CTRLA) register to '1' will allow the system clock to be enabled in Standby sleep mode.

If RUNSTDBY is '0', the system clock will be disabled in Standby sleep mode. If the Filter, edge detector, or Sequential logic is enabled, the LUT output will be forced to '0' in Standby sleep mode. In Idle sleep mode, the truth table decoder will continue operation, and the LUT output will be refreshed accordingly, regardless of the RUNSTDBY bit.

If the Clock Source (CLKSRC) bit in the LUT n Control A (CCL.LUTnCTRLA) register is written to '1', the LUT input 2 (IN[2]) will always clock the Filter, edge detector, and Sequential block. The availability of the IN[2] clock in sleep modes will depend on the sleep settings of the peripheral employed.

28.3.5 Configuration Change Protection

Not applicable.

28.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		RUNSTDBY						ENABLE
0x01	SEQCTRL0	7:0					SEQSEL[3:0]			
0x02 ... 0x04	Reserved									
0x05	LUT0CTRLA	7:0	EDGEDET	CLKSRC	FILTSEL[1:0]		OUTEN			ENABLE
0x06	LUT0CTRLB	7:0	INSEL1[3:0]				INSEL0[3:0]			
0x07	LUT0CTRLC	7:0					INSEL2[3:0]			
0x08	TRUTH0	7:0	TRUTH[7:0]							
0x09	LUT1CTRLA	7:0	EDGEDET	CLKSRC	FILTSEL[1:0]		OUTEN			ENABLE
0x0A	LUT1CTRLB	7:0	INSEL1[3:0]				INSEL0[3:0]			
0x0B	LUT1CTRLC	7:0					INSEL2[3:0]			
0x0C	TRUTH1	7:0	TRUTH[7:0]							

28.5 Register Description

28.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

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

Bit 6 – RUNSTDBY Run in Standby
This bit indicates if the peripheral clock (CLK_PER) is kept running in Standby sleep mode. The setting is ignored for configurations where the CLK_PER is not required.

Value	Description
0	System clock is not required in Standby sleep mode
1	System clock is required in Standby sleep mode

Bit 0 – ENABLE Enable

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

28.5.2 Sequential Control 0

Name: SEQCTRL0
Offset: 0x01 [ID-00000485]
Reset: 0x00
Property: Enable-Protected

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

Bits 3:0 – SEQSEL[3:0] Sequential Selection
This bit field selects the sequential configuration for LUT0 and LUT1.

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
Other	-	Reserved

28.5.3 LUT n Control A

Name: LUTCTRLA
Offset: 0x05 + n*0x04 [n=0..1]
Reset: 0x00
Property: Enable-Protected

Bit	7	6	5	4	3	2	1	0
	EDGEDET	CLKSRC	FILTSEL[1:0]		OUTEN			ENABLE
Access	R/W	R/W	R/W	R/W	R/W			R/W
Reset	0	0	0	0	0			0

Bit 7 – EDGEDET Edge Detection

Value	Description
0	Edge detector is disabled
1	Edge detector is enabled

Bit 6 – CLKSRC Clock Source Selection
This bit selects whether the peripheral clock (CLK_PER) or any input present on input line 2 (IN[2]) is used as clock (CLK_MUX_OUT) for an LUT.
The CLK_MUX_OUT of the even LUT is used for clocking the Sequential block of an LUT pair.

Value	Description
0	CLK_PER is clocking the LUT
1	IN[2] is clocking the LUT

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

Bit 3 – OUTEN Output Enable
This bit enables the LUT output to the LUTnOUT pin. When written to '1', the pin configuration of the PORT I/O Controller is overridden.

Value	Description
0	Output to pin disabled
1	Output to pin enabled

Bit 0 – ENABLE LUT Enable

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

28.5.4 LUT n Control B

Name: LUTCTRLB
Offset: 0x06 + n*0x04 [n=0..1]
Reset: 0x00
Property: Enable-Protected

Note:
1. SPI connections to the CCL work only in master SPI mode.
2. USART connections to the CCL work only in asynchronous/synchronous USART Master mode.

Bit	7	6	5	4	3	2	1	0
	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

Bits 7:4 – INSEL1[3:0] LUT n Input 1 Source Selection
This bit field selects the source for input 1 of LUT n:

Value	Name	Description
0x0	MASK	Masked input
0x1	FEEDBACK	Feedback input
0x2	LINK	Linked other LUT as input source
0x3	EVENT0	Event input source 0
0x4	EVENT1	Event input source 1
0x5	IO	I/O-pin LUTn-IN1 input source
0x6	AC0	AC0 OUT input source
0x7	TCB0	TCB0 WO input source
0x8	TCA0	TCA0 WO1 input source
0x9	TCD0	TCD0 WOB input source
0xA	USART0	USART0 TXD input source
0xB	SPI0	SPI0 MOSI input source
0xC	AC1	AC1 OUT input source
0xD	TCB1	TCB1 WO input source
0xE	AC2	AC2 OUT input source
Other	-	Reserved

Bits 3:0 – INSEL0[3:0] LUT n Input 0 Source Selection
This bit field selects the source for input 0 of LUT n:

Value	Name	Description
0x0	MASK	Masked input
0x1	FEEDBACK	Feedback input
0x2	LINK	Linked other LUT as input source
0x3	EVENT0	Event input source 0
0x4	EVENT1	Event input source 1
0x5	IO	I/O-pin LUTn-IN0 input source
0x6	AC0	AC0 OUT input source
0x7	TCB0	TCB0 WO input source
0x8	TCA0	TCA0 WO0 input source
0x9	TCD0	TCD0 WOA input source
0xA	USART0	USART0 XCK input source
0xB	SPI0	SPI0 SCK input source
0xC	AC1	AC1 OUT input source
0xD	TCB1	TCB1 WO input source
0xE	AC2	AC2 OUT input source

.....continued		
Value	**Name**	**Description**
Other	-	Reserved

28.5.5 LUT n Control C

Name: LUTCTRLC
Offset: 0x07 + n*0x04 [n=0..1]
Reset: 0x00
Property: Enable-Protected

Bit	7	6	5	4	3	2	1	0
					INSEL2[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 3:0 – INSEL2[3:0] LUT n Input 2 Source Selection
This bit field selects the source for input 2 of LUT n:

Value	Name	Description
0x0	MASK	Masked input
0x1	FEEDBACK	Feedback input
0x2	LINK	Linked other LUT as input source
0x3	EVENT0	Event input source 0
0x4	EVENT1	Event input source 1
0x5	IO	I/O-pin LUTn-IN2 input source
0x6	AC0	AC0 OUT input source
0x7	TCB0	TCB0 WO input source
0x8	TCA0	TCA0 WO2 input source
0x9	TCD0	TCD0 WOA input source
0xA	-	Reserved
0xB	SPI0	SPI0 MISO input source
0xC	AC1	AC1 OUT input source
0xD	TCB1	TCB1 WO input source
0xE	AC2	AC2 OUT input source
Other	-	Reserved

28.5.6 TRUTHn

Name: TRUTH
Offset: 0x08 + n*0x04 [n=0..1]
Reset: 0x00
Property: Enable-Protected

Bit	7	6	5	4	3	2	1	0
	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

Bits 7:0 – TRUTH[7:0] Truth Table
This bit field defines the value of the Truth logic as a function of inputs IN[2:0].

29. AC - Analog Comparator

29.1 Features

- 50 ns Response Time for Supply Voltage Above 2.7V
- Zero-Cross Detection
- Selectable Hysteresis:
  - None
  - 10 mV
  - 25 mV
  - 50 mV
- Analog Comparator Output Available on Pin
- Comparator Output Inversion Available
- Flexible Input Selection:
  - Four Positive pins
  - Two Negative pins
  - Output from the DAC
  - Internal reference voltage
- Interrupt Generation On:
  - Rising edge
  - Falling edge
  - Both edges
- Event Generation:
  - Comparator output

29.2 Overview

The Analog Comparator (AC) compares the voltage levels on two inputs and gives a digital output based on this comparison. The AC can be configured to generate interrupt requests and/or events upon several different combinations of input change.

The dynamic behavior of the AC can be adjusted by a hysteresis feature. The hysteresis can be customized to optimize the operation for each application.

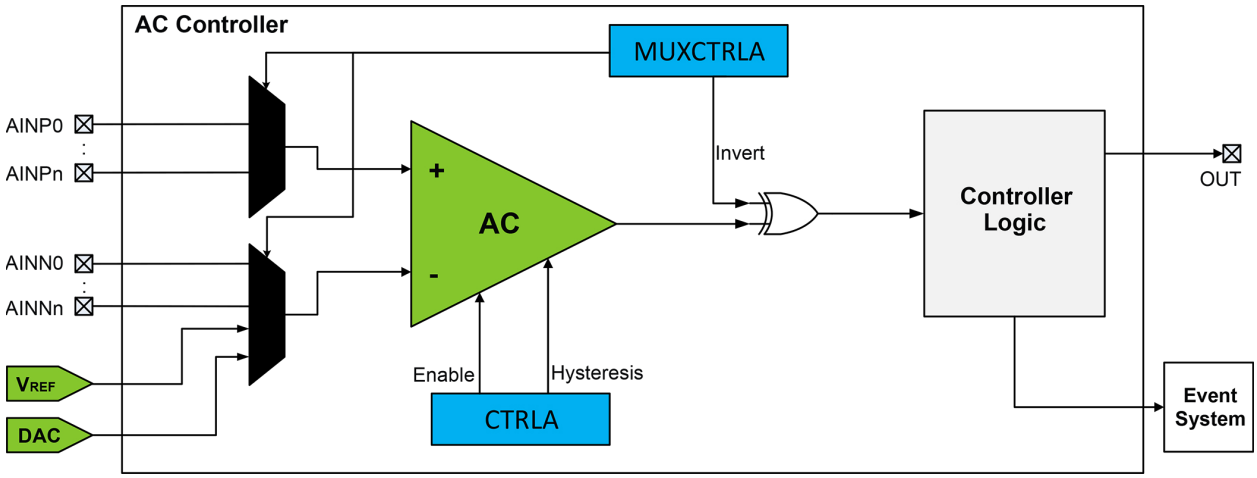
The input selection includes analog port pins, DAC output and internal reference voltage. The analog comparator output state can also be output on a pin for use by external devices.

An AC has one positive input and one negative input. The positive input source is one of the analog input pins. The negative input is chosen either from analog input pins or from internal inputs, such as an internal voltage reference.

The digital output from the comparator is '1' when the difference between the positive and the negative input voltage is positive, and '0' otherwise.

29.2.1 Block Diagram

Figure 29-1. Analog Comparator



Note: Refer to 29.1 Features for the number of AINN and AINP.

29.2.2 Signal Description

Signal	Description	Type
AINNn	Negative Input n	Analog
AINPn	Positive Input n	Analog
OUT	Comparator Output for AC	Digital

29.2.3 System Dependencies

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

Table 29-1. AC System Dependencies

Dependency	Applicable	Peripheral
Clocks	Yes	CLKCTRL
I/O Lines and Connections	Yes	PORT
Interrupts	Yes	CPUINT
Events	Yes	EVSYS
Debug	Yes	UPDI

29.2.3.1 Clocks

This peripheral depends on the peripheral clock.

29.2.3.2 I/O Lines and Connections

The I/O pins AINNn and AINPn are all analog inputs to the AC.

For correct operation, the pins must be configured in the port and port multiplexing peripherals.

It is recommended to disable the GPIO input when using the AC.

29.2.3.3 Interrupts

Using the interrupts of this peripheral requires the interrupt controller to be configured first.

29.2.3.4 Events

The events of this peripheral are connected to the Event System.

29.2.3.5 Debug Operation

This peripheral is unaffected by entering Debug mode.

If the peripheral is configured to require periodic service by the CPU through interrupts or similar, improper operation or data loss may result during halted debugging.

29.3 Functional Description

29.3.1 Initialization

For basic operation, follow these steps:

1. Configure the desired input pins in the port peripheral.
2. Select the positive and negative input sources by writing the Positive and Negative Input MUX Selection (MUXPOS and MUXNEG) bit fields in the MUX Control A (ACn.MUXCTRLA) register.
3. Optional: Enable the output to pin by writing a '1' to the Output Pad Enable (OUTEN) bit in the Control A (ACn.CTRLA) register.
4. Enable the AC by writing a '1' to the ENABLE bit in the ACn.CTRLA register.

During the start-up time after enabling the AC, the output of the AC may be invalid.

The start-up time of the AC by itself is at most 2.5 µs. If an internal reference is used, the reference start-up time is normally longer than the AC start-up time. The VREF start-up time is 60 µs at most.

29.3.2 Operation

29.3.2.1 Input Hysteresis

Applying an input hysteresis helps to prevent constant toggling of the output when the noise-afflicted input signals are close to each other.

The input hysteresis can either be disabled or have one of three levels. The hysteresis is configured by writing to the Hysteresis Mode Select (HYSMODE) bit field in the Control A (ACn.CTRLA) register.

29.3.2.2 Input Sources

An AC has one positive input and one negative input. The inputs can be pins and internal sources, such as a voltage reference.

Each input is selected by writing to the Positive and Negative Input MUX Selection (MUXPOS and MUXNEG) bit field in the MUX Control A (ACn.MUXCTRLA) register.

29.3.2.2.1 Pin Inputs

The following analog input pins on the port can be selected as input to the analog comparator:

- AINN0
- AINN1
- AINP0
- AINP1
- AINP2
- AINP3

29.3.2.2.2 Internal Inputs

The AC has the following internal inputs:

- Output from the DAC
- AC voltage reference

29.3.2.3 Low-Power Mode

For power sensitive applications, the AC provides a Low-Power mode with reduced power consumption and increased propagation delay.

This mode is enabled by writing a '1' to the Low-Power Mode (LPMODE) bit in the Control A (ACn.CTRLA) register.

29.3.3 Events

The AC will generate the following event automatically when the AC is enabled:

- The digital output from the AC (OUT in the block diagram) is available as an Event System source. The events from the AC are asynchronous to any clocks in the device.

The AC has no event inputs.

29.3.4 Interrupts

Table 29-2. Available Interrupt Vectors and Sources

Offset	Name	Vector Description	Conditions
0x00	COMP0	Analog comparator interrupt	AC output is toggling as configured by INTMODE in ACn.CTRLA

When an Interrupt condition occurs, the corresponding Interrupt flag is set in the STATUS (ACn.STATUS) register.

An interrupt source is enabled or disabled by writing to the corresponding bit in the peripheral's Interrupt Control (ACn.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled and the Interrupt flag is set. The interrupt request remains active until the Interrupt flag is cleared. See the ACn.STATUS register description for details on how to clear Interrupt flags.

29.3.5 Sleep Mode Operation

In Idle sleep mode, the AC will continue to operate as normal.

In Standby sleep mode, the AC is disabled by default. If the Run in Standby Mode (RUNSTDBY) bit in the Control A (ACn.CTRLA) register is written to '1', the AC will continue to operate, but the Status register will not be updated, and no interrupts are generated if no other modules request the CLK_PER, but events and the pad output will be updated.

In Power-Down sleep mode, the AC and the output to the pad are disabled.

29.3.6 Configuration Change Protection

Not applicable.

29.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY	OUTEN	INTMODE[1:0]		LPMODE	HYSMODE[1:0]		ENABLE
0x01	Reserved									
0x02	MUXCTRLA	7:0	INVERT			MUXPOS[1:0]			MUXNEG[1:0]	
0x03 ... 0x05	Reserved									
0x06	INTCTRL	7:0								CMP
0x07	STATUS	7:0				STATE				CMP

29.5 Register Description

29.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY	OUTEN	INTMODE[1:0]		LPMODE	HYSMODE[1:0]		ENABLE
Access	R/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 – RUNSTDBY Run in Standby Mode
Writing a '1' to this bit allows the AC to continue operation in Standby sleep mode. Since the clock is stopped, interrupts and status flags are not updated.

Value	Description
0	In Standby sleep mode, the peripheral is halted
1	In Standby sleep mode, the peripheral continues operation

Bit 6 – OUTEN Analog Comparator Output Pad Enable
Writing this bit to '1' makes the OUT signal available on the pin.

Bits 5:4 – INTMODE[1:0] Interrupt Modes
Writing to this bit field selects what edges of the AC output triggers an interrupt request.

Value	Name	Description
0x0	BOTHEDGE	Both negative and positive edge
0x1	-	Reserved
0x2	NEGEDGE	Negative edge
0x3	POSEDGE	Positive edge

Bit 3 – LPMODE Low-Power Mode
Writing a '1' to this bit reduces the current through the comparator. This reduces the power consumption but increases the reaction time of the AC.

Value	Description
0	Low-Power mode disabled
1	Low-Power mode enabled

Bits 2:1 – HYSMODE[1:0] Hysteresis Mode Select
Writing to this bit field selects the Hysteresis mode for the AC input.

Value	Name	Description
0x0	OFF	OFF
0x1	10	±10 mV
0x2	25	±25 mV
0x3	50	±50 mV

Bit 0 – ENABLE Enable AC
Writing this bit to '1' enables the AC.

29.5.2 MUX Control A

Name: MUXCTRLA
Offset: 0x02
Reset: 0x00
Property: -

ACn.MUXCTRLA controls the analog comparator MUXes.

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

Bit 7 – INVERT Invert AC Output
Writing a '1' to this bit enables inversion of the output of the AC. This effectively inverts the input to all the peripherals connected to the signal and affects the internal status signals.

Bits 4:3 – MUXPOS[1:0] Positive Input MUX Selection
Writing to this bit field selects the input signal to the positive input of the AC.

Value	Name	Description
0x0	AINP0	Positive pin 0
0x1	AINP1	Positive pin 1
0x2	AINP2	Positive pin 2
0x3	AINP3	Positive pin 3

Bits 1:0 – MUXNEG[1:0] Negative Input MUX Selection
Writing to this bit field selects the input signal to the negative input of the AC.

Value	Name	Description
0x0	AINN0	Negative pin 0
0x1	AINN1	Negative pin 1
0x2	VREF	Voltage Reference
0x3	DAC	DAC output Instance n of the AC will use instance n of the DAC: for example, AC0 will use DAC0 and AC1 will use DAC1

29.5.3 Interrupt Control

Name: INTCTRL
Offset: 0x06
Reset: 0x00
Property: -

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

Bit 0 – CMP Analog Comparator Interrupt Enable
Writing this bit to '1' enables analog comparator interrupt.

29.5.4 Status

Name: STATUS
Offset: 0x07
Reset: 0x00
Property: -

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

Bit 4 – STATE Analog Comparator State
This shows the current status of the OUT signal from the AC. This will have a synchronizer delay to get updated in the I/O register (three cycles).

Bit 0 – CMP Analog Comparator Interrupt Flag
This is the Interrupt flag for AC. Writing a '1' to this bit will clear the Interrupt flag.

30. ADC - Analog-to-Digital Converter

30.1 Features

- 10-Bit Resolution
- 0V to V_{DD} Input Voltage Range
- Multiple Internal ADC Reference Voltages
- External Reference Input
- Free-Running and Single Conversion Mode
- Interrupt Available on Conversion Complete
- Optional Interrupt on Conversion Results
- Temperature Sensor Input Channel
- Optional Event-Triggered Conversion
- Window Comparator Function for Accurate Monitoring or Defined Thresholds
- Accumulation up to 64 Samples per Conversion

30.2 Overview

The Analog-to-Digital Converter (ADC) peripheral produces 10-bit results. The ADC input can either be internal (e.g., a voltage reference) or external through the analog input pins. The ADC is connected to an analog multiplexer, which allows the selection of multiple single-ended voltage inputs. The single-ended voltage inputs refer to 0V (GND).

The ADC supports sampling in bursts where a configurable number of conversion results are accumulated into a single ADC result (Sample Accumulation). Further, a sample delay can be configured to tune the ADC sampling frequency associated with a single burst. This is to tune the sampling frequency away from any harmonic noise aliased with the ADC sampling frequency (within the burst) from the sampled signal. An automatic sampling delay variation feature can be used to randomize this delay to slightly change the time between samples.

The ADC input signal is fed through a sample-and-hold circuit that ensures that the input voltage to the ADC is held at a constant level during sampling.

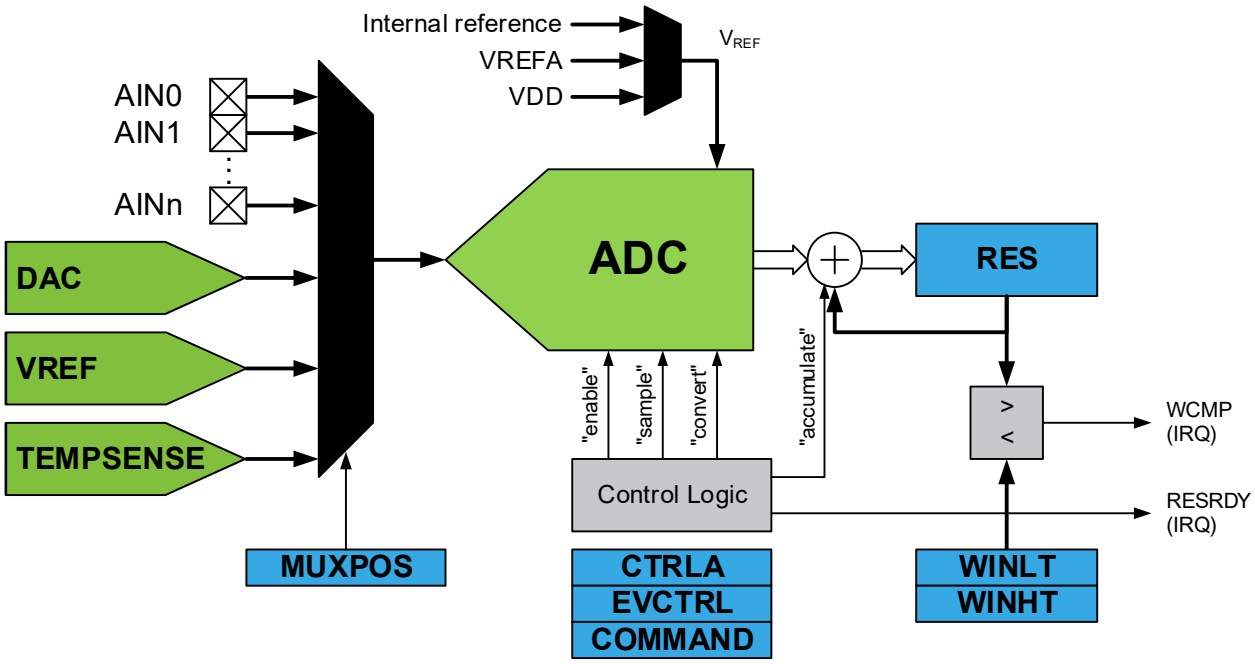
The selectable voltage references from the internal Voltage Reference (VREF) peripheral, are V_{DD} supply voltage, or external VREF pin (VREFA).

A window compare feature is available for monitoring the input signal and can be configured to only trigger an interrupt on user-defined thresholds for under, over, inside, or outside a window, with minimum software intervention required.

When the Peripheral Touch Controller (PTC) is enabled, ADC0 is fully controlled by the PTC peripheral.

30.2.1 Block Diagram

Figure 30-1. ADC Block Diagram



The analog input channel is selected by writing to the MUXPOS bits in the MUXPOS (ADCn.MUXPOS) register. Any of the ADC input pins, GND, internal Voltage Reference (V_{REF}), or temperature sensor, can be selected as a single-ended input to the ADC. The ADC is enabled by writing a '1' to the ADC ENABLE bit in the Control A (ADCn.CTRLA) register. The voltage reference and input channel selections will not go into effect before the ADC is enabled. The ADC does not consume power when the ENABLE bit in ADCn.CTRLA is '0'.

The ADC generates a 10-bit result that can be read from the Result (ADCn.RES) Register. The result is presented right-adjusted.

30.2.2 Signal Description

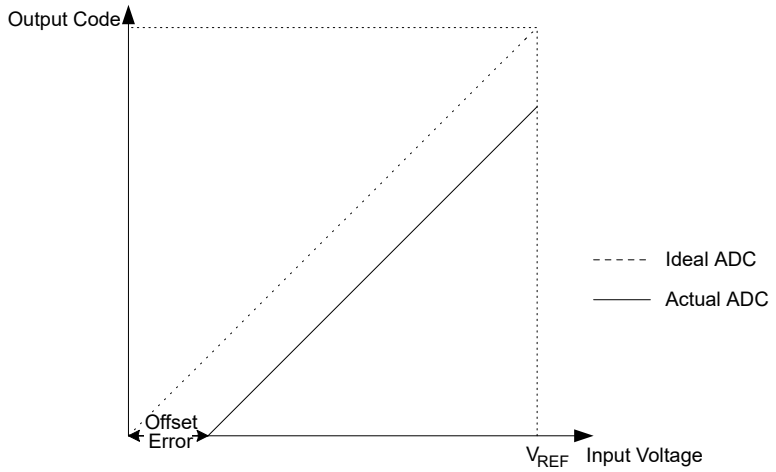
Pin Name	Type	Description
AIN[n:0]	Analog input	Analog input pin
VREFA	Analog input	External voltage reference pin

30.2.2.1 Definitions

An ideal n-bit single-ended ADC converts a voltage linearly between GND and V_{REF} in 2^n steps (LSbs). The lowest code is read as '0', and the highest code is read as 2^{n-1}. Several parameters describe the deviation from the ideal behavior.

Offset Error — The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSb). Ideal value: 0 LSb.

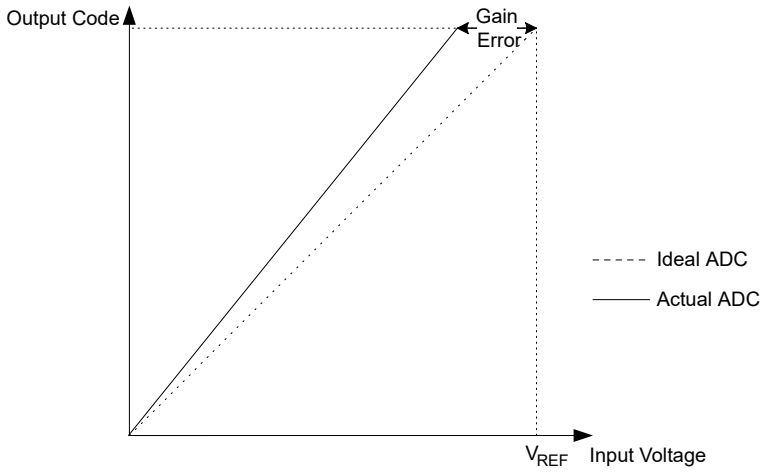
Figure 30-2. Offset Error



Gain Error

After adjusting for offset, the gain error is found as the deviation of the last transition (0x3FE to 0x3FF) compared to the ideal transition (at 1.5 LSb below maximum). Ideal value: 0 LSb.

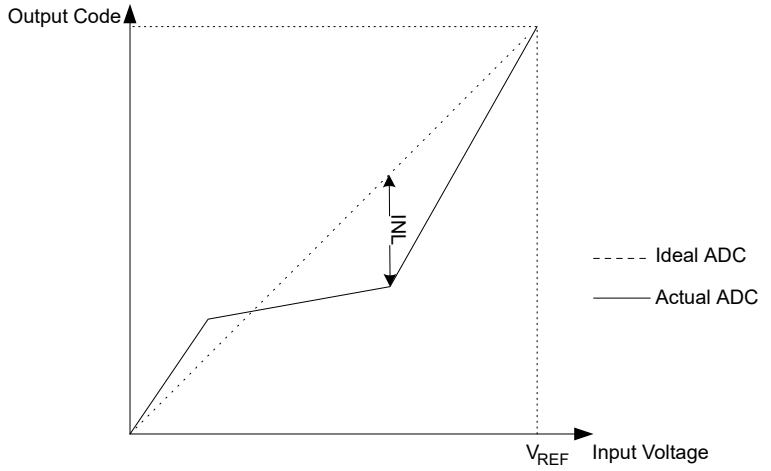
Figure 30-3. Gain Error



Integral Nonlinearity (INL)

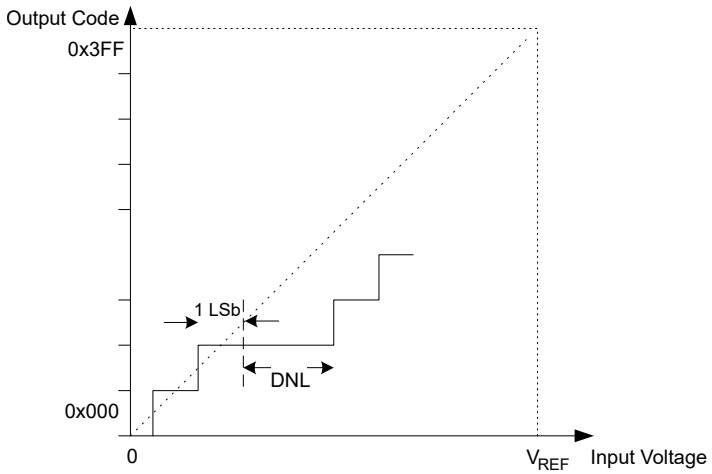
After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSb.

Figure 30-4. Integral Nonlinearity



Differential Nonlinearity (DNL) The maximum deviation of the actual code width (the interval between two adjacent transitions) from the ideal code width (1 LSb). Ideal value: 0 LSb.

Figure 30-5. Differential Nonlinearity



Quantization Error Due to the quantization of the input voltage into a finite number of codes, a range of input voltages (1 LSb wide) will code to the same value. Always ±0.5 LSb.

Absolute Accuracy The maximum deviation of an actual (unadjusted) transition compared to an ideal transition for any code. This is the compound effect of all aforementioned errors. Ideal value: ±0.5 LSb.

30.3 Functional Description

30.3.1 Initialization

The following steps are recommended to initialize the ADC operation:

1. Configure the resolution by writing to the Resolution Selection (RESSEL) bit in the Control A (ADCn.CTRLA) register.
2. Optional: Enable the Free-Running mode by writing a '1' to the Free-Running (FREERUN) bit in ADCn.CTRLA.
3. Optional: Configure the number of samples to be accumulated per conversion by writing the Sample Accumulation Number Select (SAMPNUM) bits in the Control B (ADCn.CTRLB) register.
4. Configure a voltage reference by writing to the Reference Selection (REFSEL) bit in the Control C (ADCn.CTRLC) register. The default is the internal voltage reference of the device (V_{REF}, as configured there).
5. Configure the CLK_ADC by writing to the Prescaler (PRESC) bit field in the Control C (ADCn.CTRLC) register.
6. Configure an input by writing to the MUXPOS bit field in the MUXPOS (ADCn.MUXPOS) register.
7. Optional: Enable Start Event input by writing a '1' to the Start Event Input (STARTEI) bit in the Event Control (ADCn.EVCTRL) register. Configure the Event System accordingly.
8. Enable the ADC by writing a '1' to the ENABLE bit in ADCn.CTRLA.

Following these steps will initialize the ADC for basic measurements, which can be triggered by an event (if configured) or by writing a '1' to the Start Conversion (STCONV) bit in the Command (ADCn.COMMAND) register.

30.3.1.1 I/O Lines and Connections

The I/O pins AINx and VREF are configured by the port - I/O Pin Controller.

The digital input buffer should be disabled on the pin used as input for the ADC to disconnect the digital domain from the analog domain to obtain the best possible ADC results. This is configured by the PORT peripheral.

30.3.2 Operation

30.3.2.1 Starting a Conversion

Once the input channel is selected by writing to the MUXPOS (ADCn.MUXPOS) register, a conversion is triggered by writing a '1' to the ADC Start Conversion (STCONV) bit in the Command (ADCn.COMMAND) register. This bit is '1' as long as the conversion is in progress. In Single Conversion mode, STCONV is cleared by hardware when the conversion is completed.

If a different input channel is selected while a conversion is in progress, the ADC will finish the current conversion before changing the channel.

Depending on the accumulator setting, the conversion result is from a single sensing operation or a sequence of accumulated samples. Once the triggered operation is finished, the Result Ready (RESRDY) flag in the Interrupt Flag (ADCn.INTFLAG) register is set. The corresponding interrupt vector is executed if the Result Ready Interrupt Enable (RESRDY) bit in the Interrupt Control (ADCn.INTCTRL) register is '1' and the Global Interrupt Enable bit is '1'.

A single conversion can be started by writing a '1' to the STCONV bit in ADCn.COMMAND. The STCONV bit can be used to determine if a conversion is in progress. The STCONV bit will be set during a conversion and cleared once the conversion is complete.

The RESRDY interrupt flag in ADCn.INTFLAG will be set even if the specific interrupt is disabled, allowing software to check for finished conversion by polling the flag. A conversion can thus be triggered without causing an interrupt.

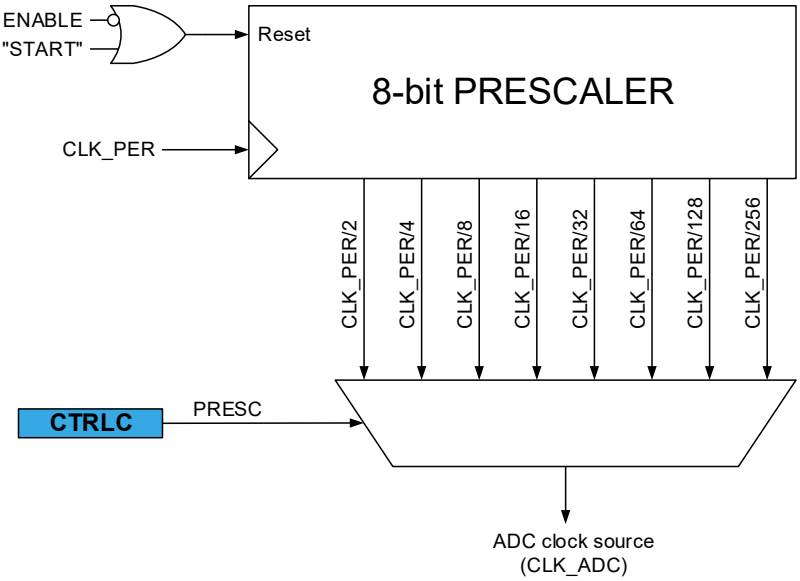
Alternatively, a conversion can be triggered by an event. This is enabled by writing a '1' to the Start Event Input (STARTEI) bit in the Event Control (ADCn.EVCTRL) register. Any incoming event routed to the ADC through the Event System (EVSYS) will trigger an ADC conversion. This provides a method to start conversions at predictable intervals or specific conditions.

The event trigger input is edge sensitive. When an event occurs, STCONV in ADCn.COMMAND is set. STCONV will be cleared when the conversion is complete.

In Free-Running mode, the first conversion is started by writing the STCONV bit to '1' in ADCn.COMMAND. A new conversion cycle is started immediately after the previous conversion cycle has completed. A conversion complete will set the RESRDY flag in ADCn.INTFLAGS.

30.3.2.2 Clock Generation

Figure 30-6. ADC Prescaler



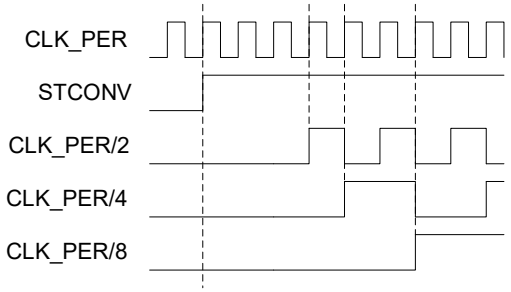
The ADC requires an input clock frequency between 50 kHz and 1.5 MHz for maximum resolution. If a lower resolution than ten bits is selected, the input clock frequency to the ADC can be higher than 1.5 MHz to get a higher sample rate.

The ADC module contains a prescaler which generates the ADC clock (CLK_ADC) from any CPU clock (CLK_PER) above 100 kHz. The prescaling is selected by writing to the Prescaler (PRESC) bits in the Control C (ADCn.CTRLC) register. The prescaler starts counting from the moment the ADC is switched on by writing a '1' to the ENABLE bit in ADCn.CTRLA. The prescaler keeps running as long as the ENABLE bit is '1'. The prescaler counter is reset to zero when the ENABLE bit is '0'.

When initiating a conversion by writing a '1' to the Start Conversion (STCONV) bit in the Command (ADCn.COMMAND) register or from an event, the conversion starts at the following rising edge of the CLK_ADC clock cycle. The prescaler is kept reset as long as there is no ongoing conversion. This assures a fixed delay from the trigger to the actual start of a conversion in CLK_PER cycles, as follows:

$$\text{StartDelay} = \frac{\text{PRESC}_{\text{factor}}}{2} + 2$$

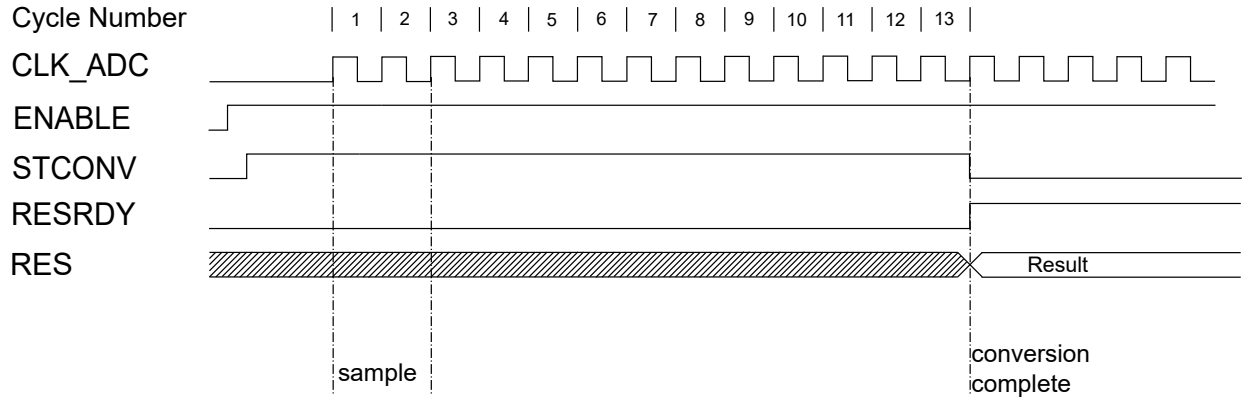
Figure 30-7. Start Conversion and Clock Generation



30.3.2.3 Conversion Timing

A normal conversion takes 13 CLK_ADC cycles. The actual sample-and-hold takes place two CLK_ADC cycles after the start of a conversion. The start of a conversion is initiated by writing a '1' to the STCONV bit in ADCn.COMMAND. When a conversion is complete, the result is available in the Result (ADCn.RES) register, and the Result Ready interrupt flag is set (RESRDY in ADCn.INTFLAG). The interrupt flag will be cleared when the result is read from the Result registers, or by writing a '1' to the RESRDY bit in ADCn.INTFLAG.

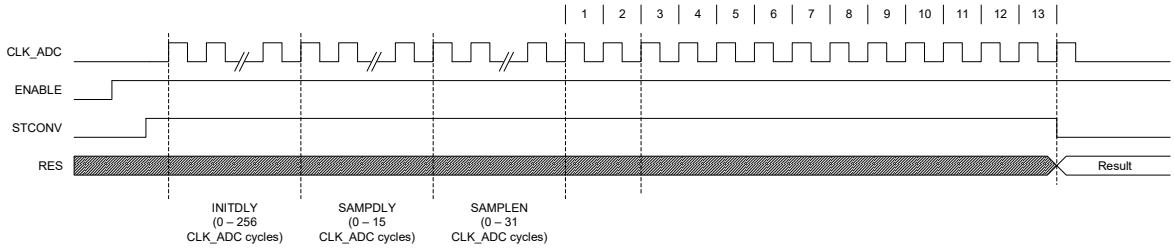
Figure 30-8. ADC Timing Diagram - Single Conversion



Both sampling time and sampling length can be adjusted using the Sample Delay bit field in the Control D (ADCn.CTRLD) register and the Sample Length bit field in the Sample Control (ADCn.SAMPCTRL) register. Both of these control the ADC sampling time in some CLK_ADC cycles. This allows sampling of high-impedance sources without relaxing conversion speed. See the register description for further information. Total sampling time is given by:

$$\text{SampleTime} = \frac{(2 + \text{SAMPDLY} + \text{SAMPLEN})}{f_{\text{CLK\_ADC}}}$$

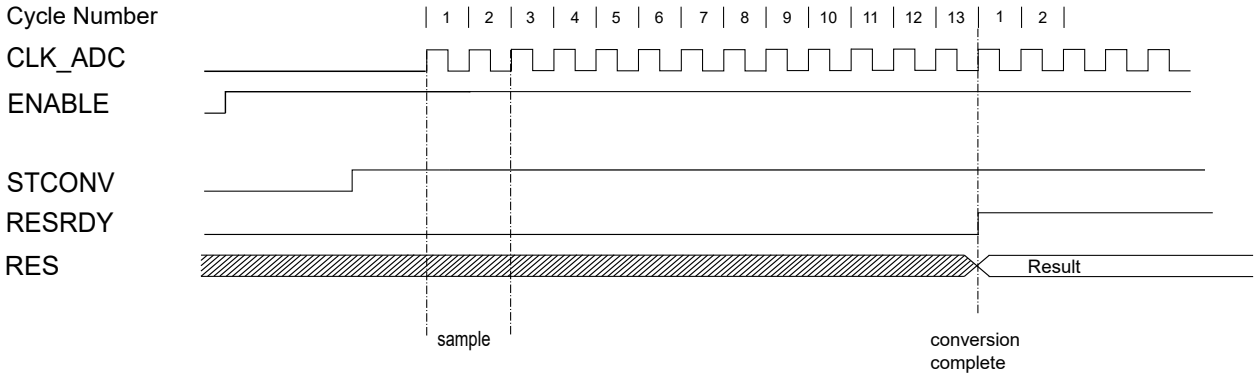
Figure 30-9. ADC Timing Diagram - Single Conversion With Delays



In Free-Running mode, a new conversion will be started immediately after the conversion completes, while the STCONV bit is '1'. The sampling rate R_S in Free-Running mode is calculated by:

$$R_S = \frac{f_{CLK\_ADC}}{(13 + SAMPDLY + SAMPLEN)}$$

Figure 30-10. ADC Timing Diagram - Free-Running Conversion



30.3.2.4 Changing Channel or Reference Selection

The MUXPOS bits in the ADCn.MUXPOS register and the REFSEL bits in the ADCn.CTRLC register are buffered through a temporary register to which the CPU has random access. This ensures that the channel and reference selections only take place at a safe point during the conversion. The channel and reference selections are continuously updated until a conversion is started.

Once the conversion starts, the channel and reference selections are locked to ensure sufficient sampling time for the ADC. Continuous updating resumes in the last CLK_ADC clock cycle before the conversion completes (RESRDY in ADCn.INTFLAGS is set). The conversion starts on the following rising CLK_ADC clock edge after the STCONV bit is written to '1'.

30.3.2.4.1 ADC Input Channels

When changing channel selection, the user must observe the following guidelines to ensure that the correct channel is selected:

In Single Conversion mode: The channel should be selected before starting the conversion. The channel selection may be changed one ADC clock cycle after writing '1' to the STCONV bit.

In Free-Running mode: The channel should be selected before starting the first conversion. The channel selection may be changed one ADC clock cycle after writing '1' to the STCONV bit. Since the next conversion has already started automatically, the next result will reflect the previous channel selection. The subsequent conversions will reflect the new channel selection.

The ADC requires a settling time after switching the input channel - refer to the Electrical Characteristics section for details.

30.3.2.4.2 ADC Voltage Reference

The reference voltage for the ADC (V_{REF}) controls the conversion range of the ADC. Input voltages that exceed the selected V_{REF} will be converted to the maximum result value of the ADC. For an ideal 10-bit ADC, this value is 0x3FF.

V_{REF} can be selected by writing the Reference Selection (REFSEL) bits in the Control C (ADCn.CTRLC) register as either V_{DD}, external reference V_{REFA}, or an internal reference from the VREF peripheral. V_{DD} is connected to the ADC through a passive switch.

When using the external reference voltage V_{REFA}, configure ADCnREFSEL[0:2] in the corresponding VREF.CTRLn register to the value that is closest, but above the applied reference voltage. For external references higher than 4.3V, use ADCnREFSEL[0:2] = `0x3`.

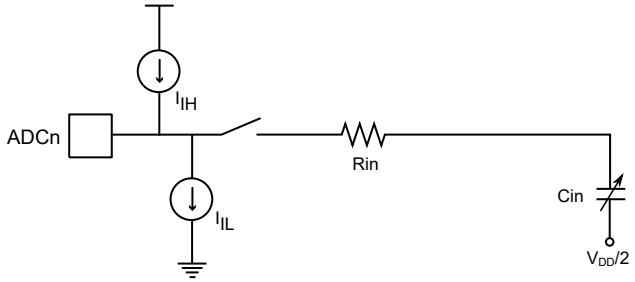
The internal reference is generated from an internal band gap reference through an internal amplifier, controlled by the Voltage Reference (VREF) peripheral.

30.3.2.4.3 Analog Input Circuitry

The analog input circuitry is illustrated in Figure 30-11. An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin (represented by I_H and I_L), regardless of whether that channel is selected as input for the ADC or not. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10 kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long the source needs to charge the S/H capacitor, which can vary substantially.

Figure 30-11. Analog Input Schematic



30.3.2.5 ADC Conversion Result

After the conversion is complete (RESRDY is '1'), the conversion result RES is available in the ADC Result (ADCn.RES) register. The result of a 10-bit conversion is given as follows:

$$\mathrm{RES} = \frac{1023 \times V_{\mathrm{IN}}}{V_{\mathrm{REF}}}$$

where V_{IN} is the voltage on the selected input pin and V_{REF} the selected voltage reference (see description for REFSEL in ADCn.CTRLC and ADCn.MUXPOS).

30.3.2.6 Temperature Measurement

The temperature measurement is based on an on-chip temperature sensor. For temperature measurement, follow these steps:

1. Configure the internal voltage reference to 1.1V by configuring the VREF peripheral.
2. Select the internal voltage reference by writing the REFSEL bits in ADCn.CTRLC to 0x0.
3. Select the ADC temperature sensor channel by configuring the MUXPOS (ADCn.MUXPOS) register. This enables the temperature sensor.
4. In ADCn.CTRLD select $\mathrm{INITDLY} \geq 32\ \mu s \times f_{\mathrm{CLK\_ADC}}$.
5. In ADCn.SAMPCTRL select $\mathrm{SAMPLEN} \geq 32\ \mu s \times f_{\mathrm{CLK\_ADC}}$.
6. In ADCn.CTRLC select $\mathrm{SAMPCAP} = 1$.
7. Acquire the temperature sensor output voltage by starting a conversion.
8. 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 are determined during the production test and saved in the Signature Row:

- SIGROW.TEMPSENSE0 is a gain/slope correction
- SIGROW.TEMPSENSE1 is an offset correction

To achieve accurate results, the result of the temperature sensor measurement must be processed in the application software using factory calibration values. The temperature (in Kelvin) is calculated by this rule:

```
Temp = (((RESH << 8) | RESL) - TEMPSENSE1) * TEMPSENSE0) >> 8
```

RESH and RESL are the high and low bytes of the Result register (ADCn.RES), and TEMPSENSEn are the respective values from the Signature row.

It is recommended to follow these steps in user code:

```
int8_t sigrow_offset = SIGROW.TEMPSENSE1;  // Read signed value from signature row
uint8_t sigrow_gain = SIGROW.TEMPSENSE0;    // Read unsigned value from signature row
uint16_t adc_reading = 0;   // ADC conversion result with 1.1 V internal reference

uint32_t temp = adc_reading - sigrow_offset;
temp *= sigrow_gain;  // Result might overflow 16 bit variable (10bit+8bit)
temp += 0x80;               // Add 1/2 to get correct rounding on division below
temp >>= 8;                 // Divide result to get Kelvin
uint16_t temperature_in_K = temp;
```

30.3.2.7 Window Comparator Mode

The ADC can raise the WCMP flag in the Interrupt and Flag (ADCn.INTFLAG) register and request an interrupt (WCMP) when the result of a conversion is above and/or below certain thresholds. The available modes are:

- The result is under a threshold
- The result is over a threshold
- The result is inside a window (above a lower threshold, but below the upper one)
- The result is outside a window (either under the lower or above the upper threshold)

The thresholds are defined by writing to the Window Comparator Threshold registers (ADCn.WINLT and ADCn.WINHT). Writing to the Window Comparator mode (WINCM) bit field in the Control E (ADCn.CTRLE) register selects the conditions when the flag is raised and/or the interrupt is requested.

Assuming the ADC is already configured to run, follow these steps to use the Window Comparator mode:

1. Choose which Window Comparator to use (see the WINCM description in ADCn.CTRLE), and set the required threshold(s) by writing to ADCn.WINLT and/or ADCn.WINHT.
2. Optional: enable the interrupt request by writing a '1' to the Window Comparator Interrupt Enable (WCMP) bit in the Interrupt Control (ADCn.INTCTRL) register.
3. Enable the Window Comparator and select a mode by writing a non-zero value to the WINCM bit field in ADCn.CTRLE.

When accumulating multiple samples, the comparison between the result and the threshold will happen after the last sample was acquired. Consequently, the flag is raised only once, after taking the last sample of the accumulation.

30.3.2.8 PTC Operation

When the Peripheral Touch Controller (PTC) is enabled, it takes complete control of ADC0.

When the PTC is disabled, ADC0 is available as a normal ADC.

Refer to the *Peripheral Touch Controller (PTC)* section for further information.

30.3.3 Events

An ADC conversion can be triggered automatically by an event input if the Start Event Input (STARTEI) bit in the Event Control (ADCn.EVCTRL) register is written to '1'.

When a new result can be read from the Result (ADCn.RES) register, the ADC will generate a result ready event. The event is a pulse with a length of one clock period and handled by the Event System (EVSYS). The ADC result ready event is always generated when the ADC is enabled.

See also the description of the Asynchronous User Channel n Input Selection in the Event System (EVSYS.ASYNCUSERn).

30.3.4 Interrupts

Table 30-1. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
RESRDY	Result Ready interrupt	The conversion result is available in the Result register (ADCn.RES)
WCMP	Window Comparator interrupt	As defined by WINCM in ADCn.CTRLE

When an interrupt condition occurs, the corresponding interrupt flag is set in the peripheral's Interrupt Flags (*peripheral*.INTFLAGS) register.

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control (*peripheral*.INTCTRL) register.

An interrupt request is generated when the corresponding interrupt source is enabled, and the interrupt flag is set. The interrupt request remains active until the interrupt flag is cleared. See the peripheral's INTFLAGS register for details on how to clear interrupt flags.

30.3.5 Sleep Mode Operation

The ADC is by default disabled in Standby sleep mode.

The ADC can stay fully operational in Standby sleep mode if the Run in Standby (RUNSTDBY) bit in the Control A (ADCn.CTRLA) register is written to '1'.

When the device is entering Standby sleep mode when RUNSTDBY is '1', the ADC will stay active, hence any ongoing conversions will be completed, and interrupts will be executed as configured.

In Standby sleep mode, an ADC conversion must be triggered via the Event System (EVSYS), or the ADC must be in Free-Running mode with the first conversion triggered by software before entering a sleep mode. The peripheral clock is requested if needed and is turned off after the conversion is completed.

When an input event trigger occurs, the positive edge will be detected, the Start Conversion (STCONV) bit in the Command (ADCn.COMMAND) register is set, and the conversion will start. When the conversion is completed, the Result Ready (RESRDY) flag in the Interrupt Flags (ADCn.INTFLAGS) register is set, and the STCONV bit in ADCn.COMMAND is cleared.

The reference source and supply infrastructure need time to stabilize when activated in Standby sleep mode. Configure a delay for the start of the first conversion by writing a non-zero value to the Initial Delay (INITDLY) bits in the Control D (ADCn.CTRLD) register.

In Power-Down sleep mode, no conversions are possible. Any ongoing conversions are halted and will be resumed when going out of a sleep mode. At the end of conversion, the Result Ready (RESRDY) flag will be set, but the content of the result (ADCn.RES) registers is invalid since the ADC was halted in the middle of a conversion.

30.4 Register Summary - ADCn

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTBY					RESSEL	FREERUN	ENABLE
0x01	CTRLB	7:0						SAMPNUM[2:0]		
0x02	CTRLC	7:0		SAMPCAP	REFSEL[1:0]			PRESC[2:0]		
0x03	CTRLD	7:0	INITDLY[2:0]			ASDV	SAMPDLY[3:0]			
0x04	CTRLE	7:0						WINCM[2:0]		
0x05	SAMPCTRL	7:0				SAMPLEN[4:0]				
0x06	MUXPOS	7:0				MUXPOS[4:0]				
0x07	Reserved									
0x08	COMMAND	7:0								STCONV
0x09	EVCTRL	7:0								STARTEI
0x0A	INTCTRL	7:0							WCMP	RESRDY
0x0B	INTFLAGS	7:0							WCMP	RESRDY
0x0C	DBGCTRL	7:0								DBGRUN
0x0D	TEMP	7:0	TEMP[7:0]							
0x0E ... 0x0F	Reserved									
0x10	RES	7:0	RES[7:0]							
		15:8	RES[15:8]							
0x12	WINLT	7:0	WINLT[7:0]							
		15:8	WINLT[15:8]							
0x14	WINHT	7:0	WINHT[7:0]							
		15:8	WINHT[15:8]							
0x16	CALIB	7:0								DUTYCYC

30.5 Register Description

30.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RUNSTBY					RESSEL	FREERUN	ENABLE
Access	R/W					R/W	R/W	R/W
Reset	0					0	0	0

Bit 7 – RUNSTBY Run in Standby
This bit determines whether the ADC needs to run when the chip is in Standby sleep mode.

Bit 2 – RESSEL Resolution Selection
This bit selects the ADC resolution.

Value	Description
0	Full 10-bit resolution. The 10-bit ADC results are accumulated or stored in the ADC Result (ADC.RES) register.
1	8-bit resolution. The conversion results are truncated to eight bits (MSbs) before they are accumulated or stored in the ADC Result (ADC.RES) register. The two Least Significant bits (LSbs) are discarded.

Bit 1 – FREERUN Free-Running
Writing a '1' to this bit will enable the Free-Running mode for the data acquisition. The first conversion is started by writing the STCONV bit in ADC.COMMAND high. In the Free-Running mode, a new conversion cycle is started immediately after or as soon as the previous conversion cycle has completed. This is signaled by the RESRDY flag in ADCn.INTFLAGS.

Bit 0 – ENABLE ADC Enable

Value	Description
0	ADC is disabled
1	ADC is enabled

30.5.2 Control B

Name: CTRLB
Offset: 0x01
Reset: 0x00
Property: -

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

Bits 2:0 – SAMPNUM[2:0] Sample Accumulation Number Select
These bits select how many consecutive ADC sampling results are accumulated automatically. When this bit is written to a value greater than 0x0, the according number of consecutive ADC sampling results are accumulated into the ADC Result (ADC.RES) register in one complete conversion.

Value	Name	Description
0x0	NONE	No accumulation
0x1	ACC2	2 results accumulated
0x2	ACC4	4 results accumulated
0x3	ACC8	8 results accumulated
0x4	ACC16	16 results accumulated
0x5	ACC32	32 results accumulated
0x6	ACC64	64 results accumulated
0x7	-	Reserved

30.5.3 Control C

Name: CTRLC
Offset: 0x02
Reset: 0x00
Property: -

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

Bit 6 – SAMPCAP Sample Capacitance Selection
This bit selects the sample capacitance, and hence, the input impedance. The best value is dependent on the reference voltage and the application's electrical properties.

Value	Description
0	Recommended for reference voltage values below 1V
1	Reduced size of sampling capacitance. Recommended for higher reference voltages.

Bits 5:4 – REFSEL[1:0] Reference Selection
These bits select the voltage reference for the ADC.

Value	Name	Description
0x0	INTERNAL	Internal reference
0x1	VDD	V_{DD}
0x2	VREFA	External reference V_{REFA}
Other	-	Reserved

Bits 2:0 – PRESC[2:0] Prescaler
These bits define the division factor from the peripheral clock (CLK_PER) to the ADC clock (CLK_ADC).

Value	Name	Description
0x0	DIV2	CLK_PER divided by 2
0x1	DIV4	CLK_PER divided by 4
0x2	DIV8	CLK_PER divided by 8
0x3	DIV16	CLK_PER divided by 16
0x4	DIV32	CLK_PER divided by 32
0x5	DIV64	CLK_PER divided by 64
0x6	DIV128	CLK_PER divided by 128
0x7	DIV256	CLK_PER divided by 256

30.5.4 Control D

Name: CTRLD
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	INITDLY[2:0]			ASDV	SAMPDLY[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 7:5 – INITDLY[2:0] Initialization Delay
These bits define the initialization/start-up delay before the first sample when enabling the ADC or changing to an internal reference voltage. Setting this delay will ensure that the reference, MUXes, etc. are ready before starting the first conversion. The initialization delay will also take place when waking up from deep sleep to do a measurement. The delay is expressed as a number of CLK_ADC cycles.

Value	Name	Description
0x0	DLY0	Delay 0 CLK_ADC cycles
0x1	DLY16	Delay 16 CLK_ADC cycles
0x2	DLY32	Delay 32 CLK_ADC cycles
0x3	DLY64	Delay 64 CLK_ADC cycles
0x4	DLY128	Delay 128 CLK_ADC cycles
0x5	DLY256	Delay 256 CLK_ADC cycles
Other	-	Reserved

Bit 4 – ASDV Automatic Sampling Delay Variation
Writing this bit to '1' enables automatic sampling delay variation between ADC conversions. The purpose of varying sampling instant is to randomize the sampling instant and thus avoid standing frequency components in the frequency spectrum. The value of the SAMPDLY bits is automatically incremented by one after each sample. When the Automatic Sampling Delay Variation is enabled, and the SAMPDLY value reaches 0xF, it wraps around to 0x0.

Value	Name	Description
0	ASVOFF	The Automatic Sampling Delay Variation is disabled
1	ASVON	The Automatic Sampling Delay Variation is enabled

Bits 3:0 – SAMPDLY[3:0] Sampling Delay Selection
These bits define the delay between consecutive ADC samples. The programmable sampling delay allows modifying the sampling frequency during hardware accumulation to suppress periodic noise sources that may otherwise disturb the sampling. The SAMPDLY field can also be modified automatically from one sampling cycle to another, by setting the ASDV bit. The delay is expressed as CLK_ADC cycles and is given directly by the bit field setting. The sampling cap is kept open during the delay.

30.5.5 Control E

Name: CTRLE
Offset: 0x4
Reset: 0x00
Property: -

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

Bits 2:0 – WINCM[2:0] Window Comparator Mode
This bit field enables and defines when the interrupt flag is set in Window Comparator mode. RESULT is the 16-bit accumulator result. WINLT and WINHT are 16-bit lower threshold value and 16-bit higher threshold value, respectively.

Value	Name	Description
0x0	NONE	No Window Comparison (default)
0x1	BELOW	*RESULT < WINLT*
0x2	ABOVE	*RESULT > WINHT*
0x3	INSIDE	*WINLT < RESULT < WINHT*
0x4	OUTSIDE	*RESULT < WINLT or RESULT > WINHT*
Other	-	Reserved

30.5.6 Sample Control

Name: SAMPCTRL
Offset: 0x5
Reset: 0x00
Property: -

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

Bits 4:0 – SAMPLEN[4:0] Sample Length
These bits extend the ADC sampling length in several CLK_ADC cycles. By default, the sampling time is two CLK_ADC cycles. Increasing the sampling length allows sampling sources with higher impedance. The total conversion time increases with the selected sampling length.

30.5.7 MUXPOS

Name: MUXPOS
Offset: 0x06
Reset: 0x00
Property: -

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

Bits 4:0 – MUXPOS[4:0] MUXPOS
This bit field selects which single-ended analog input is connected to the ADC. If these bits are changed during a conversion, the change will not take effect until this conversion is complete.

Value	Name	Description
0x00	AIN0	ADC input pin 0
0x01	AIN1	ADC input pin 1
0x02	AIN2	ADC input pin 2
0x03	AIN3	ADC input pin 3
0x04	AIN4	ADC input pin 4
0x05	AIN5	ADC input pin 5
0x06	AIN6	ADC input pin 6
0x07	AIN7	ADC input pin 7
0x08	AIN8	ADC input pin 8
0x09	AIN9	ADC input pin 9
0x0A	AIN10	ADC input pin 10
0x0B	AIN11	ADC input pin 11
0x1B	PTC	ADC0: Reserved ADC1: DAC2
0x1C	DAC0	DAC0
0x1D	INTREF	Internal reference (from VREF peripheral)
0x1E	TEMPSENSE	ADC0: Temperature sensor ADC1: DAC1
0x1F	GND	0V (GND)
Other	-	Reserved

30.5.8 Command

Name: COMMAND
Offset: 0x08
Reset: 0x00
Property: -

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

Bit 0 – STCONV Start Conversion
Writing a '1' to this bit will start a single measurement. If in Free-Running mode, this will start the first conversion. STCONV will read as '1' as long as a conversion is in progress. When the conversion is complete, this bit is automatically cleared.

30.5.9 Event Control

Name: EVCTRL
Offset: 0x09
Reset: 0x00
Property: -

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

Bit 0 – STARTEI Start Event Input
This bit enables using the event input as a trigger for starting a conversion.

30.5.10 Interrupt Control

Name: INTCTRL
Offset: 0x0A
Reset: 0x00
Property: -

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

Bit 1 – WCMP Window Comparator Interrupt Enable
Writing a '1' to this bit enables the window comparator interrupt.

Bit 0 – RESRDY Result Ready Interrupt Enable
Writing a '1' to this bit enables the result ready interrupt.

30.5.11 Interrupt Flags

Name: INTFLAGS
Offset: 0x0B
Reset: 0x00
Property: -

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

Bit 1 – WCMP Window Comparator Interrupt Flag
This window comparator interrupt flag is set when the measurement is complete and if the result matches the selected Window Comparator mode defined by WINCM (ADCn.CTRLE). The comparison is done at the end of the conversion. The flag is cleared by either writing a '1' to the bit position or by reading the Result (ADCn.RES) register. Writing a '0' to this bit has no effect.

Bit 0 – RESRDY Result Ready Interrupt Flag
The Result Ready interrupt flag is set when a measurement is complete, and a new result is ready. The flag is cleared by either writing a '1' to the bit location or by reading the Result (ADCn.RES) register. Writing a '0' to this bit has no effect.

30.5.12 Debug Run

Name: DBGCTRL
Offset: 0x0C
Reset: 0x00
Property: -

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

Bit 0 – DBGRUN Debug Run

Value	Description
0	The peripheral is halted in Break Debug mode and ignores events
1	The peripheral will continue to run in Break Debug mode when the CPU is halted

30.5.13 Temporary

Name: TEMP
Offset: 0x0D
Reset: 0x00
Property: -

The Temporary register is used by the CPU for 16-bit single-cycle access to the 16-bit registers of this peripheral. The register is common for all the 16-bit registers of this peripheral and can be read and written by software. For more details on reading and writing 16-bit registers, refer to *Accessing 16-Bit Registers* in the *AVR CPU* section.

Bit	7	6	5	4	3	2	1	0
	TEMP[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 – TEMP[7:0] Temporary
Temporary register for read/write operations in 16-bit registers.

30.5.14 Result

Name: RES
Offset: 0x10
Reset: 0x00
Property: -

The ADCn.RESL and ADCn.RESH register pair represents the 16-bit value, ADCn.RES. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

If the analog input is higher than the reference level of the ADC, the 10-bit ADC result will be equal the maximum value of 0x3FF. Likewise, if the input is below 0V, the ADC result will be 0x000. As the ADC cannot produce a result above 0x3FF values, the accumulated value will never exceed 0xFFC0 even after the maximum allowed 64 accumulations.

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

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

Bits 15:8 – RES[15:8] Result high byte
These bits constitute the MSB of the ADCn.RES register, where the MSb is RES[15]. The ADC itself has a 10-bit output, ADC[9:0], where the MSb is ADC[9]. The data format in ADC and Digital Accumulation is 1's complement, where 0x0000 represents the zero, and 0xFFFF represents the largest number (full scale).

Bits 7:0 – RES[7:0] Result low byte
These bits constitute the LSB of ADC/Accumulator Result, (ADCn.RES) register. The data format in ADC and Digital Accumulation is 1's complement, where 0x0000 represents the zero, and 0xFFFF represents the largest number (full scale).

30.5.15 Window Comparator Low Threshold

Name: WINLT
Offset: 0x12
Reset: 0x00
Property: -

This register is the 16-bit low threshold for the digital comparator monitoring the ADCn.RES register. The ADC itself has a 10-bit output, RES[9:0], where the MSb is RES[9]. The data format in ADC and Digital Accumulation is 1's complement, where 0x0000 represents the zero, and 0xFFFF represents the largest number (full scale).

The ADCn.WINLTH and ADCn.WINLTL register pair represents the 16-bit value, ADCn.WINLT. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

When accumulating samples, the window comparator thresholds are applied to the accumulated value and not on each sample.

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 15:8 – WINLT[15:8] Window Comparator Low Threshold High Byte
These bits hold the MSB of the 16-bit register.

Bits 7:0 – WINLT[7:0] Window Comparator Low Threshold Low Byte
These bits hold the LSB of the 16-bit register.

30.5.16 Window Comparator High Threshold

Name: WINHT
Offset: 0x14
Reset: 0x00
Property: -

This register is the 16-bit high threshold for the digital comparator monitoring the ADCn.RES register. The ADC itself has a 10-bit output, RES[9:0], where the MSb is RES[9]. The data format in ADC and Digital Accumulation is 1's complement, where 0x0000 represents the zero, and 0xFFFF represents the largest number (full scale).

The ADCn.WINHTH and ADCn.WINHTL register pair represents the 16-bit value, ADCn.WINHT. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + `0x01`.

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 15:8 – WINHT[15:8] Window Comparator High Threshold High Byte
These bits hold the MSB of the 16-bit register.

Bits 7:0 – WINHT[7:0] Window Comparator High Threshold Low Byte
These bits hold the LSB of the 16-bit register.

30.5.17 Calibration

Name: CALIB
Offset: 0x16
Reset: 0x01
Property: -

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

Bit 0 – DUTYCYC Duty Cycle
This bit determines the duty cycle of the ADC clock.
ADC_{clk} > 1.5 MHz requires a minimum operating voltage of 2.7V.

Value	Description
0	50% Duty Cycle must be used if ADC_{clk} > 1.5 MHz
1	25% Duty Cycle (high 25% and low 75%) must be used for ADC_{clk} ≤ 1.5 MHz

31. DAC - Digital-to-Analog Converter

31.1 Features

- 8-bit Resolution
- Up to 350 ksps Conversion Rate
- High Drive Capabilities (DAC0)
- Functioning as Input to Analog Comparator (AC) or Analog-to-Digital Converter (ADC)

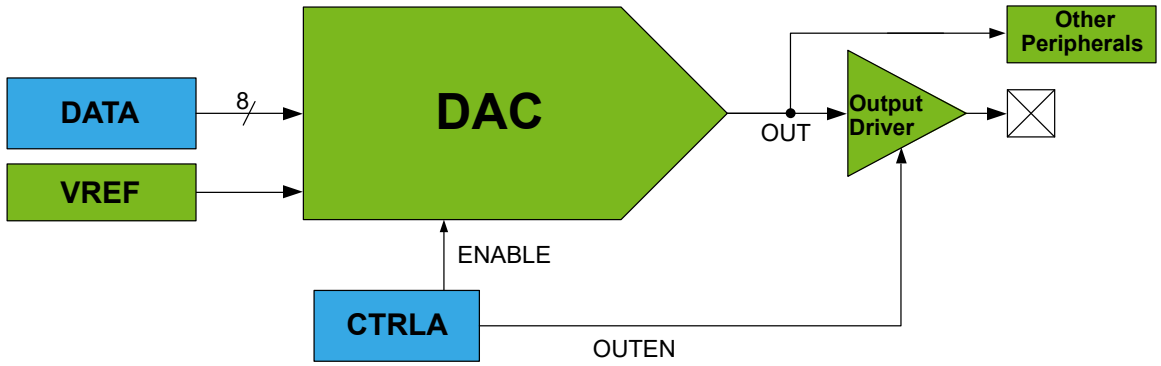
31.2 Overview

The Digital-to-Analog Converter (DAC) converts a digital value written to the Data (DAC.DATA) register to an analog voltage. The conversion range is between GND and the selected reference voltage.

The DAC features an 8-bit resistor-string type DAC, capable of converting 350,000 samples per second (350 ksps). The DAC uses the internal Voltage Reference (VREF) as the upper limit for conversion and has one continuous time output with high drive capabilities, which can drive a 5 kΩ and/or 30 pF load. The DAC conversion can be started from the application by writing to the Data Conversion registers.

31.2.1 Block Diagram

Figure 31-1. DAC Block Diagram



Note: Only DAC0 has an output driver for an external pin.

31.2.2 Signal Description

Signal	Description	Type
OUT	DAC output	Analog

31.2.3 System Dependencies

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

Table 31-1. DAC System Dependencies

Dependency	Applicable	Peripheral
Clocks	Yes	CLKCTRL
I/O Lines and Connections	Yes	PORT
Interrupts	No	-
Events	No	-

.....continued		
Dependency	**Applicable**	**Peripheral**
Debug	Yes	UPDI

31.2.3.1 Clocks

This peripheral depends on the peripheral clock.

31.2.3.2 I/O Lines and Connections

Using the I/O lines of the peripheral requires configuration of the I/O pins.

Table 31-2. I/O Lines

Instance	Signal	I/O Line	Peripheral Function
DAC0	OUT	PA6	A

The DAC0 has one analog output pin (OUT) that must be configured before it can be used.

A DAC is also internally connected to the AC and the ADC. To use this internal OUT as input, both output and input must be configured in their respective registers.

31.2.3.3 Events

Not applicable.

31.2.3.4 Interrupts

Not applicable.

31.2.3.5 Debug Operation

This peripheral is unaffected by entering Debug mode.

If the peripheral is configured to require periodic service by the CPU through interrupts or similar, improper operation or data loss may result during halted debugging.

31.3 Functional Description

31.3.1 Initialization

To operate the DAC, the following steps are required:

1. Select the DAC reference voltage in the Voltage Reference (VREF) peripheral by writing the DAC and AC Reference Selection (DACnREFSEL) bits in the Control x (VREF.CTRLx) register.
2. The conversion range is between GND and the selected reference voltage.
3. Configure the further usage of the DAC output:
   - 3.1. Configure an internal peripheral (e.g., AC, ADC) to use the DAC output. Refer to the according peripheral's documentation.
   - 3.2. Enable the output to a pin by writing a '1' to the Output Enable (OUTEN) bit in the Control A (DAC.CTRLA) register. This requires a configuration of the Port peripheral.

   For DAC0, either one or both options are valid. Other instances of the DAC only support internal signaling.
4. Write an initial digital value to the Data (DAC.DATA) register.
5. Enable the DAC by writing a '1' to the ENABLE bit in the DAC.CTRLA register.

31.3.2 Operation

31.3.2.1 Enabling, Disabling and Resetting

The DAC is enabled by writing a '1' to the ENABLE bit in the Control A (DACn.CTRLA) register and disabled by writing a '0' to this bit.

The OUT output to a pin is enabled by writing the Output Enable (OUTEN) bit in the DACn.CTRLA register.

31.3.2.2 Starting a Conversion

When the DAC is enabled (ENABLE = '1' in DACn.CTRLA), a conversion starts as soon as the Data (DACn.DATA) register is written.

When the DAC is disabled (ENABLE = '0' in DACn.CTRLA), writing to the DACn.DATA register does not trigger a conversion. Instead, the conversion starts on writing a '1' to the ENABLE bit in the DACn.CTRLA register.

31.3.2.3 DAC as Source For Internal Peripherals

The analog output of the DAC is internally connected to both the AC and the ADC and is available to these peripherals when the DAC is enabled (ENABLE = '1' in DAC.CTRLA). When the DAC analog output is only being used internally, it is not necessary to enable the pin output driver (i.e., OUTEN = '0' in DAC.CTRLA is acceptable).

31.3.3 Sleep Mode Operation

If the Run in Standby (RUNSTDBY) bit in the Control A (DAC.CTRLA) register is written to '1' and CLK_PER is available, the DAC will continue to operate in Standby sleep mode. If the RUNSTDBY bit is '0', the DAC will stop the conversion in Standby sleep mode.

If the conversion is stopped in Standby sleep mode, the DAC and the output buffer are disabled to reduce power consumption. When the device is exiting Standby sleep mode, the DAC and the output buffer (if configured by OUTEN = '1' in DAC.CTRLA) are enabled again. Therefore, certain start-up time is required before a new conversion is initiated.

In Power-Down sleep mode, the DAC and output buffer are disabled to reduce the power consumption.

31.3.4 Configuration Change Protection

Not applicable.

31.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY	OUTEN						ENABLE
0x01	DATA	7:0	DATA[7:0]							

31.5 Register Description

31.5.1 Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY	OUTEN						ENABLE
Access	R/W	R/W						R/W
Reset	0	0						0

Bit 7 – RUNSTDBY Run in Standby Mode
If this bit is written to '1', the DAC or output buffer will not automatically be disabled when the device is entering Standby sleep mode.

Bit 6 – OUTEN Output Buffer Enable
Writing a '1' to this bit enables the output buffer and sends the OUT signal to a pin.

Bit 0 – ENABLE DAC Enable
Writing a '1' to this bit enables the DAC.

31.5.2 DATA

Name: DATA
Offset: 0x01
Reset: 0x00
Property: -

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
This bit field contains the digital data, which will be converted to an analog voltage.

32. PTC - Peripheral Touch Controller

32.1 Overview

The Peripheral Touch Controller (PTC) acquires signals to detect a touch on the capacitive sensors. The external capacitive touch sensor is typically designed as part of the printed circuit board (PCB) layout, and the sensor electrodes are connected to the analog front end of the PTC through the I/O pins of the device. The PTC supports both self and mutual capacitance sensors.

In the Mutual Capacitance mode, sensing is done using capacitive touch matrices in various X-Y configurations, including indium tin oxide (ITO) sensor grids. The PTC requires one pin per X-line and one pin per Y-line.

In the Self Capacitance mode, the PTC requires only one pin (Y-line) for each touch sensor.

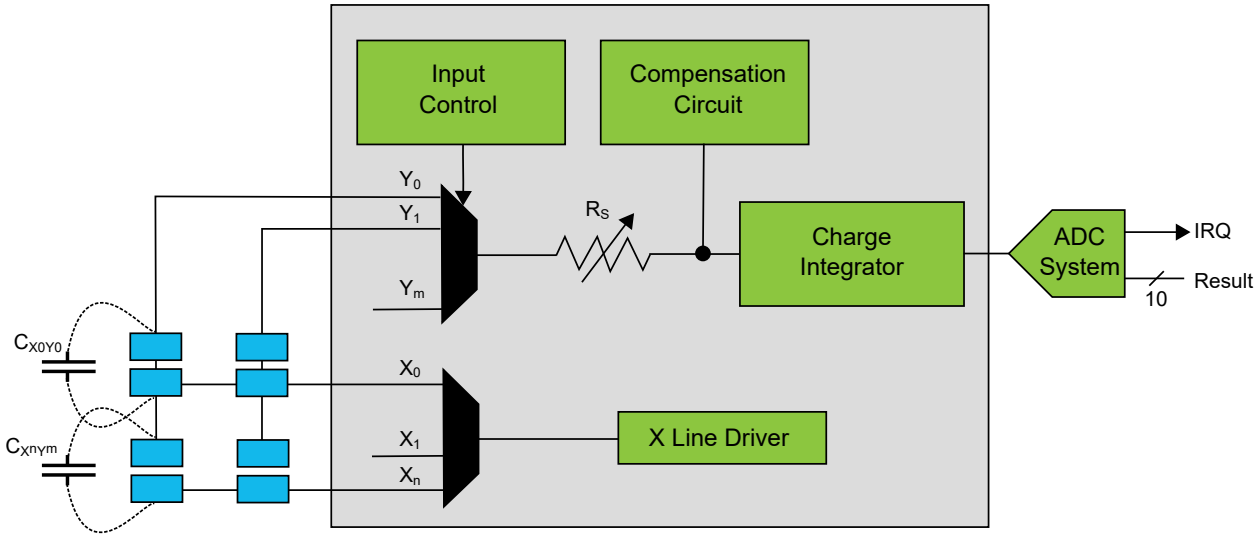
The number of available pins and the assignment of X- and Y-lines is depending on both package type and device configuration. Refer to the *Configuration Summary* and *I/O Multiplexing and Considerations* sections for further details.

32.2 Features

- Low-Power, High-Sensitivity, Environmentally Robust Capacitive Touch Buttons, Sliders, and Wheels
- Supports Wake-up on Touch from Standby Sleep Mode
- Supports Mutual Capacitance and Self Capacitance Sensing
  - Mix-and-match mutual and self-capacitance sensors
- One Pin per Electrode – No External Components
- Load Compensating Charge Sensing
  - Parasitic capacitance compensation and adjustable gain for superior sensitivity
- Zero Drift Over the Temperature and V_{DD} Range
  - Auto-calibration and recalibration of sensors
- Single-Shot and Free-Running Charge Measurement
- Hardware Noise Filtering and Noise Signal Desynchronization for High Conducted Immunity
- Driven Shield for Better Noise Immunity and Moisture Tolerance
  - Any PTC X/Y line can be used for the driven shield
  - All enabled sensors will be driven at the same potential as the sensor scanned
- Selectable Channel Change Delay Allows Choosing the Settling Time on a New Channel, as Required
- Acquisition-Start Triggered by Command or Through Auto-Triggering Feature
- Low CPU Utilization Through Interrupt on Acquisition-Complete
- Using ADC Peripheral for Signal Conversion and Acquisition

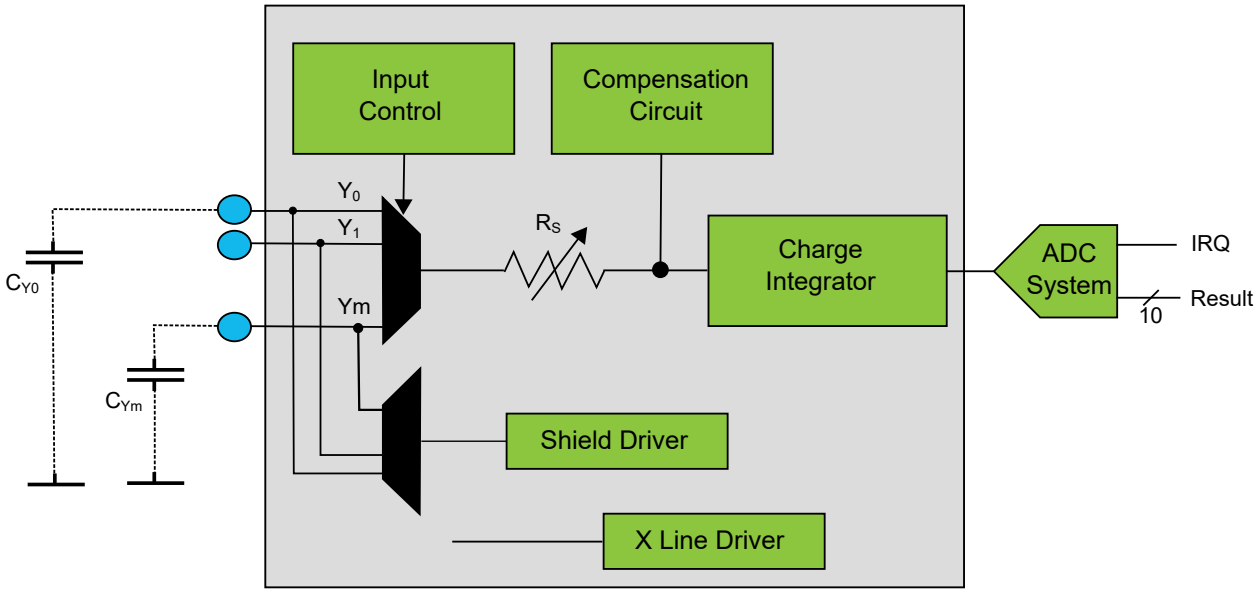
32.3 Block Diagram

Figure 32-1. PTC Block Diagram Mutual Capacitance



Note: For ATtiny3216/3217 the R_S = 0, 20, 50, 70, 100, 200 kΩ.

Figure 32-2. PTC Block Diagram Self Capacitance



32.4 Signal Description

Table 32-1. Signal Description for PTC

Name	Type	Description
Y[m:0]	Analog	Y-line (Input/Output)
X[n:0]	Digital	X-line (Output)

Note: The number of X- and Y-lines are device-dependent. Refer to *Configuration Summary* for details.

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

32.5 System Dependencies

To use this peripheral, configure the other components of the system, as described in the following sections.

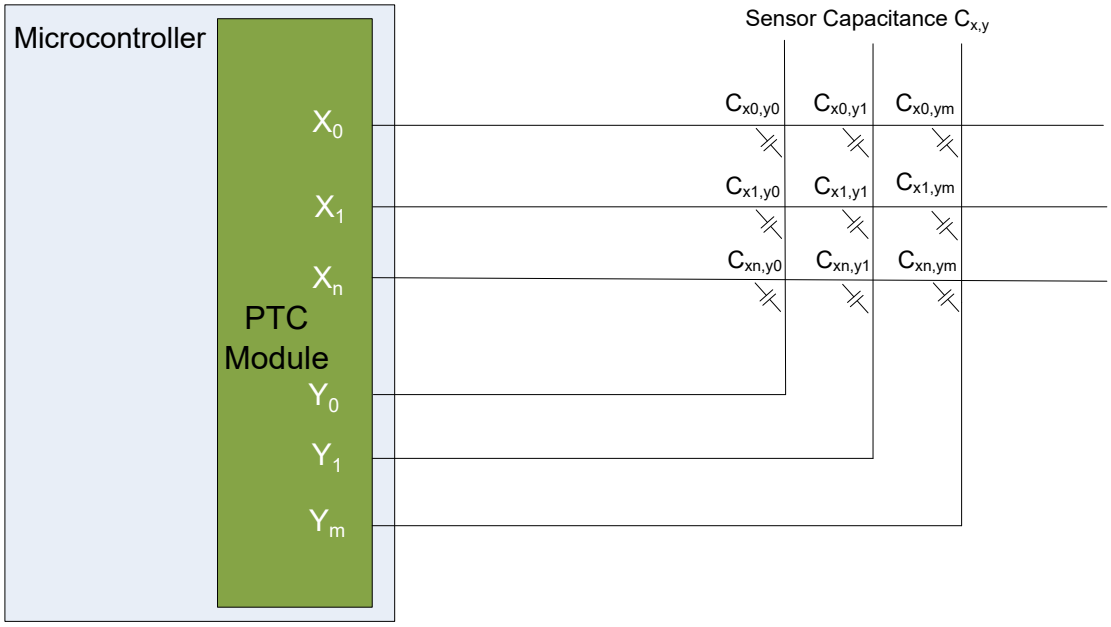
32.5.1 I/O Lines

The I/O lines used for analog X-lines and Y-lines must be connected to external capacitive touch sensor electrodes. External components are not required for normal operation. However, to improve the EMC performance, a series resistor of 1 kΩ or more can be used on X-lines and Y-lines.

32.5.1.1 Mutual Capacitance Sensor Arrangement

A mutual capacitance sensor is designed as part of the printed circuit board (PCB) layout between two I/O lines - an X electrode for transmitting and Y electrode for sensing. The mutual capacitance between the X and Y electrode is measured by the peripheral touch controller.

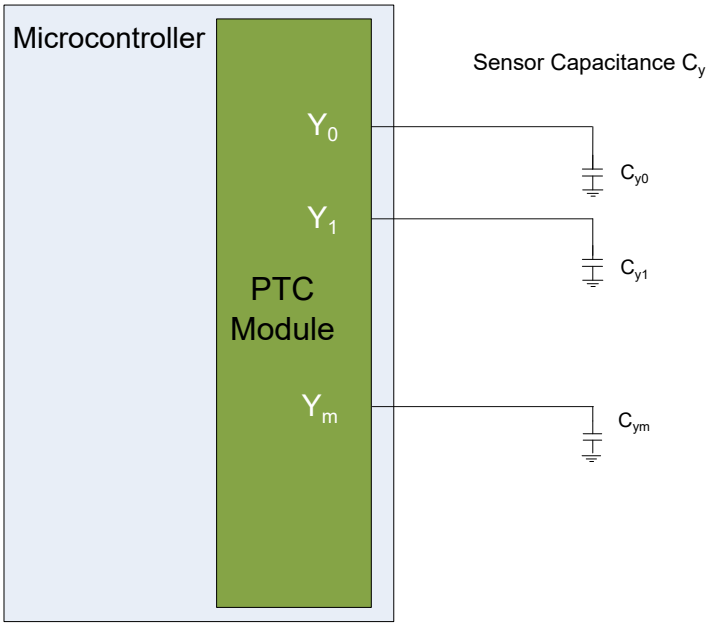
Figure 32-3. Mutual Capacitance Sensor Arrangement



32.5.1.2 Self Capacitance Sensor Arrangement

A self-capacitance sensor is connected to a single pin on the peripheral touch controller through the Y electrode for sensing the signal. The sense electrode capacitance is measured by the peripheral touch controller.

Figure 32-4. Self-Capacitance Sensor Arrangement



For more information about designing the touch sensor, refer to *Capacitive Touch Sensor Design Guide* (www.microchip.com/DS00002934).

32.5.2 Clocks

The PTC is clocked by the CLK_PER clock. Refer to the *Clock Controller (CLKCTRL)* section for details on configuring CLK_PER.

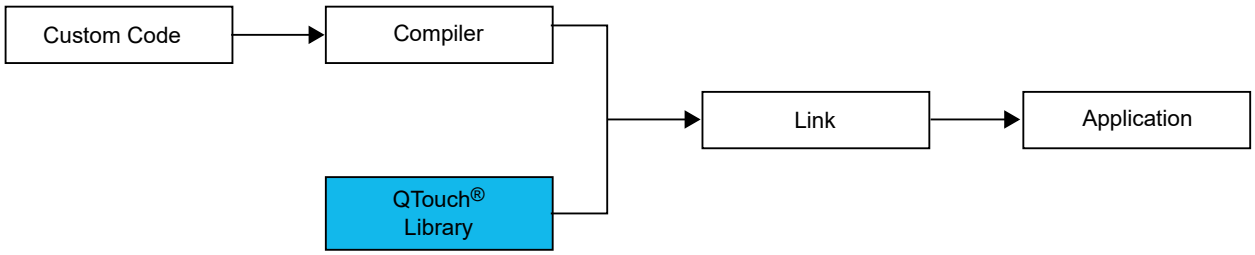
32.5.3 Analog-to-Digital Converter (ADC)

The PTC is using the ADC for signal conversion and acquisition. The ADC must be enabled and configured appropriately to allow correct behavior of the PTC. Refer to the *Analog-to-Digital Converter (ADC)* section for further details.

32.6 Functional Description

To access the PTC, the user must use the Atmel START QTouch® Configurator to configure and link the QTouch Library firmware with the application software. The QTouch Library can be used to implement buttons, sliders, and wheels in a variety of combinations on a single interface.

Figure 32-5. QTouch® Library Usage



For more information about QTouch Library, refer to the *QTouch® Modular Library Peripheral Touch Controller User's Guide* (www.microchip.com/DS40001986).

33. UPDI - Unified Program and Debug Interface

33.1 Features

- UPDI One-Wire Interface for External Programming and On-Chip-Debugging (OCD)
  - Enable programming by high-voltage or fuse
  - Uses the $\overline{\text{RESET}}$ pin of the device for programming
  - No GPIO pins occupied during the operation
  - Asynchronous half-duplex UART protocol towards the programmer
- Programming:
  - Built-in error detection and error signature generation
  - Override of response generation for faster programming
- Debugging:
  - Memory-mapped access to device address space (NVM, RAM, I/O)
  - No limitation on the device clock frequency
  - Unlimited number of user program breakpoints
  - Two hardware breakpoints
  - Support for advanced OCD features
    - Run-time readout of the CPU Program Counter (PC), Stack Pointer (SP) and Status Register (SREG) for code profiling
    - Detection and signalization of the Break/Stop condition in the CPU
    - Program flow control for Run, Stop and Reset debug instructions
  - Nonintrusive run-time chip monitoring without accessing the system registers
  - Interface for reading the result of the CRC check of the Flash on a locked device

33.2 Overview

The Unified Program and Debug Interface (UPDI) is a proprietary interface for external programming and OCD of a device.

The UPDI supports programming of Nonvolatile Memory (NVM) space, Flash, EEPROM, fuses, lock bits, and the user row. Some memory-mapped registers are accessible only with the correct access privilege enabled (key, lock bits) and only in the OCD Stopped mode or certain Programming modes. These modes are unlocked by sending the correct key to the UPDI. See the *NVMCTRL - Nonvolatile Memory Controller* section for programming via the NVM controller and executing NVM controller commands.

The UPDI is partitioned into three separate protocol layers: the UPDI Physical (PHY) Layer, the UPDI Data Link (DL) Layer and the UPDI Access (ACC) Layer. The default PHY layer handles bidirectional UART communication over the UPDI pin line towards a connected programmer/debugger and provides data recovery and clock recovery on an incoming data frame in the One-Wire Communication mode. Received instructions and corresponding data are handled by the DL layer, which sets up the communication with the ACC layer based on the decoded instruction. Access to the system bus and memory-mapped registers is granted through the ACC layer.

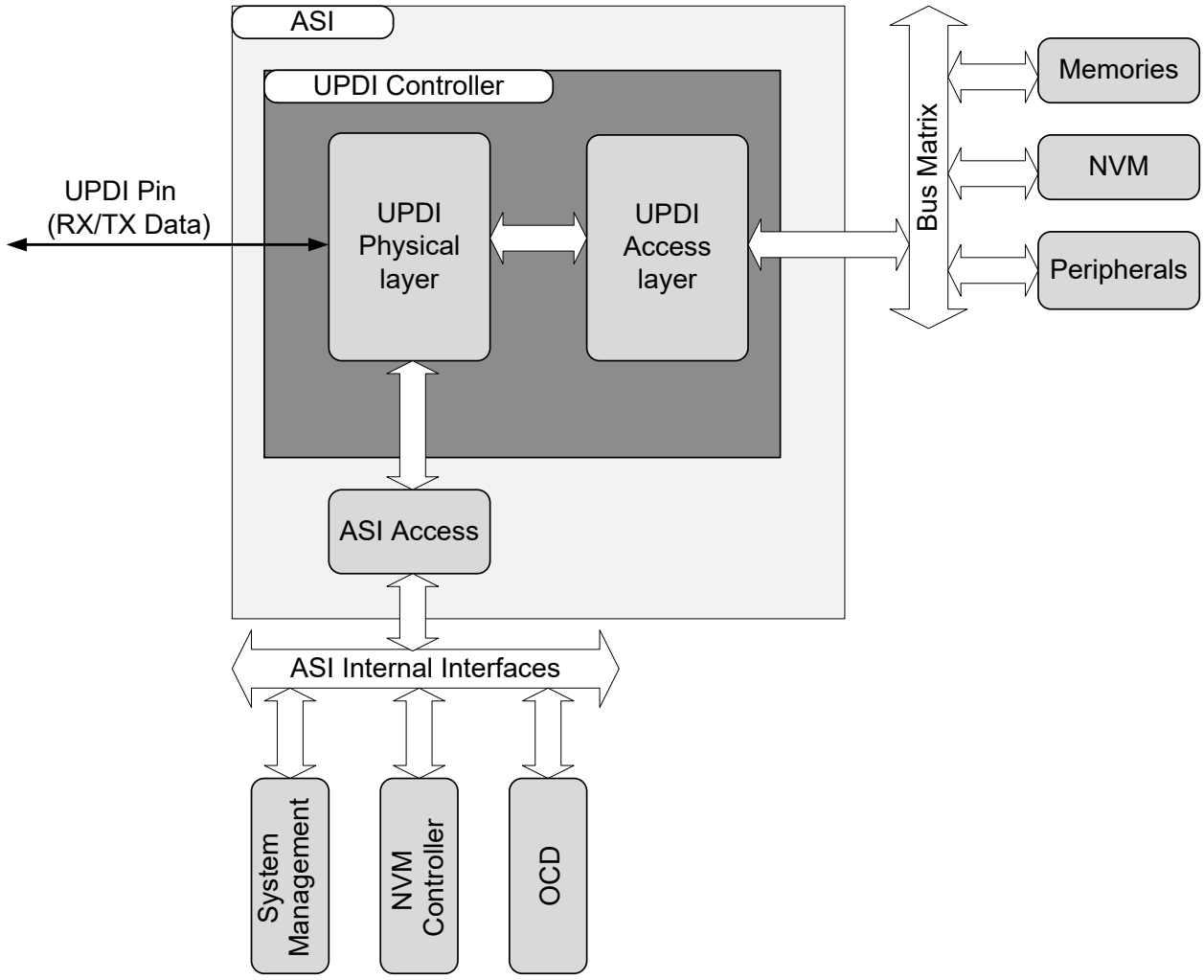
Programming and debugging are done through the PHY layer, which is a one-wire UART based on a half-duplex interface using the $\overline{\text{RESET}}$ pin for data reception and transmission. The clocking of the PHY layer is done by a dedicated internal oscillator.

The ACC layer is the interface between the UPDI and the connected bus matrix. This layer grants access via the UPDI interface to the bus matrix with memory-mapped access to system blocks such as memories, NVM, and peripherals.

The Asynchronous System Interface (ASI) provides direct interface access to select features in the OCD, NVM, and System Management systems. This gives the debugger direct access to system information without requesting bus access.

33.2.1 Block Diagram

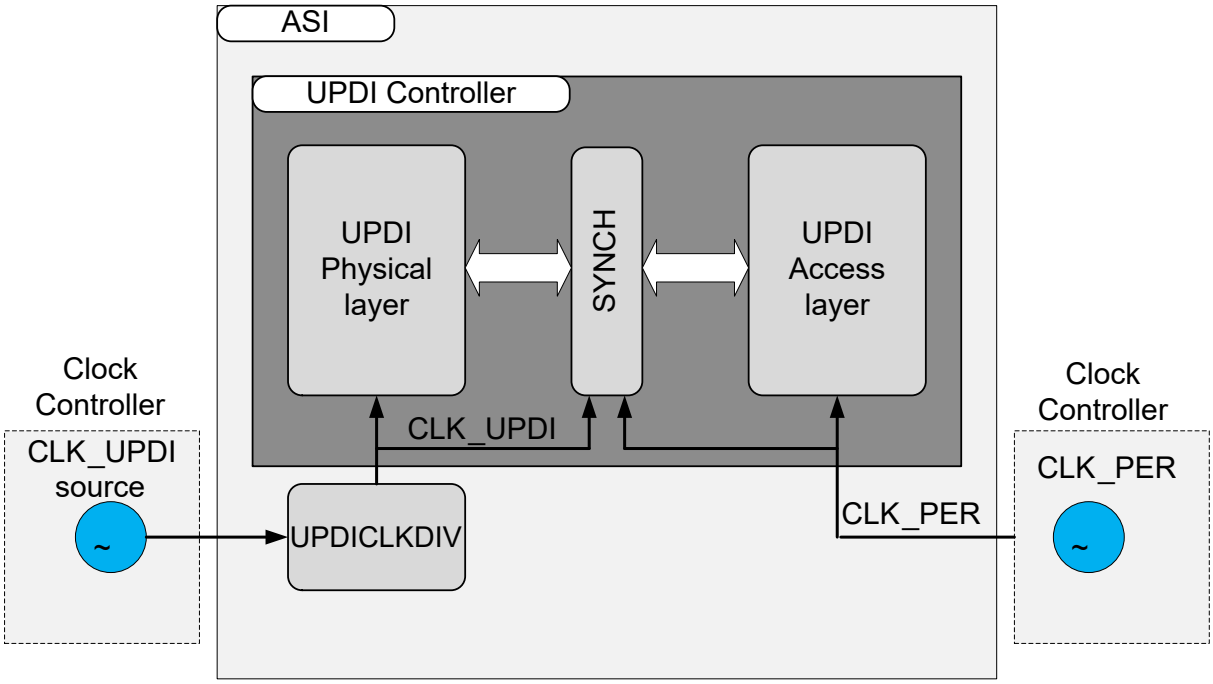
Figure 33-1. UPDI Block Diagram



33.2.2 Clocks

The PHY layer and the ACC layer can operate on different clock domains. The PHY layer clock is derived from the dedicated internal oscillator, and the ACC layer clock is the same as the peripheral clock. There is a synchronization boundary between the PHY and the ACC layer, which ensures correct operation between the clock domains. The UPDI clock output frequency is selected through the ASI, and the default UPDI clock start-up frequency is 4 MHz after enabling or resetting the UPDI. The UPDI clock frequency can be changed by writing to the UPDI Clock Divider Select (UPDICLKDIV) bit field in the ASI Control A (UPDI.ASI_CTRLA) register.

Figure 33-2. UPDI Clock Domains



33.2.3 Physical Layer

The PHY layer is the communication interface between a connected programmer/debugger and the device. The main features of the PHY layer can be summarized as follows:

- Support for UPDI One-Wire Asynchronous mode, using half-duplex UART communication on the UPDI pin
- Internal baud detection, clock and data recovery on the UART frame
- Error detection (parity, clock recovery, frame, system errors)
- Transmission response generation (ACK)
- Generation of error signatures during operation
- Guard time control

33.2.4 I/O Lines and Connections

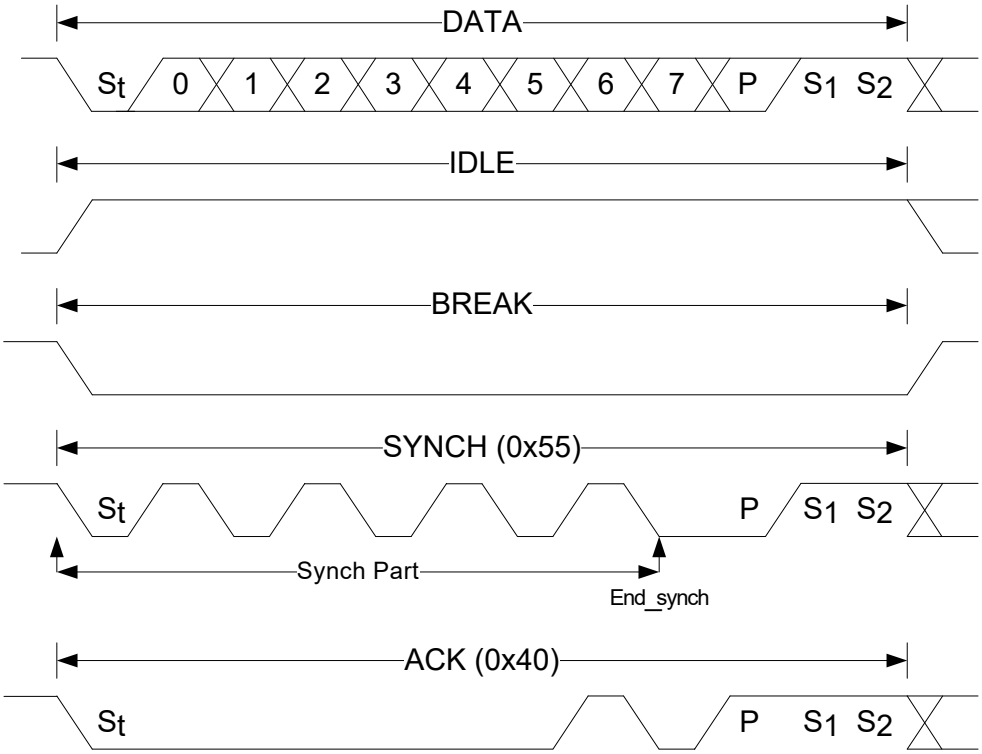
To operate the UPDI the $\overline{\text{RESET}}$ pin must be set to UPDI mode. This is not done through the port I/O pin configuration as regular I/O pins but through setting the $\overline{\text{RESET}}$ Pin Configuration (RSTPINCFG) bits in FUSE.SYSCFG0 as described in 33.3.2.1.2 UPDI Enable with Fuse Override of RESET Pin, or by following the UPDI high-voltage enable sequence from 33.3.2.1.3 UPDI Enable with High-Voltage Override of RESET Pin. Pull enable, input enable and output enable settings are automatically controlled by the UPDI when active.

33.3 Functional Description

33.3.1 Principle of Operation

The communication through the UPDI is based on standard UART communication, using a fixed frame format, and automatic baud rate detection for clock and data recovery. In addition to the data frame, several control frames are important to the communication: DATA, IDLE, BREAK, SYNCH, ACK.

Figure 33-3. Supported UPDI Frame Formats



Frame	Description
DATA	A DATA frame consists of one Start (St) bit which is always low, eight Data bits, one Parity (P) bit for even parity and two Stop (S1 and S2) bits which are always high. If the Parity bit or Stop bits have an incorrect value, an error will be detected and signalized by the UPDI. The parity bit-check in the UPDI can be disabled by writing to the Parity Disable (PARD) bit in the Control A (UPDI.CTRLA) register, in which case the parity generation from the debugger is ignored.
IDLE	This is a special frame that consists of 12 high bits. This is the same as keeping the transmission line in an Idle state.
BREAK	This is a special frame that consists of 12 low bits. It is used to reset the UPDI back to its default state and is typically used for error recovery.
SYNCH	The SYNCH frame is used by the Baud Rate Generator to set the baud rate for the coming transmission. A SYNCH character is always expected by the UPDI in front of every new instruction, and after a successful BREAK has been transmitted.
ACK	The ACK frame is transmitted from the UPDI whenever an `ST` or an `STS` instruction has successfully crossed the synchronization boundary and gained bus access. When an ACK is received by the debugger, the next transmission can start.

33.3.1.1 UPDI UART

The communication is initiated from the master debugger/programmer side, and every transmission must start with a SYNCH character, which the UPDI can use to recover the transmission baud rate and store this setting for the incoming data. The baud rate set by the SYNCH character will be used for both reception and transmission of the subsequent instruction and data bytes. See the 33.3.3 UPDI Instruction Set section for details on when the next SYNCH character is expected in the instruction stream.

There is no writable Baud Rate register in the UPDI, so the baud rate sampled from the SYNCH character is used for data recovery when sampling the data byte.

The transmission baud rate of the PHY layer is related to the selected UPDI clock, which can be adjusted by writing to the UPDI Clock Divider Select (UPDICLKDIV) bit field in the ASI Control A (UPDI.ASI_CTRLA) register. The receive and transmit baud rates are always the same within the accuracy of the auto-baud.

Table 33-1. Recommended UART Baud Rate Based on UPDICLKDIV Setting

0.150 kbps	Max. Recommended Baud Rate	Min. Recommended Baud Rate
0x1 (16 MHz)	0.9 Mbps	0.300 kbps
0x2 (8 MHz)	450 kbps	0.150 kbps
0x3 (4 MHz) - Default	225 kbps	0.075 kbps

The UPDI Baud Rate Generator utilizes fractional baud counting to minimize the transmission error. With the fixed frame format used by the UPDI, the maximum and recommended receiver transmission error limits can be seen in the following table:

Table 33-2. Receiver Baud Rate Error

Data + Parity Bits	R_{slow}	R_{fast}	Max. Total Error [%]	Recommended Max. RX Error [%]
9	96.39	104.76	+4.76/-3.61	+1.5/-1.5

33.3.1.2 BREAK Character

The BREAK character is used to reset the internal state of the UPDI to the default setting. This is useful if the UPDI enters an Error state due to a communication error or when the synchronization between the debugger and the UPDI is lost.

To ensure that a BREAK is successfully received by the UPDI in all cases, the debugger must send two consecutive BREAK characters. The first BREAK will be detected if the UPDI is in Idle state and will not be detected if it is sent while the UPDI is receiving or transmitting (at a very low baud rate). However, this will cause a frame error for the reception (RX) or a contention error for the transmission (TX), and abort the ongoing operation. The UPDI will then detect the next BREAK successfully.

Upon receiving a BREAK, the UPDI oscillator setting in the ASI Control A (UPDI.ASI_CTRLA) register is reset to the 4 MHz default UPDI clock selection. This changes the baud rate range of the UPDI, according to Table 33-1.

33.3.1.2.1 BREAK in One-Wire Mode

In Asynchronous mode, the programmer/debugger and UPDI can be totally out of synch, requiring a worst-case length for the BREAK character to be sure that the UPDI can detect it. Assuming the slowest UPDI clock speed of 4 MHz (250 ns), the maximum length of the 8-bit SYNCH pattern value that can be contained in 16 bits is:
$65535 \times 250\,ns = 16.4\,ms/byte = 16.4\,ms/8\,bits = 2.05\,ms/bit$

This gives a worst-case BREAK frame duration of 2.05 ms*12 bits ≈ 24.6 ms for the slowest prescaler setting. When the prescaler setting is known, the time of the BREAK frame can be relaxed according to the values from the next table:

Table 33-3. Recommended BREAK Character Duration

UPDICLKDIV[1:0]	Recommended BREAK Character Duration
0x0	Reserved
0x1 (16 MHz)	6.15 ms
0x2 (8 MHz)	12.30 ms
0x3 (4 MHz) - Default	24.60 ms

33.3.1.3 SYNCH Character

The SYNCH character has eight bits and follows the regular UPDI frame format. It has a fixed data bit value of '0x55'. The SYNCH character has two main purposes:

1. It acts as the enabling character for the UPDI after a disable.

2. It is used by the Baud Rate Generator to set the baud rate for the subsequent transmission. If an invalid SYNCH character is sent, the next transmission will not be sampled correctly.

33.3.1.3.1 SYNCH in One-Wire Mode

The SYNCH character is used before each new instruction. When using the `REPEAT` instruction, the SYNCH character is expected only before the first instruction after `REPEAT`.

The SYNCH is a known character which, through its property of toggling for each bit, allows the UPDI to measure how many UPDI clock cycles are needed to sample the 8-bit SYNCH pattern. The information obtained through the sampling is used to provide Asynchronous Clock Recovery and Asynchronous Data Recovery on reception, and to keep the baud rate of the connected programmer when doing transmit operations.

33.3.2 Operation

The UPDI must be enabled before the UART communication can start.

33.3.2.1 UPDI Enabling

The enable sequence for the UPDI is device independent and is described in the following paragraphs.

33.3.2.1.1 One-Wire Enable

The UPDI pin has a constant pull-up enable, and by driving the UPDI pin low for more than 200 ns, a connected programmer will initiate the start-up sequence.

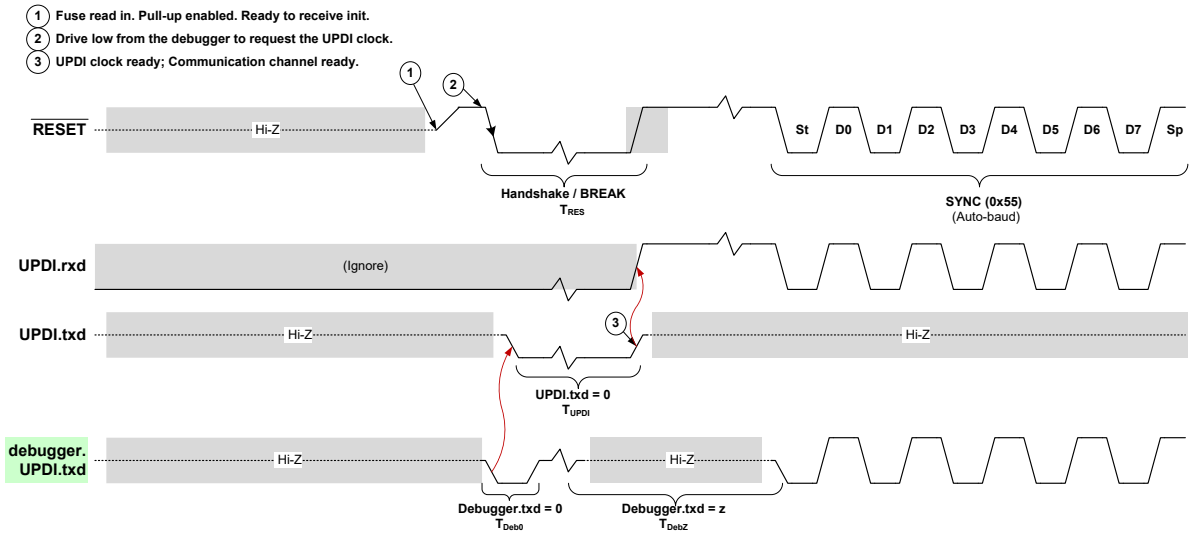
The negative edge transition will cause an edge detector (located in the high-voltage domain if it is in a Multi-Voltage System) to start driving the UPDI pin low, so when the programmer releases the line, it will stay low until the requested UPDI oscillator is ready. The expected arrival time for the clock will depend on the oscillator implementation regarding the accuracy, overshoot and readout of the oscillator calibration. For a Multi-Voltage System, the line will be driven low until the regulator is at the correct level, and the system is powered up with the selected oscillator ready and stable. The programmer must poll the UPDI pin after releasing it the first time to detect when the pin transitions to high again. This transition means that the edge detector has released the pin (pull-up), and the UPDI can receive a SYNCH character. Upon successful detection of the SYNCH character, the UPDI is enabled and will prepare for the reception of the first instruction.

The enable transmission sequence is shown in the next figure, where the active driving periods for the programmer and edge detector are included. The “UPDI pin” waveform shows the pin value at any given time.

The delay given for the edge detector active drive period is a typical start-up time waiting for 256 cycles on a 32 MHz oscillator + the calibration readout. Refer to the *Electrical Characteristics* section for details on the expected start-up times.

Note: The first instruction issued after the initial enable SYNCH does not need an extra SYNCH to be sent because the enable sequence SYNCH sets up the Baud Rate Generator for the first instruction.

Figure 33-4. UPDI Enable Sequence with UPDI PAD Enabled By Fuse

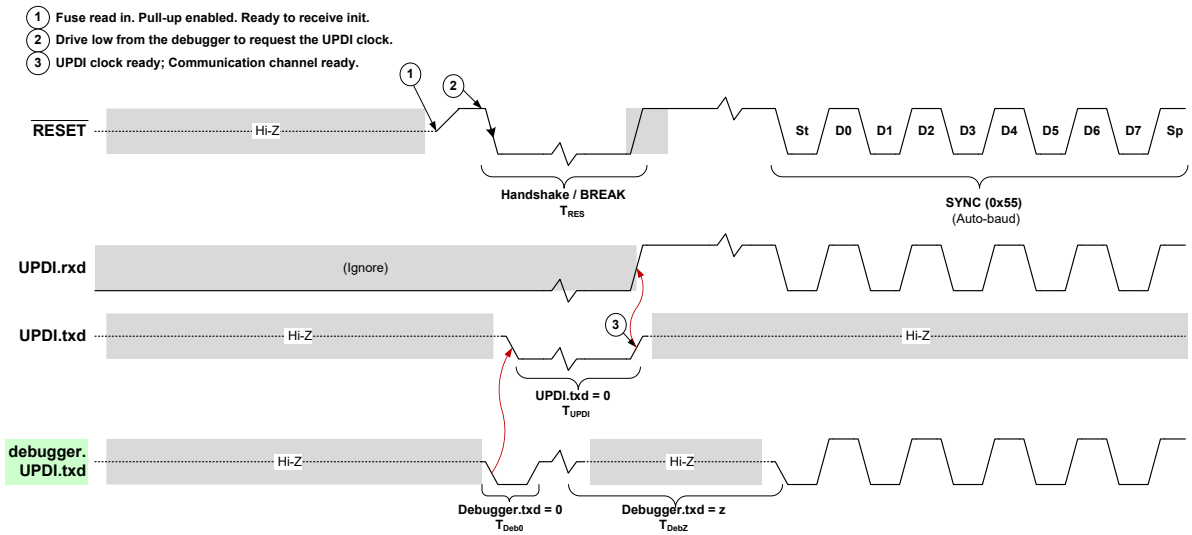


To avoid the UPDI from staying enabled if an accidental trigger of the edge detector happens, the UPDI will automatically disable itself and lower its clock request. See the *Disable During Start-up* section for more details.

33.3.2.1.2 UPDI Enable with Fuse Override of $\overline{\text{RESET}}$ Pin

When the $\overline{\text{RESET}}$ Pin Configuration (RSTPINCFG) bit in FUSE.SYSCFG0 is 0x1, the $\overline{\text{RESET}}$ pin will be overridden, and the UPDI will take control of the pin and configure it as an input with pull-up. When the pull-up is detected by a connected debugger, the UPDI enable sequence, as depicted below, is started.

Figure 33-5. UPDI Enable Sequence with UPDI PAD Enabled By Fuse



When the pull-up is detected, the debugger initiates the enable sequence by driving the line low for a duration of T_{Deb0}.

The negative edge is detected by the UPDI, which starts the UPDI clock. The UPDI will continue to drive the line low until the clock is stable and ready for the UPDI to use. The duration of T_{UPDI} will vary, depending on the status of the oscillator when the UPDI is enabled. After this duration, the data line will be released by the UPDI and pulled high.

When the debugger detects that the line is high, the initial SYNCH character (0x55) must be transmitted to synchronize the UPDI communication data rate. If the Start bit of the SYNCH character is not sent within maximum

T_{DebZ}, the UPDI will disable itself, and the UPDI enabling sequence must be reinitiated. The UPDI is disabled if the timing is violated to avoid the UPDI being enabled unintentionally.

After successful SYNCH character transmission, the first instruction frame can be transmitted.

33.3.2.1.3 UPDI Enable with High-Voltage Override of $\overline{\text{RESET}}$ Pin

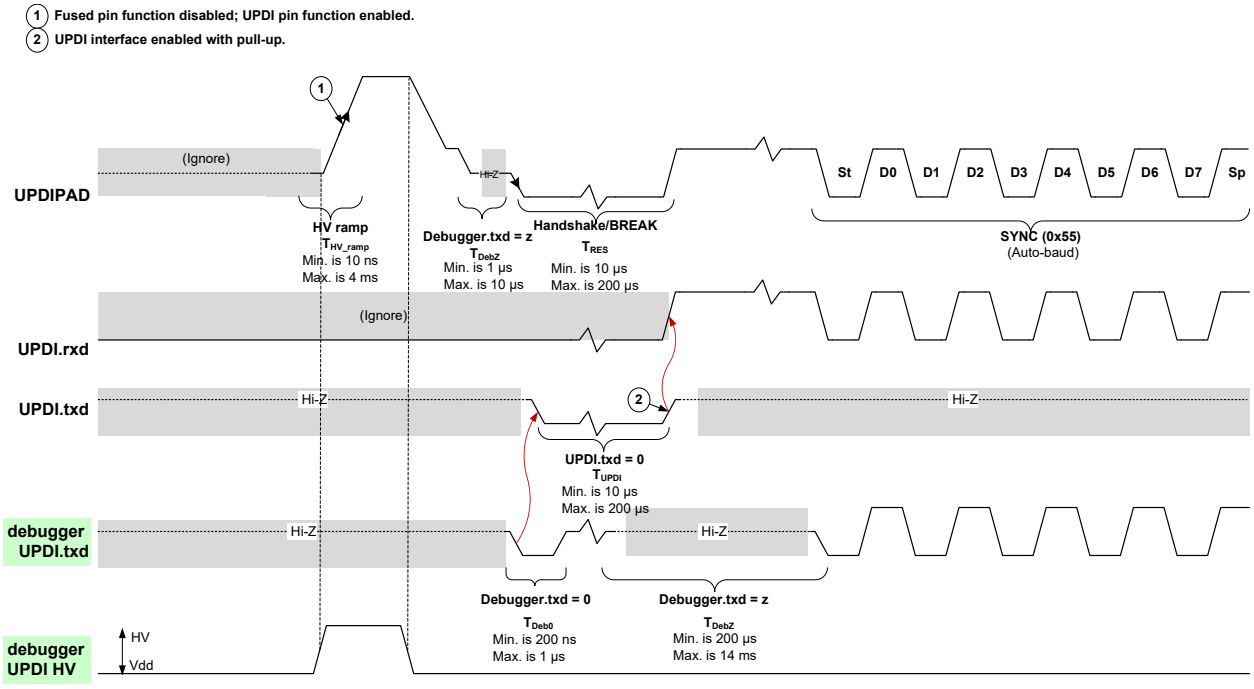
GPIO or Reset functionality on the $\overline{\text{RESET}}$ pin can be overridden by the UPDI by using high-voltage (HV) programming. Applying a HV pulse to the $\overline{\text{RESET}}$ pin will switch the pin functionality to UPDI. This is independent of the $\overline{\text{RESET}}$ Pin Configuration (RSTPINCFG) in FUSE.SYSCFG0. Follow these steps to override the pin functionality:

1. **Recommended:** Reset the device before starting the HV enable sequence.
2. Apply the HV signal, as described in the figure below.
3. Send the NVMPROG key using the `key` instruction after the first SYNC character.
4. After the programming is finished, reset the UPDI by writing the UPDI Disable (UPDIDIS) bit in the Control B (UPDI.CTRLB) register to '`1`' using the `STCS` instruction.

During power-up, the Power-on Reset (POR) must be released before the HV pulse can be applied. The duration of the pulse is recommended in the range from 100 µs to 1 ms before tri-stating.

When applying the rising edge of the HV pulse, the UPDI will be reset. After tri-stating the UPDI will remain in Reset until the $\overline{\text{RESET}}$ pin is driven low by the debugger. This will release the UPDI Reset, and initiate the same enable sequence as explained in 33.3.2.1.2 UPDI Enable with Fuse Override of RESET Pin.

Figure 33-6. UPDI Enable Sequence by High-Voltage (HV) Programming



When enabled by an HV pulse, only a POR will disable the UPDI configuration on the $\overline{\text{RESET}}$ pin and restore the default setting. If issuing a UPDI Disable command through the UPDIDIS bit in UPDI.CTRLB, the UPDI will be reset, and the clock request will be canceled, but the $\overline{\text{RESET}}$ pin will remain in UPDI configuration.

Note: If insufficient external protection is added to the UPDI pin, an ESD pulse can be interpreted by the device as a high-voltage override and enable the UPDI.

Note: The actual threshold voltage for the UPDI HV activation depends on V_{DD}. See the Electrical Characteristics for details.

33.3.2.1.4 Output Enable Timer Protection for GPIO Configuration

When the $\overline{\text{RESET}}$ Pin Configuration (RSTPINCFG) bit in FUSE.SYSCFG0 is '`0x0`', the $\overline{\text{RESET}}$ pin is configured as GPIO. To avoid a potential conflict between the GPIO actively driving the output and a UPDI high-voltage (HV) enable sequence initiation, the GPIO output driver is disabled for a minimum of 8.8 ms after a System Reset.

It is always recommended to issue a System Reset before entering the HV programming sequence.

33.3.2.2 UPDI Disabling

33.3.2.2.1 Disable During Start-up

During the enable sequence, the UPDI can disable itself in case of an invalid enable sequence. There are two mechanisms implemented to reset any requests the UPDI has given to the Power Management and set the UPDI to the disabled state. A new enable sequence must then be initiated to enable the UPDI.

Time-Out Disable

When the start-up negative edge detector releases the pin after the UPDI has received its clock, or when the regulator is stable and the system has power in a Multi-Voltage system, the default pull-up drives the UPDI pin high. If the programmer does not detect that the pin is high, and does not initiate a transmission of the SYNCH character within 16.4 ms at 4 MHz UPDI clock after the UPDI has released the pin, the UPDI will disable itself.

Note: Start-up oscillator frequency is device-dependent. The UPDI will count for 65536 cycles on the UPDI clock before issuing the time-out.

Incorrect SYNCH pattern

An incorrect SYNCH pattern is detected if the length of the SYNCH character is longer than the number of samples that can be contained in the UPDI Baud Rate register (overflow), or shorter than the minimum fractional count that can be handled for the sampling length of each bit. If any of these errors are detected, the UPDI will disable itself.

33.3.2.2.2 UPDI Regular Disable

Any programming or debugging session that does not require any specific operation from the UPDI after disconnecting the programmer has to be terminated by writing the UPDI Disable (UPDIDIS) bit in the Control B (UPDI.CTRLB) register, upon which the UPDI will issue a System Reset and disable itself. The Reset will restore the CPU to the Run state, independent of the previous state. It will also lower the UPDI clock request to the system, and reset any UPDI KEYs and settings.

If the disable operation is not performed, the UPDI and the oscillator's request will remain enabled. This causes increased power consumption for the application.

33.3.2.3 UPDI Communication Error Handling

The UPDI contains a comprehensive error detection system that provides information to the debugger when recovering from an error scenario. The error detection consists of detecting physical transmission errors like parity error, contention error, and frame error, to more high-level errors like access time-out error. See the UPDI Error Signature (PESIG) bit field in the Status B (UPDI.STATUSB) register for an overview of the available error signatures.

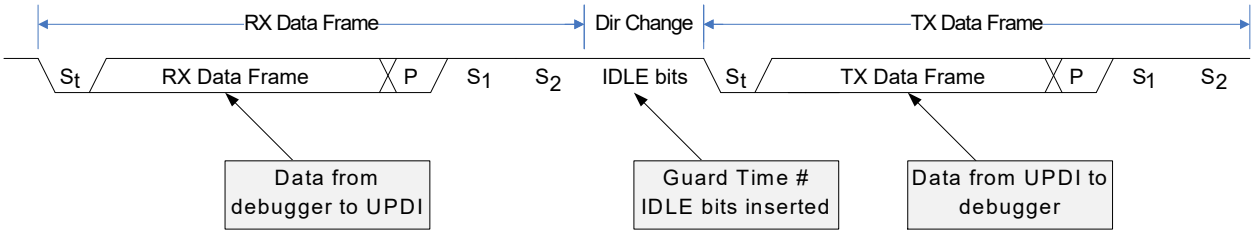
Whenever the UPDI detects an error, it will immediately enter an internal Error state to avoid unwanted system communication. In the Error state, the UPDI will ignore all incoming data requests, except when a BREAK character is received. The following procedure must always be applied when recovering from an Error condition.

1. Send a BREAK character. See the 33.3.1.2 BREAK Character section for recommended BREAK character handling.
2. Send a SYNCH character at the desired baud rate for the next data transfer.
3. Execute a Load Control Status (`LDCS`) instruction to read the UPDI Error Signature (PESIG) bit field in the Status B (UPDI.STATUSB) register and get the information about the occurred error.
4. The UPDI has now recovered from the Error state and is ready to receive the next SYNCH character and instruction.

33.3.2.4 Direction Change

To ensure correct timing for a half-duplex UART operation, the UPDI has a built-in guard time mechanism to relax the timing when changing direction from RX to TX mode. The guard time is represented by Idle bits inserted before the next Start bit of the first response byte is transmitted. The number of Idle bits can be configured through the Guard Time Value (GTVAL) bit field in the Control A (UPDI.CTRLA) register. The duration of each Idle bit is given by the baud rate used by the current transmission.

Figure 33-7. UPDI Direction Change by Inserting Idle Bits



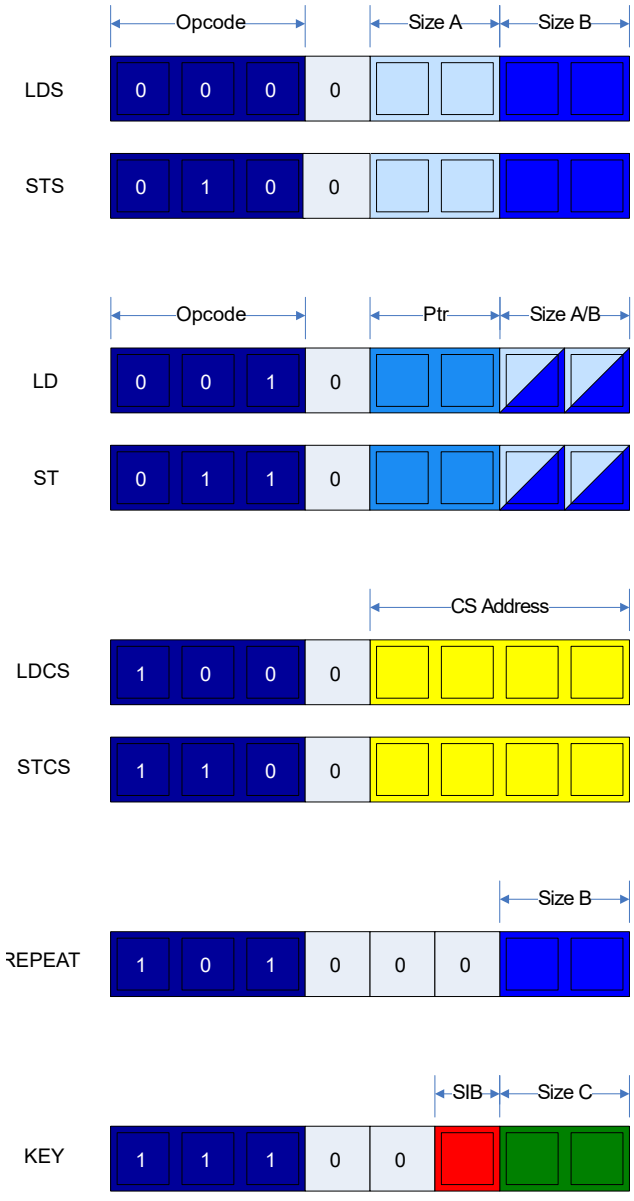
The UPDI guard time is the minimum Idle time that the connected debugger will experience when waiting for data from the UPDI. The maximum Idle time is the same as time-out. The Idle time before a transmission will be more than the expected guard time when the synchronization time plus the data bus accessing time is longer than the guard time.

It is recommended to always use the insertion of minimum two Guard Time bits on the UPDI side, and one guard time cycle insertion from the debugger side.

33.3.3 UPDI Instruction Set

The communication through the UPDI is based on a small instruction set. These instructions are part of the UPDI Data Link (DL) layer. The instructions are used to access the UPDI registers, since they are mapped into an internal memory space called “ASI Control and Status (CS) space”, as well as the memory-mapped system space. All instructions are byte instructions and must be preceded by a SYNCH character to determine the baud rate for the communication. See 33.3.1.1 UPDI UART for information about setting the baud rate for the transmission. The following figure gives an overview of the UPDI instruction set.

Figure 33-8. UPDI Instruction Set Overview



OPCODE			
0	0	0	LDS
0	0	1	LD
0	1	0	STS
0	1	1	ST
1	0	0	LDCS (LDS Control/Status)
1	0	1	REPEAT
1	1	0	STCS (STS Control/Status)
1	1	1	KEY

Size A - Address Size		
0	0	Byte - can address 0-255 B
0	1	Word (2 Bytes) - for memories up to 64 KB in size
1	0	3 Bytes - for memories above 64 KB in size
1	1	Reserved

Ptr - Pointer Access		
0	0	*(ptr)
0	1	*(ptr++)
1	0	ptr
1	1	Reserved

Size B - Data Size		
0	0	Byte
0	1	Word (2 Bytes)
1	0	Reserved
1	1	Reserved

CS Address (CS - Control/Status reg.)

Size C - Key Size		
0	0	64 bits (8 Bytes)
0	1	128 bits (16 Bytes)
1	0	Reserved
1	1	Reserved

SIB - System Information Block Sel.	
0	Receive KEY
1	Send SIB

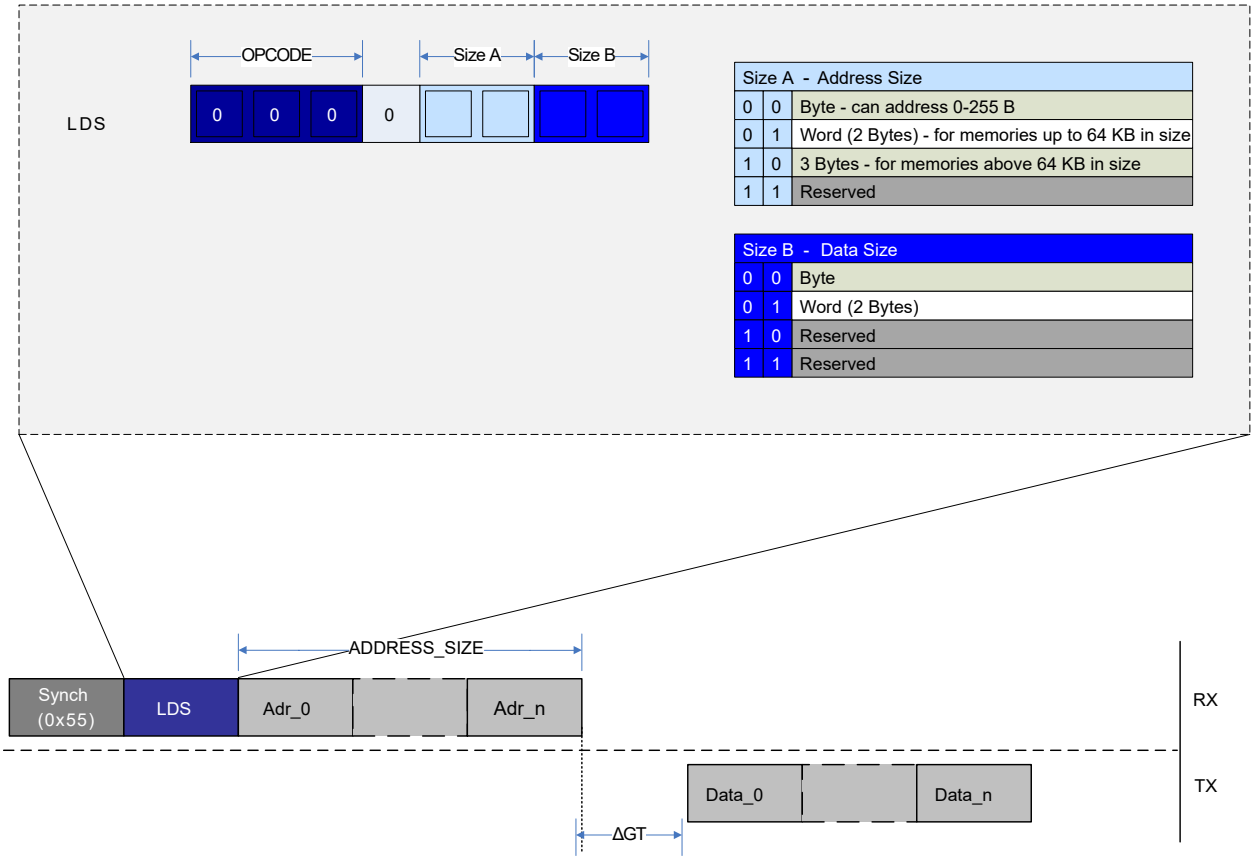
33.3.3.1 LDS - Load Data from Data Space Using Direct Addressing

The LDS instruction is used to load data from the system bus into the PHY layer shift register for serial readout. The LDS instruction is based on direct addressing, and the address must be given as an operand to the instruction for the

data transfer to start. The maximum supported size for the address and data is 32 bits. The LDS instruction supports repeated memory access when combined with the REPEAT instruction.

After issuing the LDS instruction, the number of desired address bytes, as indicated by the Size A field followed by the output data size, which is selected by the Size B field, must be transmitted. The output data is issued after the specified Guard Time (GT). When combined with the REPEAT instruction, the address must be sent in for each iteration of the repeat, meaning after each time the output data sampling is done. There is no automatic address increment when using REPEAT with LDS, as it uses a direct addressing protocol.

Figure 33-9. LDS Instruction Operation



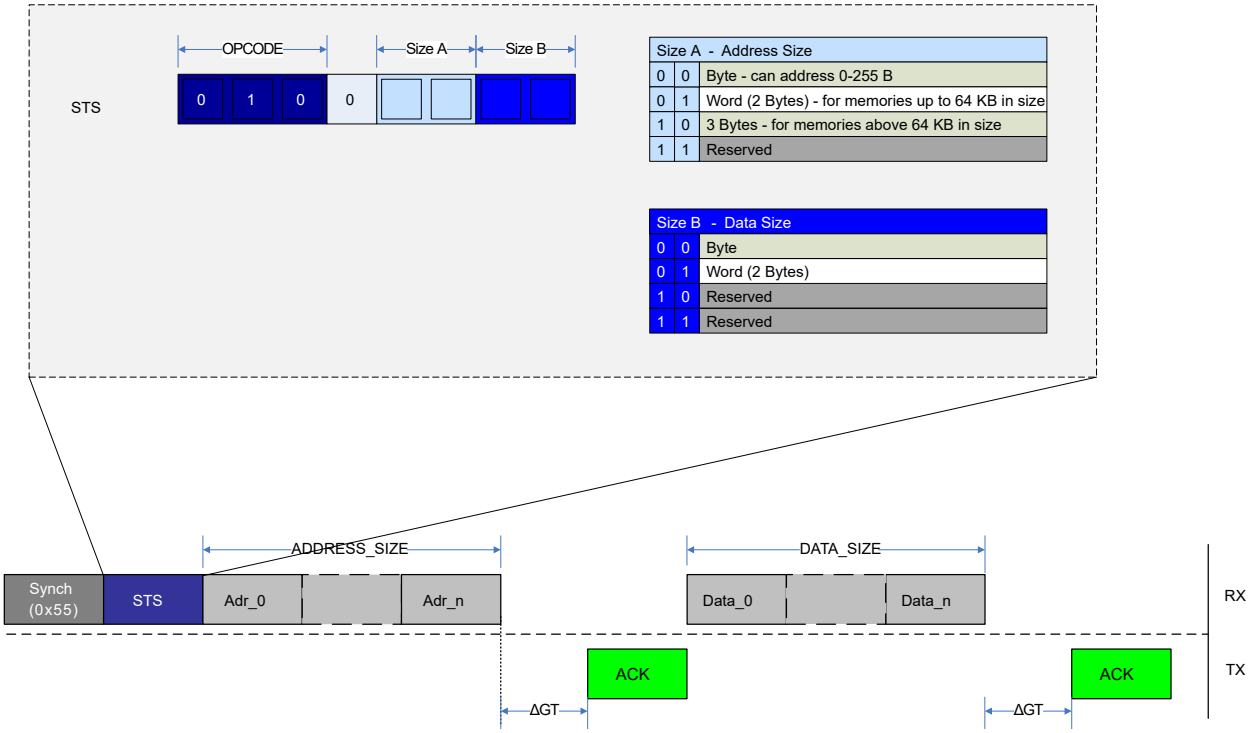
When the instruction is decoded, and the address byte(s) are received as dictated by the decoded instruction, the DL layer will synchronize all required information to the ACC layer, which will handle the bus request and synchronize data buffered from the bus back again to the DL layer. This will create a synchronization delay that must be taken into consideration upon receiving the data from the UPDI.

33.3.3.2 STS - Store Data to Data Space Using Direct Addressing

The STS instruction is used to store data that are shifted serially into the PHY layer shift register to the system bus address space. The STS instruction is based on direct addressing, and the address must be given as an operand to the instruction for the data transfer to start. The address is the first set of operands, and data are the second set. The size of the address and data operands are given by the size fields presented in the figure below. The maximum size for both address and data is 32 bits.

The STS supports repeated memory access when combined with the REPEAT instruction.

Figure 33-10. `STS` Instruction Operation



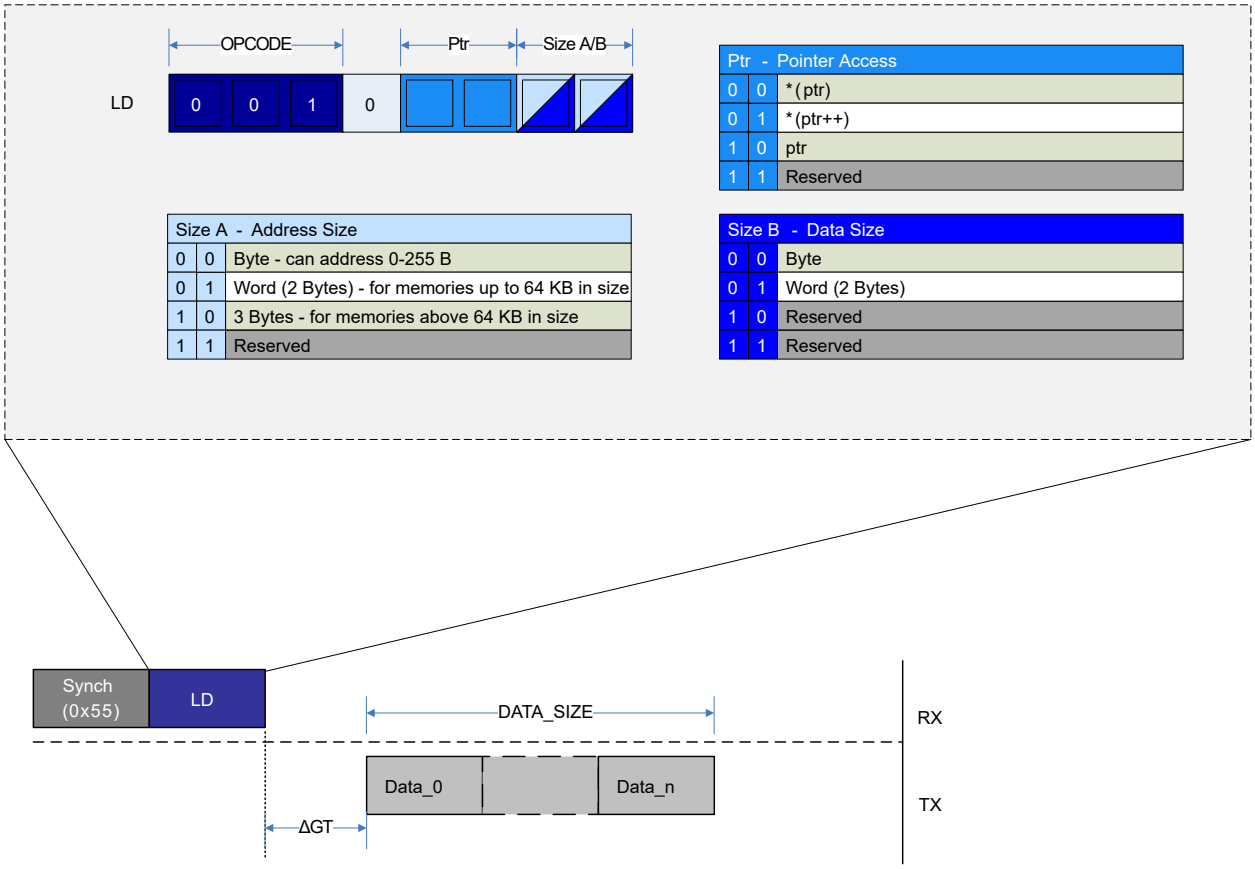
The transfer protocol for an `STS` instruction is depicted in the above figure, following this sequence:

1. The address is sent.
2. An Acknowledge (ACK) is sent back from the UPDI if the transfer was successful.
3. The number of bytes, as specified in the `STS` instruction, is sent.
4. A new ACK is received after the data have been successfully transferred.

33.3.3.3 `LD` - Load Data from Data Space Using Indirect Addressing

The `LD` instruction is used to load data from the data space and into the PHY layer shift register for serial readout. The `LD` instruction is based on indirect addressing, which means that the Address Pointer in the UPDI needs to be written before the data space read access. Automatic pointer post-increment operation is supported and is useful when the `LD` instruction is utilized with the `REPEAT` instruction. It is also possible to do an `LD` from the UPDI Pointer register. The maximum supported size for address and data load is 32 bits.

Figure 33-11. LD Instruction Operation



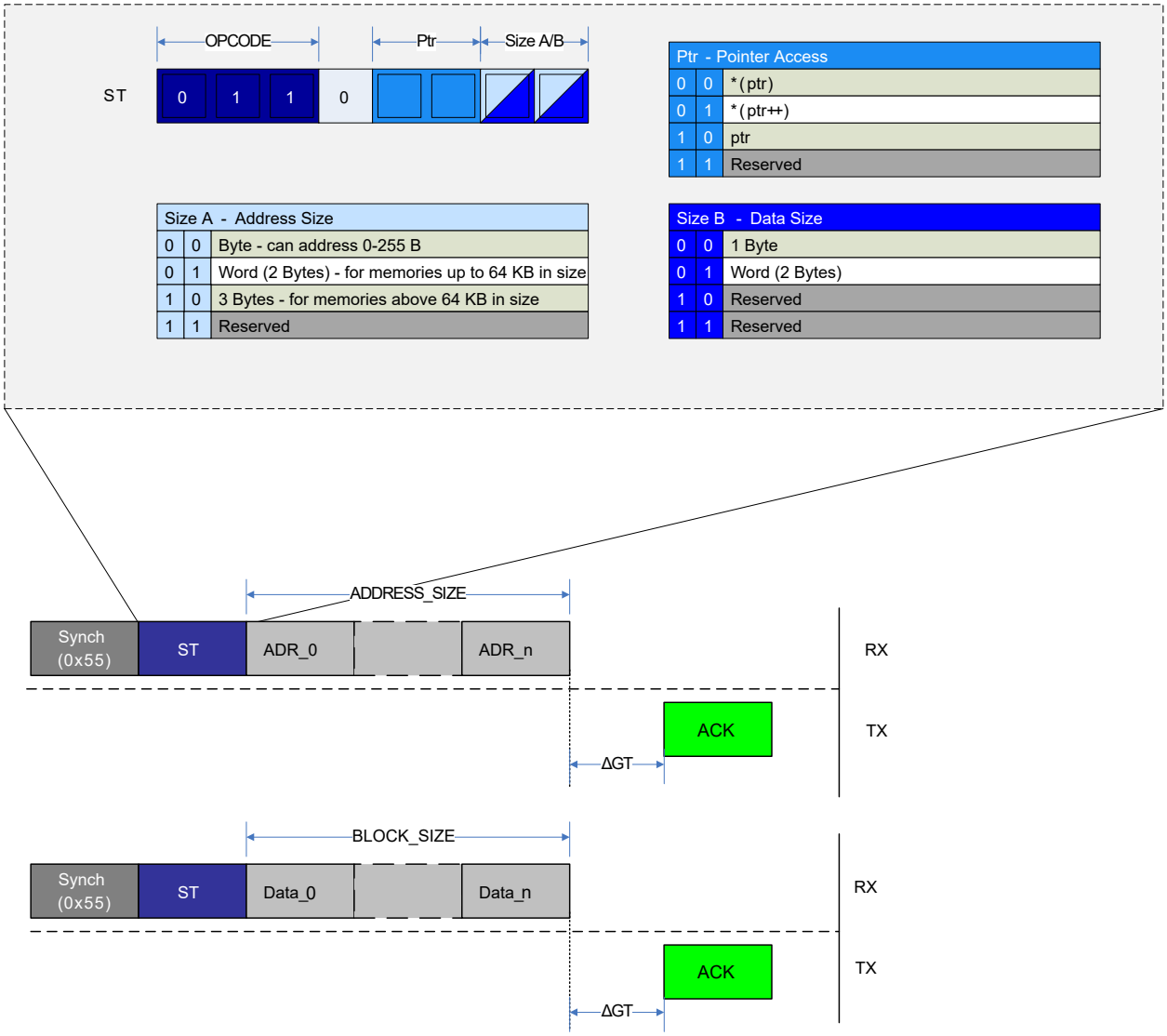
The figure above shows an example of a typical LD sequence, where the data are received after the Guard Time (GT) period. Loading data from the UPDI Pointer register follows the same transmission protocol.

For the LD instruction from the data space, the pointer register must be set up by using an ST instruction to the UPDI Pointer register. After the ACK has been received on a successful Pointer register write, the LD instruction must be set up with the desired DATA SIZE operands. An LD to the UPDI Pointer register is done directly with the LD instruction.

33.3.3.4 ST - Store Data from UPDI to Data Space Using Indirect Addressing

The ST instruction is used to store data from the UPDI PHY shift register to the data space. The ST instruction is used to store data that are shifted serially into the PHY layer. The ST instruction is based on indirect addressing, which means that the Address Pointer in the UPDI needs to be written before the data space. The automatic pointer post-increment operation is supported and is useful when the ST instruction is utilized with the REPEAT instruction. The ST instruction is also used to store the UPDI Address Pointer into the Pointer register. The maximum supported size for storing address and data is 32 bits.

Figure 33-12. ST Instruction Operation



The figure above gives an example of an ST instruction to the UPDI Pointer register and the storage of regular data. A SYNCH character is sent before each instruction. In both cases, an Acknowledge (ACK) is sent back by the UPDI if the ST instruction was successful.

To write the UPDI Pointer register, the following procedure has to be followed:

1. Set the PTR field in the ST instruction to signature 0x2.
2. Set the address size (Size A) field to the desired address size.
3. After issuing the ST instruction, send Size A bytes of address data.
4. Wait for the ACK character, which signifies a successful write to the Address register.

After the Address register is written, sending data is done in a similarly:

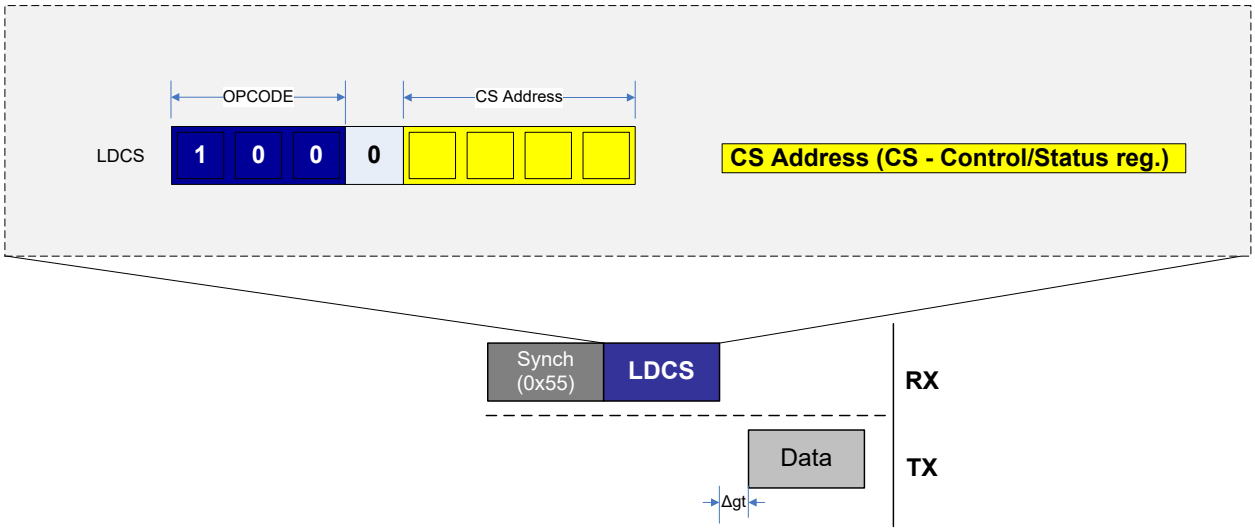
1. Set the PTR field in the ST instruction to signature 0x0 to write to the address specified by the UPDI Pointer register. If the PTR field is set to 0x1, the UPDI pointer is automatically updated to the next address according to the data size Size B field of the instruction after the write is executed.
2. Set the Size B field in the instruction to the desired data size.
3. After sending the ST instruction, send Size B bytes of data.
4. Wait for the ACK character, which signifies a successful write to the bus matrix.

When used with the REPEAT instruction, it is recommended to set up the Address register with the start address for the block to be written and use the Pointer Post Increment register to automatically increase the address for each repeat cycle. When using the REPEAT instruction, the data frame of Size B data bytes can be sent after each received ACK.

33.3.3.5 LDCS - Load Data from Control and Status Register Space

The LDCS instruction is used to load serial readout data from the UPDI Control and the Status register space located in the DL layer into the PHY layer shift register. The LDCS instruction is based on direct addressing, where the address is part of the instruction operands. The LDCS instruction can access only the UPDI CS register space. This instruction supports only byte access, and the data size is not configurable.

Figure 33-13. LDCS Instruction Operation

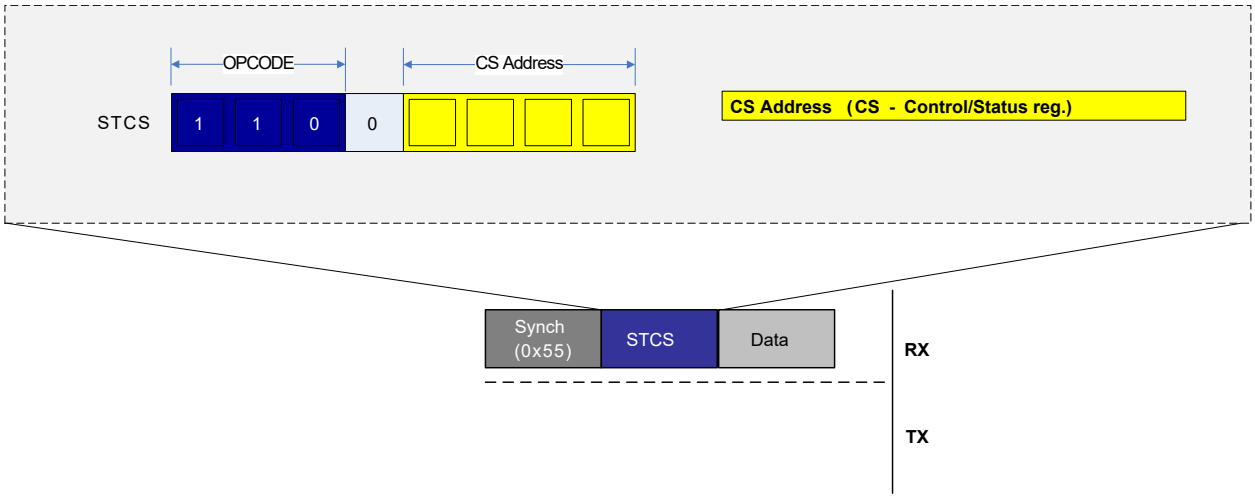


The figure above shows a typical example of LDCS data transmission. A data byte from the LDCS is transmitted from the UPDI after the guard time is completed.

33.3.3.6 STCS (Store Data to Control and Status Register Space)

The STCS instruction is used to store data to the UPDI Control and Status register space. Data are shifted in serially into the PHY layer shift register and written as a whole byte to a selected CS register. The STCS instruction is based on direct addressing, where the address is part of the instruction operand. The STCS instruction can access only the internal UPDI register space. This instruction supports only byte access, and the data size is not configurable.

Figure 33-14. STCS Instruction Operation



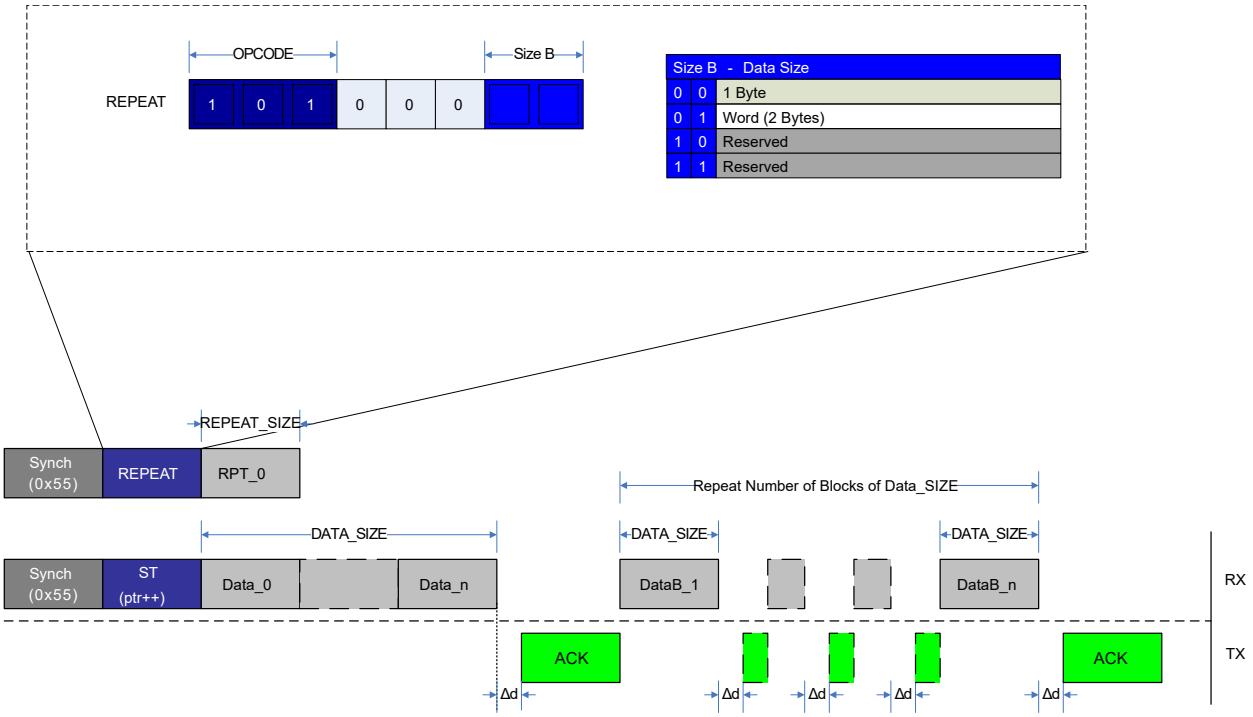
The figure above shows the data frame transmitted after the SYNCH character and the instruction frames. The STCS instruction byte can be immediately followed by the data byte. There is no response generated from the STCS instruction, as is the case for the ST and STS instructions.

33.3.3.7 REPEAT - Set Instruction Repeat Counter

The REPEAT instruction is used to store the repeat count value into the UPDI Repeat Counter register on the DL layer. When instructions are used with REPEAT, the protocol overhead for SYNCH and instruction frame can be omitted on all instructions except the first instruction after the REPEAT is issued. REPEAT is most useful for memory instructions (LD, ST, LDS, STS), but all instructions can be repeated, except for the REPEAT instruction itself.

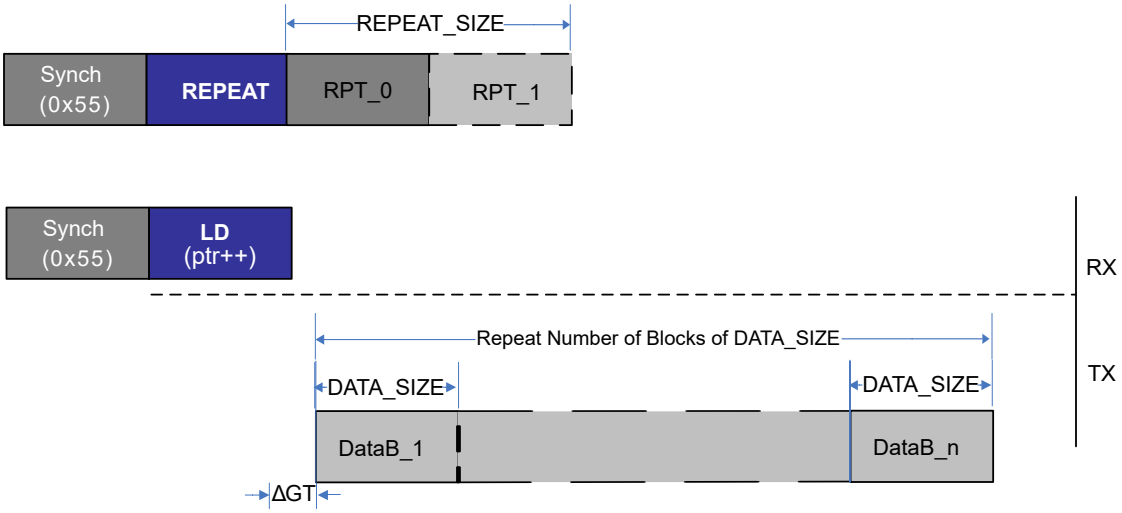
The DATA_SIZE operand field refers to the size of the repeat value. Only up to 255 repeats are supported. The instruction loaded directly after the REPEAT instruction will be issued for RPT_0 + 1 times. If the Repeat Counter register is '0', the instruction will run just once. An ongoing repeat can be aborted only by sending a BREAK character.

Figure 33-15. REPEAT Instruction Operation used with ST Instruction



The figure above gives an example of repeat operation with an ST instruction using pointer post-increment operation. After the REPEAT instruction is sent with RPT_0 = ***n***, the first ST instruction is issued with SYNCH and instruction frame, while the next ***n*** ST instructions are executed by only sending data bytes according to the ST operand DATA_SIZE, and maintaining the Acknowledge (ACK) handshake protocol.

Figure 33-16. `REPEAT` used with `LD` Instruction



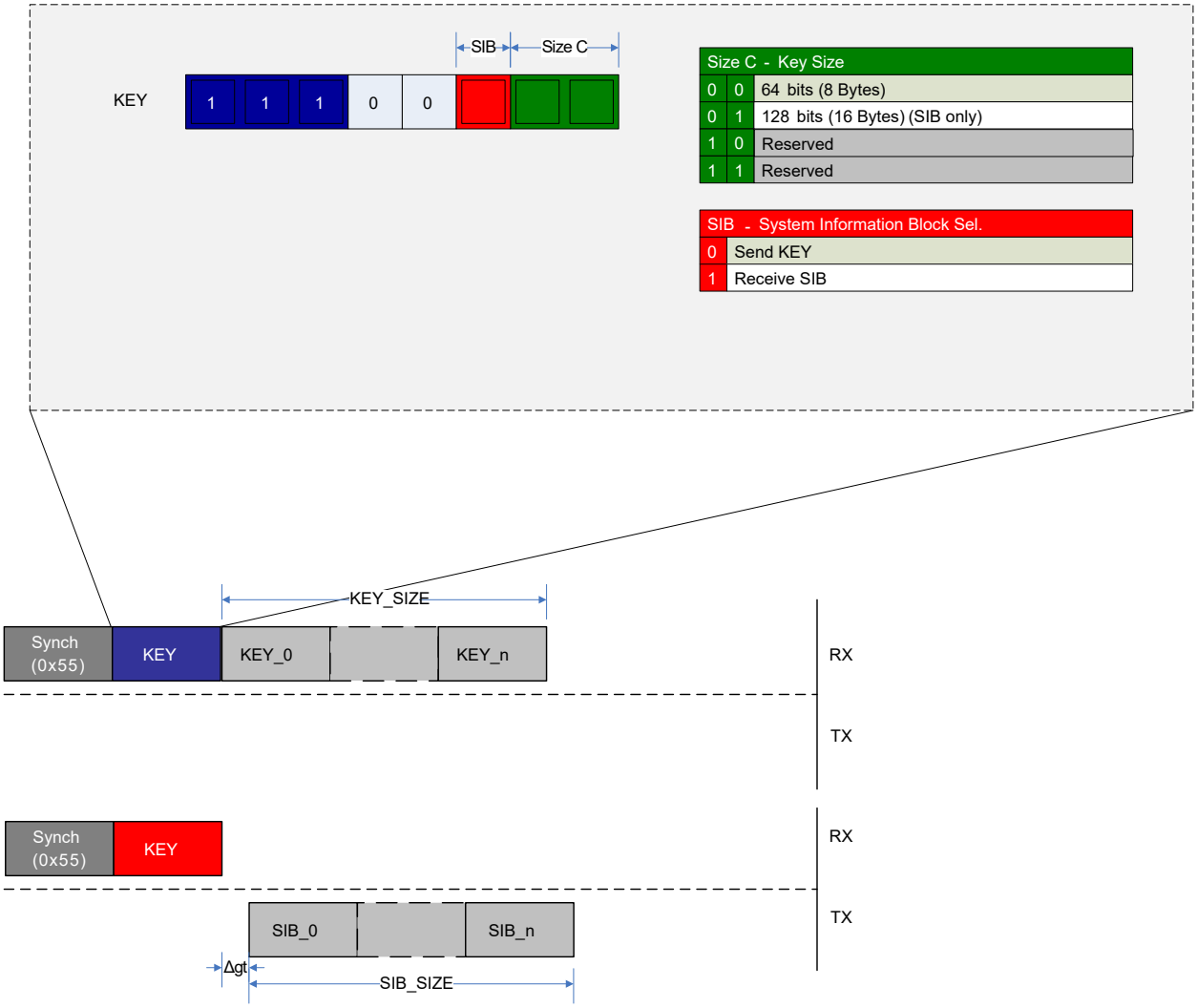
For `LD`, data will come out continuously after the `LD` instruction. Note the guard time on the first data block.

If using indirect addressing instructions (`LD/ST`), it is recommended to always use the pointer post-increment option when combined with `REPEAT`. The `ST/LD` instruction is necessary only before the first data block (number of data bytes determined by DATA_SIZE). Otherwise, the same address will be accessed in all repeated access operations. For direct addressing instructions (`LDS/STS`), the address must always be transmitted as specified in the instruction protocol, before data can be received (`LDS`) or sent (`STS`).

33.3.3.8 `KEY` - Set Activation Key or Send System Information Block

The `KEY` instruction is used for communicating key bytes to the UPDI or for providing the programmer with a System Information Block (SIB), opening up for executing protected features on the device. See Table 33-4 for an overview of functions that are activated by keys. For the `KEY` instruction, only a 64-bit key size is supported. The maximum supported size for SIB is 128 bits.

Figure 33-17. `KEY` Instruction Operation



The figure above shows the transmission of a key and the reception of a SIB. In both cases, the Size C (`SIZE_C`) field in the operand determines the number of frames being sent or received. There is no response after sending a `KEY` to the UPDI. When requesting the SIB, data will be transmitted from the UPDI according to the current guard time setting.

33.3.4 CRC Checking of Flash During Boot

Some devices support running a CRC check of the Flash contents as part of the boot process. This check can be performed even when the device is locked. The result of this CRC check can be read from the ASI_CRC_STATUS register. Refer to the *CRCSCAN* section in the device data sheet for more information on this feature.

33.3.5 System Clock Measurement with UPDI

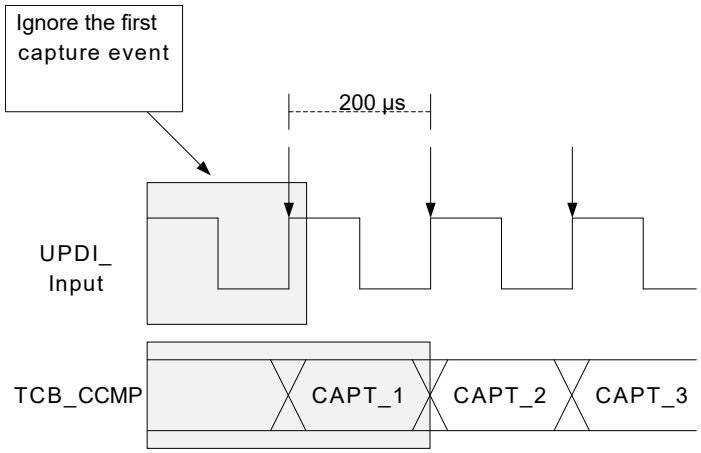
It is possible to use the UPDI to get an accurate measurement of the system clock frequency by utilizing the UPDI event connected to TCB with Input Capture capabilities. A recommended setup flow for this feature is given by the following steps:

- Set up TCBn.CTRLB with setting CNTMODE = 0x3, Input Capture Frequency Measurement mode.
- Write CAPTEI = `1` in TCBn.EVCTRL to enable Event Interrupt. Keep EDGE = `0` in TCBn.EVCTRL
- Configure the Event System to route the UPDI SYNCH event (generator) to the TCB (user).
- For the SYNCH character used to generate the UPDI events, it is recommended to use a slow baud rate in the range of 10-50 kbps to get a more accurate measurement on the value captured by the timer between each UPDI event. One particular thing is that if the capture is set up to trigger an interrupt, the first captured value

should be ignored. The second captured value based on the input event should be used for the measurement. See the figure below for an example using 10 kbps UPDI SYNCH character pulses, giving a capture window of 200 µs for the timer.

- It is possible to read out the captured value directly after the SYNCH character by reading the TCBn.CCMP register or the value can be written to memory by the CPU once the capture is done.

Figure 33-18. UPDI System Clock Measurement Events



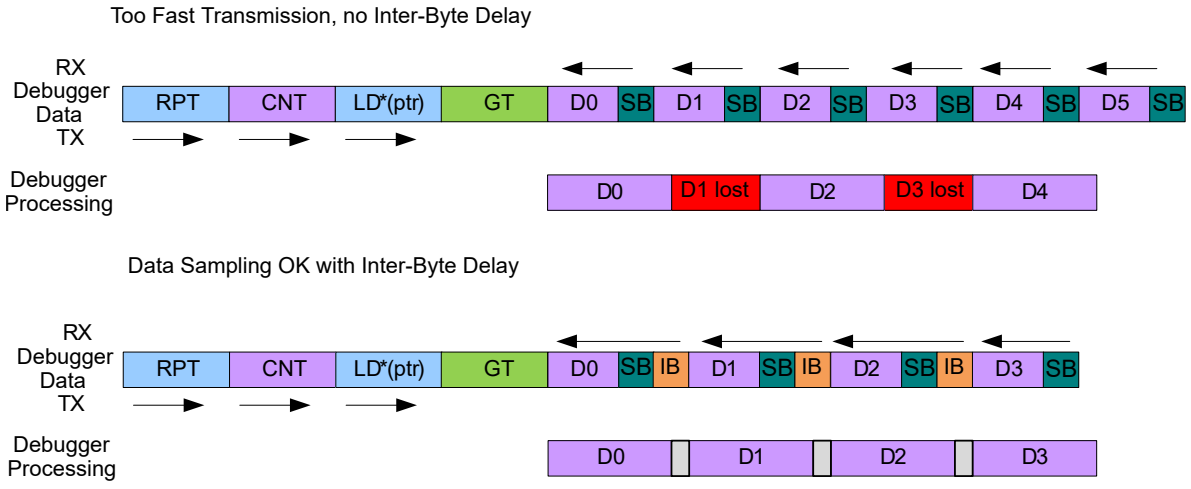
33.3.6 Inter-Byte Delay

When performing a multi-byte transfer (`LD` combined with `REPEAT`), or reading out the System Information Block (SIB), the output data will come out in a continuous stream. Depending on the application, on the receiver side, the data might come out too fast, and there might not be enough time for the data to be processed before the next Start bit arrives.

The inter-byte delay works by inserting a fixed number of Idle bits for multi-byte transfers. The reason for adding an inter-byte delay is that there is no guard time inserted when all data is going in the same direction.

The inter-byte delay feature can be enabled by writing a '`1`' to the Inter-Byte Delay Enable (IBDLY) bit in the Control A (UPDI.CTRLA) register. As a result, two extra Idle bits will be inserted between each byte to relax the sampling time for the debugger.

Figure 33-19. Inter-Byte Delay Example with LD and RPT



Note:

1. GT denotes the guard time insertion.
2. SB is for Stop bit.
3. IB is the inserted inter-byte delay.
4. The rest of the frames are data and instructions.

33.3.7 System Information Block

The System Information Block (SIB) can be read out at any time by setting the SIB bit according to the KEY instruction from 33.3.3.8 KEY - Set Activation Key or Send System Information Block. The SIB is always accessible to the debugger, regardless of lock bit settings, and provides a compact form of supplying information about the device and system parameters for the debugger. The information is vital in identifying and setting up the proper communication channel with the device. The output of the SIB is interpreted as ASCII symbols. The key size field must be set to 16 bytes when reading out the complete SIB, and an 8-byte size can be used to read out only the Family_ID. See the figure below for SIB format description and which data are available at different readout sizes.

Figure 33-20. System Information Block Format

16	8	[Byte][Bits]	Field Name
		[6:0] [55:0]	Family_ID
		[7][7:0]	Reserved
		[10:8][23:0]	NVM_VERSION
		[13:11][23:0]	OCD_VERSION
		[14][7:0]	RESERVED
		[15][7:0]	DBG_OSC_FREQ

33.3.8 Enabling of Key Protected Interfaces

The access to some internal interfaces and features is protected by the UPDI key mechanism. To activate a key, the correct key data must be transmitted by using the KEY instruction, as described in 33.3.3.8 KEY - Set Activation Key or Send System Information Block. The table below describes the available keys and the condition required when doing the operation with the key active.

Table 33-4. Key Activation Overview

Key Name	Description	Requirements for Operation	Conditions for Key Invalidation
Chip Erase	Start NVM chip erase. Clear lock bits	-	UPDI Disable/UPDI Reset
NVMPROG	Activate NVM Programming	Lock bits cleared. ASI_SYS_STATUS.NVMPROG set	Programming done/UPDI Reset
USERROW-Write	Program the user row on the locked device	Lock bits set. ASI_SYS_STATUS.UROWPROG set	Write to key Status bit/ UPDI Reset

The table below gives an overview of the available key signatures that must be shifted in to activate the interfaces.

Table 33-5. Key Activation Signatures

Key Name	Key Signature (LSB Written First)	Size
Chip Erase	0x4E564D4572617365	64 bits
NVMPROG	0x4E564D50726F6720	64 bits
USERROW-Write	0x4E564D5573267465	64 bits

33.3.8.1 Chip Erase

The following steps must be followed to issue a chip erase:

1. Enter the Chip Erase key by using the KEY instruction. See Table 33-5 for the CHIPERASE signature.

2. **Optional:** Read the Chip Erase (CHIPERASE) bit in the ASI Key Status (UPDI.ASI_KEY_STATUS) register to see that the key is successfully activated.
3. Write the signature to the Reset Request (RSTREQ) bit in the ASI Reset Request (UPDI.ASI_RESET_REQ) register. This will issue a System Reset.
4. Write `0x00` to the ASI Reset Request (UPDI.ASI_RESET_REQ) register to clear the System Reset.
5. Read the NVM Lock Status (LOCKSTATUS) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
6. The chip erase is done when LOCKSTATUS bit is '0'. If the LOCKSTATUS bit is '1', return to step 5.
7. Check the Chip Erase Key Failed (ERASE_FAILED) bit in the ASI System Status (UPDI.ASI_SYS_STATUS) register to verify if the chip erase was successful.
8. If the ERASE_FAILED bit is '0', the chip erase was successful.

After a successful chip erase, the lock bits will be cleared, and the UPDI will have full access to the system. Until the lock bits are cleared, the UPDI cannot access the system bus, and only CS-space operations can be performed.

⚠ CAUTION During chip erase, the BOD is forced in ON state by writing to the Active (ACTIVE) bit field from the Control A (BOD.CTRLA) register and uses the BOD Level (LVL) bit field from the BOD Configuration (FUSE.BODCFG) fuse and the BOD Level (LVL) bit field from the Control B (BOD.CTRLB) register. If the supply voltage V_{DD} is below that threshold level, the device is unavailable until V_{DD} is increased adequately. See the *BOD* section for more details.

33.3.8.2 NVM Programming

If the device is unlocked, it is possible to write directly to the NVM Controller or to the Flash memory using the UPDI. This will lead to unpredictable code execution if the CPU is active during the NVM programming. To avoid this, the following NVM Programming sequence has to be executed.

1. Follow the chip erase procedure, as described in 33.3.8.1 Chip Erase. If the part is already unlocked, this point can be skipped.
2. Enter the NVMPROG key by using the `KEY` instruction. See Table 33-5 for the NVMPROG signature.
3. **Optional:** Read the NVM Programming Key Status (NVMPROG) bit from the ASI Key Status (UPDI.KEY_STATUS) register to see if the key has been activated.
4. Write the signature to the Reset Request (RSTREQ) bit in the ASI Reset Request (UPDI.ASI_RESET_REQ) register. This will issue a System Reset.
5. Write `0x00` to the ASI Reset Request (UPDI.ASI_RESET_REQ) register to clear the System Reset.
6. Read the NVM Programming Key Status (NVMPROG) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
7. NVM Programming can start when the NVMPROG bit is '1'. If the NVMPROG bit is '0', return to step 6.
8. Write data to NVM through the UPDI.
9. Write the signature to the Reset Request (RSTREQ) bit in the ASI Reset Request (UPDI.ASI_RESET_REQ) register. This will issue a System Reset.
10. Write `0x00` to the ASI Reset Request (UPDI.ASI_RESET_REQ) register to clear the System Reset.
11. Programming is complete.

33.3.8.3 User Row Programming

The User Row Programming feature allows programming new values to the user row (USERROW) on a locked device. To program with this functionality enabled, the following sequence must be followed:

1. Enter the USERROW-Write key located in Table 33-5 by using the `KEY` instruction. See Table 33-5 for the USERROW-Write signature.
2. **Optional:** Read the User Row Write Key Status (UROWWRITE) bit from the ASI Key Status (UPDI.ASI_KEY_STATUS) register to see if the key has been activated.
3. Write the signature to the Reset Request (RSTREQ) bit in the ASI Reset Request (UPDI.ASI_RESET_REQ) register. This will issue a System Reset.
4. Write `0x00` to the ASI Reset Request (UPDI.ASI_RESET_REQ) register to clear the System Reset.

5. Read the Start User Row Programming (UROWPROG) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
6. User Row Programming can start when the UROWPROG bit is '1'. If UROWPROG is '0', return to step 5.
7. The data to be written to the User Row must first be written to a buffer in the RAM. The writable area in the RAM has a size of 64 bytes, and it is only possible to write user row data to the first 64 byte addresses of the RAM. Addressing outside this memory range will result in a nonexecuted write. The data will map 1:1 with the user row space when the data is copied into the user row upon completion of the Programming sequence.
8. When all user row data has been written to the RAM, write the User Row Programming Done (UROWDONE) bit in the ASI System Control A (UPDI.ASI_SYS_CTRLA) register.
9. Read the Start User Row Programming (UROWPROG) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
10. The User Row Programming is completed when UROWPROG bit is '0'. If UROWPROG bit is '1', return to step 9.
11. Write to the User Row Write Key Status (UROWWRITE) bit in the ASI Key Status (UPDI.ASI_KEY_STATUS) register.
12. Write the signature to the Reset Request (RSTREQ) bit in the ASI Reset Request (UPDI.ASI_RESET_REQ) register. This will issue a System Reset.
13. Write 0x00 to the ASI Reset Request (UPDI.ASI_RESET_REQ) register to clear the System Reset.
14. The User Row Programming is complete.

It is not possible to read back data from the RAM in this mode. Only writes to the first 64 bytes of the RAM are allowed.

33.3.9 Events

The UPDI can generate the following events:

Table 33-6. Event Generators in UPDI

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Module	Event				
UPDI	SYNCH	SYNCH character	Level	CLK_UPDI	SYNCH char on UPDI pin synchronized to CLK_UPDI

This event is set on the UPDI clock for each detected positive edge in the SYNCH character, and it is not possible to disable this event from the UPDI.

The UPDI has no event users.

Refer to the *Event System* section for more details regarding event types and Event System configuration.

33.3.10 Sleep Mode Operation

The UPDI PHY layer runs independently of all sleep modes, and the UPDI is always accessible for a connected debugger independent of the device's sleep state. If the system enters a sleep mode that turns the system clock off, the UPDI will not be able to access the system bus and read memories and peripherals. When enabled, the UPDI will request the system clock so that the UPDI always has contact with the rest of the device. Thus, the UPDI PHY layer clock is unaffected by the sleep mode's settings. By reading the System Domain in Sleep (INSLEEP) bit in the ASI System Status (UPDI.ASI_SYS_STATUS) register, it is possible to monitor if the system domain is in a sleep mode.

It is possible to prevent the system clock from stopping when going into a sleep mode, by writing to the Request System Clock (CLKREQ) bit in the ASI System Control A (UPDI.ASI_SYS_CTRLA) register. If this bit is set, the system sleep mode state is emulated, and the UPDI can access the system bus and read the peripheral registers even in the deepest sleep modes.

The CLKREQ bit is by default '1' when the UPDI is enabled, which means that the default operation is keeping the system clock in ON state during the sleep modes.

33.4 Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	STATUSA	7:0	UPDIREV[3:0]							
0x01	STATUSB	7:0						PESIG[2:0]		
0x02	CTRLA	7:0	IBDLY		PARD	DTD	RSD	GTVAL[2:0]		
0x03	CTRLB	7:0				NACKDIS	CCDETDIS	UPDIDIS		
0x04 ... 0x06	Reserved									
0x07	ASI_KEY_STATUS	7:0			UROWWRITE	NVMPROG	CHIPERASE			
0x08	ASI_RESET_REQ	7:0	RSTREQ[7:0]							
0x09	ASI_CTRLA	7:0							UPDICLKSEL[1:0]	
0x0A	ASI_SYS_CTRLA	7:0							UROWWRITE_FINAL	CLKREQ
0x0B	ASI_SYS_STATUS	7:0			RSTSYS	INSLEEP	NVMPROG	UROWPROG		LOCKSTATUS
0x0C	ASI_CRC_STATUS	7:0						CRC_STATUS[2:0]		

33.5 Register Description

These registers are readable only through the UPDI with special instructions and are not readable through the CPU.

33.5.1 Status A

Name: STATUSA
Offset: 0x00
Reset: 0x10
Property: -

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

Bits 7:4 – UPDIREV[3:0] UPDI Revision
This bit field contains the revision of the current UPDI implementation.

33.5.2 Status B

Name: STATUSB
Offset: 0x01
Reset: 0x00
Property: -

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

Bits 2:0 – PESIG[2:0] UPDI Error Signature
This bit field describes the UPDI error signature and is set when an internal UPDI Error condition occurs. The PESIG bit field is cleared on a read from the debugger.

Table 33-7. Valid Error Signatures

PESIG[2:0]	Error Type	Error Description
0x0	No error	No error detected (Default)
0x1	Parity error	Wrong sampling of the Parity bit
0x2	Frame error	Wrong sampling of the Stop bits
0x3	Access Layer Time-Out Error	UPDI can get no data or response from the Access layer
0x4	Clock Recovery error	Wrong sampling of the Start bit
0x5	-	Reserved
0x6	Bus error	Address error or access privilege error
0x7	Contention error	Signalize Driving Contention on the UPDI pin

33.5.3 Control A

Name: CTRLA
Offset: 0x02
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	IBDLY		PARD	DTD	RSD	GTVAL[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 – IBDLY Inter-Byte Delay Enable
Writing a '1' to this bit enables a fixed-length inter-byte delay between each data byte transmitted from the UPDI when doing multi-byte LD(S). The fixed length is two IDLE bits.

Bit 5 – PARD Parity Disable
Writing a '1' to this bit will disable the parity detection in the UPDI by ignoring the Parity bit. This feature is recommended to be used only during testing.

Bit 4 – DTD Disable Time-Out Detection
Writing a '1' to this bit will disable the time-out detection on the PHY layer, which requests a response from the ACC layer within a specified time (65536 UPDI clock cycles).

Bit 3 – RSD Response Signature Disable
Writing a '1' to this bit will disable any response signatures generated by the UPDI. This reduces the protocol overhead to a minimum when writing large blocks of data to the NVM space. When accessing the system bus, the UPDI may experience delays. If the delay is predictable, the response signature may be disabled, otherwise loss of data may occur.

Bits 2:0 – GTVAL[2:0] Guard Time Value
This bit field selects the guard time value that will be used by the UPDI when the transmission direction switches from RX to TX.

Value	Description
0x0	UPDI guard time: 128 cycles (default)
0x1	UPDI guard time: 64 cycles
0x2	UPDI guard time: 32 cycles
0x3	UPDI guard time: 16 cycles
0x4	UPDI guard time: 8 cycles
0x5	UPDI guard time: 4 cycles
0x6	UPDI guard time: 2 cycles
0x7	Reserved

33.5.4 Control B

Name: CTRLB
Offset: 0x03
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
				NACKDIS	CCDETDIS	UPDIDIS		
Access				R/W	R/W	R/W		
Reset				0	0	0		

Bit 4 – NACKDIS Disable NACK Response
Writing a '1' to this bit disables the NACK signature sent by the UPDI when a System Reset is issued during ongoing LD(S) and ST(S) operations.

Bit 3 – CCDETDIS Collision and Contention Detection Disable
Writing a '1' to this bit disables the contention detection. Writing a '0' to this bit enables the contention detection.

Bit 2 – UPDIDIS UPDI Disable
Writing a '1' to this bit disables the UPDI PHY interface. The clock request from the UPDI is lowered, and the UPDI is reset. All the UPDI PHY configurations and keys will be reset when the UPDI is disabled.

33.5.5 ASI Key Status

Name: ASI_KEY_STATUS
Offset: 0x07
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
			UROWWRITE	NVMPROG	CHIPERASE			
Access			R/W	R	R			
Reset			0	0	0			

Bit 5 – UROWWRITE User Row Write Key Status
This bit is set to '1' if the UROWWRITE key is successfully decoded. This bit must be written as the final part of the user row write procedure to correctly reset the programming session.

Bit 4 – NVMPROG NVM Programming Key Status
This bit is set to '1' if the NVMPROG key is successfully decoded. The bit is cleared when the NVM Programming sequence is initiated, and the NVMPROG bit in ASI_SYS_STATUS is set.

Bit 3 – CHIPERASE Chip Erase Key Status
This bit is set to '1' if the Chip Erase key is successfully decoded. The bit is cleared by the Reset Request issued as part of the Chip Erase sequence described in the 33.3.8.1 Chip Erase section.

33.5.6 ASI Reset Request

Name: ASI_RESET_REQ
Offset: 0x08
Reset: 0x00
Property: -

A Reset is signalized to the System when writing the Reset signature to this register.

Bit	7	6	5	4	3	2	1	0
	RSTREQ[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 – RSTREQ[7:0] Reset Request
The UPDI will not be reset when issuing a System Reset from this register.

Value	Name	Description
0x00	RUN	Clear Reset condition
0x59	RESET	Normal Reset
Other		Reset condition is cleared

33.5.7 ASI Control A

Name: ASI_CTRLA
Offset: 0x09
Reset: 0x03
Property: -

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

Bits 1:0 – UPDICLKSEL[1:0] UPDI Clock Select
Writing these bits selects the UPDI clock output frequency. The default setting after Reset and enable is 4 MHz. Any other clock output selection is only recommended when the BOD is at the highest level. For all other BOD settings, the default 4 MHz selection is recommended.

Value	Description
0x0	Reserved
0x1	16 MHz UPDI clock
0x2	8 MHz UPDI clock
0x3	4 MHz UPDI clock (Default Setting)

33.5.8 ASI System Control A

Name: ASI_SYS_CTRLA
Offset: 0x0A
Reset: 0x00
Property: -

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

Bit 1 – UROWWRITE_FINAL User Row Programming Done
This bit must be written when the user row data have been written to the RAM. Writing a '1' to this bit will start the process of programming the user row data to the Flash.
If this bit is written before the user row data is written to the RAM by the UPDI, the CPU will proceed without the written data.
This bit is writable only if the USERROW-Write key is successfully decoded.

Bit 0 – CLKREQ Request System Clock
If this bit is written to '1', the ASI is requesting the system clock, independent of the system Sleep modes. This makes it possible for the UPDI to access the ACC layer, also if the system is in Sleep mode.
Writing a '0' to this bit will lower the clock request.
This bit will be reset when the UPDI is disabled.
This bit is set by default when the UPDI is enabled in any mode (Fuse, high-voltage).

33.5.9 ASI System Status

Name: ASI_SYS_STATUS
Offset: 0x0B
Reset: 0x01
Property: -

Bit	7	6	5	4	3	2	1	0
			RSTSYS	INSLEEP	NVMPROG	UROWPROG		LOCKSTATUS
Access			R	R	R	R		R
Reset			0	0	0	0		1

Bit 5 – RSTSYS System Reset Active
When this bit is set to '1', there is an active Reset on the system domain. When this bit is set to '0', the system is not in the Reset state.
This bit is set to '0' on read.
A Reset held from the ASI_RESET_REQ register will also affect this bit.

Bit 4 – INSLEEP System Domain in Sleep
When this bit is set to '1', the system domain is in Idle or deeper Sleep mode. When this bit is set to '0', the system is not in any sleep mode.

Bit 3 – NVMPROG Start NVM Programming
When this bit is set to '1', NVM Programming can start from the UPDI.
When the UPDI is done, it must reset the system through the UPDI Reset register.

Bit 2 – UROWPROG Start User Row Programming
If this bit is set, User Row Programming can start from the UPDI.
When this bit is set to '1', User Row Programming can start from the UPDI.
When the User Row data have been written to the RAM, the UROWDONE bit in the ASI_SYS_CTRLA register must be written.

Bit 0 – LOCKSTATUS NVM Lock Status
When this bit is set to '1', the device is locked. If a chip erase is done, and the lock bits are set to '0', this bit will be read as '0'.

33.5.10 ASI CRC Status

Name: ASI_CRC_STATUS
Offset: 0x0C
Reset: 0x00
Property: -

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

Bits 2:0 – CRC_STATUS[2:0] CRC Execution Status
This bit field signalizes the status of the CRC conversion. This bit field is one-hot encoded.

Value	Description
0x0	Not enabled
0x1	CRC enabled, busy
0x2	CRC enabled, done with OK signature
0x4	CRC enabled, done with FAILED signature
Other	Reserved

34. Instruction Set Summary

The instruction set summary can be found as part of the *AVR Instruction Set Manual*, located at www.microchip.com/DS40002198. Refer to the CPU version called AVRxt, for details regarding the devices documented in this data sheet.

35. Conventions

35.1 Numerical Notation

Table 35-1. Numerical Notation

Symbol	Description
165	Decimal number
0b0101	Binary number
'0101'	Binary numbers are given without prefix if unambiguous
0x3B24	Hexadecimal number
X	Represents an unknown or do not care value
Z	Represents a high-impedance (floating) state for either a signal or a bus

35.2 Memory Size and Type

Table 35-2. Memory Size and Bit Rate

Symbol	Description
KB	kilobyte (2^{10}B = 1024B)
MB	megabyte (2^{20}B = 1024 KB)
GB	gigabyte (2^{30}B = 1024 MB)
b	bit (binary '0' or '1')
B	byte (8 bits)
1 kbit/s	1,000 bit/s rate
1 Mbit/s	1,000,000 bit/s rate
1 Gbit/s	1,000,000,000 bit/s rate
word	16-bit

35.3 Frequency and Time

Table 35-3. Frequency and Time

Symbol	Description
kHz	1 kHz = 10^3 Hz = 1,000 Hz
MHz	1 MHz = 10^6 Hz = 1,000,000 Hz
GHz	1 GHz = 10^9 Hz = 1,000,000,000 Hz
ms	1 ms = 10^{-3}s = 0.001s
µs	1 µs = 10^{-6}s = 0.000001s
ns	1 ns = 10^{-9}s = 0.000000001s

35.4 Registers and Bits

Table 35-4. Register and Bit Mnemonics

Symbol	Description
R/W	Read/Write accessible register bit. The user can read from and write to this bit.
R	Read-only accessible register bit. The user can only read this bit. Writes will be ignored.
W	Write-only accessible register bit. The user can only write this bit. Reading this bit will return an undefined value.
BITFIELD	Bitfield names are shown in uppercase. Example: INTMODE.
BITFIELD[n:m]	A set of bits from bit n down to m. Example: PINA[3:0] = {PINA3, PINA2, PINA1, PINA0}.
Reserved	Reserved bits, bit fields, and bit field values are unused and reserved for future use. For compatibility with future devices, always write reserved bits to '0' when the register is written. Reserved bits will always return zero when read.
PERIPHERALn	If several instances of the peripheral exist, the peripheral name is followed by a single number to identify one instance. Example: USARTn is the collection of all instances of the USART module, while USART3 is one specific instance of the USART module.
PERIPHERALx	If several instances of the peripheral exist, the peripheral name is followed by a single capital letter (A-Z) to identify one instance. Example: PORTx is the collection of all instances of the PORT module, while PORTB is one specific instance of the PORT module.
Reset	Value of a register after a Power-on Reset. This is also the value of registers in a peripheral after performing a software Reset of the peripheral, except for the Debug Control registers.
SET/CLR/TGL	Registers with SET/CLR/TGL suffix allow the user to clear and set bits in a register without doing a read-modify-write operation. Each SET/CLR/TGL register is paired with the register it is affecting. Both registers in a register pair return the same value when read. Example: In the PORT peripheral, the OUT and OUTSET registers form such a register pair. The contents of OUT will be modified by a write to OUTSET. Reading OUT and OUTSET will return the same value. Writing a '1' to a bit in the CLR register will clear the corresponding bit in both registers. Writing a '1' to a bit in the SET register will set the corresponding bit in both registers. Writing a '1' to a bit in the TGL register will toggle the corresponding bit in both registers.

35.4.1 Addressing Registers from Header Files

In order to address registers in the supplied C header files, the following rules apply:

1. A register is identified by <peripheral_instance_name>.<register_name>, e.g., CPU.SREG, USART2.CTRLA, or PORTB.DIR.
2. The peripheral name is given in the "Peripheral Address Map" in the "Peripherals and Architecture" section.
3. <peripheral_instance_name> is obtained by substituting any n or x in the peripheral name with the correct instance identifier.
4. When assigning a predefined value to a peripheral register, the value is constructed following the rule: <peripheral_name>_<bit_field_name>_<bit_field_value>_gc

   <peripheral_name> is <peripheral_instance_name>, but remove any instance identifier.

   <bit_field_value> can be found in the "Name" column in the tables in the Register Description sections describing the bit fields of the peripheral registers.

Example 35-1. Register Assignments

```
// EVSYS channel 0 is driven by TCB3 OVF event
EVSYS.CHANNEL0 = EVSYS_CHANNEL0_TCB3_OVF_gc;

// USART0 RXMODE uses Double Transmission Speed
USART0.CTRLB = USART_RXMODE_CLK2X_gc;
```

Note: For peripherals with different register sets in different modes, <peripheral_instance_name> and <peripheral_name> must be followed by a mode name, for example:

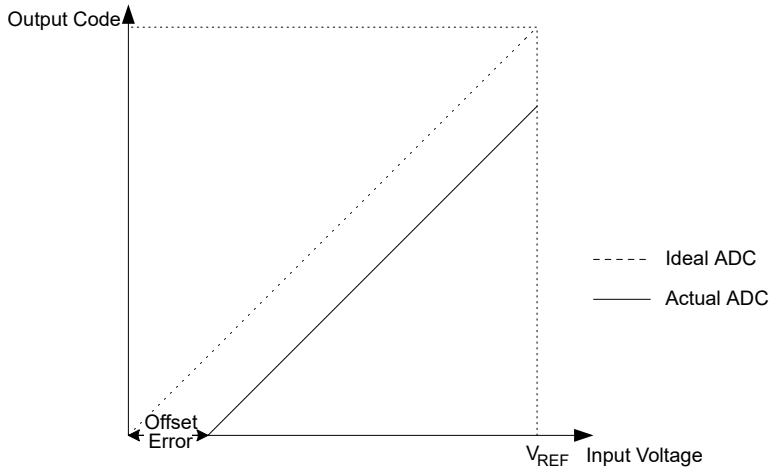
```
// TCA0 in Normal Mode (SINGLE) uses waveform generator in frequency mode
TCA0.SINGLE.CTRL=TCA_SINGLE_WGMODE_FRQ_gc;
```

35.5 ADC Parameter Definitions

An ideal n-bit single-ended ADC converts a voltage linearly between GND and V_{REF} in 2^n steps (LSb). The lowest code is read as '0', and the highest code is read as '2^n-1'. Several parameters describe the deviation from the ideal behavior:

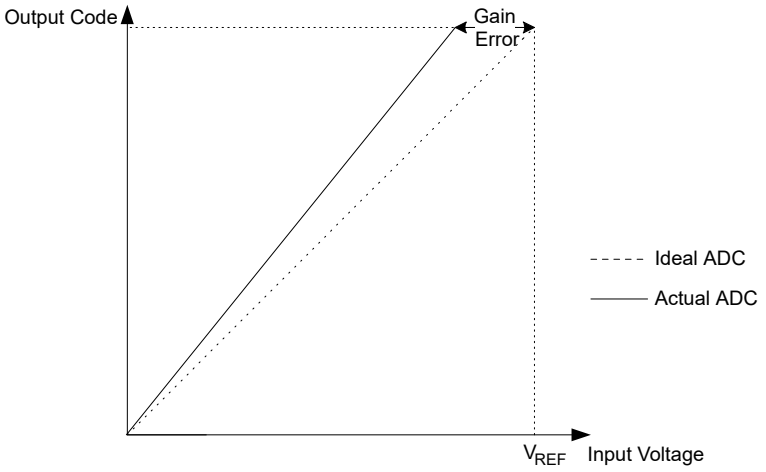
Offset Error The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSb). Ideal value: 0 LSb.

Figure 35-1. Offset Error



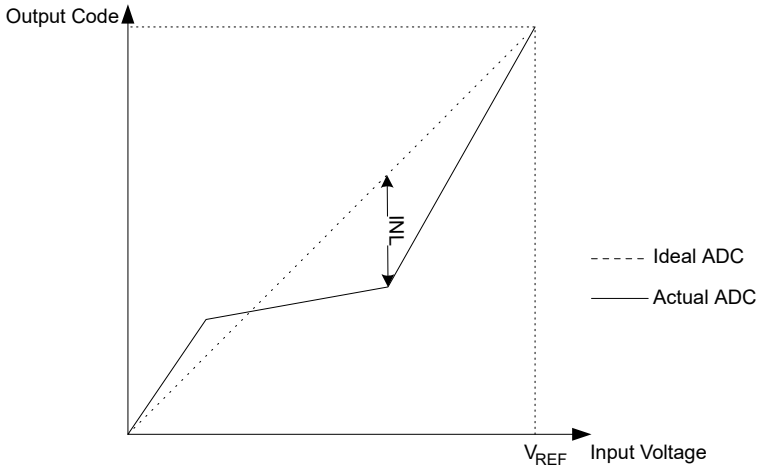
Gain Error After adjusting for offset, the gain error is found as the deviation of the last transition (e.g., 0x3FE to 0x3FF for a 10-bit ADC) compared to the ideal transition (at 1.5 LSb below maximum). Ideal value: 0 LSb.

Figure 35-2. Gain Error



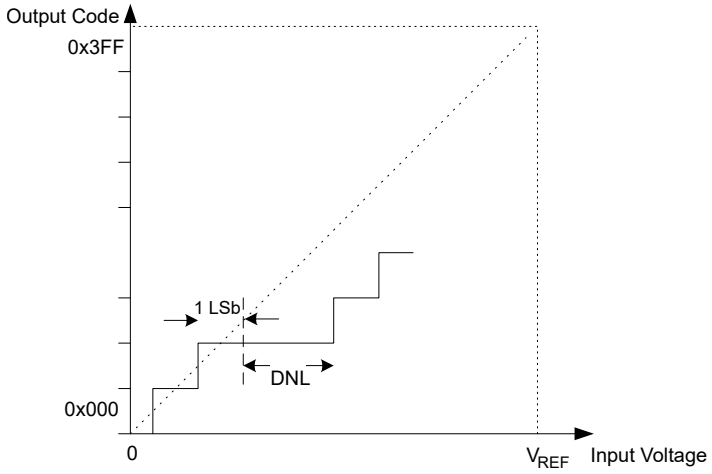
Integral Nonlinearity (INL) After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSb.

Figure 35-3. Integral Nonlinearity



Differential Nonlinearity (DNL) The maximum deviation of the actual code width (the interval between two adjacent transitions) from the ideal code width (1 LSb). Ideal value: 0 LSb.

Figure 35-4. Differential Nonlinearity



Quantization Error	Due to the quantization of the input voltage into a finite number of codes, a range of input voltages (1 LSb wide) will code to the same value. Always ±0.5 LSb.
Absolute Accuracy	The maximum deviation of an actual (unadjusted) transition compared to an ideal transition for any code. This is the compound effect of all errors mentioned before. Ideal value: ±0.5 LSb.

36. Electrical Characteristics

36.1 Disclaimer

All typical values are measured at T = 25°C and V_{DD} = 3V unless otherwise specified. All minimum and maximum values are valid across operating temperature and voltage unless otherwise specified.

36.2 Absolute Maximum Ratings

Stresses beyond those listed in this section may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 36-1. Absolute Maximum Ratings

Symbol	Description	Conditions	Min.	Max.	Unit
V_{DD}	Power supply voltage		-0.5	6	V
I_{VDD}	Current into a VDD pin	T = [-40, 85]°C	-	200	mA
		T = [85, 125]°C	-	100	mA
I_{GND}	Current out of a GND pin	T = [-40, 85]°C	-	200	mA
		T = [85, 125]°C	-	100	mA
V_{RST}	$\overline{\text{RESET}}$ pin voltage with respect to GND		-0.5	13	V
V_{PIN}	Pin voltage with respect to GND		-0.5	V_{DD}+0.5	V
I_{PIN}	I/O pin sink/source current		-40	40	mA
I_{c1}(1)	I/O pin injection current except $\overline{\text{RESET}}$ pin	V_{PIN} < GND – 0.6V or 5.5V < V_{PIN} ≤ 6.1V 4.9V < V_{DD} ≤ 5.5V	-1	1	mA
I_{c2}(1)	I/O pin injection current except $\overline{\text{RESET}}$ pin	V_{PIN} < GND – 0.6V or V_{PIN} ≤ 5.5V V_{DD} ≤ 4.9V	-15	15	mA
I_{ctot}	Sum of I/O pin injection current except $\overline{\text{RESET}}$ pin		-45	45	mA
$T_{storage}$	Storage temperature		-65	150	°C

Note:

1. – If the V_{PIN} is lower than GND – 0.6V, then a current limiting resistor is required. The negative DC injection current limiting resistor is calculated as R = (GND – 0.6V – V_{PIN})/I_{Cn}.
   – If the V_{PIN} is greater than V_{DD}+0.6V, then a current limiting resistor is required. The positive DC injection current limiting resistor is calculated as R = (V_{PIN} – (V_{DD} + 0.6))/I_{Cn}.

⚠ CAUTION

V_{RST}MAX = 13V
Care should be taken to avoid overshoot (overvoltage) when connecting the RESET pin to a 12V source. Exposing the pin to a voltage above the rated absolute maximum can activate the pin's ESD protection circuitry, which will remain activated until the voltage has been brought below approximately 10V. A 12V driver can keep the ESD protection in an activated state (if activated by an overvoltage condition) while driving currents through it, potentially causing permanent damage to the part.

36.3 General Operating Ratings

The device must operate within the ratings listed in this section for all other electrical characteristics and typical characteristics of the device to be valid.

Table 36-2. General Operating Conditions

Symbol	Description	Condition	Min.	Max.	Unit
V_{DD}	Operating supply voltage		1.8(2)	5.5	V
T	Operating temperature range(1)	Standard temperature range	-40	105	°C
		Extended temperature range(3)	-40	125	

Note:

1. Refer to the device ordering codes for the device temperature range.
2. Operation is ensured down to 1.8V or BOD triggering level, V_{BOD}. V_{BOD} may be below the minimum Operating Supply Voltage for some devices. Where this is the case, the device is tested down to $V_{DD} = V_{BOD}$ during production.
   - During Chip Erase, the BOD is forced ON. If the supply voltage V_{DD} is below the configured V_{BOD}, the Chip Erase will fail. See *Chip Erase*.
3. The extended temperature range is only ensured down to 2.7V.

Table 36-3. Operating Voltage and Frequency

Symbol	Description	Condition	Min.	Max.	Unit
CLK_CPU	Operating system clock frequency	V_{DD} = [1.8, 5.5]V T = [-40, 105]°C(1)	0	5	MHz
		V_{DD} = [2.7, 5.5]V T = [-40, 105]°C(2)	0	10	
		V_{DD} = [4.5, 5.5]V T = [-40, 105]°C(3)	0	20	
		V_{DD} = [2.7, 5.5]V T = [-40, 125]°C(2)	0	8	
		V_{DD} = [4.5, 5.5]V T = [-40, 125]°C(3)	0	16	

Note:

1. Operation ensured down to BOD triggering level, V_{BOD} with BODLEVEL0.
2. Operation ensured down to BOD triggering level, V_{BOD} with BODLEVEL2.
3. Operation ensured down to BOD triggering level, V_{BOD} with BODLEVEL7.

The maximum CPU clock frequency depends on V_{DD}. As shown in the following figure, the maximum frequency vs. V_{DD} is linear between $1.8V < V_{DD} < 2.7V$ and $2.7V < V_{DD} < 4.5V$.

Figure 36-1. Maximum Frequency vs. V_{DD} for [-40, 105]°C, Standard Temperature Range

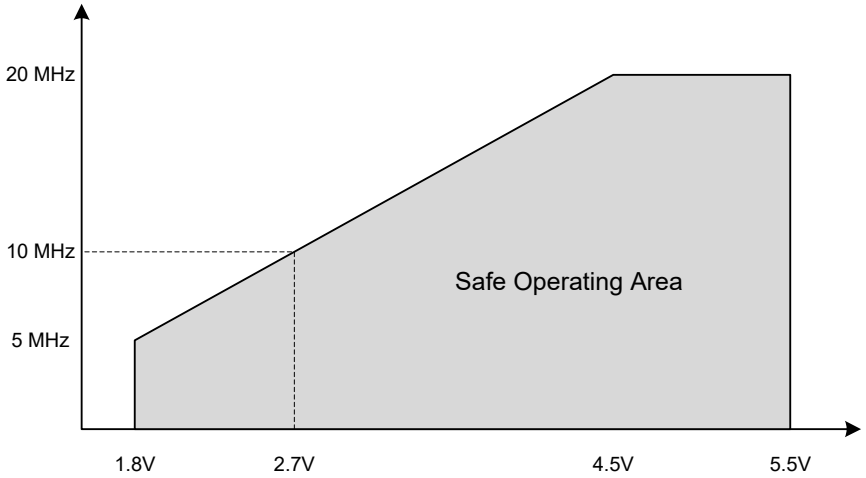
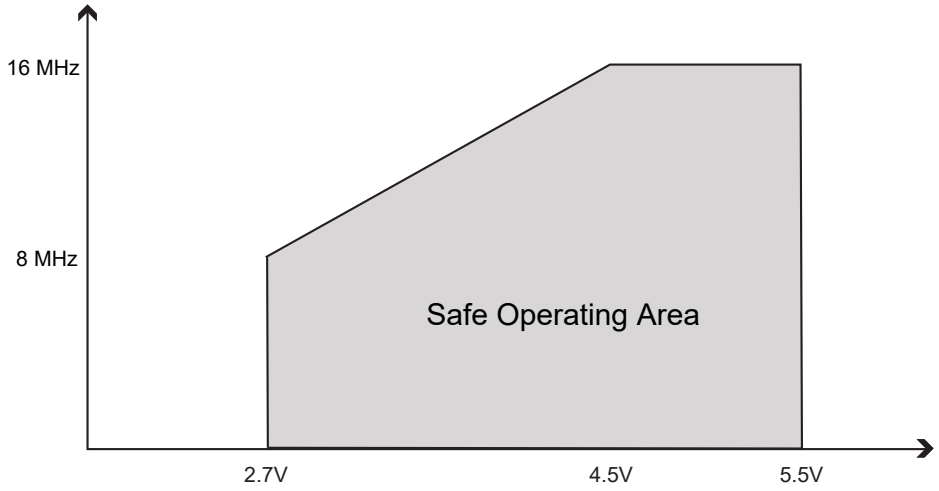


Figure 36-2. Maximum Frequency vs. V_{DD} for [-40, 125]°C, Extended Temperature Range



36.4 Power Consumption

The values are measured power consumption under the following conditions, except where noted:

- V_{DD} = 3V
- T = 25°C
- OSC20M used as the system clock source, except where otherwise specified
- System power consumption measured with peripherals disabled and without I/O drive

Table 36-4. Power Consumption in Active and Idle Mode

Mode	Description	Condition		Typ.	Max.	Unit
Active	Active power consumption	CLK_CPU = 20 MHz (OSC20M)	V_{DD} = 5V	10.0	-	mA
		CLK_CPU = 10 MHz (OSC20M div2)	V_{DD} = 5V	5.3	-	mA
			V_{DD} = 3V	3.0	-	mA
		CLK_CPU = 5 MHz (OSC20M div4)	V_{DD} = 5V	3.0	-	mA
			V_{DD} = 3V	1.7	-	mA
			V_{DD} = 2V	1.1	-	mA
		CLK_CPU = 32.768 kHz (OSCULP32K)	V_{DD} = 5V	19.2	-	µA
			V_{DD} = 3V	10.9	-	µA
			V_{DD} = 2V	7.4	-	µA
Idle	Idle power consumption	CLK_CPU = 20 MHz (OSC20M)	V_{DD} = 5V	2.9	6.3(1)	mA
		CLK_CPU = 10 MHz (OSC20M div2)	V_{DD} = 5V	1.5	3.1(1)	mA
			V_{DD} = 3V	0.85	1.9(1)	mA
		CLK_CPU = 5 MHz (OSC20M div4)	V_{DD} = 5V	0.75	1.6(1)	mA
			V_{DD} = 3V	0.4	1.2	mA
			V_{DD} = 2V	0.3	-	mA
		CLK_CPU = 32.768 kHz (OSCULP32K)	V_{DD} = 5V	5.7	20(1)	µA
			V_{DD} = 3V	2.9	15(1)	µA
			V_{DD} = 2V	1.9	-	µA

Note:

1. These values are based on characterization and not covered by production test limits.

Table 36-5. Power Consumption in Power-Down, Standby, and Reset Mode

Mode	Description	Condition		Typ. 25°C	Max. 25°C	Max. 85°C(1)	Max. 125°C	Unit
Standby	Standby power consumption	RTC running at 1.024 kHz from external XOSC32K (CL=7.5 pF)	V_{DD} = 3V	0.69	-	-	-	µA
		RTC running at 1.024 kHz from internal OSCULP32K	V_{DD} = 3V	0.71	3.0	6.0	8.0	µA
Power Down/ Standby	Power-down/ Standby power consumption are the same when all peripherals are stopped	All peripherals stopped	V_{DD} = 3V	0.1	2.0	5.0	7.0	µA

.....continued								
Mode	**Description**	**Condition**		**Typ. 25°C**	**Max. 25°C**	**Max. 85°C(1)**	**Max. 125°C**	**Unit**
Reset	Reset power consumption	Reset line pulled down	V_{DD} = 3V	100	-	-	-	µA

Note:

1. These values are based on characterization and not covered by production test limits.

36.5 Wake-Up Time

Wake-up time from sleep mode is measured from the edge of the wake-up signal to the first instruction executed.

Operating conditions:

- V_{DD} = 3V
- T = 25°C
- OSC20M as the system clock source, unless otherwise specified

Table 36-6. Start-Up, Reset, and Wake-Up Time from OSC20M

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
t_{wakeup}	Start-up time from any Reset release		-	200	-	µs
	Wake-up from Idle sleep mode	OSC20M @ 20 MHz V_{DD} = 5V	-	1	-	
		OSC20M @ 10 MHz V_{DD} = 3V	-	2	-	
		OSC20M @ 5 MHz V_{DD} = 2V	-	4	-	
	Wake-up from Standby and Power-Down sleep mode		-	10	-	

36.6 Peripherals Power Consumption

The table below can be used to calculate the additional current consumption for the different I/O peripherals in the various operating modes.

Operating conditions:

- V_{DD} = 3V
- T = 25°C
- OSC20M at 1 MHz used as the system clock source, except where otherwise specified
- In Idle sleep mode, except where otherwise specified

Table 36-7. Peripherals Power Consumption

Peripheral	Conditions	Typ.(1)	Unit
BOD	Continuous	19	µA
	Sampling @ 1 kHz	1	
TCA	16-bit count @ 1 MHz	13	µA
TCB	16-bit count @ 1 MHz	7.5	µA
RTC	16-bit count @ 32.768 kHz @ OSCULP32K	1	µA

.....continued			
Peripheral	**Conditions**	**Typ.(1)**	**Unit**
WDT (including OSCULP32K)		1	µA
OSC20M		125	µA
AC	Fast mode(2)	92	µA
	Low-Power mode(2)	45	µA
ADC	50 ksps	325	µA
	100 ksps	340	µA
XOSC32K	C_L = 7.5 pF	0.5	µA
OSCULP32K		0.5	µA
USART	Enable @ 9600 Baud	13	µA
SPI (Master)	Enable @ 100 kHz	2	µA
TWI (Master)	Enable @ 100 kHz	24	µA
TWI (Slave)	Enable @ 100 kHz	17	µA
Flash programming	Erase Operation	1.5	mA
	Write Operation	3.0	

Note:

1. The Current consumption of the module only. To calculate the total power consumption of the system, add this value to the base power consumption as listed in *Power Consumption*.
2. The CPU in Standby sleep mode.

36.7 BOD and POR Characteristics

Table 36-8. Power Supply Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
SRON	Power-on Slope		-	-	100	V/ms

Table 36-9. Power-on Reset (POR) Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{POR}	POR threshold voltage on V_{DD} falling	V_{DD} falls/rises at 0.5 V/ms or slower	0.8	-	1.6	V
	POR threshold voltage on V_{DD} rising		1.4	-	1.8	

Table 36-10. Brown-out Detection (BOD) Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{BOD}	BOD triggering level (falling/rising)	BODLEVEL7	3.9	4.2	4.5	V
		BODLEVEL2	2.4	2.6	2.9	
		BODLEVEL0	1.7	1.8	2.0	

.....continued						
Symbol	**Description**	**Condition**	**Min.**	**Typ.**	**Max.**	**Unit**
V_{VLM}	VLM threshold relative to BOD triggering level	BOD.VLMLVL = 0x0	-	4	-	%
		BOD.VLMLVL = 0x1	-	13	-	
		BOD.VLMLVL = 0x2	-	25	-	
V_{HYS}	Hysteresis	BODLEVEL7	-	80	-	mV
		BODLEVEL2	-	40	-	
		BODLEVEL0	-	25	-	
T_{BOD}	Detection time	Continuous	-	7	-	µs
		Sampled, 1 kHz	-	1	-	ms
		Sampled, 125 Hz	-	8	-	
T_{Start}	Start-up time	Time from enable to ready	-	40	-	µs

36.8 External Reset Characteristics

Table 36-11. External Reset Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{HVRST}	Ensured detection for a high-voltage Reset		11.5	-	12.5	V
V_{RST_VIH}	Input high-voltage for $\overline{\text{RESET}}$		0.8 × V_{DD}	-	V_{DD} + 0.2	
V_{RST_VIL}	Input low-voltage for $\overline{\text{RESET}}$		-0.2	-	0.2 × V_{DD}	
t_{RST}	Minimum pulse-width on the $\overline{\text{RESET}}$ pin		-	-	2.5	µs
R_{RST}	$\overline{\text{RESET}}$ pull-up resistor	V_{Reset} = 0V	20	-	60	kΩ

36.9 Oscillators and Clocks

Operating conditions:

- V_{DD} = 3V, except where specified otherwise

Table 36-12. Internal Oscillator (OSC20M) Characteristics

Symbol	Description	Condition		Min.	Typ.	Max.	Unit
f_{OSC20M}	Accuracy with 16 MHz and 20 MHz frequency selection relative to the factory-stored frequency value	Factory calibrated V_{DD} = 3V(1)	T = [0, 70]°C, V_{DD} = [1.8, 4.5]V(3)	-2.0	-	2.0	%
		Factory calibrated V_{DD} = 5V(1)	T = [0, 70]°C, V_{DD} = [4.5, 5.5]V(3)	-2.0	-	2.0	
	Accuracy with 16 MHz and 20 MHz frequency selection	Factory calibrated	T = 25°C, 3.0V	-3.0	-	3.0	%
			T = [0, 70]°C, V_{DD} = [1.8, 3.6]V(3)	-4.0	-	4.0	
			Full operation range(3)	-5.0	-	5.0	
f_{CAL}	User calibration range	OSC20M(2) = 16 MHz		14.5	-	17.5	MHz
		OSC20M(2) = 20 MHz		18.5	-	21.5	

.....continued

Symbol	Description	Condition		Min.	Typ.	Max.	Unit
%CAL	Calibration step size			-	1.5	-	%
DC	Duty cycle			-	50	-	%
T_{Start}	Start-up time	Within 2% accuracy		-	8	-	µs

Note:

1. See the description of OSC20M on calibration.
2. Oscillator frequencies above speed specification must be divided so that the CPU clock always is within specification.
3. These values are based on characterization and not covered by production test limits.

Table 36-13. 32.768 kHz Internal Oscillator (OSCULP32K) Characteristics

Symbol	Description	Condition		Min.	Typ.	Max.	Unit
$f_{OSCULP32K}$	Accuracy	Factory calibrated	T = 25°C, 3.0V	-3	-	3	%
			T = [0, 70]°C, V_{DD} = [1.8, 3.6]V(1)	-10	-	10	
			Full operation range(1)	-30	-	30	
DC	Duty cycle			-	50	-	%
T_{Start}	Start-up time			-	250	-	µs

Note:

1. These values are based on characterization and not covered by production test limits.

Table 36-14. 32.768 kHz External Crystal Oscillator (XOSC32K) Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
F_{out}	Frequency		-	32.768	-	kHz
T_{Start}	Start-up time	C_L = 7.5 pF	-	300	-	ms
C_L	Crystal load capacitance		7.5	-	12.5	pF
C_{TOSC1}	Parasitic capacitor load		-	5.5	-	pF
C_{TOSC2}			-	5.5	-	pF
ESR	Equivalent Series Resistance - Safety Factor = 3	C_L = 7.5 pF	-	-	80	kΩ
		C_L = 12.5 pF	-	-	40	

Figure 36-3. External Clock Waveform Characteristics

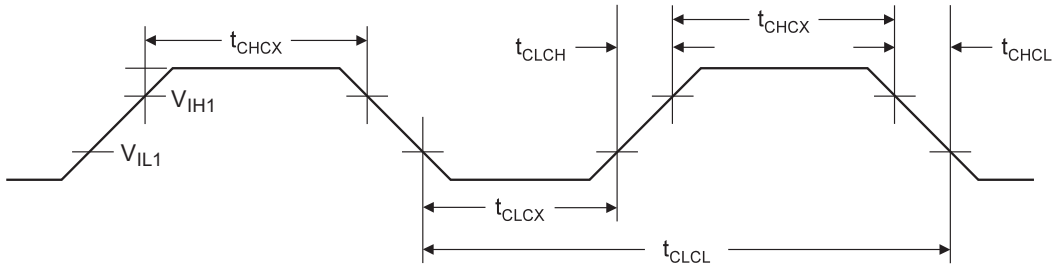


Table 36-15. External Clock Characteristics

Symbol	Description	Condition	V_{DD} = [1.8, 5.5]V		V_{DD} = [2.7, 5.5]V		V_{DD} = [4.5, 5.5]V		Unit
			Min.	Max.	Min.	Max.	Min.	Max.	
f_{CLCL}	Frequency		0	5.0	0.0	10.0	0.0	20.0	MHz
t_{CLCL}	Clock period		200	-	100	-	50	-	ns
t_{CHCX}	High time		80	-	40	-	20	-	ns
t_{CLCX}	Low time		80	-	40	-	20	-	ns

36.10 I/O Pin Characteristics

Table 36-16. I/O Pin Characteristics (T_A = [-40, 105]°C, V_{DD} = [1.8, 5.5]V Unless Otherwise Stated)

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{IL}	Input low-voltage, except $\overline{\text{RESET}}$ pin as I/O		-0.2	-	0.3 × V_{DD}	V
V_{IH}	Input high-voltage, except $\overline{\text{RESET}}$ pin as I/O		0.7 × V_{DD}	-	V_{DD} + 0.2V	V
I_{IH} / I_{IL}	I/O pin input leakage current, except $\overline{\text{RESET}}$ pin as I/O	V_{DD} = 5.5V, pin high	-	< 0.05	-	µA
		V_{DD} = 5.5V, pin low	-	< 0.05	-	
V_{OL}	I/O pin drive strength	V_{DD} = 1.8V, I_{OL} = 1.5 mA	-	-	0.36	V
		V_{DD} = 3.0V, I_{OL} = 7.5 mA	-	-	0.6	
		V_{DD} = 5.0V, I_{OL} = 15 mA	-	-	1	
V_{OH}	I/O pin drive strength	V_{DD} = 1.8V, I_{OH} = 1.5 mA	1.44	-	-	V
		V_{DD} = 3.0V, I_{OH} = 7.5 mA	2.4	-	-	
		V_{DD} = 5.0V, I_{OH} = 15 mA	4	-	-	
I_{total}	Maximum combined I/O sink current per pin group(1)		-	-	100	mA
	Maximum combined I/O source current per pin group(1)		-	-	100	
V_{IL2}	Input low-voltage on $\overline{\text{RESET}}$ pin as I/O		-0.2	-	0.3 × V_{DD}	V
V_{IH2}	Input high-voltage on $\overline{\text{RESET}}$ pin as I/O		0.7 × V_{DD}	-	V_{DD} + 0.2V	V
V_{OL2}	I/O pin drive strength on $\overline{\text{RESET}}$ pin as I/O	V_{DD} = 1.8V, I_{OL} = 0.1 mA	-	-	0.36	V
		V_{DD} = 3.0V, I_{OL} = 0.25 mA	-	-	0.6	
		V_{DD} = 5.0V, I_{OL} = 0.5 mA	-	-	1	
V_{OH2}	I/O pin drive strength on $\overline{\text{RESET}}$ pin as I/O	V_{DD} = 1.8V, I_{OH} = 0.1 mA	1.44	-	-	V
		V_{DD} = 3.0V, I_{OH} = 0.25 mA	2.4	-	-	
		V_{DD} = 5.0V, I_{OH} = 0.5 mA	4	-	-	
t_{RISE}	Rise time	V_{DD} = 3.0V, load = 20 pF	-	2.5	-	ns
		V_{DD} = 5.0V, load = 20 pF	-	1.5	-	
t_{FALL}	Fall time	V_{DD} = 3.0V, load = 20 pF	-	2.0	-	ns
		V_{DD} = 5.0V, load = 20 pF	-	1.3	-	
C_{PIN}	I/O pin capacitance except TOSC and TWI pins		-	3	-	pF
C_{PIN}	I/O pin capacitance on TOSC pins		-	5.5	-	pF

.....continued						
Symbol	**Description**	**Condition**	**Min.**	**Typ.**	**Max.**	**Unit**
C_{PIN}	I/O pin capacitance on TWI pins		-	10	-	pF
R_P	Pull-up resistor		20	35	50	kΩ

Note:

1. Pin group x (Px[7:0]). The combined continuous sink/source current for all I/O ports should not exceed the limits.

36.11 TCD

Operating conditions:

- CLK_TCD frequencies above maximum CLK_TCD_SYNC must be prescaled with the Synchronization Prescaler (SYNCPRES in TCDn.CTRLA), so the synchronizer clock meets these specifications

Table 36-17. Timer/Counter D Maximum Frequency(1)

Symbol	Description	Condition		Max.	Unit
$f_{CLK\_TCD\_SYNC}$	CLK_TCD_SYNC maximum frequencies	V_{DD} = [1.8, 5.5]V	T_A = [-40, 125]°C	8	MHz
		V_{DD} = [2.7, 5.5]V	T_A = [-40, 125]°C	16	
		V_{DD} = [4.5, 5.5]V	T_A = [-40, 105]°C	32	
			T_A = [-40, 125]°C	20	

Note:

1. These parameters are for design guidance only and are not covered by production test limits.

36.12 USART

Figure 36-4. USART in SPI Mode - Timing Requirements in Master Mode

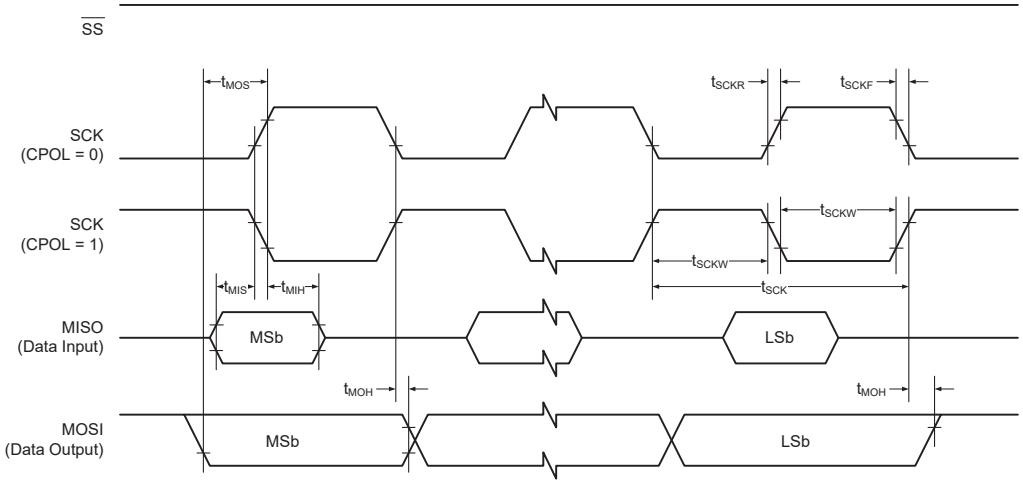


Table 36-18. USART in SPI Master Mode - Timing Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
f_{SCK}	SCK clock frequency	Master	-	-	10	MHz
t_{SCK}	SCK period	Master	100	-	-	ns

.....continued						
Symbol	**Description**	**Condition**	**Min.**	**Typ.**	**Max.**	**Unit**
t_{SCKW}	SCK high/low width	Master	-	0.5 × t_{SCK}	-	ns
t_{SCKR}	SCK rise time	Master	-	2.7	-	ns
t_{SCKF}	SCK fall time	Master	-	2.7	-	ns
t_{MIS}	MISO setup to SCK	Master	-	10	-	ns
t_{MIH}	MISO hold after SCK	Master	-	10	-	ns
t_{MOS}	MOSI setup to SCK	Master	-	0.5 × t_{SCK}	-	ns
t_{MOH}	MOSI hold after SCK	Master	-	1.0	-	ns

36.13 SPI

Figure 36-5. SPI - Timing Requirements in Master Mode

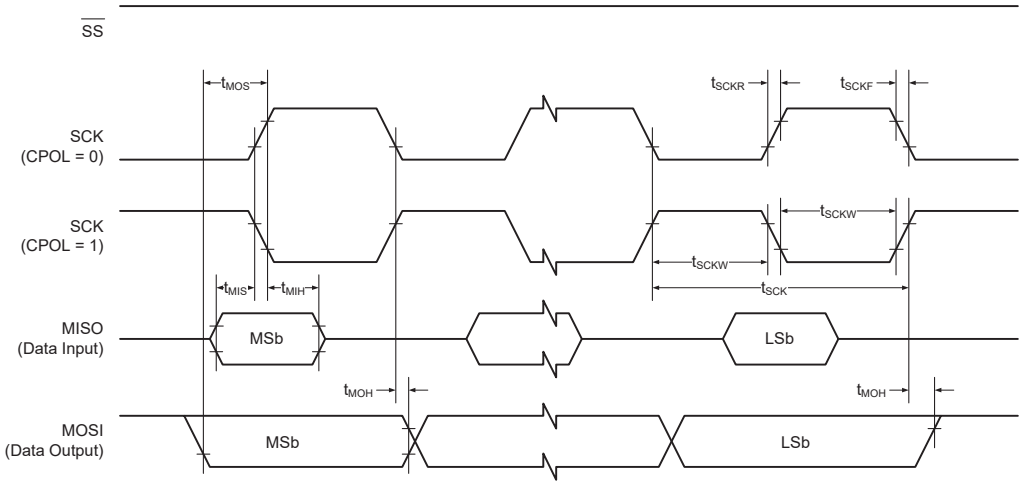


Figure 36-6. SPI - Timing Requirements in Slave Mode

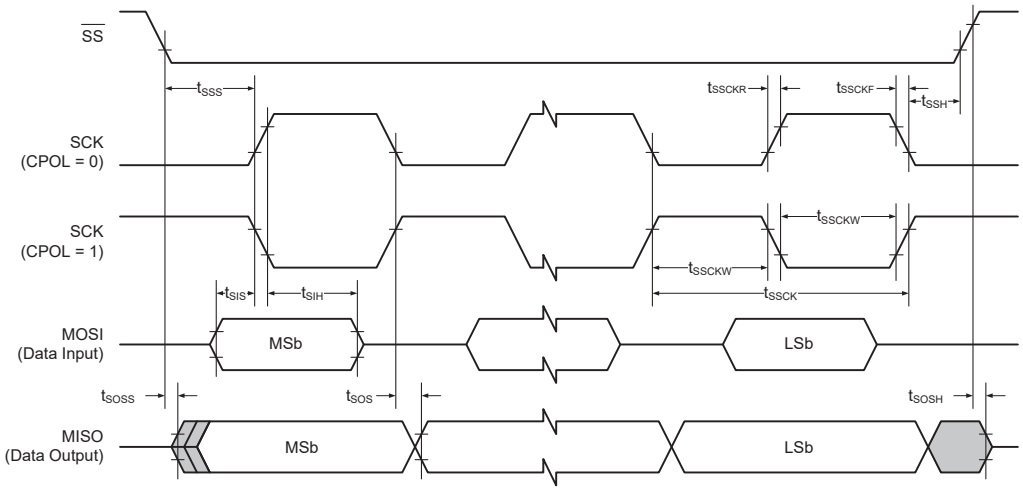


Table 36-19. SPI - Timing Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
f_{SCK}	SCK clock frequency	Master	-	-	10	MHz
t_{SCK}	SCK period	Master	100	-	-	ns
t_{SCKW}	SCK high/low width	Master	-	0.5 × t_{SCK}	-	ns
t_{SCKR}	SCK rise time	Master	-	2.7	-	ns
t_{SCKF}	SCK fall time	Master	-	2.7	-	ns
t_{MIS}	MISO setup to SCK	Master	-	10	-	ns
t_{MIH}	MISO hold after SCK	Master	-	10	-	ns
t_{MOS}	MOSI setup to SCK	Master	-	0.5 × t_{SCK}	-	ns
t_{MOH}	MOSI hold after SCK	Master	-	1.0	-	ns
f_{SSCK}	Slave SCK clock frequency	Slave	-	-	5	MHz
t_{SSCK}	Slave SCK Period	Slave	4 × t_{CLK_PER}	-	-	ns
t_{SSCKW}	SCK high/low width	Slave	2 × t_{CLK_PER}	-	-	ns
t_{SSCKR}	SCK rise time	Slave	-	-	1600	ns
t_{SSCKF}	SCK fall time	Slave	-	-	1600	ns
t_{SIS}	MOSI setup to SCK	Slave	3.0	-	-	ns
t_{SIH}	MOSI hold after SCK	Slave	t_{CLK_PER}	-	-	ns
t_{SSS}	SS setup to SCK	Slave	21	-	-	ns
t_{SSH}	SS hold after SCK	Slave	20	-	-	ns
t_{SOS}	MISO setup to SCK	Slave	-	8.0	-	ns
t_{SOH}	MISO hold after SCK	Slave	-	13	-	ns
t_{SOSS}	MISO setup after SS low	Slave	-	11	-	ns
t_{SOSH}	MISO hold after SS low	Slave	-	8.0	-	ns

36.14 TWI

Figure 36-7. TWI - Timing Requirements

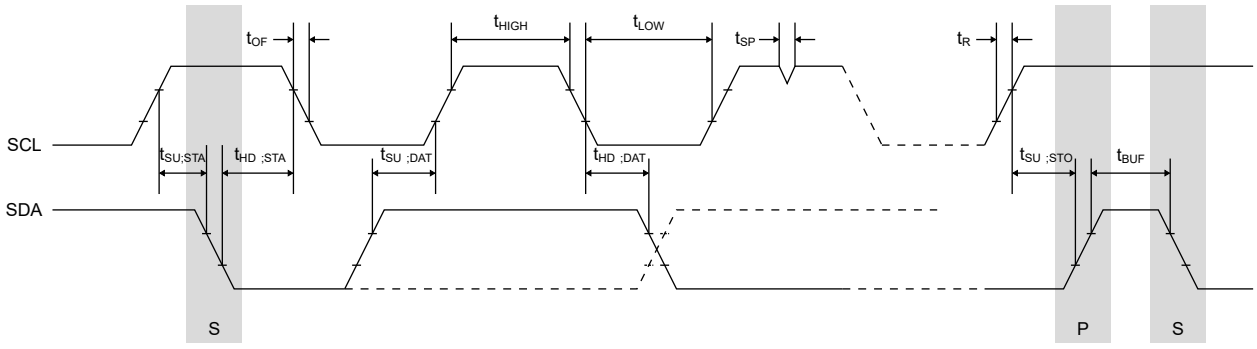


Table 36-20. TWI - Timing Characteristics

Symbol	Description	Condition		Min.	Typ.	Max.	Unit
f_{SCL}	SCL clock frequency	Max. frequency requires the system clock running at 10 MHz, which, in turn, requires V_{DD} = [2.7, 5.5]V and T = [-40, 105]°C		0	-	1000	kHz
V_{IH}	Input high voltage			0.7 × V_{DD}	-	-	V
V_{IL}	Input low voltage			-	-	0.3 × V_{DD}	V
V_{HYS}	Hysteresis of Schmitt trigger inputs			0.1 × V_{DD}		0.4 × V_{DD}	V
V_{OL}	Output low voltage	I_{load} = 20 mA, Fast mode+		-	-	0.2 × V_{DD}	V
		I_{load} = 3 mA, Normal mode, V_{DD} > 2V		-	-	0.4	
		I_{load} = 3 mA, Normal mode, V_{DD} ≤ 2V		-	-	0.2 × V_{DD}	
I_{OL}	Low-level output current	f_{SCL} ≤ 400 kHz, V_{OL} = 0.4V		3	-	-	mA
		f_{SCL} ≤ 1 MHz, V_{OL} = 0.4V		20	-	-	
C_B	Capacitive load for each bus line	f_{SCL} ≤ 100 kHz		-	-	400	pF
		f_{SCL} ≤ 400 kHz		-	-	400	
		f_{SCL} ≤ 1 MHz		-	-	550	
t_R	Rise time for both SDA and SCL	f_{SCL} ≤ 100 kHz		-	-	1000	ns
		f_{SCL} ≤ 400 kHz		20	-	300	
		f_{SCL} ≤ 1 MHz		-	-	120	
t_{OF}	Output fall time from V_{IHmin} to V_{ILmax}	10 pF < Capacitance of bus line < 400 pF	f_{SCL} ≤ 400 kHz	20 × (V_{DD}/ 5.5V)	-	250	ns
			f_{SCL} ≤ 1 MHz	20 × (V_{DD}/ 5.5V)	-	120	
t_{SP}	Spikes suppressed by Input filter			0	-	50	ns
I_L	Input current for each I/O pin	0.1×V_{DD} < V_I < 0.9×V_{DD}		-	-	1	µA
C_I	Capacitance for each I/O pin			-	-	10	pF
R_P	Value of pull-up resistor	f_{SCL} ≤ 100 kHz		(V_{DD} - V_{OL}(max)) /I_{OL}	-	1000 ns/ (0.8473 × C_B)	Ω
		f_{SCL} ≤ 400 kHz		-	-	300 ns/ (0.8473 × C_B)	
		f_{SCL} ≤ 1 MHz		-	-	120 ns/ (0.8473 × C_B)	

.....continued

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
$t_{HD;STA}$	Hold time (repeated) Start condition	f_{SCL} ≤ 100 kHz	4.0	-	-	µs
		f_{SCL} ≤ 400 kHz	0.6	-	-	
		f_{SCL} ≤ 1 MHz	0.26	-	-	
t_{LOW}	Low period of SCL Clock	f_{SCL} ≤ 100 kHz	4.7	-	-	µs
		f_{SCL} ≤ 400 kHz	1.3	-	-	
		f_{SCL} ≤ 1 MHz	0.5	-	-	
t_{HIGH}	High period of SCL Clock	f_{SCL} ≤ 100 kHz	4.0	-	-	µs
		f_{SCL} ≤ 400 kHz	0.6	-	-	
		f_{SCL} ≤ 1 MHz	0.26	-	-	
$t_{SU;STA}$	Setup time for a repeated Start condition	f_{SCL} ≤ 100 kHz	4.7	-	-	µs
		f_{SCL} ≤ 400 kHz	0.6	-	-	
		f_{SCL} ≤ 1 MHz	0.26	-	-	
$t_{HD;DAT}$	Data hold time	f_{SCL} ≤ 100 kHz	0	-	3.45	µs
		f_{SCL} ≤ 400 kHz	0	-	0.9	
		f_{SCL} ≤ 1 MHz	0	-	0.45	
$t_{SU;DAT}$	Data setup time	f_{SCL} ≤ 100 kHz	250	-	-	ns
		f_{SCL} ≤ 400 kHz	100	-	-	
		f_{SCL} ≤ 1 MHz	50	-	-	
$t_{SU;STO}$	Setup time for Stop condition	f_{SCL} ≤ 100 kHz	4	-	-	µs
		f_{SCL} ≤ 400 kHz	0.6	-	-	
		f_{SCL} ≤ 1 MHz	0.26	-	-	
t_{BUF}	Bus free time between a Stop and Start condition	f_{SCL} ≤ 100 kHz	4.7	-	-	µs
		f_{SCL} ≤ 400 kHz	1.3	-	-	
		f_{SCL} ≤ 1 MHz	0.5	-	-	

Table 36-21. SDA Hold Time(1,2)

Symbol	Description	Condition			Min.	Typ.	Max.	Unit
$t_{HD;DAT}$	Data hold time	Master(3)	f_{CLK_PER} = 5 MHz	SDAHOLD = 0x00	-	800	-	ns
				SDAHOLD = 0x01	830	850	950	
				SDAHOLD = 0x02	830	850	950	
				SDAHOLD = 0x03	830	850	1270	
			f_{CLK_PER} = 10 MHz	SDAHOLD = 0x00	-	400	-	
				SDAHOLD = 0x01	430	450	550	
				SDAHOLD = 0x02	430	450	580	
				SDAHOLD = 0x03	430	550	1270	
			f_{CLK_PER} = 20 MHz	SDAHOLD = 0x00	-	200	220	
				SDAHOLD = 0x01	230	250	350	
				SDAHOLD = 0x02	260	450	580	
				SDAHOLD = 0x03	380	600	1270	
$t_{HD;DAT}$	Data hold time	Slave(4)	All Frequencies	SDAHOLD = 0x00	90	150	220	ns
				SDAHOLD = 0x01	130	200	350	
				SDAHOLD = 0x02	260	400	580	
				SDAHOLD = 0x03	390	550	1270	

Note:

1. These parameters are for design guidance only and are not covered by production test limits.
2. SDAHOLD is the data hold time after the SCL signal is detected as low. The actual hold time is, therefore, higher than the configured hold time.
3. For Master mode, the data hold time is whatever is largest of the following:
   - $4 \times t_{CLK\_PER}$ + 50 ns (typical)
   - SDAHOLD configuration + SCL filter delay
4. For Slave mode, the hold time is given by:
   - SDAHOLD configuration + SCL filter delay

36.15 VREF

Table 36-22. Internal Voltage Reference Characteristics

Symbol	Description	Min.	Typ.	Max.	Unit
T_{Start}	Start-up time	-	25	-	µs
$V_{DDINT055V}$	Power supply voltage range for INT055V	1.8	-	5.5	V
$V_{DDINT11V}$	Power supply voltage range for INT11V	1.8	-	5.5	
$V_{DDINT15V}$	Power supply voltage range for INT15V	1.9	-	5.5	
$V_{DDINT25V}$	Power supply voltage range for INT25V	2.9	-	5.5	
$V_{DDINT43V}$	Power supply voltage range for INT43V	4.75	-	5.5	

Table 36-23. ADC Internal Voltage Reference Characteristics(1)

Symbol(2)	Description	Condition	Min.	Typ.	Max.	Unit
INT11V	Internal reference voltage	V_{DD} = [1.8V, 3.6V] T = [0, 105]°C	-2.0		2.0	%
INT055V INT15V INT25V	Internal reference voltage	V_{DD} = [1.8V, 3.6V] T = [0, 105]°C	-3.0		3.0	
INT055V INT11V INT15V INT25V INT43V	Internal reference voltage	V_{DD} = [1.8V, 5.5V] T = [-40, 125]°C	-5.0		5.0	

Note:

1. These values are based on characterization and not covered by production test limits.
2. The symbols INTxxV refer to the respective values of the ADC0REFSEL and DAC0REFSEL bit fields in the VREF.CTRLA register.

Table 36-24. DAC and AC Internal Voltage Reference Characteristics(1)

Symbol(2)	Description	Condition	Min.	Typ.	Max.	Unit
INT055V INT11V INT15V INT25V	Internal reference voltage	V_{DD} = [1.8V, 3.6V] T = [0, 105]°C	-3.0		3.0	%
INT055V INT11V INT15V INT25V INT43V	Internal reference voltage	V_{DD} = [1.8V, 5.5V] T = [-40, 125]°C	-5.0		5.0	

Note:

1. These values are based on characterization and not covered by production test limits.
2. The symbols INTxxV refer to the respective values of the ADC0REFSEL and DAC0REFSEL bit fields in the VREF.CTRLA register.

36.16 ADC

Operating conditions:

- V_{DD} = 1.8V to 5.5V
- Temperature = -40°C to 125°C
- DUTYCYC = 25%
- $CLK_{ADC} = 13 \times f_{ADC}$
- SAMPCAP is 10 pF for 0.55V reference, while it is set to 5 pF for $V_{REF} \geq 1.1V$
- Applies for all allowed combinations of V_{REF} selections and Sample Rates unless otherwise stated

Table 36-25. Power Supply, Reference, and Input Range

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
V_{DD}	Supply voltage		1.8	-	5.5	V

.....continued

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
V_{REF}	Reference voltage	REFSEL = Internal reference	0.55	-	V_{DD} - 0.4	V
		REFSEL = V_{DD}	1.8	-	5.5	
C_{IN}	Input capacitance	SAMPCAP = 5 pF	-	5	-	pF
		SAMPCAP = 10 pF	-	10	-	
R_{IN}	Input resistance		-	14	-	kΩ
V_{IN}	Input voltage range		0	-	V_{REF}	V
I_{BAND}	Input bandwidth	1.1V ≤ V_{REF}	-	-	57.5	kHz

Table 36-26. Clock and Timing Characteristics

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
f_{ADC}	Sample rate	1.1V ≤ V_{REF}	15	-	115	ksps
		1.1V ≤ V_{REF} (8-bit resolution)	15	-	150	
		V_{REF} = 0.55V (10 bits)	7.5	-	20	
CLK_{ADC}	Clock frequency	V_{REF} = 0.55V (10 bits)	100	-	260	kHz
		1.1V ≤ V_{REF} (10 bits)	200	-	1500	
		1.1V ≤ V_{REF} (8-bit resolution)	200	-	2000(1)	
Ts	Sampling time		2	2	33	CLK_{ADC} cycles
T_{CONV}	Conversion time (latency)	Sampling time = 2 CLK_{ADC}	8.7	-	50	µs
T_{START}	Start-up time	Internal V_{REF}	-	22	-	µs

Note:

1. 50% duty cycle is required for clock frequencies above 1500 kHz.

Table 36-27. Accuracy Characteristics(2)

Symbol	Description	Conditions		Min.	Typ.	Max.	Unit
RES	Resolution			-	10	-	bit
INL	Integral nonlinearity	REFSEL = INTERNAL V_{REF} = 0.55V	f_{ADC} = 7.7 ksps	-	1.0	-	LSb
		REFSEL = INTERNAL or V_{DD}	f_{ADC} = 15 ksps	-	1.0	-	
		REFSEL = INTERNAL or V_{DD} 1.1V ≤ V_{REF}	f_{ADC} = 77 ksps	-	1.0	-	
			f_{ADC} = 115 ksps	-	1.2	-	

.....continued

Symbol	Description	Conditions		Min.	Typ.	Max.	Unit
DNL[1]	Differential nonlinearity	REFSEL = INTERNAL V_{REF} = 0.55V	f_{ADC} = 7.7 ksps	-	0.6	-	LSb
		REFSEL = INTERNAL or V_{DD}	f_{ADC} = 15 ksps	-	0.4	-	
		REFSEL = INTERNAL or V_{DD} 1.1V ≤ V_{REF}	f_{ADC} = 77 ksps	-	0.4	-	
		REFSEL = INTERNAL 1.1V ≤ V_{REF}	f_{ADC} = 115 ksps	-	0.6	-	
		REFSEL = V_{DD} 1.1V ≤ V_{REF}	f_{ADC} = 115 ksps	-	0.6	-	
EABS	Absolute accuracy	REFSEL = INTERNAL V_{REF} = 1.1V	T = [0, 105]°C V_{DD} = [1.8V, 3.6V]	-	3	-	LSb
			V_{DD} = [1.8V, 3.6V]	-	3	-	
		REFSEL = V_{DD}		-	2	-	
		REFSEL = INTERNAL		-	3	-	
EGAIN	Gain error	REFSEL = INTERNAL V_{REF} = 1.1V	T = [0, 105]°C V_{DD} = [1.8V, 3.6V]	-	5	-	LSb
			V_{DD} = [1.8V, 3.6V]	-	5	-	
		REFSEL = V_{DD}		-	2	-	
		REFSEL = INTERNAL		-	5	-	
EOFF	Offset error			-	-0.5	-	LSb

Note:

1. A DNL error of ≤ 1 LSb ensures a monotonic transfer function with no missing codes.
2. These values are based on characterization and not covered by production test limits.

36.17 TEMPSENSE

Operating conditions:

- VDD = 3V
- T_A = 25°C (unless otherwise stated)

Table 36-28. Temperature Sensor, Accuracy Characteristics

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{DD}	Supply voltage		1.8	-	5.5	V
T_{ACC}	Sensor accuracy[1,2]	T_A = 25°C	-	±3	-	°C

.....continued

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
T_{RES}	Conversion resolution	10 bits	-	0.55	-	°C
t_{CONV}	Conversion time	1 MHz ADC clock	-	13	-	µs

Note:

1. These values are based on characterization and not covered by production test limits.
2. Characteristics over temperature can be found in the *Typical Characteristics* section.

36.18 DAC

V_{DD} = 3V, unless stated otherwise.

Accuracy characteristics calculated based on 5% to 95% range of the DAC.

Table 36-29. Power Supply, Reference, and Input Range

Symbol	Description	Min.	Typ.	Max.	Unit
V_{DD}	Supply voltage(1)	1.8	3	5.5	V
R_{Load}	Resistive external load	5	-	-	kΩ
C_{Load}	Capacitive external load	-	-	30	pF
V_{OUT}	Output voltage range	0.2	-	V_{DD}-0.2	V
I_{OUT}	Output sink/source	-	1	-	mA

Note: 1. The supply voltage must meet the V_{DD} specification for the V_{REF} level used as DAC reference.

Table 36-30. Clock and Timing Characteristics

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
f_{DAC}	Maximum conversion rate	0.55V ≤ V_{REF} ≤ 2.5V	-	350	-	ksps
		V_{REF} = 4.3V	-	270	-	ksps

Table 36-31. Accuracy Characteristics(3)

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
RES	Resolution		-		8	bits
INL	Integral nonlinearity	0.55V ≤ V_{REF} ≤ 4.3V	-1.2	0.3	1.2	LSb
DNL	Differential nonlinearity	0.55V ≤ V_{REF} ≤ 4.3V	-1	0.25	1	LSb
EOFF(1)	Offset error	0.55V ≤ V_{REF} ≤ 1.5V	-25	-3	20	mV
		V_{REF} = 2.5V	-30	-6	10	
		V_{REF} = 4.3V	-40	-10	0	
EGAIN(2)	Gain error	V_{REF} = 1.1V, V_{DD} = 3.0V, T = 25°C	-	±1	-	LSb
		0.55V ≤ V_{REF} ≤ 4.3V	-10	-1	10	

Note:

1. Offset including the DAC output buffer offset, this measured at DAC output pin.
2. V_{REF} accuracy is included in the Gain accuracy specification.
3. These values are based on characterization and not covered by production test limits.

36.19 AC

Table 36-32. Analog Comparator Characteristics, Low-Power Mode Disabled

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{IN}	Input voltage		-0.2	-	V_{DD}	V
C_{IN}	Input pin capacitance	PA6	-	9	-	pF
		PA7, PB5, PB4	-	5	-	
V_{OFF}	Input offset voltage	0.7V < V_{IN} < (V_{DD} - 0.7V)	-20	±5	20	mV
		V_{IN} = [-0.2V, V_{DD}]	-40	±20	40	
I_L	Input leakage current		-	5	-	nA
T_{START}	Start-up time		-	1.3	-	µs
V_{HYS}	Hysteresis	HYSMODE = 0x0	0	0	10	mV
		HYSMODE = 0x1	0	10	30	
		HYSMODE = 0x2	10	30	90	
		HYSMODE = 0x3	20	55	150	
t_{PD}	Propagation delay	25 mV Overdrive, V_{DD} ≥ 2.7V, Low-Power mode disabled	-	50	-	ns

Table 36-33. Analog Comparator Characteristics, Low-Power Mode Enabled

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
V_{IN}	Input voltage		0	-	V_{DD}	V
C_{IN}	Input pin capacitance	PA6	-	9	-	pF
		PA7, PB5, PB4	-	5	-	
V_{OFF}	Input offset voltage	0.7V < V_{IN} < (V_{DD} - 0.7V)	-30	±10	30	mV
		V_{IN} = [0V, V_{DD}]	-50	±30	50	
I_L	Input leakage current		-	5	-	nA
T_{START}	Start-up time		-	1.3	-	µs
V_{HYS}	Hysteresis	HYSMODE = 0x0	0	0	10	mV
		HYSMODE = 0x1	0	10	30	
		HYSMODE = 0x2	5	30	90	
		HYSMODE = 0x3	12	55	190	
t_{PD}	Propagation delay	25 mV Overdrive, V_{DD} ≥ 2.7V	-	150	-	ns

36.20 PTC

Table 36-34. Peripheral Touch Controller Characteristics - Operating Ratings

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
C_{LOAD}	Maximum load		-	48	-	pF
C_{INT}	Maximum size of integration capacitor		-	30	-	pF

.....continued						
Symbol	**Description**	**Condition**	**Min.**	**Typ.**	**Max.**	**Unit**
C_{DS}	Driven Shield capacitive drive		-	300	-	pF
CLK_{ADC}	Supported ADC clock frequency	25% duty cycle	200	-	1500	kHz
		50% duty cycle	200	-	2000	

Table 36-35. Peripheral Touch Controller Characteristics - Pad Capacitance

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
$C_{X/Y}$	Pad capacitance X/Y-line	PA4, X0/Y0	-	4	-	pF
		PA5, X1/Y1	-	24	-	
		PA6, X2/Y2	-	9	-	
		PA7, X3/Y3		6		
		PB5, X12/Y12	-	4	-	
		PB4, X13/Y13	-	4	-	
		PB1, X4/Y4	-	13	-	
		PB0,X5/Y5	-	13	-	
		PC0, X6/Y6	-	6	-	
		PC1, X7/Y7	-	6	-	
		PC2, X8/Y8	-	6	-	
		PC3, X9/Y9	-	6	-	
		PC4, X10/Y10	-	6	-	
		PC5, X11/Y11	-	6	-	

36.21 UPDI Timing

UPDI Enable Sequence with UPDI PAD Enabled by Fuse(1)

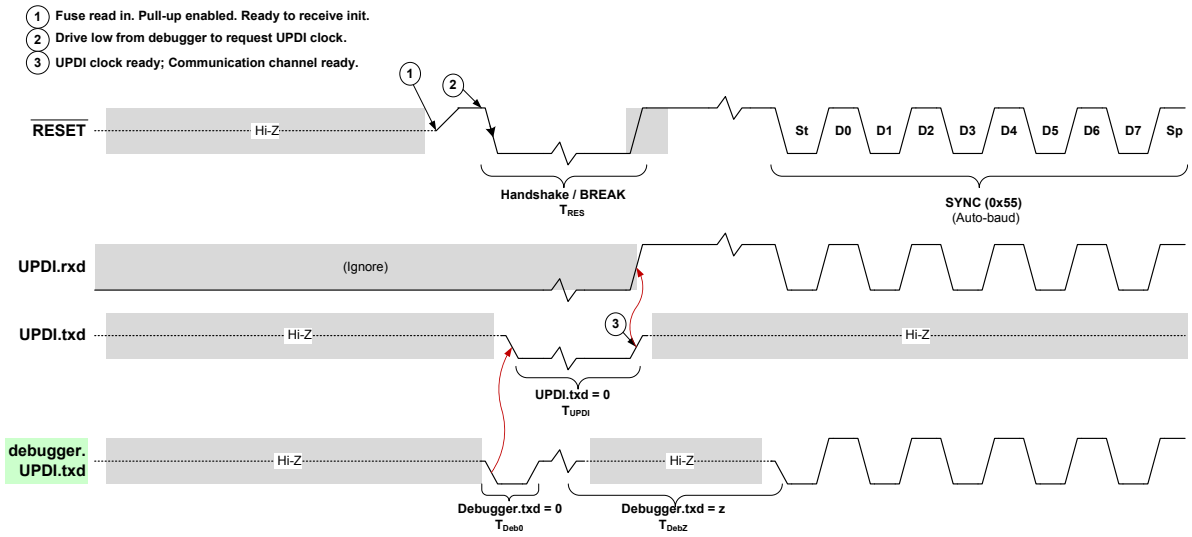


Table 36-36. UPDI Timing Characteristics(1)

Symbol	Description	Min.	Max.	Unit
T_{RES}	Duration of Handshake/BREAK on $\overline{RESET}$	10	200	µs
T_{UPDI}	Duration of UPDI.txd = 0	10	200	µs
T_{Deb0}	Duration of Debugger.txd = 0	0.2	1	µs
T_{DebZ}	Duration of Debugger.txd = z	200	14000	µs

Note:
1. These parameters are for design guidance only and are not covered by production test limits.

Table 36-37. UPDI Max. Bit Rates vs. VDD(1)

Symbol	Description	Condition	Max	Unit
f_{UPDI}	UPDI baud rate	V_{DD} = [1.8, 5.5]V T_A = [0, 50]°C	225	kbps
		V_{DD} = [2.2, 5.5]V T_A = [0, 50]°C	450	kbps
		V_{DD} = [2.7, 5.5]V T_A = [0, 50]°C	0.9	Mbps

Note:
1. These parameters are for design guidance only and are not covered by production test limits.

36.22 Programming Time

See the following table for typical programming times for Flash and EEPROM.

Table 36-38. Programming Times

Symbol	Typical Programming Time
Page Buffer Clear (PBC)	Seven CLK_CPU cycles

.....continued

Symbol	Typical Programming Time
Page Write (WP)	2 ms
Page Erase (ER)	2 ms
Page Erase-Write (ERWP)	4 ms
Chip Erase (CHER)	4 ms
EEPROM Erase (EEER)	4 ms

37. Typical Characteristics

37.1 Power Consumption

37.1.1 Supply Currents in Active Mode

Figure 37-1. Active Supply Current vs. Frequency (1-20 MHz) at T = 25°C

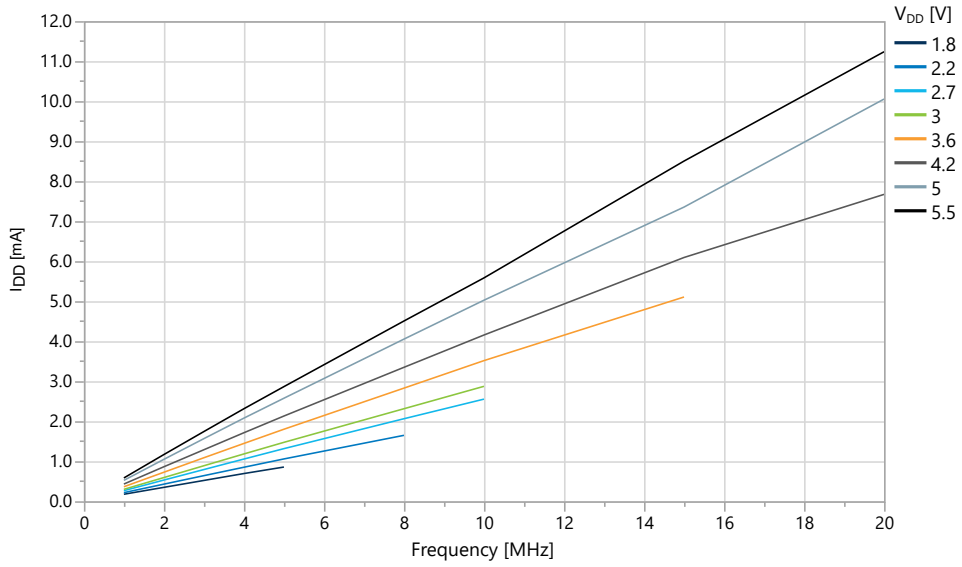


Figure 37-2. Active Supply Current vs. Frequency [0.1, 1.0] MHz at T = 25°C

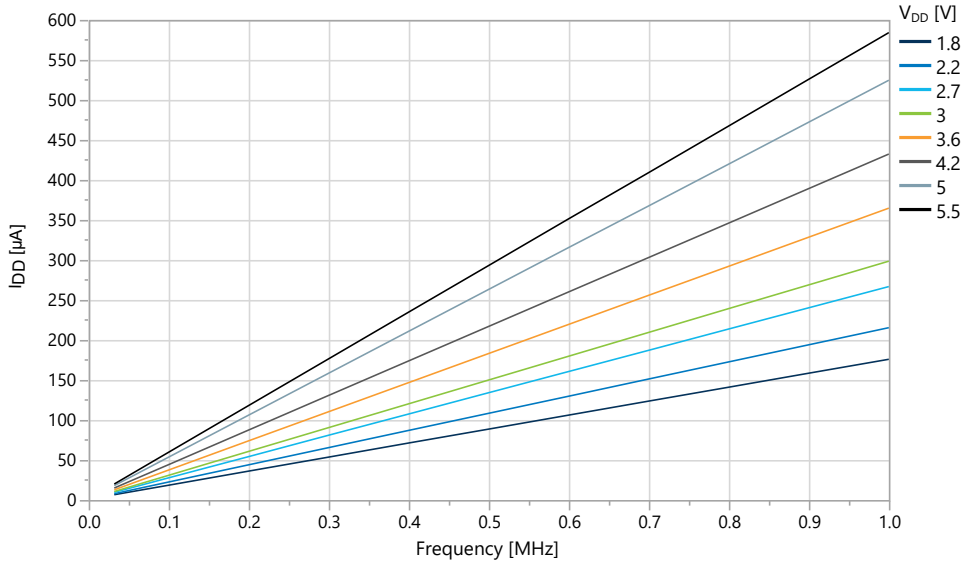


Figure 37-3. Active Supply Current vs. Temperature (f = 20 MHz OSC20M)

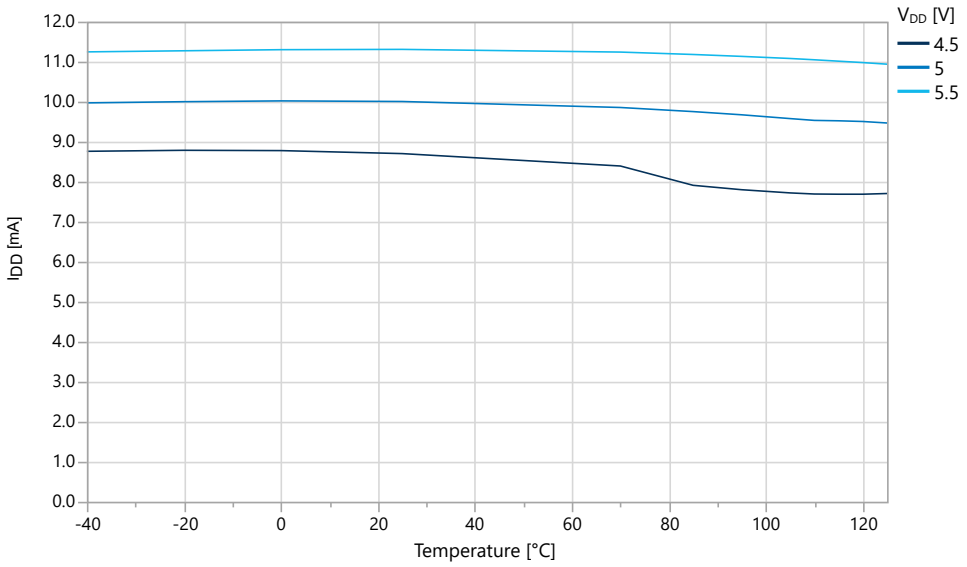


Figure 37-4. Active Supply Current vs. V_{DD} (f = [1.25, 20] MHz OSC20M) at T = 25°C

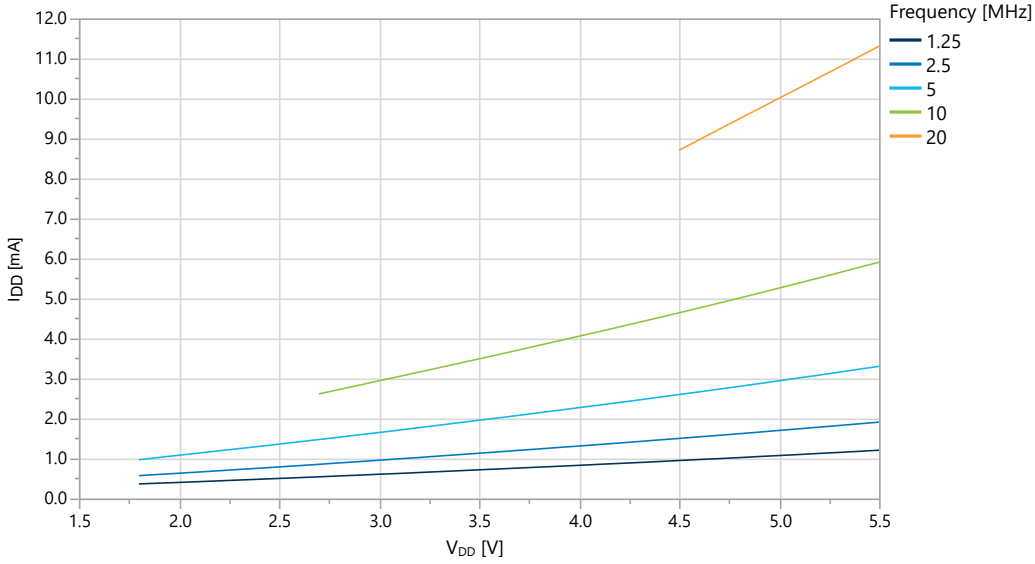
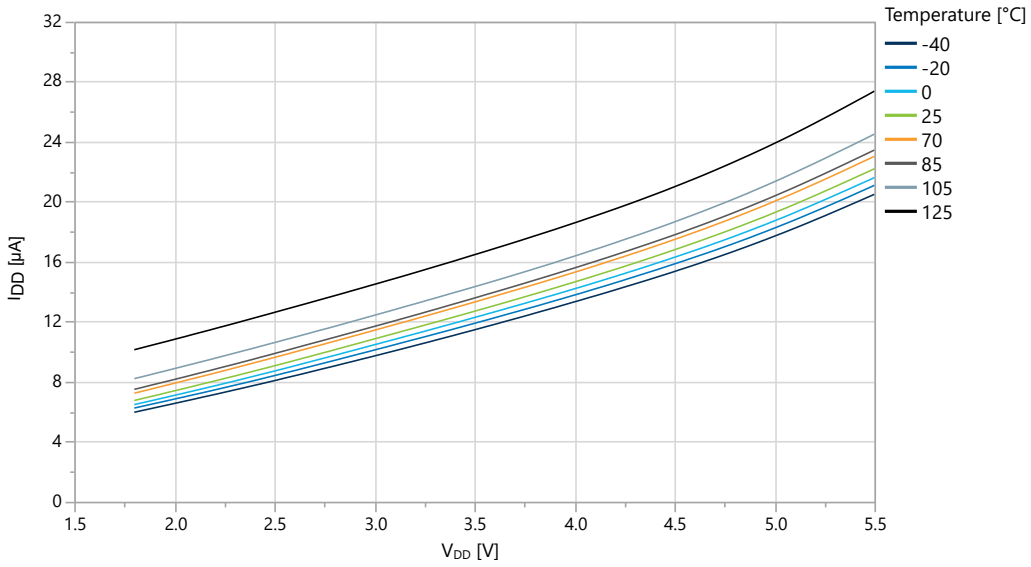


Figure 37-5. Active Supply Current vs. V_{DD} (f = 32.768 kHz OSCULP32K)



37.1.2 Supply Currents in Idle Mode

Figure 37-6. Idle Supply Current vs. Frequency (1-20 MHz) at T = 25°C

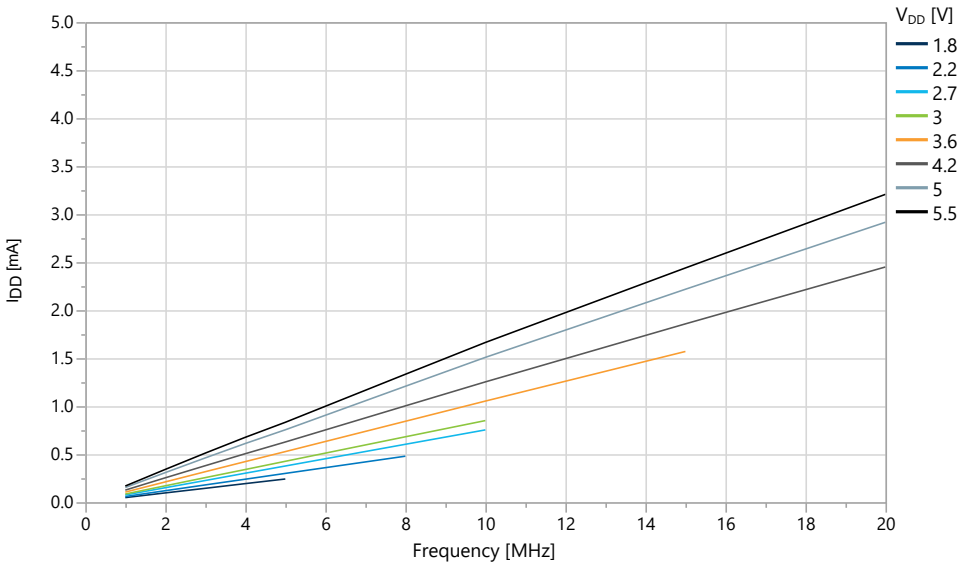


Figure 37-7. Idle Supply Current vs. Low Frequency (0.1-1.0 MHz) at T = 25°C

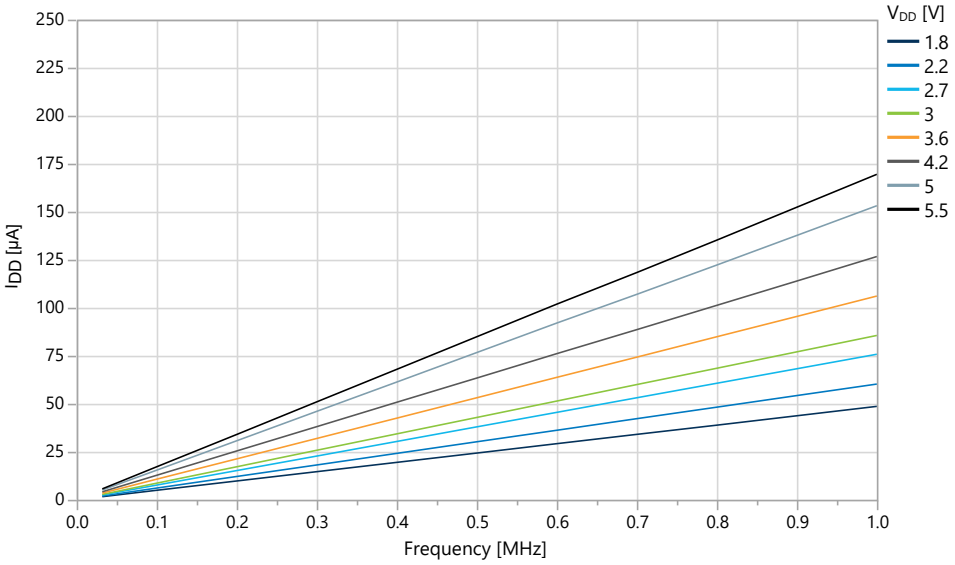


Figure 37-8. Idle Supply Current vs. Temperature (f = 20 MHz OSC20M)

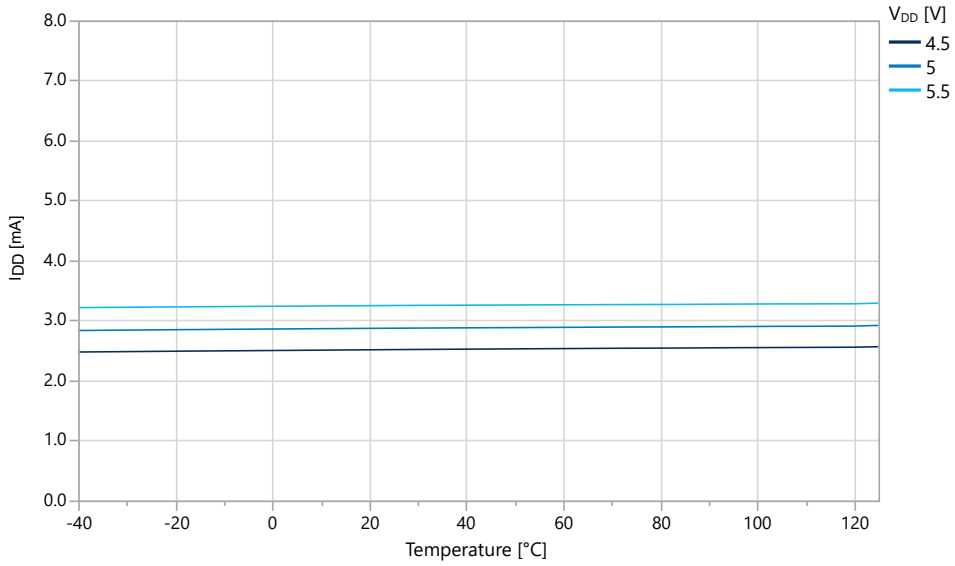
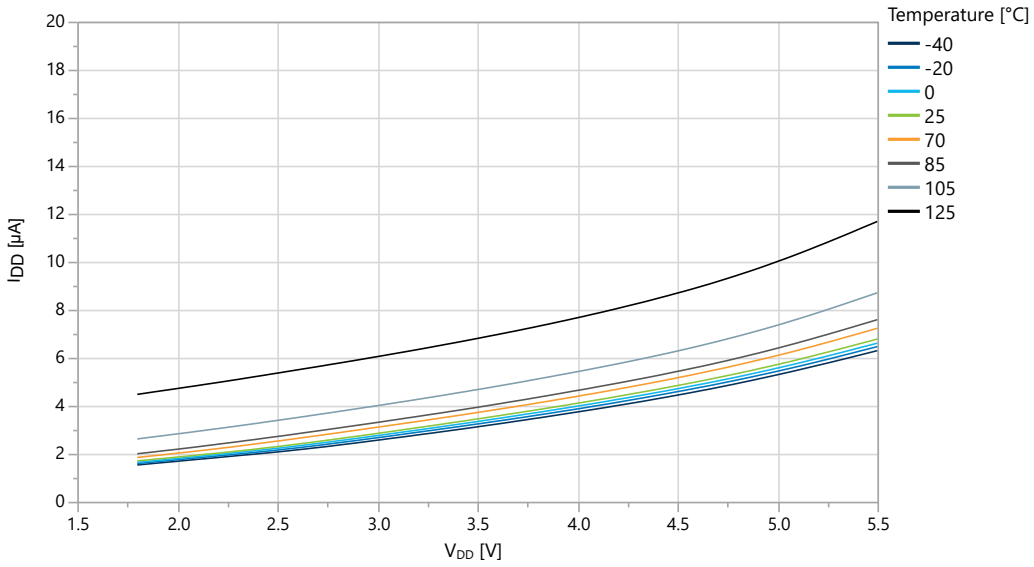


Figure 37-9. Idle Supply Current vs. V_{DD} (f = 32.768 kHz OSCULP32K)



37.1.3 Supply Currents in Standby Mode

Figure 37-10. Standby Mode Supply Current vs. V_{DD} (RTC Running with Internal OSCULP32K)

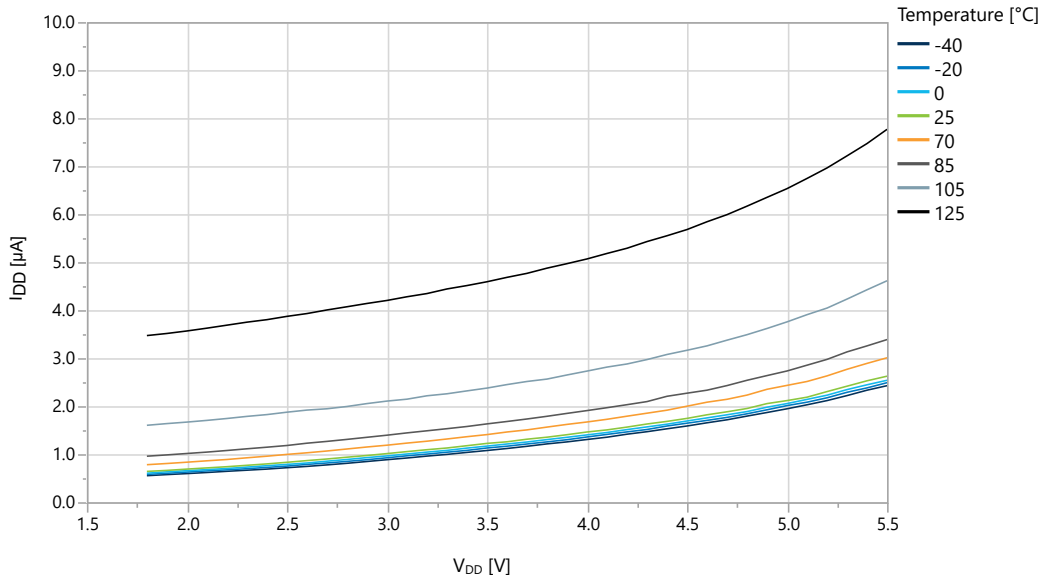


Figure 37-11. Standby Mode Supply Current vs. V_{DD} (Sampled BOD Running at 125 Hz)

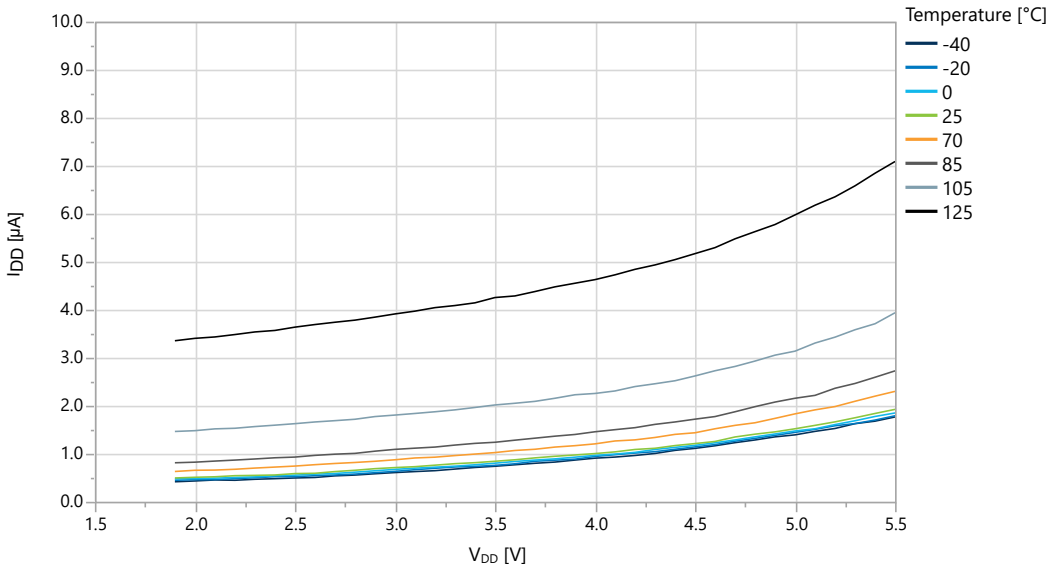
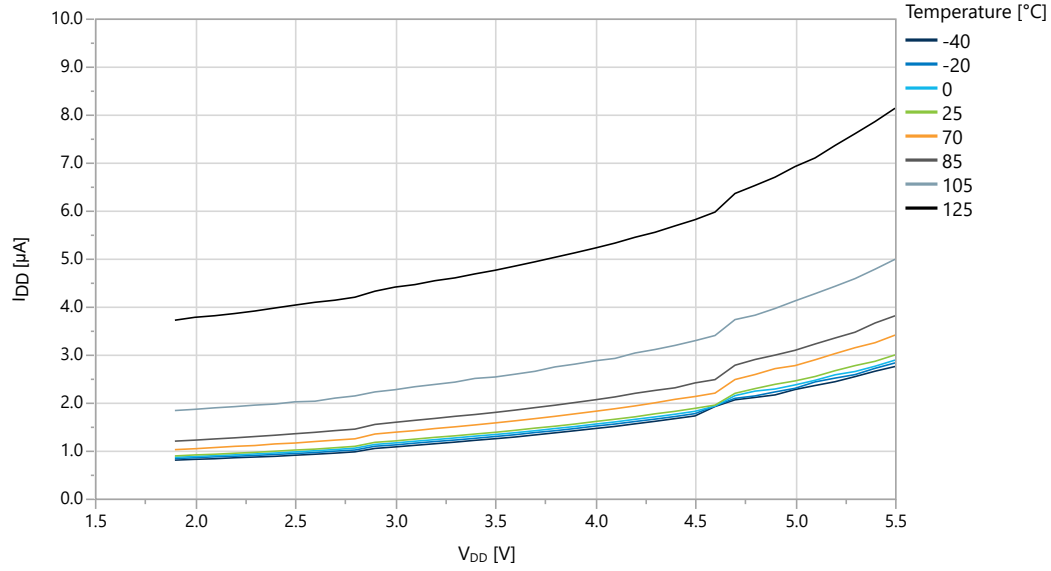


Figure 37-12. Standby Mode Supply Current vs. V_{DD} (Sampled BOD Running at 1 kHz)



37.1.4 Supply Currents in Power-Down Mode

Figure 37-13. Power-Down Mode Supply Current vs. Temperature (All Functions Disabled)

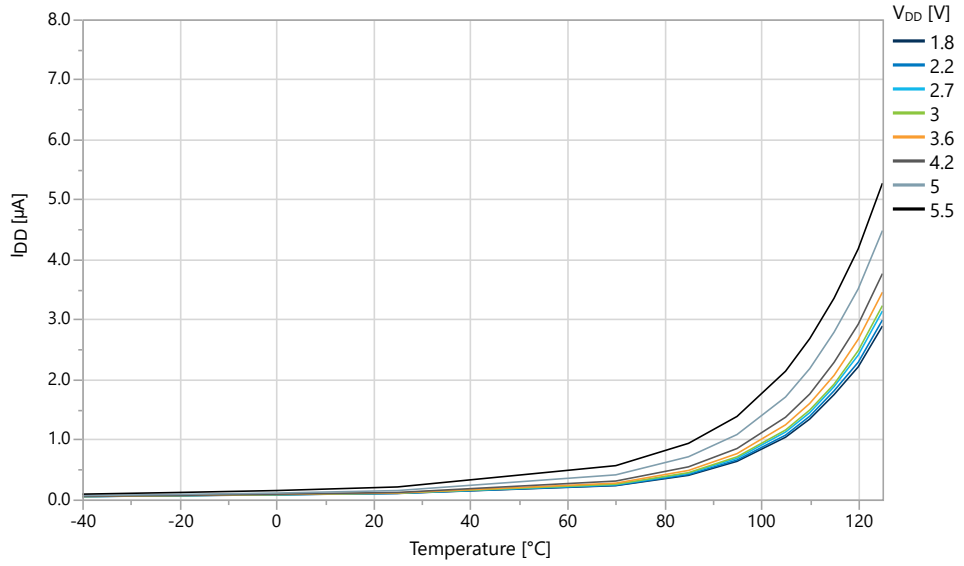
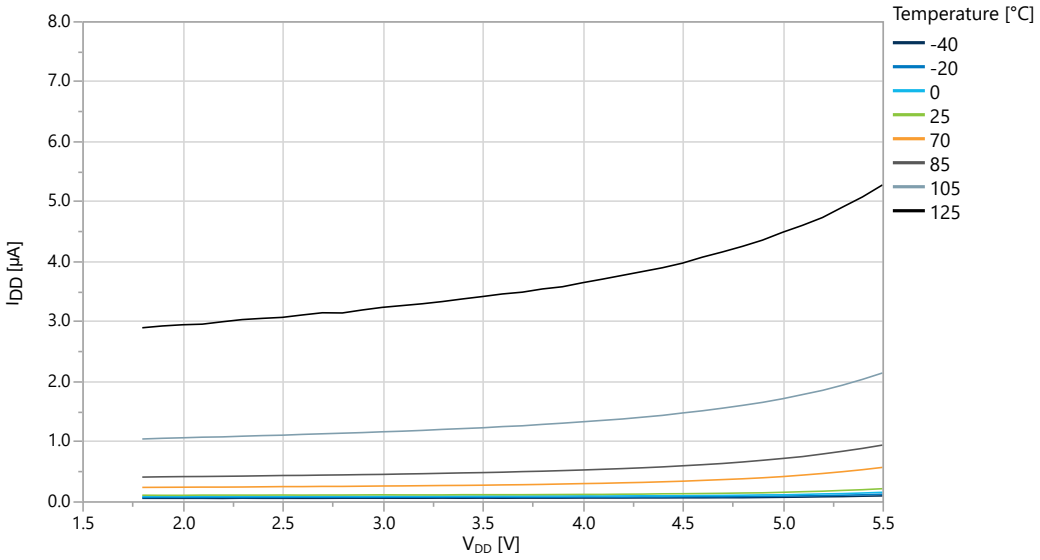


Figure 37-14. Power-Down Mode Supply Current vs. V_{DD} (All Functions Disabled)



37.2 GPIO

37.2.1 GPIO Input Characteristics

Figure 37-15. I/O Pin Input Hysteresis vs. V_{DD}

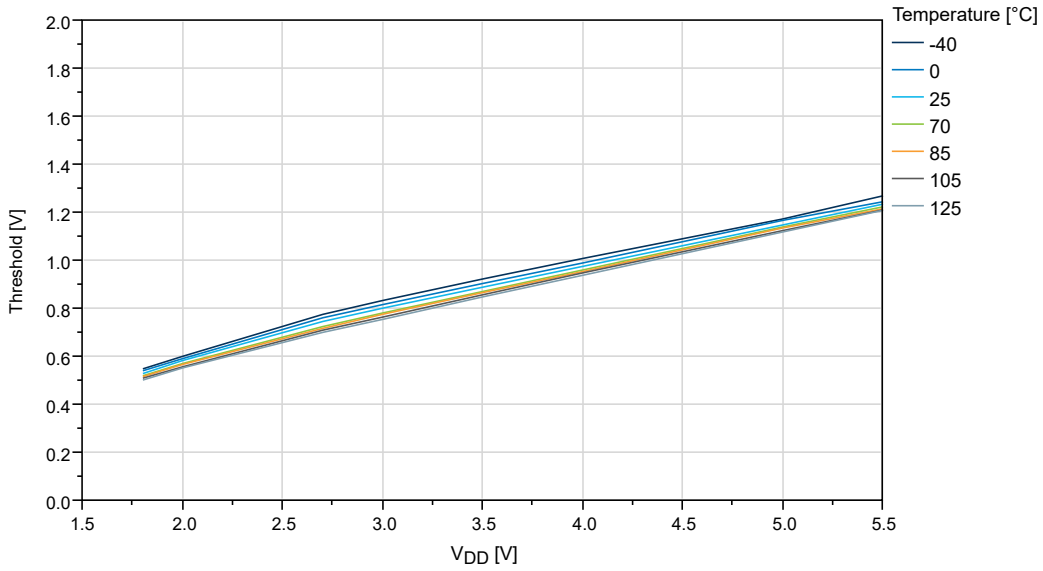


Figure 37-16. I/O Pin Input Threshold Voltage vs. V_{DD} (T = 25°C)

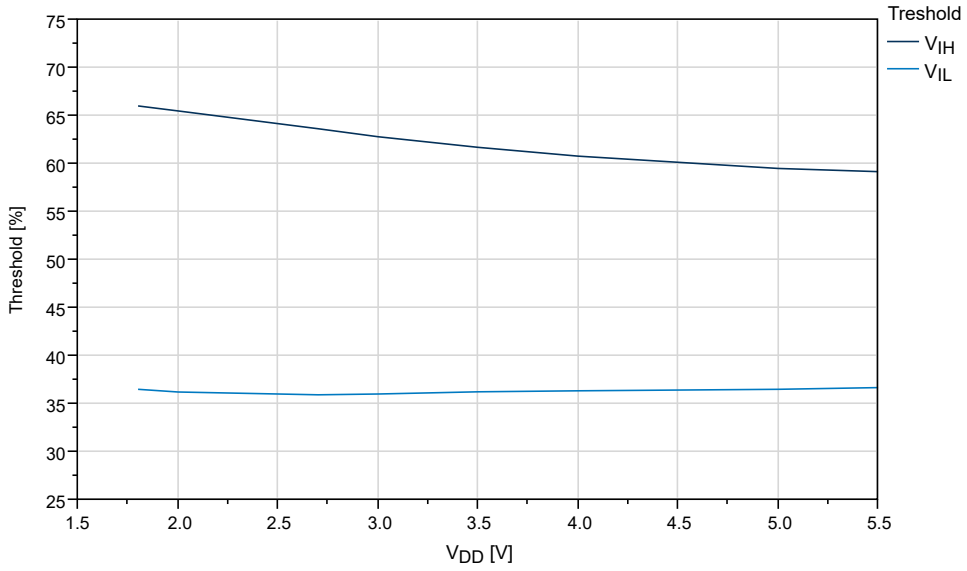


Figure 37-17. I/O Pin Input Threshold Voltage vs. V_{DD} (V_{IH})

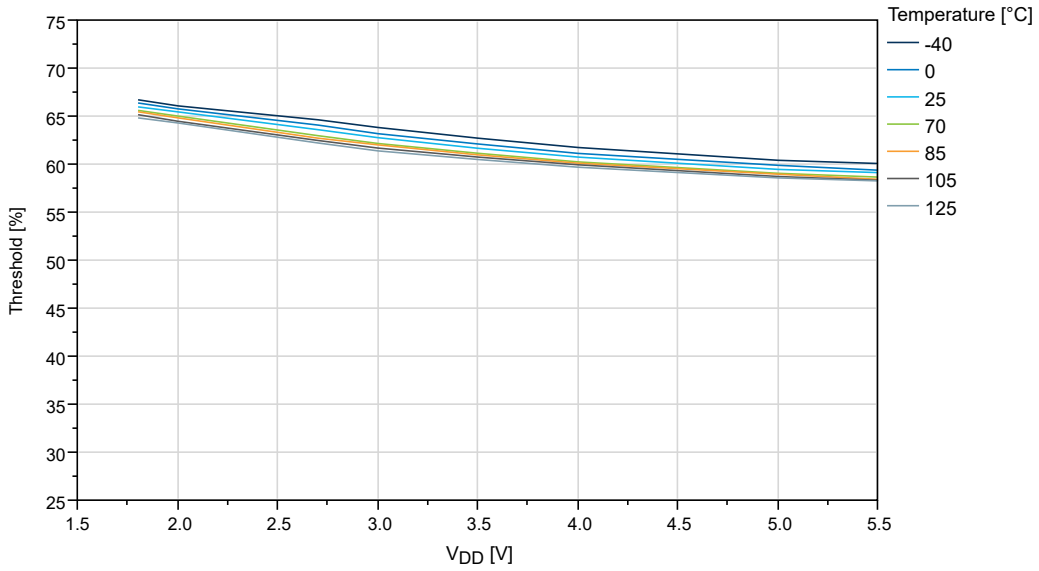
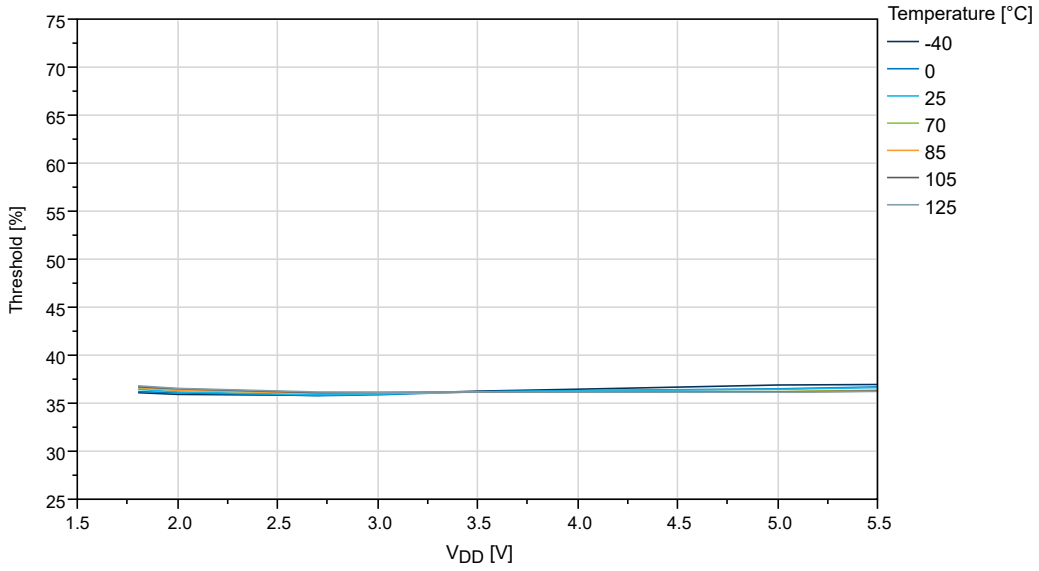


Figure 37-18. I/O Pin Input Threshold Voltage vs. V_{DD} (V_{IL})



37.2.2 GPIO Output Characteristics

Figure 37-19. I/O Pin Output Voltage vs. Sink Current (V_{DD} = 1.8V)

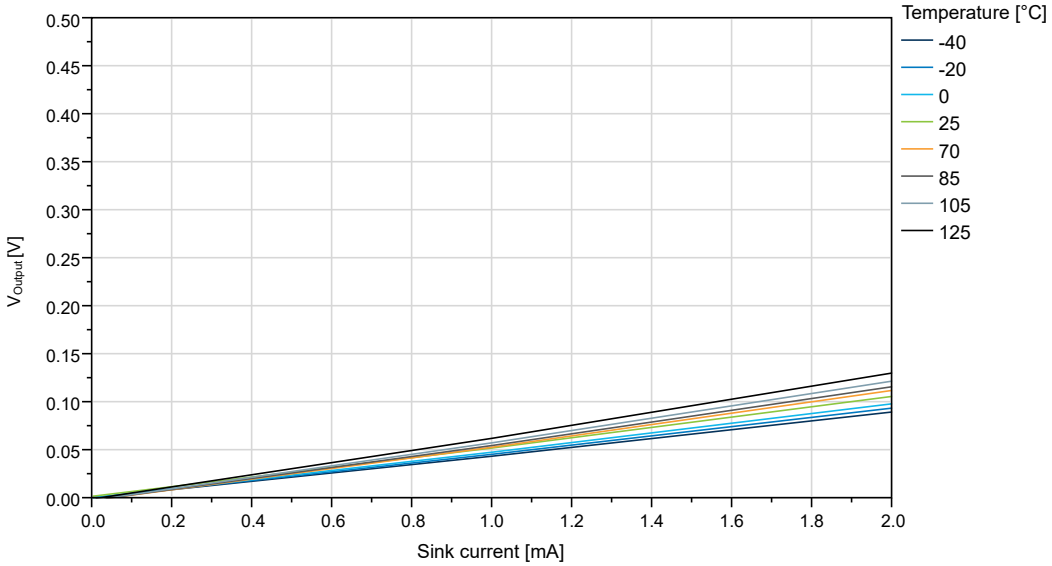


Figure 37-20. I/O Pin Output Voltage vs. Sink Current (V_{DD} = 3.0V)

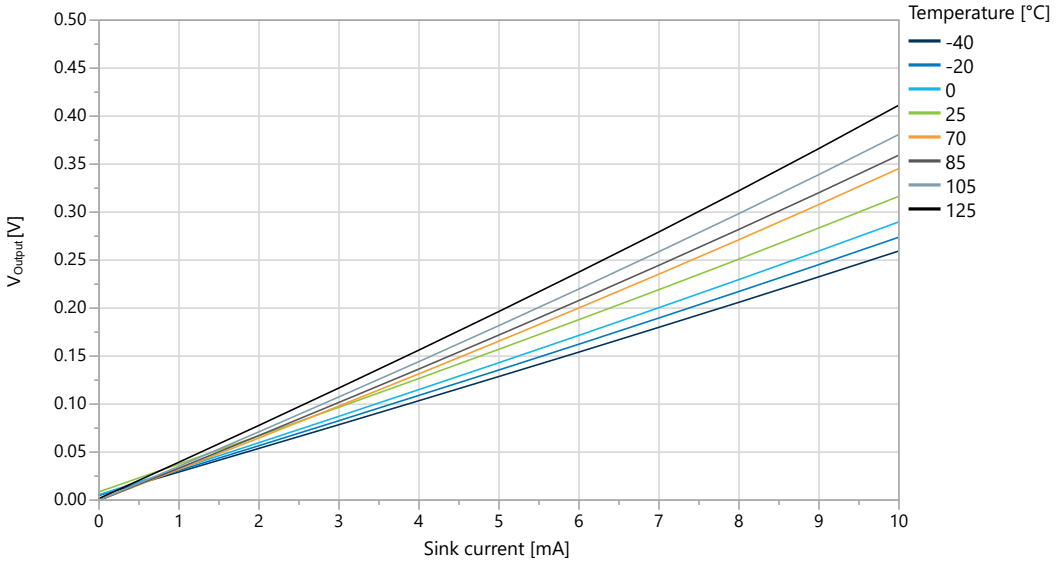


Figure 37-21. I/O Pin Output Voltage vs. Sink Current (V_{DD} = 5.0V)

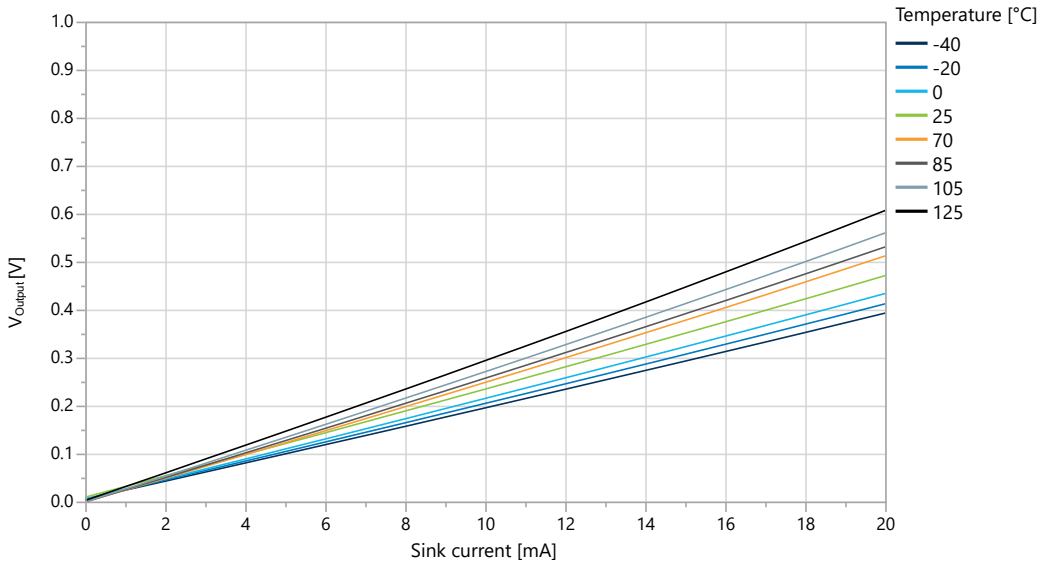


Figure 37-22. I/O Pin Output Voltage vs. Sink Current (T = 25°C)

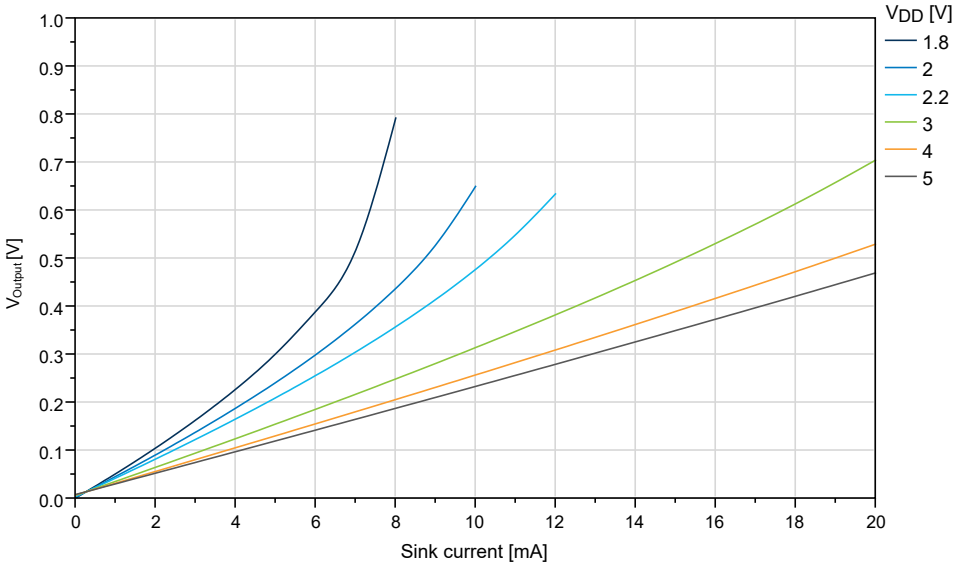


Figure 37-23. I/O Pin Output Voltage vs. Source Current (V_{DD} = 1.8V)

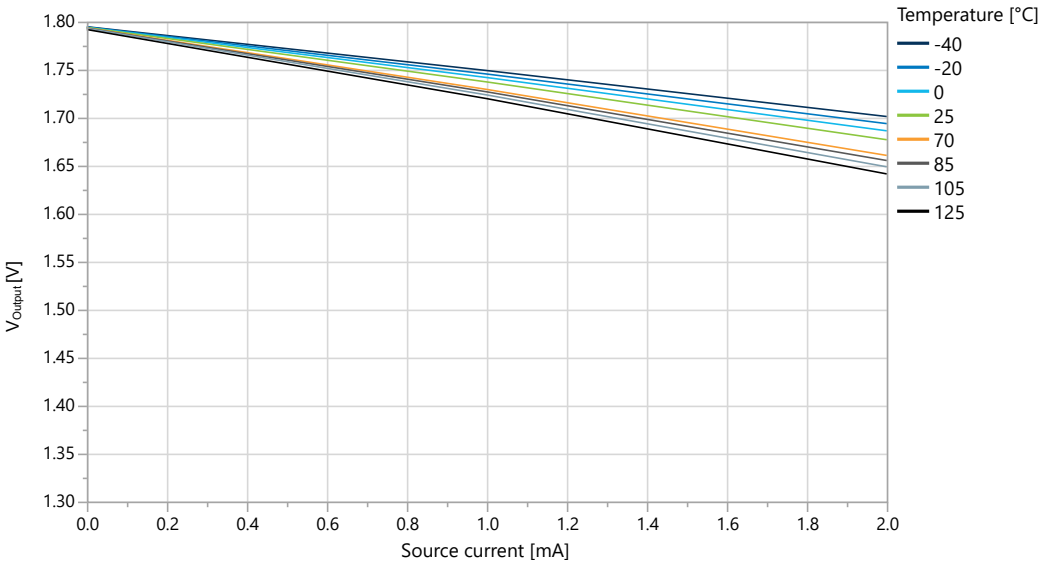


Figure 37-24. I/O Pin Output Voltage vs. Source Current (V_{DD} = 3.0V)

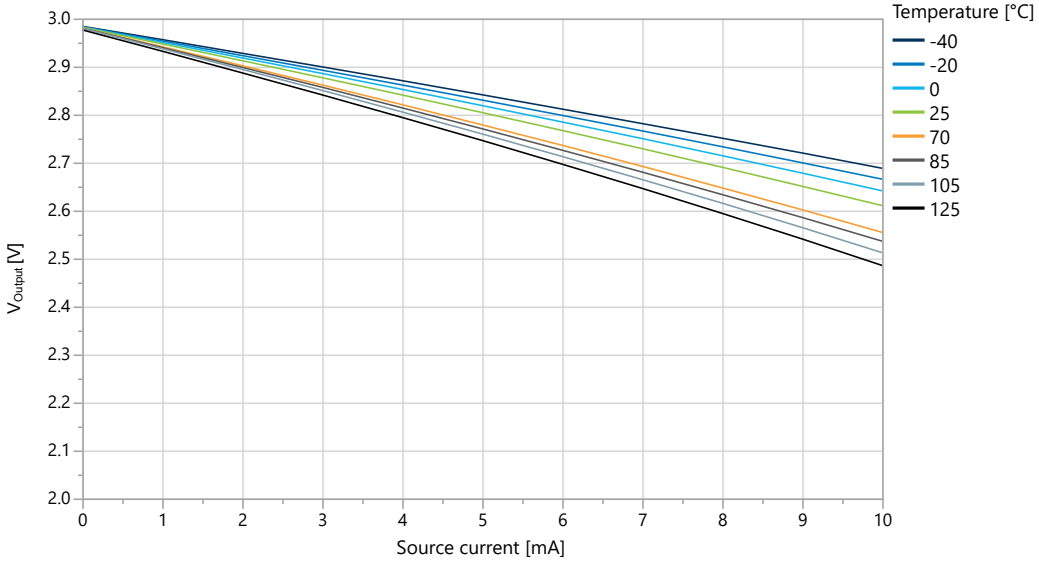


Figure 37-25. I/O Pin Output Voltage vs. Source Current (V_{DD} = 5.0V)

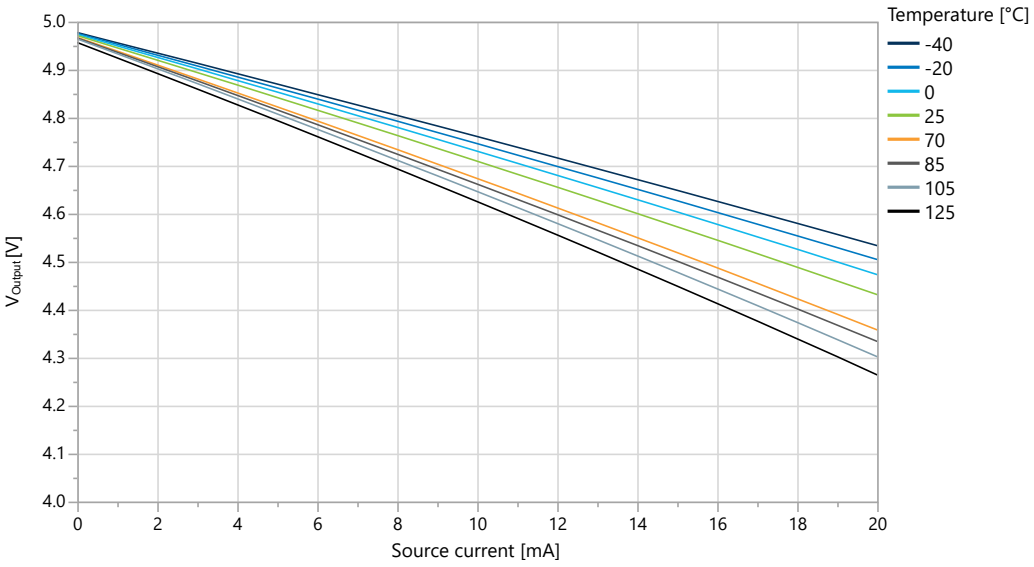
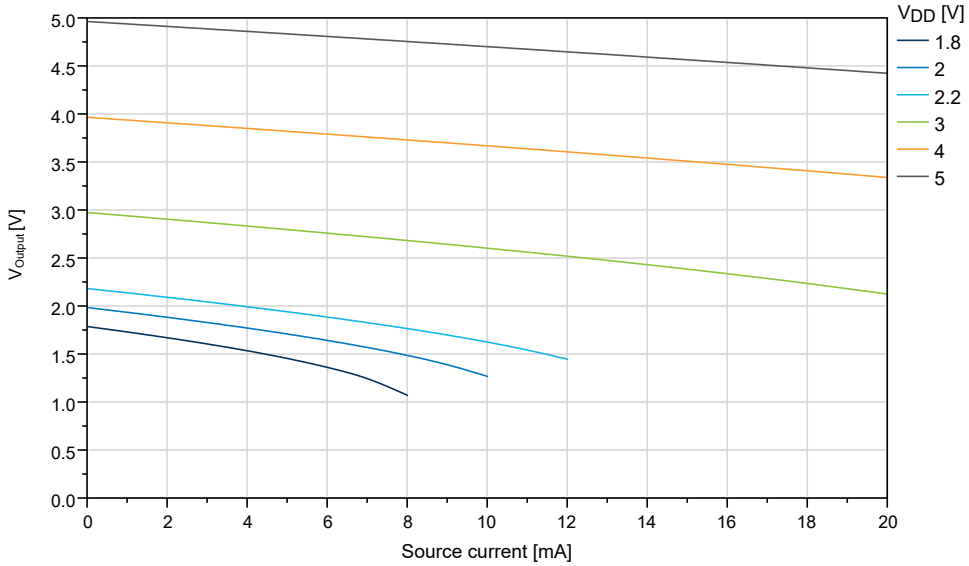


Figure 37-26. I/O Pin Output Voltage vs. Source Current (T = 25°C)



37.2.3 GPIO Pull-Up Characteristics

Figure 37-27. I/O Pin Pull-Up Resistor Current vs. Input Voltage (V_{DD} = 1.8V)

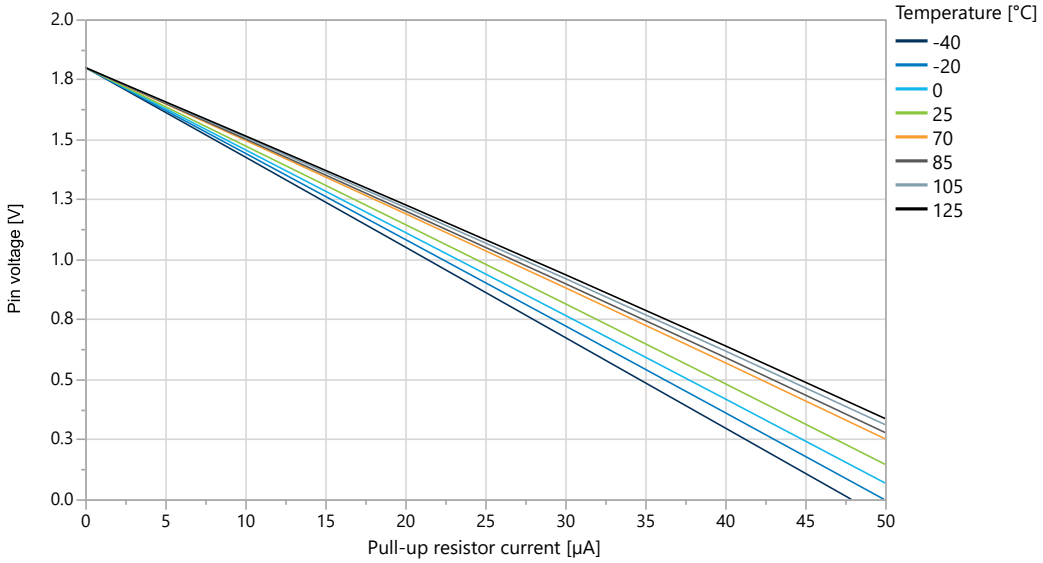


Figure 37-28. I/O Pin Pull-Up Resistor Current vs. Input Voltage (V_{DD} = 3.0V)

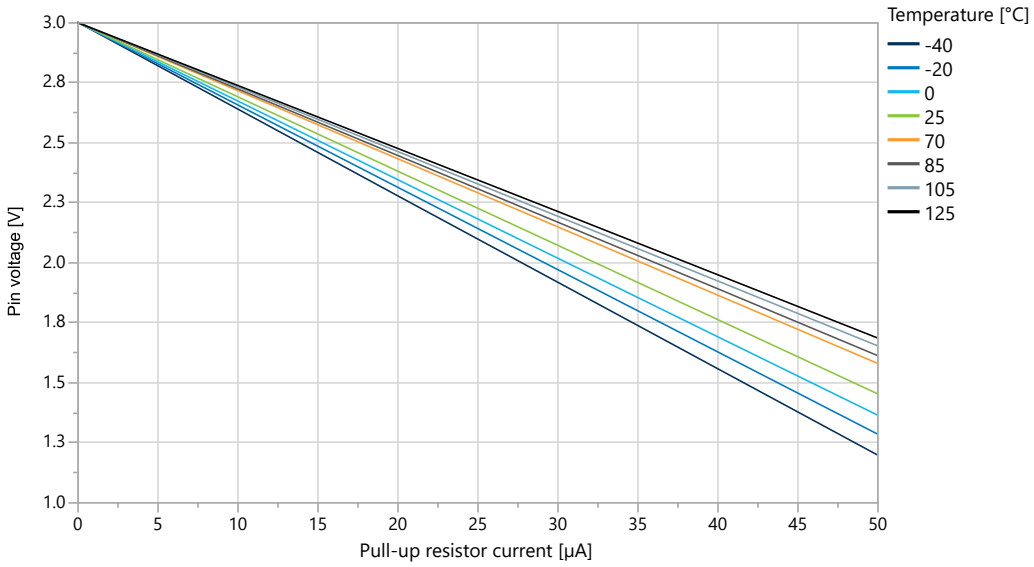
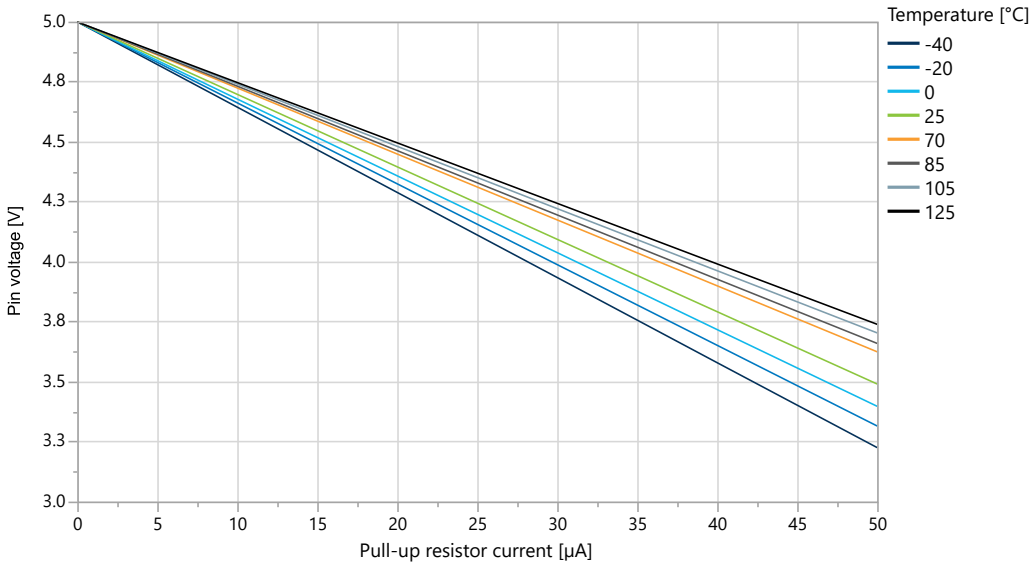


Figure 37-29. I/O Pin Pull-Up Resistor Current vs. Input Voltage (V_{DD} = 5.0V)



37.3 VREF Characteristics

Figure 37-30. Internal 0.55V Reference vs. Temperature

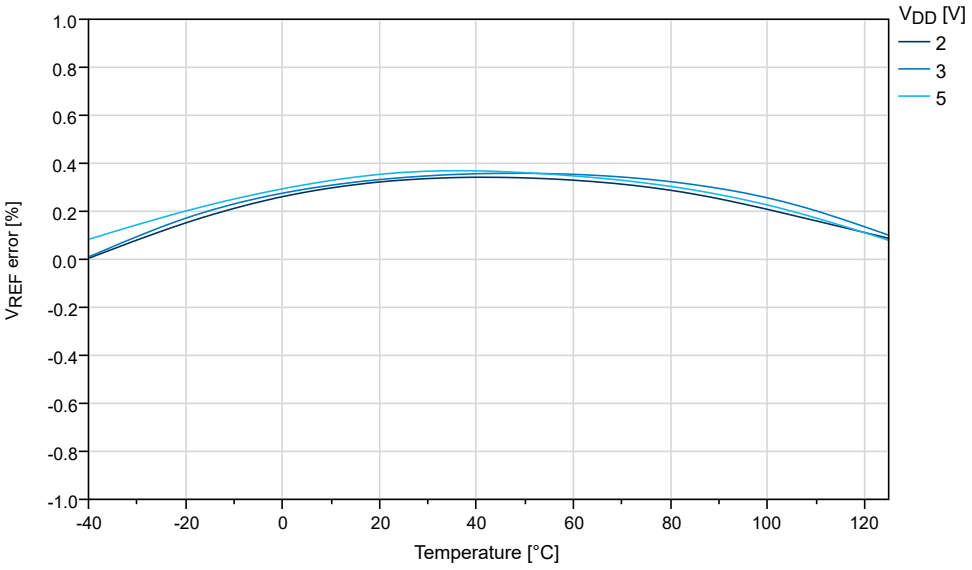


Figure 37-31. Internal 1.1V Reference vs. Temperature

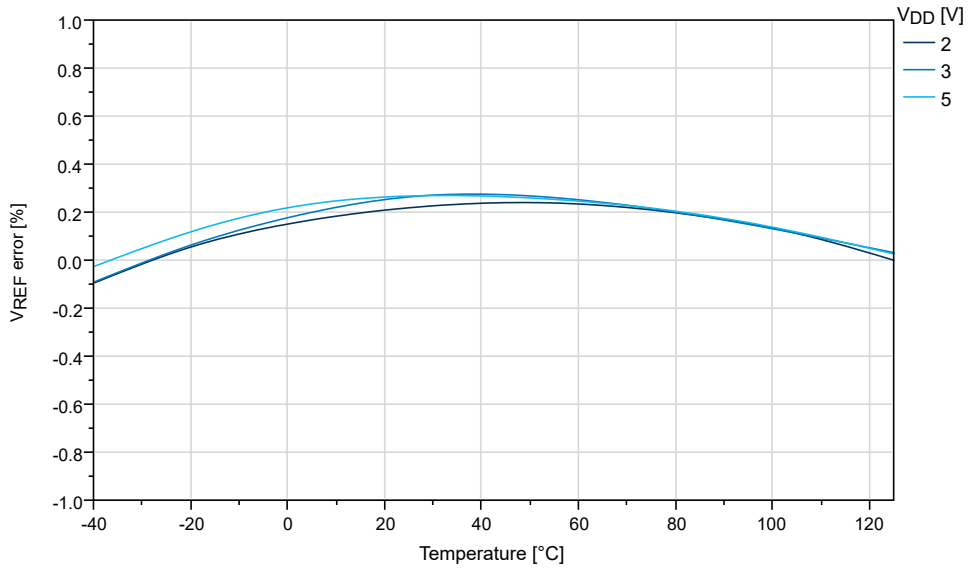


Figure 37-32. Internal 2.5V Reference vs. Temperature

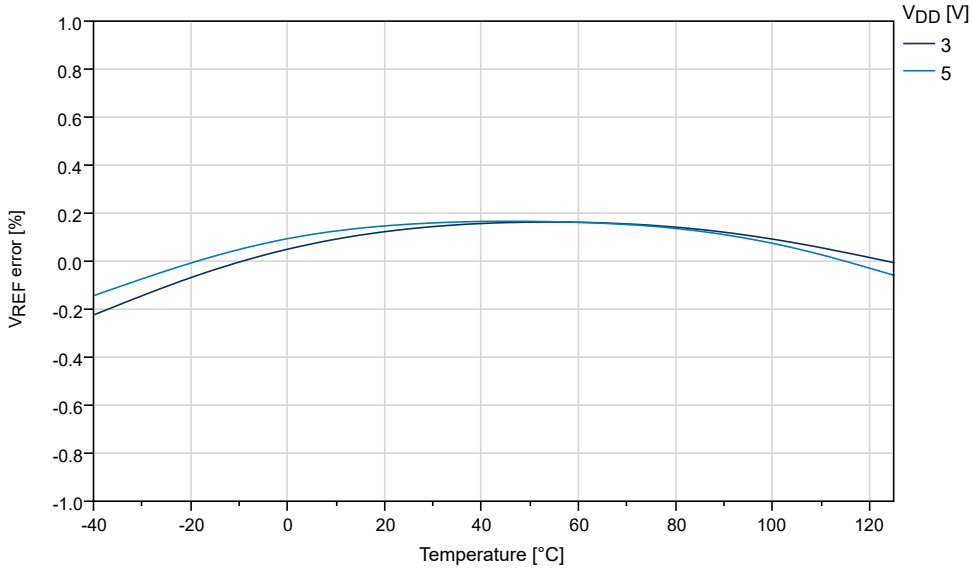
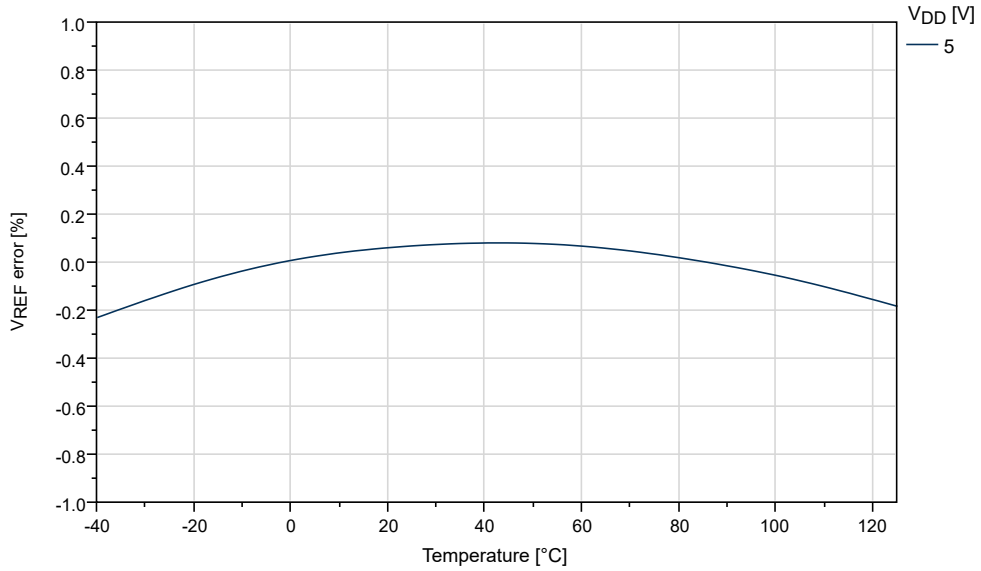


Figure 37-33. Internal 4.3V Reference vs. Temperature



37.4 BOD Characteristics

37.4.1 BOD Current vs. V_{DD}

Figure 37-34. BOD Current vs. V_{DD} (Continuous Mode Enabled)

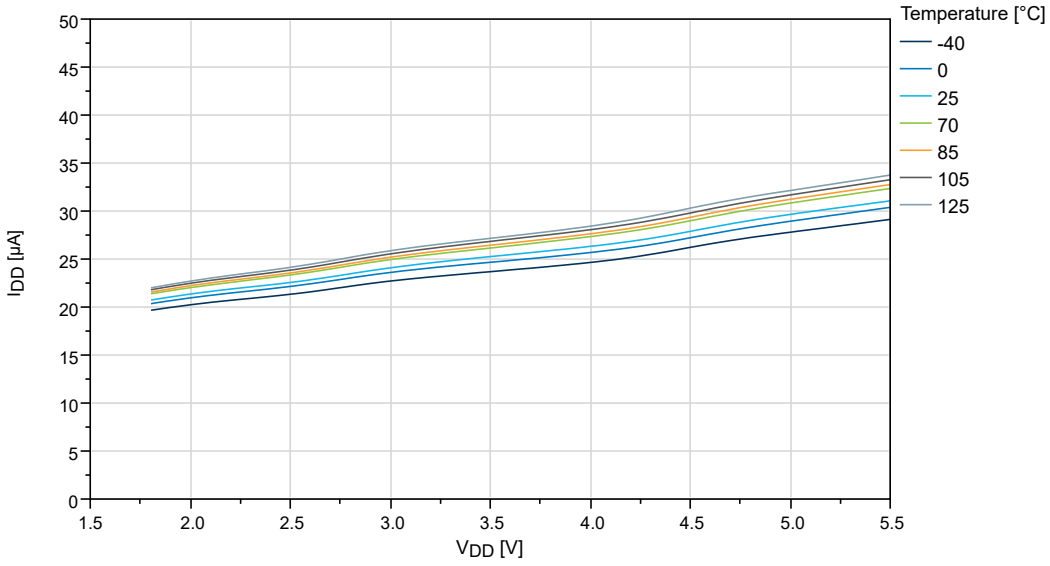


Figure 37-35. BOD Current vs. V_{DD} (Sampled BOD at 125 Hz)

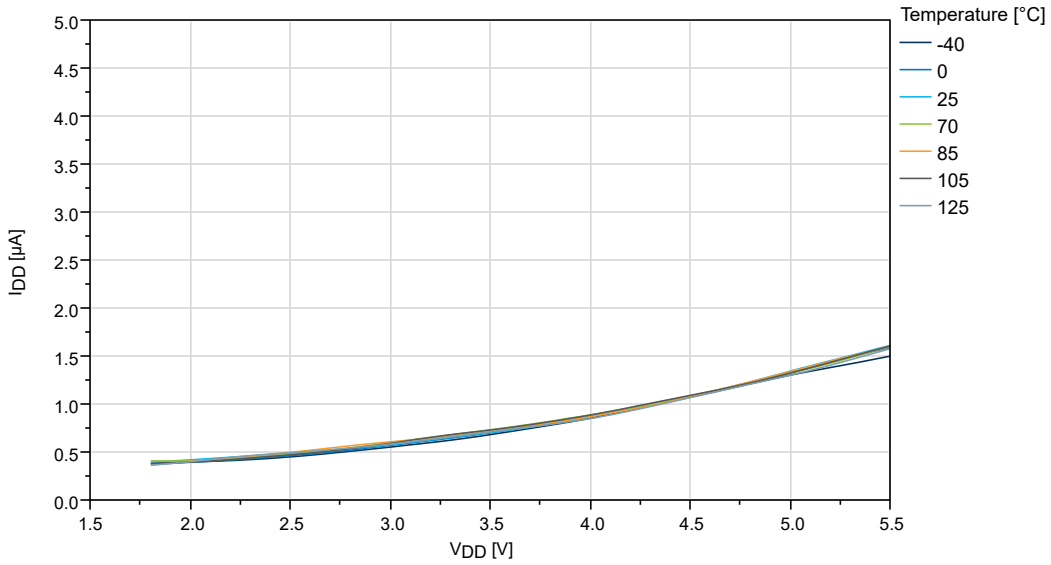
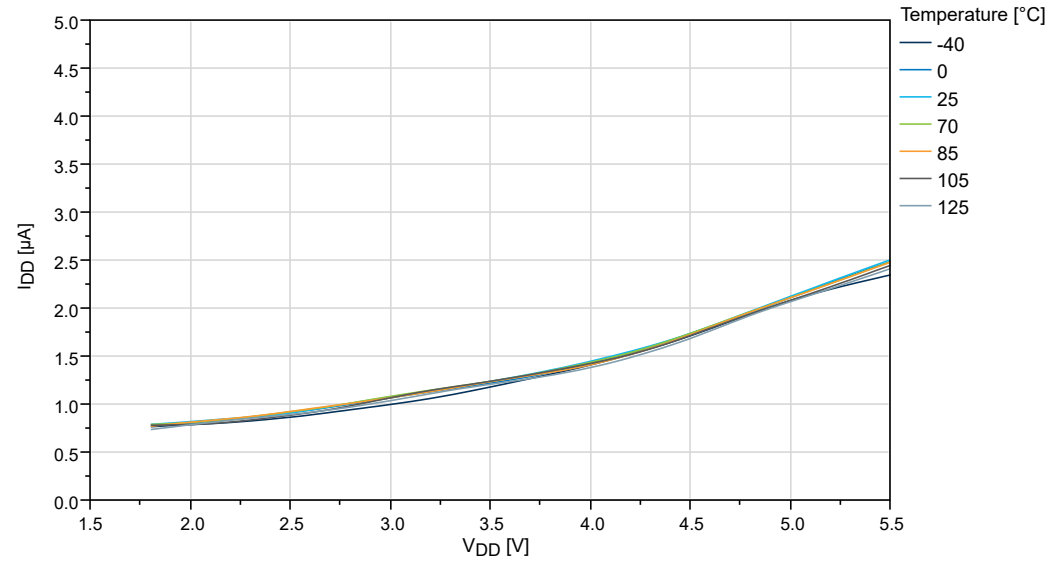


Figure 37-36. BOD Current vs. V_{DD} (Sampled BOD at 1 kHz)



37.4.2 BOD Threshold vs. Temperature

Figure 37-37. BOD Threshold vs. Temperature (Level 1.8V)

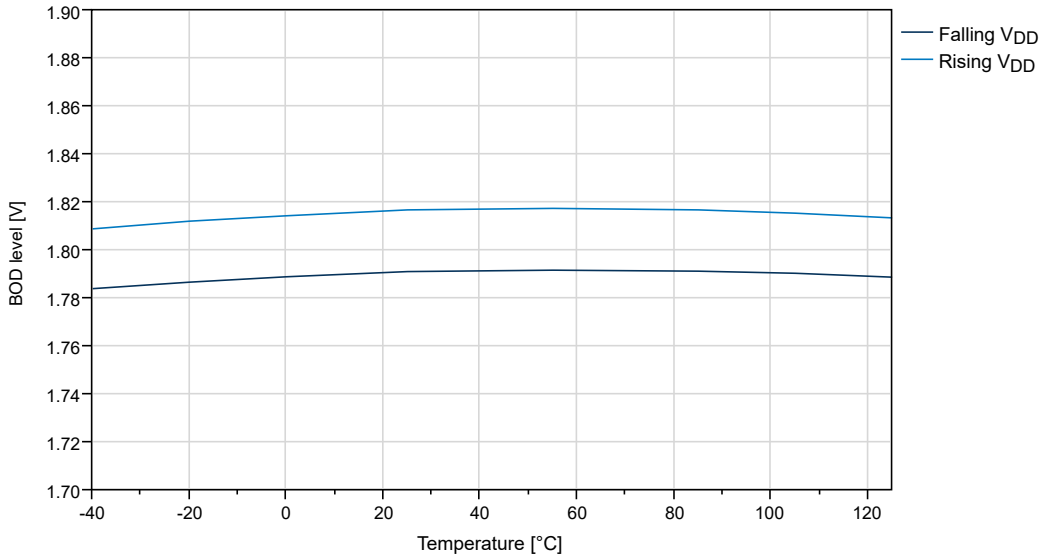


Figure 37-38. BOD Threshold vs. Temperature (Level 2.6V)

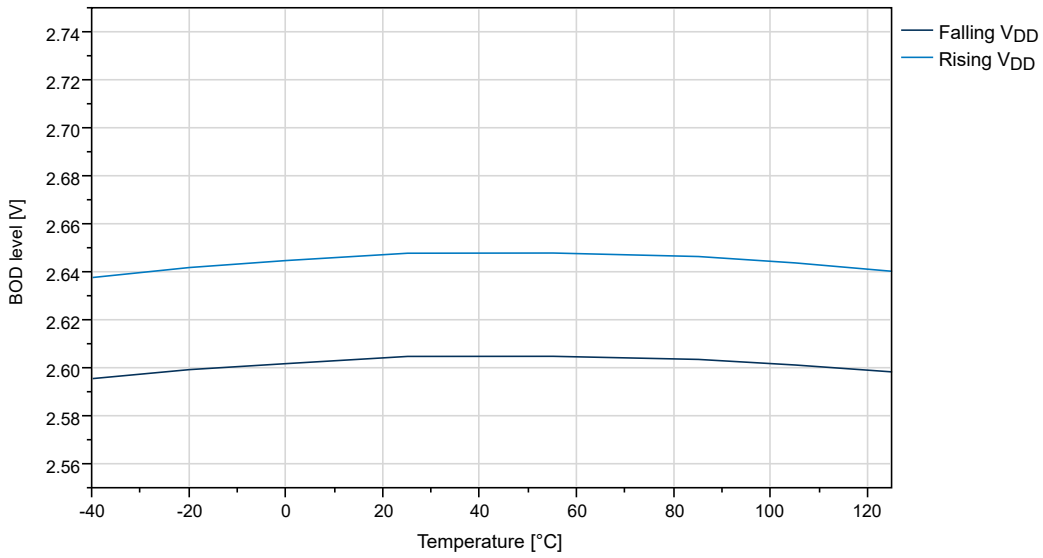
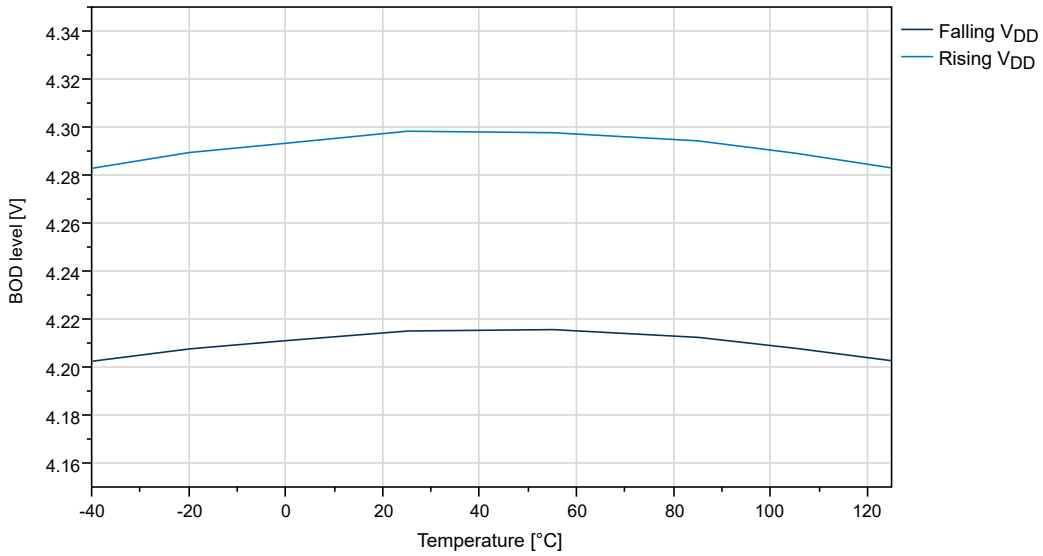


Figure 37-39. BOD Threshold vs. Temperature (Level 4.3V)



37.5 ADC Characteristics

Figure 37-40. Absolute Accuracy vs. V_{DD} (f_{ADC} = 115 ksps) at T = 25°C, REFSEL = Internal Reference

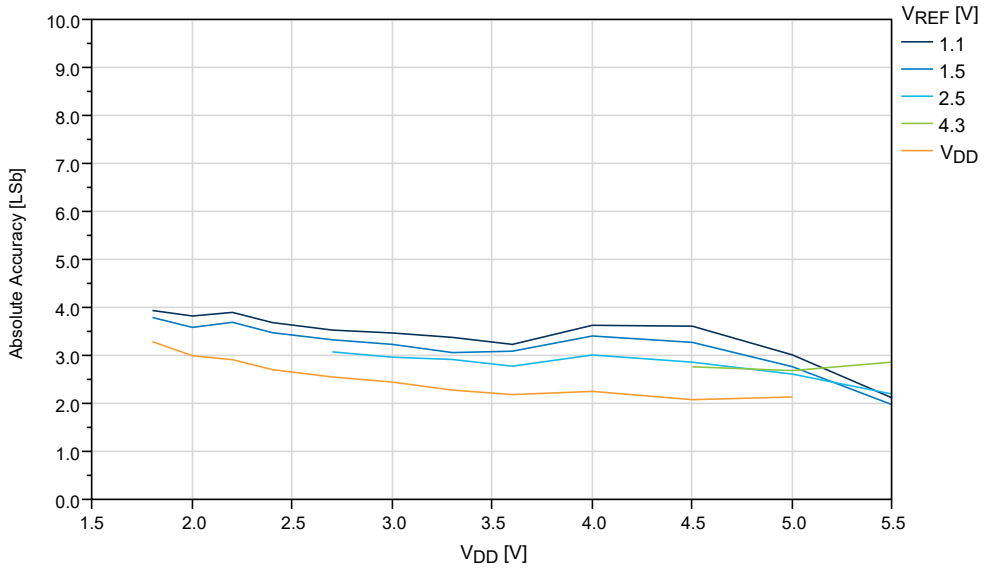


Figure 37-41. Absolute Accuracy vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps), REFSEL = Internal Reference

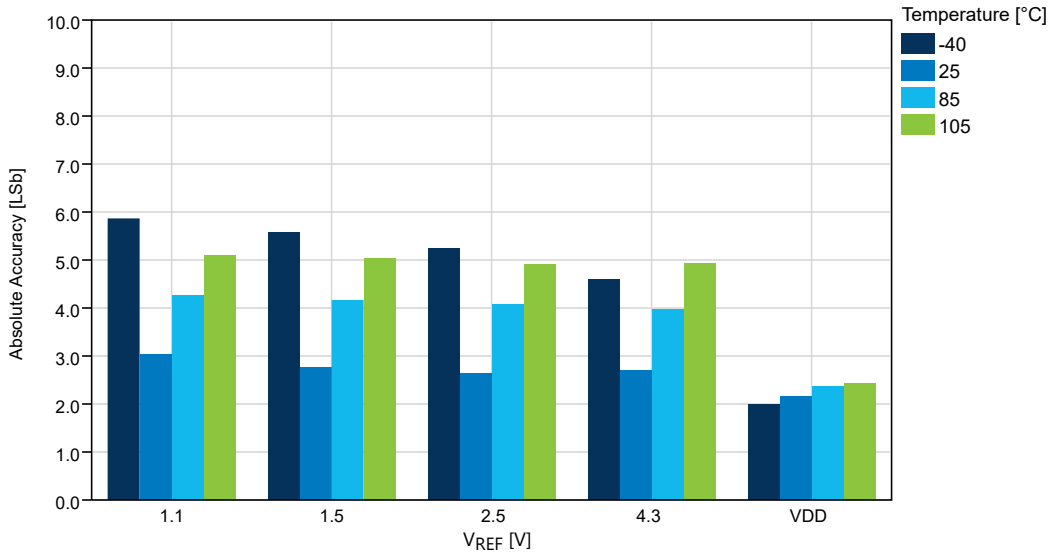


Figure 37-42. DNL Error vs. V_{DD} (f_{ADC} = 115 ksps) at T = 25°C, REFSEL = Internal Reference

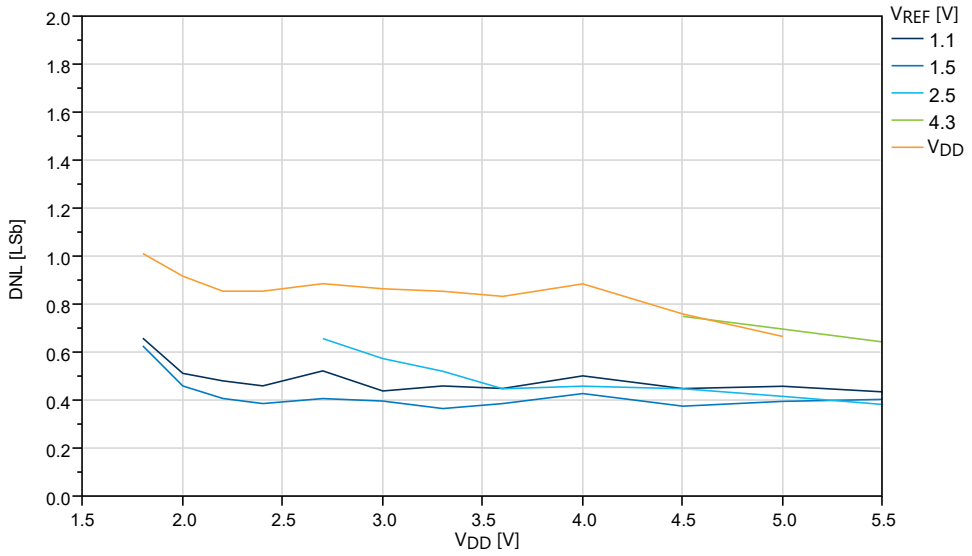


Figure 37-43. DNL vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps), REFSEL = Internal Reference

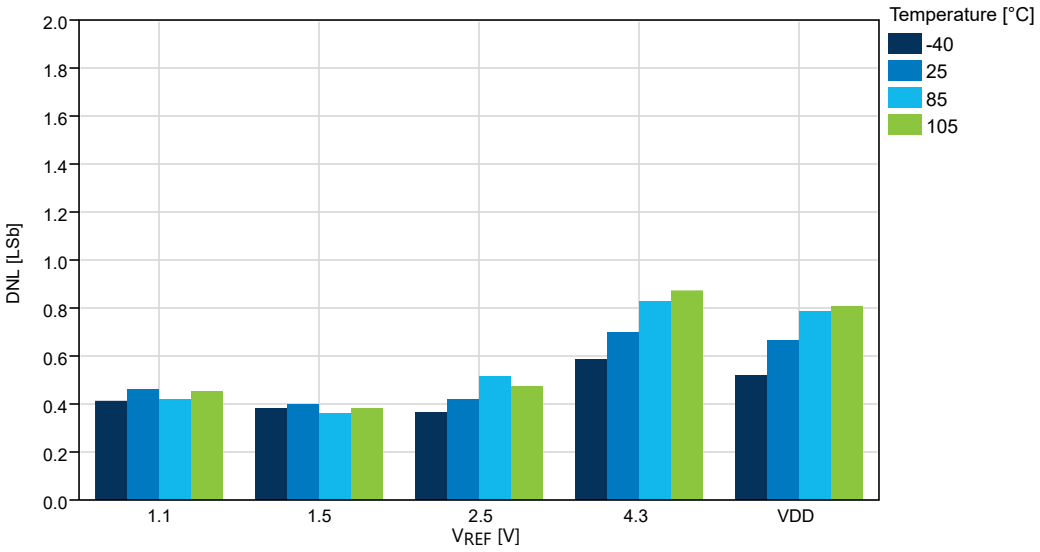


Figure 37-44. Gain Error vs. V_{DD} (f_{ADC} = 115 ksps) at T = 25°C, REFSEL = Internal Reference

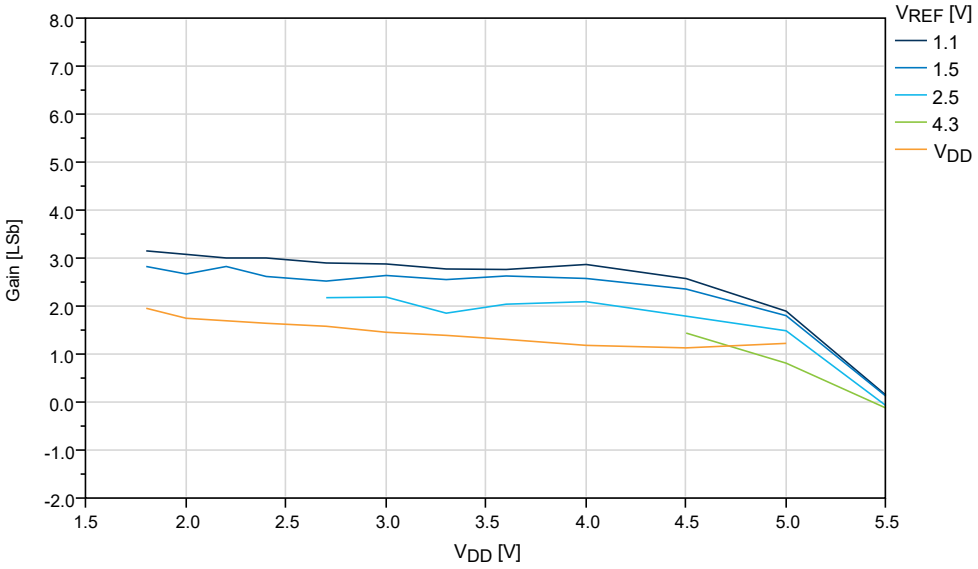


Figure 37-45. Gain Error vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps), REFSEL = Internal Reference

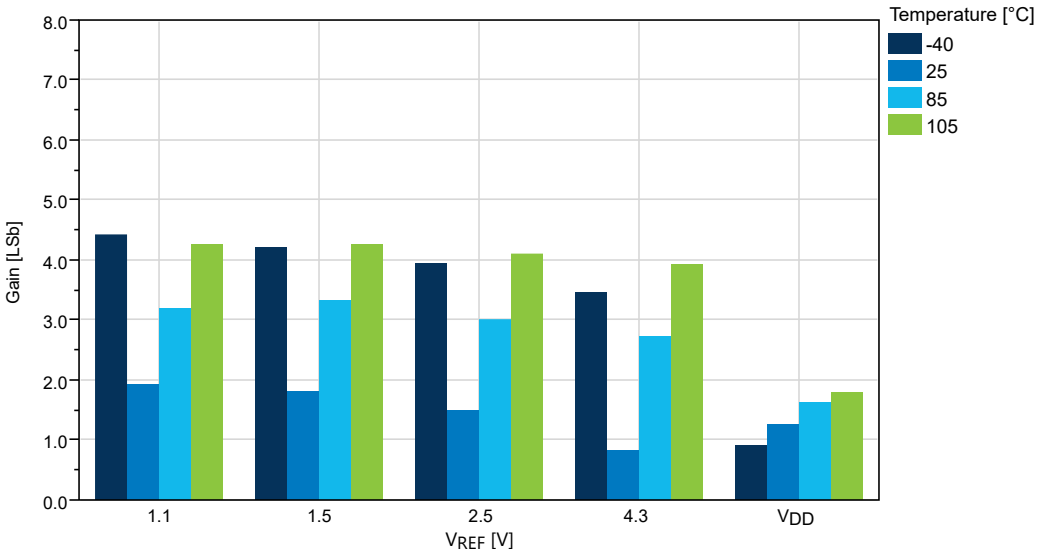


Figure 37-46. INL vs. V_{DD} (f_{ADC} = 115 ksps) at T = 25°C, REFSEL = Internal Reference

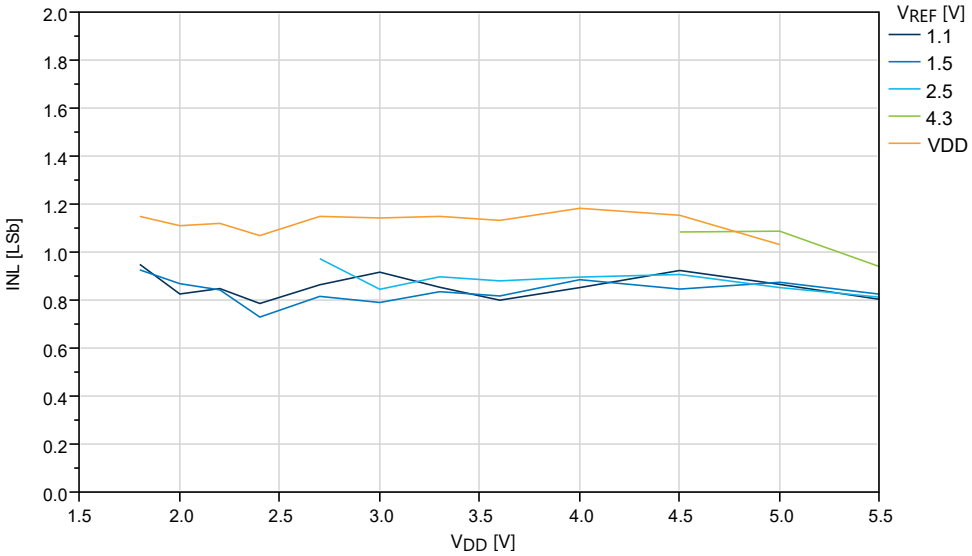


Figure 37-47. INL vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps), REFSEL = Internal Reference

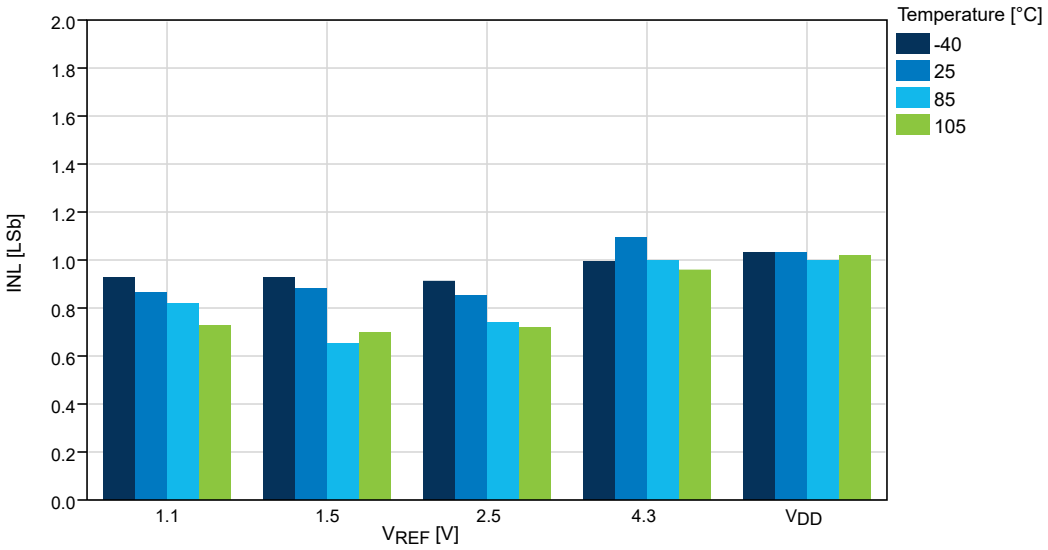


Figure 37-48. Offset Error vs. V_{DD} (f_{ADC} = 115 ksps) at T = 25°C, REFSEL = Internal Reference

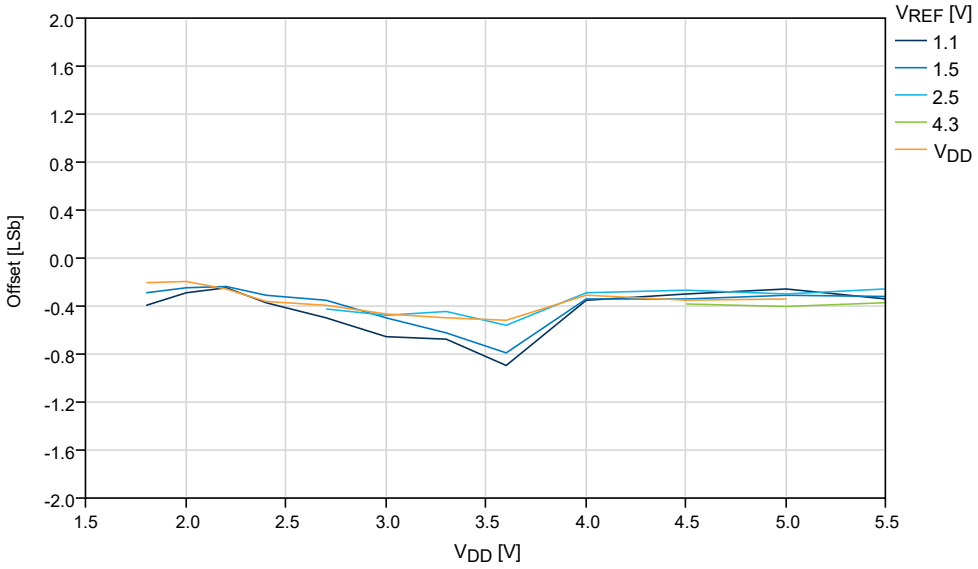


Figure 37-49. Offset Error vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps), REFSEL = Internal Reference

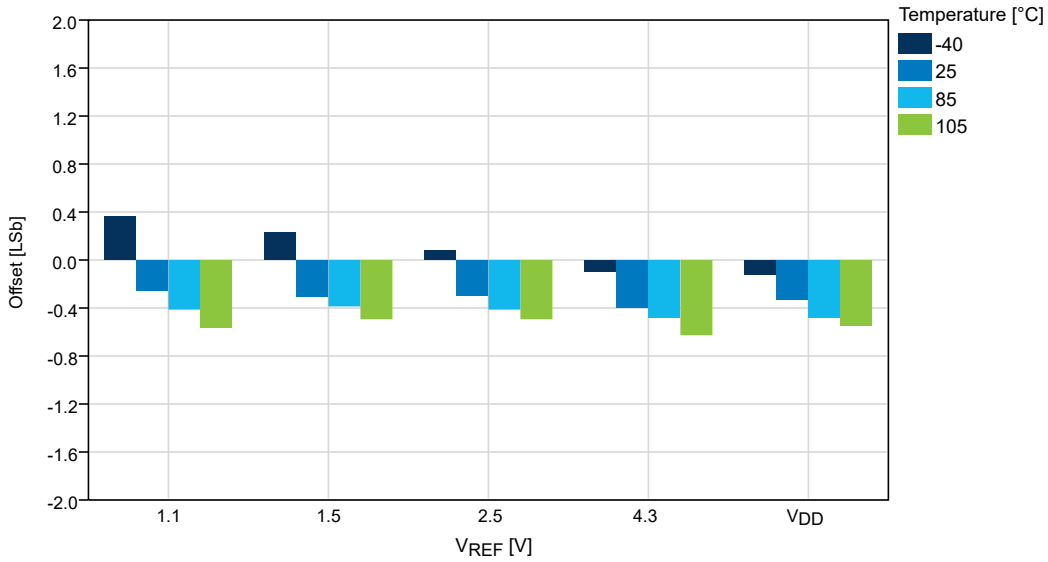


Figure 37-50. Absolute Accuracy vs. V_{DD} (f_{ADC} = 115 ksps, T = 25°C), REFSEL = External Reference

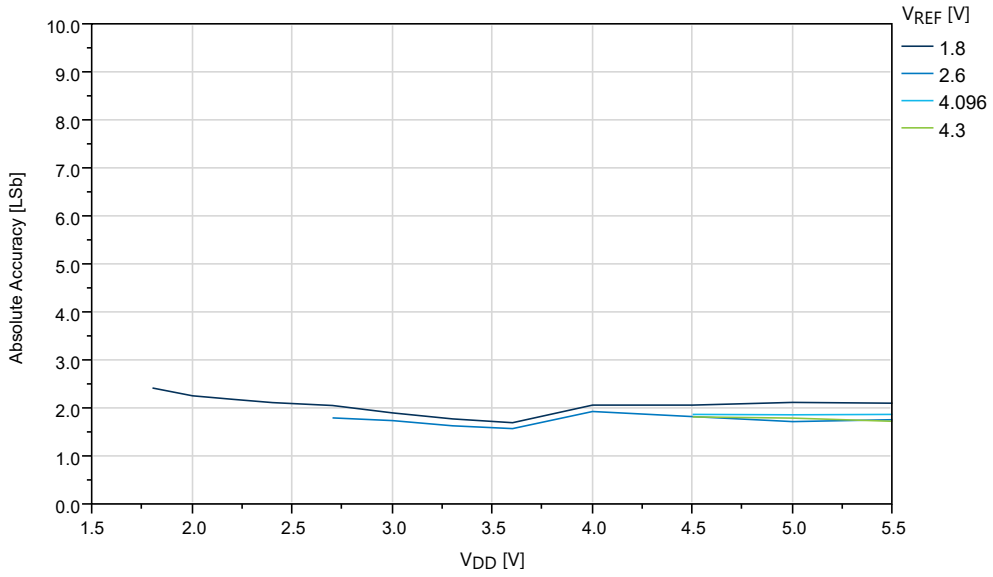


Figure 37-51. Absolute Accuracy vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps, REFSEL = External Reference)

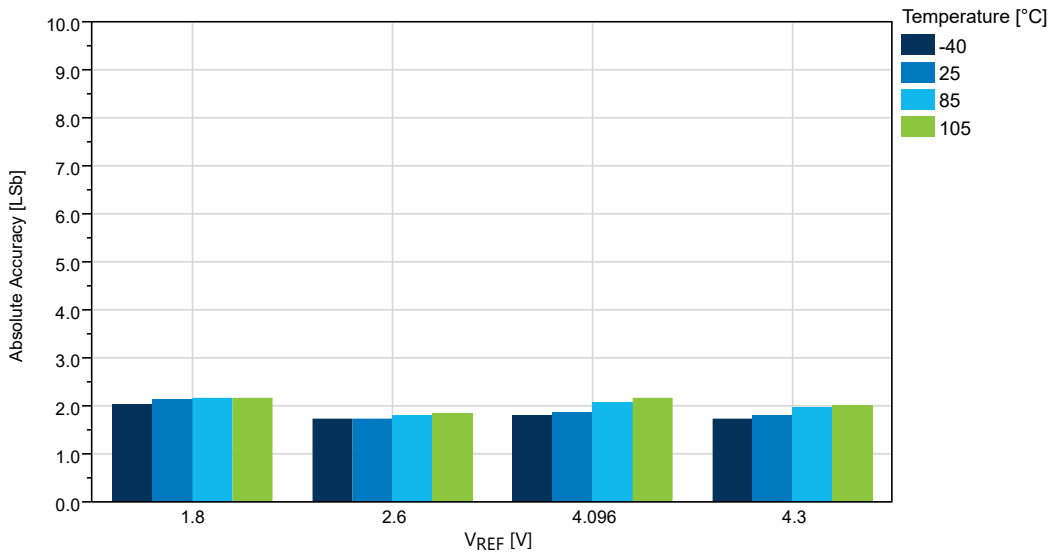


Figure 37-52. DNL vs. V_{DD} (f_{ADC} = 115 ksps, T = 25°C, REFSEL = External Reference)

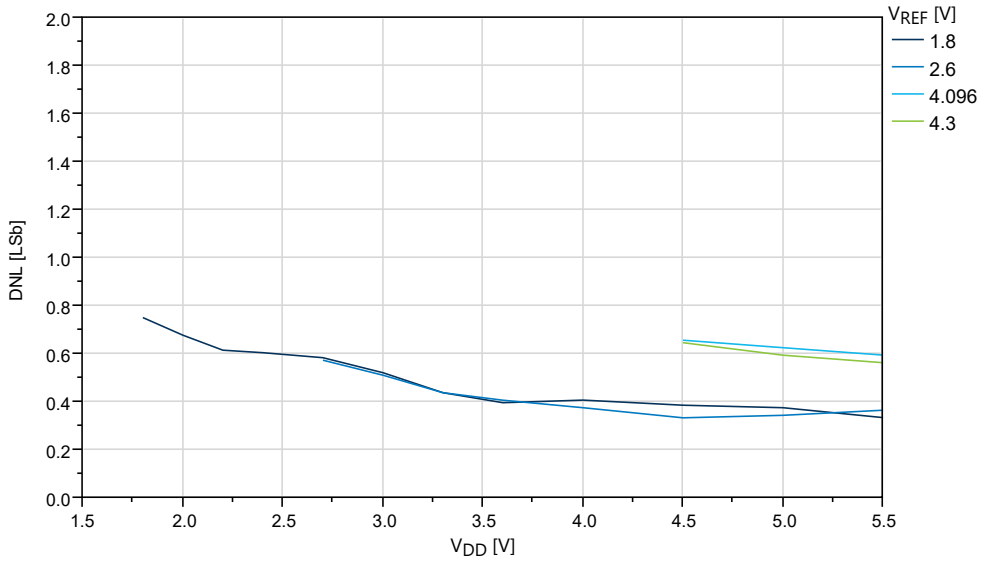


Figure 37-53. DNL vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps, REFSEL = External Reference)

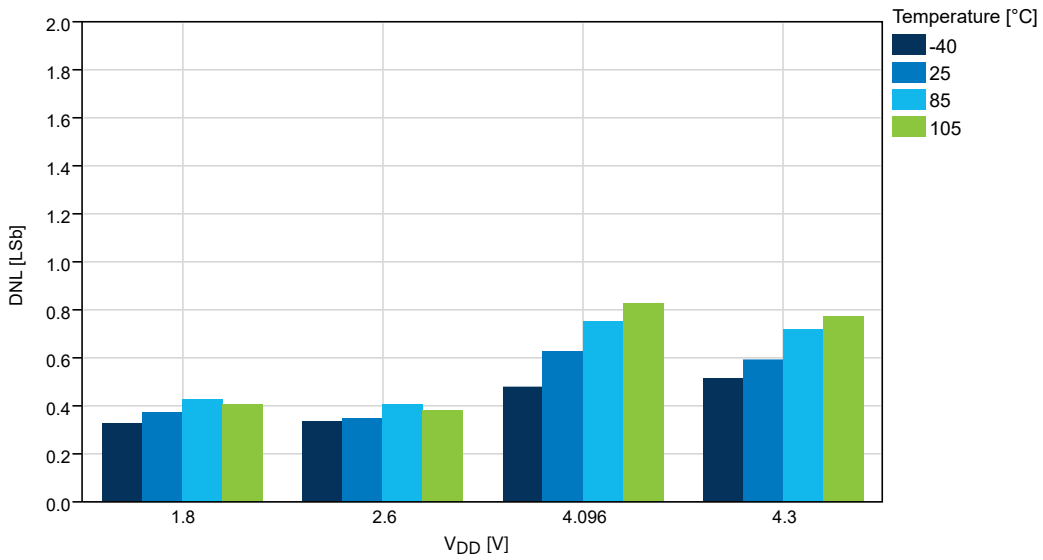


Figure 37-54. Gain vs. V_{DD} (f_{ADC} = 115 ksps, T = 25°C, REFSEL = External Reference)

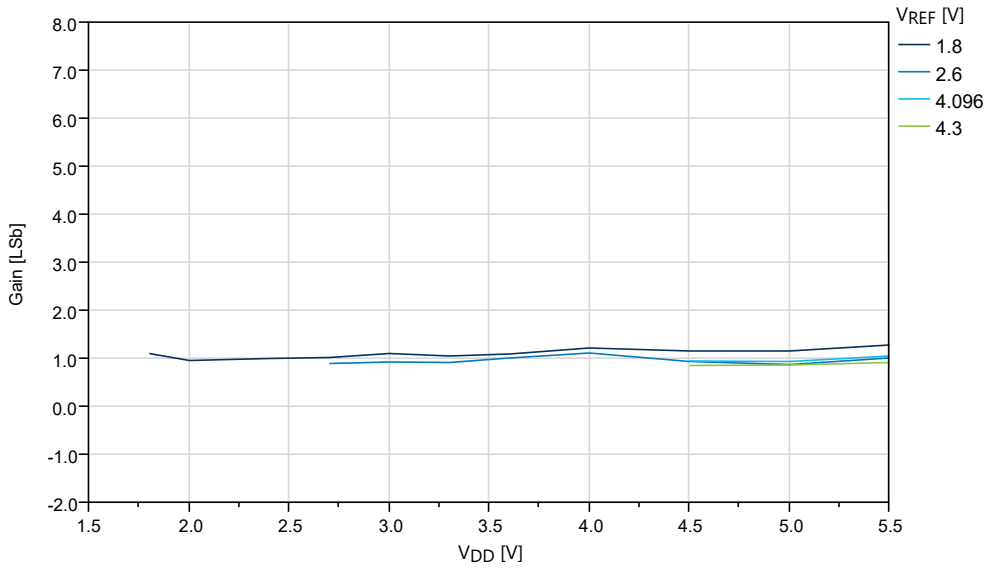


Figure 37-55. Gain vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps, REFSEL = External Reference)

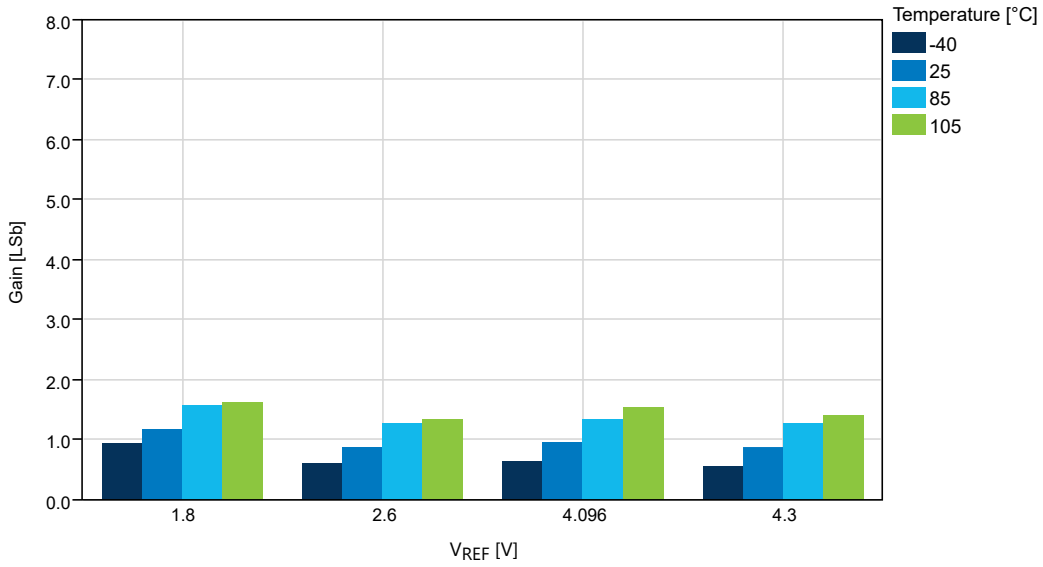


Figure 37-56. INL vs. V_{DD} (f_{ADC} = 115 ksps, T = 25°C, REFSEL = External Reference)

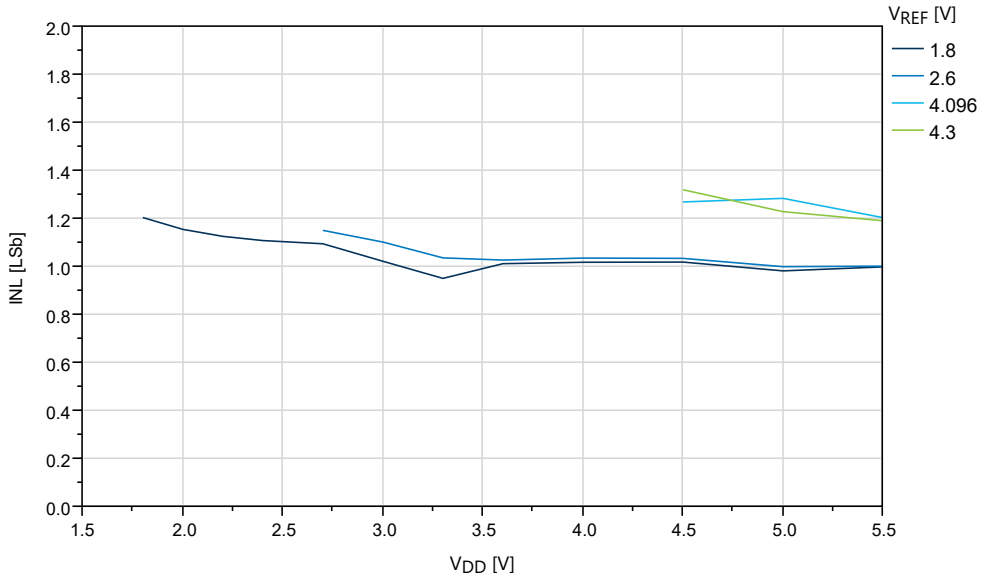


Figure 37-57. INL vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps, REFSEL = External Reference)

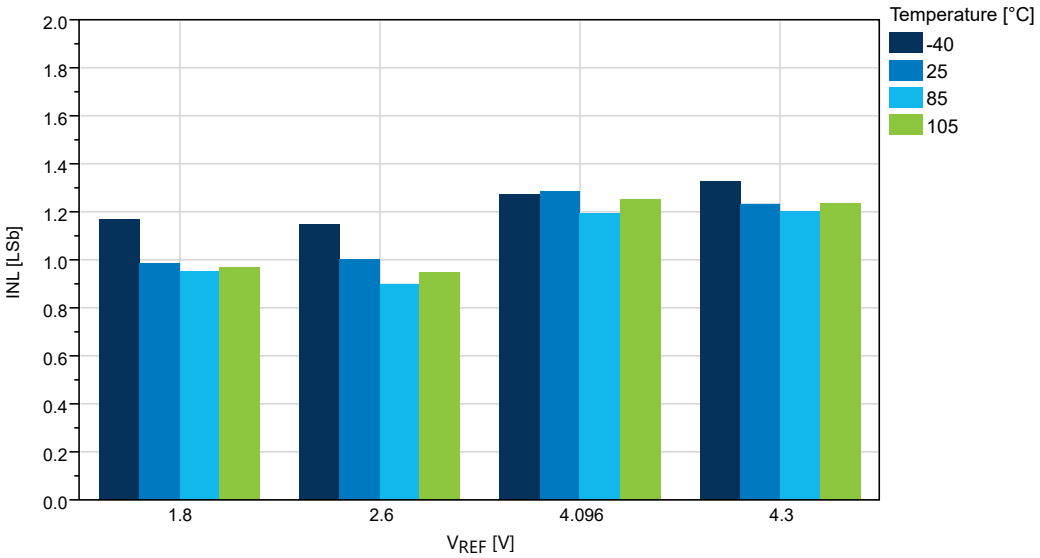


Figure 37-58. Offset vs. V_{DD} (f_{ADC} = 115 ksps, T = 25°C, REFSEL = External Reference)

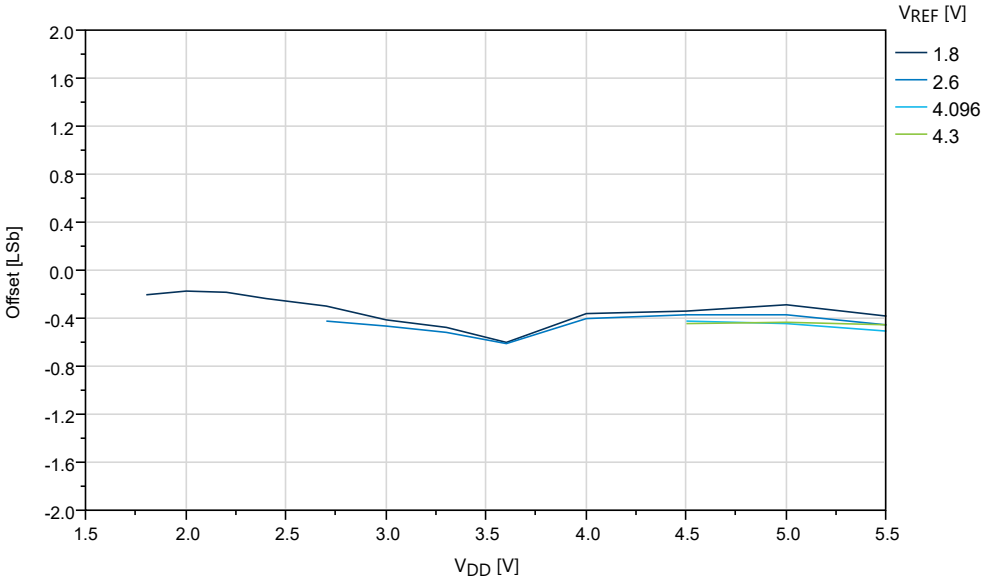
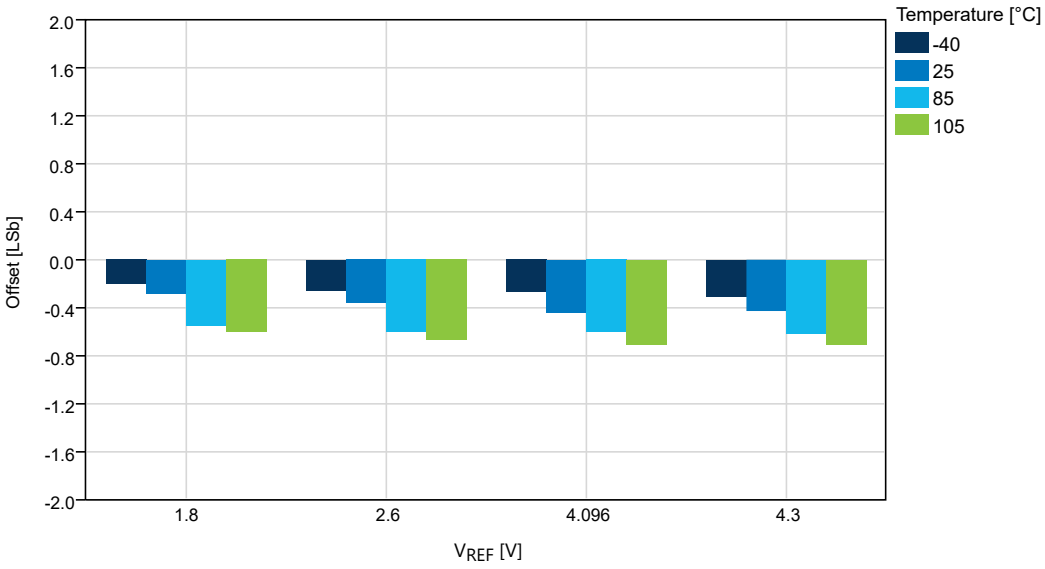
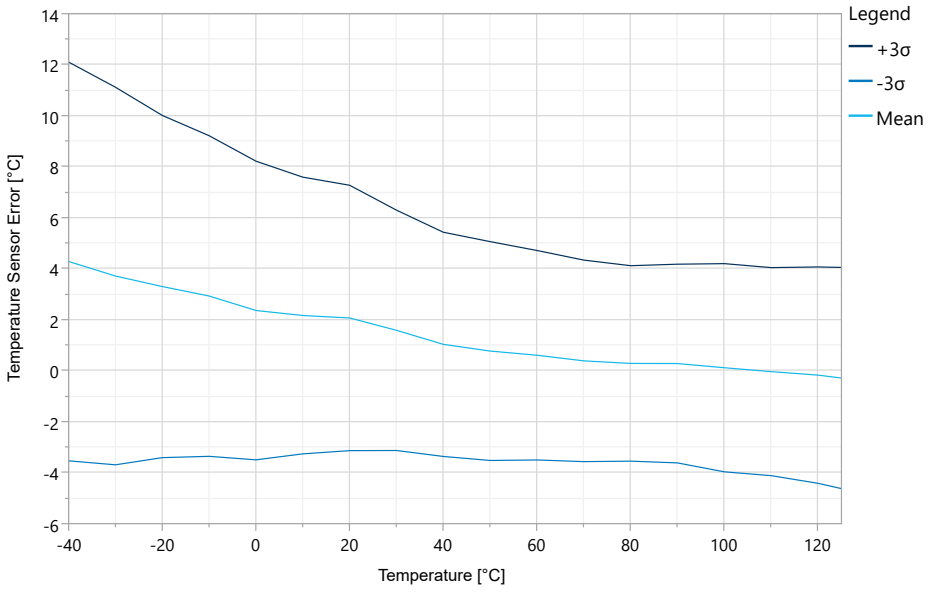


Figure 37-59. Offset vs. V_{REF} (V_{DD} = 5.0V, f_{ADC} = 115 ksps, REFSEL = External Reference)



37.6 TEMPSENSE Characteristics

Figure 37-60. Temperature Sensor Error vs. Temperature ±3σ



37.7 AC Characteristics

Figure 37-61. Hysteresis vs. V_{CM} - 10 mV (V_{DD} = 5V)

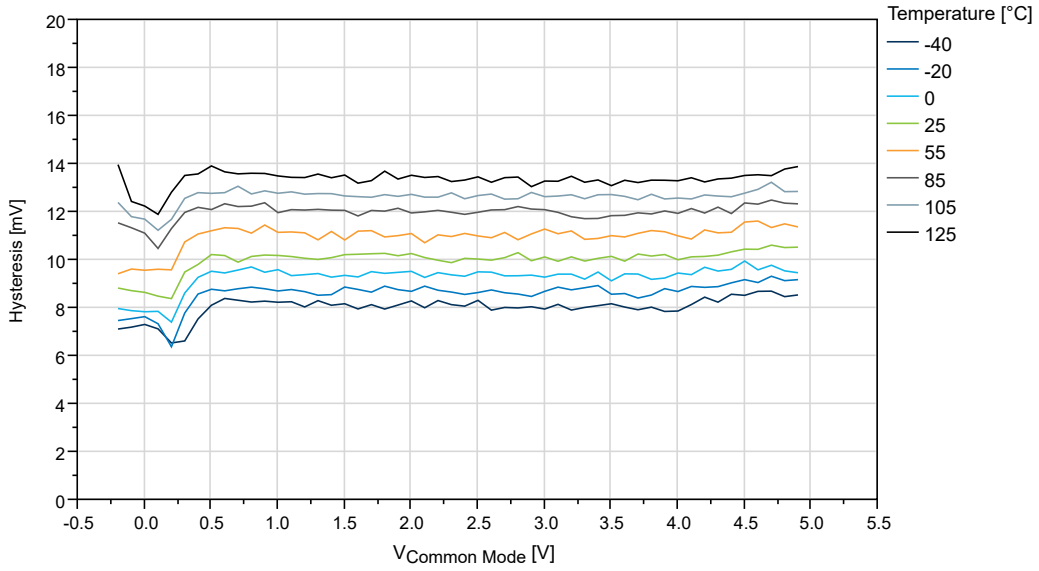


Figure 37-62. Hysteresis vs. V_{CM} - 10 mV to 50 mV (V_{DD} = 5V, T = 25°C)

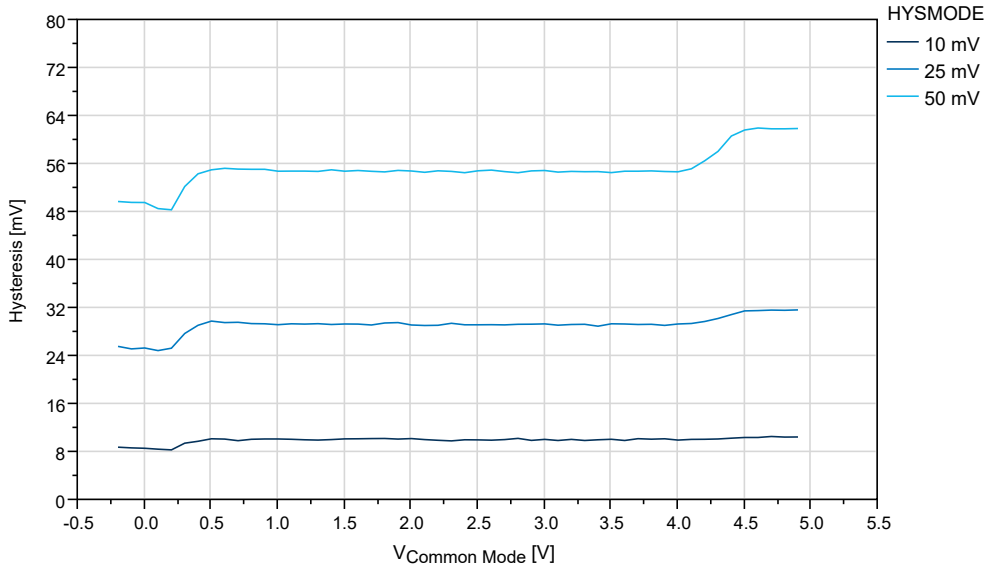


Figure 37-63. Offset vs. V_{CM} - 10 mV (V_{DD} = 5V)

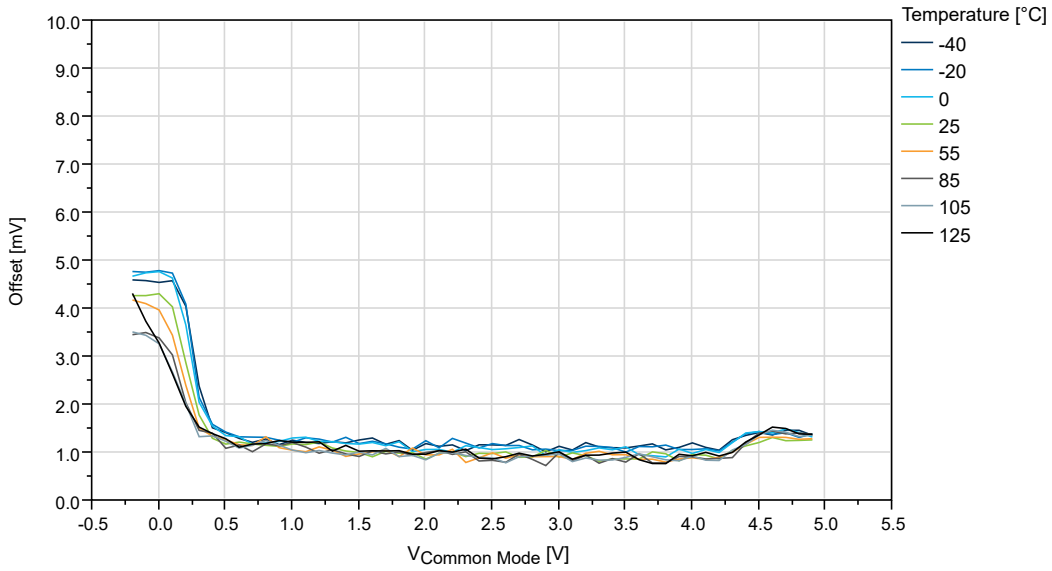


Figure 37-64. Offset vs. V_{CM} - 10 mV to 50 mV (V_{DD} = 5V, T = 25°C)

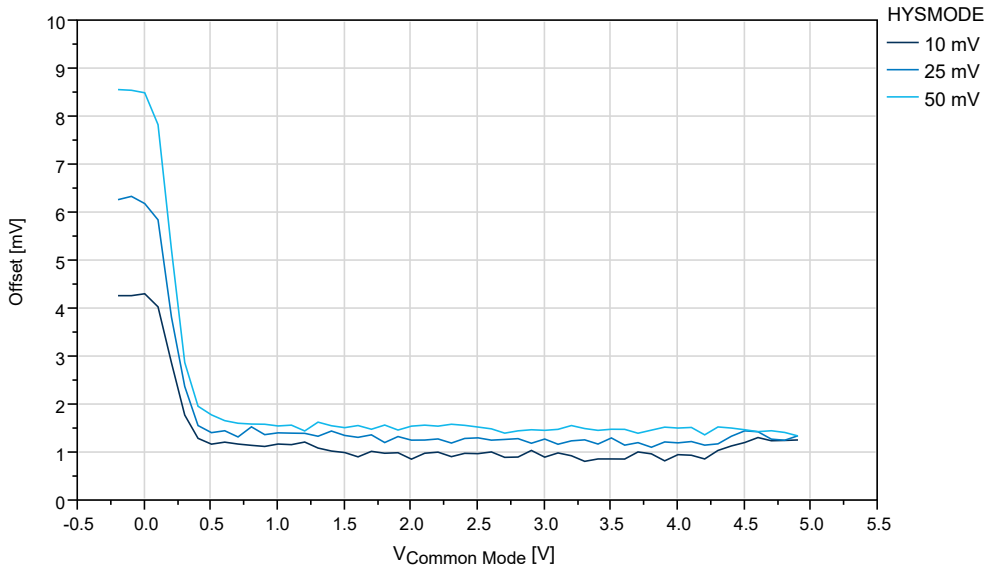


Figure 37-65. Propagation Delay vs. V_{CM} LPMODE Enabled, Falling Positive Input, V_{OD} = 25 mV (T = 25°C)

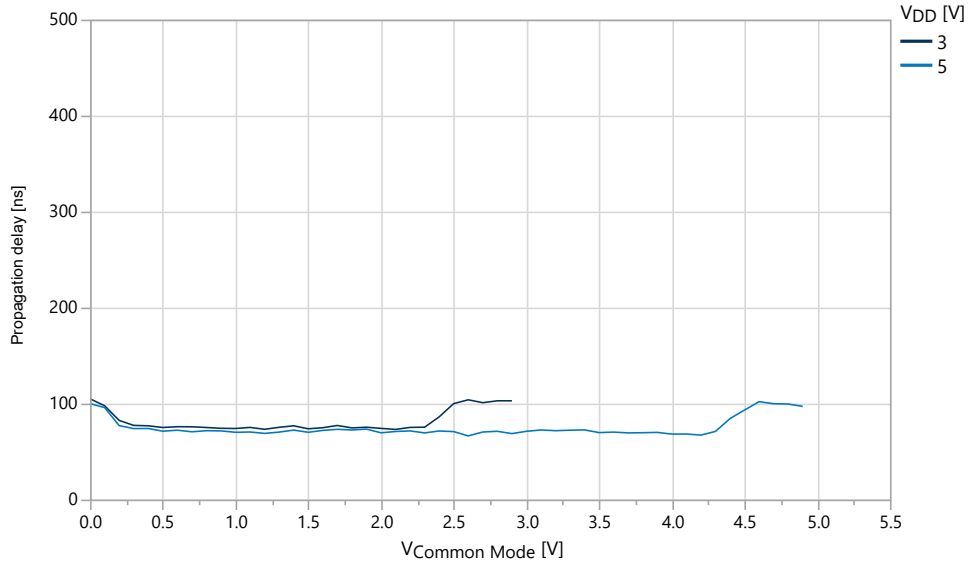


Figure 37-66. Propagation Delay vs. V_{CM} LPMODE Enabled, Rising Positive Input, V_{OD} = 30 mV (T = 25°C)

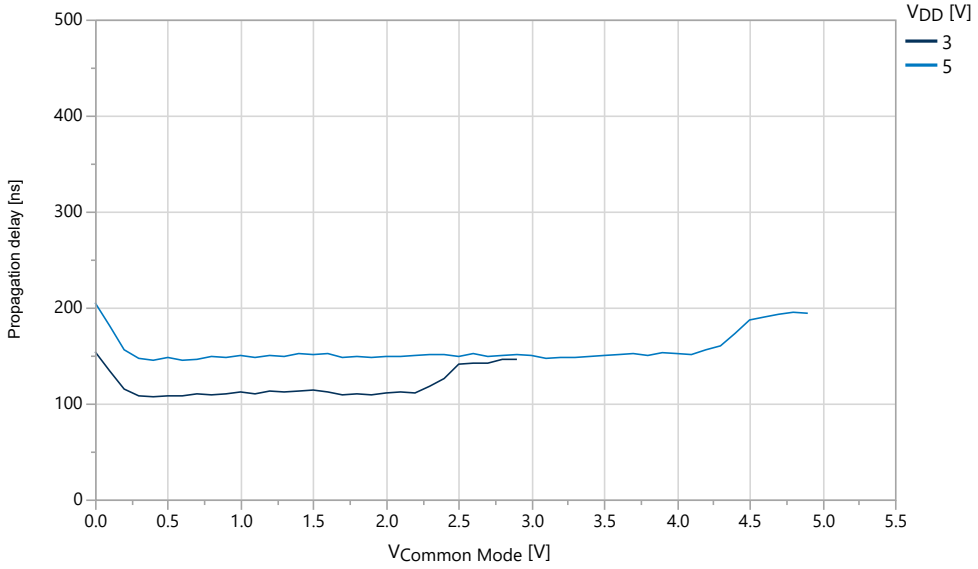


Figure 37-67. Propagation Delay vs. V_{CM} LPMODE Disabled, Falling Positive Input, V_{OD} = 30 mV (T = 25°C)

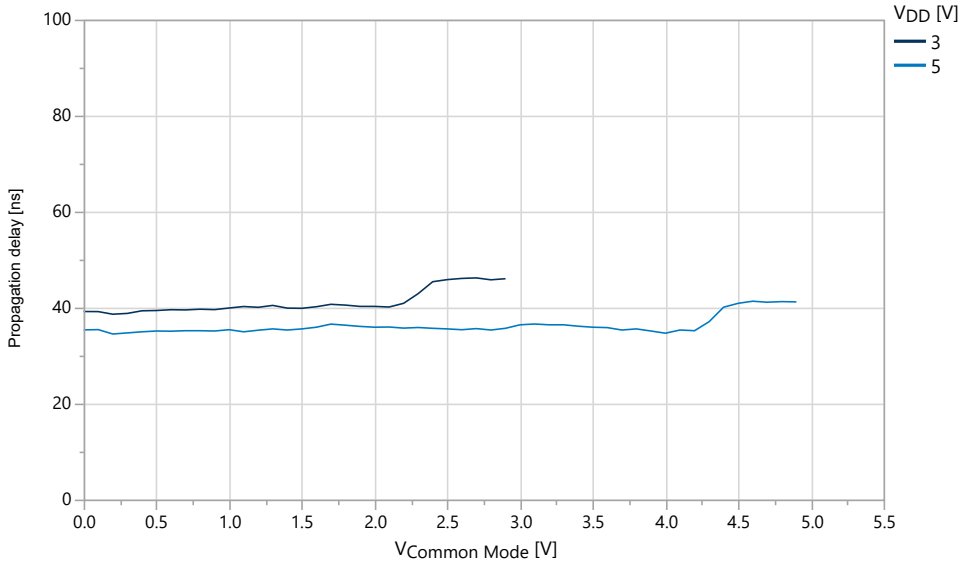
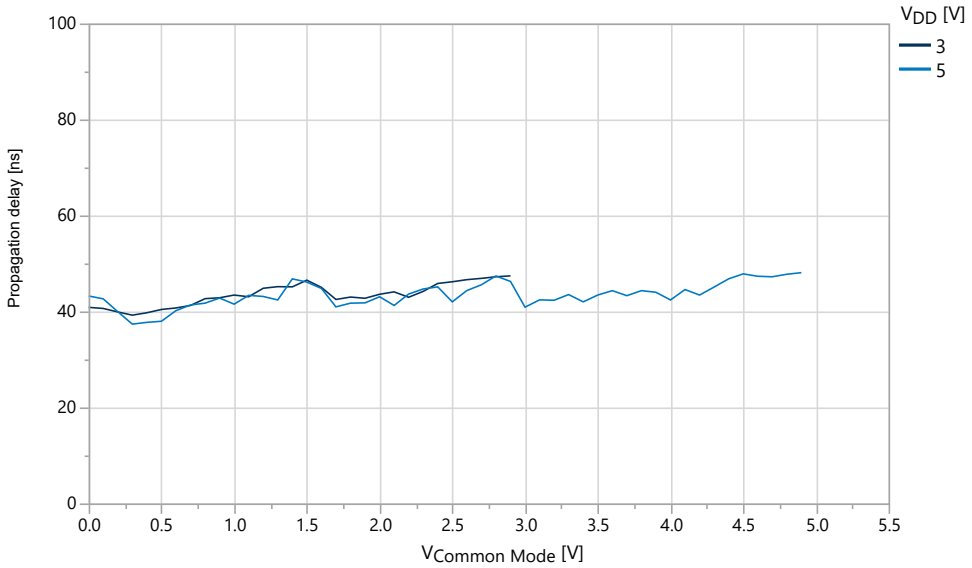


Figure 37-68. Propagation Delay vs. V_{CM} LPMODE Disabled, Rising Positive Input, V_{OD} = 30 mV (T = 25°C)



37.8 OSC20M Characteristics

Figure 37-69. OSC20M Internal Oscillator: Calibration Stepsize vs. Calibration Value (V_{DD} = 3V)

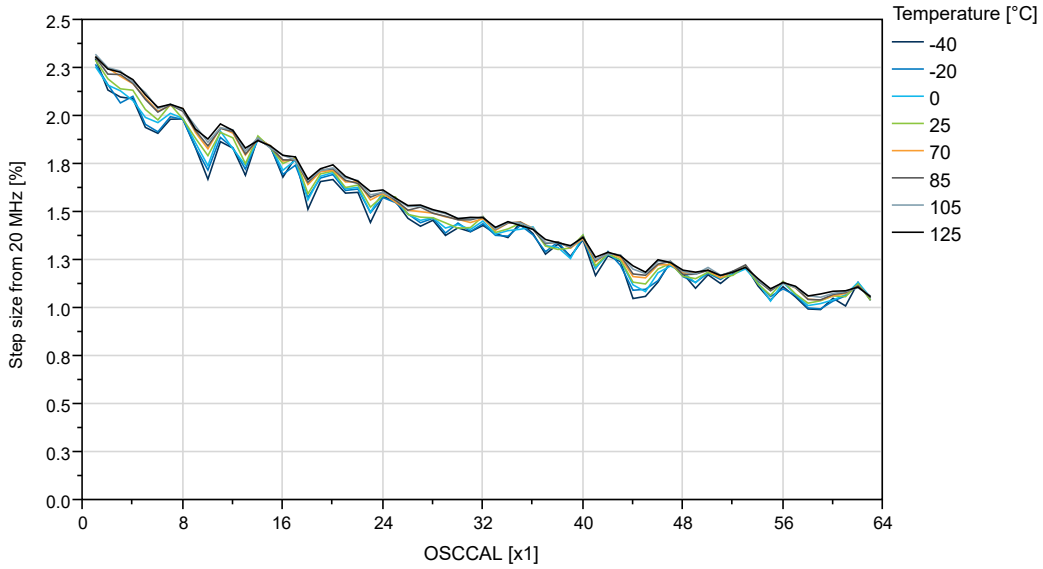


Figure 37-70. OSC20M Internal Oscillator: Frequency vs. Calibration Value (V_{DD} = 3V)

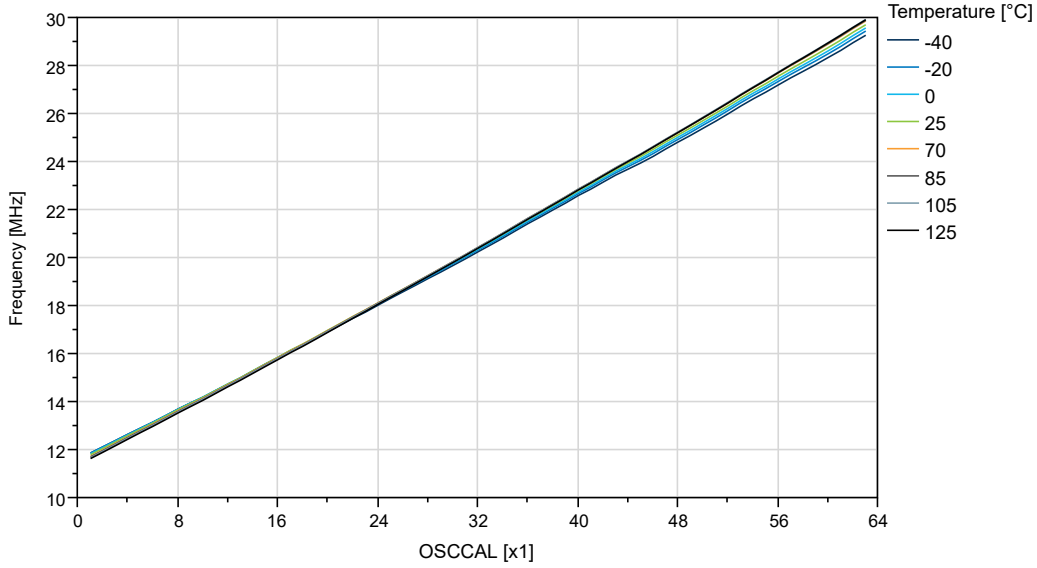


Figure 37-71. OSC20M Internal Oscillator: Frequency vs. Temperature

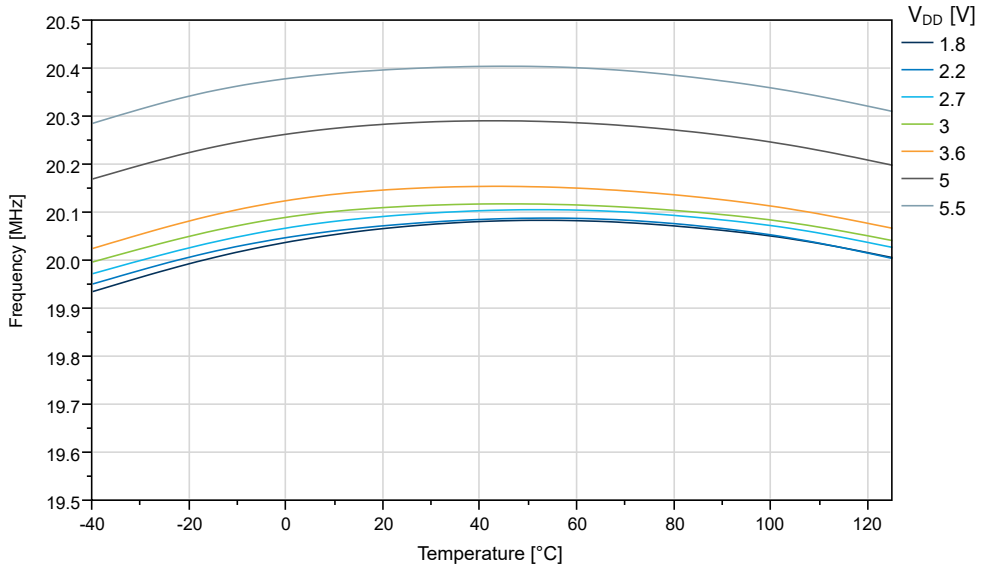
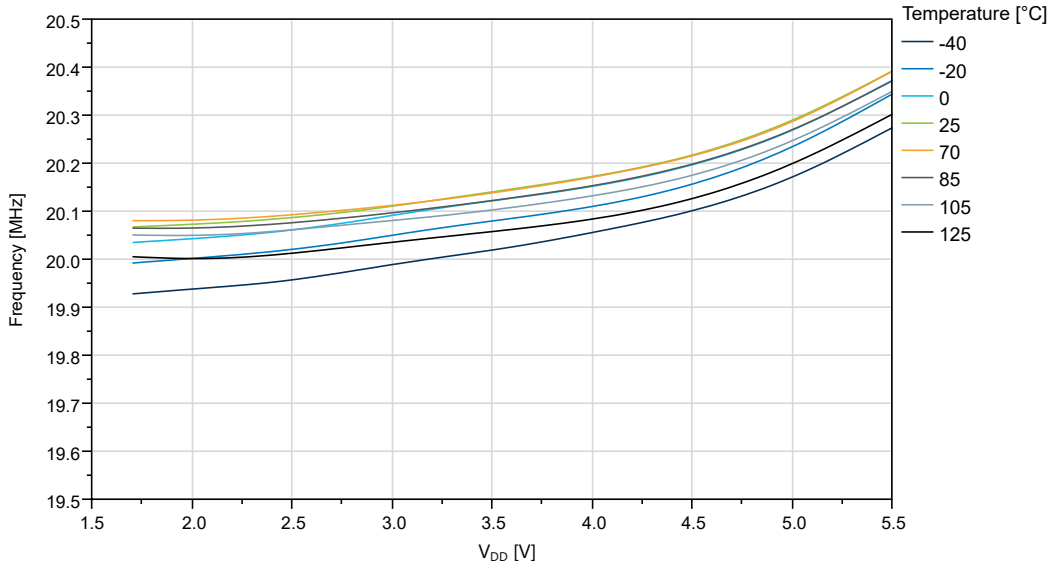


Figure 37-72. OSC20M Internal Oscillator: Frequency vs. V_{DD}



37.9 OSCULP32K Characteristics

Figure 37-73. OSCULP32K Internal Oscillator Frequency vs. Temperature

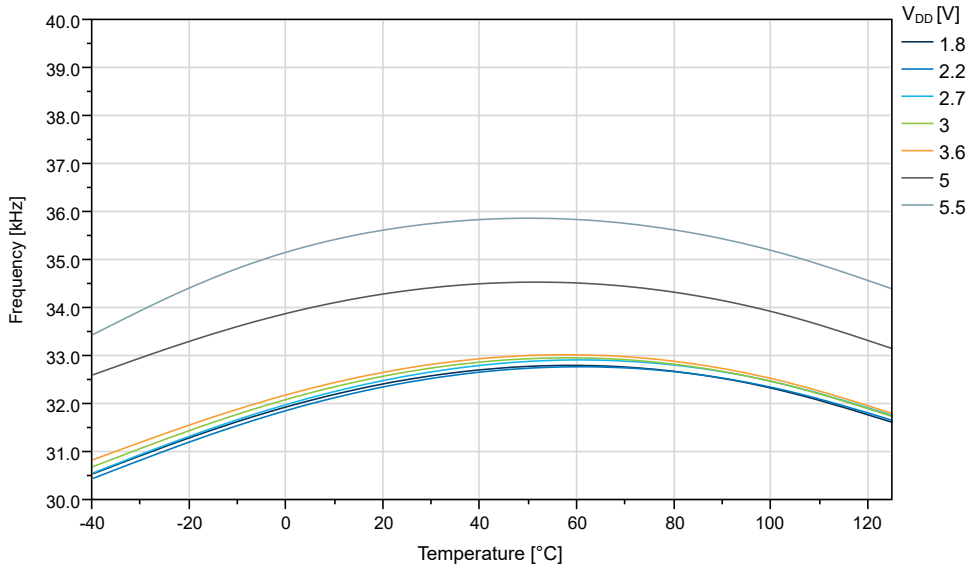
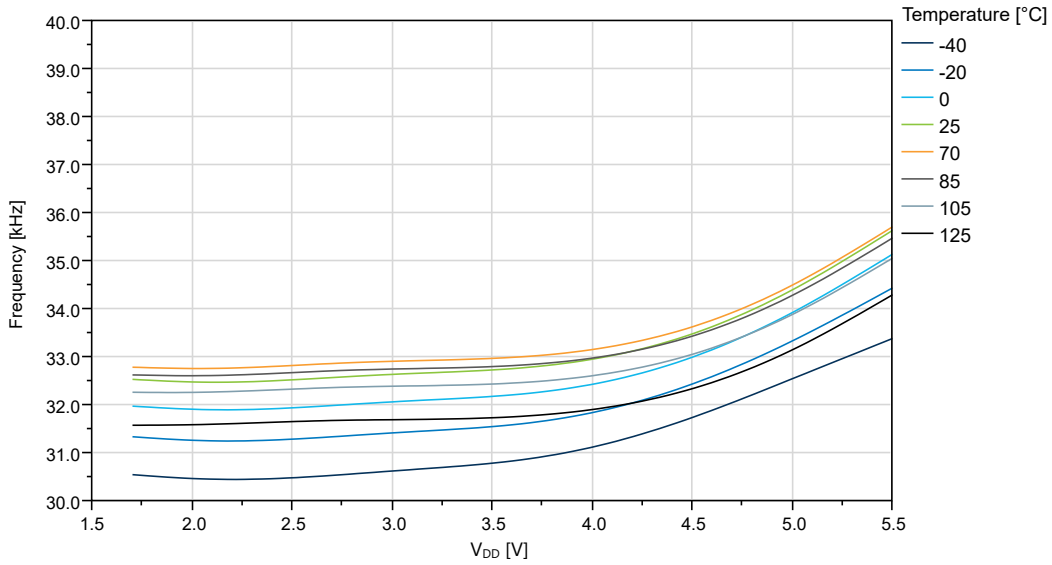
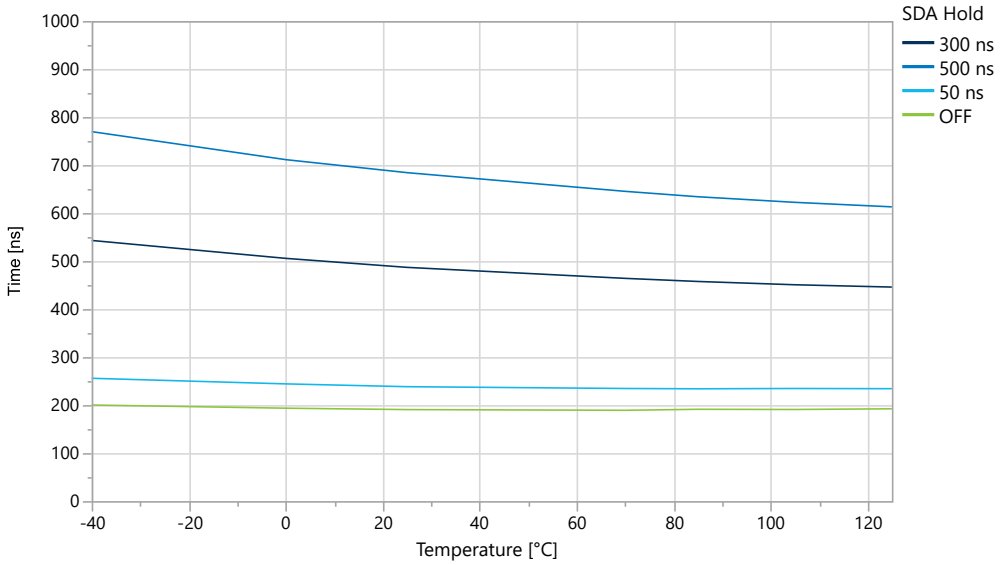


Figure 37-74. OSCULP32K Internal Oscillator Frequency vs. V_{DD}



37.10 TWI SDA Hold Timing

Figure 37-75. TWI SDA Hold Time vs. Temperature (Slave Mode) V_{DD} = 3V, CLK_{CPU} = 10 MHz



38. Ordering Information

- Available ordering options can be found by:
  - Clicking on one of the following product page links:
    - ATtiny3216 Product Page
    - ATtiny3217 Product Page
  - Searching by product name at microchipdirect.com
  - Contacting the local sales representative

38.1 Product Information

Note: For the latest information on available ordering codes, refer to the *ATtiny3216/3217 Silicon Errata and Data Sheet Clarification* (www.microchip.com/DS80000887) found on the product page.

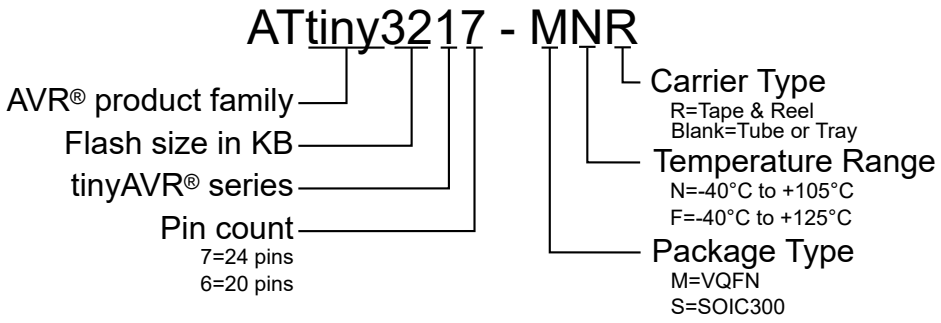
Ordering Code[1]	Flash/SRAM	Pin Count	Max. CPU Speed	Supply Voltage	Package Type[2,3]	Temperature Range
ATtiny3216-SNR	32 KB/2 KB	20	20 MHz	1.8V to 5.5V	SOIC	-40°C to +105°C
ATtiny3216-SN	32 KB/2 KB	20	20 MHz	1.8V to 5.5V	SOIC	-40°C to +105°C
ATtiny3216-SFR	32 KB/2 KB	20	16 MHz	2.7V to 5.5V	SOIC	-40°C to +125°C
ATtiny3216-SF	32 KB/2 KB	20	16 MHz	2.7V to 5.5V	SOIC	-40°C to +125°C
ATtiny3217-MNR	32 KB/2 KB	24	20 MHz	1.8V to 5.5V	VQFN	-40°C to +105°C
ATtiny3217-MN	32 KB/2 KB	24	20 MHz	1.8V to 5.5V	VQFN	-40°C to +105°C
ATtiny3217-MFR	32 KB/2 KB	24	16 MHz	2.7V to 5.5V	VQFN	-40°C to +125°C
ATtiny3217-MF	32 KB/2 KB	24	16 MHz	2.7V to 5.5V	VQFN	-40°C to +125°C

Note:

1. Pb-free packaging complies with the European Directive for Restriction of Hazardous Substances (RoHS directive). It is also Halide free and fully Green.
2. Available in Tape & Reel, Tube or Tray packing media.
3. Package outline drawings can be found in section 39. Package Drawings.

38.2 Product Identification System

To order or obtain information, for example, on pricing or delivery, refer to the factory or the listed sales office.



39. Package Drawings

39.1 Online Package Drawings

For the most recent package drawings:

1. Go to http://www.microchip.com/packaging.
2. Go to the package type-specific page, for example, VQFN.
3. Search for Drawing Number and Style to find the most recent package drawing.

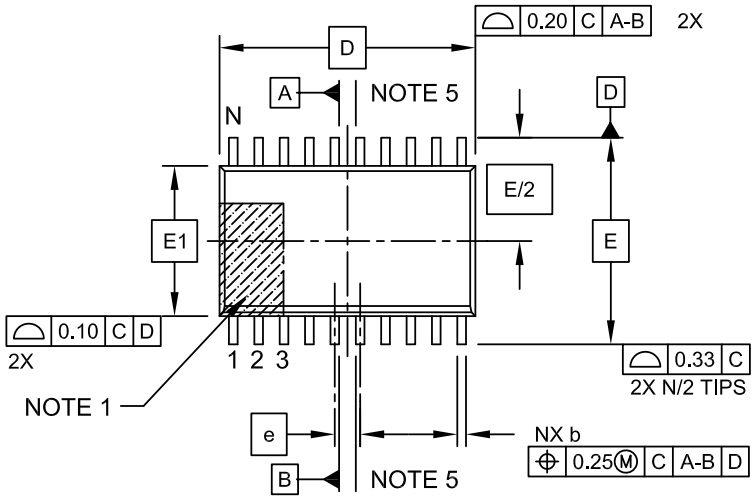
Table 39-1. Drawing Numbers

Pin Count	Package Type	Drawing Number	Style
20	SOIC	C04-00094	SO
24	VQFN	C04-21386	RLB

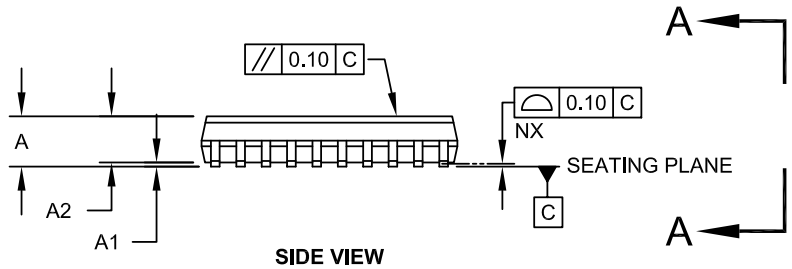
39.2 20-Pin SOIC

20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

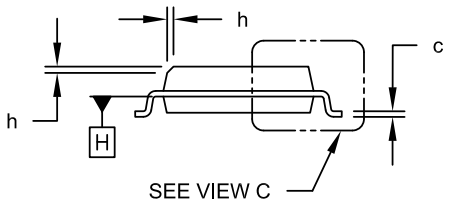
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW



SIDE VIEW

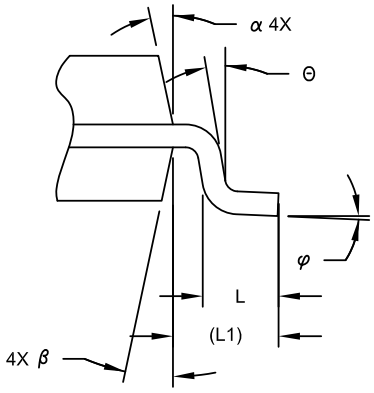


VIEW A-A

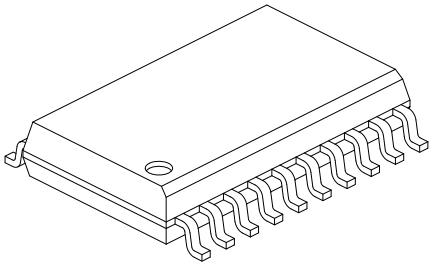
Microchip Technology Drawing C04-094C Sheet 1 of 2

20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



VIEW C



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		20	
Pitch	e		1.27 BSC	
Overall Height	A	-	-	2.65
Molded Package Thickness	A2	2.05	-	-
Standoff §	A1	0.10	-	0.30
Overall Width	E		10.30 BSC	
Molded Package Width	E1		7.50 BSC	
Overall Length	D		12.80 BSC	
Chamfer (Optional)	h	0.25	-	0.75
Foot Length	L	0.40	-	1.27
Footprint	L1		1.40 REF	
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.20	-	0.33
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

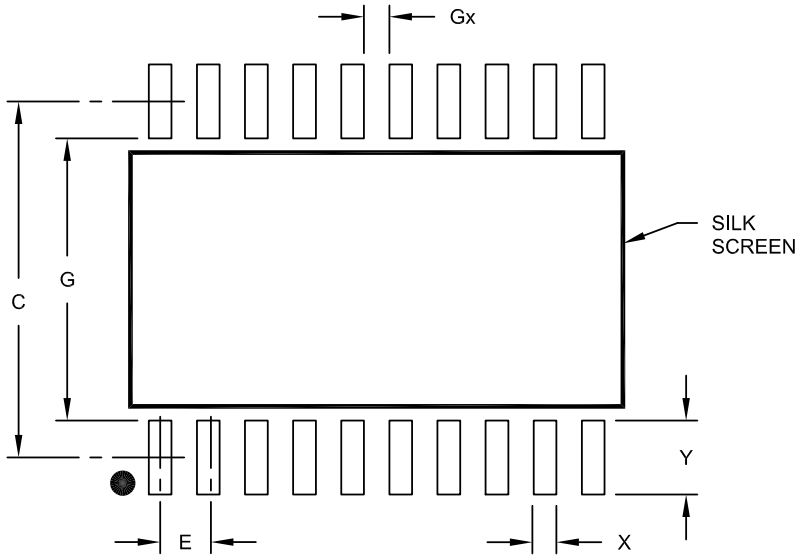
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm 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.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-094C Sheet 2 of 2

20-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

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	1.27 BSC		
Contact Pad Spacing	C		9.40	
Contact Pad Width (X20)	X			0.60
Contact Pad Length (X20)	Y			1.95
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	7.45		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2094A

Table 39-2. Device and Package Maximum Weight

Maximum Weight	542 mg

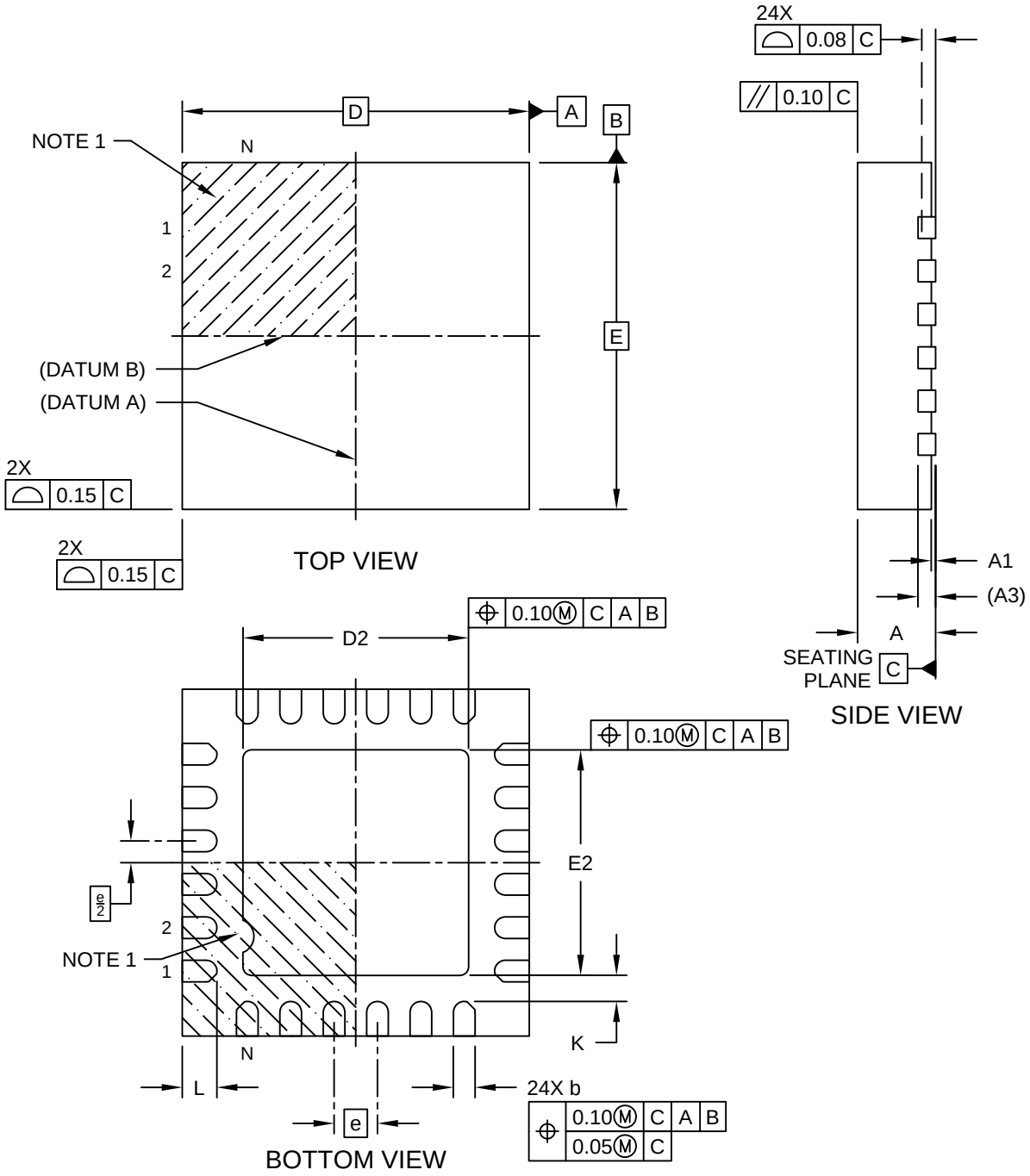
Table 39-3. Package Reference

JEDEC Drawing Reference	N/A
JESD97 Classification	E3

39.3 24-Pin VQFN

24-Lead Very Thin Plastic Quad Flat, No Lead Package (RLB) - 4x4 mm Body [VQFN] Atmel Legacy Global Package Code ZHA

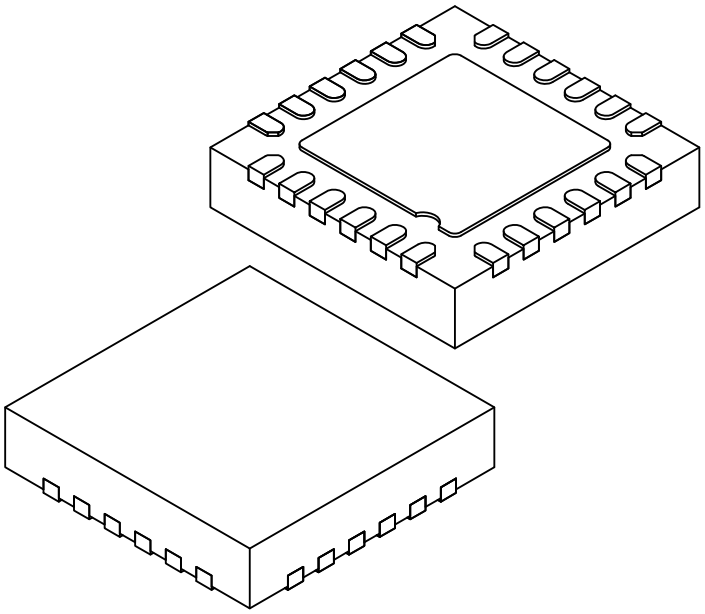
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-21386 Rev A Sheet 1 of 2

24-Lead Very Thin Plastic Quad Flat, No Lead Package (RLB) - 4x4 mm Body [VQFN]
Atmel Legacy Global Package Code ZHA

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		24	
Pitch	e		0.50 BSC	
Overall Height	A	0.80	0.85	0.90
Standoff	A1	0.00	-	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		4.00 BSC	
Exposed Pad Length	D2	2.45	2.60	2.75
Overall Width	E		4.00 BSC	
Exposed Pad Width	E2	2.45	2.60	2.75
Terminal Width	b	0.18	0.25	0.30
Terminal Length	L	0.35	0.40	0.45
Terminal-to-Exposed-Pad	K	0.20	-	-

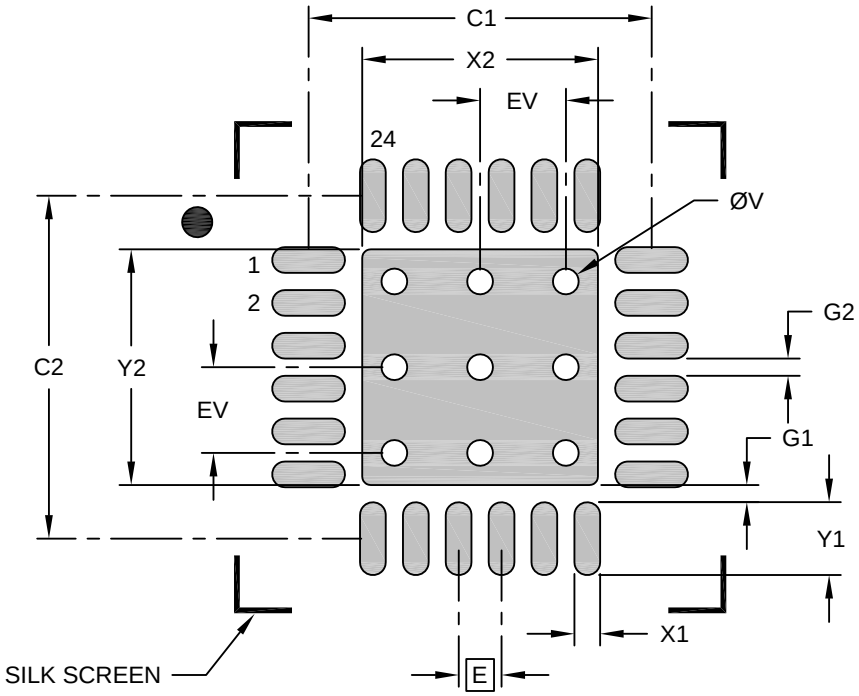
Notes:

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 C04-21386 Rev A Sheet 2 of 2

24-Lead Very Thin Plastic Quad Flat, No Lead Package (RLB) - 4x4 mm Body [VQFN] Atmel Legacy Global Package Code ZHA

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			2.75
Optional Center Pad Length	Y2			2.75
Contact Pad Spacing	C1		4.00	
Contact Pad Spacing	C2		4.00	
Contact Pad Width (X24)	X1			0.30
Contact Pad Length (X24)	Y1			0.85
Contact Pad to Center Pad (X24)	G1	0.20		
Contact Pad to Contact Pad (X20)	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 C04-23386 Rev A

Table 39-4. Device and Package Maximum Weight

Maximum Weight	41.4 mg

Table 39-5. Package Reference

JEDEC Drawing Reference	N/A
JESD97 Classification	E3

39.4 Thermal Considerations

39.4.1 Thermal Resistance Data

The following table summarizes the thermal resistance data depending on the package.

Table 39-6. Thermal Resistance Data

Pin Count	Package Type	θ_{JA} [°C/W]	θ_{JC} [°C/W]
20	SOIC	44	21
24	VQFN	60.6	25

39.4.2 Junction Temperature

The average chip-junction temperature, T_J, in °C, can be obtained from the following equations:

- **Equation 1**: $T_J = T_A + (P_D \times \theta_{JA})$
- **Equation 2**: $T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$

where:

- θ_{JA} = Package thermal resistance, Junction-to-ambient (°C/W), see Table 39-6
- θ_{JC} = Package thermal resistance, Junction-to-case thermal resistance (°C/W), see Table 39-6
- $\theta_{HEATSINK}$ = Thermal resistance (°C/W) specification of the external cooling device
- P_D = Device power consumption (W)
- T_A = Ambient temperature (°C)

From the first equation, the user can derive the estimated lifetime of the chip and decide whether a cooling device is necessary or not. If a cooling device has to be fitted on the chip, the second equation must be used to compute the resulting average chip-junction temperature T_J in °C.

40. Errata

40.1 Errata - ATtiny3216/3217

Errata can be found in the *ATtiny3216/3217 Silicon Errata and Data Sheet Clarification* (www.microchip.com/DS80000887).

41. 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).

41.1 Rev. A - 05/2020

Section	Changes
Document	• Initial document release The content for the devices described in this document has been restructured from: • ATtiny1614 Data Sheet • ATtiny1616/3216 Data Sheet • ATtiny1617/3217 Data Sheet to: • ATtiny1614/1616/1617 Data Sheet • **ATtiny3216/3217 Data Sheet (this document)** Refer to 41.2 Appendix - Obsolete Revision History for further details. The following items are referring to changes between the latest revisions of the obsolete documents and this document: • Updated the document to Microchip editing standard • Removed related links • Removed the *Acronyms and Abbreviations* section • Removed the content of the *Instruction Set Summary* section. This section now refers to the external Instruction Set Manual instead. • Removed device-specific information from peripheral sections • Restructured sections related to system dependencies within the peripheral sections

.....continued

Section	Changes
Device	• Device-specific information restructured/changed to comply with the devices documented in this document – Features – Pinout – I/O Multiplexing and Considerations – Ordering Information – Package Drawings • Pinout diagrams updated: – *20-Pin SOIC* – *24-Pin VQFN* • Memories – *Memory Map* figure updated – *Memory Access (FUSE.LOCKBIT Invalid Key)* table updated – Documentation about GPIOR added • Peripherals and Architecture – Peripheral Touch Controller (PTC) added in *Peripheral Address Match* table – *Interrupt Vector Mapping* table updated • *Base Address* column renamed to *Program Address (word)* • *Peripheral Source* column cleaned up • *Description* column added • Package Drawings – Updated the *Drawing Numbers* table – Removed MSL numbers – *Thermal Considerations* section moved inside *Package Drawings* section
AVR CPU	• Missing information about Flash Offset Address (0x8000) added (1) • Removed duplicate information after the *AVR CPU architecture* • Emphasized that the Arithmetic Logic Unit (ALU) is doing its operations against working registers in the register file • Added *Stack Pointer Instructions* table • Restructured and improved documentation in: – The *Register File* section – The *X-, Y-, and Z-Registers* section – The *Accessing 16-bit Registers* section • Added *On-Chip Debug Capabilities* section • Updated bit names in the Status Register (SREG): – From *Bit Copy Storage* to *Transfer Bit* – From *Sign Bit* to *Sign Flag*
NVMCTRL	• *NVMCTRL Block Diagram* figure updated • Missing *Flash Sections* figure(1) added • *Write Access After Reset* section added
CLKCTRL	• Additional documentation on CLK_TCD added
SLPCTRL	• *Sleep Mode Activity Overview* table updated • *Debug Operations*section added

.....continued

Section	Changes
RSTCTRL	• Figures added: – *MCU Start-up, $\overline{RESET}$ Tied to V_{DD}* – *MCU Start-up, $\overline{RESET}$ Extended Externally* – *Brow-out Detection Reset* – *Exernal Reset Characteristics*
CPUINT	• *Minimum Interrupt Response Time* table added • General improvement of the documentation and its structure
EVSYS	• Register names updated – From ASYNCCH to ASYNCCHn – From SYNCCH to SYNCCHn – From ASYNCUSER to ASYNCUSERn – From SYNCUSER to SYNCUSERn
PORT	• Block diagram updated • *Asynchronous Sensing Pin Properties section* added • *Debug Operations* section added • *Event Generators in PORTx* table added • General improvement of the documentation and its structure
BOD	• Block diagram updated • Offset in the *Available Interrupt Vectors and Sources* table removed • Name column added to bit field description tables: – CTRLA.ACTIVE – CTRLA.SLEEP – INTCTRL.VLMCFG
WDT	• Values in the Period bit field updated
TCA	• Block diagram updated • *Timer/Counter Clock Logic* figure updated • *Signal Description* table updated • *Timer/Counter Block Diagram Split Mode* figure updated • *Event Generators in TCA* table added • *Event Users in TCA* table added • Offset in the *Available Interrupt Vectors and Sources in Normal Mode* and *Available Interrupt Vectors and Sources in Split mode* tables removed • Tables for the CTRLB.WGMODE bit field combined into one table • General improvement of the documentation and its structure

.....continued

Section	Changes
TCB	• Block digram updated • *Timer/Counter Clock Logic* figure added • Figures updated: – *Periodic Interrupt Mode* – *Time-Out Check Mode* – *Input Capture on Event* – *Input Capture Frequency Measurement* – *Input Capture Pulse-Width Measurement* – *Input Capture Frequency and Pulse-Width Measurement Mode* – *Single-Shot Mode* – *8-Bit PWM Mode* • *Event Generators in TCB* table added • *Event Users and Available Event Actions in TCB* table added • Offset in the *Available Interrupt Vectors and Sources* table removed • Name column added to bit field description tables: – CTRLA.CLKSEL – CTRLB.CNTMODE
TCD	• Block diagram updated • *Event Generators in TCD* table added • Offset in the *Available Interrupt Vectors and Sources* table removed • Name column added to bit field description tables: – CTRLA.CLKSEL – CTRLA.CNTPRES – CTRLA.SYNCPRES – CTRLA.ENABLE – DLYCTRL.DLYPRESC – DLYCTRL.DLYTRIG – DLYCTRL.DLYSEL • General improvement of the documentation and its structure
RTC	• *Event Generators in TCD* table added • Offset in the *Available Interrupt Vectors and Sources* table removed • General improvement of the documentation and its structure
USART	• *Event Generators in USART* table added • *Event Users in USART* table added • Offset in the *Available Interrupt Vectors and Sources* table removed • CTRLD register added • General improvement of the documentation and its structure
SPI	• Block diagram updated • *Event Generators in TCD* table added • Offset in the *Available Interrupt Vectors and Sources* table removed • *Interrupt Flags* register separate for Normal and Buffer mode • General improvement of the documentation and its structure

.....continued

Section	Changes
TWI	• Offset in the *Available Interrupt Vectors and Sources* table removed • Name column added to bit field description tables: – CTRLA.FMPEN – MCTRLB.ACKACT – MCTRLB.MCMD • General improvement of the documentation and its structure
CRCSCAN	• Offset in the *Available Interrupt Vectors and Sources* table removed
CCL	• The INSELn bit bields updated
AC	• DACREF removed as internal input
ADC	• Block diagram updated • Typo of WCOMP to WCMP fixed • Removed offset in the *Available Interrupt Vectors and Sources* table • Updated the CTRLA.MUXPOS bit field description
DAC	• Removed repeating notes "Only DAC0 has an output driver for an external pin"
PTC	• Added note about Rs values for mutual capacitance • Updated links to new external documentation
UPDI	• Updated figures: – *UPDI Clock Domains* – *UPDI Instruction Set Overview* – *LDS Instruction Operation* – *STS Instruction Operation* – *LD Instruction Operation* – *ST Instruction Operation* – *LCDS Instruction Operation* – *STCS Instruction Operation* – *REPEAT Instruction Operation* – *Inter Delay Example with LD and RPT* • Added sections: – *BREAK in One-Wire mode* – *SYNCH* and *SYNCH in One-Wire mode* • Extended and improved the documentation related to enabling the UPDI peripheral • Extended and improved the documentation related to disabling the UPDI peripheral • Renamed the *UPDI Enable with 12V Override of RESET pin* section to *UPDI Enable with High-Voltage Override of RESET pin* • Added the *REPEAT Used With LD Instruction Operation* figure • Extended and improved the *Chip Erase* section • Added *Event Generators in UPDI* table • Added documentation for Bus error to the UPDI Error Signature bit field • Reset value for the ASI Control A register is updated • Removed implementation-specific details that are considered not useful for the end users

.....continued	
Section	**Changes**
Electrical Characterization	• Added maximum numbers to the *Power Consumption* section • Rounded numbers in the *Peripherals Power Consumtion* table • Added TCD section • Updated *TWI - Timing Requirements* figure • Updated numbers for t_{OF} in the *TWI - Timing Characteristics* table • Added *SDA Hold Time* table • Added a note about 50% duty cycle requirement for ADC • Added TEMPSENSE section • Updated *Accuracy Characteristics* table for DAC • Updated tables in the AC section • Updated *Peripheral Touch Controller Characteristics - Operating Ratings* table • Added *UPDI Max. Bit Rates vs. VDD* table
Typical Characterization	• Added *Temperature Sensor Error vs. Temperature* ±3σ figure • Added *TWI SDA Hold Time vs Temperature* figure

Note:
1. Change only applies when compared to ATtiny1616/3216 Data Sheet (DS40001997C).
2. Change only applies when compared to ATtiny1617/3217 Data Sheet (DS40001999C).

41.2 Appendix - Obsolete Revision History

Note: Due to document structure change from pin organized documents, the following document history is provided as reference.

- ATtiny1616/3216 Data Sheet (DS40001997C)
- ATtiny1617/3217 Data Sheet (DS40001999C)

41.2.1 ATtiny1616/3216 - DS40001997

Obsolete Publication DS40001997C - 07/2019

Section	Changes
Document	• Editorial updates.
Device	• Introduction: – Added a note for automotive data sheets – Changed text to align with all tinyAVR® 0- and 1-series data sheets • Retention endurance numbers updated • Data Sheet Clarification Document chapter added • Ordering Information moved • I/O Multiplexing and Considerations updated
Configuration and User Fuses	• CRCAPPDIS and CRCBOOTDIS replaced by CRCSRC • Clarified text and added a note for 16k devices in TOUTDIS bit • RSTPINCFG: Time-out after a system reset when fused to be GPIO explained
PORTMUX	• Updated with missing information

.....continued

Section	Changes
BOD - Brown-out Detector	• Removed levels not characterized by minimum and maximum values • Added a note for typical values and reference to electrical characteristics
VREF - Voltage Reference	• Missing CTRLC and CTRLD registers added
TCA	• Added a note for alternative WOn pins
USART	• Clarified One-Wire mode • Clarified text about Disabling the Transmitter
SPI	• Clarified functionality for SPI SS pin
CRCSCAN	• Added a missing MODE bit field
CCL	• Removed reference to interrupts
UPDI	• GPIO functionality disabled for a period after a system reset, changed from ms to clock cycles
Electrical Characteristics	• Added a note for Chip Erase in General Operating Ratings • Corrected the number of PTC channels
Errata	• Errata moved to separate document.
Ordering Information	• Updated with product page links and ordering codes
Product Identification System	• Updated with Tube and Tray packing media
Package Drawings	• Updated package drawings to Microchip standard

Obsolete Publication DS40001997B - 06/2018

Section	Changes
ATtiny3216 Errata	These errata were actually removed from ATtiny3216 die revision C: • ADC: Pending event stuck when disabling ADC • All for CCL, RTC, and USART

Obsolete Publication DS40001997A - 06/2018

Section	Changes
Document	Initial Release

Note:
The ATtiny1616 device was previously described in Microchip document 40001893 rev. C (common data sheet for ATtiny1617/1616/1614 devices).

With the introduction of the ATtiny3216 and ATtiny3217 devices, 40001893 rev. C was replaced by three new data sheets:

- ATtiny1617 and ATtiny3217
- ATtiny1616 and ATtiny3216 (this document, DS40001997)
- ATtiny1614

41.2.2 ATtiny1617/3217 - DS40001999

Obsolete Publication DS40001999C - 07/2019

Section	Changes
Document	• Editorial updates.
Device	• Introduction: – Added a note for automotive data sheets – Changed text to align with all tinyAVR® 0- and 1-series data sheets • Retention endurance numbers updated • Data Sheet Clarification Document chapter added • Ordering Information moved • I/O Multiplexing and Considerations updated
Configuration and User Fuses	• CRCAPPDIS and CRCBOOTDIS replaced by CRCSRC • Clarified text and added a note for 16k devices in TOUTDIS bit • RSTPINCFG: Time-out after a system reset when fused to be GPIO explained
PORTMUX	• Updated with missing information
BOD - Brown-out Detector	• Removed levels not characterized by minimum and maximum values • Added a note for typical values and reference to electrical characteristics
VREF - Voltage Reference	• Missing CTRLC and CTRLD registers added
TCA	• Added a note for alternative WOn pins
USART	• Clarified One-Wire mode • Clarified text about Disabling the Transmitter
SPI	• Clarified functionality for SPI SS pin
CRCSCAN	• Removed unsupported BACKGROUND and CONTINUOUS scan functionality
CCL	• Removed reference to interrupts
UPDI	• GPIO functionality disabled for a period after a system reset, changed from ms to clock cycles
Electrical Characteristics	• Added a note for Chip Erase in General Operating Ratings
Errata	• Errata moved to separate document
Ordering Information	• Updated with product page links and ordering codes
Product Identification System	• Updated with Tube and Tray packing media
Package Drawings	• Updated package drawings to Microchip standard

Obsolete Publication DS40001999B - 06/2018

Section	Changes
ATtiny3217 Errata	These errata were actually removed from ATtiny3217 die revision C: • ADC: Pending event stuck when disabling ADC • All for CCL, RTC, and USART

Obsolete Publication DS40001999A - 06/2018

Section	Changes
Document	Initial Release

Note:
The ATtiny1617 device was previously described in Microchip document 40001893 rev. C (common data sheet for ATtiny1617/1616/1614 devices).

With the introduction of the ATtiny3216 and ATtiny3217 devices, 40001893 rev. C was replaced by three new data sheets:

- ATtiny1617 and ATtiny3217 (this document, DS40001999)
- ATtiny1616 and ATtiny3216
- ATtiny1614

The Microchip Website

Microchip provides online support via our website at http://www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to http://www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

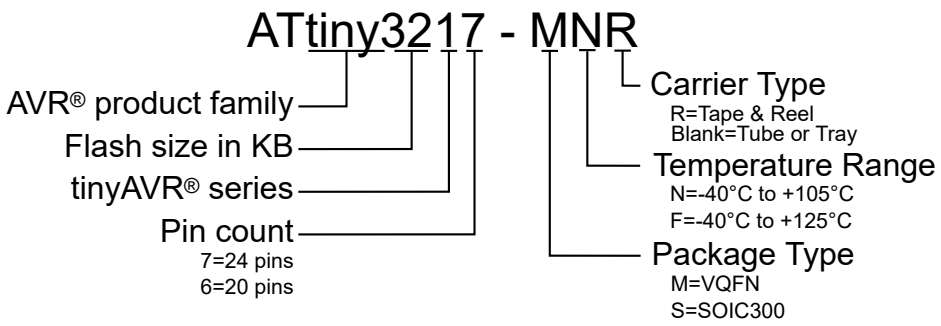
- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: http://www.microchip.com/support

Product Identification System

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.



Note: Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication regarding device applications and the like 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. 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 ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. 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.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox,

ISBN: 978-1-5224-6010-7

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit http://www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: http://www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820