Preliminary Data Sheet

AVR16LA14/20/28/32 Preliminary Data Sheet

Product Page Links

Introduction

The AVR16LA14/20/28/32 microcontrollers (MCUs) of the AVR® LA family use the AVR CPU with a hardware multiplier running at clock speeds of up to 20 MHz. They come with 16 KB of Flash memory, 2 KB of SRAM, and 512 bytes of EEPROM. The microcontrollers are available in 14-, 20-, 28-, and 32-pin packages. The AVR LA family uses the latest technology from Microchip, featuring a flexible and low-power architecture, that includes an Event System, accurate analog subsystems, advanced digital peripherals, and a Peripheral Touch Controller.

Features

- AVR CPU:
  - Running at up to 20 MHz
  - Single-cycle I/O register access
  - Two-level interrupt controller
  - Two-cycle hardware multiplier
- Program and Debug Interface Disable (PDID)
- Supply Voltage Range: 1.62–5.5V
- Memories:
  - 16 KB In-System Programmable (ISP) Flash memory with true read-while-write operation
  - 2 KB SRAM
  - 512B EEPROM
  - 64B of User Row in nonvolatile memory that can retain data during a chip erase and be programmed while the device is locked
  - 64B of Boot Row in nonvolatile memory, accessible only when running from the boot section
  - Write/erase endurance
    - Flash: 10,000 cycles
    - EEPROM: 100,000 cycles
  - Data retention: 40 years at 55°C
- System:
  - Power-On Reset (POR) circuit
  - Brown-out Detector (BOD) with user-programmable levels
  - Voltage Level Monitor (VLM) with an interrupt at a programmable level above the BOD level
  - Clock options:
    - High-precision internal oscillator with a selectable frequency of up to 20 MHz (OSCHF)
      - Auto-tuning for improved internal oscillator accuracy
    - Internal ultra-low power 32.768 kHz oscillator (OSC32K)

    - External 32.768 kHz crystal oscillator (XOSC32K)
    - External clock input
  - Single-pin Unified Program and Debug Interface (UPDI)
  - Three sleep modes:
    - Idle mode, with all peripherals running for immediate wake-up
    - Standby mode, with configurable operation of the selected peripherals
    - Power-Down mode, with full data retention
- Peripherals:
  - Two 16-bit Timer/Counters type B (TCB) with capture and signal measurement capabilities
  - One 16-bit Timer/Counter type E (TCE) with three compare channels for PWM and waveform generator
  - One 16-bit Real-Time Counter (RTC) that can run from an external crystal or an internal oscillator
  - One Universal Synchronous and Asynchronous Receiver and Transmitter (USART):
    - Fractional baud rate generator, auto-baud, start-of-frame detection, and collision detection
    - Supports RS-485, LIN Host/Client, RTS/CTS hardware handshaking, and SPI Host protocols
    - Protocol conversion for accelerated single-wire communication
  - One Host/Client Serial Peripheral Interface (SPI)
  - One Two-Wire Interface (TWI) with dual address match:
    - Independent host and client operation (Dual mode)
    - Inter-Integrated Circuit (I^2C) compatible
    - Standard mode (Sm, 100 kHz)
    - Fast mode (Fm, 400 kHz)
    - Fast mode Plus (Fm+, 1 MHz)
  - 4-channel Event System for CPU-independent and predictable inter-peripheral signaling
  - Configurable Custom Logic (CCL) with up to four programmable Look-up Tables (LUTs)
  - One Analog Comparator (AC)
  - One 10-bit, 166 ksps, Analog-to-Digital Converter (ADC)
  - Peripheral Touch Controller (PTC) with Driven Shield+ and Boost Mode technologies for capacitive touch buttons, sliders, wheels and 2D surfaces
    - Supports up to 27 PTC pins for both self-capacitance and mutual-capacitance channels
    - Supports External Integration Capacitors for large sensors
    - Uses the 10-bit ADC peripheral for signal acquisition and conversion
  - Internal 0.512V, 1.024V, 2.048V, 2.500V, 4.096V voltage references, with an external reference option
  - Automated Cyclic Redundancy Check (CRC) Flash program memory scan
  - Watchdog Timer (WDT) with Window mode and a separate on-chip oscillator
  - External interrupts on all general purpose pins
- I/O and Packages:
  - 12 to 28 programmable I/O pins
  - 14-pin SOIC and TSSOP
  - 20-pin SSOP and VQFN 3x3 with Wettable Flanks
  - 28-pin SSOP, SPDIP and VQFN 4x4 with Wettable Flanks
  - 32-pin TQFP 7x7 and VQFN 5x5 with Wettable Flanks

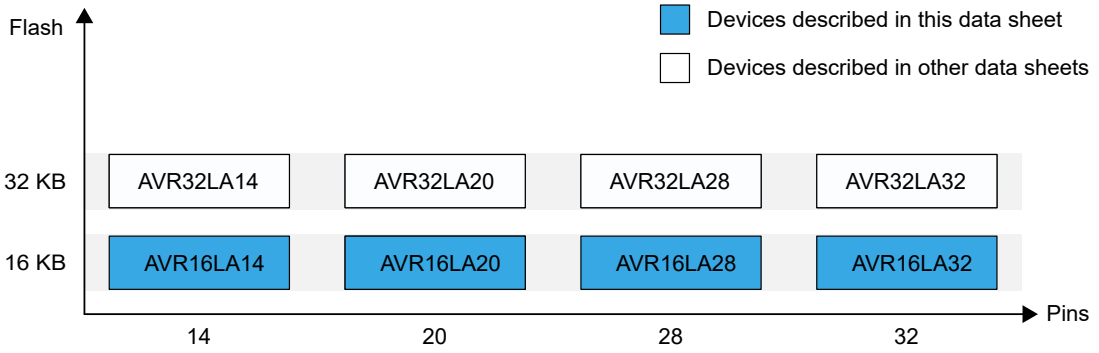
- Temperature Ranges:
  - Industrial: -40°C to 85°C ambient
  - Extended: -40°C to 125°C ambient

AVR® LA Family Overview

The figure below shows the AVR LA family of devices, illustrating the pin-count variants and memory sizes:

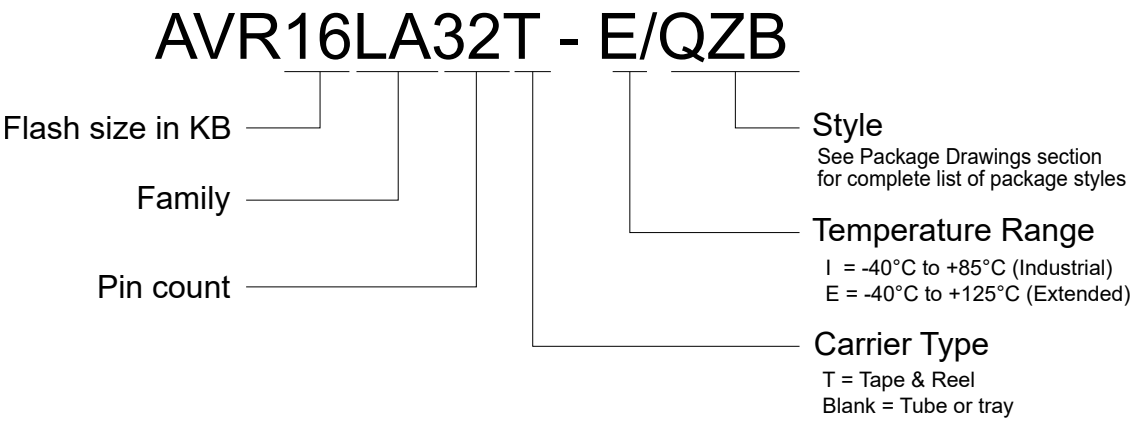
- Vertical migration is possible without code modification, as these devices are fully pin- and feature-compatible
- Horizontal migration to the left reduces the pin count and, therefore, the available features

Figure 1. AVR® LA family Overview



The name of a device in the AVR LA family is decoded as follows:

Figure 2. AVR® LA family Device Designations



Memory Overview

Table 1. Memory Overview

Devices	AVR16LA14 AVR16LA20 AVR16LA28 AVR16LA32
Flash memory	16 KB
Non Read-While-Write (NRWW) section of Flash	4 KB
Read-While-Write (RWW) section of Flash	12 KB
SRAM	2 KB
EEPROM	512B
User row	64B
Boot row	64B

Peripheral Overview

Table 2. Peripheral Overview

Feature	AVR16LA14	AVR16LA20	AVR16LA28	AVR16LA32
Pins	14	20	28	32
Maximum frequency (MHz)	20	20	20	20
16-bit Timer/Counter type B (TCB)	2	2	2	2
16-bit Timer/Counter type E (TCE)	1	1	1	1
Real-Time Counter (RTC)	1	1	1	1
USART	1	1	1	1
SPI	1	1	1	1
TWI/I^2C	1(1)	1(1)	1(1)	1(1)
10-bit ADC (channels)	1 (4)	1 (10)	1 (16)	1 (20)
Analog Comparator (AC)	1	1	1	1
Peripheral Touch Controller (PTC) (Pins)	1 (11)	1 (17)	1 (23)	1 (27)
Configurable Custom Logic Look-up Table (CCL LUT)	4	4	4	4
Watchdog Timer (WDT)	1	1	1	1
Event System (EVSYS) channels	4	4	4	4
General Purpose I/O pins (input/output(2))	12/11	18/17	24/23	28/27
PORT	PA[1:0] PC[3:0] PD[7:4] PF[7:6]	PA[7:0] PC[3:0] PD[7:4] PF[7:6]	PA[7:0] PC[3:0] PD[7:0] PF[7,6,1,0]	PA[7:0] PC[3:0] PD[7:0] PF[7:0]
External interrupts	12	18	24	28
CRCSCAN	1	1	1	1
Unified Program and Debug Interface (UPDI)	1	1	1	1

Notes:

1. The TWI/I^2C can operate simultaneously as a Host and Client on different pins.
2. PF6/$\overline{\text{RESET}}$ pin is input only.

Security Concept

The AVR® LA family of MCUs are general purpose microcontrollers that offer fundamental security features to enable secure firmware upgrades and authenticate the application firmware. When these security features are

used correctly, they protect against remote attacks and certain PCB-level attacks in which the application code is modified to change the product's functionality.

The cornerstone of the security features is the *Program and Debug Interface Disable (PDID)*, a mechanism that prevents access to the device's reprogrammable Flash memory over the Unified Program and Debug Interface (UPDI). After activating PDID, as described in the *Memories* chapter, UPDI is prevented from making any changes to the device. However, UPDI can still read the device information and CRC status.

The only way to program the device after activating PDID is by using software stored in the Boot Code section of Flash to update the software in the Application Code section. This application-specific software must be able to receive new data and program the Application Code section. It is impossible to alter the code stored in the Boot Code section using this mechanism, as it is accessible only through the UPDI.

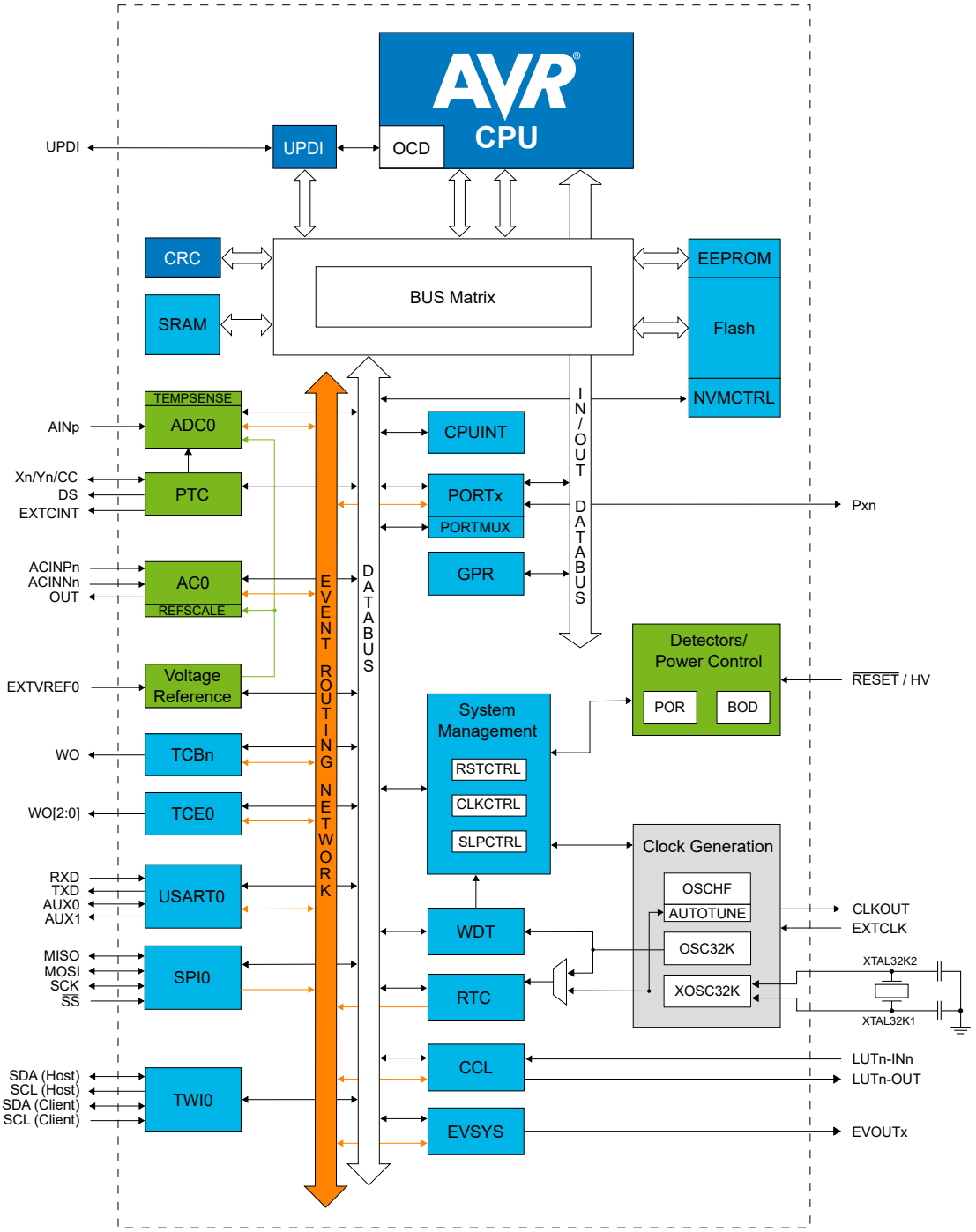
In addition, there is a separate storage space accessible only by code in the Boot Code section, which can hold data intended to be accessed exclusively from the Boot Code section. One example is a cryptographic key used to validate data sent to a bootloader for updating the application software on the device.

This creates a two-layer security system: The device is prevented from being erased or reprogrammed over UPDI, and the code in the Boot Code section is protected. Additionally, the code in the Boot Code section can use a cryptographic key (accessible only by code in this section of Flash) to verify that any new application code received for a device software update is authentic.

Using the *Program and Debug Interface Disable (PDID)* feature in software requires cryptographic expertise to ensure compliance with cybersecurity standards such as ISO/SAE DIS 21434.

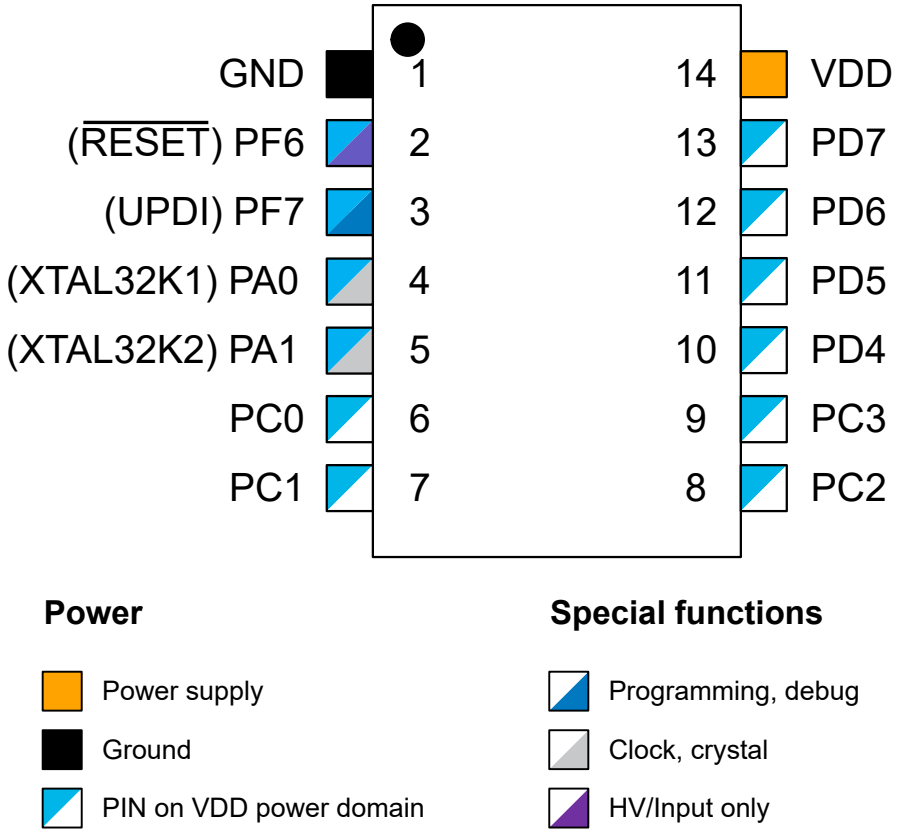
Table of Contents

1. Block Diagram

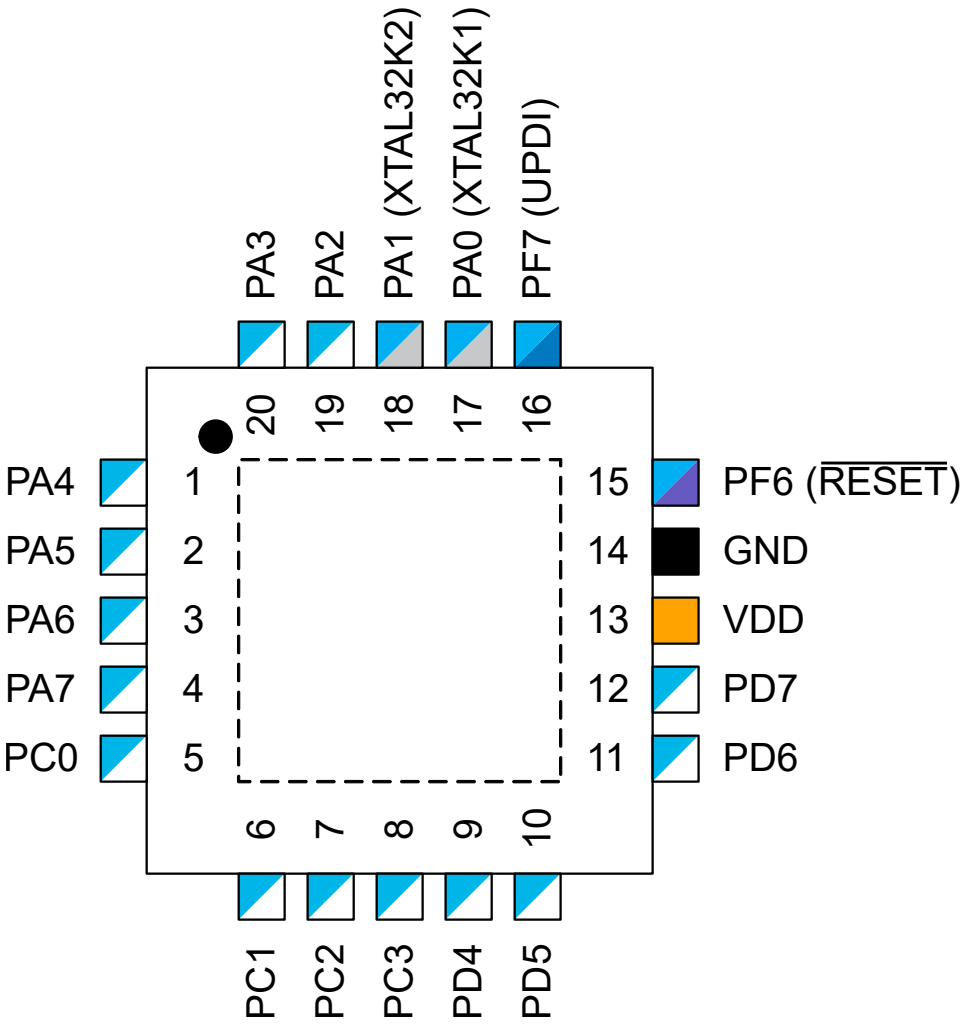


2. Pinout

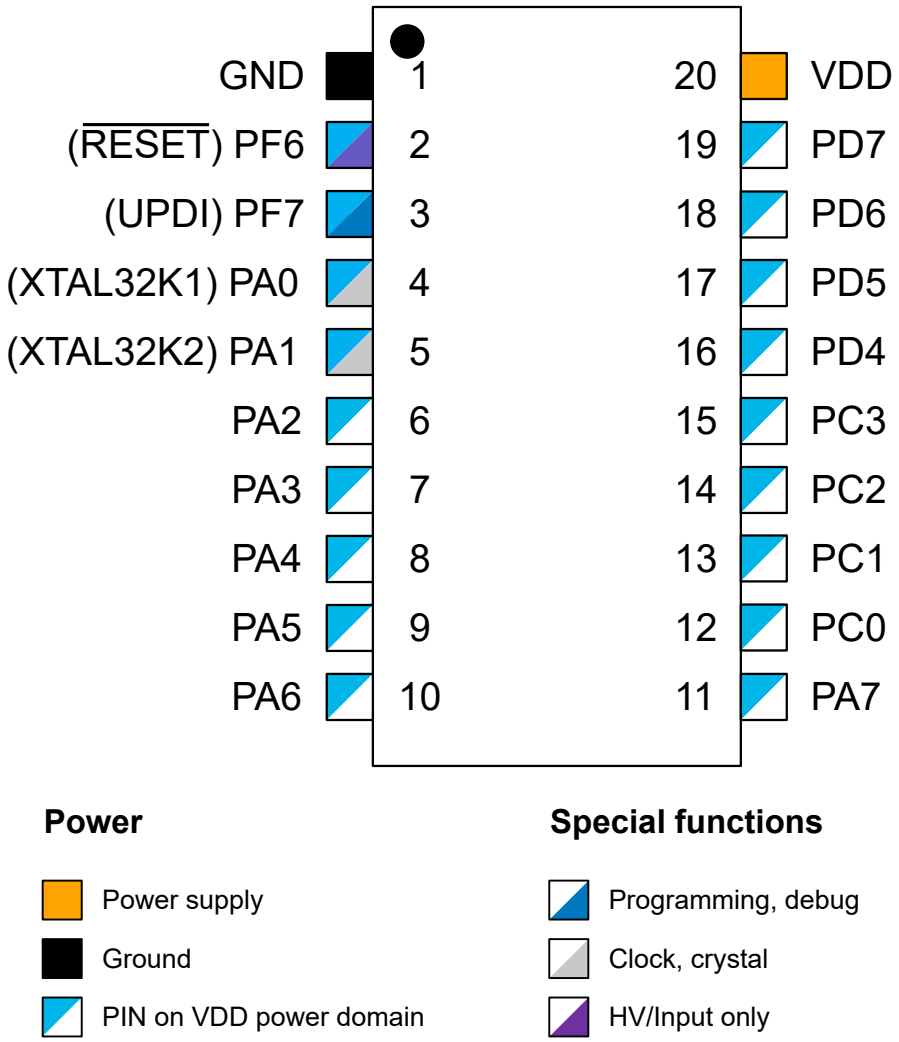
2.1. 14-Pin TSSOP and SOIC



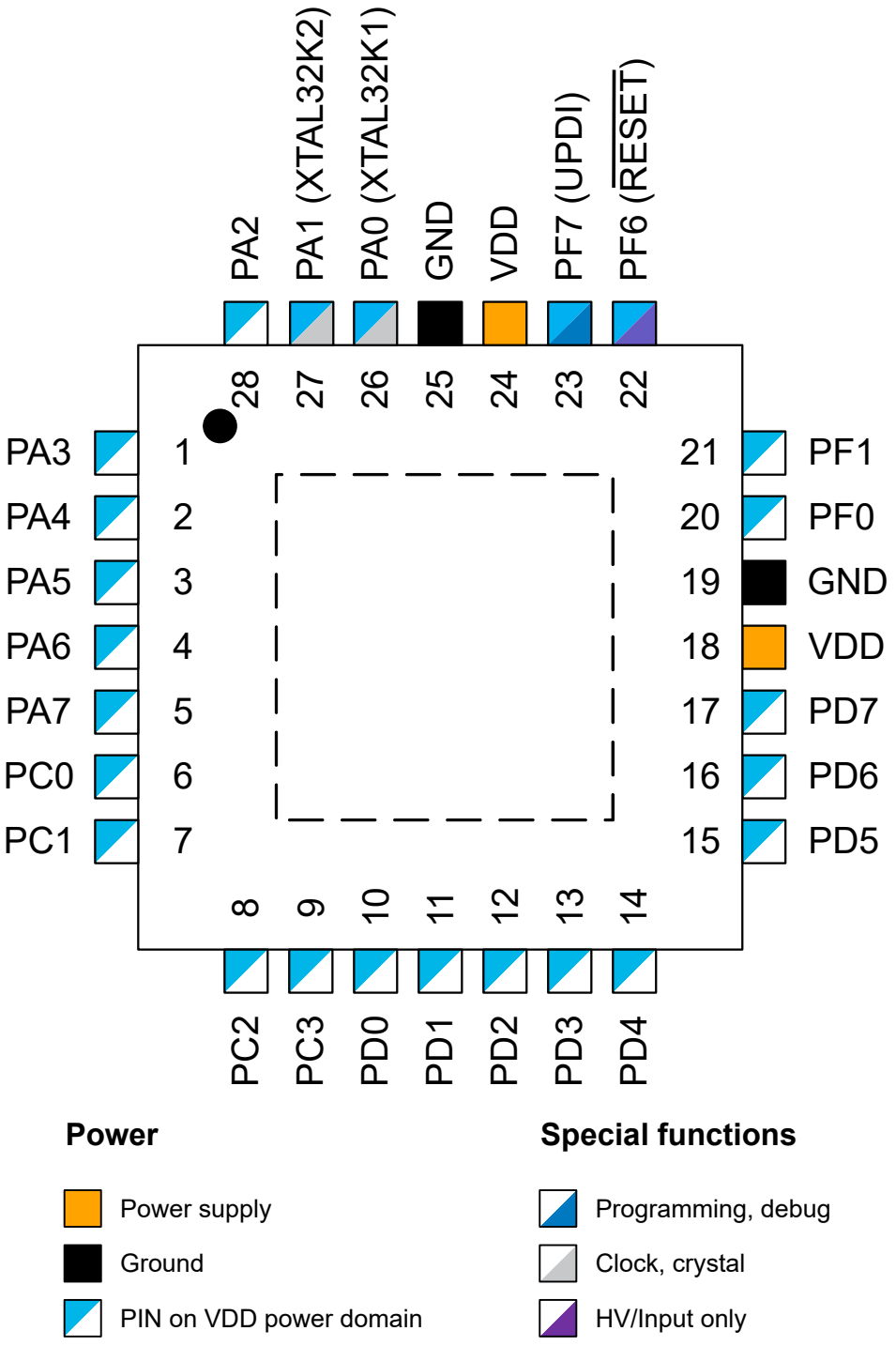
2.2. 20-Pin VQFN



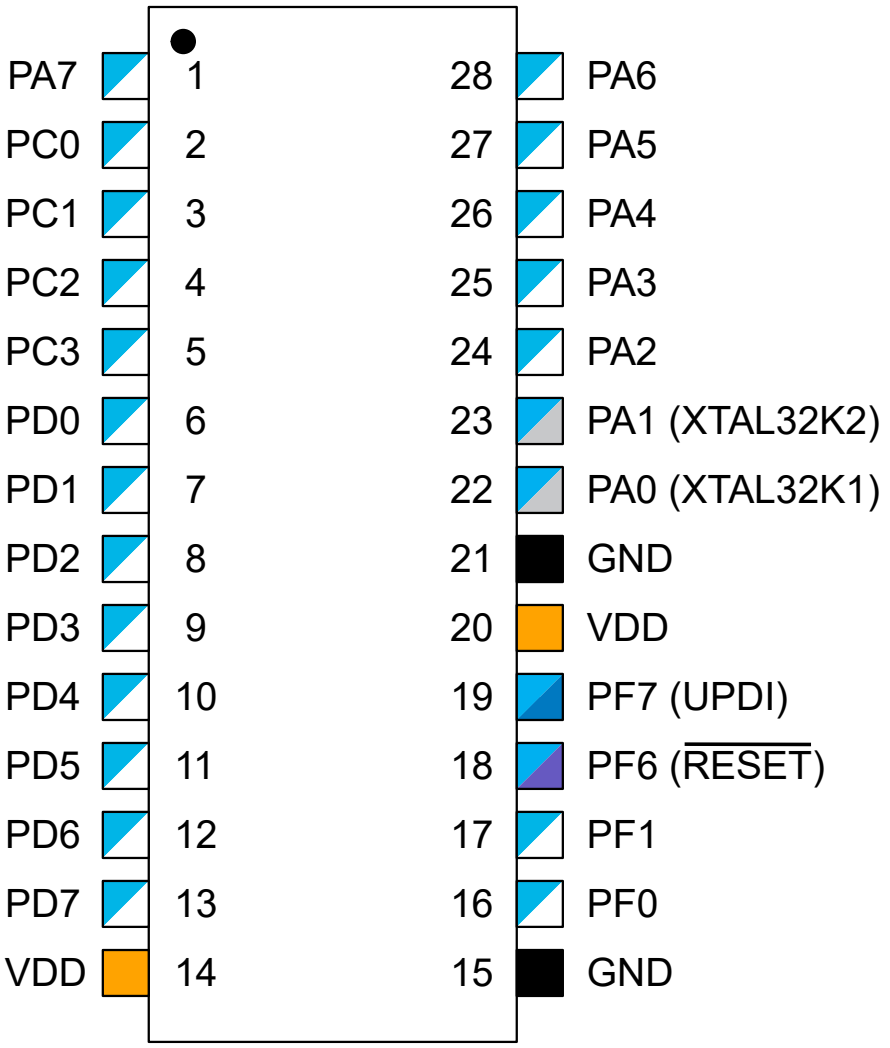
2.3. 20-Pin SSOP



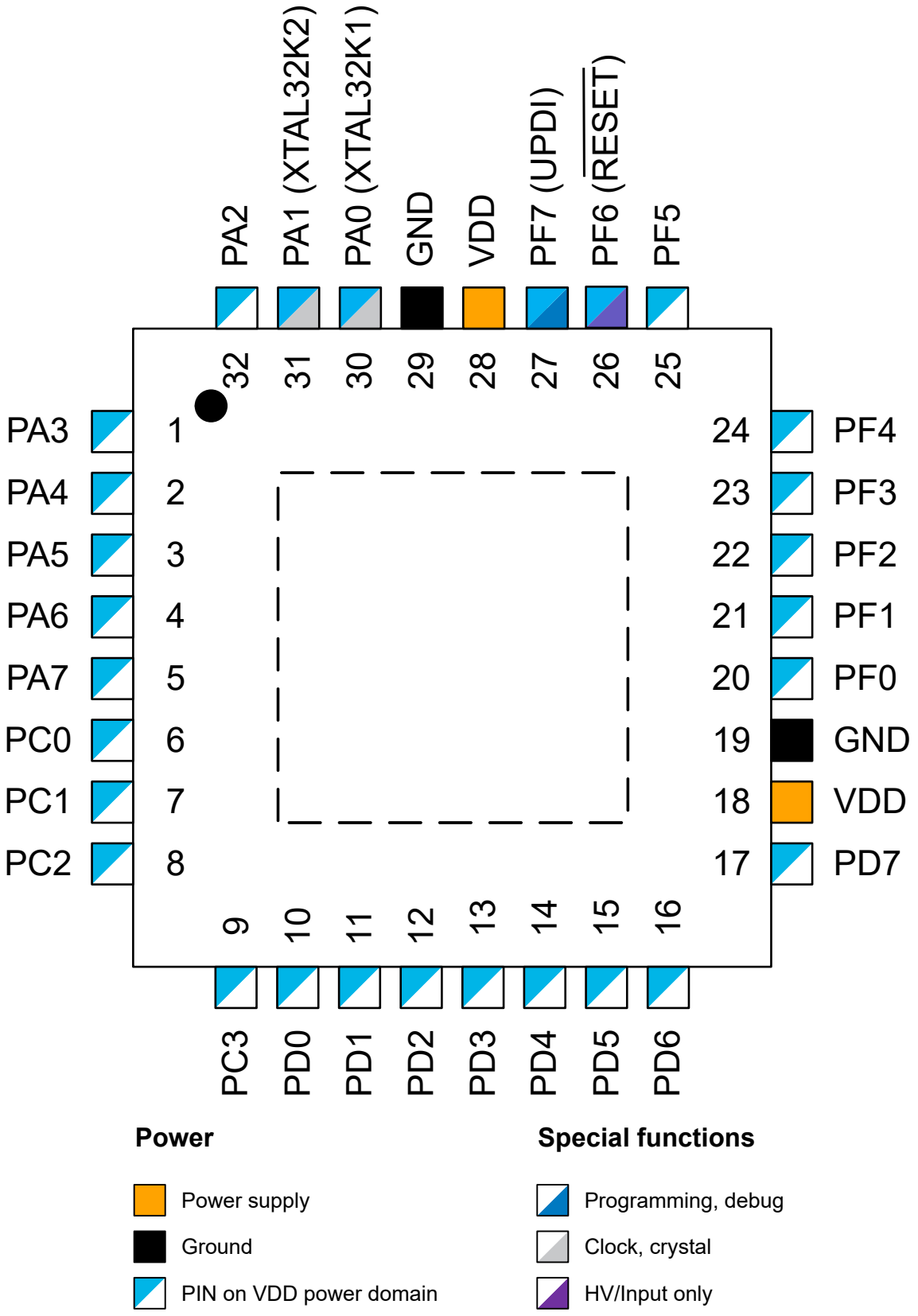
2.4. 28-Pin VQFN



2.5. 28-Pin SSOP and SPDIP



2.6. 32-Pin VQFN and TQFP



3. I/O Multiplexing and Considerations

3.1. I/O Multiplexing

VQFN/TQFP 32-Pin	VQFN 28-Pin	SSOP/SDIP 28-Pin	VQFN 20-Pin	SSOP 20-Pin	SOIC/TSSOP 14-Pin	Pin Name[1]	Special	ADC0	AC0	PTC	USART0	SPI0	TWI0[3]	TCE0	TCBn	EVSYS	CCL-LUTn
30	26	22	17	4	4	PA0	XTAL32K1 EXTCLK			X0/Y0	TxD	MOSI[2]	SDA (HC)[2]	WO0			0, IN0 0, IN0[2]
31	27	23	18	5	5	PA1	XTAL32K2			X1/Y1	RxD	MISO[2]	SCL (HC)[2]	WO1			0, IN1 0, IN1[2]
32	28	24	19	6	-	PA2		AIN22		X2/Y2	AUX0 TxD[2]		SDA (HC) SDA (HC)[2]	WO2	0, WO	EVOUTA	0, IN2 0, IN2[2]
1	1	25	20	7	-	PA3		AIN23		X3/Y3	AUX1 RxD[2]		SCL (HC) SCL (HC)[2]		1, WO		0, OUT
2	2	26	1	8	-	PA4		AIN24		X4/Y4	TxD[2]	MOSI					
3	3	27	2	9	-	PA5		AIN25		X5/Y5	RxD[2]	MISO					
4	4	28	3	10	-	PA6		AIN26		X6/Y6	AUX0[2]	SCK					0, OUT[2]
5	5	1	4	11	-	PA7	CLKOUT	AIN27	OUT	X7/Y7	AUX1[2]	$\overline{SS}$				EVOUTA[2]	
6	6	2	5	12	6	PC0				X48/Y48		SCK[2] MOSI[2]		WO0[2]			1, IN0
7	7	3	6	13	7	PC1				X49/Y49	TxD[2]	$\overline{SS}$[2] MISO[2] MOSI[2]		WO1[2]			1, IN1
8	8	4	7	14	8	PC2				X50/Y50	RxD[2]	SCK[2] MISO[2]	SDA (C) SDA (HC)[2] SDA (C)[2]	WO2[2]		EVOUTC	1, IN2
9	9	5	8	15	9	PC3				X51/Y51	AUX0[2]	$\overline{SS}$[2] SCK[2]	SCL (C) SCL (HC)[2] SCL (C)[2]				1, OUT
10	10	6	-	-	-	PD0		AIN0	ACINN1	X16/Y16				WO0[2]			2, IN0 2, IN0[2]
11	11	7	-	-	-	PD1		AIN1		X17/Y17				WO1[2]			2, IN1 2, IN1[2]
12	12	8	-	-	-	PD2		AIN2	ACINP0	X18/Y18				WO2[2]		EVOUTD	2, IN2 2, IN2[2]
13	13	9	-	-	-	PD3		AIN3	ACINN0	X19/Y19							2, OUT
14	14	10	9	16	10	PD4	EXTCINT0[4]	AIN4		X20/Y20	TxD[2]	MOSI[2]					
15	15	11	10	17	11	PD5	EXTCINT1[4]	AIN5		X21/Y21	RxD[2]	MISO[2]					
16	16	12	11	18	12	PD6		AIN6	ACINP3	X22/Y22	AUX0[2]	SCK[2]					2, OUT[2]
17	17	13	12	19	13	PD7	EXTVREF0	AIN7	ACINN2	X23/Y23	AUX1[2]	$\overline{SS}$[2]				EVOUTD[2]	
18	18	14	13	20	14	VDD											
19	19	15	14	1	1	GND											
20	20	16	-	-	-	PF0		AIN16		X32/Y32				WO0[2]			3, IN0
21	21	17	-	-	-	PF1		AIN17		X33/Y33				WO1[2]			3, IN1
22	-	-	-	-	-	PF2		AIN18		X34/Y34				WO2[2]		EVOUTF	3, IN2
23	-	-	-	-	-	PF3		AIN19		X35/Y35							3, OUT
24	-	-	-	-	-	PF4		AIN20		X36/Y36					0, WO[2]		
25	-	-	-	-	-	PF5		AIN21		X37/Y37					1, WO[2]		
26	22	18	15	2	2	PF6[5]	$\overline{RESET}$				RxD[2]						
27	23	19	16	3	3	PF7	UPDI			X39/Y39	TxD[2]	$\overline{SS}$[2]				EVOUTF[2]	
28	24	20	-	-	-	VDD											
29	25	21	-	-	-	GND											

Notes:

1. The pin names are of the P*xn* type, where *x* denotes the PORT instance (A, B, C, ...) and *n* denotes the pin number. The notation for signals is PORT*x*_PIN*n*. All pins can be used as event inputs.
2. Alternative pin positions. For selecting alternate positions, refer to the *PORTMUX - Port Multiplexer* chapter.
3. The TWI pins that can be used as host or client pins are marked “HC”, while pins marked “C” can only be used as client pins.
4. External Integration capacitor pins for the PTC.
5. Input-only.

4. Hardware Guidelines

This section contains guidelines for designing or reviewing electrical schematics using AVR 8-bit microcontrollers. The information presented here is a brief overview of the most common topics. More detailed information can be found in application notes, listed in this section where applicable.

4.1. General Guidelines

Unused pins must be soldered to their respective soldering pads. The soldering pads must not be connected to the circuit.

The PORT pins are in their default state after Reset. Follow the recommendations in the *PORT - I/O Pin Configuration* chapter to reduce power consumption.

All values are typical values and serve only as a starting point for circuit design.

Refer to the following application notes for further information:

- *AVR040 - EMC Design Considerations*
- *AVR042 - AVR Hardware Design Considerations*

4.1.1. Special Consideration for Packages With Center Pad

Flat packages often include an exposed pad located on the bottom, often referred to as the center pad or the thermal pad. This pad is not electrically connected to the internal circuit of the chip but is mechanically bonded to the internal substrate. It serves as a thermal heat sink and provides added mechanical stability. This pad must be connected to GND, as the ground plane is the best heat sink (largest copper area) of the Printed Circuit Board (PCB).

4.2. Connection for Power Supply

The basics and details of power supply design lie beyond the scope of these guidelines. See the application notes mentioned at the beginning of this section for more detailed information about this subject.

A decoupling capacitor must be placed close to the microcontroller for each supply pin pair (VDD or other power supply pin and its corresponding GND pin). If the decoupling capacitor is placed too far from the microcontroller, a high-current loop might form that will result in increased noise and increased radiated emission.

Each supply pin pair (power input pin and ground pin) must have separate decoupling capacitors.

It is recommended to place the decoupling capacitor on the same side of the PCB as the microcontroller. If space does not allow it, the decoupling capacitor may be placed on the other side through a via, but make sure to keep the distance to the supply pin as short as possible.

If the board is experiencing high-frequency noise (upward of tens of MHz), add a second ceramic type capacitor parallel to the decoupling capacitor described above. Place this second capacitor next to the primary decoupling capacitor.

On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first and then to the device pins, ensuring that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

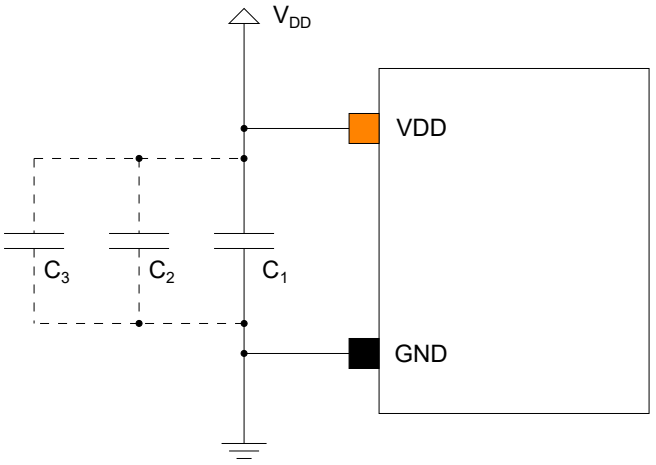
As mentioned at the beginning of this section, all values used in examples are typical values. The actual design may require other values.

4.2.1. Power Supply

For higher pin-count package types, there are several V_{DD} and corresponding GND pins. All the V_{DD} pins in the microcontroller are internally connected. The same voltage must be applied to each of the V_{DD} pins.

The figure below shows the recommended connection of the power supply to the device's V_{DD} pin(s).

Figure 4-1. Recommended V_{DD} Connection Circuit Schematic



Typical values (recommended):
C_1: 100 nF (primary decoupling capacitor)
C_2: 1-10 nF (HF decoupling capacitor)
C_3(*): 1 µF (decoupling capacitor - optional)

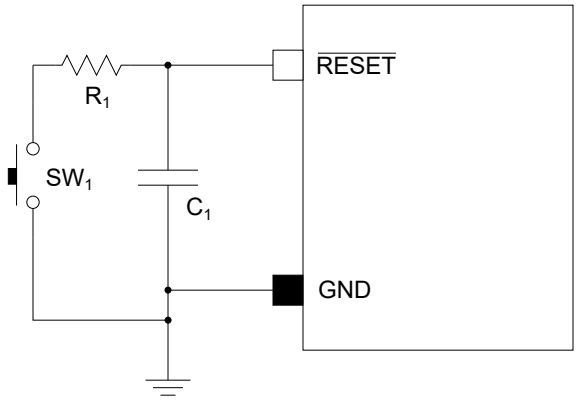
Important: For systems that frequently cycle V_{DD} or experience fast V_{DD} transients, it is recommended to add a decoupling capacitor (C_3) if the power supply slew rate exceeds the specified limits. Refer to the *Supply Voltage* section in the *Electrical Characteristics* for details about the power supply's slew rate limits.

4.3. Connection for $\overline{\text{RESET}}$

The $\overline{\text{RESET}}$ pin on the device is active-low with an internal pull-up resistor, and externally pulling the pin low will result in a device Reset. An external pull-up resistor is usually not required.

The following figure shows the recommended method for connecting an external Reset switch to the device.

Figure 4-2. Recommended External Reset Circuit Schematic



Typical values (recommended):
C_1: 100 nF (filtering capacitor)
R_1: 330Ω (switch series resistance)

Shorting the filtering capacitor may cause a noise spike that can harm the system. To prevent this, a resistor in series with the switch can safely discharge the filtering capacitor, preventing a current surge.

UPDI Enable With High-Voltage Override

It is possible to enable a disabled UPDI by applying a high-voltage pulse on the $\overline{\text{RESET}}$ pin. Take care in the reset circuit design and with any components connected to the $\overline{\text{RESET}}$ pin to prevent damage if such a high-voltage pulse is applied.

See the *Connection for UPDI Programming* section and the *UPDI - Unified Program and Debug Interface* chapter for more details.

4.4. Connection for UPDI Programming

The Unified Program and Debug Interface (UPDI) connection provides a one-wire interface for external programming and On-Chip Debugging (OCD). This section covers only the physical connection and not the details of the signal protocol or the features of the UPDI peripheral. These details are described in the *UPDI - Unified Program and Debug Interface* chapter.

The recommended UPDI connection has changed since its initial introduction. For this reason, both connections are described below: The initial UPDI connection layout is named **UPDI Connection v1**, while the newer layout is named **UPDI Connection v2**. The difference between the two connections is the inclusion of a $\overline{\text{RESET}}$ signal in the v2 connection.

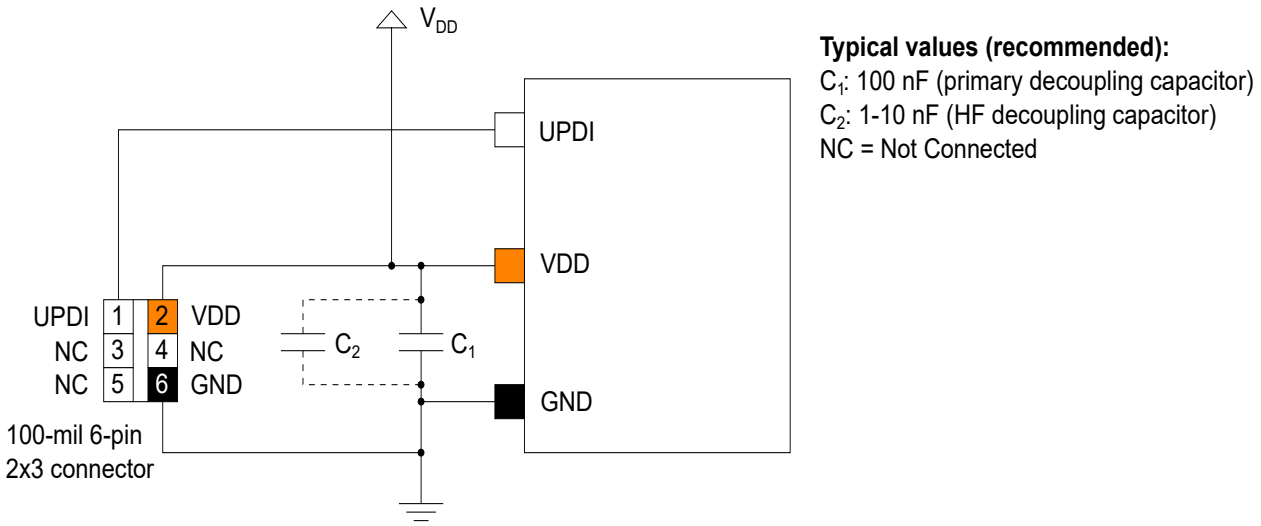
4.4.1. UPDI Connection v1

This was the initial layout for the UPDI connection used by older programming tools (like the Atmel ICE).

The **UPDI Connection v1** is a 100-mil 6-pin 2x3 header. Even though using only three pins for programming, it is recommended to use a 2x3 header since most programming tools using this connection are delivered with 100-mil 6-pin 2x3 connectors.

The following figure shows the recommendation for a UPDI connection to the device using the **UPDI Connection v1**.

Figure 4-3. Recommended UPDI Programming Circuit Schematic



The decoupling capacitor between VDD and GND must be placed as close to the pin pair as possible. Include the decoupling capacitor even if the UPDI connector is not included in the circuit.

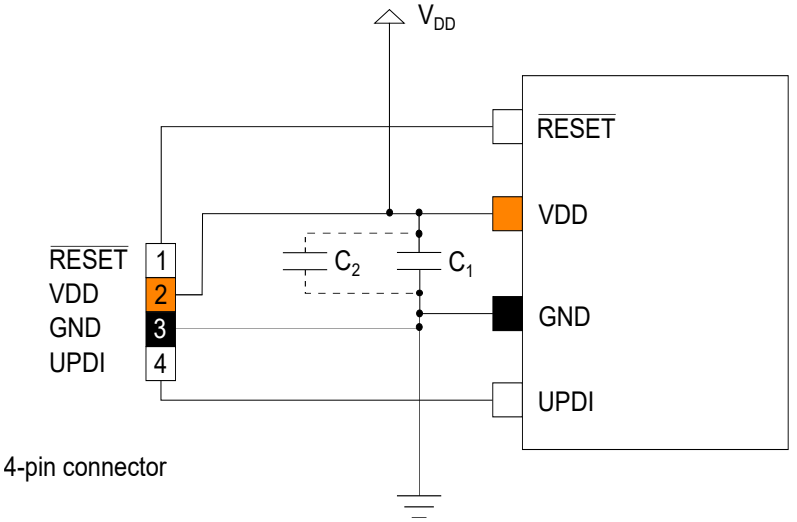
4.4.2. UPDI Connection v2

This connection is compatible with any AVR device but requires an adapter cable for users with older programmers/debuggers, such as the *Atmel-ICE* and the *Atmel PowerDebugger* with the 100-mil 2x3 header connector. This connection is directly compatible with the *PICkit™ 4 In-Circuit Debugger* programming tool.

The *UPDI Connection v2* uses a 100-mil 4-pin 1x4 header. Although three pins are sufficient for programming many AVR devices, it is recommended to use a single-row 100-mil 4-pin header to allow inclusion of the $\overline{\text{RESET}}$ signal. This connector is also compatible with the *PICkit 4* programmer.

The following figure shows the recommended method for connecting a UPDI connector to the device.

Figure 4-4. Recommended UPDI Programming Circuit Schematic



Typical values (recommended):
C_1: 100 nF (primary decoupling capacitor)
C_2: 1-10 nF (HF decoupling capacitor)

The decoupling capacitor between VDD and GND must be placed as close to the pin pair as possible. Include the decoupling capacitor even if the UPDI connector is not included in the circuit.

Enabling UPDI Using $\overline{\text{RESET}}$

By design or by mistake, it may be possible to disable UPDI by writing to the appropriate fuse. For details on disabling UPDI, see the *FUSE - Configuration and User Fuses* section of the *Memories* chapter. Note that for devices with a dedicated UPDI pin, there is no fuse to disable UPDI.

A high-voltage pulse must be applied to the $\overline{\text{RESET}}$ pin to re-enable UPDI. See the *UPDI - Unified Program and Debug Interface* chapter for details on how to apply the high-voltage pulse to the $\overline{\text{RESET}}$ pin.

Take additional care in the circuit design if the $\overline{\text{RESET}}$ pin is connected to other components. If a high-voltage pulse is applied to the $\overline{\text{RESET}}$ pin, other components connected to the line might be damaged. In this case, the design must allow these components to be disconnected from the circuit before the high-voltage pulse is applied. One example is a removable jumper.

Note: On devices that feature the *Program and Debug Interface Disable (PDID)*, UPDI cannot be re-enabled using the $\overline{\text{RESET}}$ pin after the PDID feature has been activated.

4.5. Connecting External Crystal Oscillators

The use of external oscillators and the design of oscillator circuits are not trivial because of many variables: V_{DD}, operating temperature range, crystal type and manufacture, loading capacitors, circuit layout, and PCB material. Some typical guidelines to help with the basic oscillator circuit design are presented in this section.

- Even the best performing oscillator circuits and high-quality crystals will not perform well if the layout and materials used during the assembly are not carefully considered
- The crystal circuit must be placed on the same side of the board as the device. Place the crystal circuit as close to the respective oscillator pins as possible and avoid long traces. This will reduce parasitic capacitance and increase immunity against noise and crosstalk. Mount the load capacitors on the same side of the board and next to the crystal. Do not use sockets.

- Place a grounded copper area around the crystal circuit to isolate it from surrounding circuits. If the circuit board has two sides, the copper area on the bottom layer must be a solid area covering the crystal circuit. The copper area on the top layer must surround the crystal circuit and be connected to the bottom layer area by using via(s).
- Do not run any signal traces or power traces inside the grounded copper area. Avoid routing digital lines, especially clock lines, close to the crystal lines.
- If using a two-sided PCB, avoid any traces beneath the crystal. For a multilayer PCB, avoid routing signals below the crystal lines.
- Dust and humidity will increase parasitic capacitance and reduce signal isolation. A protective coating is recommended.
- Successful oscillator design requires good specifications of operating conditions, a component selection phase with initial testing, and testing in actual operating conditions to ensure that the oscillator performs as desired

For more detailed information about oscillators and oscillator circuit design, see the following application notes:

- *AN2648 - Selecting and Testing 32 kHz Crystal Oscillators for AVR® Microcontrollers*
- *AN949 - Making Your Oscillator Work*

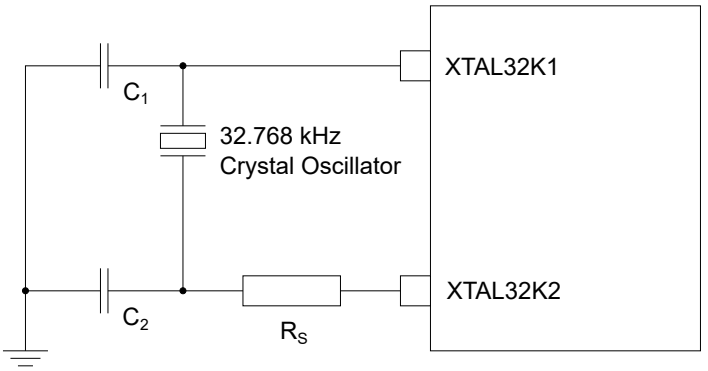
4.5.1. Connection for XTAL32K (External 32.768 kHz Crystal Oscillator)

Ultra-low power 32.768 kHz oscillators typically dissipate significantly below 1 µW, and the current flowing in the circuit is, therefore, extremely small. The crystal frequency is highly dependent on the capacitive load.

A series resistor R_S may be required to prevent overdriving the oscillator. The gain from the oscillator driver may sometimes be too high for low-frequency oscillators, and adding impedance with R_S can decrease the gain. The overdrive causes the oscillator to not swing properly, as the signal will be saturated (clipped or “squashed”). Overdriving the crystal can also lead to the circuit jumping to a higher harmonic.

The following figure shows how to connect an external 32.768 kHz crystal oscillator:

Figure 4-5. Recommended External 32.768 kHz Oscillator Connection Circuit Schematic

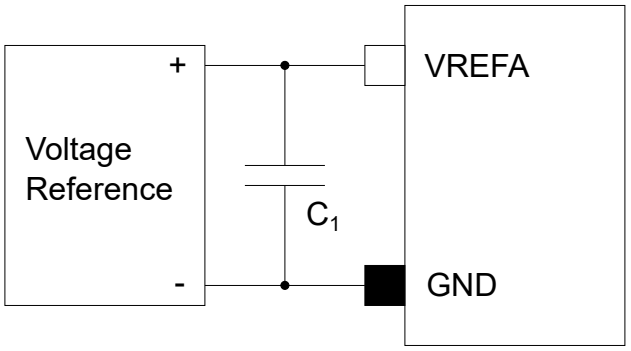


4.6. Connection for External Voltage Reference

If the design includes using an external voltage reference, the general recommendation is to use a suitable capacitor connected in parallel to the reference. The nature of the reference and the type of electrical noise that needs to be filtered out gives the capacitor value.

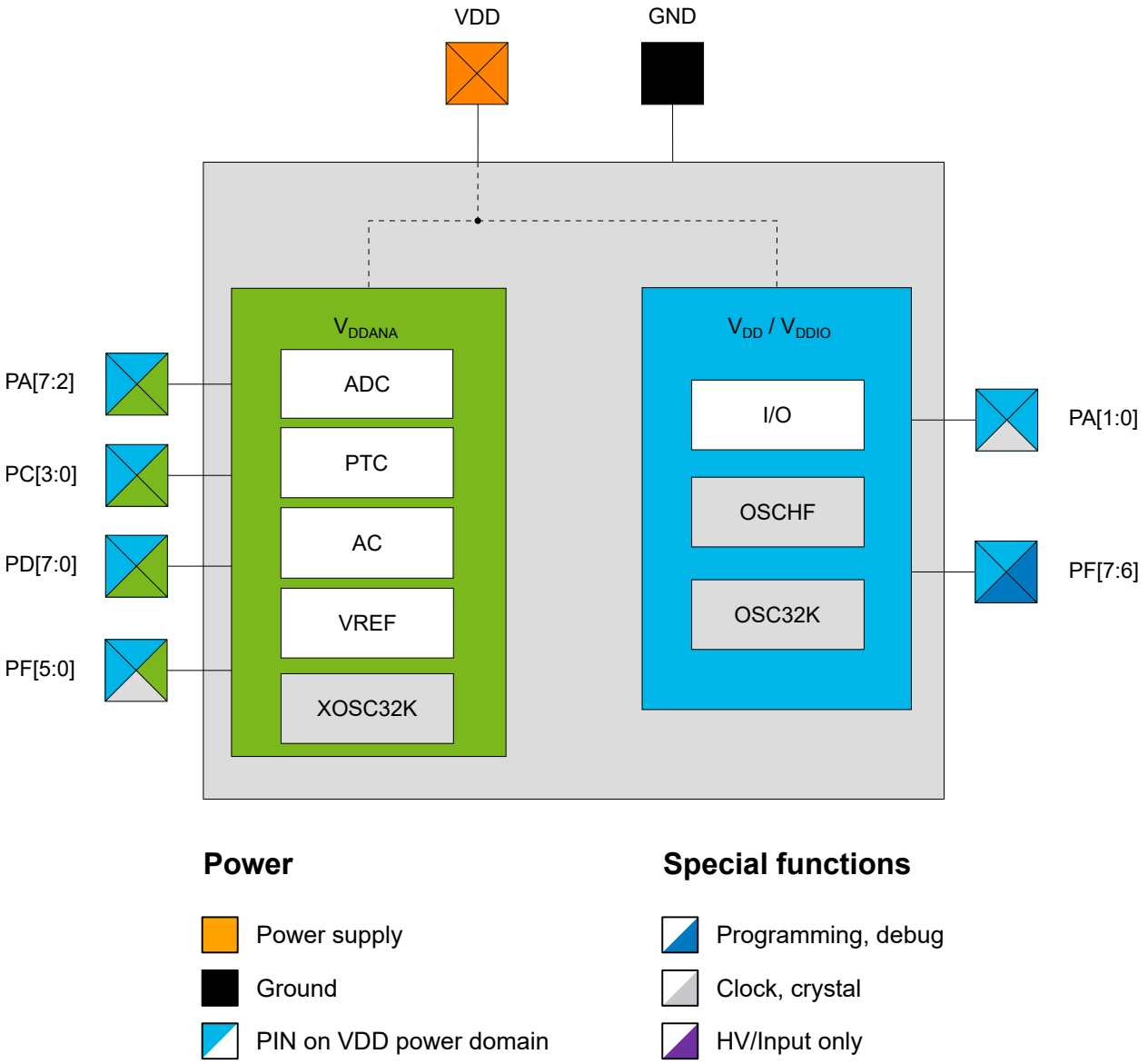
Additional filtering components may be necessary depending on the type of external voltage reference used.

Figure 4-6. Recommended External Voltage Reference Connection



5. Power Supply

5.1. Power Domains



The AVR LA family of devices has several power domains with the following power supply pins:

Domain	Pin	Description
V_{DD}/V_{DDIO}	VDD	Powers I/O lines
V_{DDANA}	VDD	XTAL32K (external 32.768 kHz oscillator) and the analog peripherals

Notes:

- The same voltage must be applied to all VDD pins. This common voltage is referred to as VDD in the data sheet.
- The ground pins (GND) are common to all power supply pins (VDD)
- For recommendations on layout and decoupling, refer to the *Hardware Guidelines* chapter

5.2. Power-up Sequence

The voltage ramp-up on all V_{DD} pins must be closely matched during the power-up sequence to ensure proper operation.

The Power-On Reset (POR) and the Brown-out Detector (BOD) monitor V_{DD} and keep the system in Reset if the voltage level is below the respective thresholds. Refer to the *RSTCTRL - Reset Controller* and *BOD - Brown-out Detector* chapters for further information.

Refer to the *Electrical Characteristics* chapter for further information on voltage thresholds and supply voltage slope.

6. Conventions

6.1. Numerical Notation

Table 6-1. 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

6.2. Memory Size and Type

Table 6-2. 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

6.3. Frequency and Time

Table 6-3. 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

6.4. Registers and Bits

Table 6-4. 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	Bit field names are shown in uppercase. Example: INTMODE.

Table 6-4. Register and Bit Mnemonics (continued)

Symbol	Description
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.

6.4.1. Addressing Registers from Header Files

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 6-1. 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:

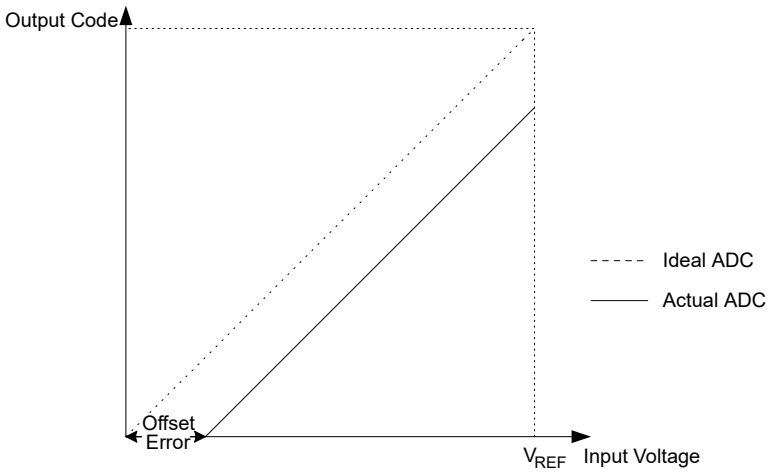
```
// TCA0 in Normal Mode (SINGLE) uses waveform generator in frequency mode
TCA0.SINGLE.CTRL=TCA_SINGLE_WGMODE_FRQ_gc;
```

6.5. 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:

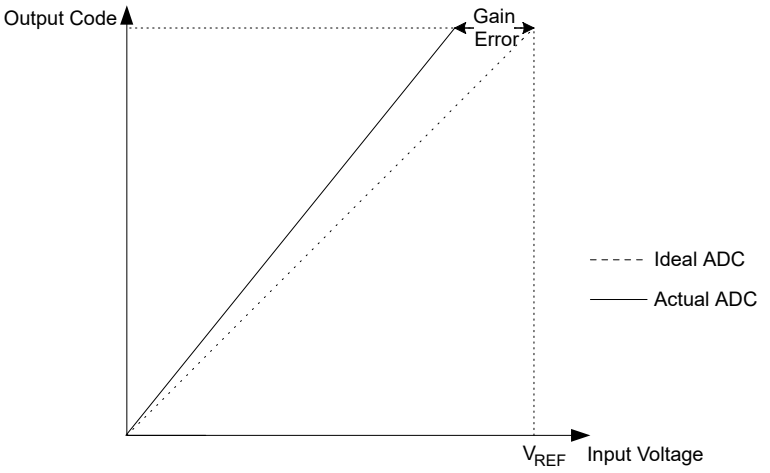
Offset Error The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSb). Ideal value: 0 LSb.

Figure 6-1. Offset Error



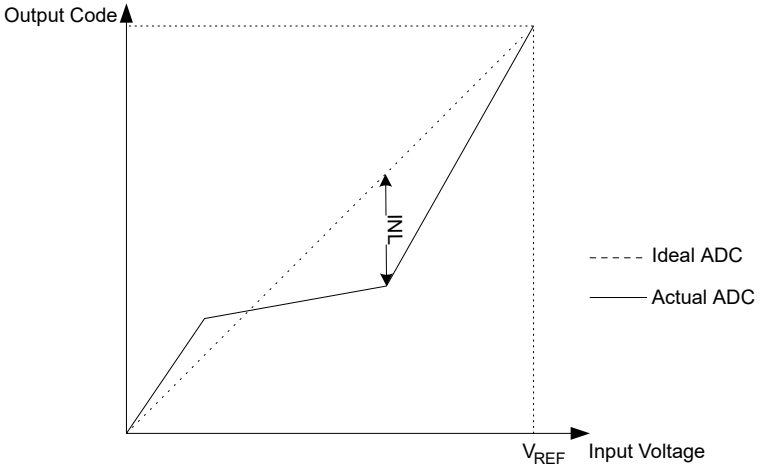
Gain Error After adjusting for offset, the gain error is found as the deviation of the last transition (e.g., 0x3FE to 0x3FF for a 10-bit ADC) compared to the ideal transition (at 1.5 LSb below maximum). Ideal value: 0 LSb.

Figure 6-2. Gain Error



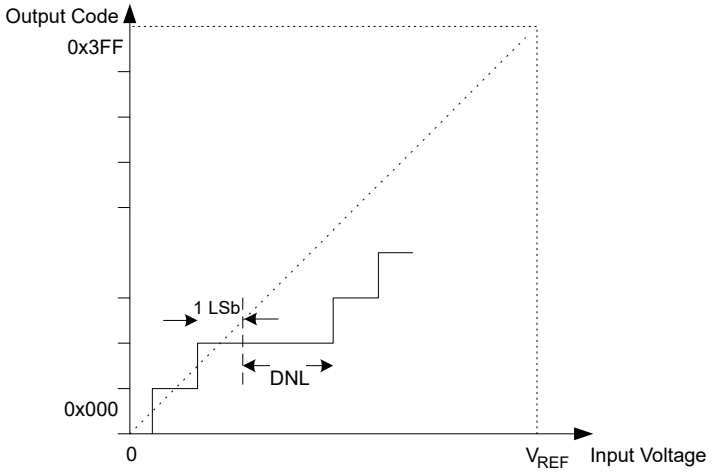
Integral Nonlinearity (INL) After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSb.

Figure 6-3. 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 6-4. 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.

7. AVR® CPU

7.1. 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 Limit Check
- Illegal Opcode Check
- Stack Pointer Accessible in I/O Memory Space
- Direct Addressing of up to 64 KB of 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

7.2. Overview

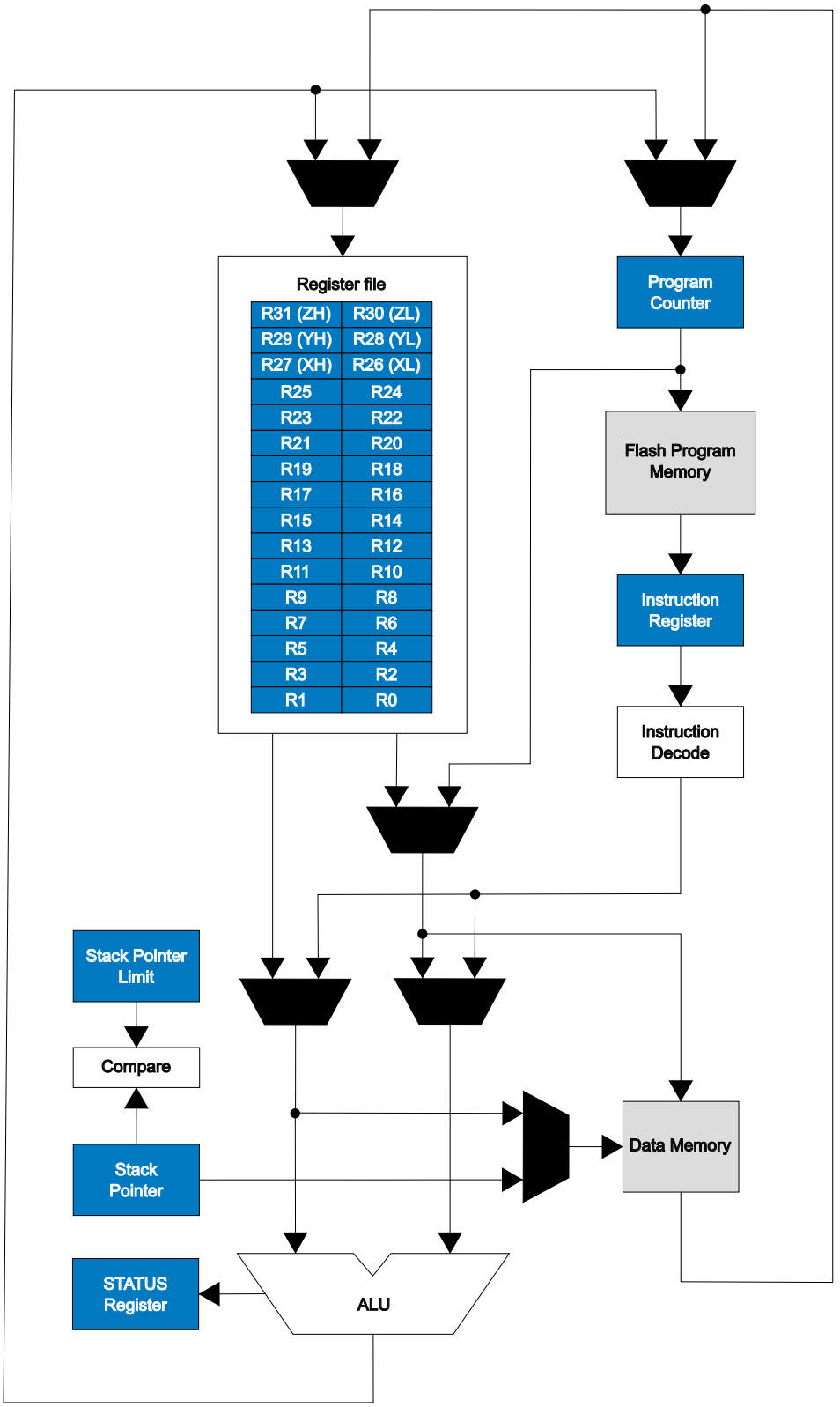
The AVR CPU can access memories, perform calculations, control peripherals, execute instructions from the program memory, and handle interrupts. Interrupt handling is described in a separate section.

7.3. Architecture

The AVR CPU uses a Harvard architecture with separate buses for program and data to maximize performance and parallelism. The instructions in the program memory execute with a single-level pipeline. While one instruction executes, the next instruction is prefetched from the program memory, enabling instructions to be executed on every clock cycle.

Refer to the *Instruction Set Summary* section for an overview of all AVR instructions.

Figure 7-1. AVR CPU Architecture



7.3.1. 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 connection with all the 32 general-purpose working registers in the register file. The arithmetic operations between working registers or between a working register and an immediate operand execute in a single clock cycle, and the result is stored in the register file. After an arithmetic or logic operation, the CPU's Status Register (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 an efficient implementation of 32-bit arithmetic, while the hardware multiplier supports signed and unsigned multiplication and fractional formats.

7.3.1.1. 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.

7.4. Functional Description

7.4.1. Program Flow

After a reset, the CPU will execute instructions from the lowest address in the flash program memory, `0x0000`. The instruction address on the fetch bus is masked modulo the flash size so that the instruction fetched after the instruction at the last address in the Flash will be the instruction at the first address in the Flash. In other words, the address space wraps around, making it possible to use relative jump instructions past the start and end of the Flash. Note that masking using modulo two assumes that the flash size is a power of two (2^n) to work as intended.

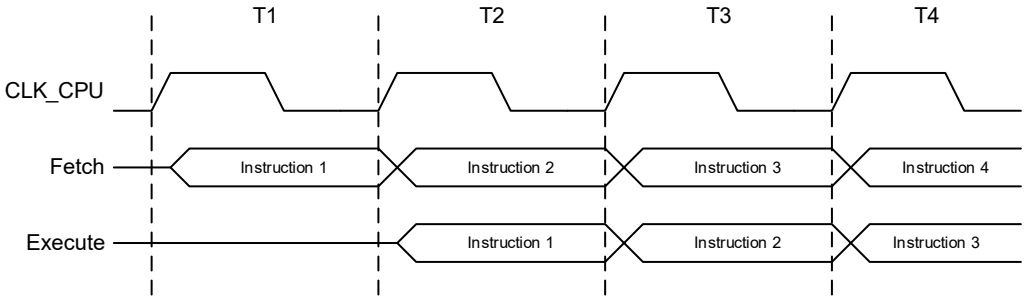
Program flow is supported by conditional and unconditional JUMP and CALL instructions, capable of directly addressing the whole address space. Most AVR instructions use a 16-bit word format, and a limited number use a 32-bit format.

During interrupts and subroutine calls, the return address program counter (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 limited only by the total SRAM size and the usage of the SRAM. After a reset, the Stack Pointer (SP) 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 SRAM data can easily be accessed through the five different addressing modes supported by the AVR CPU. See the *Instruction Set Summary* section for details.

7.4.2. Instruction Execution Timing

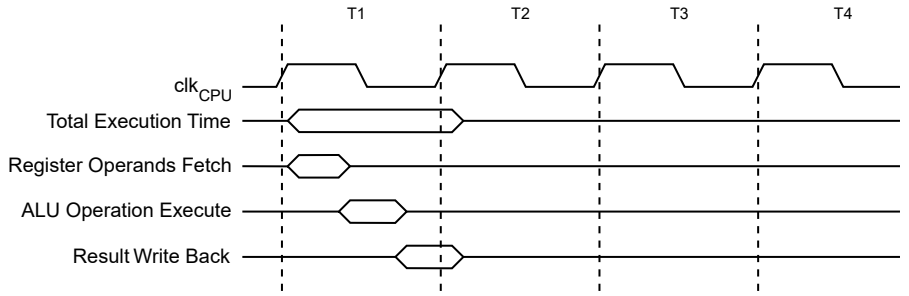
The CPU clock, CLK_CPU, clocks the AVR CPU. No internal clock division is applied. The figure below shows the parallel instruction fetches and executions enabled by the Harvard architecture and the fast-access register file concept, which is the basic pipelining concept enabling up to 1 MIPS/MHz performance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

Figure 7-2. The Parallel Instruction Fetches and Executions



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

Figure 7-3. Single Cycle ALU Operation



7.4.3. Status Register

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

CPU.SREG is updated after all ALU operations, as specified in the *Instruction Set Summary* section, which 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.

7.4.4. Stack and Stack Pointer

The stack is used to store return addresses after interrupts and subroutine calls and for storing temporary data. The Stack Pointer (SP) always points to the top of the stack. The address pointed to by the SP is stored in the Stack Pointer (CPU.SP) register. The CPU.SP is implemented as two 8-bit registers accessible in the I/O memory space.

Data are pushed and popped from the stack using the instructions in the table below or by executing interrupts. The stack grows from higher to lower memory locations, implying that pushing data to the stack will decrease the SP, and popping data from the stack will increase the SP.

The SP is automatically set to the highest address of the internal SRAM after a reset. If the stack needs to be allocated to a different SRAM address location (or if multiple stacks are used), the address must fall within the SRAM address space, with sufficient space reserved for the anticipated stack size. See the *SRAM Data Memory* topic in the *Memories* section for the SRAM start address and SRAM size. The new SP must be defined before any subroutine calls execute and interrupts are enabled. See the table below for SP details.

Table 7-1. 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 either a subroutine or an interrupt

During interrupts or subroutine calls, the return address is pushed automatically on the stack as a word, 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). For 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 one when data are pushed on the stack with the PUSH instruction and incremented by one when data are removed from 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.

7.4.4.1. Stack Pointer Limit Check

The Stack Pointer Limit (SPLIM) register can be programmed with the lower stack limit, meaning the lowest SRAM address location for the stack. The Stack Pointer (SP) is compared continuously to the SPLIM. Whenever SP < SPLIM, a limit check signal is asserted, and the Stack Pointer Limit Error (SPLIM) flag in the Interrupt Flags (INTFLAGS) register is set.

The Stack Pointer Limit Check will also detect a stack underflow, which occurs when a POP instruction makes the SP wrap around from address 0xFFFF to address 0x0000 (the POP instruction increases the SP by one, as described previously).

7.4.5. Register File

The register file consists of 32 8-bit general purpose working registers used by the CPU. The register file is 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 7-4. AVR® CPU General Purpose Working Registers

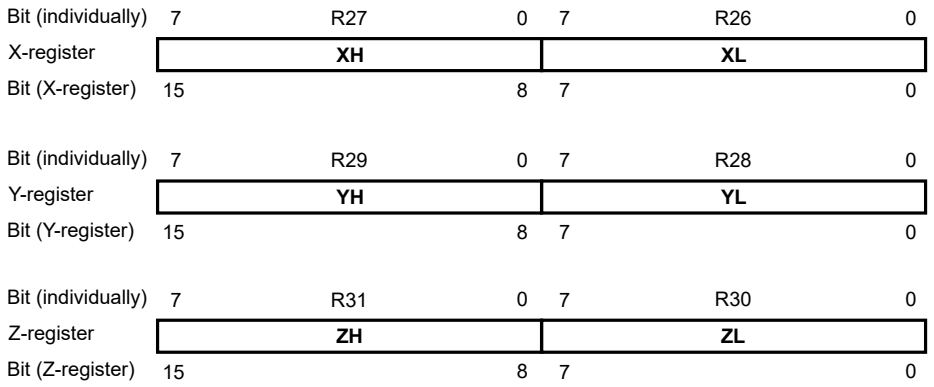
7	0	Addr.	
R0		0x00	
R1		0x01	
R2		0x02	
...			
R13		0x0D	
R14		0x0E	
R15		0x0F	
R16		0x10	
R17		0x11	
...			
R26		0x1A	X-register Low Byte
R27		0x1B	X-register High Byte
R28		0x1C	Y-register Low Byte
R29		0x1D	Y-register High Byte
R30		0x1E	Z-register Low Byte
R31		0x1F	Z-register High Byte

7.4.5.1. 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 an Address Pointer for program memory.

Figure 7-5. 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.

7.4.6. Configuration Change Protection (CCP)

System-critical I/O register settings are protected from accidental modification. Flash self-programming is protected from unexpected execution. This is handled globally by the Configuration Change Protection (CCP) register. An attempted write to a CCP-protected Special Function Register (SFR) outside of a CCP sequence will be discarded .

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).

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 ordinary, and the request is kept pending. After finalizing the CCP period, any pending interrupts are executed according to their level and priority.

The following operations will terminate any active configuration change period immediately:

- Any data write instruction to SFR registers (such as `ST`, `OUT`, `SBI`, ...)
- `SLEEP`
- `LPM`/`SPM`
- Any `LD`/`ST` to flash

It is not possible to restart a configuration change period that has not yet completed by writing to the CPU.CCP register. Any writes to this register before the period has ended will be disregarded. The period must either run out or be aborted by any of the operations listed above.

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

7.4.6.1. Sequence for Write Operation to Configuration Change Protected (CCP) I/O Registers

The following steps are required to write to CCP-protected I/O registers:

1. The software writes the signature that enables a change of protected I/O registers to the CCP bit field in the CPU.CCP register.
2. Within the next four CPU instructions executed, the software must write the appropriate data to the protected register.

The configuration change protection re-enables automatically after the CPU executes the write instruction or after four other CPU instructions execute.

7.4.6.2. Sequence for Execution of Self-Programming

Self-programming in this context means the execution of writes (commands) to the Non-Volatile Memory Controller (NVMCTRL). The following steps are required to execute self-programming:

1. The software enables self-programming by writing the Self-Program Memory (SPM) signature to the CCP register (CPU.CCP).
2. Within the next four CPU instructions, the software must execute the appropriate command in the NVM Controller.

The configuration change protection re-enables automatically after the CPU executes the write instruction or after four other CPU instructions execute.

7.4.6.3. Accessing CCP Protected Registers in the CPU

Some of the CPU Special Function Registers (SFRs) are CCP-protected. If accessed outside a CCP sequence, the access is discarded.

7.4.7. Illegal Opcode

Attempted execution of an illegal opcode results in a `NOP` operation and the setting of the Illegal Opcode Error (OPC) flag in the Interrupt Flags (INTFLAGS) register. A `BREAK` instruction is considered an Illegal Opcode when the device is LOCKED.

7.4.8. On-Chip Debug Capabilities

The AVR CPU includes native On-Chip Debug (OCD) support. It contains 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 *UPDI - Unified Program and Debug Interface* section for details about OCD.

7.4.9. Interrupts

Table 7-2. Available Interrupts Vectors and Sources

Name	Vector Description	Interrupt Flag	Conditions
CPU	CPU Interrupt	SPLIM	Stack Pointer Limit Error
		BUSERR	Bus Error
		OPC	Illegal Opcode Error

The Interrupt Flags (INTFLAGS) register of the CPU is connected to the CPU Interrupt Controller (CPUINT) as Non-Maskable Interrupts (NMIs). The CPU does not have a corresponding Interrupt Control register since NMIs can, by design, not be masked or disabled. For more information about NMIs, see the *CPUINT - CPU Interrupt Controller* section.

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

7.4.10. Events

None.

7.4.11. Sleep Mode Operation

The CPU will halt in all sleep modes.

7.4.12. CPU Registers Under Configuration Change Protection

The CPU has registers that are under Configuration Change Protection (CCP). In order to write to these registers, a specific key must first be written to the CPU's CCP register, followed by a write access to the protected register within four CPU instructions.

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

The following registers are under CCP:

Table 7-3. CPU - Registers under Configuration Change Protection

Register	Key
CPU.CTRLA	IOREG
CPU.INTFLAGS	IOREG

7.5. Functional Safety

The Functional Safety aspects of the CPU are detailed in the *Functional Safety Concept* section.

7.6. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	SPLIM	7:0	SPLIML[7:0]							
		15:8	SPLIMH[15:8]							
0x02 ... 0x03	Reserved									
0x04	CCP	7:0	CCP[7:0]							
0x05	CTRLA	7:0								SPLOCK
0x06	INTFLAGS	7:0						SPLIM		OPC
0x07 ... 0x0C	Reserved									
0x0D	SP	7:0	SPL[7:0]							
		15:8	SPH[7:0]							
0x0F	SREG	7:0	I	T	H	S	V	N	Z	C

7.7. Register Description

7.7.1. Stack Pointer Limit

Name: SPLIM
Offset: 0x0
Reset: Top of stack
Property: -

The Stack Pointer Limit Register (SPLIM) marks the lower limit of the stack. If the Stack Pointer (SP) has a value lower than SPLIM, a limit check signal is asserted, and the Stack Pointer Limit Error (SPLIM) flag in the Interrupt Flags (INTFLAGS) register is set.

The CPU.SPLIML and CPU.SPLIMH register pair represents the 16-bit value, CPU.SPLIM. 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
	SPLIMH[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
	SPLIML[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bits 15:8 – SPLIMH[15:8] Stack Pointer Limit High Byte
These bits hold the MSB of the 16-bit register. The contents of this register can be locked, preventing it from further updates, by the SPLOCK bit in the Control A (CTRLA) register.

Bits 7:0 – SPLIML[7:0] Stack Pointer Limit Low Byte
These bits hold the LSB of the 16-bit register. The contents of this register can be locked, preventing it from further updates, by the SPLOCK bit in the Control A (CTRLA) register.

7.7.2. Configuration Change Protection

Name: CCP
Offset: 0x04
Reset: 0x0
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 writing to protected I/O registers or executing protected instructions within the following four CPU instructions. During this period all interrupts are suspended, but the interrupt flags are set. After completing the period, any pending interrupts execute according to their level and priority.

After writing the protected I/O register signature, CCP[0] will read as '1' as long as the CCP feature is active.

After writing the protected self-programming signature, CCP[1] will read as '1' as long as the CCP feature is active.

CCP[7:2] will always read as '0'.

Value	Name	Description
0x9D	SPM	Allow self-programming
0xD8	IOREG	Un-protect protected I/O registers

7.7.3. Control A

Name: CTRLA
Offset: 0x5
Reset: 0x0
Property: Configuration Change Protection

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

Bit 0 – SPLOCK SPLIM Lock
Write this bit to '1' to lock SPLIM. When SPLIM is locked it is not possible to change the SPLIM value.

7.7.4. Interrupt Flags

Name: INTFLAGS
Offset: 0x6
Reset: 0xXX
Property: Configuration Change Protection

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

Bit 2 – SPLIM Stack Pointer Limit Error
This flag is set when the stack pointer limit is exceeded. In other words, when the stack has overflowed or underflowed.
Write this bit to '1' to clear the flag.
The CPU interrupt request will be asserted when this flag is set.

Bit 0 – OPC Illegal Opcode Error
This flag is set when an illegal opcode has been fetched and decoded.
Write this bit to '1' to clear the flag.
The CPU interrupt request will be asserted when this flag is set.

Note: This register has no accompanying INTCTRL register since all flags are registered as NMIs, which are always enabled.

Note: This register is under Configuration Change Protection since the interrupt flags may indicate a hardware error and are considered more severe than ordinary interrupt flags.

7.7.5. Stack Pointer

Name: SP
Offset: 0xD
Reset: 0x7FFF
Property: -

This 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 SRAM is implemented for each device. Unused bits will always read as '0'.

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

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

Bit	15	14	13	12	11	10	9	8
	SPH[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	SPL[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 – SPH[7:0] Stack Pointer High Byte
These bits hold the MSB of the 16-bit register SP.

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

7.7.6. Status Register

Name: SREG
Offset: 0xF
Reset: 0x0
Property: -

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

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
Writing a '1' to this bit enables interrupts on the device.
Writing a '0' to this bit disables the 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 Bit Copy Storage
The bit copy instructions, Bit Load (BLD) and Bit Store (BST), use the T bit as the source or destination for the operated bit.
A bit from a register in the register file can be copied into this bit by the BST instruction, and this bit can be copied into a bit in a register in the register file by the BLD instruction.

Bit 5 – H Half Carry Flag
The Half Carry (H) flag is set when there is a half carry in an arithmetic operation that supports this and is cleared otherwise. Half carry is helpful for performing BCD arithmetic.

Bit 4 – S Sign Bit, S = N ⊕ V
The Sign (S) bit is always an exclusive or (XOR) between the Negative flag (N) and the Two's Complement Overflow (V) flag.

Bit 3 – V Two's Complement Overflow Flag
The Two's Complement Overflow (V) flag is set when there is an overflow in the arithmetic operation that support this and is cleared otherwise.

Bit 2 – N Negative Flag
The Negative (N) flag is set when there is a negative result in the arithmetic or logic operation and is cleared otherwise.

Bit 1 – Z Zero Flag
The Zero (Z) flag is set when there is a zero result in the arithmetic or logic operation and is cleared otherwise.

Bit 0 – C Carry Flag
The Carry (C) flag is set when there is a carry in the arithmetic or logic operation and is cleared otherwise.

8. Memories

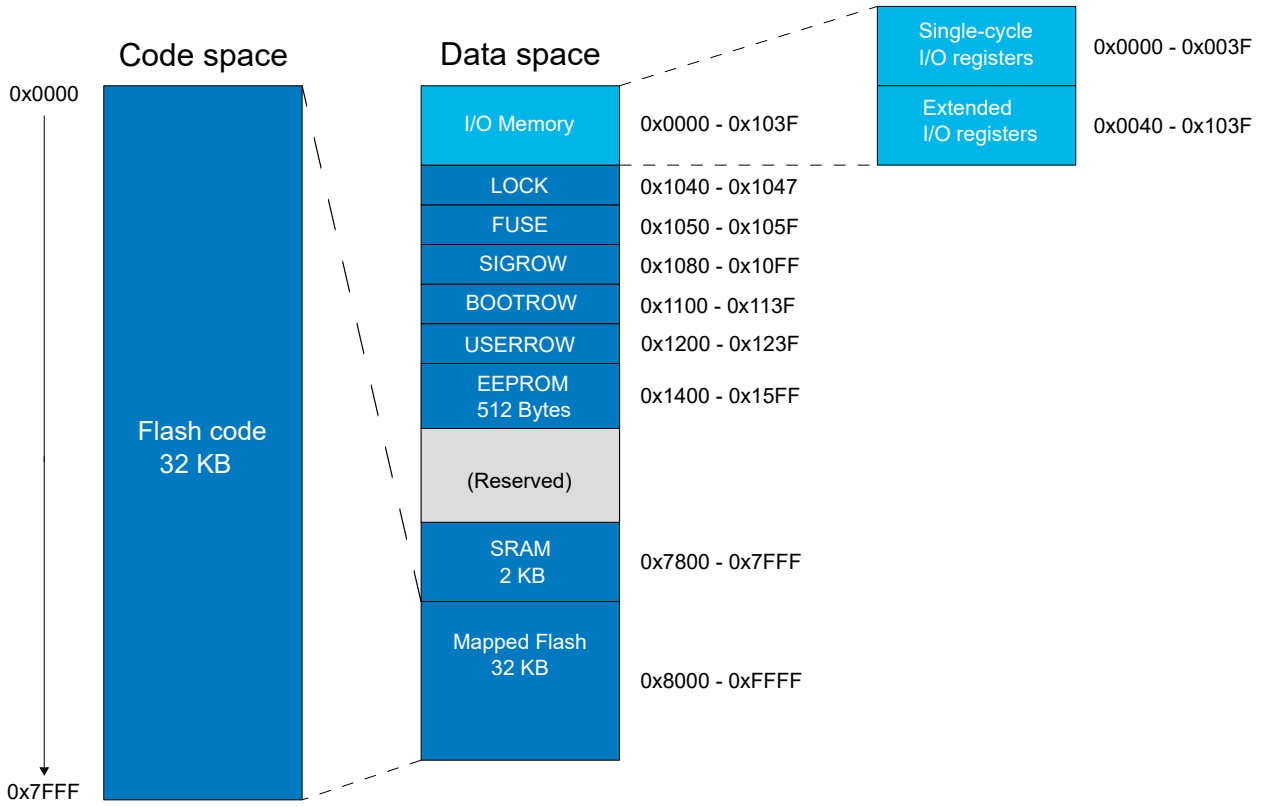
8.1. Overview

The main memories of the AVR16LA14/20/28/32 devices are the SRAM data memory space, the EEPROM data memory space, and the Flash program memory space. In addition, the peripheral registers are located in the I/O memory space.

8.2. Memory Map

The figure below shows the memory map for the largest memory derivative in the AVR LA family. Refer to the subsequent sections and the Peripheral Address Map table for further details.

Figure 8-1. Memory Map: Flash 32 KB, Internal SRAM 2 KB, EEPROM 512B



8.3. In-System Reprogrammable Flash Program Memory

The AVR16LA14/20/28/32 contains 16 KB on-chip, in-system reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized with a 16-bit data width. For write protection, the Flash program memory space can be divided into three sections: The Boot Loader (BOOT) section, the Application Code (APPCODE) section, and the Application Data (APPDATA) section. Code placed in one section may be restricted from writing to addresses in other sections. See the *NVMCTRL - Nonvolatile Memory Controller* chapter for more details.

The Program Counter (PC) can address the whole program memory. The procedure for writing Flash memory is described in detail in the NVMCTRL chapter.

The Flash memory is mapped into the data space and is accessible with ordinary `LD/ST` instructions. See the NVMCTRL section for details on which Flash section is mapped into the data space. For

LD/ST instructions, Flash is mapped from address 0x8000. The Flash memory can also be read with the LPM instruction; for LPM, the Flash start address is 0x0000.

The AVR16LA14/20/28/32 has a CRC module hosted on the data bus.

The table and figure below show the mapping of the physical and logical memory sections.

- **NRWW** = No-Read-While-Write
- **RWW** = Read-While-Write

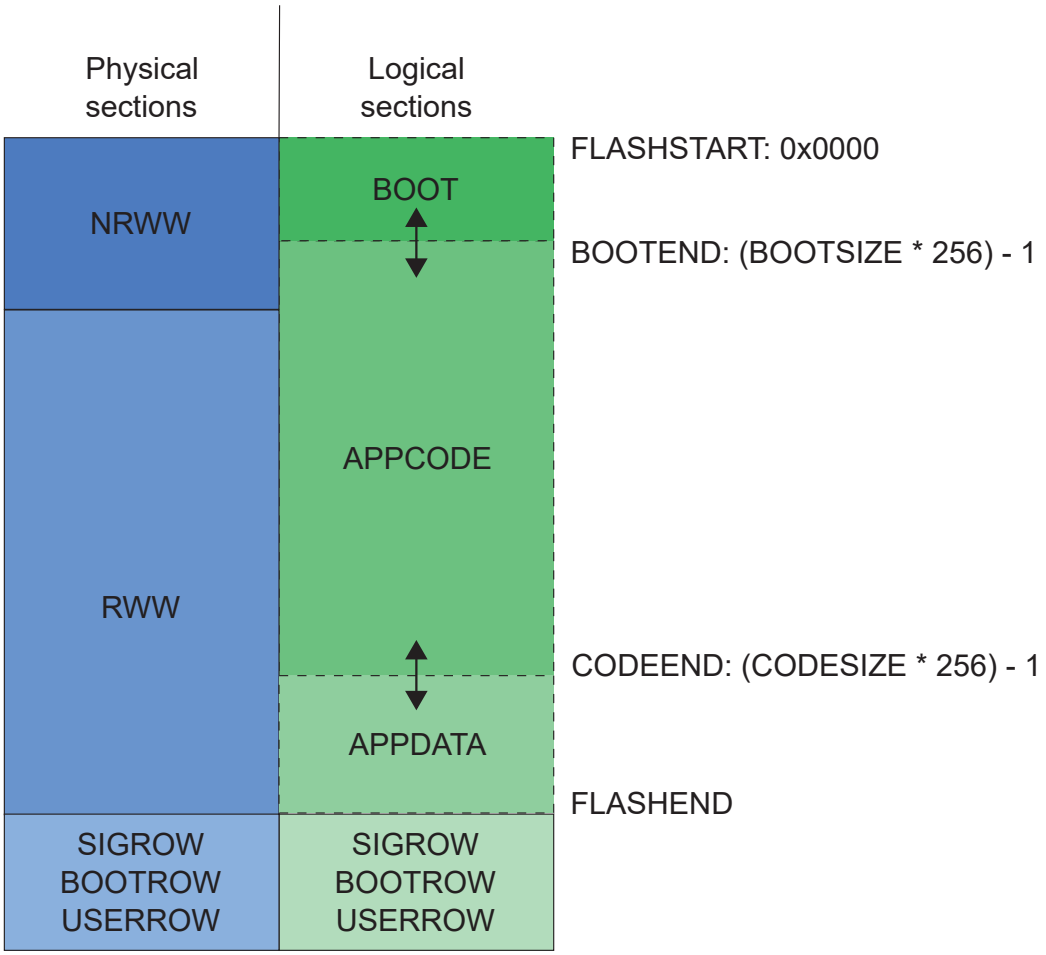
See the *NVMCTRL - Nonvolatile Memory Controller* chapter for more details about the different physical and logical memory sections and their configuration.

Table 8-1. Physical Properties of Flash Memory

Property	AVR16LA32 AVR16LA28 AVR16LA20 AVR16LA14
Size	16 KB
NRWW size	4 KB
RWW size	12 KB
Page size	64B
Number of pages	256
Start address in data space	0x8000
Start address in code space	0x0000

Note: The NRWW section always starts at the start address of the Flash memory.

Figure 8-2. Flash Sections



8.4. Program and Debug Interface Disable (PDID)

After activating the *Program and Debug Interface Disable (PDID)*, the only way to write to the reprogrammable Flash memory (Nonvolatile Memory - NVM) is from the Boot Code section of the NVM. Consequently, CHIPERASE or other reprogramming attempts through UPDI will fail. Reading any NVM content through UPDI will also fail.

Use the following procedure to enable the PDID feature (restrict access to NVM):

1. Write `0xB452` to the PDI Configuration (PDICFG) fuse:
   - Provide the NVM Protection Active (NVMACT) key by writing `0xB45` to bits PDICFG[15:4] (KEY)
   - Bits PDICFG[3:2] are unused; ensure they are set to zero
   - Select the Protection Level *NVM Access Disabled* (NVMACCDIS) by writing `0x2` to PDICFG[1:0] (LEVEL)
2. Write the Lock Key Bits (KEY) in the LOCK.KEY fuse to LOCKED.
3. Reset the device.

Once the protection level NVMACCDIS is invoked, the following access rules apply:

- NVM access through UPDI is disabled
- Updates to the application software can be performed only by code located in the Boot Code section (the bootloader)

- Chip Erase is disabled
- User Row write access is disabled
- CRC status will be available

WARNING Unlike locked devices, performing a CHIPERASE through the UPDI interface after the PDID feature is activated is impossible. The only way to alter NVM content after PDID activation is by executing NVM writes from the Boot Code section (the bootloader). The application software must ensure that the bootloader implementation fulfills the security requirements.

8.5. SRAM Data Memory

The primary task of SRAM is to store application data. In addition, the program stack is located at the end of SRAM.

It is not possible to execute code from SRAM.

Table 8-2. Physical Properties of SRAM

Property	AVR16LA32 AVR16LA28 AVR16LA20 AVR16LA14
Size	2 KB
Start address	0x7800
End address	0x7FFF

8.6. EEPROM Data Memory

The primary task of EEPROM memory is to store nonvolatile application data. The EEPROM supports single- and multi-byte read and write operations and is controlled by the Nonvolatile Memory Controller (NVMCTRL).

Table 8-3. Physical Properties of EEPROM

Property	AVR16LA32 AVR16LA28 AVR16LA20 AVR16LA14
Size	512B
Page size	8B
Number of pages	64
Start address	0x1400

8.7. SIGROW - Signature Row

The content of the Signature Row (SIGROW) fuses are preprogrammed and cannot be altered. SIGROW holds the Device ID, serial number, and calibration values.

All AVR16LA14/20/28/32 devices have a three-byte Device ID that identifies the device. This Device ID can be read using the UPDI interface, even when the device is locked. The three bytes reside in the Signature Row. The signature bytes are given in the following table.

Table 8-4. Device ID

Device Name	Signature Bytes Address		
	0x00	0x01	0x02
AVR16LA14	0x1E	0x94	0x54
AVR16LA20	0x1E	0x94	0x53
AVR16LA28	0x1E	0x94	0x52
AVR16LA32	0x1E	0x94	0x51

8.7.1. Register Summary - SIGROW

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	Reserved									
0x04	TEMPSENSE0	7:0	TEMPSENSE[7:0]							
		15:8	TEMPSENSE[15:8]							
0x06	TEMPSENSE1	7:0	TEMPSENSE[7:0]							
		15:8	TEMPSENSE[15:8]							
0x08 ... 0x0B	Reserved									
0x0C	OSCHF20TUNE	7:0	OSCHFTUNE[7:0]							
0x0D	OSCHF16TUNE	7:0	OSCHFTUNE[7:0]							
0x0E ... 0x0F	Reserved									
0x10	SERNUM0	7:0	SERNUM[7:0]							
...										
0x1F	SERNUM15	7:0	SERNUM[7:0]							

8.7.2. Signature Row Description

8.7.2.1. Device ID n

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

Each device has a Device ID that identifies the device and its properties, including memory sizes, pin count, and die revision. Use this ID to identify a device and its available features. 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

8.7.2.2. Temperature Sensor Calibration n

Name: TEMPSENSEn
Offset: 0x04 + n*0x02 [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. SIGROW.TEMPSENSE0 is a correction factor for the gain/slope (signed) and SIGROW.TEMPSENSE1 is a correction factor for the offset (signed).

Bit	15	14	13	12	11	10	9	8
	TEMPSENSE[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

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

Bits 15:0 – TEMPSENSE[15:0] Temperature Sensor Calibration
Refer to the *ADC - Analog-to-Digital Converter* chapter for using these registers.

8.7.2.3. OSCHF 20 MHz Tune

Name: OSCHF20TUNE
Offset: 0x0C
Reset: [Oscillator frequency tune value]
Property: -

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

Bits 7:0 – OSCHFTUNE[7:0] Oscillator Tuning
This bit field holds a fine-tuning value for the Internal High-Frequency Oscillator when operating at 20 MHz, which is programmed during production. To achieve optimal factory calibration accuracy of the internal High-Frequency Oscillator, it is suggested to load CLKCTRL.OSCHFTUNE bitfield with this fine tune value.

8.7.2.4. OSCHF 16 MHz Tune

Name: OSCHF16TUNE
Offset: 0x0D
Reset: [Oscillator frequency tune value]
Property: -

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

Bits 7:0 – OSCHFTUNE[7:0] Oscillator Tuning
This bit field holds a fine-tuning value for the Internal High-Frequency Oscillator when operating at 16 MHz, which is programmed during production. To achieve optimal factory calibration accuracy of the internal High-Frequency Oscillator, it is suggested to load CLKCTRL.OSCHFTUNE bitfield with this fine tune value.

8.7.2.5. Serial Number Byte n

Name: SERNUMn
Offset: 0x10 + n*0x01 [n=0..15]
Reset: [Byte n of device serial number]
Property: -

Each device has a unique serial number that represents a unique ID. This serial number can be used to identify a specific device in the field and consists of sixteen bytes: SIGROW.SERNUM[15:0].

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

8.8. BOOTROW - Boot Row

The AVR16LA14/20/28/32 devices have a special 64B memory section called the Boot Row (BOOTROW). The BOOTROW can be used for bootloader data and is erased by a chip erase.

On a locked device, this section can be accessed only when executing from the BOOT section, making it ideal for storing device-specific information for use by the bootloader. Refer to the *System Memory Address Map* for further details.

8.9. USERROW - User Row

The AVR16LA14/20/28/32 devices have a special 64B memory section called the User Row (USERROW). The User Row can be used for end-of-production data and is not affected by a chip erase. It can be written by the Unified Program and Debug Interface (UPDI) even if the device is locked, enabling storage of final configuration data without having access to any other memory. When the device is locked, UPDI is not allowed to read the contents of the User Row.

The CPU can read from and write to this memory as an ordinary Flash. Refer to the *System Memory Address Map* for further details.

8.10. FUSE - Configuration and User Fuses

Fuses are part of the nonvolatile memory and store the device configuration. The fuses can be read by both the CPU and UPDI, but only UPDI can program or clear them. 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 preprogrammed but can be altered by the user. Modified values in the configuration fuses will take effect only after a Reset.

The fuses are not affected by a chip erase.

Note: When writing the fuses, all reserved bits must be written to '0'.

8.10.1. Fuse Summary

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					OSCHFFRQ			
0x03	PINCFG	7:0							UPDIPINCFG	RSTPINCFG
0x04	HWMONCFG	7:0								CPUMON
0x05	SYSCFG0	7:0	CRCBOOT	CRCSEL					BOOTROWSAVE	EESAVE
0x06	SYSCFG1	7:0						SUT[2:0]		
0x07	CODESIZE	7:0	CODESIZE[7:0]							
0x08	BOOTSIZE	7:0	BOOTSIZE[7:0]							
0x09	Reserved									
0x0A	PDICFG	7:0	KEY[3:0]						LEVEL[1:0]	
		15:8	KEY[11:4]							

8.10.2. Fuse Description

8.10.2.1. Watchdog Configuration

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

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

Bit	7	6	5	4	3	2	1	0
	WINDOW[3:0]				PERIOD[3:0]			
Access	R	R	R	R	R	R	R	R
Default	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 at the end of the start-up sequence, after power-on or 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 at the end of the start-up sequence after power-on or Reset.

8.10.2.2. Brown-Out Detector Configuration

Name: BODCFG
Offset: 0x01
Default: 0x00
Property: -

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

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

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
Default	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.
Note: Values in the **Description** column are typical. Refer to the *Electrical Characteristics* chapter for further details.

Value	Name	Description
0x0	BODLEVEL0	1.65V
0x1	BODLEVEL1	1.90V
0x2	BODLEVEL2	2.60V
0x3	BODLEVEL3	4.30V
other	-	Reserved

Bit 4 – SAMPFREQ BOD Sample Frequency
This value is loaded into the Sample Frequency (SAMPFREQ) bit of the BOD Control A (BOD.CTRLA) register during Reset. Refer to the *BOD - Brown-out Detector* chapter for further details.

Value	Name	Description
0x0	128HZ	The sample frequency is 128 Hz
0x1	32HZ	The sample frequency is 32 Hz

Bits 3:2 – ACTIVE[1:0] BOD Operation in Active and Idle Sleep Mode
This value is loaded into the ACTIVE bit field of the BOD Control A (BOD.CTRLA) register during Reset. Refer to the *BOD - Brown-out Detector* chapter for further details.

Value	Name	Description
0x0	DISABLE	BOD disabled
0x1	ENABLE	BOD enabled in Continuous mode
0x2	SAMPLE	BOD enabled in Sampled mode
0x3	ENABLEWAIT	Enabled in continuous mode with code execution halted until the BOD is ready

Bits 1:0 – SLEEP[1:0] BOD Operation in Power Down and Standby Sleep Mode
The value is loaded into the SLEEP bit field of the BOD Control A (BOD.CTRLA) register during Reset. Refer to the *BOD - Brown-out Detector* chapter for further details.

Value	Name	Description
0x0	DISABLE	BOD disabled
0x1	ENABLE	BOD enabled in Continuous mode
0x2	SAMPLE	BOD enabled in Sampled mode
0x3	-	Reserved

8.10.2.3. Oscillator Configuration

Name: OSCCFG
Offset: 0x02
Default: 0x00
Property: -

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

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

Bit 3 – OSCHFFRQ Internal High-frequency Oscillator Frequency
This fuse bit controls the operating frequency of the internal high-frequency oscillator (OSCHF).

Value	Description
0	OSCHF running at 20 MHz
1	OSCHF running at 16 MHz

8.10.2.4. Pin Configuration

Name: PINCFG
Offset: 0x03
Default: 0x02
Property: -

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

Bit	7	6	5	4	3	2	1	0
							UPDIPINCFG	RSTPINCFG
Access							R/W	R/W
Default							1	0

Bit 1 – UPDIPINCFG Configuration of UPDI Pin at Start-Up
This fuse bit selects the UPDI pin configuration at start-up.

Value	Name	Description
0x0	GPIO	The UPDI pin is configured as GPIO
0x1	UPDI	The UPDI pin is configured as UPDI with pull-up enabled on PF7

Bit 0 – RSTPINCFG Configuration of Reset Pin at Start-up
This bit fuse controls the Reset pin configuration at start-up.

Value	Name	Description
0x0	INPUT	PF6 pin is configured as a general purpose input pin
0x1	RESET	PF6 pin is configured as an external reset pin with pull-up enabled

8.10.2.5. Hardware Monitor Configuration

Name: HWMONCFG
Offset: 0x04
Default: 0x00
Property: -

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

Bit	7	6	5	4	3	2	1	0
								CPUMON
Access								R/W
Default								0

Bit 0 – CPUMON CPU Hardware Monitor
This fuse bit controls whether the CPU safety functions checks for illegal operation codes (opcodes). Refer to the *CPU* section for more information about this functionality.

Value	Name	Description
0x0	DISABLE	Check for illegal opcode is disabled
0x1	ENABLE	Check for illegal opcode is enabled

8.10.2.6. System Configuration 0

Name: SYSCFG0
Offset: 0x05
Default: 0x00
Property: -

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

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

Bit 7 – CRCBOOT Cyclic Redundancy Check (CRC) on Boot Section During System Initialization
This fuse bit controls whether the boot section of Flash is checked by the CRCSCAN peripheral during the Reset initialization. Refer to the *CRCSCAN - Cyclic Redundancy Check Memory Scan* chapter for more information about functionality.

Value	Name	Description
0x0	DISABLE	No CRC
0x1	ENABLE	CRC of the boot section

Bit 6 – CRCSEL CRC Polynomial Selection
This fuse bit controls the type of CRC performed by the CRCSCAN peripheral. Refer to the *CRCSCAN - Cyclic Redundancy Check Memory Scan* chapter for more information about the functionality.

Value	Name	Description
0x0	CRC16	CRC16 - CCITT
0x1	CRC32	CRC32 (IEEE 802.3)

Bit 1 – BOOTROWSAVE BOOTROW Saved During Chip Erase
This fuse bit controls whether the BOOTROW will be erased or preserved during a Chip Erase.

Value	Name	Description
0x0	DISABLE	The BOOTROW is erased during a Chip Erase
0x1	ENABLE	The BOOTROW is preserved during a Chip Erase regardless of whether the device is locked

Bit 0 – EESAVE EEPROM Saved During Chip Erase
This fuse bit controls whether the EEPROM is erased or preserved during a Chip Erase. If enabled, only the Flash memory will be erased by a Chip Erase.

Value	Name	Description
0x0	DISABLE	The EEPROM is erased during a Chip Erase
0x1	ENABLE	The EEPROM is preserved during a Chip Erase regardless of whether the device is locked

8.10.2.7. System Configuration 1

Name: SYSCFG1
Offset: 0x06
Default: 0x07
Property: -

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

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

Bits 2:0 – SUT[2:0] Start-up Time
This fuse bit field controls the start-up time, meaning the time between power-on and the start of code execution.

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

8.10.2.8. Code Section Configuration

Name: CODESIZE
Offset: 0x07
Default: 0x00
Property: -

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

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

Bits 7:0 – CODESIZE[7:0] Code Section Size
This fuse bit field defines the combined size of the Boot Code section and the Application Code section in blocks of 256 bytes. For more details, refer to the *NVMCTRL - Nonvolatile Memory Controller* chapter.
Note: If FUSE.BOOTSIZE is `0x00`, the entire Flash is set as the Boot Code section, and the FUSE.CODESIZE value is not used.

8.10.2.9. Boot Section Configuration

Name: BOOTSIZE
Offset: 0x08
Default: 0x00
Property: -

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

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

Bits 7:0 – BOOTSIZE[7:0] Boot Section Size
This fuse bit field controls the size of the boot section in blocks of 256 bytes. A value of `0x00` defines the entire Flash as BOOT section.
For more details, refer to the *NVMCTRL - Nonvolatile Memory Controller* chapter.

8.10.2.10. Programming and Debug Interface Configuration

Name: PDICFG
Offset: 0x0A
Default: 0x0003
Property: -

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

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

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

Bits 15:4 – KEY[11:0] NVM Protection Activation Key
This fuse bit field contains the 12-bit key used to enable NVM protection. After it is enabled, performing a chip erase is no longer possible. The protection is not enabled unless the device also has code protection enabled (the KEY bit field of the LOCK register set to the LOCKED state).

Value	Name	Description
Other	—	NVM protection not active. Chip erase is working.
0xB45	NVMACT	NVM protection active. Chip erase is disabled.

Bits 1:0 – LEVEL[1:0] UPDI Protection Level
This fuse bit field controls the level of UPDI protection.

Value	Name	Description
Other	—	Reserved
0x2	NVMACCDIS	NVM access through UPDI is disabled. All erase and write commands to NVM must be executed from the Boot Code section (bootloader). Chip Erase and User Row writes are disabled. The CRC status remains available.
0x3	BASIC	The UPDI interface and UPDI pin function as normal. This is the default setting.

Important: There is no way to recover from this mode; once NVMACCDIS is enabled, it is not possible to re-enable UPDI access for device programming. Ensure that the consequences of enabling this feature are fully understood before enabling this mode.

WARNING After activation of NVMACCDIS, advanced failure analysis will not be possible.

8.11. LOCK - Memory Sections Access Protection

The device can be locked so that the memories cannot be read using the Unified Program and Debug Interface (UPDI). Locking protects Flash (all Boot Code, Application Code, and Application

Data sections), SRAM, and the EEPROM, including the FUSE data, thereby preventing application data or code from being read through the debugger interface. Regular memory access from within the application remains enabled.

The CPU and UPDI can read the Lock Key, but it can be programmed or cleared only by UPDI. The device is locked by writing an invalid key to the Lock Key (LOCK.KEY) register.

Table 8-5. Memory Access Unlocked (LOCK.KEY Valid Key)[1]

Memory Section	CPU Access		UPDI Access	
	Read	Write	Read	Write
Flash	Yes	Yes	Yes	Yes
SRAM	Yes	Yes	Yes	Yes
EEPROM	Yes	Yes	Yes	Yes
SIGROW	Yes	No	Yes	No
BOOTROW	Yes[2]	Yes[2]	Yes	Yes
USERROW	Yes	Yes	Yes	Yes
FUSE	Yes	No	Yes	Yes
LOCK	Yes	No	Yes	Yes
Registers	Yes	Yes	Yes	Yes

Table 8-6. Memory Access Locked (LOCK.KEY Invalid Key)[1]

Memory Section	CPU Access		UPDI Access	
	Read	Write	Read	Write
Flash	Yes	Yes	No	No
SRAM	Yes	Yes	No	No
EEPROM	Yes	Yes	No	No
SIGROW	Yes	No	No	No
BOOTROW	Yes[2]	Yes[2]	No	No
USERROW	Yes	Yes	No	Yes[3]
FUSE	Yes	No	No	No
LOCK	Yes	No	No	No
Registers	Yes	Yes	No	No

Notes:

1. Read operations marked No in the tables may appear successful, but the data are invalid. Therefore, any attempt at code validation through UPDI will fail for these memory sections.
2. Write and read operations are possible only from the Boot section.
3. In Locked mode, the USERROW can be written using the Fuse Write command, but the current USERROW values cannot be read.

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

8.11.1. Register Summary - LOCK

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	KEY	7:0	KEY[7:0]							
		15:8	KEY[15:8]							
		23:16	KEY[23:16]							
		31:24	KEY[31:24]							

8.11.1.1. Lock Description

8.11.1.1.1. Lock Key

Name: KEY
Offset: 0x00
Reset: Initial factory value 0x5CC5C55C
Property: -

Bit	31	30	29	28	27	26	25	24
	KEY[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bit	23	22	21	20	19	18	17	16
	KEY[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

Bit	15	14	13	12	11	10	9	8
	KEY[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	x	x	x	x	x	x	x	x

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

Bits 31:0 – KEY[31:0] Lock Key
This bit field controls whether the device is locked.

Value	Name	Description
0x5CC5C55C	UNLOCKED	Device unlocked
Other	LOCKED	Device locked

8.12. I/O Memory

All AVR16LA14/20/28/32 devices have their I/O and peripheral registers located in the I/O memory space. Refer to the *Peripheral Address Map* table for further details.

For compatibility with future devices, when writing to a register that contains reserved bits, those bits must be written as '0'. Reserved I/O memory addresses must never be written.

Single-Cycle I/O Registers

The I/O memory range from 0x00 to 0x3F can be accessed by single-cycle CPU instructions using the IN or OUT instructions.

The peripherals available in the single-cycle I/O registers are as follows:

- VPORTx
  - Refer to the *PORT I I/O Configuration* chapter for further details
- GPR
  - Refer to the *GPR - General Purpose Register* chapter for further details
- CPU
  - Refer to the *AVR CPU* chapter for further details

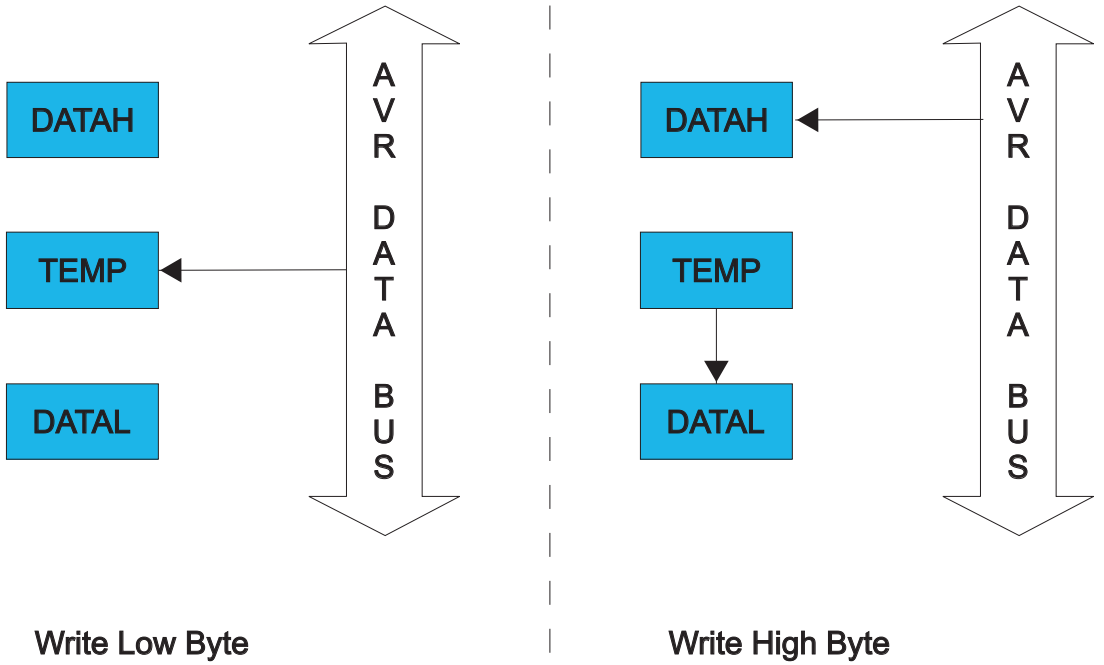
The single-cycle I/O registers ranging from `0x00` to `0x1F` (VPORTx and GPR) are also directly bit-accessible using the `SBI` or `CBI` instruction. In these single-cycle I/O registers, individual bits can be checked using the `SBIS` or `SBIC` instruction.

Refer to the *Instruction Set Summary* chapter for further details.

8.12.1. Accessing 16-Bit Register

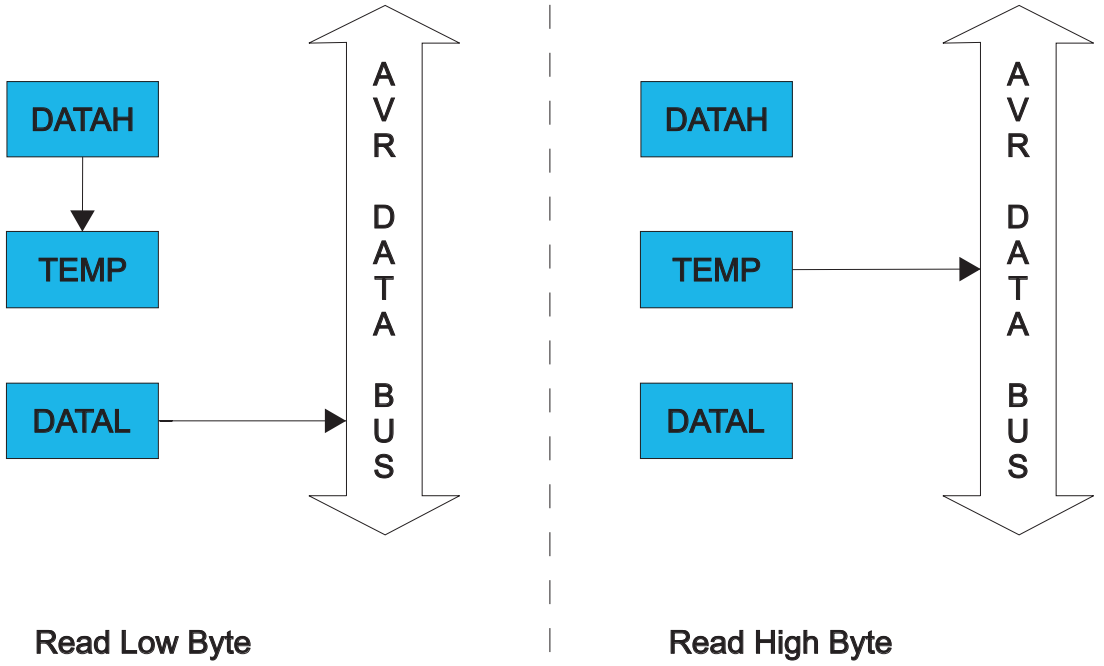
Most registers in AVR16LA14/20/28/32 devices are 8-bit registers, but they also feature a few 16-bit registers. Since the AVR data bus has a width of eight bits, accessing 16-bit requires two read or write operations. All 16-bit registers in AVR16LA14/20/28/32 devices are connected to the 8-bit bus through a temporary (TEMP) register.

Figure 8-3. 16-Bit Register Write Operation



For a 16-bit write operation, the low-byte register (for example, DATAL) of the 16-bit register must be written before the high-byte register (for example, DATAH). Writing the low-byte register results in a write to the temporary (TEMP) register instead of directly to the low-byte register, as shown on the left side of the figure above. When the high-byte register of the 16-bit register is written, the content of TEMP are copied into the low-byte of the 16-bit register in the same clock cycle, as shown on the right side of the figure.

Figure 8-4. 16-Bit Register Read Operation



For a 16-bit read operation, the low-byte register (for example, DATAL) of the 16-bit register must be read before the high-byte register (for example, 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 shown on the left side of the figure above. Reading the high-byte register then returns the value from TEMP instead of the high-byte register, as shown on the right side of the same figure.

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

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

8.12.2. Accessing 24- and 32-Bit Registers

For 24- and 32-bit registers, read and write access is performed in the same way as described for 16-bit registers, except that there are two temporary registers for 24-bit registers and three temporary registers for 32-bit registers. The Most Significant Byte (MSB) must be written last when writing to the register, and the Least Significant Byte (LSB) must be read first when reading the register.

9. GPR - General Purpose Registers

AVR16LA14/20/28/32 devices provide four General Purpose Registers (GPRs). These registers can be used to store any type of information and are particularly useful for storing global variables and interrupt flags. No implicit or explicit semantics apply to the bits in the General Purpose Registers; the interpretation of the bit values is determined by software.

General Purpose Registers, which reside in the address range 0x001C–0x001F, are directly bit-accessible using the `SBI`, `CBI`, `SBIS`, and `SBIC` instructions.

9.1. Register Summary - GPR

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	GPR0	7:0	GPR[7:0]							
0x01	GPR1	7:0	GPR[7:0]							
0x02	GPR2	7:0	GPR[7:0]							
0x03	GPR3	7:0	GPR[7:0]							

9.2. Register Description

9.2.1. General Purpose Register n

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

There are four General Purpose Registers (GPRs) 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
	GPR[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 – GPR[7:0] General Purpose Register Byte

10. Peripherals and Architecture

10.1. Peripheral Address Map

The address map shows the base address for each peripheral. Refer to the respective peripheral sections for a complete register description and summary for each peripheral.

Table 10-1. Peripheral Address Map

Base Address	Name	Description	14-Pin	20-Pin	28-Pin	32-Pin
0x0000	VPORTA	Virtual Port A	X	X	X	X
0x0008	VPORTC	Virtual Port C	X	X	X	X
0x000C	VPORTD	Virtual Port D	X	X	X	X
0x0014	VPORTF	Virtual Port F	X	X	X	X
0x001C	GPR	General Purpose Registers	X	X	X	X
0x0030	CPU	CPU	X	X	X	X
0x0040	RSTCTRL	Reset Controller	X	X	X	X
0x0050	SLPCTRL	Sleep Controller	X	X	X	X
0x0060	CLKCTRL	Clock Controller	X	X	X	X
0x00A0	BOD	Brown-out Detector	X	X	X	X
0x00B0	VREF	Voltage Reference	X	X	X	X
0x0100	WDT	Watchdog Timer	X	X	X	X
0x0110	CPUINT	Interrupt Controller	X	X	X	X
0x0120	CRCSCAN	Cyclic Redundancy Check Memory Scan	X	X	X	X
0x0140	RTC	Real-Time Counter	X	X	X	X
0x01C0	CCL	Configurable Custom Logic	X	X	X	X
0x0200	EVSYS	Event System	X	X	X	X
0x0400	PORTA	Port A Configuration	X	X	X	X
0x0440	PORTC	Port C Configuration	X	X	X	X
0x0460	PORTD	Port D Configuration	X	X	X	X
0x04A0	PORTF	Port F Configuration	X	X	X	X
0x05E0	PORTMUX	Port Multiplexer	X	X	X	X
0x0600	ADC0	Analog-to-Digital Converter	X	X	X	X
0x0680	AC0	Analog Comparator	X	X	X	X
0x0800	USART0	Universal Synchronous Asynchronous Receiver Transmitter	X	X	X	X
0x0900	TWI0	Two-Wire Interface	X	X	X	X
0x0940	SPI0	Serial Peripheral Interface	X	X	X	X
0x0A00	TCE0	Timer/Counter Type E instance 0	X	X	X	X
0x0B00	TCB0	Timer/Counter Type B instance 0	X	X	X	X
0x0B10	TCB1	Timer/Counter Type B instance 1	X	X	X	X
0x0F00	SYSCFG	System Configuration	X	X	X	X
0x0F80	OCD	On-Chip Debug	X	X	X	X
0x1000	NVMCTRL	Nonvolatile Memory Controller	X	X	X	X

Table 10-2. System Memory Address Map

Base Address	Name	Description	14-Pin	20-Pin	28-Pin	32-Pin
0x1040	LOCKBIT	Lock bits	X	X	X	X
0x1050	FUSE	User Configuration Fuses	X	X	X	X

Table 10-2. System Memory Address Map (continued)

Base Address	Name	Description	14-Pin	20-Pin	28-Pin	32-Pin
0x1080	SIGROW	Signature Row	X	X	X	X
0x1100	BOOTROW	Boot Row	X	X	X	X
0x1200	USERROW	User Row	X	X	X	X

10.2. Interrupt Vector Mapping

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

An interrupt flag is set in the interrupt flag (<*peripheral*>.INTFLAGS) register of the peripheral when the interrupt condition occurs, even if the interrupt is not enabled.

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

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. For details on how to clear interrupt flags, see the peripheral's interrupt flag (<*peripheral*>.INTFLAGS) register.

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

Table 10-3. Interrupt Vector Mapping

Vector Number	Program Address (Word)	Interrupt Vector (Name)	Interrupt Source (Name)	Description	14-Pin	20-Pin	28-Pin	32-Pin
0	0x00	RESET	-	Device Reset interrupt	X	X	X	X
1	0x02	NMI	OPC	Non-Maskable Illegal Opcode interrupt from CPU	X	X	X	X
			SPLIM	Non-Maskable Stack Pointer Limit interrupt from CPU	X	X	X	X
			ERROR	Non-Maskable interrupt available for CRCSCAN	X	X	X	X
2	0x04	BOD_VLM	VLM	Voltage Level Monitoring interrupt from BOD	X	X	X	X
3	0x06	CRCSCAN_INT	DONE	Scan Done interrupt from CRCSCAN	X	X	X	X
			PERIOD	Scan Period Done interrupt from CRCSCAN	X	X	X	X
4	0x08	RTC_CNT	CMP	Compare interrupt from RTC	X	X	X	X
			OVF	Overflow interrupt from RTC	X	X	X	X
5	0x0A	RTC_PIT	PIT	Periodic Interrupt Timer interrupt from RTC	X	X	X	X
6	0x0C	CCL_CCL	LUTn	LUTn interrupt from CCL	X	X	X	X
7	0x0E	PORTA_PORT	PAn	Pin n interrupt from PORTA	X	X	X	X
8	0x10	TCE0_OVF	OVF	Overflow interrupt from TCE0	X	X	X	X
9	0x12	TCE0_CMP0	CMP0	Compare 0 interrupt from TCE0	X	X	X	X
10	0x14	TCE0_CMP1	CMP1	Compare 1 interrupt from TCE0	X	X	X	X

Table 10-3. Interrupt Vector Mapping (continued)

Vector Number	Program Address (Word)	Interrupt Vector (Name)	Interrupt Source (Name)	Description	14-Pin	20-Pin	28-Pin	32-Pin
11	0x16	TCE0_CMP2	CMP2	Compare 2 interrupt from TCE0	X	X	X	X
12	0x18	TCB0_INT	CAPT	Capture interrupt from TCB0	X	X	X	X
			OVF	Overflow interrupt from TCB0	X	X	X	X
13	0x1A	TCB1_INT	CAPT	Capture interrupt from TCB1	X	X	X	X
			OVF	Overflow interrupt from TCB1	X	X	X	X
14	0x1C	TWI0_TWIC	DIF	Client Data Transmit or Receive Complete interrupt from TWI0 in Client mode	X	X	X	X
			APIF	Client Address or Stop interrupt from TWI0 in Client mode	X	X	X	X
15	0x1E	TWI0_TWIH	RIF	Host Read Complete interrupt from TWI0 in Host mode	X	X	X	X
			WIF	Host Write Complete interrupt from TWI0 in Host mode	X	X	X	X
16	0x20	SPI0_INT	RXC	Receive Complete interrupt from SPI0 in Buffered mode	X	X	X	X
			TXC	Transmit Complete interrupt from SPI0 in Buffered mode	X	X	X	X
			DRE	Data Register Empty interrupt from SPI0 in Buffered mode	X	X	X	X
			SS	Slave Select interrupt from SPI0 in Buffered mode	X	X	X	X
			IF	Transmit Complete interrupt from SPI0 in Normal mode	X	X	X	X
			WRCOL	Write Collision interrupt from SPI0 in Normal mode	X	X	X	X
17	0x22	USART0_ERR	ISF	Auto-Baud Error/Invalid Sync Field interrupt from USART0	X	X	X	X
			PERR	Parity Error interrupt from USART0	X	X	X	X
			FERR	Frame Error interrupt from USART0	X	X	X	X
			BUFOVF	Buffer Overflow interrupt from USART0	X	X	X	X
			COLL	Transmit Collision Detection interrupt from USART0	X	X	X	X
18	0x24	USART0_RXC	RXC	Receive Complete interrupt from USART0	X	X	X	X
			RXS	Receive Start-of-Frame interrupt from USART0	X	X	X	X
			RXBRK	Receive Valid Break and Synchronization interrupt from USART0	X	X	X	X

Table 10-3. Interrupt Vector Mapping (continued)

Vector Number	Program Address (Word)	Interrupt Vector (Name)	Interrupt Source (Name)	Description	14-Pin	20-Pin	28-Pin	32-Pin
19	0x26	USART0_DRE	DRE	Data Register Empty interrupt from USART0	X	X	X	X
			CTSIC	Clear to Send Input Change interrupt from USART0	X	X	X	X
20	0x28	USART0_TXC	TXC	Transmit Complete interrupt from USART0	X	X	X	X
21	0x2A	PORTD_PORT	PDn	Pin n interrupt from PORTD	X	X	X	X
22	0x2C	AC0_AC	CMP	Compare interrupt from AC0	X	X	X	X
23	0x2E	ADC0_ERROR	TRIGOVR	Trigger Overrun interrupt from ADC0	X	X	X	X
			SAMPOVR	Sample Overwrite interrupt from ADC0	X	X	X	X
			RESOVR	Result Overwrite interrupt from ADC0	X	X	X	X
24	0x30	ADC0_RESRDY	RESRDY	Result Ready interrupt from ADC0	X	X	X	X
			WCMP	Window Compare interrupt from ADC0	X	X	X	X
25	0x32	ADC0_SAMPRDY	SAMPRDY	Sample Ready interrupt from ADC0	X	X	X	X
			WCMP	Window Compare interrupt from ADC0	X	X	X	X
26	0x34	PORTC_PORT	PCn	Pin n interrupt from PORTC	X	X	X	X
27	0x36	PORTF_PORT	PFn	Pin n interrupt from PORTF	X	X	X	X
28	0x38	NVMCTRL_NVMREADY	EEREADY	EEPROM Ready interrupt	X	X	X	X
			FLREADY	Flash Ready interrupt	X	X	X	X

10.3. SYSCFG - System Configuration

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

It also contains a register that selects the operating voltage range for some peripherals.

10.3.1. Register Summary - SYSCFG

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	Reserved									
0x01	REVID	7:0	MAJOR[3:0]				MINOR[3:0]			
0x02 ... 0x06	Reserved									
0x07	VDDCTRL	7:0								VDDR

10.3.2. Register Description

10.3.2.1. Device Revision ID

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

This register is read-only and displays the device revision ID. Revisions include A0, A1, .., B0, B1, .. , and so on.

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

Bits 7:4 – MAJOR[3:0] Major Revision
This bit field contains the major device revision. `0x01` = A, `0x02` = B, and so on.

Bits 3:0 – MINOR[3:0] Minor Revision
This bit field contains the minor device revision, starting at `0x0`.

10.3.2.2. VDD Range Control

Name: VDDCTRL
Offset: 0x07
Reset: 0x01
Property: -

This register is used to set the operational voltage range for certain peripherals.

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

Bit 0 – VDDR VDD Voltage Range
This bit is used to select the operating voltage range for peripherals. This configuration is used by:

- TWI
  - Used for setting the correct input levels for SMBus 3.0

Value	Name	Description
0x0	LOW	VDD in range 1.8V to 3.6V
0x1	HIGH	VDD in range 2.4V to 5.5V

11. NVMCTRL - Nonvolatile Memory Controller

11.1. Features

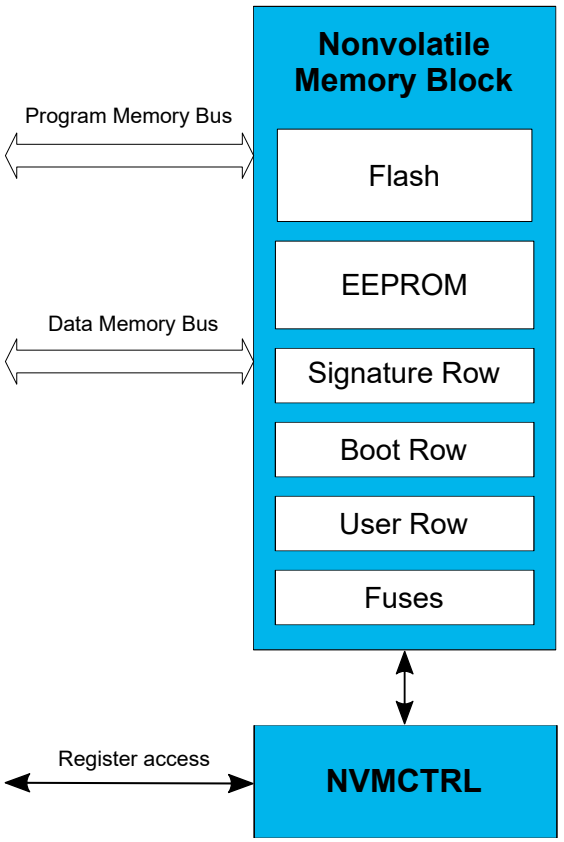
- In-System Programmable
- Self-Programming and Boot Loader Support
- True Read-While-Write Support
- Configurable Memory Sections:
  - Boot code section
  - Application code section
  - Application data section
- Signature Row for Factory-Programmed Data:
  - ID for each device type
  - Serial number 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 the UPDI on a locked device
  - Content is kept after a chip erase
- Boot Row for Boot Data:
  - For storage of device specific data
  - Accessible from the BOOT section only

11.2. 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 when not powered. The Flash is mainly used for program storage but can also be used for data storage. EEPROM, Signature Row, User Row, and fuses are used solely for data storage.

11.2.1. Block Diagram

Figure 11-1. NVMCTRL Block Diagram



11.3. Functional Description

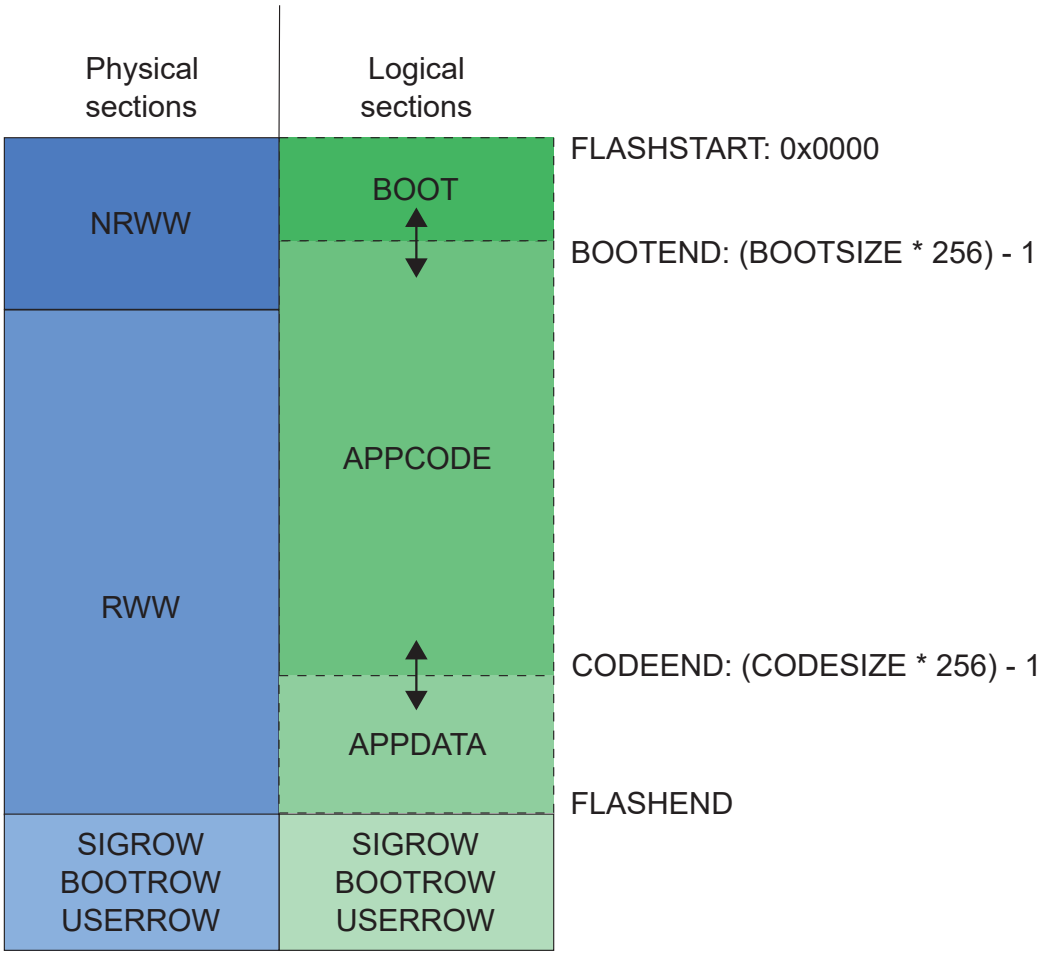
11.3.1. Memory Organization

11.3.1.1. Flash

The Flash is divided into a set of pages. A page is the smallest addressable unit when erasing the Flash. It is only possible to write or erase an entire page at a time. One page consists of 64 bytes.

Independent of page size, the Flash is organized in sections. The Flash is split into two physical sections to optimize the structure for boot loader applications: Read-While-Write (RWW) and No-Read-While-Write (NRWW). It is possible to divide these two sections into three logical sections: Boot Loader Code (BOOT), Application Code (APPCODE), and Application Data (APPDATA).

Figure 11-2. Flash Sections



11.3.1.1.1. Physical Sections

The Flash is divided physically into two fixed sections: A **Read-While-Write (RWW)** section and a **No-Read-While-Write (NRWW)** section.

The main differences between the two sections are as follows:

- When erasing or writing a page located inside the RWW Flash, the NRWW Flash can be read during the operation
- When erasing or writing a page located inside the NRWW Flash, the CPU is halted during the entire operation

The syntax “Read-While-Write section” refers to which section to program (erase or write), not the one read. Only the code located inside the NRWW Flash is accessible by either executing a CPU instruction or reading data while the RWW Flash is being written or erased.

Note: The interrupt code in the RWW section may halt the CPU if the associated interrupt is triggered while the RWW section is erased or written. Disable or place the interrupt code in the Boot Code (BOOT) section to avoid this.

The figures and table below explain these two physical Flash sections in detail:

Figure 11-3. Read-While-Write Scenarios

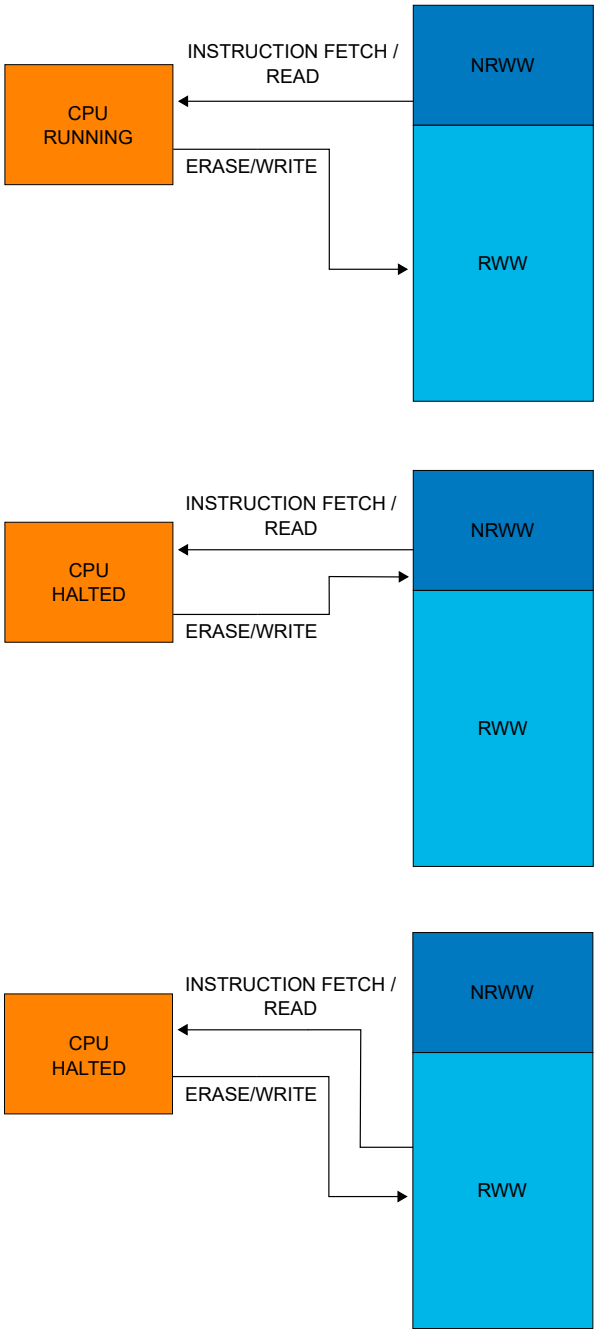


Table 11-1. Read-While-Write Scenarios

Flash Section Erased/Written	Flash Section Accessed	CPU
RWW section	NRWW section	Running
NRWW section	NRWW section	Halted
RWW section	RWW section	Halted

Notes:

- The User Row is located in the RWW Flash. When erasing or writing a page in the User Row, the NRWW Flash can be read during the operation.
- The Boot Row is located in the RWW Flash. When erasing or writing a page in the Boot Row, the NRWW Flash can be read during the operation.
- The physical section sizes are device-dependent. Refer to the *Memory Overview* section for further details.

11.3.1.1.2. Logical Sections

The Flash can be divided into three logical sections, each consisting of a variable number of pages. These sections are:

- **Boot Code (BOOT)** - The code executed from the BOOT area has write access to all other Flash sections. Boot loader software must be placed in this section if used.
- **Application Code (APPCODE)** - The code executed from the APPCODE area has limited write access to other Flash sections. Executable application code is typically placed in this section.
- **Application Data (APPDATA)** - Flash section without write access. Parameters are usually placed in this section.

Note: For detailed inter-section read/write access, refer to the *Flash Access Protection* section.

Section Sizes

The Boot Size (FUSE.BOOTSIZE) fuse and Code Size (FUSE.CODESIZE) fuse set the sizes of these sections. The fuses select section sizes in blocks of 256 bytes. The BOOT section stretches from FLASHSTART to BOOTEND, and the APPCODE section extends from BOOTEND to CODEEND. The remaining area is the APPDATA section.

If FUSE.BOOTSIZE is written to '0', the entire Flash is regarded as the BOOT section. If FUSE.CODESIZE is written to '0' and FUSE.BOOTSIZE > 0, the APPCODE section runs from BOOTEND to the end of Flash (no APPDATA section).

When FUSE.CODESIZE ≤ FUSE.BOOTSIZE, the APPCODE section is removed, and the APPDATA runs from BOOTEND until the end of the Flash.

Table 11-2. Setting Up Flash Sections

BOOTSIZE	CODESIZE	BOOT Section	APPCODE Section	APPDATA Section
0	-	0 to FLASHEND	-	-
> 0	0	0 to BOOTEND	BOOTEND to FLASHEND	-
> 0	≤ BOOTSIZE	0 to BOOTEND	-	BOOTEND to FLASHEND
> 0	> BOOTSIZE	0 to BOOTEND	BOOTEND to CODEEND	CODEEND to FLASHEND

Using the BOOT section for the application code is recommended if there is no boot loader software.

Notes:

1. After Reset, the default vector table location is at the start of the APPCODE section. The peripheral interrupts can be used in the code running in the BOOT section by relocating the interrupt vector table at the beginning of this section. That is done by setting the IVSEL bit in the CPUINT.CTRLA register. Refer to the *CPUINT* section for details.
2. If BOOTEND/CODEEND, as determined by the BOOTSIZE/CODESIZE fuse settings, exceeds the device FLASHEND, the corresponding fuse setting is ignored, and the default value is used. Refer to *Fuse* in the *Memories* section for default values.

Example 11-1. Size of Flash Sections Example

If FUSE.BOOTSIZE is written to `0x04` and FUSE.CODESIZE 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.

11.3.1.1.3. Flash Access Protections

Inter-Section Write Protection

For security reasons, it is impossible to write to the section of Flash from which the code is currently executing. Code writing to the APPCODE section must execute from the BOOT section, and code writing to the APPDATA section must execute from either the BOOT section or the APPCODE section.

Table 11-3. Write Protection for Self-Programming

Program Execution Section	Section Being Addressed	Programming Allowed?
BOOT	BOOT	No
	APPCODE	Yes
	APPDATA	
	EEPROM	
	USERROW	
	BOOTROW	Yes
APPCODE	BOOT	No
	APPCODE	
	APPDATA	Yes
	EEPROM	
	USERROW	
	BOOTROW	No
APPDATA	BOOT	No
	APPCODE	
	APPDATA	
	EEPROM	
	USERROW	
	BOOTROW	No

Flash Read/Write Protection

In addition to the inter-section write protection, the NVMCTRL provides a security mechanism to avoid unwanted access to the Flash memory sections. Even if the CPU can never write to the BOOT section, a Boot Section Read Protection (BOOTRP) bit in the Control B (NVMCTRL.CTRLB) register is provided to prevent the read and execution of code from the BOOT section. This bit can be set only from the code executed in the BOOT section and has an effect only when leaving the BOOT section.

The three write protection bits (EEWP, APPDATAWP and APPCODEWP) in the Control B (NVMCTRL.CTRLB) register can be set to prevent writes respectively to the EEPROM or the APPDATA or APPCODE sections.

11.3.1.2. EEPROM

The EEPROM is a 512 bytes nonvolatile memory section divided into a set of pages, where one page consists of multiple bytes. The EEPROM has byte granularity on erase/write. Each write/erase can contain one or more bytes inside a page. When writing data to the page buffer, the address for that byte is marked to be updated when performing the next erase/write. The remaining bytes on the page will not be erased/written. It can also perform a byte erase and write in one operation.

11.3.1.3. User Row

The User Row is 64 bytes of Read-While-Write (RWW) Flash. Use this section to store various data, such as calibration/configuration data and serial numbers. This section is not erased by a chip erase.

The User Row section can be read or written from the CPU. When erasing the User Row, the entire row is erased at once.

This section can be written through UPDI on a locked device.

11.3.1.4. Boot Row

The Boot Row is 64 bytes of Read-While-Write (RWW) Flash. This section stores information only accessible to the boot loader, such as keys. This section is optionally erased by a chip erase controlled from the fuse settings (FUSE).

The Boot Row section can be read or written from the CPU. When erasing the Boot Row, the entire row is erased at once.

The Boot Row cannot be accessed from the application section.

11.3.1.5. Fuses

The fuses contain device configuration values and are copied to their target registers at the end of the start-up sequence.

The fuses can be read by the CPU or the UPDI but can only be programmed or cleared by the UPDI.

11.3.1.6. Signature Row

The Signature Row contains a device ID that identifies each microcontroller device type and a serial number for each manufactured device. The serial number consists of the production lot number, wafer number, and the device's wafer coordinates. The Signature Row can be read by the CPU or the UPDI interface but not written or erased.

11.3.2. Memory Access

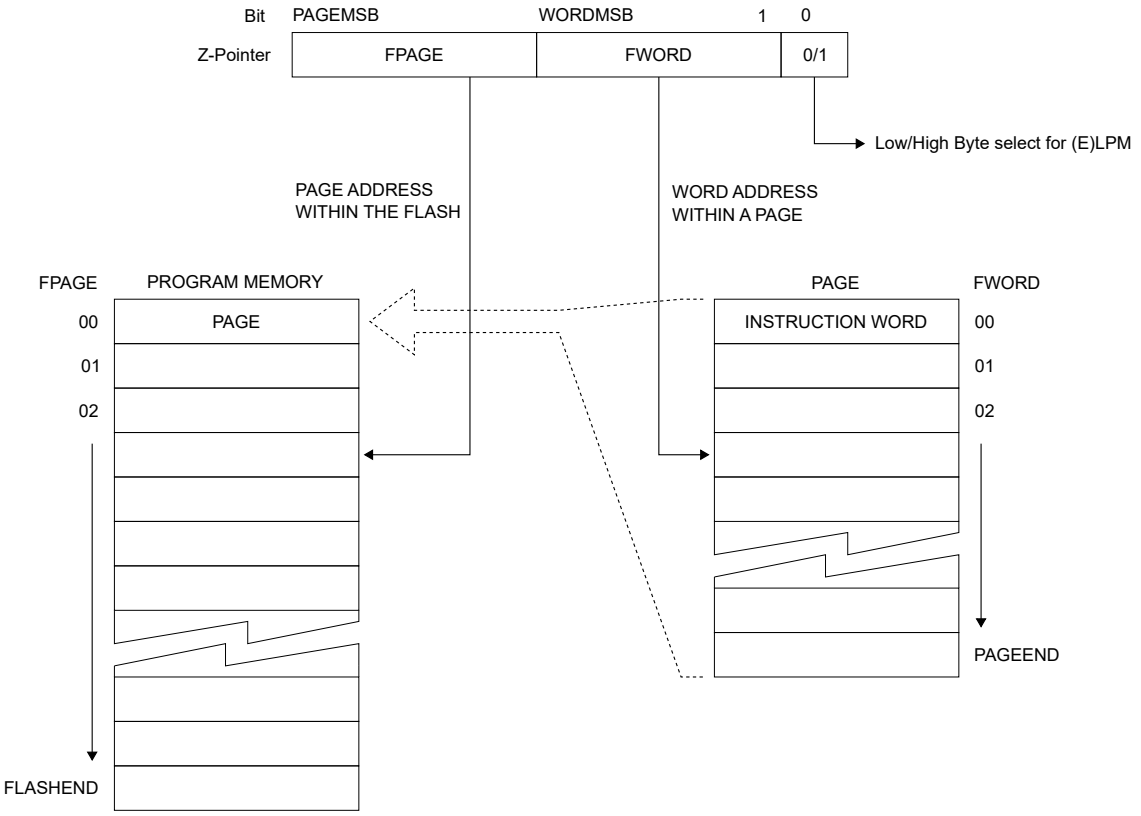
The Flash memory can be accessed from either the code space or the CPU data space for read/write operations. When using the code space, the Flash is accessible through the `LPM` and `SPM` instructions.

Additionally, the Flash memory is byte accessible through the CPU data space, which means that it shares the same address space and instructions as SRAM, EEPROM and I/O registers and is accessible using `LD/ST` instructions in assembly.

Addressing Flash Memory in Code Space

For read and write access to the Flash memory in the code space, use the Z-pointer for `LPM`/`SPM` access.

Figure 11-4. Flash Addressing for Self-Programming



The Flash is word-accessed and organized in pages, so the Address Pointer can be treated as having two sections, as shown in the figure above. The word address in the page (FWORD) is held by the Least Significant bits (LSb) in the Address Pointer, while the Most Significant bits (MSb) in the Address Pointer hold the Flash page address (FPAGE). FWORD and FPAGE hold an absolute address to a word in the Flash.

For Flash read operations, read one byte at a time. For this, use the Least Significant bit (bit 0) in the Address Pointer to select the low byte or high byte in the word address. If this bit is '0', the low byte is read, and if this bit is '1', the high byte is read.

Once initializing a programming operation, the address is latched, and the Address Pointer can be updated and used for other tasks.

Addressing Flash in CPU Data Space

The Flash area in the data space has only 32 KB. For devices with a Flash memory size greater than 32 KB, the Flash memory is divided into blocks of 32 KB. Those blocks are mapped into data space using the FLMAP bit field of the NVMCTRL.CTRLB register.

For read and write access to the Flash memory in the CPU data space, use the LD/ST instructions to access one byte at a time.

11.3.2.1. Read

Reading the Flash is done using Load Program Memory (LPM) instructions or Load (LD*) instructions with an address according to the memory map. Reading the EEPROM and Signature Row is done using LD* instructions. Performing a read operation 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.

11.3.2.2. Page Buffer Load

The page buffer is loaded by writing directly to the memories, as defined in the memory map. EEPROM has a separate page buffer, while Flash and User Row share the same page buffer, so only one section can be programmed at once. Use the address's Least Significant bits (LSb) to select where to write the data in the page buffer. 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

Note: Any operation on the page buffer will halt the CPU until the previous NVMCTRL operation (command) is completed.

11.3.2.3. 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 content of the Flash page must be erased (read back 0xFF). Programming an unerased Flash page will corrupt its content.

Two options are available to make sure that the Flash page content is programmed correctly:

Option 1: The Flash is programmed using one command that handles both erase and write.

1. Make sure the Flash is ready by reading the Flash Busy (FLBUSY) flag in the NVMCTRL.STATUS register.
2. Fill the page buffer.
3. Write the page buffer to Flash with the Flash Page Erase and Page Write (FLPERW) command.

Option 2: The Flash is programmed using separate commands for page erase and page write.

1. Make sure the Flash is ready by reading the Flash Busy (FLBUSY) flag in the NVMCTRL.STATUS register.
2. Write to a location on the page to set up the address.
3. Perform a Flash Page Erase (FLPER) command.
4. Fill the page buffer.
5. Perform a Flash Page Write (FLPW) command.

The NVM command set supports both single Page Erase and Write (FLPERW/EEPERW) operations and split Page Erase (FLPER/EEPER) and Page Write (FLPW/EEPW) commands for both Flash and EEPROM. These split commands enable a shorter programming time for each command, and the erase operations can be done during non-time-critical programming execution.

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

Table 11-4. Programming Granularity

Memory Section	Erase Granularity	Write Granularity
Flash array	Page	Page
EEPROM array	Byte	Byte
User Row	Page(1)	Page(1)
Boot Row	Page(2)	Page(2)

Notes:

1. User Row page is 64 bytes.
2. Boot Row page is 64 bytes.

11.3.2.4. Command Modes

Reading the memory arrays is handled using the LD*/LPM[1] instructions.

The EEPROM Erase (EECHER) command is started by writing a command to the NVMCTRL.CTRLA register. The other write/erase operations are just enabled by writing commands to the NVMCTRL.CTRLA register and must be followed by writes using ST*/SPM[1] instructions to the memory arrays.

Note:

1. LPM/SPM cannot be used for EEPROM.

To write a command in the NVMCTRL.CTRLA register, the following sequence needs to be executed:

1. Confirm that any previous operations are completed by reading the Busy (EEBUSY and FLBUSY) flags in the NVMCTRL.STATUS register.
2. Write the appropriate key to the Configuration Change Protection (CPU.CCP) register to unlock the NVM Control A (NVMCTRL.CTRLA) register.
3. Write the desired command value to the CMD bit field in the Control A (NVMCTRL.CTRLA) register within the following four instructions.

11.3.2.4.1. Page Write Command

The Page Write (FLPW/EEPW) commands write the content of the page buffer to the Flash or EEPROM.

If the write is to the Flash, the CPU will execute code as explained in Physical Sections.

If the write is to the EEPROM, the CPU will continue executing code while the operation is ongoing. Erase the page/byte before performing a write.

The page buffer used will automatically be cleared after finishing the operation.

11.3.2.4.2. Page Erase Command

The Page Erase (FLPER/EEPER) commands erase the current page. Write one byte in the page buffer for the Page Erase command to take effect.

For erasing the Flash, a dummy write to one address in the desired page is required first, followed by command execution. The whole page in the Flash will then be erased. The CPU will stop or continue based on the same conditions as the Page Write command.

When executing the command for the EEPROM, only the bytes written in the page buffer will be erased. To erase a specific byte, write to its corresponding address before executing the command. Update all the bytes in the page buffer before writing the command to erase a whole page. The CPU will continue running code while the operation is ongoing.

The page buffer used will automatically be cleared after finishing the operation.

11.3.2.4.3. Flash Multi-Page Erase Mode

The Multi-Page Erase (FLMPERn) mode will allow each write to the memory array to erase multiple pages. When using FLMPERn, it is possible to select between erasing 2, 4, 8, 16, or 32 pages.

The LSbs of the page address are ignored when defining which Flash pages are erased. Using FLMPER4 as an example, erasing any page in the 0x08 - 0x0B range will cause the erase of all pages in the range.

Table 11-5. Flash Multi-Page Erase

CMD	Pages Erased	Description
FLMPER2	2	Pages matching FPAGE[N:1] are erased. The value in FPAGE[0] is ignored.
FLMPER4	4	Pages matching FPAGE[N:2] are erased. The value in FPAGE[1:0] is ignored.
FLMPER8	8	Pages matching FPAGE[N:3] are erased. The value in FPAGE[2:0] is ignored.
FLMPER16	16	Pages matching FPAGE[N:4] are erased. The value in FPAGE[3:0] is ignored.
FLMPER32	32	Pages matching FPAGE[N:5] are erased. The value in FPAGE[4:0] is ignored.

Note: FPAGE is the page number when doing a Flash erase. Refer to Memory Access for details.

11.3.2.4.4. Page Erase/Write Operation

Page Erase and Write (FLPERW/EEPERW) commands combine Page Erase and Page Write commands - but without clearing the page buffer after the Page Erase command. The erase/write operation erases the selected page, then writes the content of the page buffer to the same page.

When performed on the Flash, the CPU will stop or continue based on the same conditions as the Page Write command. When done on the EEPROM, the CPU will continue executing code.

The page buffer used will automatically be cleared after finishing the operation.

11.3.2.4.5. Page Buffer Clear Commands

The Page Buffer Clear (FLPBCLR/EEPBCLR) commands clear the corresponding page buffer. The contents of the page buffer will be all '1's after the operation. The CPU will halt while the operation executes (seven CPU cycles).

11.3.2.4.6. EEPROM Erase Command

The EEPROM Erase (EECHER) command erases the EEPROM. All EEPROM bytes will read back `0xFF` after the operation. The CPU is halted during the EEPROM erase.

11.3.3. Preventing Flash/EEPROM Corruption

A Flash/EEPROM write or erase can cause memory corruption if the supply voltage is too low for the CPU and the Flash/EEPROM to operate correctly. These issues are the same on board-level systems using Flash/EEPROM. The internal or an external Brown-out Detector (BOD) is recommended to ensure that the operating voltage is high enough.

Two circumstances may cause Flash/EEPROM corruption when the voltage is too low:

1. A regular write sequence to the Flash, requiring a minimum voltage to operate correctly.
2. The CPU can execute instructions incorrectly when the supply voltage is too low.

The chip erase does not clear fuses. If the BOD is enabled by fuses before starting the Chip Erase command, it is automatically enabled at its previous configured level during the chip erase.

Refer to the *Electrical Characteristics* section for Maximum Frequency vs. V_{DD}.

Attention: Taking the following measures may avoid Flash/EEPROM corruption:

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

11.3.4. Interrupts

Table 11-6. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
EEREADY	NVM	The EEPROM is ready for new write/erase operations
FLREADY		The Flash is ready for new write/erase operations

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

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

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

11.3.5. 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, which is valid for all sleep modes, including the Power-Down sleep mode.

The EEPROM Ready and Flash Ready interrupts will only wake the device from Idle sleep mode.

The page buffer is cleared when waking up from sleep.

11.3.6. 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 11-7. NVMCTRL - Registers Under Configuration Change Protection

Register	Key
NVMCTRL.CTRLA	SPM
NVMCTRL.CTRLB	IOREG

11.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		CMD[6:0]						
0x01	CTRLB	7:0	FLMAPLOCK		FLMAP[1:0]		EEWP	APPDATAWP	BOOTRP	APPCODEWP
0x02	CTRLC	7:0							BOOTROWWP	UROWWP
0x03	Reserved									
0x04	INTCTRL	7:0							FLREADY	EEREADY
0x05	INTFLAGS	7:0							FLREADY	EEREADY
0x06	STATUS	7:0			ERROR[2:0]				FLBUSY	EEBUSY
0x07	Reserved									
0x08	DATA	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
0x0A ... 0x0B	Reserved									
0x0C	ADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24								

11.5. Register Description

11.5.1. Control A

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

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

Bits 6:0 – CMD[6:0] Command
Write this bit field to enable or issue a command. The Chip Erase and EEPROM erase commands start when writing either command to the bit field. The other commands enable an erase or a write operation. These operations start when doing a store to the Flash address being erased or written. Including a NOCMD or NOOP instruction is recommended when going from one command to the next to prevent a Command Collision error (in the ERROR bit field in NVMCTRL.STATUS).
The Configuration Change Protection key for self-programming (SPM) protects these bits.

Value	Name	Description
0x00	NOCMD	No command
0x01	NOOP	No operation
0x04	FLPW	Flash Page Write
0x05	FLPERW	Flash Page Erase and Page Write
0x08	FLPER	Flash Page Erase
0x09	FLMPER2	Flash 2-page Erase Enable
0x0A	FLMPER4	Flash 4-page Erase Enable
0x0B	FLMPER8	Flash 8-page Erase Enable
0x0C	FLMPER16	Flash 16-page Erase Enable
0x0D	FLMPER32	Flash 32-page Erase Enable
0x0F	FLPBCLR	Flash Page Buffer Clear
0x14	EEPW	EEPROM Page Write
0x15	EEPERW	EEPROM Page Erase and Page Write
0x17	EEPER	EEPROM Page Erase
0x1F	EEPBCLR	EEPROM Page Buffer Clear
0x20	CHER	Erase Flash and EEPROM. EEPROM is skipped if the EESAVE fuse is set (UPDI access only).
0x30	EECHER	EEPROM Erase
Other	-	Reserved

11.5.2. Control B

Name: CTRLB
Offset: 0x01
Reset: 0x30
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
	FLMAPLOCK		FLMAP[1:0]		EEWP	APPDATAWP	BOOTRP	APPCODEWP
Access	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	0		1	1	0	0	0	0

Bit 7 – FLMAPLOCK Flash Mapping Lock
Setting this bit to '1' prevents further updates of FLMAP[1:0]. Only a Reset can clear this bit.

Bits 5:4 – FLMAP[1:0] Flash Section Mapped into Data Space
Select which part (in blocks of 32 KB) of the Flash will be mapped as part of the CPU data space and accessible through LD/ST instructions. Changing this bitfield value doesn't affect devices with 32 KB or less Flash size, as the entire Flash is mapped to the CPU data space.

Value	Name	Mapped Flash Section (16 KB)
0x0	SECTION0	0-16
0x1	SECTION1	
0x2	SECTION2	
0x3	SECTION3	

Bit 3 – EEWP EEPROM Write Protection
Writing a '1' to this bit protects the EEPROM from further writes.
This bit can only be written to '1'. Only Reset clears it to '0'.

Bit 2 – APPDATAWP Application Data Section Write Protection
Writing a '1' to this bit protects the application data section from further writes.
This bit can only be written to '1'. Only Reset clears it to '0'.

Bit 1 – BOOTRP Boot Section Read Protection
Writing a '1' to this bit protects the BOOT section from a read and instruction fetch. If issuing a read from the application section, it will return '0'. An instruction fetch from the BOOT section returns an NOP instruction.
This bit can only be written to '1' from the BOOT section. Only Reset clears it to '0'. The bit will only take effect when the BOOT section is left the first time after the bit is written.
The BOOTROW will also be locked for access from the application section.

Bit 0 – APPCODEWP 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'. Only Reset clears it to '0'.

11.5.3. Control C

Name: CTRLC
Offset: 0x02
Reset: 0x0
Property: Configuration Change Protection

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

Bit 1 – BOOTROWWP Boot Row Write Protection
Writing this bit to '1' prevents further updates to the Boot Row. Only a Reset can clear this bit.

Bit 0 – UROWWP User Row Write Protection
Writing this bit to '1' prevents further updates to the User Row. Only a Reset can clear this bit.

11.5.4. Interrupt Control

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

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

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

Bit 0 – EEREADY EEPROM Ready Interrupt Enable
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 '1'. Thus, the interrupt must not be enabled before triggering an NVM command, as the EEBUSY flag will not be set before the NVM command is issued. The interrupt must be disabled in the interrupt handler.

11.5.5. Interrupt Flags

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

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

Bit 1 – FLREADY Flash Ready Interrupt Flag
This flag is set continuously as long as the Flash is not busy. This flag is cleared by writing a '1' to it.

Bit 0 – EEREADY EEPROM Ready 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.

11.5.6. Status

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

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

Bits 6:4 – ERROR[2:0] Error Code
This bit field will show the last error that occurred. The error code can be cleared by writing '0' to the bit field.

Value	Name	Description
0x00	NONE	No error
0x02	WRITEPROTECT	Attempting to write a write-protected section
0x03	CMDCOLLISION	Selecting a new write command while a write command is already selected
0x04	WRONGSECTION	Wrong write command for the address used

Bit 1 – FLBUSY Flash Busy
This bit will read '1' when a Flash programming operation is ongoing.

Bit 0 – EEBUSY EEPROM Busy
This bit will read '1' when an EEPROM programming operation is ongoing.

11.5.7. Data

Name: DATA
Offset: 0x08
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
The Data register will contain the last read value from Flash, EEPROM, or NVMCTRL. For EEPROM access, only DATA[7:0] is used.

11.5.8. Address

Name: ADDR
Offset: 0x0C
Reset: 0x00
Property: -

NVMCTRL.ADDR0, NVMCTRL.ADDR1, NVMCTRL.ADDR2 and NVMCTRL.ADDR3 represent the 32-bit value NVMCTRL.ADDR.

The low byte [7:0] (suffix 0) is accessible at the original offset.

The high byte [15:8] (suffix 1) can be accessed at offset +`0x01`.

The extended byte [23:16] (suffix 2) can be accessed at offset +`0x02`.

The byte [31:24] (suffix 3) can be accessed at offset +`0x03`, but it never contains any data.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

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

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

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

Bits 23:0 – ADDR[23:0] Address
The Address register contains the address of the last accessed memory location. Only the number of bits required to access the memory is used.

12. CLKCTRL - Clock Controller

12.1. Features

- All Clocks and Clock Sources Are Automatically Enabled When Requested by Peripherals
- Internal Oscillators:
  - Up to 20 MHz Internal High-Frequency Oscillator (OSCHF)
  - 32.768 kHz Ultra-Low Power Oscillator (OSC32K)
- Auto-Tuning for Improved Internal Oscillator Accuracy
- External Clock Options:
  - 32.768 kHz Crystal Oscillator (XOSC32K)
  - External clock input
- Main Clock Features:
  - Fuse-selectable start-up clock source
  - Safe run-time switching
  - System clock output to I/O pin
  - Prescaler with a division factor ranging from 1 to 64
  - Status bit for observing when clock switching is complete

12.2. Overview

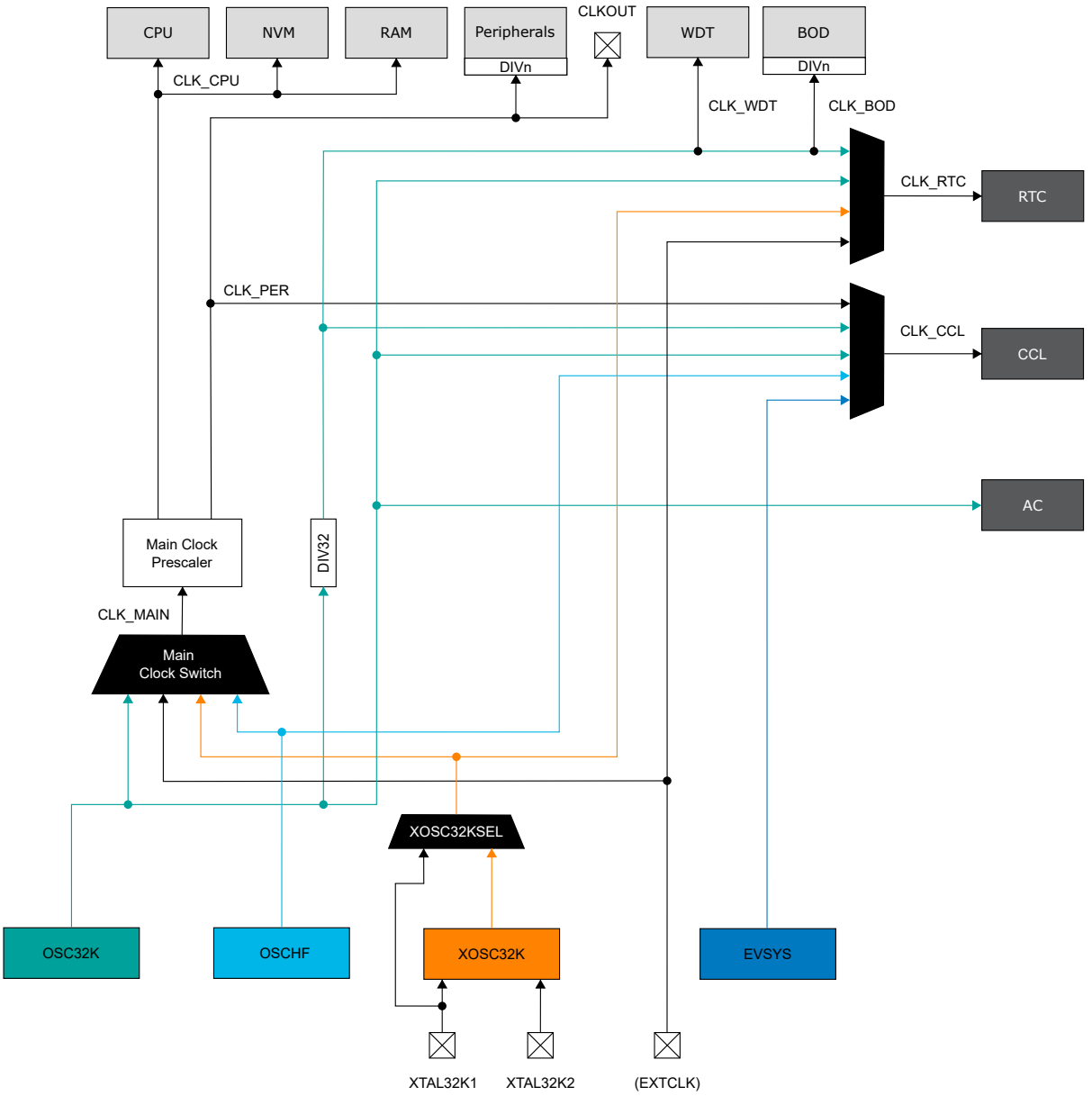
The Clock Controller (CLKCTRL) controls, distributes and prescales the clock signals from the available oscillators and supports internal and external clock sources.

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. The request is routed to the correct clock source if multiple clock sources are available.

The Main Clock (CLK_MAIN) is used by the CPU, Nonvolatile Memory (NVM), SRAM, and all peripherals connected to 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.

12.2.1. Block Diagram - CLKCTRL

Figure 12-1. CLKCTRL Block Diagram



The clock system consists of the main clock and clocks derived from the main clock, as well as several asynchronous clocks:

- The Main Clock (CLK_MAIN) is always running in Active mode and Idle sleep mode. If requested, it will also run in Standby sleep mode.
- CLK_MAIN is prescaled and distributed by the clock controller:
  - CLK_CPU is used by the CPU, the SRAM, and the Nonvolatile Memory Controller (NVMCTRL)
  - CLK_PER is used by all peripherals that are not listed under asynchronous clocks and can also be routed to the CLKOUT pin
  - All clock sources can be used as the main clock
- Clocks running asynchronously to the main clock domain:

- CLK_WDT is used by the Watchdog Timer (WDT). It is requested when the WDT is enabled.
- CLK_BOD is used by the Brown-out Detector (BOD). It is requested when the BOD is enabled in Sampled mode. The alternative clock source is controlled by a fuse.
- CLK_RTC is used by the Real-Time Counter (RTC) and the Periodic Interrupt Timer (PIT). It is requested when the RTC/PIT is enabled. The clock source for CLK_RTC may be changed only if the peripheral is disabled.
- CLK_CCL is used by the Configurable Custom Logic (CCL). It is requested when the CCL is enabled. The clock source may be changed only if the peripheral is disabled.

The clock source for the main clock domain is configured by writing to the Clock Select (CLKSEL) bit field in the Main Clock Control A (CLKCTRL.MCLKCTRLA) register. This register has Configuration Change Protection (CCP), and the appropriate key must be written to the CCP register before writing to the CLKSEL bit field. The asynchronous clock sources are configured by the registers in the respective peripheral.

12.2.2. Signal Description

Signal	Type	Description
CLKOUT	Digital output	CLK_PER output
EXTCLK	Analog input	Input for the external clock source (EXTCLK)
XTAL32K1	Analog input	Input for an external 32.768 kHz clock source or one pin of a 32.768 kHz crystal
XTAL32K2	Analog input	Input for one pin of a 32.768 kHz crystal

For more details, refer to the *I/O Multiplexing* chapter.

12.3. Functional Description

12.3.1. Initialization

To initialize a clock source as the main clock, follow these steps:

1. Optional: Force the clock to always run by writing the Run Standby (RUNSTDBY) bit in the respective clock source CTRLA register to '1'.
2. Configure the clock source as needed in the corresponding clock source CTRLA register and, if applicable, enable the clock source by writing a '1' to the Enable bit.
3. Optional: If RUNSTDBY is '1', wait for the clock source to stabilize by polling the respective status bit in CLKCTRL.MCLKSTATUS.
4. The following sub-steps must be performed in an order such that the main clock frequency never exceeds the allowed maximum clock frequency. Refer to the *Electrical Characteristics* chapter for further information.
   a. If required, divide the clock source frequency by writing to the Prescaler Division (PDIV) bit field and enable the main clock prescaler by writing a '1' to the Prescaler Enable (PEN) bit in CLKCTRL.MCLKCTRLB.
   b. Select the configured clock source as the main clock in the Clock Select (CLKSEL) bit field in CLKCTRL.MCLKCTRLA.
5. Wait for the main clock to change by polling the Main Clock Oscillator Changing (SOSC) bit in the Main Clock Status (CLKCTRL.MCLKSTATUS) register.
6. Optional: Clear the RUNSTDBY bit in the clock source CTRLA register.

12.3.2. Main Clock Selection and Prescaler

All available oscillators and the external clock (EXTCLK) can be used as the main clock source for the Main Clock (CLK_MAIN). The main clock source is selectable from software and can be safely changed during ordinary operation.

The Configuration Change Protection mechanism prevents unsafe clock switching. For more details, refer to the *Configuration Change Protection* section.

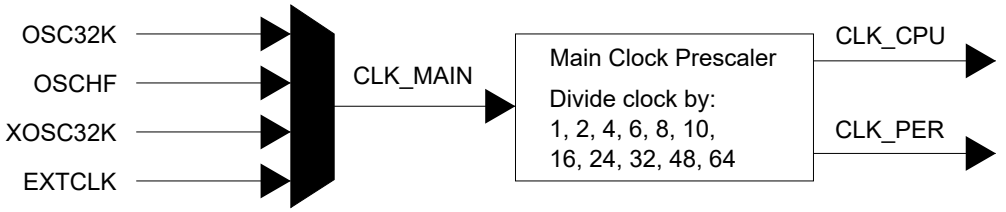
When selecting an external clock source, a switch to the chosen clock source will occur if a sufficient number of edges are detected. If an insufficient number of clock edges is detected, the clock source remains unchanged, and it is impossible to change to another clock source without executing a Reset.

The Main Clock Oscillator Changing (SOSC) bit in the Main Clock Status (CLKCTRL.MCLKSTATUS) register indicates an ongoing clock source switch. The stability of the external clock sources is indicated by the respective Status (EXTS and XOSC32KS) bits in CLKCTRL.MCLKSTATUS.

⚠ CAUTION If an external clock source fails while it is used as the CLK_MAIN source, the Watchdog Timer (WDT) can provide a System Reset.

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

Figure 12-2. Main Clock and Prescaler



12.3.3. Main Clock After Reset

After any Reset, the Main Clock (CLK_MAIN) is provided by the 20 MHz Oscillator (OSCHF) with a prescaler division factor of 6.

The actual OSCHF frequency is determined by the Frequency Select bits (OSCHFFRQ) of the Oscillator Configuration fuse (FUSE.OSCCFG). Refer to the description of the FUSE.OSCCFG fuse for details of the possible frequencies after Reset.

12.3.4. Clock Sources

All the internal clock sources are automatically enabled when requested by a peripheral. The crystal oscillators, based on an external crystal, must be enabled before they can serve as a clock source.

- The XOSC32K oscillator is enabled by writing a '1' to the ENABLE bit in the 32.768 kHz Crystal Oscillator Control A (CLKCTRL.XOSC32KCTRLA) register

After Reset, the device starts running from the internal high-frequency or 32.768 kHz oscillator.

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

12.3.4.1. Internal Oscillators

The internal oscillators do not require any external components to run. Refer to the *Electrical Characteristics* section for accuracy and electrical specifications.

12.3.4.1.1. Internal High-Frequency Oscillator (OSCHF)

The OSCHF oscillator supports output frequencies of 16 and 20 MHz. It can be used as a clock source for the main clock (CLK_MAIN) and for several peripherals (see the *CLKCTRL Block Diagram*). The output frequency of the OSCHF can be tuned manually or automatically against an external 32.768 kHz oscillator (XOSC32K) to reduce drift.

Refer to the *Electrical Characterization* chapter for tuning ranges and oscillator specifications.

12.3.4.1.2. Internal 32.768 kHz Oscillator (OSC32K)

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 a 32.768 kHz clock to the Main Clock (CLK_MAIN). It also provides a 1.024 kHz and/or 32.768 kHz clock for several peripherals. See the *CLKCTRL Block Diagram* for details.

For the start-up time of this oscillator, refer to the *Electrical Characteristics* chapter.

12.3.4.2. External Clock Sources

The available external clock sources are described in the following sections.

12.3.4.2.1. 32.768 kHz Crystal Oscillator (XOSC32K)

This oscillator supports two input options:

- A crystal is connected to the XTAL32K1 and XTAL32K2 pins
- An external clock running at 32.768 kHz is connected to XTAL32K1

Configure the input option by writing the Source Select (SEL) bit in the XOSC32K Control A (CLKCTRL.XOSC32KCTRLA) register.

The XOSC32K is enabled by writing a '1' to the ENABLE bit in CLKCTRL.XOSC32KCTRLA. When enabled, the configuration of the General Purpose Input/Output (GPIO) pins used by the XOSC32K overrides that of the XTAL32K1 and XTAL32K2 pins. The oscillator must be enabled 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) bit field in XOSC32KCTRLA.

When XOSC32K is configured to use an external clock on XTAL32K1, the start-up time is fixed to two cycles.

The power mode can be adjusted to accommodate different crystal oscillators by writing to the Power Mode (PMODE) bit field in XOSC32KCTRLA.

12.3.4.2.2. External Clock (EXTCLK)

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

The maximum input frequency for the EXTCLK is 20 MHz.

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

12.3.5. Timebase

The timebase is used to generate a timing period equal to or longer than 1 µs, used for timing internal delays such as ADC start-up time, which is done by setting the TIMEBASE bitfield in the Timebase (CLKCTRL.MCLKTIMEBASE) register to a count of CLK_PER cycles that is equivalent to or larger than 1 µs. Round the timebase up to the closest integer.

12.3.6. Manual Tuning and Autotune

The output frequency of the OSCHF can be tuned either manually or automatically against an external oscillator.

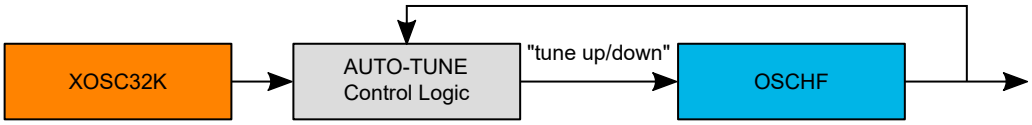
Manual Tuning

The output frequency of the OSCHF can be tuned up and down by writing to the Oscillator Tune (TUNE) bit field in the Frequency Tune (TUNE) register. The Automatic Oscillator Tune (AUTOTUNE) bit field in the CTRLA register must remain zero.

Autotune Against External Crystal Oscillator

The OSCHF output frequency can be stabilized by automatic tuning against an external crystal oscillator. Enable Autotune by selecting the external oscillator in the Automatic Oscillator Tune (AUTOTUNE) bit field in the CTRLA register, which locks the TUNE register and prevents manual tuning. The TUNE register is updated with the latest TUNE value when AUTOTUNE is disabled.

Figure 12-3. OSCHF Auto-Tune Block Diagram



Refer also to the *Electrical Characteristics* chapter for details.

12.3.7. Sleep Mode Operation

When a clock source is not used or requested, it stops. It is possible to request a clock source directly by writing a '1' to the Run Standby (RUNSTDBY) bit in the respective oscillator's Control A (CLKCTRL.*oscillator*CTRLA) register. This causes the oscillator to run constantly, except in Power-Down sleep mode. Additionally, when this bit is set to '1', the oscillator start-up time is eliminated when the clock source is requested by a peripheral.

The main clock always runs in Active mode and Idle sleep mode. In Standby sleep mode, the main clock runs only if any peripheral is requesting it or if the RUNSTDBY bit in the respective oscillator's CLKCTRL.*oscillator*CTRLA register is set to '1'.

In Power-Down sleep mode, the main clock stops after all Nonvolatile Memory (NVM) operations are completed. Refer to the *SLPCTRL - Sleep Controller* chapter for more details on sleep mode operation.

12.3.8. 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 12-1. CLKCTRL - Registers Under Configuration Change Protection

Register	Key
CLKCTRL.MCLKCTRLA	IOREG
CLKCTRL.MCLKCTRLB	IOREG
CLKCTRL.MCLKLOCK	IOREG
CLKCTRL.XOSC32KCTRLA	IOREG
CLKCTRL.OSCHFCTRLA	IOREG
CLKCTRL.OSC32KCTRLA	IOREG

12.4. Register Summary - CLKCTRL

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	MCLKCTRLA	7:0	CLKOUT				CLKSEL[3:0]			
0x01	MCLKCTRLB	7:0				PDIV[3:0]				PEN
0x02 ... 0x04	Reserved									
0x05	MCLKSTATUS	7:0				EXTS	XOSC32KS	OSC32KS	OSCHFS	SOSC
0x06	MCLKTIMEBASE	7:0				TIMEBASE[4:0]				
0x07	Reserved									
0x08	OSCHFCTRLA	7:0	RUNSTDBY						AUTOTUNE[1:0]	
0x09	OSCHFTUNE	7:0	TUNE[7:0]							
0x0A ... 0x17	Reserved									
0x18	OSC32KCTRLA	7:0	RUNSTDBY							
0x19 ... 0x1B	Reserved									
0x1C	XOSC32KCTRLA	7:0	RUNSTDBY		CSUT[1:0]		SEL	PMODE[1:0]		ENABLE

12.5. Register Description

12.5.1. Main Clock Control A

Name: MCLKCTRLA
Offset: 0x0
Reset: 0x0
Property: Configuration Change Protection

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

Bit 7 – CLKOUT Enable Clock Output to Pin
This bit controls whether the main clock is available on the Main Clock Out (CLKOUT) pin when the main clock is running.

Value	Description
0	The main clock is not available on the CLKOUT pin
1	The main clock is available on the CLKOUT pin

Bits 3:0 – CLKSEL[3:0] Clock Selection Between Available Clock Sources
This bit field selects the source for the Main Clock (CLK_MAIN).

Value	Name	Description
0x0	OSCHF	Internal high-frequency oscillator
0x1	OSC32K	Internal 32.768 kHz oscillator
0x2	XOSC32K	32.768 kHz oscillator
0x3	EXTCLK	External clock
Other	—	Reserved

12.5.2. Main Clock Control B

Name: MCLKCTRLB
Offset: 0x1
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
This bit field controls the division ratio of the Main Clock (CLK_MAIN) prescaler when the Prescaler (PEN) bit is set to '1'. This bit field can be written during normal operation to change the system clock frequency to suit application requirements.

Value	Name	Description
0x0	DIV2	Divide by 2
0x1	DIV4	Divide by 4
0x2	DIV8	Divide by 8
0x3	DIV16	Divide by 16
0x4	DIV32	Divide by 32
0x5	DIV64	Divide by 64
0x8	DIV6	Divide by 6
0x9	DIV10	Divide by 10
0xA	DIV12	Divide by 12
0xB	DIV24	Divide by 24
0xC	DIV48	Divide by 48

Note: Configuration of the input frequency (CLK_MAIN) and prescaler settings must not exceed the allowed maximum frequency of the peripheral clock (CLK_PER) or CPU clock (CLK_CPU). Refer to the *Electrical Characteristics* chapter for further information.

Bit 0 – PEN Prescaler Enable
This bit controls whether the Main Clock (CLK_MAIN) prescaler is enabled. When enabled, the division ratio is selected by the PDIV bit field. If disabled, the main clock passes through undivided (CLK_PER = CLK_MAIN), regardless of the value of PDIV.

Value	Description
0	The CLK_MAIN prescaler is disabled. Any value in the Prescaler Division (PDIV) bit field is ignored.
1	The CLK_MAIN prescaler is enabled, and the division ratio is controlled by the Prescaler Division (PDIV) bit field

12.5.3. Main Clock Status

Name: MCLKSTATUS
Offset: 0x5
Reset: 0x0
Property: -

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

Bit 4 – EXTS External Clock Status

Value	Description
0	EXTCLK is not stable
1	EXTCLK is stable

Bit 3 – XOSC32KS 32.768 kHz External Crystal Oscillator Status

Value	Description
0	XOSC32K is not stable
1	XOSC32K is stable

Bit 2 – OSC32KS 32 kHz Internal Oscillator Status

Value	Description
0	OSC32K is not stable
1	OSC32K is stable

Bit 1 – OSCHFS Internal High-Frequency Oscillator Status

Value	Description
0	OSCHF is not stable
1	OSCHF is stable

Bit 0 – SOSC System 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

12.5.4. Main Clock Timebase

Name: MCLKTIMEBASE
Offset: 0x6
Reset: 0x0
Property: -

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

Bits 4:0 – TIMEBASE[4:0] Timebase
This bit field specifies the number of CLK_PER cycles that is equivalent to or larger than 1 µs. This is used for timing internal delays, such as the ADC start-up time.
The value must be rounded up to the closest integer. The following code snippet shows how to do this using the `ceil` function.

```
#include <math.h>
#define CLK_PER 3333333ul // 20 MHz/6 = 3.333333 MHz
#define TIMEBASE_VALUE ((uint8_t) ceil(CLK_PER*0.000001))
```

12.5.5. Internal High-Frequency Oscillator Control A

Name: OSCHFCTRLA
Offset: 0x08
Reset: 0x0
Property: Configuration Change Protection

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

Bit 7 – RUNSTDBY Run in Standby
Enable the Internal High-Frequency Oscillator (OSCHF) to run in Standby sleep mode even if it is not requested by any peripheral.

Value	Description
0	The OSCHF oscillator runs only when requested by a peripheral or by the main clock(1)
1	The OSCHF oscillator always runs in Active mode, Idle sleep mode, or Standby sleep mode(2)

Notes:
1. The requesting peripheral or the main clock must take the oscillator start-up time into account.
2. The oscillator signal is available only if requested and becomes available after two OSCHF cycles.

Bits 1:0 – AUTOTUNE[1:0] Automatic oscillator tune
This bit controls whether the external 32.768 kHz crystal auto-tune functionality of the Internal High-Frequency Oscillator (OSCHF) is enabled.

Value	Name	Description
0	OFF	The OSCHF auto-tune functionality is disabled
1	XOSC32K	The OSCHF auto-tune functionality against the XOSC32K clock source is enabled
Other	—	Reserved

12.5.6. Internal High-Frequency Oscillator Frequency Tune

Name: OSCHFTUNE
Offset: 0x09
Reset: 0x0
Property: -

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

Bits 7:0 – TUNE[7:0] Frequency Tuning Value

This bit field controls the manual tuning of the output frequency of the Internal High-Frequency Oscillator (OSCHF). The frequency can be tuned 32 steps down or 31 steps up from the oscillator's target frequency. Thus, the register's acceptable input value range is -32 to +31.
Writing to bits 6 and 7 has no effect, as bit 5 is mirrored to bits 6 and 7 due to the 6-bit value in this bit field being represented in signed (two's complement) form.

Note:
The TUNE value is locked when the Auto-Tune Enable (AUTOTUNE) bit in the Internal High-Frequency Oscillator Control A (CLKCTRL.OSCHFCTRLA) register is enabled. The TUNE register is updated with the latest tune value when AUTOTUNE is disabled.

After AUTOTUNE is disabled, there is a delay before the TUNE register is updated. This delay depends on the system clock frequency. The delay consists of a fixed auto-tune logic delay (0.75 µs) and three system clock cycles:

- 20 MHz system clock: t_{UPD} = 0.75 µs + 3 * 0.05 µs = 0.90 µs
- 5 MHz system clock: t_{UPD} = 0.75 µs + 3 * 0.20 µs = 1.35 µs

12.5.7. Internal 32.768 kHz Oscillator Control A

Name: OSC32KCTRLA
Offset: 0x18
Reset: 0x0
Property: Configuration Change Protection

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

Bit 7 – RUNSTDBY Run in Standby

Enable the 32.768 KHz Internal Oscillator to run in all power modes, even if it is not requested by any peripheral.

Notes:

1. The requesting peripheral or the main clock must take the oscillator start-up time into account.
2. The oscillator signal is available only if requested and becomes available after four OSC32K cycles.

Value	Description
0	The 32.768 kHz Internal Oscillator runs only if requested by a peripheral or the main clock(1)
1	The 32.768 kHz Internal Oscillator will always run in Active mode, Idle sleep mode, Standby sleep mode, and Power-Down sleep mode(2)

12.5.8. External 32.768 kHz Crystal Oscillator Control A

Name: XOSC32KCTRLA
Offset: 0x1C
Reset: 0x0
Property: Configuration Change Protection

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY		CSUT[1:0]		SEL	PMODE[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

Bit 7 – RUNSTDBY Run in Standby
This bit forces the External 32.768 kHz Crystal Oscillator (XOSC32K) to always run in all power modes except Power-Down mode when the ENABLE bit is '1', even if no peripheral has requested it.
Notes:
1. The requesting peripheral or the main clock must take the oscillator start-up time into account.
2. The oscillator signal is available only if requested and becomes available after a maximum of three XOSC32K cycles, assuming the initial crystal start-up time has passed.

Value	Description
0	The External 32.768 kHz Crystal Oscillator (XOSC32K) runs only when requested by a peripheral or by the main clock in Active mode and Idle sleep mode [1]
1	The External 32.768 kHz Crystal Oscillator (XOSC32K) always runs in Active mode, Idle sleep mode, or Standby sleep mode[2]

Bits 5:4 – CSUT[1:0] Crystal Oscillator Start-Up Time
This bit field controls the External 32.768 kHz Crystal Oscillator (XOSC32K) start-up time when the Crystal Select (SEL) bit is '0' (external crystal selected). If an external clock is selected, the start-up time will not be applied.
Note: This bit field is read-only when the oscillator is enabled (ENABLE bit is '1') or when the XOSC32K Status (XOSCS) bit in the Main Clock Status (CLKCTRL.MCLKSTATUS) register is '1'.

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

Bit 3 – SEL External Source Select
This bit controls the source of the External 32.768 kHz Crystal Oscillator (XOSC32K).
Note: This bit is read-only when the oscillator is enabled (ENABLE bit is '1') or when the XOSC32K Status (XOSCS) bit in the Main Clock Status (CLKCTRL.MCLKSTATUS) register is '1'.

Value	Name	Description
0	CRYSTAL	An external crystal is connected to the XTAL32K1 and XTAL32K2 pins
1	EXTCLK	An external clock is connected to the XTAL32K1 pin

Bits 2:1 – PMODE[1:0] Power Mode
This bit field sets the External 32.768 kHz Crystal Oscillator (XOSC32K) power mode.
Note: Selecting the power mode requires a basic understanding of crystal oscillators. Changing the power mode may alter the start-up time of a crystal.

Value	Name	Description
0x0	LOW	Low power mode. Typical crystal values: C_L = 8pF, ESR = 50 kΩ
0x1	MEDIUM	Medium power mode. Typical crystal values: C_L = 12.5pF, ESR = 70 kΩ
0x2	HIGH	High power mode. Typical crystal values: C_L = 24pF, ESR = 100 kΩ
0x3	—	Reserved

Bit 0 – ENABLE Crystal Oscillator Enable

This bit controls whether the External 32.768 kHz Crystal Oscillator (XOSC32K) is enabled.

Note: When this bit is enabled, the SEL and CSUT bits become read-only.

Value	Description
0	The External 32.768 kHz Crystal Oscillator (XOSC32K) is disabled
1	The External 32.768 kHz Crystal Oscillator (XOSC32K) is enabled and overrides ordinary Port pin operation for the respective oscillator pins (XTAL32K1 and XTAL32K2)

13. SLPCTRL - Sleep Controller

13.1. 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

13.2. Overview

Sleep modes are used to shut down peripherals and clock domains in the device 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.

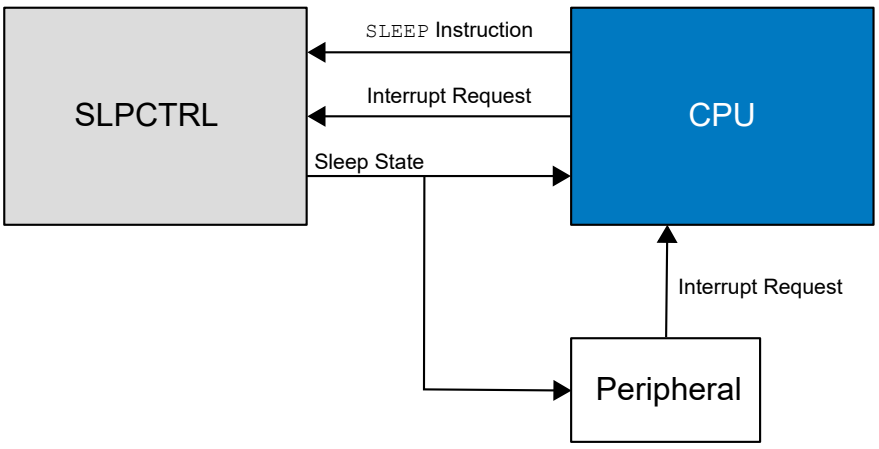
All sleep modes are available and can be entered from the Active mode. In Active mode, the CPU is executing application code. When the device enters sleep mode, the program execution is stopped. The application code decides which sleep mode to enter and when.

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.

13.2.1. Block Diagram

Figure 13-1. SLPCTRL Block Diagram



13.3. Functional Description

13.3.1. Initialization

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

1. Configure and enable the interrupts that can wake the device from sleep.

Enable also the 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.

13.3.2. Operation

13.3.2.1. Sleep Modes

Three different sleep modes can be enabled to reduce power consumption.

Idle The CPU stops executing code, resulting in reduced power consumption.
All peripherals are running, and all interrupt sources can wake the device.

Standby All high-frequency clocks are stopped apart from any peripheral or clock that are enabled to run in Standby sleep mode. This is enabled by writing the corresponding RUNSTDBY bit to '1'. The power consumption is dependent on the enabled functionality.
A subset of interrupt sources can wake the device(1).

Power-Down All high-frequency clocks are stopped, resulting in a power consumption lower than the Idle sleep mode.
A subset of the peripherals are running, and a subset of interrupt sources can wake the device(1).

Note:

1. Refer to the *Sleep Mode Activity* tables for further information.

Refer to the *Wake-up Time* section for information on how the wake-up time is affected by the different sleep modes.

Table 13-1. Sleep Mode Activity Overview for Peripherals

Peripheral	Active in Sleep Mode		
	Idle	Standby	Power-Down
CPU	-	-	-
RTC	X	X(1,2)	X(2)
WDT	X	X	X
BOD	X	X	X
EVSYS	X	X	X
NVM(3)	X	X	X
AC0	X	X(1)	-
ADC0			
CCL			
TCBn			
TCE0			
All other peripherals	X	-	-

Notes:

1. The RUNSTDBY bit of the corresponding peripheral must be set to enter an active state.
2. In Standby sleep mode, only the RTC functionality requires the RUNSTDBY bit to be set to enter an active state. In Power-Down sleep mode, only the PIT functionality is available.
3. Programming in progress will be completed, then the NVM peripheral will be disabled.

Table 13-2. Sleep Mode Activity Overview for Clock Sources

Clock Source	Active in Sleep Mode		
	Idle	Standby	Power-Down
Main clock source	X	X(1)	-
RTC clock source	X	X(1,2)	X(2)
WDT oscillator	X	X	X
BOD oscillator(3)	X	X	X
CCL clock source	X	X(1)	-

Notes:

1. The RUNSTDBY bit of the corresponding peripheral must be set to enter an active state.
2. In Standby sleep mode, only the RTC functionality requires the RUNSTDBY bit to be set to enter an active state. In Power-Down sleep mode, only the PIT functionality is available.
3. Sampled mode only.

Table 13-3. Sleep Mode Wake-Up Sources

Wake-up Source	Active in Sleep Mode		
	Idle	Standby	Power-Down
PORT Pin interrupt	X	X	X(1)
TWI Address Match interrupt	X	X	X
BOD VLM interrupt	X	X	X
CCL interrupts	X	X(2)	X(3)
RTC interrupts	X	X(2,4)	X(4)
USART Start-of-Frame interrupt	X	X(2)	-
TCE0 interrupts			
TCBn interrupts			
ADC0 interrupts			
AC0 interrupts			
All other interrupts	X	-	-

Notes:

1. The I/O pin has to be configured according to *Asynchronous Sensing Pin Properties* in the PORT section.
2. The RUNSTDBY bit of the corresponding peripheral must be set to enter an active state.
3. The CCL can wake up the device if the path through LUTn is asynchronous (FILTSEL = 0x0 and EDGEDET = 0x0 in the LUT n Control A (CCL.LUTnCTRLA) register).
4. In Standby sleep mode, only the RTC functionality requires the RUNSTDBY bit to be set to enter an active state. In Power-Down sleep mode, only the PIT functionality is available.

13.3.2.2. 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 and the time it takes to start the regulator if it has been switched off:

- 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 internal 32.768 kHz oscillator and the Real-Time Clock (RTC) clock source may be running. These are used by the Brown-out Detector (BOD), Watchdog Timer (WDT) or Periodic Interrupt Timer (PIT). All the other clock sources will be OFF.

Table 13-4. Sleep Modes and Start-Up Time

Sleep Mode	Start-Up Time
Idle	Six clock cycles
Standby	Six clock cycles + one OSC start-up
Power-Down	Six clock cycles + one OSC start-up

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

In addition to the normal wake-up time, it is possible to make the device wait until the BOD is ready before executing the code. This is done by writing `0x3` to the BOD operation mode in the Active and Idle (ACTIVE) bit field in the BOD Configuration (FUSE.BODCFG) fuse. 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.

13.3.3. 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.

13.4. Register Summary

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

13.5. Register Description

13.5.1. 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.

14. RSTCTRL - Reset Controller

14.1. Features

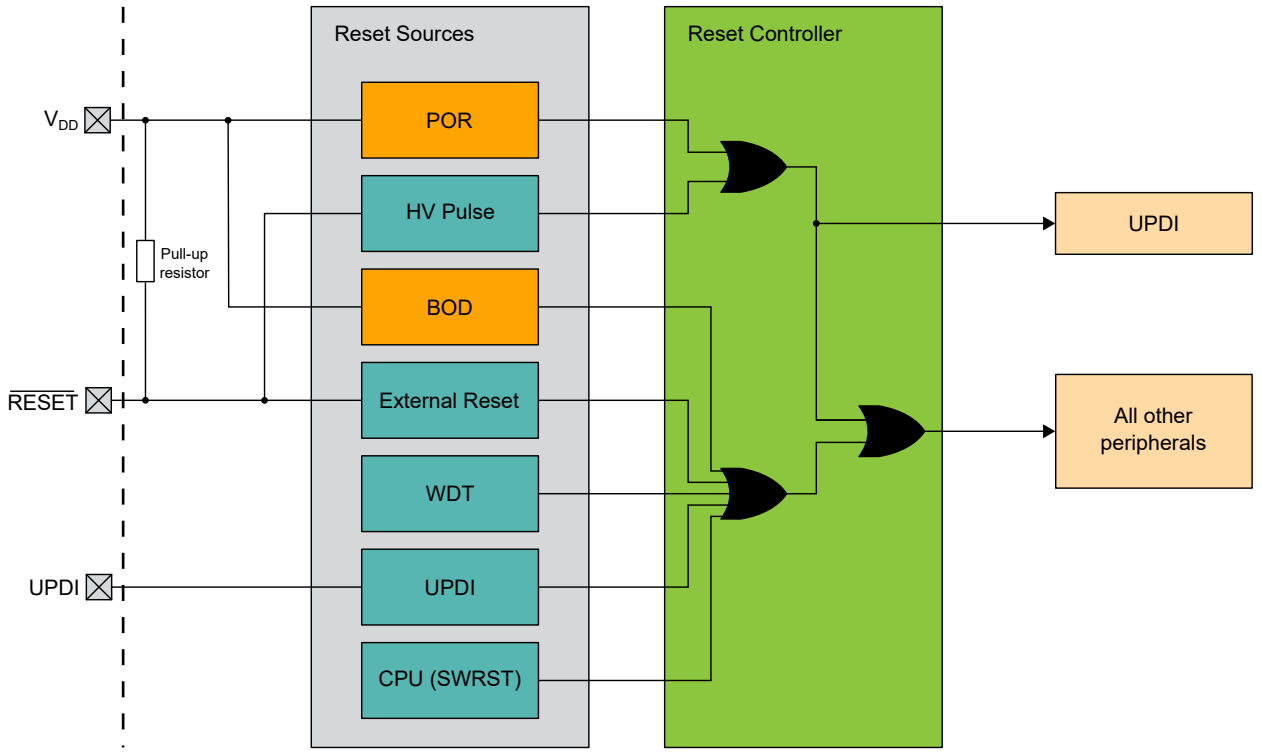
- Returns the Device to an Initial State after a Reset
- Identifies the Previous Reset Source
- Power Supply Reset Sources:
  - Power-on Reset (POR)
  - Brown-out Detector (BOD) Reset
- User Reset Sources:
  - External Reset ($\overline{\text{RESET}}$)
  - Watchdog Timer (WDT) Reset
  - Software Reset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

14.2. 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 software to identify the Reset source.

14.2.1. Block Diagram

Figure 14-1. Reset System Overview



14.2.2. Signal Description

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

Signal Description (continued)		
Signal	**Description**	**Type**
UPDI	Unified Program and Debug Interface	Digital input

14.3. Functional Description

14.3.1. Initialization

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

The registers in the device with automatic loading from the Fuses or the Signature Row are updated. The program counter will be set to `0x0000` after a Reset from any source.

14.3.2. Operation

14.3.2.1. 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.

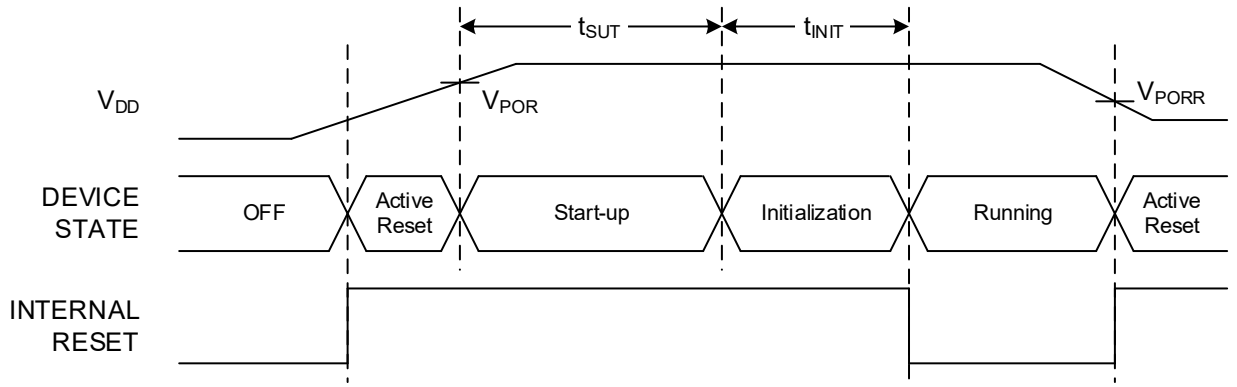
There are two types of Resets based on the source:

- 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)
  - Unified Program and Debug Interface (UPDI) Reset

14.3.2.1.1. Power-on Reset (POR)

The Power-on Reset (POR) aims to ensure a safe start-up of logic and memories, a process generated by an on-chip detection circuit that is always enabled. The POR is activated when the V_{DD} rises and sets an active Reset as long as the V_{DD} is below the POR threshold voltage (V_{POR}). The Reset will continue until the Start-up and Reset initialization sequence is completed. Fuses determine the Start-Up Time (SUT). Reset is activated again, without delay, when V_{DD} falls below the detection level (V_{PORR}).

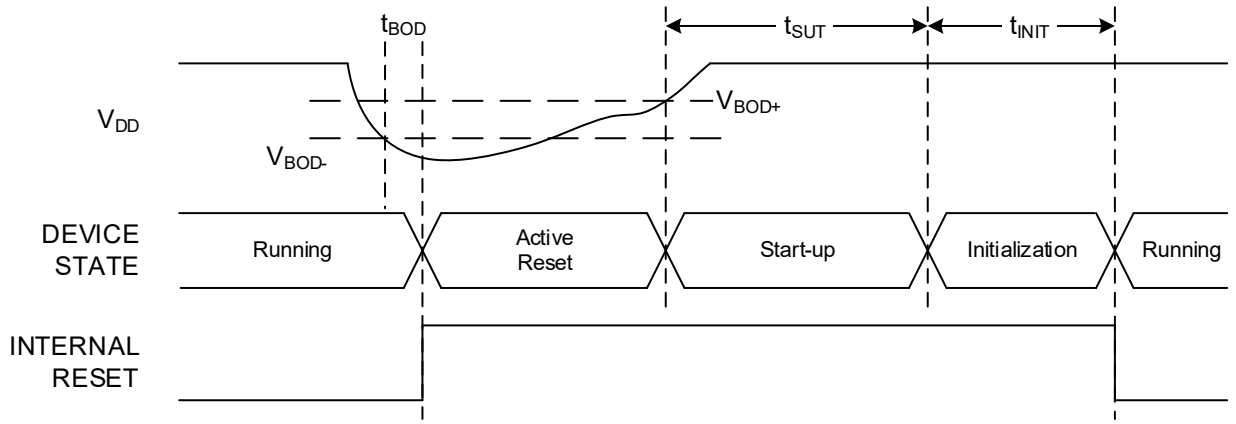
Figure 14-2. MCU Start-Up, $\overline{\text{RESET}}$ Tied to V_{DD}



14.3.2.1.2. Brown-out Detector (BOD) Reset

The on-chip Brown-out Detector (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 to ensure a safe operation during internal Reset and chip erase.

Figure 14-3. Brown-out Detector Reset

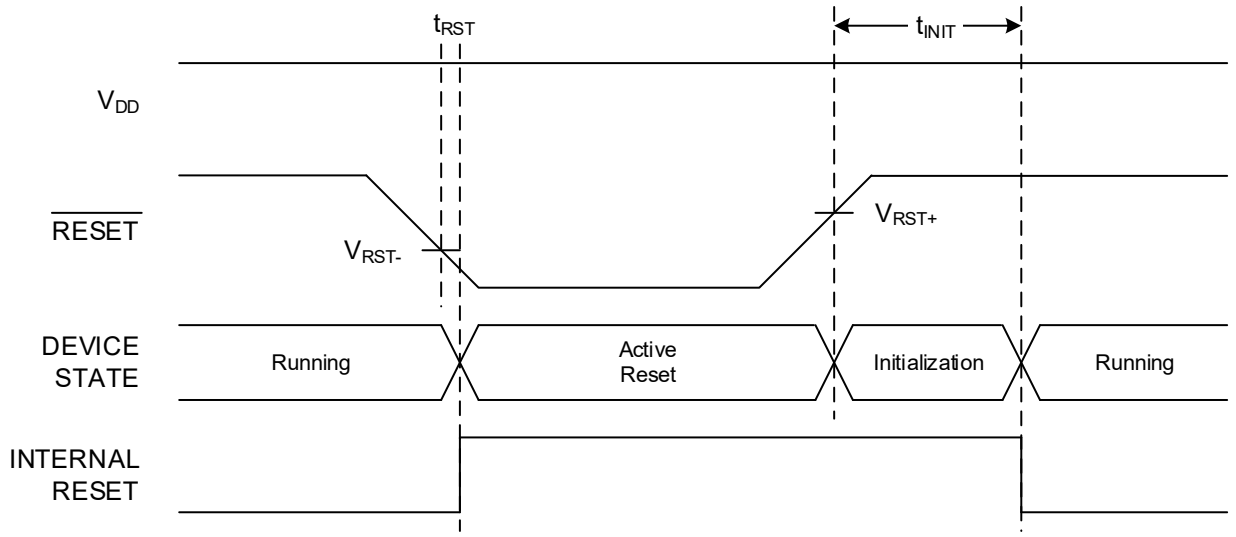


14.3.2.1.3. External Reset

The external Reset is enabled by a fuse. See the RSTPINCFG field in Pin Configuration (FUSE.PINCFG). Also, the internal pull-up resistor for the Reset-pin is enabled when the external Reset is enabled.

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.

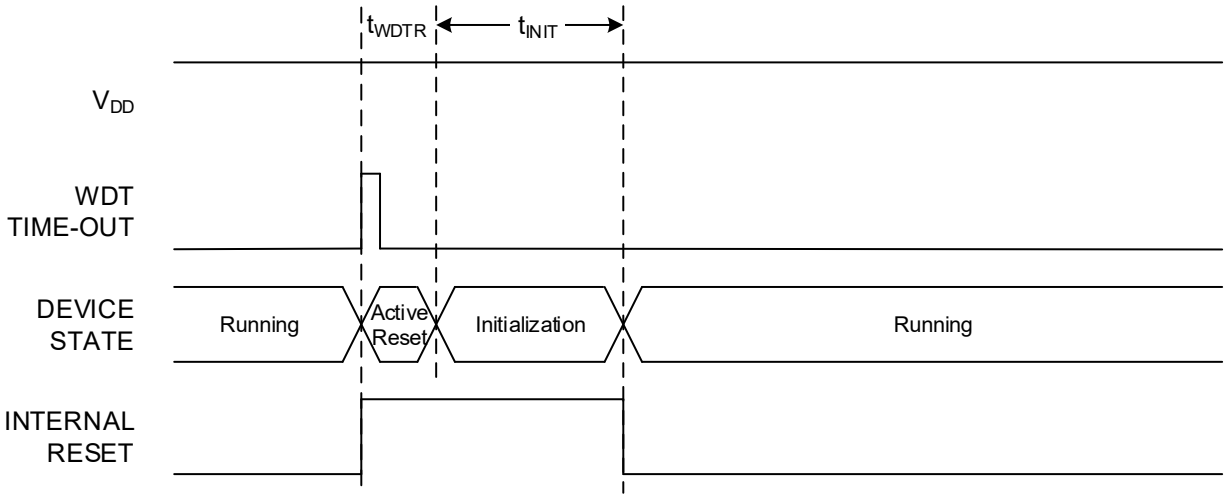
Figure 14-4. External Reset Characteristics



14.3.2.1.4. Watchdog Reset

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

Figure 14-5. Watchdog Reset



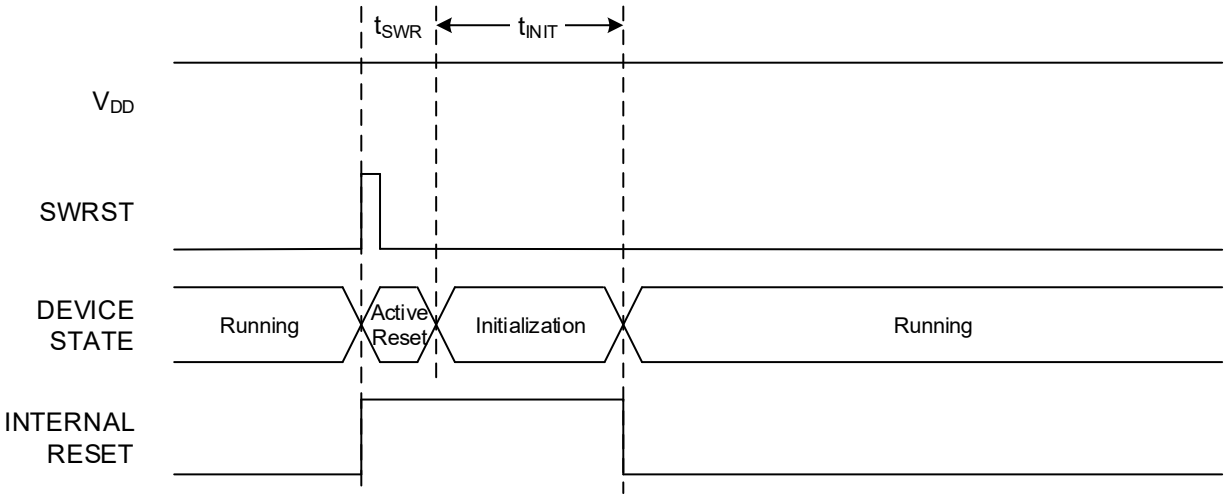
Note: The time t_{WDTR} is approximately 150 ns.

14.3.2.1.5. Software Reset (SWRST)

The Software Reset makes it possible to issue a System Reset from the software. Writing a '1' to the Software Reset (SWRST) bit in the Software Reset (RSTCTRL.SWRR) register generates the Reset.

The Reset sequence will start immediately after the bit is written.

Figure 14-6. Software Reset



Note: The time t_{SWR} is approximately 150 ns.

14.3.2.1.6. Unified Program and Debug Interface (UPDI) Reset

The Unified Program and 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. Find more details in the *UPDI - Unified Program and Debug Interface* section.

14.3.2.1.7. High Voltage (HV) Pulse

A device Reset is issued if a high voltage is applied to or removed from the $\overline{\text{RESET}}$ pin. Using the HV pulse to enable the UPDI will trigger a device reset. Refer to the *UPDI - Unified Program and Debug Interface* section for more information on the HV pulse.

14.3.2.1.8. Domains Affected By Reset

The following logic domains are affected by the various Resets:

Table 14-1. Logic Domains Affected by Various Resets

Reset Type	Fuses are Reloaded	Reset of UPDI	Reset of Other Volatile Logic
POR	X	X	X
BOD	X		X
External Reset	X		X
Watchdog Reset	X		X
Software Reset	X		X
UPDI Reset	X		X
High Voltage (HV) Pulse(1)	X	X	X

Note: 1. An HV pulse can cause a Reset, but it must not be used intentionally as a Reset source.

14.3.2.2. Reset Time

The Reset time can be divided into two parts.

The first part is when any of the Reset sources are active, which 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 when 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. If a Power Supply Reset Source has triggered the Reset, the time will increase by the start-up time specified in the Start-Up Time (SUT) bit field within the System Configuration 1 (FUSE.SYSCFG1) fuse. The internal Reset initialization time will increase if the Cyclic Redundancy Check Memory Scan (CRCSCAN) is set to run at start-up. It is possible to change this configuration in the CRC-on-Boot (CRCBOOT) bit field section in the System Configuration 0 (FUSE.SYSCFG0) fuse.

14.3.3. Sleep Mode Operation

The RSTCTRL operates in Active mode and all sleep modes.

14.3.4. 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 14-2. RSTCTRL - Registers Under Configuration Change Protection

Register	Key
RSTCTRL.SWRR	IOREG

14.4. 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

14.5. Register Description

14.5.1. Reset Flag Register

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

The Reset flags can be cleared by writing a '1' to the respective flag. All the flags will be cleared by a Power-on Reset (POR) or a Brown-out Reset (BOR), except for the Power-on Reset (PORF) and Brown-out Reset (BORF) flags.

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 either a UPDI Reset occurs or a Reset caused by an HV pulse 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.

14.5.2. 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'.

15. CPUINT - CPU Interrupt Controller

15.1. 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)

15.2. Overview

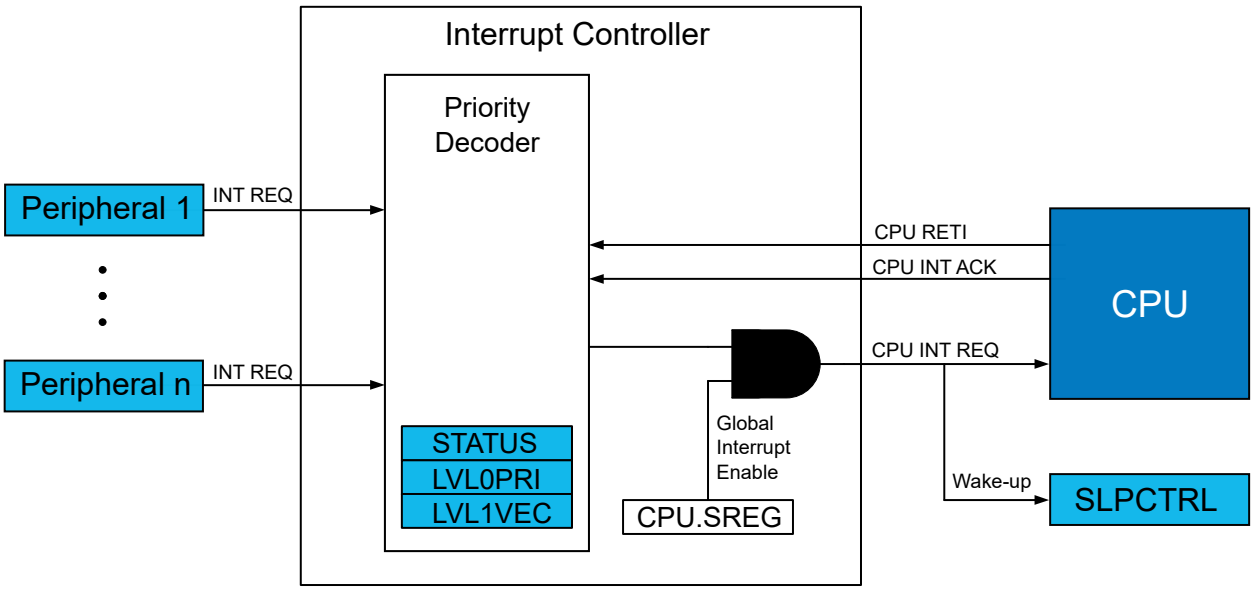
An interrupt request signals a state change inside a peripheral and can be used to alter the program execution. The peripherals can have one or more interrupts. All interrupts are individually enabled and configured. When an interrupt is enabled and configured, it will generate an interrupt request when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

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 servicing all interrupts within a certain amount of time.

15.2.1. Block Diagram

Figure 15-1. CPUINT Block Diagram



15.3. Functional Description

15.3.1. Initialization

Initialize an interrupt in the following order:

1. Optional: Configure the expected location of the interrupt vectors using the IVSEL bit in the Control A (CPUINT.CTRLA) register.
2. Optional: Enable compact vector table by writing '1' to the CVT bit in the Control A (CPUINT.CTRLA) register.
3. Optional: Enable vector prioritizing by round robin by writing a '1' to the Round Robin Priority Enable (LVL0RR) bit in CPUINT.CTRLA.
4. Optional: Select the Priority Level 1 vector by writing the interrupt vector number to the Interrupt Vector with Priority Level 1 (CPUINT.LVL1VEC) register.
5. Optional: Modify the priority of the LVL0 interrupts by configuring Interrupt Priority Level 0 (LVL0PRI) register.
6. Configure the interrupt conditions within each peripheral and enable the peripheral's interrupt.
7. Enable interrupts globally by writing a '1' to the Global Interrupt Enable (I) bit in the CPU Status (CPU.SREG) register.

15.3.2. Operation

15.3.2.1. 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.

15.3.2.2. Interrupt Vector Locations

The expected location of interrupt vectors 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.

15.3.2.3. Interrupt Response Time

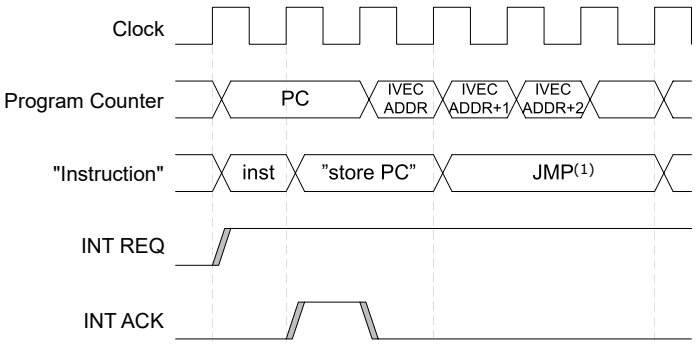
The minimum interrupt response time is represented in the following table.

Table 15-1. 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`)

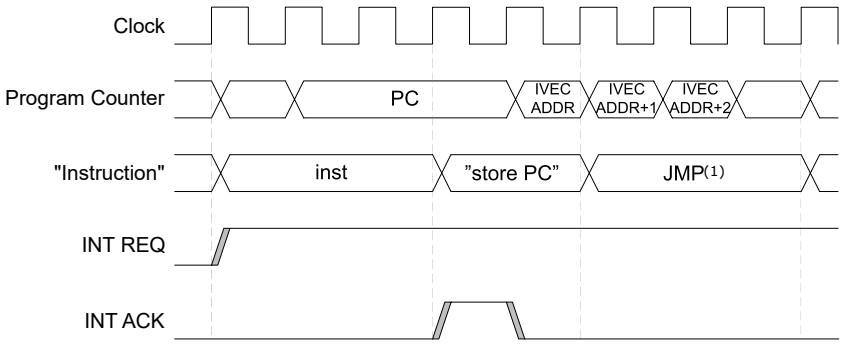
After the Program Counter is pushed on the stack, the program vector for the interrupt is executed. See the following figure.

Figure 15-2. Interrupt Execution of Single-Cycle Instruction



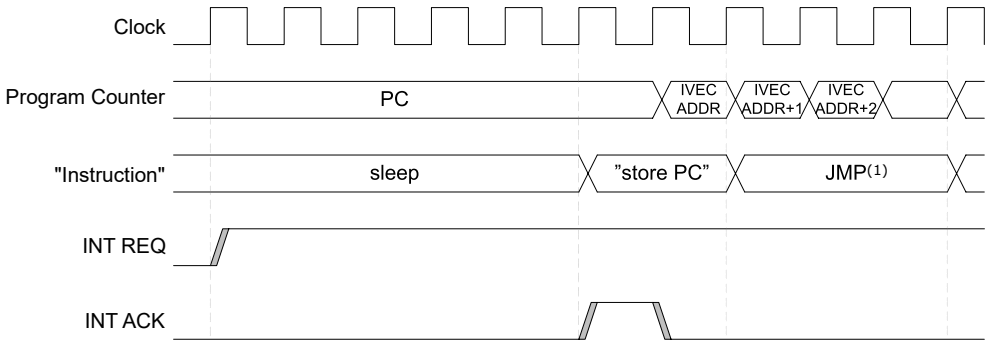
If an interrupt occurs during the execution of a multi-cycle instruction, the instruction is completed before the interrupt is served, as shown in the following figure.

Figure 15-3. 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 15-4. 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.

15.3.2.4. 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 15-2. 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

15.3.2.4.1. Non-Maskable Interrupts

A Non-Maskable Interrupt (NMI) will be executed regardless of the I bit setting 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* table of the device for available NMI sources.

15.3.2.4.2. 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.

15.3.2.4.3. Normal-Priority Interrupts

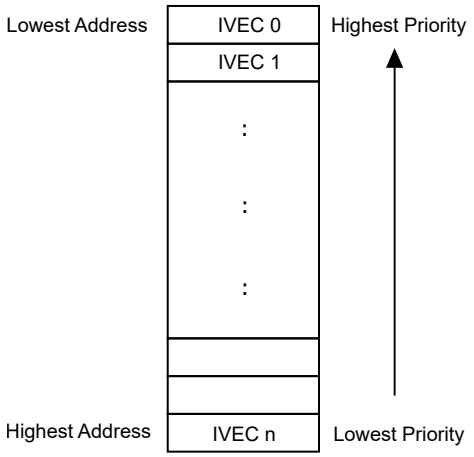
All interrupt vectors other than NMI are assigned to priority level 0 (normal) by default. The user may override this by assigning one of these vectors as a high-priority vector. The device will have many normal-priority vectors, and some of these may be pending at the same time. Two different scheduling 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* section. 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.

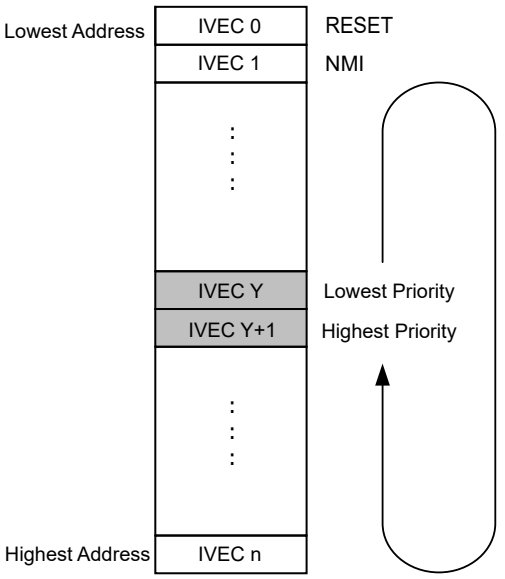
Figure 15-5. Default Static Scheduling



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 15-6. Static Scheduling When CPUINT.LVL0PRI Is Different from Zero



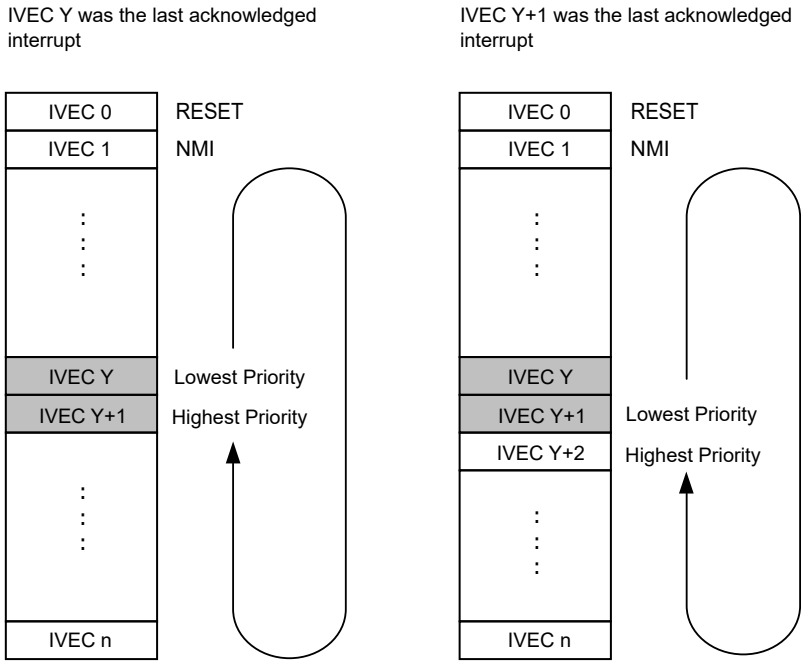
Here, value Y has been written to CPUINT.LVL0PRI so that the interrupt vector Y+1 has the highest priority. Note that, In this case, the priorities will wrap so that the lowest address no longer has the highest priority, not including RESET and NMI, which will always have the highest priority.

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 15-7. 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.

15.3.2.5. Compact Vector Table

The Compact Vector Table (CVT) is a feature to allow the 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 few of interrupt generators.

15.3.3. 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.

15.3.4. 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 15-3. CPUINT - Registers Under Configuration Change Protection

Register	Key
The IVSEL and CVT bit fields in CPUINT.CTRLA	IOREG

15.4. 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]							

15.5. Register Description

15.5.1. 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
When the entire Flash is configured as a BOOT section, this bit will be ignored.

Value	Description
0	The expected location of the interrupt vectors is directly after the BOOT section(1)
1	The expected location of the interrupt vectors is at the start of the BOOT section

Note:
1. A system reset will cause the Program Counter to be reset to 0x0000, regardless of the IVSEL bit value.

Bit 5 – CVT Compact Vector Table

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

15.5.2. 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.

15.5.3. 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.

15.5.4. 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.

16. EVSYS - Event System

16.1. Features

- System for Direct Peripheral-to-Peripheral Signaling
- Peripherals Can Directly Produce, Use, and React to Peripheral Events
- Short and Predictable Response Time
- Up to 4 Parallel Event Channels Available
- Each Channel Is Driven by One Event Generator and Can Have Multiple Event Users
- Events Can Be Sent and/or Received by Most Peripherals and by Software
- The Event System Works in Active, Idle, and Standby Sleep Modes

16.2. 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 a short and predictable response time between peripherals, allowing for autonomous peripheral control and interaction, and for synchronized timing of actions in several peripheral modules. Thus, the EVSYS peripheral makes it possible to implement Core Independent Peripherals (CIPs). Also, it is a powerful tool for reducing the complexity, size, and execution time of the software.

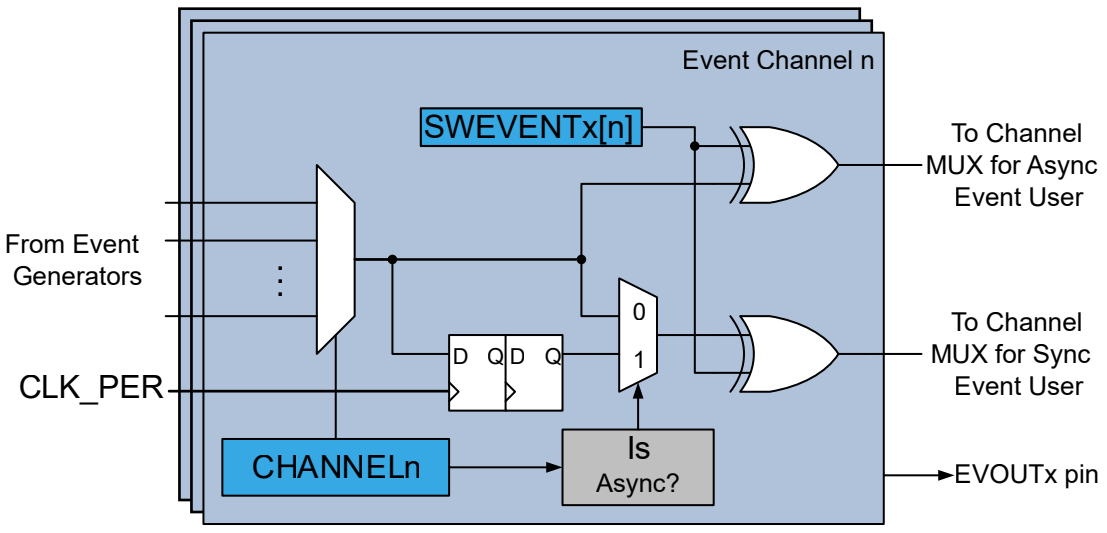
A change of the event generator's state is referred to as an event and usually corresponds to one of the peripheral's interrupt conditions. Events can be forwarded directly to other peripherals using the dedicated event routing network. The routing of each channel is configured in software, including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

Only one event signal can be routed on each channel. Multiple peripherals can use events from the same channel.

The EVSYS can connect peripherals such as ADCs, 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.

16.2.1. Block Diagram

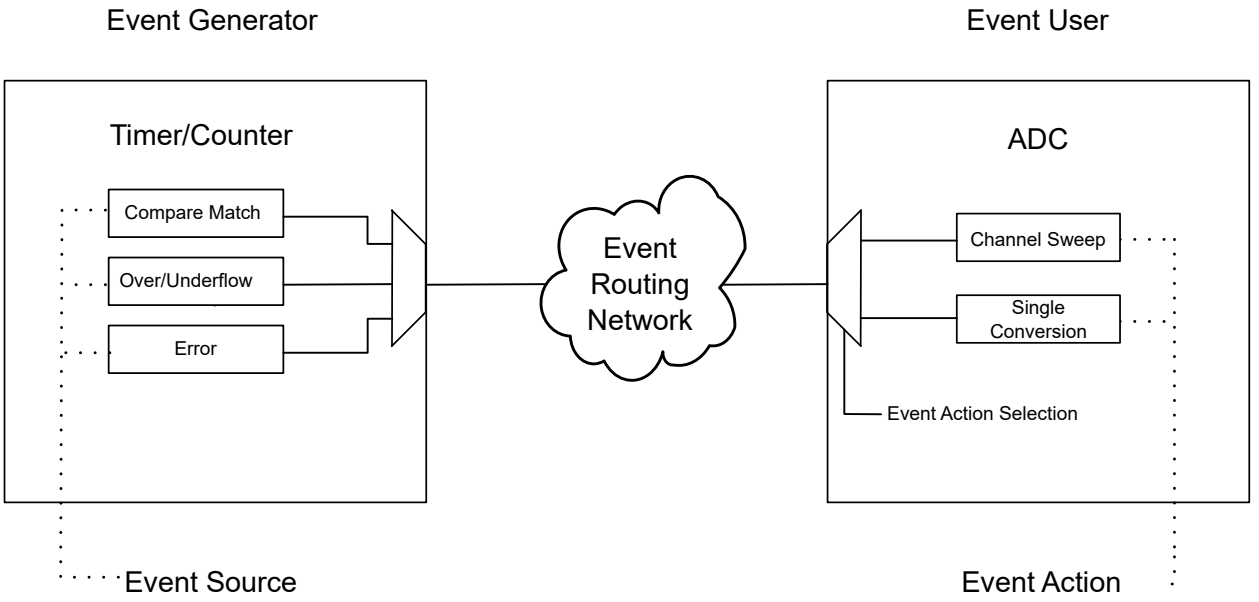
Figure 16-1. EVSYS Block Diagram



The block diagram shows the operation of an event channel. A multiplexer controlled by Channel n Generator Selection (EVSYS.CHANNELn) register at the input selects which of the event sources to route onto the event channel. Each event channel has two subchannels: one asynchronous and one synchronous. A synchronous user will listen to the synchronous subchannel, and an asynchronous user will listen to the asynchronous subchannel.

An event signal from an asynchronous source will be synchronized by the Event System before being routed to the synchronous subchannel. An asynchronous event signal to be used by a synchronous consumer must last for at least one peripheral clock cycle to ensure that it will propagate through the synchronizer. The synchronizer will delay such an event between two and three clock cycles, depending on when the event occurs.

Figure 16-2. Example of Event Source, Generator, User, and Action



16.2.2. Signal Description

Signal	Type	Description
EVOUTx	Digital output	Event output, one output per I/O Port

16.3. Functional Description

16.3.1. Initialization

To utilize events, the Event System, the generating peripheral, and the peripheral(s) using the event must be set up accordingly:

1. Configure the generating peripheral appropriately. For example, if the generating peripheral is a timer, set the prescaling, the Compare register, etc., so that the desired event is generated.
2. Configure the event user peripheral(s) appropriately. For example, if the ADC is the event user, set the ADC prescaler, resolution, conversion time, etc., as desired, and configure the ADC conversion to start at the reception of an event.
3. Configure the Event System to route the desired source. In this case, the Timer/Compare match to the desired event channel. This may, for example, be Channel 0, which is accomplished by writing to the Channel 0 Generator Selection (EVSYS.CHANNEL0) register.
4. Configure the ADC to listen to this channel by writing to the corresponding User x Channel Selection (EVSYS.USERx) register.

16.3.2. Operation

16.3.2.1. Event User Multiplexer Setup

Each event user has one dedicated event user multiplexer selecting which event channel to listen to. The application configures these multiplexers by writing to the corresponding EVSYS.USERx register.

16.3.2.2. Event System Channel

An event channel can be connected to one of the event generators.

The source for each event channel is configured by writing to the respective Channel n Generator Selection (EVSYS.CHANNELn) register.

16.3.2.3. Event Generators

Each event channel has several possible event generators, but only one can be selected at a time. The event generator for a channel is selected by writing to the respective Channel n Generator Selection (EVSYS.CHANNELn) register. By default, the channels are not connected to any event generator. For details on event generation, refer to the documentation of the corresponding peripheral.

A generated event is either synchronous or asynchronous to the device peripheral clock (CLK_PER). Asynchronous events can be generated outside the normal edges of the peripheral clock, making the system respond faster than the selected clock frequency would suggest. Asynchronous events can also be generated while the device is in a sleep mode when the peripheral clock is not running.

Any generated event is classified as either a pulse event or a level event. In both cases, the event can be either synchronous or asynchronous, with properties according to the table below.

Table 16-1. Properties of Generated Events

Event Type	Sync/Async	Description
Pulse	Sync	An event generated from CLK_PER that lasts one clock cycle
	Async	An event generated from a clock other than CLK_PER lasting one clock cycle
Level	Sync	An event generated from CLK_PER that lasts multiple clock cycles
	Async	An event generated without a clock (for example, a pin or a comparator), or an event generated from a clock other than CLK_PER that lasts multiple clock cycles

The properties of both the generated event and the intended event user must be considered in order to ensure reliable and predictable operation.

The table below shows the available event generators for this device family.

Table 16-2. Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
UPDI	SYNCH	SYNCH character	Async, Level	CLK_PDI	SYNCH character on PDI RX input synchronized to CLK_PDI
RTC	OVF	Overflow	Async, Pulse	CLK_RTC	One CLK_RTC period
	CMP	Compare Match			
	EVGEN0	Selectable prescaled RTC event	Async, Level		Given by prescaled RTC clock divided by prescale factor
	EVGEN1				
CCL	LUTn	LUT output level	Async, Level	Asynchronous	Depends on the CCL configuration
AC0	OUT	Comparator output level	Async, Level	Asynchronous	Given by the AC output level

Table 16-2. Event Generators (continued)

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
ADC0	RES	Result ready	Sync, Pulse	CLK_PER	One CLK_PER period
	SAMP	Sample ready			
	WCMP	Window comparison match			
PORTx	EVGEN0	Pin level	Async, Level	Asynchronous	Given by the pin level
	EVGEN1				
USART0	XCK	Clock signal in SPI host mode and synchronous USART host mode	Sync, Level	CLK_PER	Minimum two CLK_PER periods
SPI0	SCK	SPI host clock signal	Sync, Level	CLK_PER	Minimum two CLK_PER periods
TCE0	OVF	Overflow/Low-byte timer underflow	Sync, Pulse	CLK_PER	One CLK_PER period
	CMP0	Compare channel 0 match/ Low-byte timer compare channel 0 match	Sync, Pulse/ Level		One CLK_PER period or WOn
	CMP1	Compare channel 1 match/ Low-byte timer compare channel 1 match			
	CMP2	Compare channel 2 match/ Low-byte timer compare channel 2 match			
TCBn	CAPT	CAPT interrupt flag set	Sync, Pulse	CLK_PER	One CLK_PER period or WO
	OVF	Counter overflow	Sync, Pulse/ Level		One CLK_PER period

16.3.2.4. Event Users

The event channel to listen to is selected by configuring the event user. An event user may require the event signal to be either synchronous or asynchronous to the peripheral clock. An asynchronous event user can respond to events in sleep modes when clocks are not running. Such events can be responded to outside the normal edges of the peripheral clock, making the event user respond faster than the clock frequency would suggest. For details on the requirements of each peripheral, refer to the documentation of the corresponding peripheral.

Most event users implement edge or level detection to trigger actions in the corresponding peripheral based on the incoming event signal. In both cases, a user can either be synchronous, which requires that the incoming event is generated from the peripheral clock (CLK_PER), or asynchronous, if not. Some asynchronous event users do not apply event input detection but use the event signal directly. The different event user properties are described in general in the table below.

Table 16-3. Properties of Event Users

Input Detection	Async/Sync	Description
Edge	Sync	An event user is triggered by an event edge and requires that the incoming event is generated from CLK_PER
	Async	An event user is triggered by an event edge and has asynchronous detection or an internal synchronizer
Level	Sync	An event user is triggered by an event level and requires that the incoming event is generated from CLK_PER
	Async	An event user is triggered by an event level and has asynchronous detection or an internal synchronizer
No detection	Async	An event user will use the event signal directly

The table below shows the available event users for this device family.

Table 16-4. Event Users

USER Name		Description	Input Detection	Async/Sync
Peripheral	Input			
CCL	LUTnx	LUTn event input x	Level (No Detection)	Async
AC0	SAMPLE	AC Sample on event	Edge	Async
ADC0	START	ADC start on event	Edge	Async
EVSYS	EVOUTx	EVSYS pin output x	Level (No detection)	Async
USART0	RXD	USART0 RxD event input	Level	Sync
TCE0	CNTA	Count on positive event edge	Edge	Sync
		Count on any event edge	Edge	
		Count while event signal is high	Level	
		Event level controls count direction	Level	
	CNTB	Event level controls count direction	Level	Sync
		Restart counter on positive event edge	Edge	
		Restart counter on any event edge	Edge	
		Restart counter while event signal is high	Level	
TCBn	CAPT	Time-out check	Edge	Sync
		Input capture on event	Edge	
		Input capture frequency measurement	Edge	
		Input capture pulse-width measurement	Edge	
		Input capture frequency and pulse-width measurement	Edge	
		Single shot	Edge	Both
	COUNT	Count on event	Edge	Sync

16.3.2.5. Synchronization

Events can be either synchronous or asynchronous to the peripheral clock. Each Event System channel has two subchannels: one asynchronous and one synchronous.

The asynchronous subchannel is identical to the event output from the generator. If the event generator generates a signal asynchronous to the peripheral clock, the signal on the asynchronous subchannel will be asynchronous. If the event generator generates a signal synchronous to the peripheral clock, the signal on the asynchronous subchannel will also be synchronous.

The synchronous subchannel is identical to the event output from the generator, if the event generator generates a signal synchronous to the peripheral clock. If the event generator generates a signal asynchronous to the peripheral clock, this signal is first synchronized before being routed onto the synchronous subchannel. Depending on when it occurs, synchronization will delay the event by two to three clock cycles. The Event System automatically performs this synchronization if an asynchronous generator is selected for an event channel.

16.3.2.6. Software Event

The application can generate a software event. Software events on Channel n are issued by writing a '1' to the Software Event Channel Select (CHANNEL[n]) bit in the Software Events (EVSYS.SWEVENTx) register. A software event appears as a pulse on the Event System channel, inverting the current event signal for one clock cycle.

Event users see software events as no different from those produced by event generating peripherals.

16.3.3. Sleep Mode Operation

When configured, the Event System will work in all sleep modes. Software events represent one exception since they require a peripheral clock.

Asynchronous event users are able to respond to an event without their clock running in Standby sleep mode. Synchronous event users require their clock to be running to be able to respond to events. Such users will only work in Idle sleep mode or in Standby sleep mode, if configured to run in Standby mode by setting the RUNSTDBY bit in the appropriate register.

Asynchronous event generators are able to generate an event without their clock running, that is, in Standby sleep mode. Synchronous event generators require their clock to be running to be able to generate events. Such generators will only work in Idle sleep mode or in Standby sleep mode, if configured to run in Standby mode by setting the RUNSTDBY bit in the appropriate register.

16.3.4. Debug Operation

This peripheral is unaffected by entering Debug mode.

16.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	SWEVENTA	7:0					CH3	CH2	CH1	CH0
0x01 ... 0x0F	Reserved									
0x10	CHANNEL0	7:0	CHANNEL[7:0]							
0x11	CHANNEL1	7:0	CHANNEL[7:0]							
0x12	CHANNEL2	7:0	CHANNEL[7:0]							
0x13	CHANNEL3	7:0	CHANNEL[7:0]							
0x14 ... 0x1F	Reserved									
0x20	USERCCLLUT0A	7:0	USER[7:0]							
0x21	USERCCLLUT0B	7:0	USER[7:0]							
0x22	USERCCLLUT1A	7:0	USER[7:0]							
0x23	USERCCLLUT1B	7:0	USER[7:0]							
0x24	USERADC0START	7:0	USER[7:0]							
0x25	USEREVSYSEVOUTA	7:0	USER[7:0]							
0x26	USEREVSYSEVOUTC	7:0	USER[7:0]							
0x27	USEREVSYSEVOUTD	7:0	USER[7:0]							
0x28	USEREVSYSEVOUTF	7:0	USER[7:0]							
0x29	USERUSART0RXD	7:0	USER[7:0]							
0x2A	USERTCE0CNTA	7:0	USER[7:0]							
0x2B	USERTCE0CNTB	7:0	USER[7:0]							
0x2C	USERTCB0CAPT	7:0	USER[7:0]							
0x2D	USERTCB0COUNT	7:0	USER[7:0]							
0x2E	USERTCB1CAPT	7:0	USER[7:0]							
0x2F	USERTCB1COUNT	7:0	USER[7:0]							
0x30	USER16	7:0	USER[7:0]							
0x31	USER17	7:0	USER[7:0]							
0x32	USER18	7:0	USER[7:0]							
0x33	USER19	7:0	USER[7:0]							

16.5. Register Description

16.5.1. Software Events A

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

Refer to the *Peripheral Overview* section for the available Event System channels.

Bit	7	6	5	4	3	2	1	0
					CH3	CH2	CH1	CH0
Access					W	W	W	W
Reset					0	0	0	0

Bits 0, 1, 2, 3 – CHn Channel n Select
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will generate a single-pulse event on event channel n by inverting the signal on the event channel for one peripheral clock (CLK_PER) cycle.

16.5.2. Channel n Generator Selection

Name: CHANNELn
Offset: 0x10 + n*0x01 [n=0..3]
Reset: 0x00
Property: -

Each event channel can be connected to a single event generator.

Refer to the *Peripheral Overview* section for the available Event System channels.

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

Bits 7:0 – CHANNEL[7:0] Channel Generator Selection
This bit field controls which event generator is connected to this event channel.
Note: Not all generators can be connected to all channels. Refer to the table below for further details.

Notes:

1. Not all peripheral instances are available for all pin counts. Refer to the *Peripherals and Architecture* chapter for details.
2. Event from PORT pin will be zero if input driver is disabled.
3. The operational mode of the timer decides when the CAPT flag is raised. Refer to the TCB section for details.

GENERATOR			Async/Sync	Description	Channel Availability
Value	Name				
	Peripheral[1]	Output			
0x01	UPDI	SYNCH	Async	Rising edge of SYNCH character detection	All channels
0x06	RTC	OVF	Async	Counter overflow	All channels
0x07		CMP		Compare match	
0x08		EVGEN0		Selectable prescaled RTC event	
0x09		EVGEN1			
0x10	CCL	LUT0	Async	LUT output level	All channels
0x11		LUT1			
0x12		LUT2			
0x13		LUT3			
0x20	AC0	OUT	Async	Comparator output level	All channels
0x24	ADC0	RES	Sync	Result ready	All channels
0x25		SAMP		Sample ready	
0x26		WCMP		Window comparison match	
0x40	PORTA	EVGEN0	Async	Pin level[2]	All channels
0x41		EVGEN1			
0x44	PORTC	EVGEN0	Async	Pin level[2]	All channels
0x45		EVGEN1			
0x46	PORTD	EVGEN0	Async	Pin level[2]	All channels
0x47		EVGEN1			

CHANNEL (continued)

GENERATOR			Async/Sync	Description	Channel Availability
Value	Name				
	Peripheral[1]	Output			
0x4A	PORTF	EVGEN0	Async	Pin level[2]	All channels
0x4B		EVGEN1			
0x60	USART0	XCK	Sync	Clock signal in SPI Host mode and synchronous USART Host mode	All channels
0x68	SPI0	SCK	Sync	SPI Host clock signal	All channels
0x80	TCE0	OVF	Sync	Overflow/Low-byte timer underflow	All channels
0x84		CMP0	Sync	Compare channel 0 match/Low-byte timer compare channel 0 match	
0x85		CMP1		Compare channel 1 match/Low-byte timer compare channel 1 match	
0x86		CMP2		Compare channel 2 match/Low-byte timer compare channel 2 match	
0xA0	TCB0	CAPT	Sync	CAPT Interrupt flag set[3]	All channels
0xA1		OVF		Counter overflow	
0xA2	TCB1	CAPT	Sync	CAPT Interrupt flag set[3]	All channels
0xA3		OVF		Counter overflow	

16.5.3. User x Channel Selection

Name: USERx
Offset: 0x20 + x*0x01 [x=0..19]
Reset: 0x00
Property: -

Each event user can be connected to a single channel, and multiple users can be connected to the same channel.

Refer to the *Register Summary* section for all available User x Channel Selection (USERx) registers.

Refer to the *Event Users* section for further details.

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

Bits 7:0 – USER[7:0] User Channel Selection
This bit field controls which event channel this event user is connected to.

Value	Name	Description
0x0	OFF	This Event System user is not connected to any channel
0x1	CHANNEL0	This Event System user is connected to Event Channel 0
0x2	CHANNEL1	This Event System user is connected to Event Channel 1
0x3	CHANNEL2	This Event System user is connected to Event Channel 2
0x4	CHANNEL3	This Event System user is connected to Event Channel 3
Other	-	Reserved

17. PORTMUX - Port Multiplexer

17.1. Overview

The Port Multiplexer (PORTMUX) can enable or disable pin functionality or switch between default and alternative pin positions. Available options are described in detail in the PORTMUX register map and depend on the actual pin and its properties.

For available pins and functionality, refer to the *I/O Multiplexing and Considerations* chapter.

17.2. Register Summary - PORTMUX

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	EVSYSROUTEA	7:0			EVOUTF		EVOUTD			EVOUTA
0x01	CCLROUTEA	7:0						LUT2		LUT0
0x02	USARTROUTEA	7:0						USART0[2:0]		
0x03 ... 0x04	Reserved									
0x05	SPIROUTEA	7:0						SPI0[2:0]		
0x06	TWIROUTEA	7:0							TWI0[1:0]	
0x07	TCEROUTEA	7:0						TCE0[2:0]		
0x08	TCBROUTEA	7:0							TCB1	TCB0
0x09	Reserved									
0x0A	ACROUTEA	7:0								AC0

17.3. Register Description

17.3.1. EVSYS Pin Position

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

Bit	7	6	5	4	3	2	1	0
			EVOUTF		EVOUTD			EVOUTA
Access			R/W		R/W			R/W
Reset			0		0			0

Bit 5 – EVOUTF Event Output F
This bit controls the pin position for event output F.

Value	Name	Description
0	DEFAULT	EVOUT on PF2
1	ALT1	EVOUT on PF7

Bit 3 – EVOUTD Event Output D
This bit controls the pin position for event output D.

Value	Name	Description
0	DEFAULT	EVOUT on PD2
1	ALT1	EVOUT on PD7

Bit 0 – EVOUTA Event Output A
This bit controls the pin position for event output A.

Value	Name	Description
0	DEFAULT	EVOUT on PA2
1	ALT1	EVOUT on PA7

17.3.2. CCL LUTn Pin Position

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

Bit	7	6	5	4	3	2	1	0
						LUT2		LUT0
Access						R/W		R/W
Reset						0		0

Bit 2 – LUT2 CCL LUT 2 Signals
This bit field controls the pin positions for CCL LUT 2 signals.

Value	Name	Description			
		OUT	IN0	IN1	IN2
0	DEFAULT	PD3	PD0	PD1	PD2
1	ALT1	PD6	PD0	PD1	PD2

Bit 0 – LUT0 CCL LUT 0 Signals
This bit field controls the pin positions for CCL LUT 0 signals.

Value	Name	Description			
		OUT	IN0	IN1	IN2
0	DEFAULT	PA3	PA0	PA1	PA2
1	ALT1	PA6	PA0	PA1	PA2

17.3.3. USART 0 Pin Position

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

Bit	7	6	5	4	3	2	1	0
						USART0[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – USART0[2:0] USART 0 Signals
This bit field controls the pin positions for USART 0 signals.

Value	Name	Description			
		TxD	RxD	AUX0	AUX1
0x0	DEFAULT	PA0	PA1	PA2	PA3
0x1	ALT1	PA4	PA5	PA6	PA7
0x2	ALT2	PA2	PA3	-	-
0x3	ALT3	PD4	PD5	PD6	PD7
0x4	ALT4	PC1	PC2	PC3	-
0x6	ALT6	PF7	PF6	-	-
0x7	NONE	No pin connection			

17.3.4. SPI 0 Pin Position

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

Bit	7	6	5	4	3	2	1	0
						SPI0[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – SPI0[2:0] SPI 0 Signals
This bit field controls the pin positions for SPI 0 signals.

Value	Name	Description			
		MOSI	MISO	SCK	$\overline{SS}$
0x0	DEFAULT	PA4	PA5	PA6	PA7
0x1	-	Reserved			
0x2	-	Reserved			
0x3	ALT3	PA0	PA1	PC0	PC1
0x4	ALT4	PD4	PD5	PD6	PD7
0x5	ALT5	PC0	PC1	PC2	PC3
0x6	ALT6	PC1	PC2	PC3	PF7
0x7	NONE	No pin connection			Set to 1

17.3.5. TWI 0 Pin Positions

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

Bit	7	6	5	4	3	2	1	0
							TWI0[1:0]	
Access							R/W	R/W
Reset							0	0

Bits 1:0 – TWI0[1:0] TWI 0 Signals
This bit field controls the pin positions for TWI 0 signals.

Value	Name	Description			
		Host/Client		Dual Mode (Client)	
		SDA	SCL	SDA	SCL
0x0	DEFAULT	PA2	PA3	PC2	PC3
0x1	ALT1	PA2	PA3	-	-
0x2	ALT2	PC2	PC3	-	-
0x3	ALT3	PA0	PA1	PC2	PC3

17.3.6. TCE 0 Pin Position

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

Bit	7	6	5	4	3	2	1	0
						TCE0[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – TCE0[2:0] TCE 0 Signals
This bit field controls the pin positions for TCE 0 signals.

Value	Name	Description		
		WO0	WO1	WO2
0x0	PORTA	PA0	PA1	PA2
0x1	—	Reserved		
0x2	PORTC	PC0	PC1	PC2
0x3	PORTD	PD0	PD1	PD2
0x4	—	Reserved		
0x5	PORTF	PF0	PF1	PF2
Other	—	Reserved		

17.3.7. TCBn Pin Position

Name: TCBROUTEA
Offset: 0x08
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
This bit controls the pin position for the TCB1 output.

Value	Name	Description
0	DEFAULT	WO on PA3
1	ALT1	WO on PF5

Bit 0 – TCB0 TCB0 Output
This bit controls the pin position for the TCB0 output.

Value	Name	Description
0	DEFAULT	WO on PA2
1	ALT1	WO on PF4

17.3.8. AC 0 Pin Position

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

Bit	7	6	5	4	3	2	1	0
								AC0
Access								R/W
Reset								0

Bit 0 – AC0 AC 0 Output
This bit controls the pin position for the AC0 output.

Value	Name	Description
0	Default	OUT on PA7
1	Reserved	

18. PORT - I/O Pin Configuration

18.1. Features

- General Purpose Input and Output Pins with Individual Configuration:
  - Pull-up
  - Inverted I/O
  - Input voltage threshold
- Interrupts and Events:
  - Sense both edges
  - Sense rising edges
  - Sense falling edges
  - Sense low level
- Optional Slew Rate Control per I/O Port
- 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)

18.2. Overview

The device's I/O pins 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 define the output state. For example, DIR[3] and OUT[3] of the PORTA instance controls pin PA3.

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 pin value can be read whether the pin is configured as input or output.

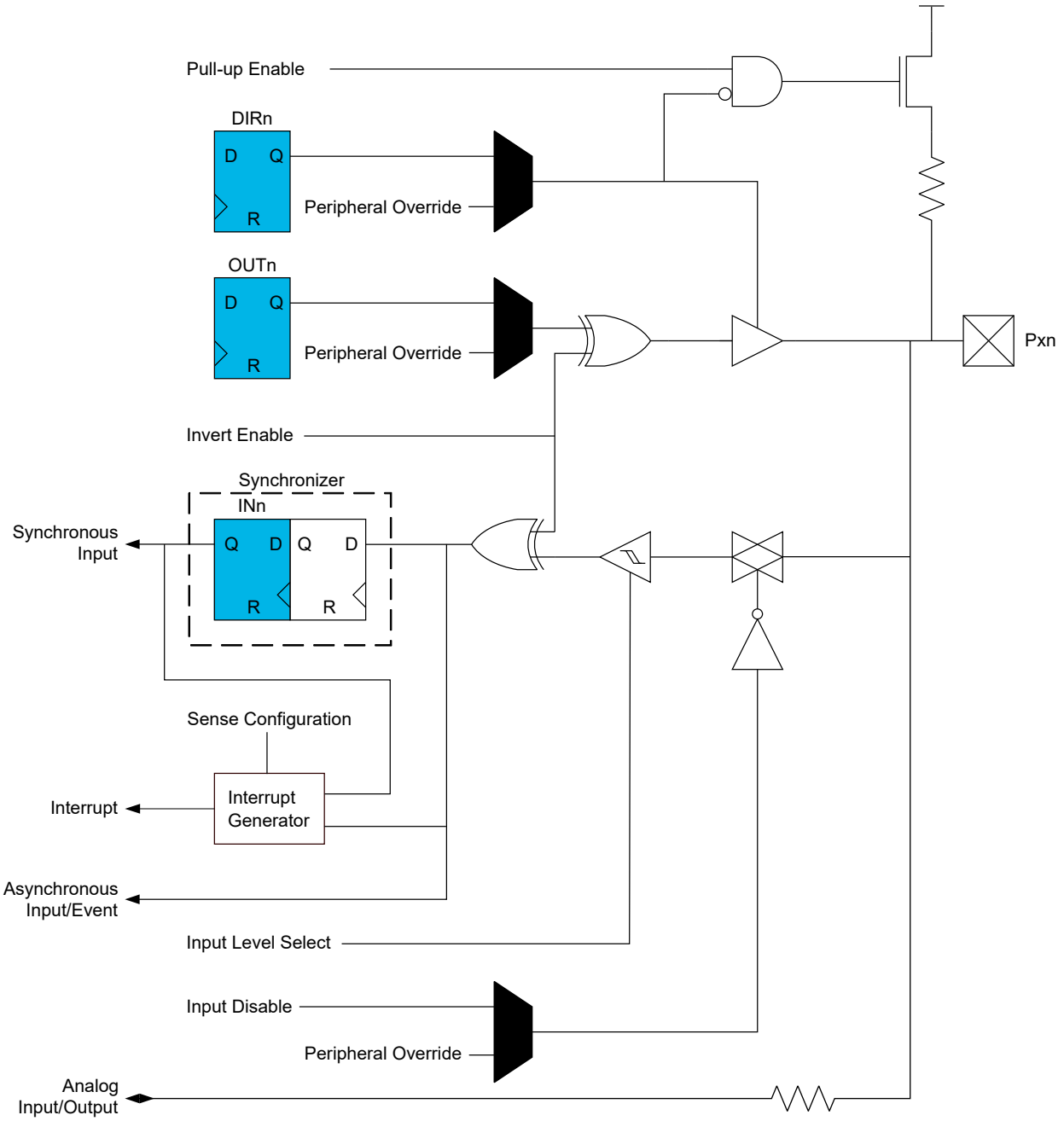
The PORT also supports asynchronous input sensing with interrupts and events for selectable pin change conditions. Asynchronous pin change sensing means that a pin change can trigger an interrupt and wake the device from sleep, including sleep modes where CLK_PER is stopped.

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

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

18.2.1. Block Diagram

Figure 18-1. PORT Block Diagram



18.2.2. Signal Description

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

18.3. Functional Description

18.3.1. 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 the 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: To achieve the lowest possible power consumption, disable the digital input buffer of unused pins and pins used as analog inputs or outputs. For pins with the digital input buffer enabled, it is recommended to transition between high and low voltage thresholds as quickly as possible.

Pins designated for special functions, such as debugger connections, must be configured according to the requirements of their intended use. For example, for pins that are configured as $\overline{\text{RESET}}$, disabling the input buffer on this pin will reset the device.

18.3.2. Operation

18.3.2.1. 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.

18.3.2.2. Port Configuration

The Port Control (PORTx.PORTCTRL) register controls the slew rate limitation for all the PORTx pins.

The slew rate limitation is enabled by writing a '1' to the Slew Rate Limit Enable (SLR) bit in PORTx.PORTCTRL. Refer to the *Electrical Characteristics* section for further details.

18.3.2.3. 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 $\text{PORTx} + 0\text{x}10 + n$.

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 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 threshold is important in determining the bit n value in the PORTx.IN register and the level at which an interrupt condition occurs if that feature is 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 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 an analog input. While configured to INPUT_DISABLE, bit n in PORTx.IN will not change since the input synchronizer is disabled.

18.3.2.4. Multi-Pin Configuration

The multi-pin configuration function can configure multiple port pins in one operation. The desired pin configuration is first written to the PORTx.PINCONFIG register, followed by a register write with the selected pins to modify, allowing changing the configuration (PORTx.PINnCTRL) for up to eight pins in one write.

Tip: The PORTx.PINCONFIG register is mirrored on all ports, allowing the a single setting across multiple ports. The PORTx.PINCTRLUPD/SET/CLR registers are not mirrored, and configurations must be written for each port.

Port pins can be configured and modified by writing to the following registers for the multi-pin configuration.

Table 18-1. Multi-Pin Configuration Registers

Register	Description
PORTx.PINCONFIG	PINnCTRL (ISC, PULLUPEN, and INVEN) setting to prepare simultaneous configuration of multiple PINnCTRL registers
PORTx.PINCTRLUPD	Writing a '1' to bit n in the PINCTRLUPD register will copy the PINCONFIG register content to the PINnCTRL register
PORTx.PINCTRLSET(1)	Writing a '1' to bit n in the PINCTRLSET register will set the individual bits in the PINnCTRL register, according to the bits set to '1' in the PINCONFIG register
PORTx.PINCTRLCLR(2)	Writing a '1' to bit n in the PINCTRLCLR register will clear the individual bits in the PINnCTRL register, according to the bits set to '1' in the PINCONFIG register

Notes:

1. Using PINCTRLSET to configure nonzero ISC bit fields will result in a bitwise OR with the PINCONFIG and PINnCTRL registers and may give an unexpected setting.
2. Using PINCTRLCLR to configure nonzero ISC bit fields will result in a bitwise inverse AND with the PINCONFIG and PINnCTRL registers and may give an unexpected setting.

The following code snippet demonstrates how to configure multiple PINnCTRL registers of several ports. Note that, because the PINCONFIG register is mirrored across all the ports, it is enough to write it only once, for PORT A, in this example.

```
PORTA.PINCONFIG = PORT_ISC_INPUT_DISABLE_gc; /* The setting to load to the PINnCTRL registers
*/
PORTA.PINCTRLUPD = 0xff;
PORTB.PINCTRLUPD = 0xff;
PORTC.PINCTRLUPD = 0xff;
PORTD.PINCTRLUPD = 0xff;
PORTE.PINCTRLUPD = 0xff;
```

18.3.2.5. 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 allowing 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 18-2. 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

Note: Avoid accessing the mapped VPORT register using the single-cycle I/O instructions immediately after accessing the regular PORT register. This may cause a memory collision since the single-cycle I/O access to VPORT is faster than the regular PORT register access.

18.3.2.6. 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.

18.3.3. Interrupts

Table 18-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
PORTx_PORT	INTn	The pin n voltage matches the configuration	The Input/Sense Configuration (ISC) bit field in the Pin n Control (PORTx.PINnCTRL) 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.

Consider the following points when setting or changing interrupt settings:

- 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 disabling an input by writing to ISC while synchronizing an interrupt, that specific interrupt may be requested on re-enabling the input, even when 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

18.3.3.1. Asynchronous Sensing Pin Properties

All PORT pins support fully 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. The pulse width needed to trigger an interrupt is less than one CLK_PER cycle.

18.3.4. Events

PORT can generate the following events:

Table 18-4. Event Generators in PORTx

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
PORTx	EVGEN0	Pin input level	Level	Asynchronous	Given by pin level
PORTx	EVGEN1	Pin input level	Level	Asynchronous	Given by pin level

All PORT pins can be configured as asynchronous Event System generators. Two event generators are available for each port. The output from PORT to the Event System is the value present on the corresponding pin if the digital input buffer is enabled. If a pin input buffer is disabled, the corresponding output to the Event System is zero.

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

18.3.5. 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.

18.3.6. Debug Operation

The PORT continues ordinary operation when halting the CPU in Debug mode. If configuring the PORT in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may occur during debugging.

18.4. 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	PORTCTRL	7:0								SRL
0x0B	PINCONFIG	7:0	INVEN				PULLUPEN	ISC[2:0]		
0x0C	PINCTRLUPD	7:0	PINCTRLUPD[7:0]							
0x0D	PINCTRLSET	7:0	PINCTRLSET[7:0]							
0x0E	PINCTRLCLR	7:0	PINCTRLCLR[7:0]							
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]		
0x18	EVGENCTRLA	7:0		EVGEN1SEL[2:0]				EVGEN0SEL[2:0]		

18.5. Register Description - PORTx

18.5.1. 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 table below shows the available configuration for each bit n in this bit field.

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

18.5.2. 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.

18.5.3. 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.

18.5.4. 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.

18.5.5. 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 affects the output when the output driver (PORTx.DIR) is enabled for the corresponding pin.
The table below shows the available configuration for each bit n in this bit field.

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

18.5.6. 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.

18.5.7. 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.

18.5.8. 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.

18.5.9. 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 table below shows the available states of each bit n in this bit field.

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

18.5.10. 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.

18.5.11. Port Control

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

This register contains the slew rate limit enable bit for this port.

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

Bit 0 – SRL Slew Rate Limit Enable
This bit controls the slew rate limitation for all pins in PORTx.

Value	Description
0	Slew rate limitation is disabled for all pins in PORTx
1	Slew rate limitation is enabled for all pins in PORTx

18.5.12. Multi-Pin Configuration

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

For faster configuration of the port module, the multi-pin configuration write enables the configuration of several port pins in a single cycle. Especially with large pin count devices, this function can significantly speed up PORT pin configuration operations.

Writing to this register may be followed by a write to either of the Multi-Pin Control (PORTx.PINCTRLUPD/SET/CLR) registers to update the Pin n Control (PORTx.PINnCTRL) registers for PORTx.

This register is mirrored across all PORTx modules.

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 the pin conditions that will trigger a port interrupt.

Value	Name	Description
0x0	INTDISABLE	Interrupt disabled but digital 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(2)
other	—	Reserved

Notes:

1. If the digital input buffer for pin n is disabled, bit n in the Input Value (PORTx.IN) register will not be updated.
2. The LEVEL interrupt will continue to trigger as long as the pin remains in a low state.

18.5.13. Multi-Pin Control Update Mask

Name: PINCTRLUPD
Offset: 0x0C
Reset: 0x00
Property: -

For faster configuration of the port module, the multi-pin configuration write enables the configuration of several port pins in a single cycle. Especially with large pin count devices, this function can significantly speed up PORT pin configuration operations.

Bit	7	6	5	4	3	2	1	0
	PINCTRLUPD[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 – PINCTRLUPD[7:0] Multi-Pin Control Update Mask
This bit field controls the copy of the Multi-Pin Configuration (PORTx.PINCONFIG) register content to the individual Pin n Control (PORTx.PINnCTRL) registers without using an individual write operation for each register.
Writing a '0' to bit n in this bit field has no effect.
Writing a '1' to bit n in this bit field will copy the PORTx.PINCONFIG register content to the corresponding PORTx.PINnCTRL register.
Reading this bit field will always return zero.

18.5.14. Multi-Pin Control Set Mask

Name: PINCTRLSET
Offset: 0x0D
Reset: 0x00
Property: -

For faster configuration of the port module, the multi-pin configuration write enables the configuration of several port pins in a single cycle. Especially with large pin count devices, this function can significantly speed up PORT pin configuration operations.

Bit	7	6	5	4	3	2	1	0
	PINCTRLSET[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 – PINCTRLSET[7:0] Multi-Pin Control Set Mask
This bit field controls the setting of bits in the individual Pin n Control (PORTx.PINnCTRL) registers without using an individual read-modify-write operation for each register.
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 individual bits in the PORTx.PINnCTRL register, according to the bits set to '1' in the Multi-Pin Configuration (PORTx.PINCONFIG) register.
Reading this bit field will always return zero.

18.5.15. Multi-Pin Control Clear Mask

Name: PINCTRLCLR
Offset: 0x0E
Reset: 0x00
Property: -

For faster configuration of the port module, the multi-pin configuration write enables the configuration of several port pins in a single cycle. Especially with large pin count devices, this function can significantly speed up PORT pin configuration operations.

Bit	7	6	5	4	3	2	1	0
	PINCTRLCLR[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 – PINCTRLCLR[7:0] Multi-Pin Control Clear Mask
This bit field controls the clearing of bits in the individual Pin n Control (PORTx.PINnCTRL) registers without using an individual read-modify-write operation for each register.
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 individual bits in the PORTx.PINnCTRL register, according to the bits set to '1' in the Multi-Pin Configuration (PORTx.PINCONFIG) register.
Reading this bit field will always return zero.

18.5.16. 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 the pin conditions that will trigger a port interrupt.

Value	Name	Description
0x0	INTDISABLE	Interrupt disabled but digital 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(2)
other	—	Reserved

Notes:

1. If the digital input buffer for pin n is disabled, bit n in the Input Value (PORTx.IN) register will not be updated.
2. The LEVEL interrupt will continue to trigger as long as the pin remains in a low state.

18.5.17. Event Generator Control A

Name: EVGENCTRLA
Offset: 0x18
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
		EVGEN1SEL[2:0]				EVGEN0SEL[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 0:2, 4:6 – EVGENnSEL Event Generator n Select
This bit field controls which pin is connected to Event Generator n.

Value	Name	Description
0x0	PIN0	Pin 0 used as event generator
0x1	PIN1	Pin 1 used as event generator
0x2	PIN2	Pin 2 used as event generator
0x3	PIN3	Pin 3 used as event generator
0x4	PIN4	Pin 4 used as event generator
0x5	PIN5	Pin 5 used as event generator
0x6	PIN6	Pin 6 used as event generator
0x7	PIN7	Pin 7 used as event generator

18.6. 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]							

18.7. Register Description - VPORTx

18.7.1. 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 allowing 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 table below shows the available configuration for each bit n in this bit field.

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

18.7.2. 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 allowing 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 affects the output when the output driver (PORTx.DIR) is enabled for the corresponding pin.
The table below shows the available configuration for each bit n in this bit field.

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

18.7.3. 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 allowing 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 table below shows the available states of each bit n in this bit field.

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

18.7.4. 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 allowing 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.

19. BOD - Brown-out Detector

19.1. 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

19.2. Overview

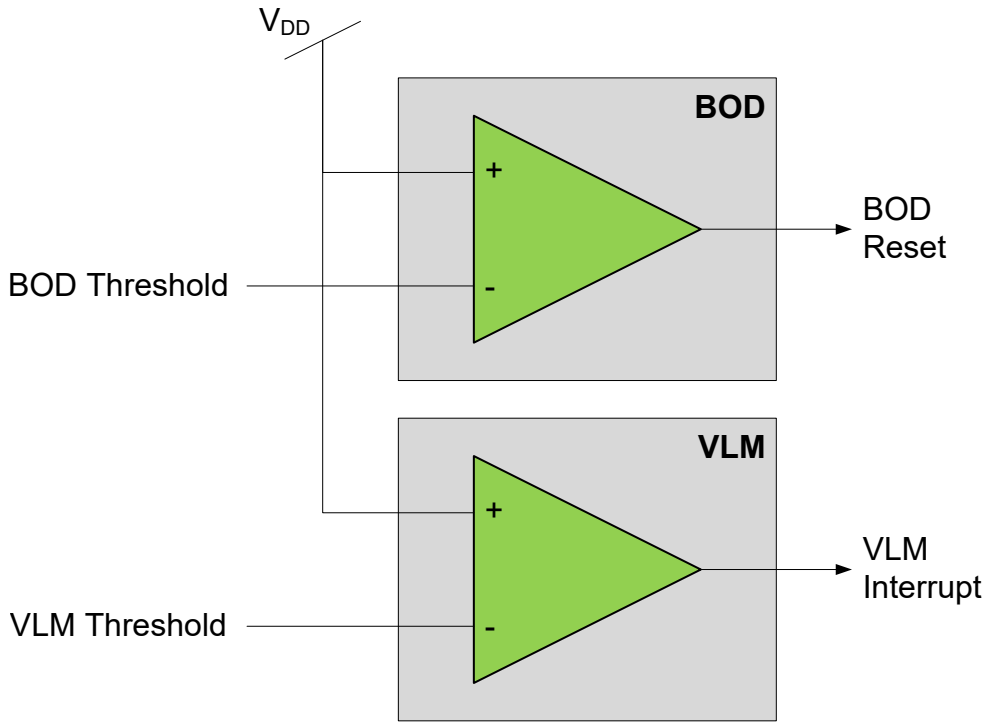
The Brown-out Detector (BOD) monitors the power supply and compares the supply voltage with the programmable brown-out threshold level. The brown-out threshold level defines when to generate a System Reset. The Voltage Level Monitor (VLM) monitors the power supply and compares it to a threshold higher 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.

19.2.1. Block Diagram

Figure 19-1. BOD Block Diagram



19.3. Functional Description

19.3.1. Initialization

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

Write a '1' to the VLM Interrupt Enable (VLMIE) bit in the Interrupt Control (BOD.INTCTRL) register to enable the Voltage Level Monitor function. 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 from either above or below the threshold.

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 the Sampled mode, the VLM will also be sampled. The VLM threshold is defined by writing the VLM Level (VLMLVL) bits in the VLM Control (BOD.VLMCTRLA) register.

19.3.2. Interrupts

Table 19-1. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
BOD_VLM	VLM	The supply voltage crosses the voltage level threshold	The Voltage Level Monitor 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.

19.3.3. 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.

19.3.4. 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 19-2. Registers Under Configuration Change Protection

Register	Key
The SLEEP bit is in the BOD.CTRLA register	IOREG

19.4. 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

19.5. Register Description

19.5.1. Control A

Name: CTRLA
Offset: 0x00
Reset: 0x00
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				0	0	0	0	0

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	Name	Description
0x0	128Hz	The sampling frequency is 128 Hz
0x1	32Hz	The sampling frequency is 32 Hz

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

Value	Name	Description
0x0	DISABLE	BOD disabled
0x1	ENABLED	BOD enabled in Continuous mode
0x2	SAMPLE	BOD enabled in Sampled mode
0x3	ENABLEWAIT	BOD enabled in Continuous mode. Execution is halted at wake-up until BOD is running.

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

Value	Name	Description
0x0	DISABLE	BOD disabled
0x1	ENABLED	BOD enabled in Continuous mode
0x2	SAMPLE	BOD enabled in Sampled mode
0x3	-	Reserved

19.5.2. Control B

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

Bit	7	6	5	4	3	2	1	0
						LVL[2:0]		
Access						R	R	R
Reset						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).
Note: Values in the Description column are typical values. Refer to the *Electrical Characteristics* section for further details

Value	Name	Description
0x0	BODLEVEL0	1.65V
0x1	BODLEVEL1	1.90V
0x2	BODLEVEL2	2.60V
0x3	BODLEVEL3	4.30V
Other	-	Reserved

19.5.3. VLM Control

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	Name	Description
0x00	OFF	VLM disabled
0x01	5ABOVE	VLM threshold 5% above the BOD threshold
0x02	15ABOVE	VLM threshold 15% above the BOD threshold
0x03	25ABOVE	VLM threshold 25% above the BOD threshold

19.5.4. 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	FALLING	V_{DD} falls below VLM threshold
0x1	RISING	V_{DD} rises above VLM threshold
0x2	BOTH	V_{DD} crosses VLM threshold
Other	-	Reserved

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

19.5.5. 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 cleared by writing a '1' to it.
This flag is set when a trigger from the VLM is given. The trigger condition is configured by writing to the VLM Configuration (VLMCFG) bit field in the Interrupt Control (INTCTRL) register. The flag is only update when the BOD is enabled.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the VLM Interrupt flag.

19.5.6. 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	Name	Description
0	ABOVE	The voltage is above the VLM threshold level
1	BELOW	The voltage is below the VLM threshold level

20. VREF - Voltage Reference

20.1. Features

- Programmable Voltage Reference Sources:
  - Reference for AC0
- Each Reference Source Supports the Following Voltages:
  - 1.024V
  - 2.048V
  - 4.096V
  - 2.500V
  - V_{DD}
  - EXTVREF0

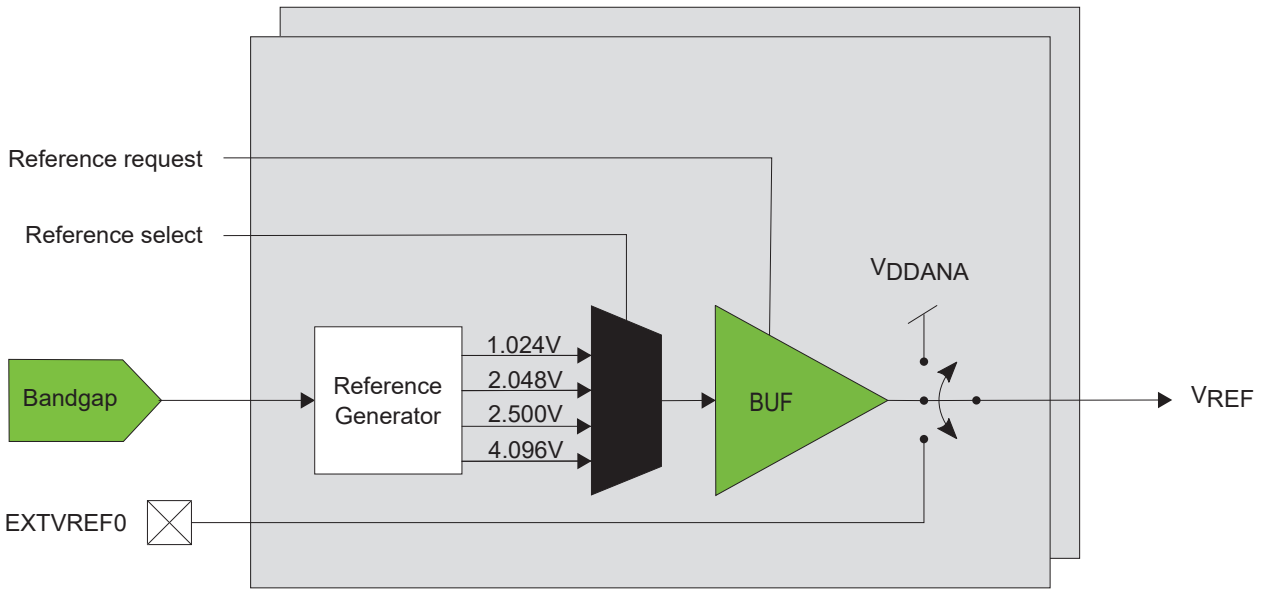
20.2. Overview

The Voltage Reference (VREF) peripheral provides control register for the voltage reference sources used by the AC peripheral. The user can select the reference voltages for AC0 by writing to the appropriate registers in the VREF peripheral.

In the default configuration for VREF, a voltage reference source is automatically enabled 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 ALWAYSON bit in the AC Reference Control (VREF.AC0REF) register. This decreases the start-up time but comes at the cost of increased power consumption.

20.2.1. Block Diagram

Figure 20-1. VREF Block Diagram



20.2.2. Peripherals Using Voltage References

Devices in the AVR LA family have a number of peripherals that can use a voltage reference.

- The **AC** peripheral uses input pins, but the negative input can be routed to use the voltage reference controlled by VREF.AC0REF. In addition, the AC peripheral has its own independent voltage divider for the incoming reference voltage, controlled by the AC0.REFSCALE register.
- The **ADC** peripheral provides its own control register for the voltage reference and is not selected by VREF. The ADC voltage reference is selected by writing to the Reference Selection (REFSCALE) bit fields in the ADC0.CTRLC register.
  - The ADC can also use the AC reference voltage divider (AC0.REFSCALE) as a positive input

20.3. Functional Description

20.3.1. Initialization

The default VREF configuration enables the respective source when AC0 request a voltage reference. The default reference voltage is 1.024V. A different reference voltage can be selected by writing to the Reference Select (REFSEL) bit field in the AC Reference Control (VREF.ACREF) registers.

20.4. Register Summary - VREF

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00 ... 0x03	Reserved									
0x04	AC0REF	7:0	ALWAYSON					REFSEL[2:0]		

20.5. Register Description

20.5.1. AC0 Reference Control

Name: AC0REF
Offset: 0x04
Reset: 0x00
Property: -

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

Bit 7 – ALWAYSON Keep Reference Always On
This bit controls whether the AC0 reference is always on.

Value	Description
0	The reference is automatically enabled when needed
1	The reference is always on

Bits 2:0 – REFSEL[2:0] Reference Select
This bit field controls the reference voltage level for AC0.

Value	Name	Description
0x0	1V024	Internal 1.024V reference(1)
0x1	2V048	Internal 2.048V reference(1)
0x02	4V096	Internal 4.096V reference(1)
0x03	2V500	Internal 2.500V reference(1)
0x04	—	Reserved
0x5	VDD	V_{DD} as reference
0x6	EXTVREF0	Reference from External Reference Pin 0
0x07	—	Reserved

Note:

1. The values given for internal references are only typical. Refer to the *Electrical Characteristics* section for further details.

21. WDT - Watchdog Timer

21.1. Features

- Issues a System Reset if the Watchdog Timer Is Not Cleared Before Its Time-Out Period
- Operates Asynchronously from the Peripheral Clock Using an Independent Oscillator
- Uses the 1.024 kHz Output of the 32.768 kHz Ultra-Low Power Oscillator (OSC32K)
- 11 Selectable Time-Out Periods, from 8 ms to 8s
- Two Operation Modes:
  - Normal mode
  - Window mode
- Configuration Lock to Prevent Unwanted Changes

21.2. Overview

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

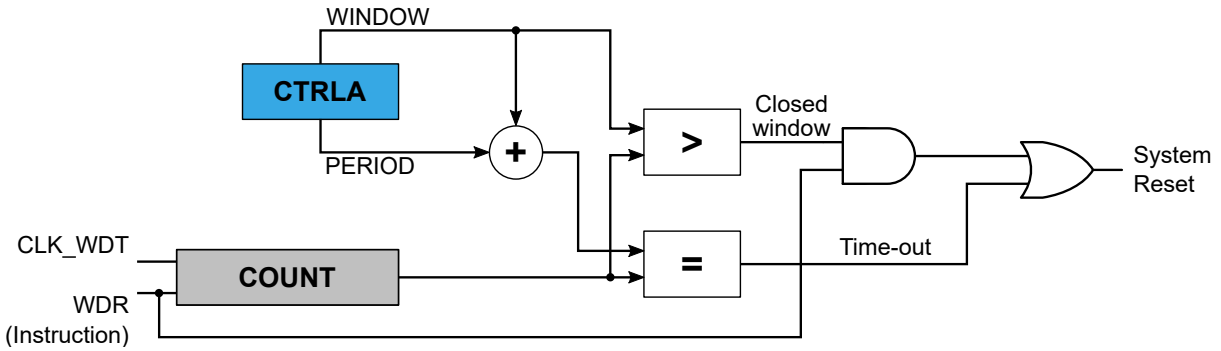
In addition to the Normal mode as described above, the WDT has a Window mode. The Window mode defines a time slot or "window" inside the time-out period during which the WDT must be reset. If the WDT is reset outside this window, either too early or too late, a system Reset will be issued. Compared to 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. Since it is asynchronous (running from a CPU-independent clock source), it will continue to operate and be able to issue a system Reset, even if the main clock fails.

The WDT has a Configuration Change Protection (CCP) mechanism and a lock functionality, ensuring the WDT settings cannot be changed by accident.

21.2.1. Block Diagram

Figure 21-1. WDT Block Diagram



21.3. Functional Description

21.3.1. Initialization

1. The WDT is enabled when a non-zero value is written to the Period (PERIOD) bit field in the Control A (WDT.CTRLA) register.

2. Optional: Write a non-zero value to the Window (WINDOW) bit field 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 (CCP) mechanism.

A fuse (FUSE.WDTCFG) defines the Reset value of the WDT.CTRLA register. If the value of the PERIOD bit field in the FUSE.WDTCFG fuse is different than zero, the WDT is enabled, and the LOCK bit in the WDT.STATUS register is set at boot time.

21.3.2. Clocks

A 1.024 kHz clock (CLK_WDT) is sourced from the internal Ultra-Low Power Oscillator, OSC32K. Due to the ultra-low power design, the oscillator is less accurate than other oscillators featured in the device, and hence, the exact time-out period may vary from device to device. This variation must be considered when designing software that uses the WDT to ensure that the time-out periods used are valid for all devices. Refer to the *Electrical Characteristics* section for more specific information.

The WDT clock (CLK_WDT) is asynchronous to the peripheral clock. Due to this asynchronicity, writing to the WDT Control A (WDT.CTRLA) register will require synchronization between the clock domains. Refer to Synchronization for further details.

21.3.3. Operation

21.3.3.1. Normal Mode

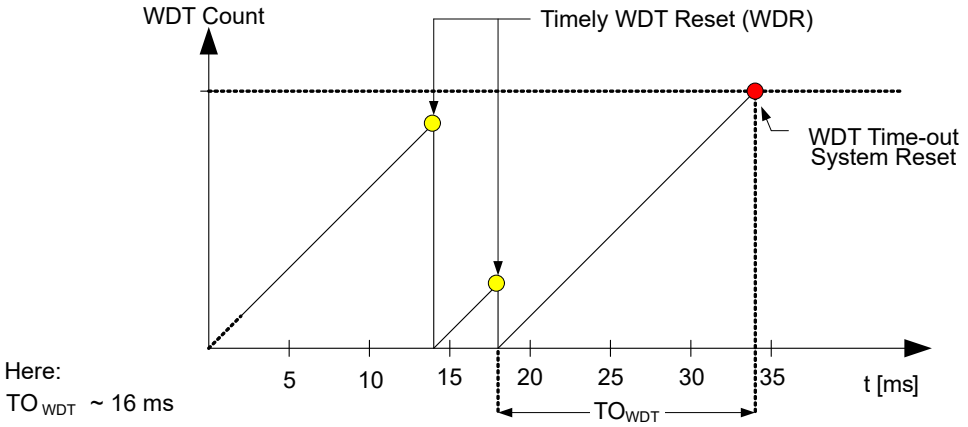
In the Normal mode operation, a single time-out period is set for the WDT. If the WDT is not reset from software using the `WDR` instruction during the defined time-out period, the WDT will issue a system Reset.

Each time the WDT is reset by software using the `WDR` instruction, a new WDT time-out period starts.

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.

The figure below shows a typical timing scheme for the WDT operating in Normal mode.

Figure 21-2. Normal Mode Operation



The Normal mode is enabled as long as the Window (WINDOW) bit field in the WDT.CTRLA register is '`0x0`'.

21.3.3.2. Window Mode

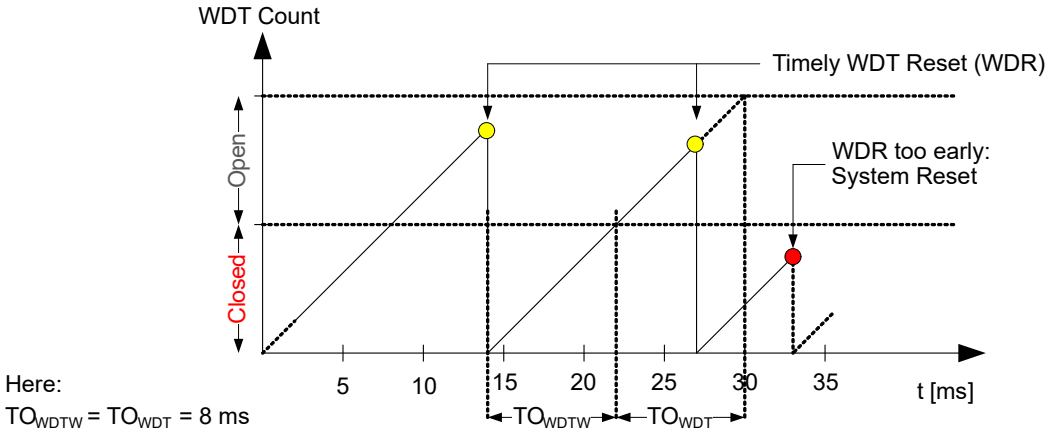
In Window mode operation, the WDT uses two different time-out periods: A closed window time-out period (TO_{WDTW}) and an open window time-out period (TO_{WDT}):

- TO_{WDTW} defines a duration from 8 ms to 8s, where the WDT should not be reset. If the WDT is reset during this period, the WDT will issue a system Reset.
- TO_{WDT}, 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 the Window mode or going out of the Debug mode, the window is activated after the first `WDR` instruction.

The figure below shows a typical timing scheme for the WDT operating in Window mode.

Figure 21-3. 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`.

21.3.3.3. Preventing Unintentional Changes

The WDT provides two security mechanisms to avoid unintentional changes to the WDT settings:

- The CCP mechanism, employing a timed write procedure for changing the WDT control registers. Refer to Configuration Change Protection for further details.
- Locking the configuration by writing a '1' to the Lock (LOCK) bit in the Status (WDT.STATUS) register. When this bit is '1', the Control A (WDT.CTRLA) register cannot be changed. The LOCK bit can only be written to '1' in software, while the device needs to be in Debug mode to be able to write it to '0'. Consequently, the WDT cannot be disabled from the software.

Note: The WDT configuration is loaded from fuses after Reset. If the PERIOD bit field is set to a non-zero value, the LOCK bit is automatically set in WDT.STATUS.

21.3.4. Sleep Mode Operation

The WDT will continue to operate in any sleep mode where the source clock is active.

21.3.5. 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 and when 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.

21.3.6. Synchronization

The Control A (WDT.CTRLA) register is synchronized when written, due to the asynchronicity between the WDT clock domain and the peripheral clock domain. 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 bit fields are synchronized when written:

- The Period (PERIOD) bit field in Control A (WDT.CTRLA) register
- The Window (WINDOW) bit field in Control A (WDT.CTRLA) register

The `WDR` instruction will need two to three cycles of the WDT clock to be synchronized.

21.3.7. 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 21-1. WDT - Registers Under Configuration Change Protection

Register	Key
WDT.CTRLA	IOREG
LOCK bit in WDT.STATUS	IOREG

21.4. Register Summary

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

21.5. Register Description

21.5.1. 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 the LOCK bit in WDT.STATUS is '1', all bits are change-protected (Access = R)
- If the LOCK bit in WDT.STATUS is '0', all bits can be changed (Access = R/W)

Value	Name	Description
0x0	OFF	-
0x1	8CLK	7.8125 ms
0x2	16CLK	15.625 ms
0x3	32CLK	31.25 ms
0x4	64CLK	62.5 ms
0x5	128CLK	0.125s
0x6	256CLK	0.250s
0x7	512CLK	0.500s
0x8	1KCLK	1.0s
0x9	2KCLK	2.0s
0xA	4KCLK	4.0s
0xB	8KCLK	8.0s
Other	-	Reserved

Note: Refer to the *Electrical Characteristics* section for specific information regarding the 32.768 kHz Ultra-Low Power Oscillator (OSC32K) accuracy.

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 the Normal mode accordingly. In the Window mode, these bits select the duration of the open window.
The bits are optionally lock-protected:

- If the LOCK bit in WDT.STATUS is '1', all bits are change-protected (Access = R)
- If the LOCK bit in WDT.STATUS is '0', all bits can be changed (Access = R/W)

Value	Name	Description
0x0	OFF	-
0x1	8CLK	7.8125 ms
0x2	16CLK	15.625 ms
0x3	32CLK	31.25 ms
0x4	64CLK	62.5 ms
0x5	128CLK	0.125s
0x6	256CLK	0.250s
0x7	512CLK	0.500s

Value	Name	Description
0x8	1KCLK	1.0s
0x9	2KCLK	2.0s
0xA	4KCLK	4.0s
0xB	8KCLK	8.0s
Other	-	Reserved

Note: Refer to the *Electrical Characteristics* section for specific information regarding the 32.768 kHz Ultra-Low Power Oscillator (OSC32K) accuracy.

21.5.2. 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 value in the WDTCFG fuse is different from zero, 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 peripheral clock domain to the WDT clock domain.
This bit is cleared after finishing the synchronization.
This bit is not under CCP.

22. TCB - 16-Bit Timer/Counter Type B

22.1. Features

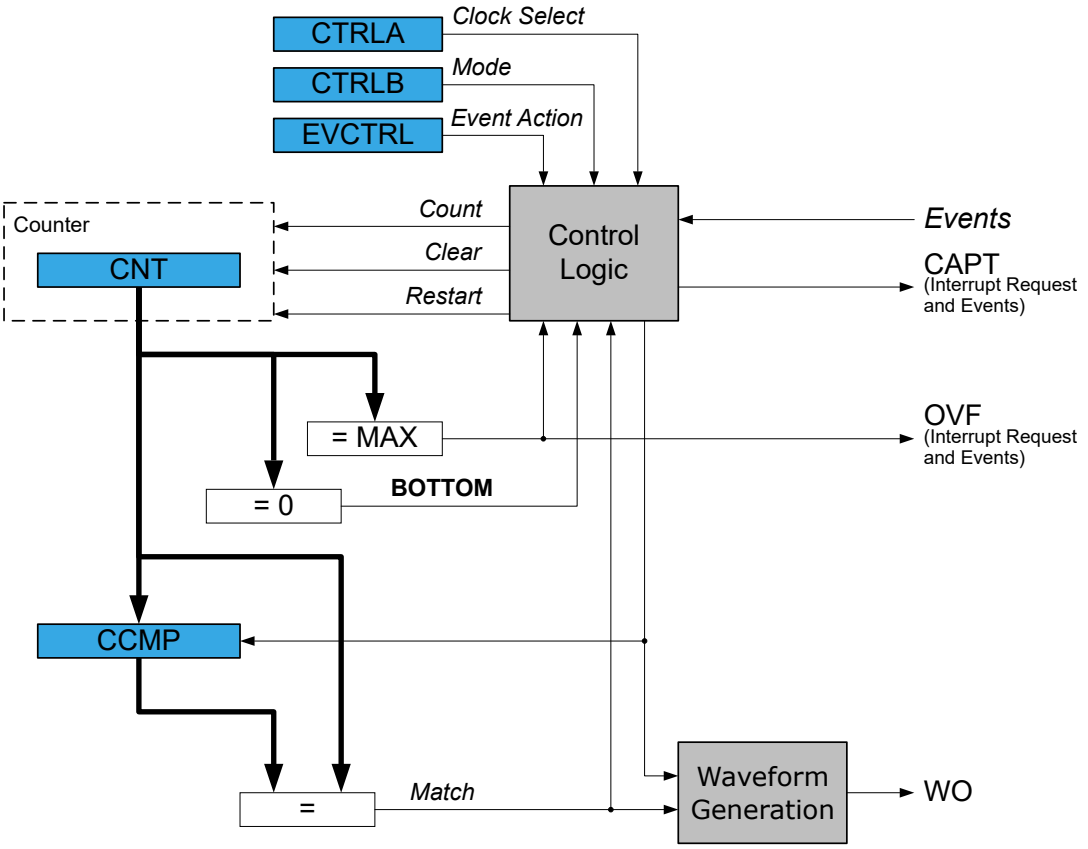
- 16-bit Counter Operation Modes:
  - Periodic interrupt
  - Time-out check
  - Input capture
    - On event
    - Frequency measurement
    - Pulse-width measurement
    - Frequency and pulse-width measurement
    - 32-bit capture
  - Single-Shot
  - 8-bit Pulse-Width Modulation (PWM)
- Noise Canceler on Event Input
- Synchronize Operation with TCEn

22.2. Overview

The 16-bit Timer/Counter type B (TCB) capabilities include frequency and waveform generation and input capture on event with time and frequency measurement of digital signals. The TCB peripheral consists of a base counter and control logic that can be set in one of eight different modes, each providing unique functionality. The base counter is clocked by the peripheral clock with optional prescaling.

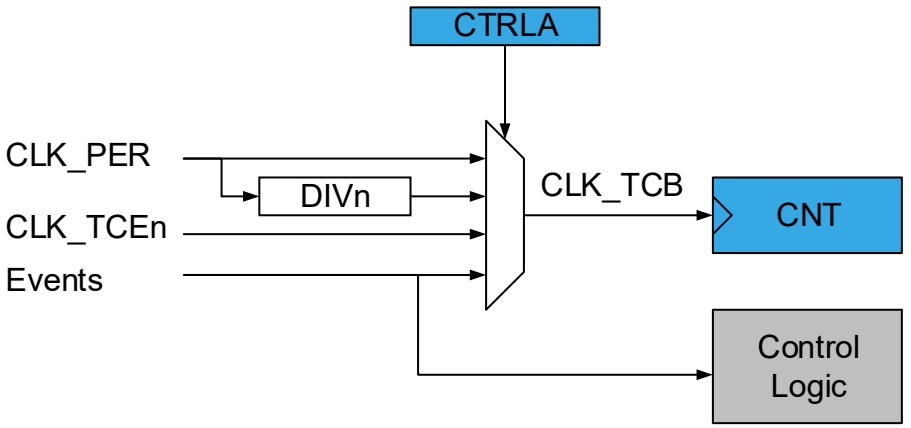
22.2.1. Block Diagram

Figure 22-1. Timer/Counter type B Block Diagram



The TCB peripheral can be clocked from the Peripheral Clock (CLK_PER), from a 16-bit Timer/Counter type E (CLK_TCEn), or the Event System (EVSYS).

Figure 22-2. Timer/Counter type B Clock Logic



The Clock Select (CLKSEL) bit field in the Control A (TCBn.CTRLA) register selects one of the prescaler outputs directly or an event channel as the clock (CLK_TCB) input.

Setting the TCB peripheral to use the clock from a TCEn allows this peripheral to run in sync with that TCEn.

Using the EVSYS, any event source, such as an external clock signal on any I/O pin, may be used as the counter clock input or a control logic input. When using an event action-controlled operation, set the clock selection to use an event channel as the counter input.

22.2.2. Signal Description

Signal	Description	Type
WO	Digital Asynchronous Output	Waveform Output

22.3. Functional Description

22.3.1. Definitions

The following definitions are used throughout the data sheet:

Table 22-1. Timer/Counter Definitions

Name	Description
BOTTOM	The counter reaches BOTTOM when it becomes `0x0000`
MAX	The counter reaches the MAXimum when it becomes `0xFFFF`
TOP	The counter reaches TOP when it becomes equal to the highest value in the count sequence
CNT	Counter (TCBn.CNT) register value
CCMP	Capture/Compare (TCBn.CCMP) 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.

22.3.2. Initialization

By default, the TCB peripheral is in Periodic Interrupt mode. Follow these steps to start using it:

1. Write a TOP value to the Capture/Compare (TCBn.CCMP) register.
2. Optional: Write the Capture/Compare 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 Counter (TCBn.CNT) register. The peripheral will generate a CAPT interrupt and event when the CNT value reaches TOP.
   a. If the Compare/Capture register is modified to a value lower than the current CNT, the peripheral will count to MAX and wrap around.
   b. At MAX, an OVF interrupt and event will be generated.

22.3.3. Operation

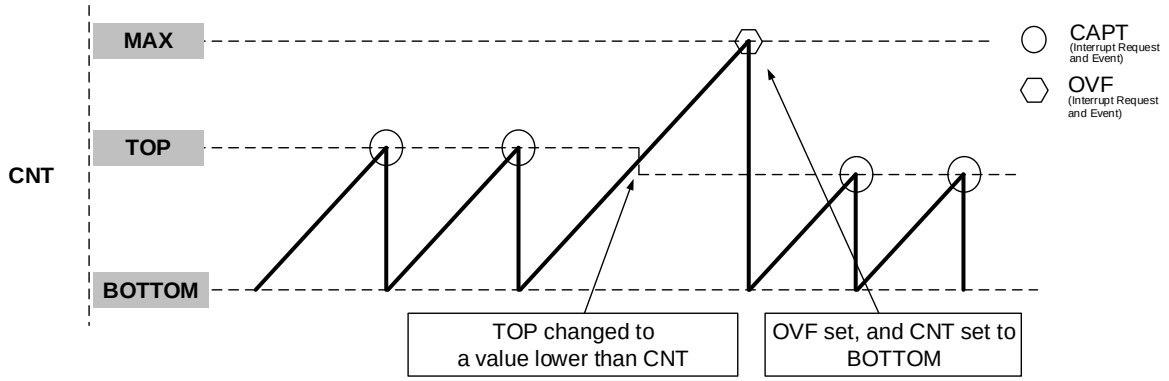
22.3.3.1. Modes

The TCB peripheral can be configured to run in one of the eight different modes described in the sections below. The event pulse must be longer than one system clock cycle to ensure edge detection.

22.3.3.1.1. Periodic Interrupt Mode

In the Periodic Interrupt mode, the counter counts to the capture value and restarts from BOTTOM. A CAPT interrupt and event are generated when the CNT equals to TOP. If TOP is updated to a value lower than CNT when reaching MAX, an OVF interrupt and event are generated, and the counter restarts from BOTTOM.

Figure 22-3. Periodic Interrupt Mode



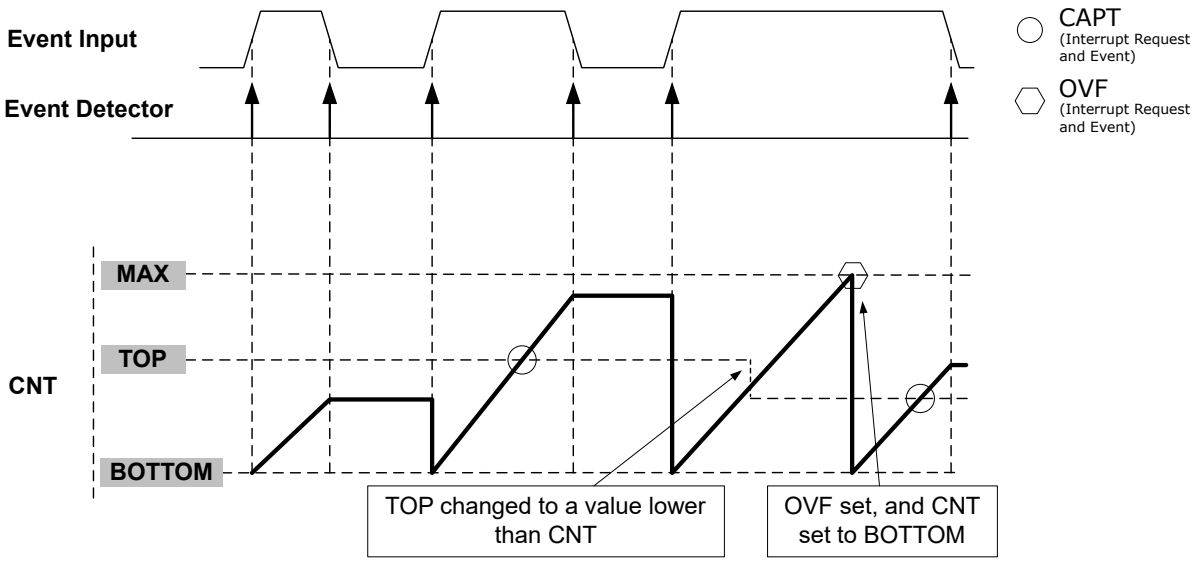
22.3.3.1.2. Time-Out Check Mode

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. CNT remains stationary after the Stop edge (Freeze state). In the Freeze state, the counter restarts on a new Start edge.

This mode requires TCB to be configured as an event user and is explained in the Events section.

The Start or Stop edge is determined by the Event Edge (EDGE) bit in the Event Control (TCBn.EVCTRL) register. If CNT reaches TOP before the second edge, a CAPT interrupt and event will be generated. If TOP is updated to a value lower than the CNT upon reaching MAX, an OVF interrupt and the simultaneous event are generated, and the counter restarts from BOTTOM. In the Freeze state, reading the Counter (TCBn.CNT) register or Compare/Capture (TCBn.CCMP) register or writing the Run (RUN) bit in the Status (TCBn.STATUS) register has no effect.

Figure 22-4. Time-Out Check Mode



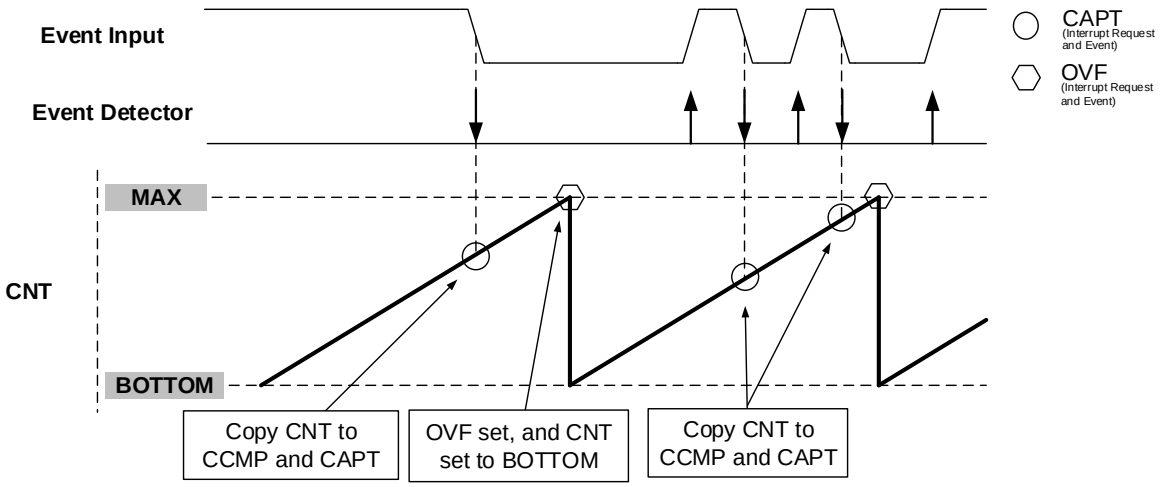
22.3.3.1.3. Input Capture on Event Mode

The counter counts from BOTTOM to MAX in the Input Capture on Event mode. When an event is detected, the Counter (TCBn.CNT) register value is transferred to the Capture/Compare (TCBn.CCMP) register, and a CAPT interrupt and event are generated. The Event edge detector can be configured to trigger a capture on either rising or falling edges.

This mode requires TCB to be configured as an event user and is explained in the Events section.

The figure below shows the input capture unit configured to capture the falling edge of the event input signal. The CAPT interrupt flag is cleared automatically after reading the low byte of the Capture/Compare (TCBn.CCMP) register. An OVF interrupt and event are generated when the CNT is MAX.

Figure 22-5. Input Capture on Event



Important: It is recommended to write `0x0000` to the Counter (TCBn.CNT) register when entering this mode from any other mode.

22.3.3.1.4. Input Capture Frequency Measurement Mode

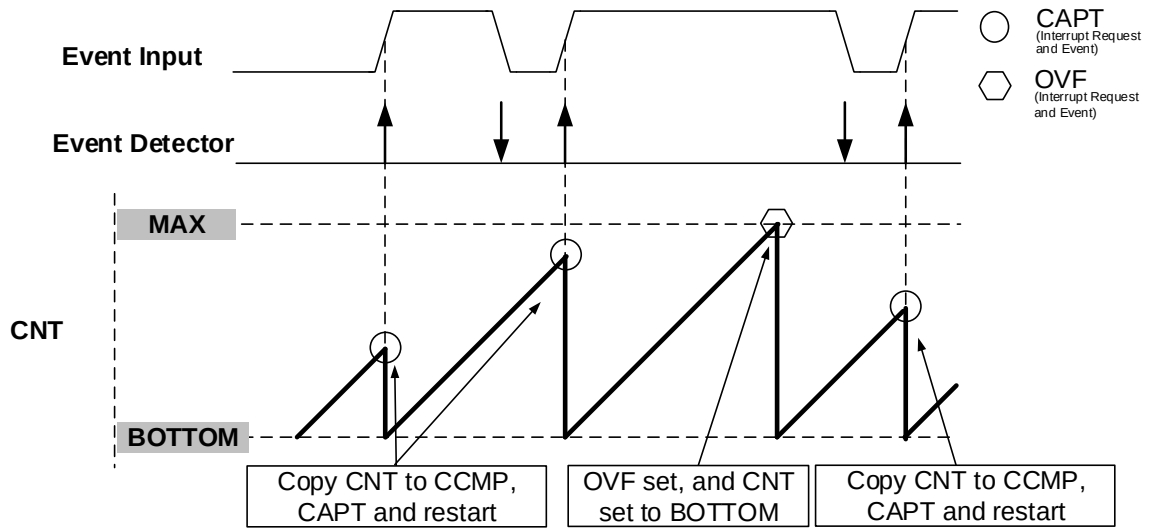
In the Input Capture Frequency Measurement mode, the TCB peripheral captures the counter value and restarts on either a positive or negative edge of the event input signal.

The CAPT interrupt flag is automatically cleared after reading the low byte of the Capture/Compare (TCBn.CCMP) register. When the CNT value is MAX, an OVF interrupt and event are generated.

This mode requires TCB to be configured as an event user and is explained in the Events section.

The figure below illustrates this mode configured to act on a rising edge.

Figure 22-6. Input Capture Frequency Measurement

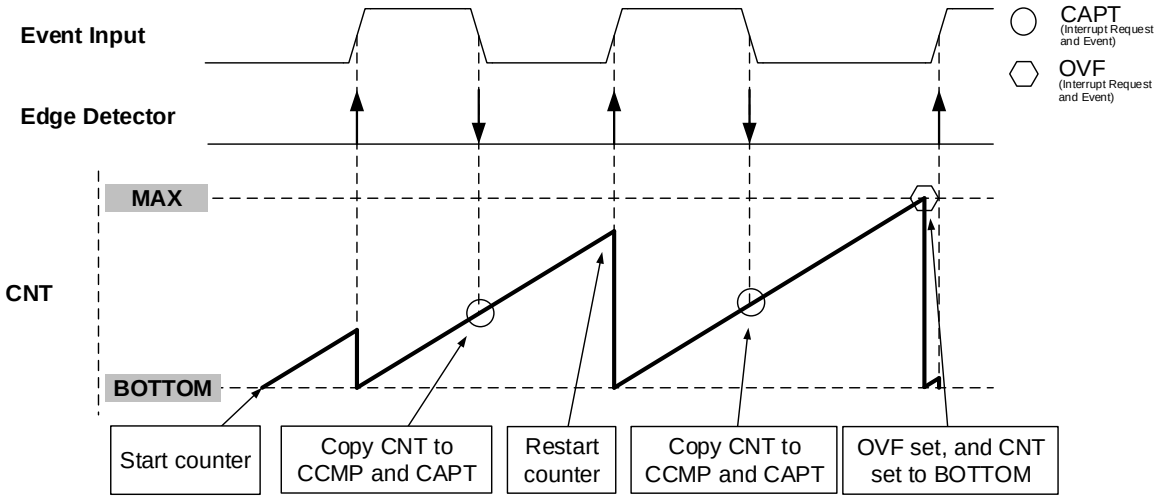


22.3.3.1.5. Input Capture Pulse-Width Measurement Mode

In the Input Capture Pulse-Width Measurement mode, the input capture pulse-width measurement restarts the counter on a positive edge and captures the next falling edge before an interrupt request is generated. The CAPT interrupt flag is cleared automatically after reading the low byte of the Compare/Capture (TCBn.CCMP) register. An OVF interrupt and event are generated when the CNT is MAX. The TCB peripheral will automatically switch between rising and falling edge detection, but a minimum edge separation of two clock cycles is required for correct behavior.

This mode requires TCB to be configured as an event user and is explained in the Events section.

Figure 22-7. Input Capture Pulse-Width Measurement



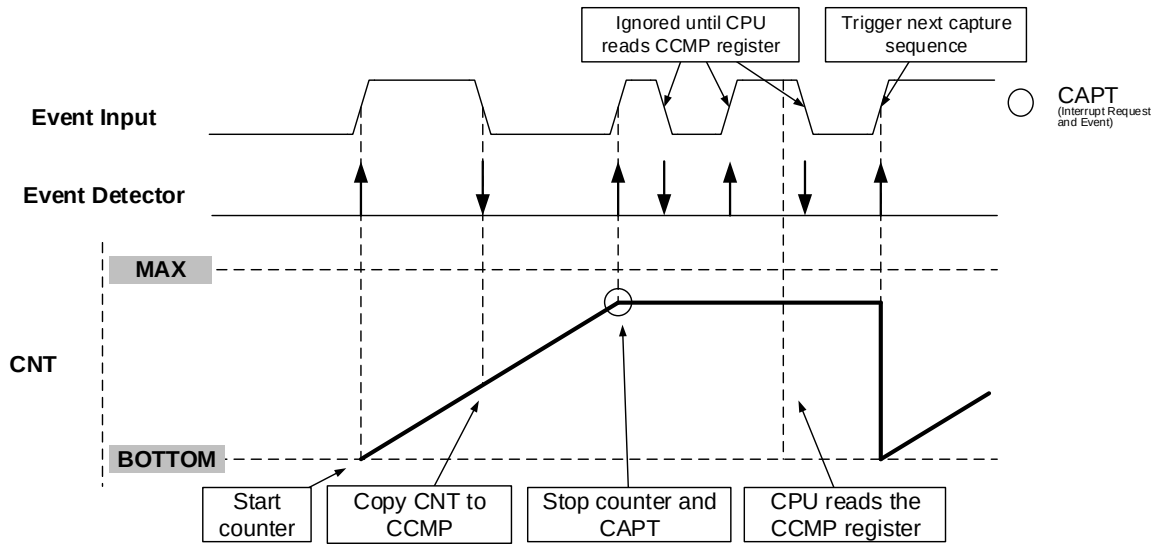
22.3.3.1.6. Input Capture Frequency and Pulse-Width Measurement Mode

In the Input Capture Frequency and Pulse-Width Measurement mode, the TCB peripheral starts 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 and will set the CAPT interrupt flag.

This mode requires TCB to be configured as an event user and is explained in the Events section.

The CAPT interrupt flag is cleared automatically after reading the low byte of the Compare/Capture (TCBn.CCMP) register, and the TCB peripheral is ready for a new capture sequence. Therefore, read the Counter (TCBn.CNT) register before the Capture/Compare (TCBn.CCMP) register since it is reset to BOTTOM at the next positive edge of the event input signal. An OVF interrupt and event are generated when the CNT value is MAX.

Figure 22-8. Input Capture Frequency and Pulse-Width Measurement



22.3.3.1.7. Single-Shot Mode

Use the Single-Shot mode to generate a pulse with a duration defined by the Capture/Compare (TCBn.CCMP) register every time a rising or falling edge is observed on a connected event channel.

This mode requires TCB to be configured as an event user and is explained in the Events section.

When the counter stops, the output pin is set low. If an event is detected on the connected event channel, the TCB peripheral will reset and start counting from BOTTOM to TOP while driving its output high. Read the Run (RUN) bit in the Status (TCBn.STATUS) register to see if the counter is counting. Once the value of CNT reaches the TCBn.CCMP register, the counter ceases counting. Simultaneously, the output pin transitions to a low state for at least one counter-clock cycle (TCB_CLK). During this period, any new event that occurs is disregarded. Following this, there is a two peripheral clock cycles (PER_CLK) delay before the output is set high after receiving a new event.

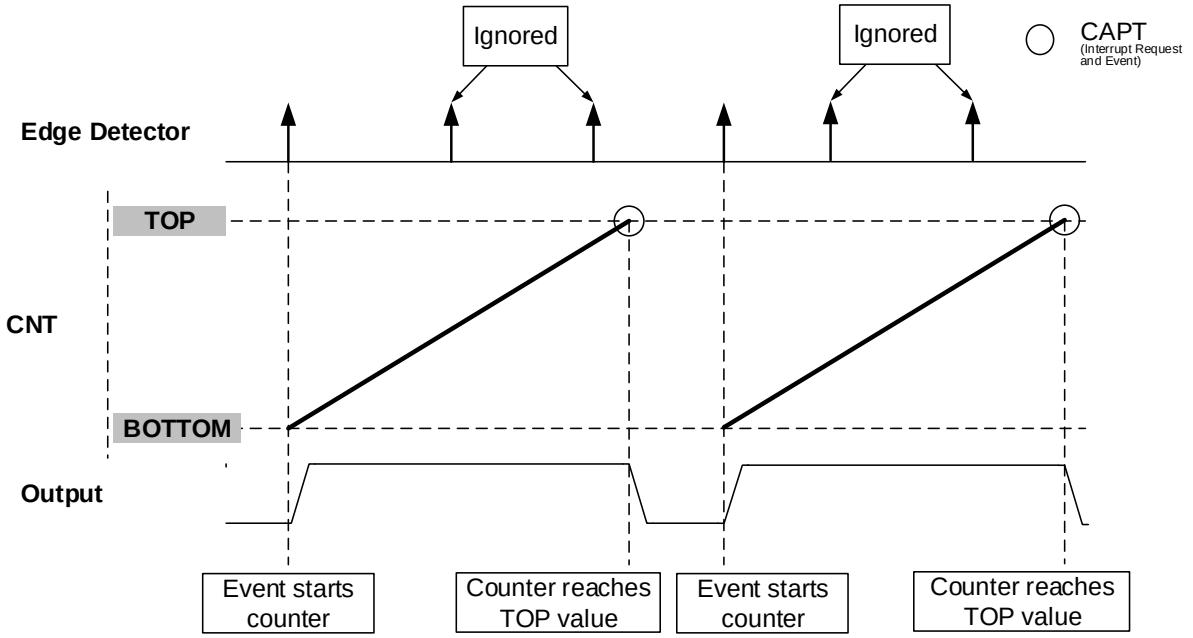
Writing a '1' to the Event Edge (EDGE) bit in the Event Control (TCBn.EVCTRL) register triggers any edge to start the counter.

Writing a '0' to the Event Edge (EDGE) bit in the Event Control (TCBn.EVCTRL) register triggers only positive edges to start the counter.

The counter starts counting as soon as the peripheral is enabled, even without triggering by an event or if the Event Edge (EDGE) bit in the Event Control (TCBn.EVCTRL) register is modified while the peripheral is enabled, which is prevented by writing TOP to the Counter (TCBn.CNT) register. A 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 (TCBn.CNT) register prevents this.

Writing a '1' to the Event Asynchronous (ASYNC) bit in the Control B (TCBn.CTRLB) register, the TCB peripheral reacts 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 complete clock cycles after receiving the event, resulting in an observed delay of two to three clock cycles.

Figure 22-9. Single-Shot Mode

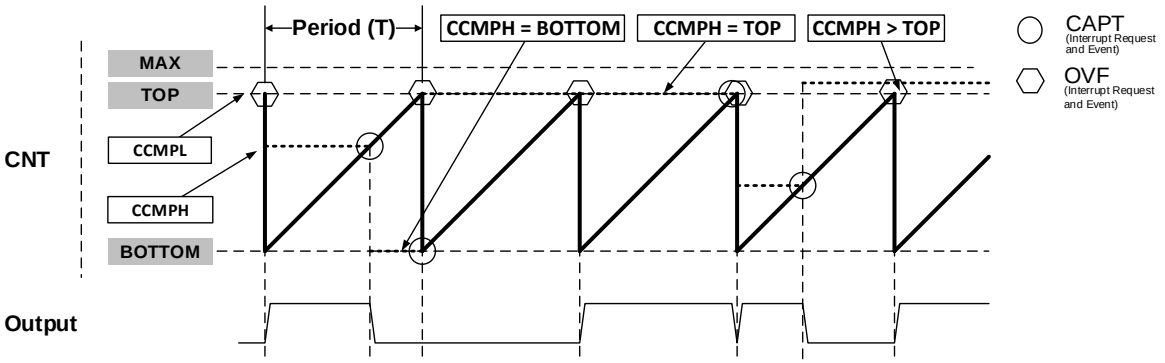


22.3.3.1.8. 8-Bit PWM Mode

The TCB peripheral can be configured to run in 8-bit PWM mode, where each register pair in the 16-bit Capture/Compare (TCBn.CCMPH and TCBn.CCMPL) registers are used as individual Compare registers. CCMPL controls the period (T), while CCMPH controls the waveform duty cycle. 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 output pulse period, the +1 resulting in an observed delay of one clock cycle.

Figure 22-10. 8-Bit PWM Mode



22.3.3.2. Output

The TCB peripheral synchronization and output logic level depend on the selected Timer Mode (CNTMODE) bit field in the Control B (TCBn.CTRLB) register. In the Single-Shot mode, the peripheral can be configured so that the signal generation happens asynchronously to an incoming event

(ASYNC = 1 in the TCBn.CTRLB register). Then, the output signal is set immediately at the incoming event instead of being synchronized to the TCB clock. Due to the synchronization delay for the counter, the waveform output will be set high for three to four CLK_TCB cycles more than what is defined by the TOP value.

Writing a '1' to the Capture/Compare Output Enable (CCMPEN) bit in the Control B (TCBn.CTRLB) register enables and makes the waveform output available on the corresponding pin, overriding the value in the corresponding PORT output register.

The table below lists the different configurations and their impact on the output.

Table 22-2. Output Configuration

CCMPEN	CNTMODE	ASYNC	Output
1	Single-Shot mode	0	The output is high when the counter starts and low when the counter stops
		1	The output is high when the event arrives and low when the counter stops
	8-bit PWM mode	Not applicable	8-bit PWM mode
	Other modes	Not applicable	The Capture/Compare Pin Initial Value (CCMPINIT) bit in the Control B (TCBn.CTRLB) register selects the initial output level
0	Not applicable	Not applicable	No output

Changing modes while the peripheral is enabled is not recommended, as this can produce an unpredictable output. An interrupt flag may be set during the timer configuration. Clearing the TCB Interrupt Flags (TCBn.INTFLAGS) register is recommended after configuring this peripheral.

22.3.3.3. 32-Bit Input Capture

The two 16-bit Timer/Counter Type B (TCBn) peripherals can be combined into a 32-bit input capture

One TCB is counting the two LSBs. Once this counter reaches MAX, an overflow (OVF) event is generated, and the counter wraps around. The second TCB is configured to count these OVF events and thus provides the two MSBs. The 32-bit counter value is concatenated from the two counter values.

To function as a 32-bit counter, set up the two TCBs and the system as described in the following paragraphs.

System Configuration

- Configure a source (TCE, events, CLK_PER) for the count input for the LSB TCB according to the application requirements
- Configure the Event System to route the OVF events from the LSB TCB (event generator) to the MSB TCB (event user)
- Configure the Event System to route the same capture event (CAPT) generator to both TCBs

Configuration of the LSB Counter

- Select the configured count input by writing the Clock Select (CLKSEL) bit field in the Control A (CTRLA) register
- Write the Timer Mode (CNTMODE) bit field in the Control B (CTRLB) register to select one of the Input Capture modes
- The Cascade Two Timer/Counters (CASCADE) bit in CTRLA must be '0'

Configuration of the MSB Counter

- Enable the 32-bit mode by writing the Cascade Two Timer/Counters (CASCADE) bit in CTRLA to '1'
- Select events as clock input by writing to the Clock Select (CLKSEL) bit field in the Control A (CTRLA) register
- Write the Timer Mode (CNTMODE) bit field in the Control B (CTRLB) register to select the same Input Capture mode as the LSB TCB

Capturing a 32-Bit Counter Value

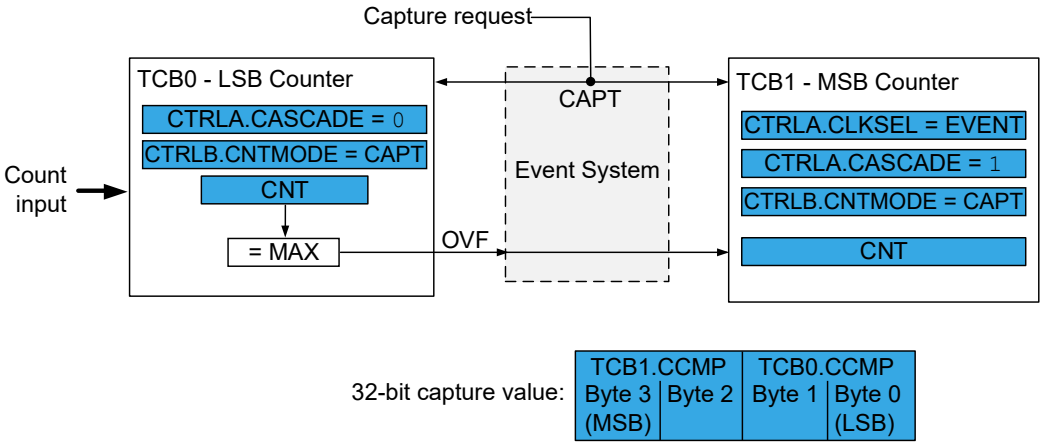
To acquire a 32-bit counter value, send a CAPT event to both TCBs. Both TCBs run in the same Capture mode, so each will capture the current counter value (CNT) in the respective Capture/Compare (CCMP) register. The 32-bit capture value is formed by concatenating the two CCMP registers.

Example 22-1. Using TCB0 as LSB Counter and TCB1 as MSB Counter

TCB0 counts the count input, and TCB1 counts the OVF signals from TCB0. Both TCBs are in Input Capture on Event mode.

A CAPT event is generated and causes both TCB0 and TCB1 to copy their current CNT values to their respective CCMP registers. The two different CASCADE bit values allow the correct timing of the CAPT event.

The captured 32-bit value is concatenated from TCB1.CCMP (MSB) and TCB0.CCMP (LSB).



22.3.3.4. 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 peripheral clock cycles between a change applied to the input and the update of the Input Compare register.

The Noise Canceler uses the peripheral clock and is, therefore, not affected by the prescaler.

22.3.3.5. Synchronized with Timer/Counter Type E

The TCB peripheral can be configured to use the clock (CLK_TCE) of a Timer/Counter type E (TCEn) by writing to the Clock Select (CLKSEL) bit field in the Control A (TCBn.CTRLA) register. In this setting, the TCB will count on the same clock source selected in TCEn.

Writing a '1' to the Synchronize Update (SYNCUPD) bit in the Control A (TCBn.CTRLA) register restarts the TCB counter whenever the TCEn counter restarts or overflows.

22.3.3.6. Port Override

Writing a '1' to the Compare/Capture Output Enable (CCMPEN) bit in the Control B (TCBn.CTRLB) enables and makes the waveform output available on the corresponding pin, overriding the value in the corresponding PORT output register.

22.3.4. Events

The TCB can generate the events described in the following table:

Table 22-3. 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
	OVF	OVF flag set			

The conditions for generating the CAPT and OVF events are identical to those that will raise the corresponding interrupt flags in the Timer/Counter Interrupt Flags (TCBn.INTFLAGS) register. Refer to the *EVSYS - Event System* section for more details regarding event users and configuration.

The TCB can receive the events described in the following table:

Table 22-4. 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
	COUNT	Event as clock source in combination with a count mode		Sync

CAPT and COUNT are TCB event users that detect and act upon input events.

The COUNT event user is enabled on the peripheral by modifying the Clock Select (CLKSEL) bit field in the Control A (TCBn.CTRLA) register to EVENT and setting up the Event System accordingly.

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 must 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 peripheral clock cycle.

22.3.5. Interrupts

Table 22-5. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
CAPT	TCB interrupt	Depending on the operating mode. See the description of the Capture Interrupt Flag (CAPT) bit in the Interrupt Flags (TCBn.INTFLAGS) register
OVF		The TCB peripheral overflows from MAX to BOTTOM

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.

22.3.6. Sleep Mode Operation

The TCB peripheral is disabled by default when in the Standby sleep mode.

The peripheral can stay fully operational in the Standby sleep mode by writing a '1' to the Run Standby (RUNSTDBY) bit in the Control A (TCBn.CTRLA) register.

All operations are halted in the Power-Down sleep mode.

22.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		RUNSTDBY	CASCADE	SYNCUPD		CLKSEL[2:0]		ENABLE
0x01	CTRLB	7:0	EVGEN	ASYNC	CCMPINIT	CCMPEN		CNTMODE[2:0]		
0x02	CTRLC	7:0						CNTSIZE[2:0]		
0x03	Reserved									
0x04	EVCTRL	7:0		FILTER		EDGE				CAPTEI
0x05	INTCTRL	7:0							OVF	CAPT
0x06	INTFLAGS	7:0							OVF	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]							

22.5. Register Description

22.5.1. Control A

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

Bit	7	6	5	4	3	2	1	0
		RUNSTDBY	CASCADE	SYNCUPD	CLKSEL[2:0]			ENABLE
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 6 – RUNSTDBY Run Standby
Writing a '1' to this bit enables the TCB peripheral to run in the Standby sleep mode.

Bit 5 – CASCADE Cascade Two Timer/Counters
Writing a '1' to this bit enables cascading two 16-bit Timer/Counters type B (TCBn) for 32-bit operation using the Event System. This bit must be '1' for the TCBn used for the two Most Significant Bytes (MSBs). When this bit is '1', the selected event source for capture (CAPT) is delayed by one peripheral clock cycle, compensating for the carry propagation delay when cascading two counters via the Event System.

Bit 4 – SYNCUPD Synchronize Update
When this bit is written to '1', the TCBn restarts whenever TCEn is restarted or overflows, which can synchronize the capture with the PWM period. If TCEn is selected as the clock source, the TCB will restart when that TCEn is restarted. For other clock selections, it will restart together with TCEn.

Bits 3:1 – CLKSEL[2:0] Clock Select
Writing this bit field selects the clock source for this peripheral.

Value	Name	Description
0x0	DIV1	CLK_PER
0x1	DIV2	CLK_PER/2
0x2	DIV8	CLK_PER/8
0x3	DIV64	CLK_PER/64
0x4	DIV1024	CLK_PER/1024
0x5	TCE0	CLK_TCE from TCE0
0x6	-	Reserved
0x7	EVENT	Positive edge on event input

Bit 0 – ENABLE Enable
Writing a '1' to this bit enables the TCB peripheral.

22.5.2. Control B

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

Bit	7	6	5	4	3	2	1	0
	EVGEN	ASYNC	CCMPINIT	CCMPEN		CNTMODE[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 – EVGEN Event Generation
Writing a '1' to this bit generates the waveform output on the event line instead of the compare match. The CCMPEN bit is ignored for the event output.

Value	Name	Description
0	PULSE	An event pulse is generated when there is a compare match or capture
1	WAVEFORM	The event output is equal to the waveform output for modes where there is waveform output

Bit 6 – ASYNC Asynchronous Enable
Writing a '1' to this bit allows 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 sets the initial output pin value 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
This bit controls whether the waveform output on the pin corresponding to WO is enabled.

Value	Description
0	Waveform output is not enabled on the corresponding pin
1	Waveform output is enabled and will be available on the pin corresponding to WO, which will override the configuration in the *I/O Configuration (PORT)* peripheral and automatically set the direction of the corresponding pin to output.

Bits 2:0 – CNTMODE[2:0] Timer Mode
Writing to this bit field 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

22.5.3. Control C

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

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

Bits 2:0 – CNTSIZE[2:0] Counter Size
This bit field controls the size of the TCB. Bits in the Counter (TCBn.CNT) register and in the Capture/Compare (TCBn.CCMP) register exceeding the selected size are treated as if being '0'.

Value	Name	Description
0x0	16BITS	The Counter and Capture/Compare registers are 16 bits registers, with a maximum value of 0xFFFF
0x1	15BITS	The Counter and Capture/Compare registers are 15 bits registers, with a maximum value of 0x7FFF
0x2	14BITS	The Counter and Capture/Compare registers are 14 bits registers, with a maximum value of 0x3FFF
0x3	13BITS	The Counter and Capture/Compare registers are 13 bits registers, with a maximum value of 0x1FFF
0x4	12BITS	The Counter and Capture/Compare registers are 12 bits registers, with a maximum value of 0x0FFF
0x5	11BITS	The Counter and Capture/Compare registers are 11 bits registers, with a maximum value of 0x07FF
0x6	10BITS	The Counter and Capture/Compare registers are 10 bits registers, with a maximum value of 0x03FF
0x7	9BITS	The Counter and Capture/Compare registers are 9 bits registers, with a maximum value of 0x01FF

22.5.4. 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 a '1' to this bit 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 Timer Mode (CNTMODE) bit field in the Control B (TCBn.CTRLB) register. "—" means an event or edge has no effect in this mode.

Timer 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	Start counter
8-bit PWM mode	0	—	—
	1	—	—

Bit 0 – CAPTEI Capture Event Input Enable
Writing a '1' to this bit enables the input capture event.

22.5.5. Interrupt Control

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

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

Bit 1 – OVF Overflow Interrupt Enable
Writing a '1' to this bit enables interrupt on overflow.

Bit 0 – CAPT Capture Interrupt Enable
Writing a '1' to this bit enables interrupt on capture.

22.5.6. Interrupt Flags

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

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

Bit 1 – OVF Overflow Interrupt Flag
This bit is set when an overflow interrupt occurs. The flag is set whenever the timer/counter wraps from MAX to BOTTOM.
The bit is cleared by writing a '1' to the bit position.

Bit 0 – CAPT Capture Interrupt Flag
This bit is set when a capture interrupt occurs. The interrupt conditions depend on the Timer Mode (CNTMODE) bit field in the Control B (TCBn.CTRLB) register.
This bit is cleared by writing a '1' or when the Capture register is read in Capture mode.

Table 22-6. Interrupt Sources Set Conditions by Timer Mode

Timer 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 CCMH	CCML	CNT == CCMH

22.5.7. 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.

22.5.8. 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

22.5.9. 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 *Memories* 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

22.5.10. Counter

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] Counter Value High
This bit field holds the MSB of the 16-bit Counter register.

Bits 7:0 – CNT[7:0] Counter Value Low
This bit field holds the LSB of the 16-bit Counter register.

22.5.11. 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 the Capture operation, this register contains the captured value of the counter at the time the capture occurs
- In Periodic Interrupt, Time-Out Check, 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
This bit field holds the MSB of the 16-bit compare, capture and top value.

Bits 7:0 – CCMP[7:0] Capture/Compare Value Low Byte
This bit field holds the LSB of the 16-bit compare, capture and top value.

23. TCE - 16-Bit Timer/Counter Type E

23.1. 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

23.2. Overview

The flexible 16-Bit PWM Timer/Counter type E (TCE) provides accurate program execution timing, frequency and waveform generation, and command execution.

The Timer/Counter consists of a base counter and compare channels. The base counter can be used to count clock cycles or events or let events control how it counts clock cycles. The counting direction and period setting control is used for accurate timing. The compare channels can be used with the base counter for 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 synchronize operations.

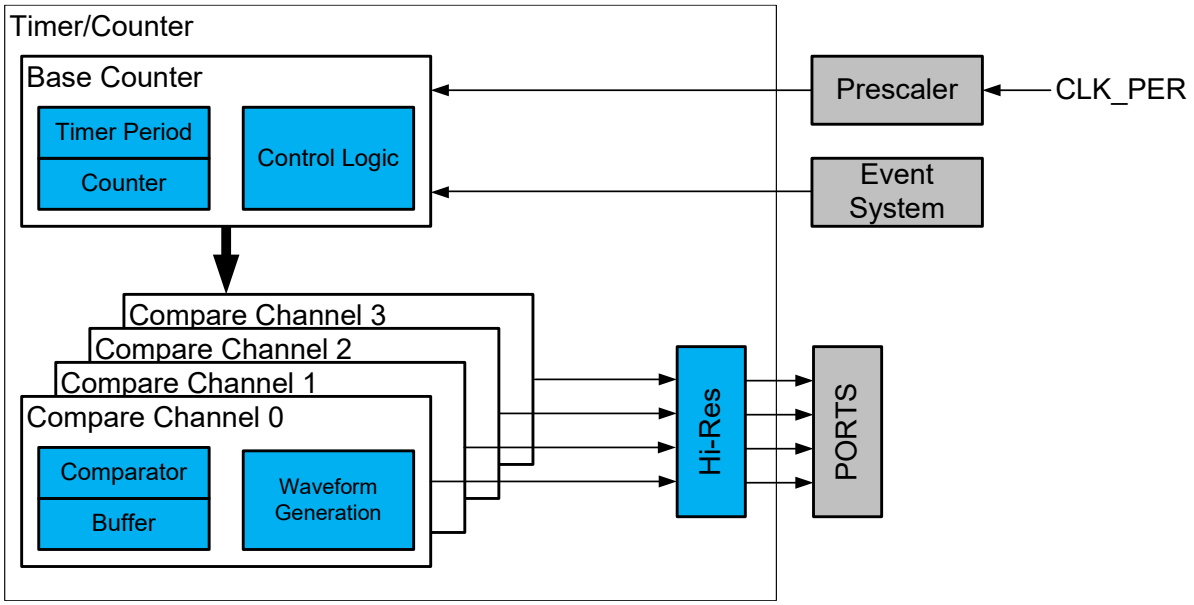
The counter register (TCEn.CNT), period registers with buffer (TCEn.PER and TCEn.PERBUF), and compare registers with buffers (TCEn.CMPn and TCEn.CMPBUFn) are 16-bit registers. All buffer registers have a buffer valid (BV) flag that indicates when the buffer contains a new value.

During normal operation, the counter value is compared continuously to zero and the period (PER) value to determine whether the counter has reached TOP or BOTTOM.

The counter value is also compared to the TCEn.CMPn registers. These comparisons can be used to generate interrupt requests. The Waveform Generator modes use these comparisons to set the waveform period or pulse width.

To control the counter, use a prescaled peripheral clock and events from the Event System.

Figure 23-1. 16-Bit Timer/Counter and Closely Related Peripherals



23.3. Functional Description

23.3.1. Definitions

The following definitions are used throughout the documentation:

Table 23-1. Timer/Counter Definitions

Name	Description
BOTTOM	The counter reaches BOTTOM when it becomes all zeros
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 Control E (TCEn.CTRLE) register has been set.
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.

23.3.2. Initialization

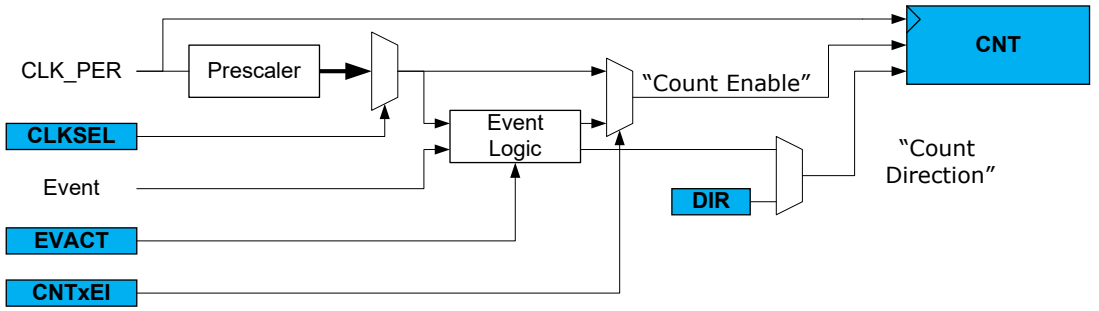
To start using the timer/counter in a basic mode, follow these steps:

1. Write a TOP value to the Period (TCEn.PER) register.
2. Enable the peripheral by writing a '1' to the ENABLE bit in the Control A (TCEn.CTRLA) register. The counter will start counting clock ticks according to the prescaler setting in the Clock Select (CLKSEL) bit field in TCEn.CTRLA.
3. Optional: By writing a '1' to the Enable Counter Event Input A (CNTAEI) bit in the Event Control (TCEn.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 (TCEn.CNT) register.

23.3.3. Clock Generation

CLK_TCE is either a prescaled peripheral clock or an event from the Event System, as shown in the figure below.

Figure 23-2. Timer/Counter Clock Logic



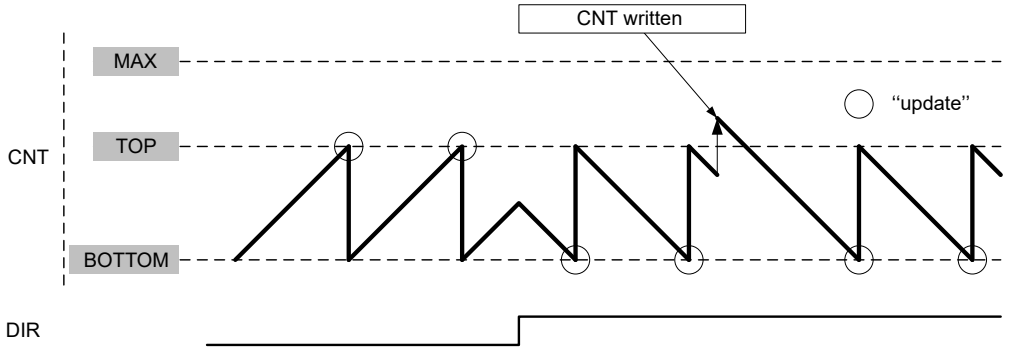
23.3.4. Operation

23.3.4.1. Normal Operation

In normal operation the counter counts clock ticks in the direction selected by the Direction (DIR) bit in the Control E (TCEn.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 (TCEn.CTRLA) register. Alternatively, the Timer/Counter can count on event input.

When reaching TOP 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 (TCEn.PER) register value when the BOTTOM is reached.

Figure 23-3. Normal Operation



The counter value in the Counter (TCEn.CNT) register can be changed while the counter is running. The write access to the TCEn.CNT register 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 the Direction (DIR) bit in the Control E (TCEn.CTRLE) register.

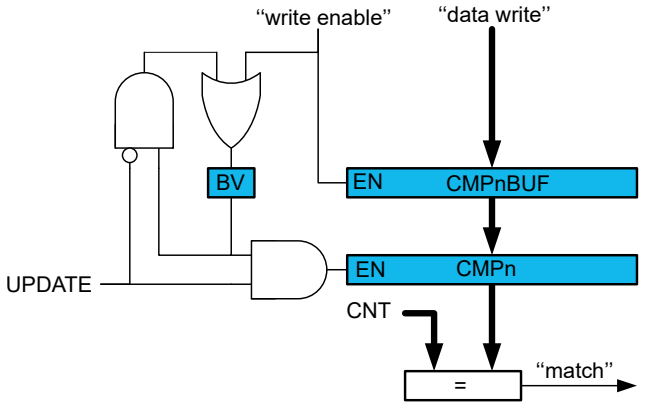
23.3.4.2. Double Buffering

The Period (TCEn.PER) and the Compare n (TCEn.CMPn) registers are all double-buffered (TCEn.PERBUF and TCEn.CMPnBUF).

Each buffer register has a Buffer Valid (BV) flag (PERBV, CMPnBV) in the Control F (TCEn.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 using the TCEn.PER and TCEn.CMPn registers for a compare operation, the BV flag is automatically set when data are written to the buffer

register and cleared on an UPDATE condition. The figure below shows the steps when writing to the Compare (CMPn) register using Double Buffering.

Figure 23-4. Compare Double Buffering



The TCEn.CMPn and TCEn.CMPnBUF registers are available as I/O registers, allowing the initialization and bypassing of the buffer register and the double-buffering function.

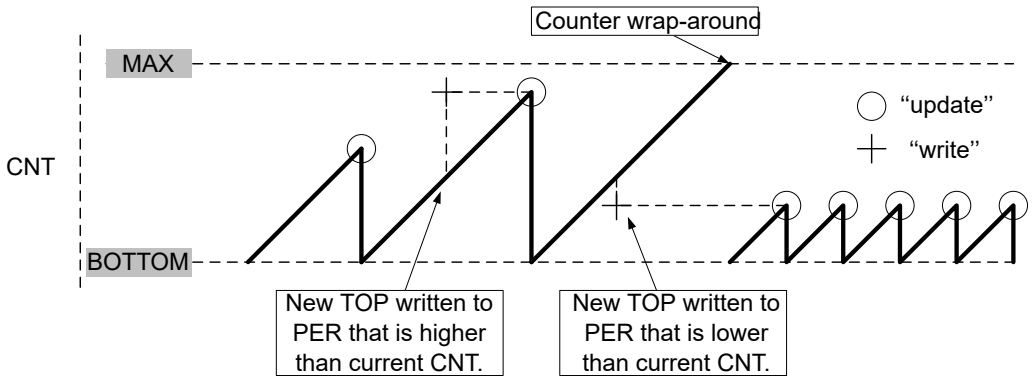
23.3.4.3. Changing the Period

The Counter period is changed by writing a new TOP value to the Period (TCEn.PER) register.

Changing the Period Without Buffering

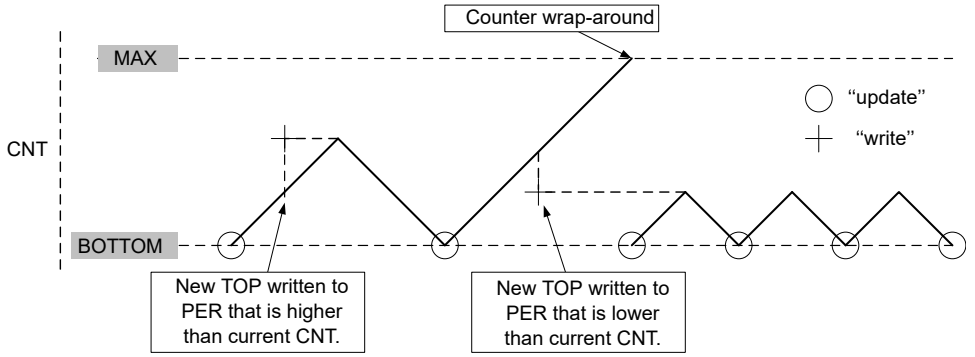
When not using Double Buffering, any period update is immediate.

Figure 23-5. Changing the Period Without Buffering



A counter wraparound can occur in any mode of operation when up-counting without buffering. This happens because the TCEn.CNT and TCEn.PER registers are continuously compared. If a new TOP value is written to TCEn.PER that is lower than current TCEn.CNT, the counter will wrap first before a compare match happens.

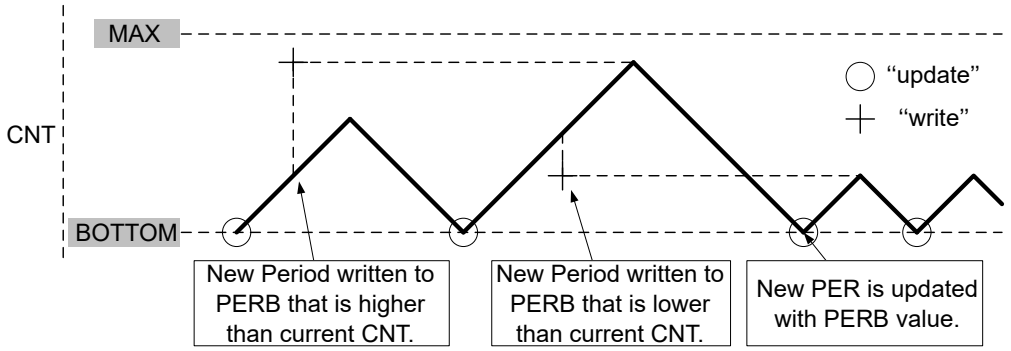
Figure 23-6. Unbuffered Dual-Slope Operation



Changing the Period Using Buffering

When using Double Buffering, the buffer can be written at any time and maintain the correct operation. The TCEn.PER is always updated on the UPDATE condition, as shown for dual-slope operation in the figure below, preventing wrap-around and the generation of odd waveforms.

Figure 23-7. Changing the Period Using Buffering



23.3.4.4. Compare Channel

Each Compare Channel n continuously compares the Counter (TCEn.CNT) register with the Compare n (TCEn.CMPn) register. If TCEn.CNT equals TCEn.CMPn, the Comparator n signals a match. The match will set the Compare Channel's interrupt flag at the next timer clock cycle, generating the optional interrupt.

The Compare n Buffer (TCEn.CMPnBUF) register provides a double-buffer capability equivalent to that for the period buffer. The double-buffering synchronizes the update of the TCEn.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.

The value in CMPnBUF is moved to CMPn at the UPDATE condition and compared to the Counter (TCEn.CNT) register from the next count. If the CMPnBUF contains the same value as TCEn.CNT, a match event will not occur.

23.3.4.4.1. Waveform Generation

The compare channels can be used for waveform generation on the corresponding port pins. The following requirements are mandatory to make the waveform visible on the connected port pin:

1. Select a Waveform Generation mode by writing the Waveform Generation Mode (WGMODE) bit field in the TCEn.CTRLB register.

2. Enable the used compare channels (CMPnEN = '1' in TCEn.CTRLB), thus overriding the output value for the corresponding pin. An alternative pin can be selected by configuring the Port Multiplexer (PORTMUX). Refer to the *PORTMUX - Port Multiplexer* section for details.
3. Optional: Enable the inverted waveform output for the associated port pin n. Refer to the *PORT - I/O Pin Configuration* section for details.

Table 23-2. Timer Waveform Generation Mode

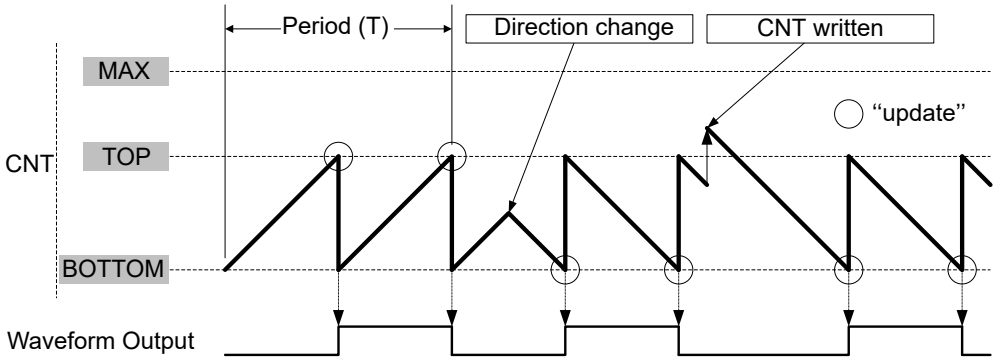
WGMODE[2:0]	Group Configuration	Mode of Operation	Top	Update	OVF
000	NORMAL	Normal	PER	TOP/BOTTOM(1)	TOP/BOTTOM(1)
001	FRQ	Frequency	CMP0	TOP/BOTTOM(1)	TOP/BOTTOM(1)
010	-	Reserved	-	-	-
011	SINGLESLOPE	Single-Slope PWM	PER	BOTTOM	BOTTOM
100	-	Reserved	-	-	-
101	DSTOP	Dual-Slope PWM	PER	BOTTOM	TOP
110	DSBOTH	Dual-Slope PWM	PER	BOTTOM	TOP and BOTTOM
111	DSBOTTOM	Dual-Slope PWM	PER	BOTTOM	BOTTOM

Note: TOP for up-count and BOTTOM for down-count.

23.3.4.4.2. Frequency Waveform Generation (FRQ)

In Frequency Generation mode, the TCEn.CMP0 register controls the period time (T), instead of the Period (TCEn.PER) register. The corresponding waveform generator output is toggled on each compare match between the TCEn.CNT and TCEn.CMPn registers.

Figure 23-8. Frequency Waveform Generation



The following equation defines the waveform frequency (f_{FRQ}):

$$f_{FRQ} = \frac{f_{CLK\_PER}}{2N(CMP0 + 1)}$$

where N represents the prescaler divider used (see the CLKSEL bit field in the TCEn.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 TCEn.CMP0 is written to 0x0000, and no prescaling is used (N = 1, CLKSEL = 0x0 in TCEn.CTRLA).

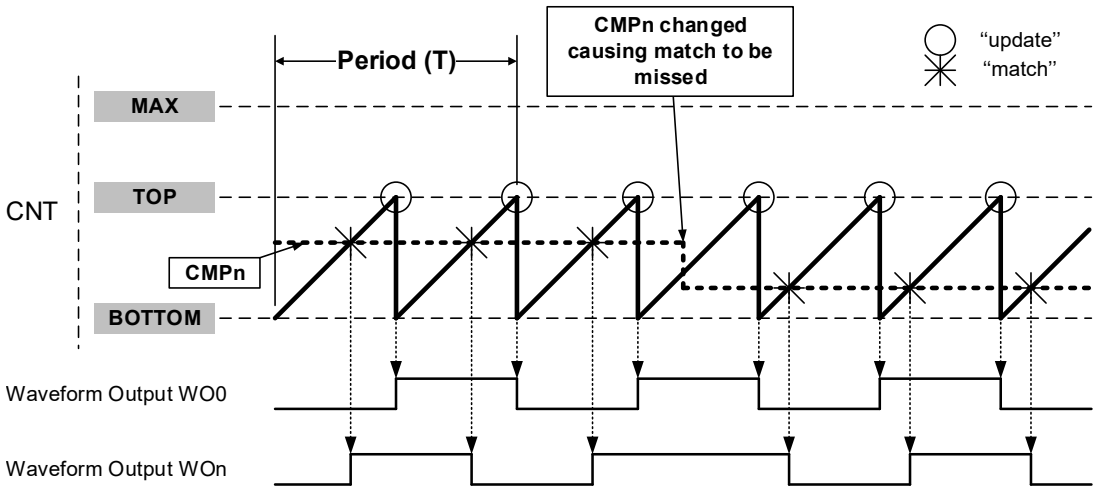
Use the TCEn.CMP1 and TCEn.CMP2 registers to get additional waveform outputs WOn. The waveforms WOn can either be identical or offset to WO0. The offset can be influenced by TCEn.CMPn, TCEn.CNT and the count direction. Use the equations shown in the table below to calculate the offset in seconds (T_{offset}). The equations are only valid when CMPn < CMP0.

Table 23-3. Offset Equation Overview

Equation	Count Direction	CMPn vs. CNT State	Offset
$T_{offset} = \left(\frac{CMP0 - CMPn}{CMP0 + 1}\right)\left(\frac{T}{2}\right)$	UP	CMPn ≥ CNT	WOn leading WO0
	DOWN	CMP0 ≤ CNT	WOn trailing WO0
		CMP0 > CNT and CMPn > CNT	WOn trailing WO0
$T_{offset} = \left(\frac{CMPn + 1}{CMP0 + 1}\right)\left(\frac{T}{2}\right)$	UP	CMPn < CNT	WOn trailing WO0
	DOWN	CMPn ≤ CNT	WOn leading WO0

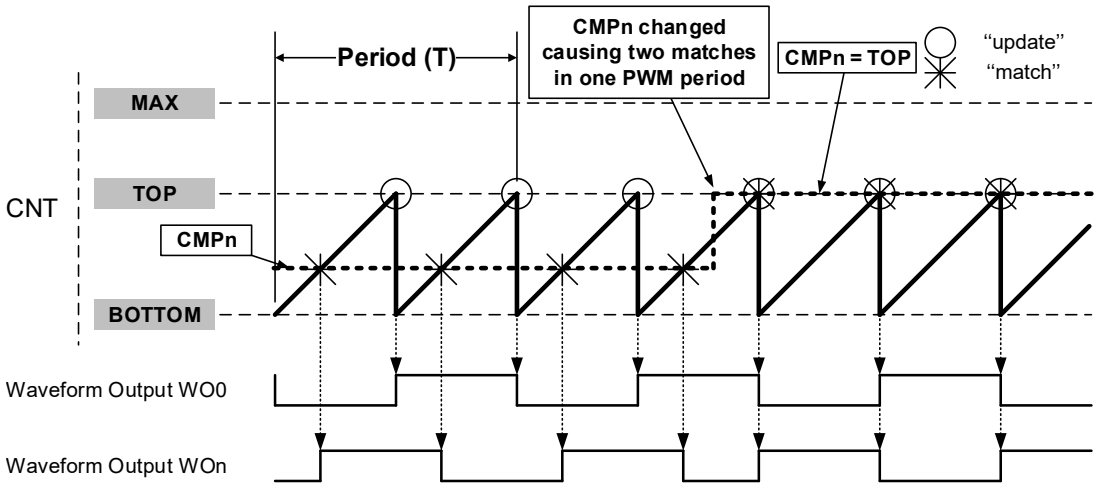
The figure below shows the leading and trailing offset for WOn, where both equations can be used. The correct equation is determined by count direction and the state of CMPn vs. CNT when the timer is enabled or the CMPn is changed.

Figure 23-9. Offset When Counting Up



The figure below shows how the waveform can be inverted by changing the CMPn during run-time.

Figure 23-10. Inverting Waveform Output



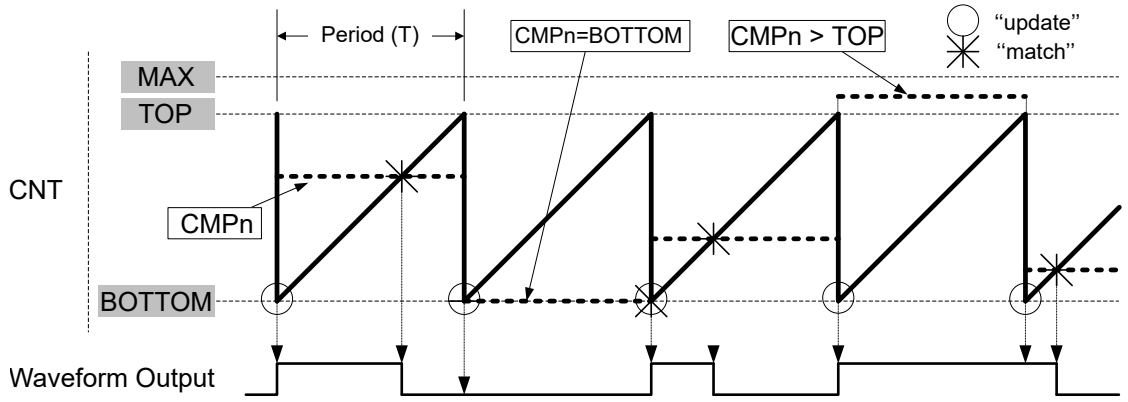
23.3.4.4.3. Single-Slope PWM Generation

For Single-Slope Pulse-Width Modulation (PWM) generation, the period (T) is controlled by the TCEn.PER register, while the values of the TCEn.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 TCEn.CNT and TCEn.CMPn registers.

CMPn = BOTTOM will produce a static low signal on WOn, while CMPn > TOP will generate a static high signal on WOn.

Figure 23-11. Single-Slope Pulse-Width Modulation



The TCEn.PER register defines the PWM resolution. The minimum resolution is two bits (TCEn.PER = `0x0002`), and the maximum is 16 bits (TCEn.PER = MAX-1).

The following equation calculates the exact resolution in bits for Single-Slope PWM (R_{PWM_SS}):

$$R_{PWM\_SS} = \frac{\log(PER+1)}{\log(2)}$$

The Single-Slope PWM frequency (f_{PWM_SS}) depends on the period setting (TCEn.PER), the peripheral clock frequency f_{CLK_PER}, and the TCE prescaler (the CLKSEL bit field in the TCEn.CTRLA register). It is calculated by the following equation, where N represents the prescaler divider used:

$$f_{PWM\_SS} = \frac{f_{CLK\_PER}}{N(PER+1)}$$

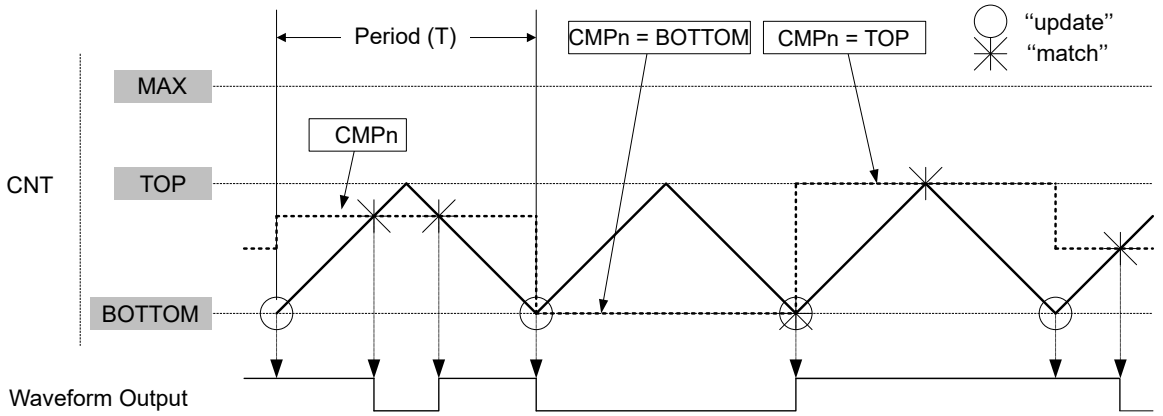
23.3.4.4.4. Dual-Slope PWM

For the Dual-Slope PWM generation, the period (T) is controlled by TCEn.PER, while the values of TCEn.CMPn control the duty cycle of the WG output.

The figure below shows how, for Dual-Slope PWM, the counter repeatedly counts 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 produces a static low signal on WOn, while CMPn = TOP produces a static high signal on WOn.

Figure 23-12. Dual-Slope Pulse-Width Modulation



The Period (TCEn.PER) register defines the PWM resolution. The minimum resolution is two bits (TCEn.PER = 0x0003), and the maximum is 16 bits (TCEn.PER = MAX).

The following equation calculates the exact resolution in bits for Dual-Slope PWM (R_{PWM_DS}):

$$R_{PWM\_DS} = \frac{\log(PER+1)}{\log(2)}$$

The PWM frequency depends on the period setting in the TCEn.PER register, the peripheral clock frequency (f_{CLK_PER}), and the prescaler divider selected in the CLKSEL bit field in the TCEn.CTRLA register. It is calculated by the following equation:

$$f_{PWM\_DS} = \frac{f_{CLK\_PER}}{2N \times PER}$$

N represents the prescaler divider used.

Using Dual-Slope PWM results in approximately half the maximum operation frequency compared to Single-Slope PWM operation due to twice the number of timer increments per period.

23.3.4.4.5. Port Override for Waveform Generation

The TCEn will override the port pin values when the compare channel is enabled (CMPnEN = 1 in the TCEn.CTRLB register) and a Waveform Generation mode is selected.

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 the PORTx.PINn register) inverts the corresponding WG output.

23.3.4.5. Timer/Counter Commands

The software can issue a command set to change the peripheral state immediately. 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 (TCEn.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 (TCEn.CTRLE) register.

The software can force a restart of the current waveform period by issuing a RESTART command. In this case, the counter, and all waveform 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 TCEn.CTRLA register).

23.3.4.6. Interfacing with Waveform Extension

The TCEn can be interfaced with the Waveform Extension (WEX) peripheral to provide extra functions to the Timer/Counter in Waveform Generation modes.

Note: The TCEn waveform outputs are inputs to the WEX peripheral, and the WOn outputs pins are overridden by the WEX module when WEX functionalities are enabled. For more details, refer to the *WEX - Waveform Extension* section.

23.3.5. Events

The TCE can generate the events described in the table below.

Table 23-4. Event Generators in TCE

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
TCE0	OVF	Overflow/Low-byte timer underflow	Sync,Pulse	CLK_PER	One CLK_PER period
	CMP0	Compare channel 0 match/Low-byte timer compare channel 0 match	Sync,Pulse/Level		One CLK_PER period or WO
	CMP1	Compare channel 1 match/Low-byte timer compare channel 1 match			
	CMP2	Compare channel 2 match/Low-byte timer compare channel 2 match			

Table 23-5. Event User in TCE

User Name		Description	Input Detection	Async/Sync
Peripheral	Input			
TCEn	CNTA	Count on a positive event edge	Edge	Sync
		Count on any event edge	Edge	
		Count with the CLKSEL setting while the event signal is high	Level	
		Count with the CLKSEL setting. The event input controls the direction.	Level	
	CNTB	Count with the CNTA or CLKSEL setting. The event controls the count direction.	Level	
		Restart the counter on a positive event edge	Edge	
		Restart the counter on any event edge	Edge	
		Restart the counter while the event signal is high	Level	

23.3.6. Interrupts

Table 23-6. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
OVF	Overflow or underflow interrupt	The counter has reached the 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
CMP3	Compare Channel 3 interrupt	Match between the counter value and the Compare 3 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 InterruptControl (*peripheral*.INTCTRL) register.

23.3.7. Sleep Mode Operation

The TCE is, by default, disabled in Standby sleep mode and will halt when entering the sleep mode.

The module can stay fully operational in Standby sleep mode if the Run Standby (RUNSTDBY) bit in the TCEn.CTRLA register is written to '1'.

All operations are halted in Power-Down sleep mode.

23.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY				CLKSEL[2:0]			ENABLE
0x01	CTRLB	7:0		CMP2EN	CMP1EN	CMP0EN	ALUPD	WGMODE[2:0]		
0x02	CTRLC	7:0		CMP2POL	CMP1POL	CMP0POL		CMP2OV	CMP1OV	CMP0OV
0x03	Reserved									
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	EVGENCTRL	7:0		CMP2EV	CMP1EV	CMP0EV				
0x09	EVCTRL	7:0		EVACTB[2:0]		CNTBEI	EVACTA[2:0]			CNTAEI
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	CMPBUFL[7:0]							
		15:8	CMPBUFH[15:8]							
0x3A	CMP1BUF	7:0	CMPBUFL[7:0]							
		15:8	CMPBUFH[15:8]							
0x3C	CMP2BUF	7:0	CMPBUFL[7:0]							
		15:8	CMPBUFH[15:8]							

23.5. Register Description

23.5.1. Control A

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

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

Bit 7 – RUNSTDBY Run Standby
Writing a '1' to this bit will enable the peripheral to run in Standby sleep mode.

Bits 3:1 – CLKSEL[2:0] Clock Select
These bits select the clock frequency for the timer/counter.

Value	Name	Description
0x0	DIV1	$f_{TCE} = f_{CLK\_PER}$ (no prescaling)
0x1	DIV2	$f_{TCE} = f_{CLK\_PER}/2$
0x2	DIV4	$f_{TCE} = f_{CLK\_PER}/4$
0x3	DIV8	$f_{TCE} = f_{CLK\_PER}/8$
0x4	DIV16	$f_{TCE} = f_{CLK\_PER}/16$
0x5	DIV64	$f_{TCE} = f_{CLK\_PER}/64$
0x6	DIV256	$f_{TCE} = f_{CLK\_PER}/256$
0x7	DIV1024	$f_{TCE} = f_{CLK\_PER}/1024$

Bit 0 – ENABLE Enable

Value	Name	Description
0	DISABLED	Peripheral disabled
1	ENABLED	Peripheral enabled

23.5.2. Control B

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 – CMPnEN Compare n Enable
In the FRQ and PWM Waveform Generation modes, the Compare n Enable (CMPnEN) bits will make the TCEn waveform output available on the pin corresponding to WOn, overriding the value in the corresponding PORT output register.
Note: If Waveform Extension (WEX) is enabled, the WEX outputs override the TCEn waveform outputs.

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 in FRQ and PWM Waveform Generation mode

Bit 3 – ALUPD Auto-Lock Update
The Auto-Lock Update bit controls the Lock Update (LUPD) bit in the TCEn.CTRLE register. When writing ALUPD to '1', the LUPD bit will be set to '1' until the Buffer Valid (CMPnBV) bits of all enabled compare channels are '1'. This condition will clear the LUPD bit.
It will remain clear until the following UPDATE condition, where the CMPnBUF registers will be transferred to the CMPn registers, and the LUPD bit will be set to '1' again, making sure that the CMPnBUF register values are not transferred to the CMPn registers until all enabled compare buffers are written.

Value	Description
0	LUPD bit in the TCEn.CTRLE register is not altered by the system
1	LUPD bit in the TCEn.CTRLE register is set and cleared automatically

Bits 2:0 – WGMODE[2:0] Waveform Generation Mode
This bit field selects the Waveform Generation mode and controls 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.

Value	Name	Description
0x0	NORMAL	Normal operation mode
0x1	FRQ	Frequency mode
0x2	-	Reserved
0x3	SINGLESLOPE	Single-Slope PWM mode
0x4	-	Reserved
0x5	DSTOP	Dual-Slope PWM mode with overflow on TOP
0x6	DSBOTH	Dual-Slope PWM mode with overflow on TOP and BOTTOM
0x7	DSBOTTOM	Dual-Slope PWM mode with overflow on BOTTOM

23.5.3. Control C

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

Bit	7	6	5	4	3	2	1	0
		CMP2POL	CMP1POL	CMP0POL		CMP2OV	CMP1OV	CMP0OV
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 4, 5, 6 – CMPnPOL Compare n Polarity
Setting these bits inverts the polarity of the waveform output.

Bits 0, 1, 2 – CMPnOV Compare n Output Value
The CMPnOV bits allow direct access to the waveform generator's output value when the timer/counter is not enabled, which is used to set or clear the Waveform Generator output value when the timer/counter is not running.

23.5.4. Control Register E Clear

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

Note: Use this register 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
This bit field is used for software control of updating, restarting, and resetting the timer/counter. The command bit field always reads 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
Before performing an update, use the lock update to ensure all buffers are valid. Writing a '1' to this bit will clear Lock Update.

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. This setting will not prevent an update issued by the Command bit field.

Bit 0 – DIR Counter Direction
Usually, this bit is controlled in hardware by the Waveform Generation mode or by event actions but can also be changed from the software. Writing a '1' to this bit will clear the DIR bit, and the counter will count up.

Value	Description
0	The counter is counting up (incrementing)
1	The counter is counting down (decrementing)

23.5.5. Control Register E Set

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

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
This bit field is used to send software commands to the TCEn. The command bit field clears automatically by hardware after command execution and always reads 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 performing an update. Writing a '1' to this bit will cause the LUPD to be set, and the Update is locked.

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. This setting will not prevent an update issued by the Command bit field.

Bit 0 – DIR Counter Direction
Usually, this bit is controlled in hardware by the Waveform Generation mode or by event actions, but can also be changed from the software. Writing a '1' to this bit will set the DIR bit, and the counter will count down.

Value	Description
0	The counter is counting up (incrementing)
1	The counter is counting down (decrementing)

23.5.6. Control Register F Clear

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

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

Bits 1, 2, 3 – CMPnBV Compare n Buffer Valid
The CMPnBV bits are set when a new value is written to the corresponding TCEn.CMPnBUF register. These bits automatically clear on an `UPDATE` condition.

Bit 0 – PERBV Period Buffer Valid
This bit is set when a new value is written to the TCEn.PERBUF register. This bit automatically clears on an `UPDATE` condition.

23.5.7. Control Register F Set

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

Bit	7	6	5	4	3	2	1	0
					CMP2BV	CMP1BV	CMP0BV	PERBV
Access					R	R	R	R/W
Reset					0	0	0	0

Bits 1, 2, 3 – CMPnBV Compare n Buffer Valid
The CMPnBV bits are set when a new value is written to the corresponding TCEn.CMPnBUF register. These bits automatically clear on an `UPDATE` condition.

Bit 0 – PERBV Period Buffer Valid
This bit is set when a new value is written to the TCEn.PERBUF register. This bit automatically clears on an `UPDATE` condition.

23.5.8. Event Generation Control

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

Bit	7	6	5	4	3	2	1	0
		CMP2EV	CMP1EV	CMP0EV				
Access		R/W	R/W	R/W				
Reset		0	0	0				

Bits 4, 5, 6 – CMPEV Compare n Event Generation
This bit selects the waveform available on the event output.
Note: The CMPnEN in TCEn.CTRLB does not need to be set to generate the event.

Value	Name	Description
0	PULSE	Event pulse generated at each compare match
1	WAVEFORM	Event output is the Waveform Output signal

Note: Generated event equals Waveform Output before output enable, but after POLARITY selection.

23.5.9. Event Control

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

Bit	7	6	5	4	3	2	1	0
	EVACTB[2:0]			CNTBEI	EVACTA[2:0]			CNTAEI
Access	R/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 – EVACTB[2:0] Event Action B
These bits define what action the counter will take upon certain event conditions.

Value	Name	Description
0x0	NONE	No action
0x1	-	Reserved
0x2	-	Reserved
0x3	UPDOWN	Count on the prescaled clock or according to the setting for event input A. The event controls the count direction: • up-counting when the event line is 0 • down-counting when the event line is 1
0x4	RESTART_POSEDGE	Restart counter on positive event edge
0x5	RESTART_ANYEDGE	Restart counter on any event edge
0x6	RESTART_HIGHLVL	Restart counter while the event signal is high
0x7	-	Reserved

Bit 4 – CNTBEI Enable Counter Event Input B

Value	Description
0	Counter Event input B is disabled
1	Counter Event input B is enabled according to EVACTB bit field

Bits 3:1 – EVACTA[2:0] Event Action A
These bits define what action the counter will take upon certain event conditions.

Value	Name	Description
0x0	CNT_POSEDGE	Count on positive event edge
0x1	CNT_ANYEDGE	Count on any event edge
0x2	CNT_HIGHLVL	Count prescaled clock cycles while the event signal is high
0x3	UPDOWN	Count on the prescaled clock. The event controls the count direction: • up-counting when the event line is 0 • down-counting when the event line is 1
0x4 – 0x7	-	Reserved

Bit 0 – CNTAEI Enable Counter Event Input A

Value	Description
0	Counter Event input A is disabled
1	Counter Event input A is enabled according to EVACTA bit field

23.5.10. Interrupt Control Register

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

Bits 4, 5, 6 – CMPn Compare Channel n 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.

23.5.11. Interrupt Flag Register

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

Bits 4, 5, 6 – CMPn Compare Channel n 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 (TCEn.CNT) register and the corresponding Compare (TCEn.CMPn) register. The CMPn flag is not cleared automatically, 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.

Note: Any individual bits can be cleared by writing a '1' to its bit location, allowing each bit to be set without using a Read-Modify-Write operation on a single register.

23.5.12. 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	Name	Description
0	DISABLED	The peripheral is halted in Break Debug mode and ignores events
1	ENABLED	The peripheral will continue to run in Break Debug mode when the CPU is halted

23.5.13. Temporary Register

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 *Memories* 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

23.5.14. Counter Register

Name: CNT
Offset: 0x20
Reset: 0x00
Property: -

The TCEn.CNTL and TCEn.CNTH register pair represents the 16-bit value, TCEn.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
This bit field holds the MSB of the 16-bit Counter register.

Bits 7:0 – CNT[7:0] Counter Low Byte
This bit field holds the LSB of the 16-bit Counter register.

23.5.15. Period Register

Name: PER
Offset: 0x26
Reset: 0xFFFF
Property: -

The TCEn.PERL and TCEn.PERH register pair represents the 16-bit value, TCEn.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`.

TCEn.PER contains the 16-bit TOP value in the timer/counter.

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
This bit field holds the MSB of the 16-bit Period register.

Bits 7:0 – PER[7:0] Period Low Byte
This bit field holds the LSB of the 16-bit Period register.

23.5.16. Compare n Register

Name: CMPn
Offset: 0x28 + n*0x02 [n=0..2]
Reset: 0x00
Property: -

The CMPn register is continuously compared to the counter value. Ordinarily, the outputs from the comparators are then used for generating waveforms. The TCEn.CMPn registers are updated with the buffer value from their corresponding TCEn.CMPnBUF register when an `UPDATE` condition occurs.

The TCEn.CMPnL and TCEn.CMPnH register pair represents the 16-bit value, TCEn.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
This bit field holds the MSB of the 16-bit Compare register.

Bits 7:0 – CMP[7:0] Compare Low Byte
This bit filed holds the LSB of the 16-bit Compare register.

23.5.17. Period Buffer Register

Name: PERBUF
Offset: 0x36
Reset: 0xFFFF
Property: -

This register serves as the buffer for the period register (TCEn.PER). Accessing this register using the CPU or UPDI will affect the PERBV flag. The TCEn.PERBUFL and TCEn.PERBUFH register pair represents the 16-bit value, TCEn.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
This bit field holds the MSB of the 16-bit Period Buffer register.

Bits 7:0 – PERBUF[7:0] Period Buffer Low Byte
This bit field holds the LSB of the 16-bit Period Buffer register.

23.5.18. Compare n Buffer Register

Name: CMPnBUF
Offset: 0x38 + n*0x02 [n=0..2]
Reset: 0x00
Property: -

The TCEn.CMPnBUFL and TCEn.CMPnBUFH register pair represents the 16-bit value, TCEn.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`.

This register serves as the buffer for the associated compare registers (TCEn.CMPn). Accessing these registers using the CPU or UPDI will affect the corresponding CMPnBV status bit.

Bit	15	14	13	12	11	10	9	8
	CMPBUFH[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
	CMPBUFL[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 – CMPBUFH[15:8] Compare High Byte
This bit field holds the MSB of the 16-bit Compare Buffer register.

Bits 7:0 – CMPBUFL[7:0] Compare Low Byte
This bit field holds the LSB of the 16-bit Compare Buffer register.

24. RTC - Real-Time Counter

24.1. 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
- Crystal Error Correction

24.2. 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 value 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 KHz Internal Oscillator (OSC32K), or the OSC32K 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).

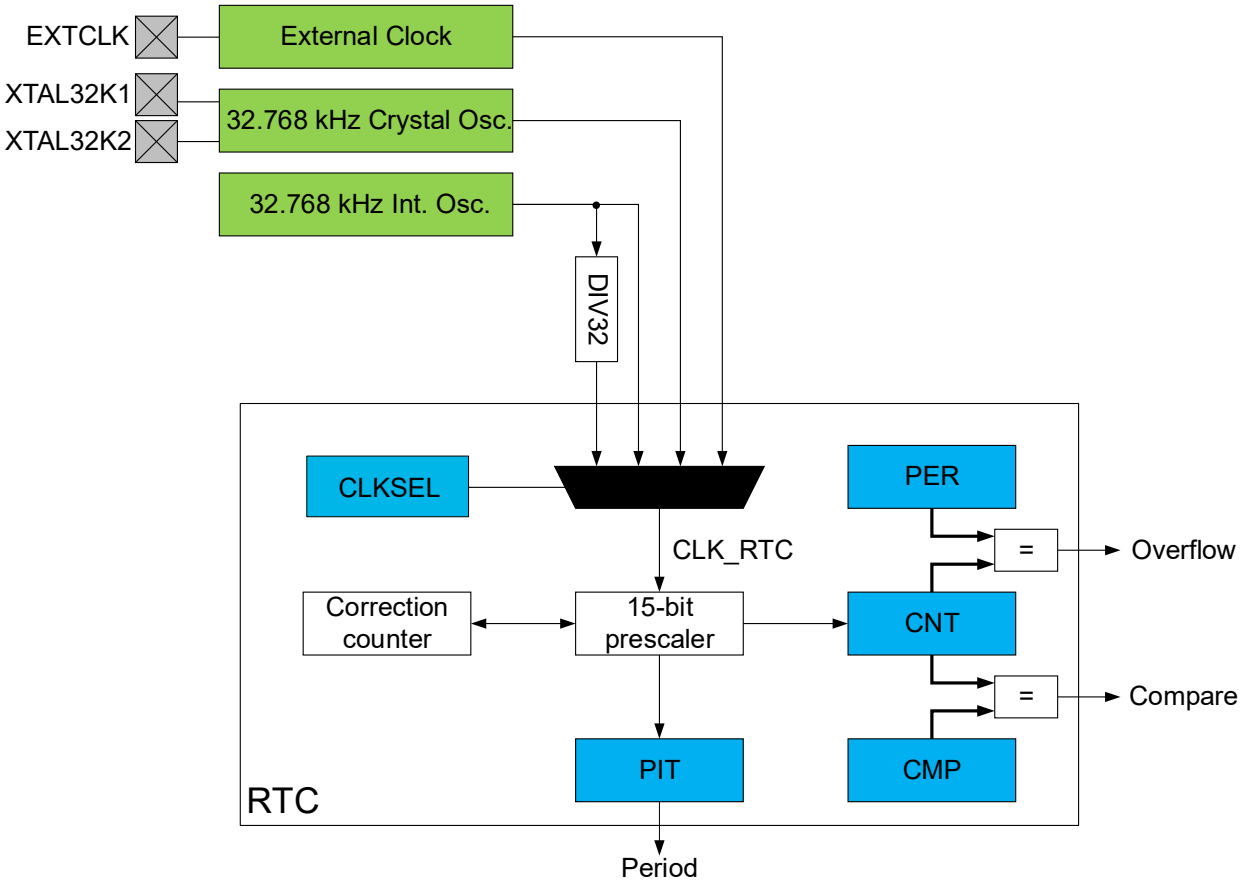
The RTC also supports crystal error correction when operated using external crystal selection. An externally calibrated value will be used for correction. The software can adjust the RTC with an accuracy of ±1 PPM, and the maximum adjustment is ±127 PPM. The RTC correction operation will either speed up (by skipping count) or slow down (by adding extra count) the prescaler to account for the crystal error.

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.

24.2.1. RTC Block Diagram

Figure 24-1. RTC Block Diagram



24.3. 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 XTAL32K1 or XTAL32K2 pins, along with any required load capacitors. Alternatively, an external digital clock can be connected to the XTAL32K1 pin.

24.4. 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.

24.4.1. 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.

24.4.1.1. 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 functionalities.

24.4.1.2. 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.

24.4.2. Operation - RTC

24.4.2.1. 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'.

24.5. 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.

24.5.1. Initialization

To operate the PIT, follow these steps:

1. Configure the RTC clock CLK_RTC as described in section 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. Enable the PIT by writing a '1' to the Periodic Interrupt Timer Enable (PITEN) bit in the RTC.PITCTRLA register.
4. 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.

24.5.2. Operation - PIT

24.5.2.1. 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'.

24.5.2.2. PIT Interrupt Timing

Timing of the First Interrupt

Both PIT and RTC functions 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

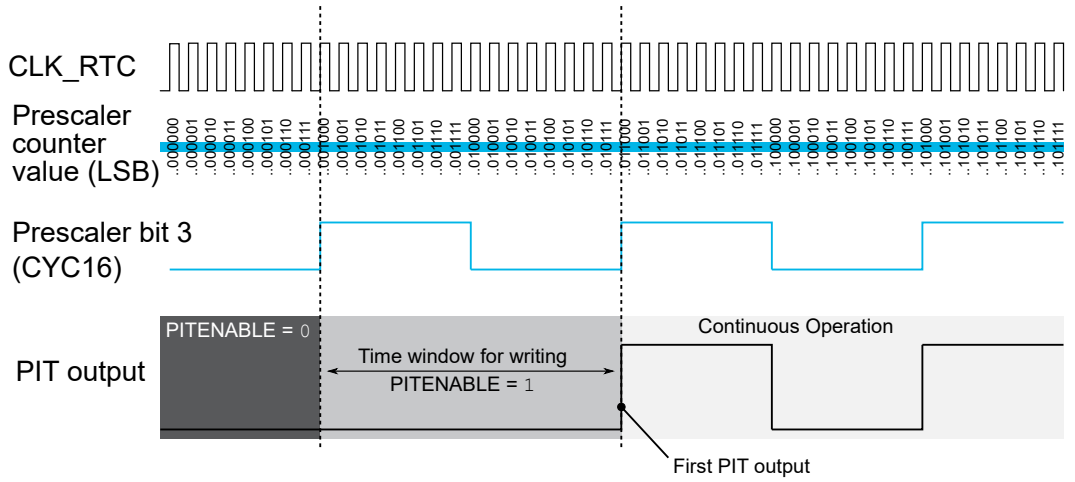
After the first interrupt, the PIT will continue toggling every ½ PIT period resulting in a full PIT period signal.

Example 24-1. 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 24-2. Timing Between PIT Enable and First Interrupt



24.6. Crystal Error Correction

The prescaler for the RTC and PIT can do internal frequency correction of the crystal clock by using the PPM error value from the Crystal Frequency Calibration (CALIB) register when the Frequency Correction Enable (CORREN) bit in the RTC.CTRLA register is '1'.

The CALIB register must be written by the user, based on the information about the frequency error. Perform the correction operation by adding or removing some cycles equal to the value given in the Error Correction Value (ERROR) bit field in the CALIB register spread throughout a million-cycle interval.

The RTC count value available through the Count (RTC.CNT) registers or in the PIT intervals will reflect the clock correction.

If disabling the correction feature, an ongoing correction cycle will be completed before the function is disabled.

Note: If using this feature with a negative correction, the minimum prescaler configuration is DIV2.

24.7. Events

The RTC can generate the events described in the following table:

Table 24-1. Event Generators in RTC

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
RTC	OVF	Overflow	Pulse	CLK_RTC	One CLK_RTC period
	COMP	Compare match			
	EVGEN0	Selectable prescaled RTC event	Level		Given by prescaled RTC clock divided by prescale factor
	EVGEN1				

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.

24.8. Interrupts

Table 24-2. 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.

24.9. 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.

24.10. 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 (CMPBUSY, PERBUSY, CNTBUSY, CTRLABUSY) flag is available in the Status (RTC.STATUS) register.

For the RTC.PITCTRLA register, a Synchronization Busy (CTRLBUSY) flag is available in the Periodic Interrupt Timer Status (RTC.PITSTATUS) register.

Check these flags before writing to the mentioned registers.

24.11. 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 does not happen during normal operation. If the PIT output is low at the break, the PIT will resume low without additional interrupt.

24.12. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY	PRESCALER[3:0]				CORREN		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	CALIB	7:0	SIGN	ERROR[6:0]						
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
0x16	PITEVGENCTRLA	7:0	EVGEN1SEL[3:0]				EVGEN0SEL[3:0]			

24.13. Register Description

24.13.1. Control A

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

Application software needs to check that the CTRLABUSY flag in the RTC.STATUS register is cleared before writing to this register.

Bit	7	6	5	4	3	2	1	0
	RUNSTDBY	PRESCALER[3:0]				CORREN		RTCEN
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 – 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.

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 2 – CORREN Frequency Correction Enable

Value	Description
0	Frequency correction is disabled
1	Frequency correction is enabled

Bit 0 – RTCEN RTC Peripheral Enable

Value	Description
0	RTC peripheral is disabled
1	RTC peripheral is enabled

24.13.2. 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.

24.13.3. 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

24.13.4. 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.

24.13.5. 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 *Memories* 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.

24.13.6. 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

24.13.7. Crystal Frequency Calibration

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

This register stores the error value and the type of correction to be done. The register is written by software with an error value based on external calibration and/or temperature correction/s.

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

Bit 7 – SIGN Error Correction Sign Bit
This bit shows the direction of the correction.

Value	Description
0x0	Positive correction causing the prescaler to count slower
0x1	Negative correction causing the prescaler to count faster. This requires that the minimum prescaler configuration is DIV2

Bits 6:0 – ERROR[6:0] Error Correction Value
The number of correction clocks for each million RTC clock cycles interval (PPM).

24.13.8. 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	OSC32K	32.768 kHz from OSC32K
0x1	OSC1K	1.024 kHz from OSC32K
0x2	XOSC32K	32.768 kHz from XOSC32K
0x3	EXTCLK	External clock from EXTCLK pin

24.13.9. 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 reading or writing 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. Application software needs to check that the CNTBUSY flag in the RTC.STATUS register is cleared before writing to this register.

Bits 7:0 – CNT[7:0] Counter Low Byte
These bits hold the LSB of the 16-bit Counter register. Application software needs to check that the CNTBUSY flag in the RTC.STATUS register is cleared before writing to this register.

24.13.10. 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. Application software needs to check that the PERBUSY flag in the RTC.STATUS register is cleared before writing to this register.

Bits 7:0 – PER[7:0] Period Low Byte
These bits hold the LSB of the 16-bit Period register. Application software needs to check that the PERBUSY flag in the RTC.STATUS register is cleared before writing to this register.

24.13.11. 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`.

Application software needs to check that the CMPBUSY flag in the RTC.STATUS register is cleared before writing to this register.

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. Application software needs to check that the CMPBUSY flag in the RTC.STATUS register is cleared before writing to this register.

24.13.12. 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.
Note: Application software needs to check that the CTRLBUSY flag in the RTC.PITSTATUS register is cleared before writing to this register.

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
Note: Application software needs to check that the CTRLBUSY flag in the RTC.PITSTATUS register is cleared before writing to this register.

Value	Description
0	Periodic Interrupt Timer disabled
1	Periodic Interrupt Timer enabled

Important: 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. Application software needs to check that the CTRLBUSY flag in the RTC.PITSTATUS register is cleared before writing to this register.

24.13.13. 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.

24.13.14. 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

24.13.15. 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.

24.13.16. 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

24.13.17. Periodic Timer Event Generation Control A

Name: PITEVGENCTRLA
Offset: 0x16
Reset: 0x00

Bit	7	6	5	4	3	2	1	0
	EVGEN1SEL[3:0]				EVGEN0SEL[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 – EVGENnSEL Event Generator n Select

Value	Name	Description
0x0	OFF	No event generated
0x1	DIV4	CLK_RTC divided by 4
0x2	DIV8	CLK_RTC divided by 8
0x3	DIV16	CLK_RTC divided by 16
0x4	DIV32	CLK_RTC divided by 32
0x5	DIV64	CLK_RTC divided by 64
0x6	DIV128	CLK_RTC divided by 128
0x7	DIV256	CLK_RTC divided by 256
0x8	DIV512	CLK_RTC divided by 512
0x9	DIV1024	CLK_RTC divided by 1024
0xA	DIV2048	CLK_RTC divided by 2048
0xB	DIV4096	CLK_RTC divided by 4096
0xC	DIV8192	CLK_RTC divided by 8192
0xD	DIV16384	CLK_RTC divided by 16384
0xE	DIV32768	CLK_RTC divided by 32768
other	-	Reserved

25. USART - Universal Synchronous and Asynchronous Receiver and Transmitter

25.1. Features

- Full-Duplex Operation
- Half-Duplex Operation:
  - One-Wire mode with Loop-back and Open-Drain
  - RS-485 mode
- Asynchronous or Synchronous Operation
- Supports Serial Frames with 5, 6, 7, 8 or 9 Data Bits and 1 or 2 Stop Bits
- Fractional Baud Rate Generator
  - Generates the desired baud rate from any peripheral clock frequency without an external oscillator
- Error Detection and Correction:
  - 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
  - Collision detection
- Separate Interrupts for:
  - Transmit complete
  - Transmit data register empty
  - Receive complete
  - Error
- SPI Host Mode
- Multiprocessor Communication Mode with address detection
- Start-of-Frame Detection
- IRCOM Module for IrDA® Compliant Pulse Modulation/Demodulation
- LIN Host and Client Support
  - Autobaud
  - Break character detection
  - Automatic LIN header transmission
- RTS/CTS Handshake Support
- Protocol Converter
  - Hardware accelerated encoding/decoding of specialized protocols

25.2. Overview

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a fast and flexible serial communication peripheral. The USART supports several different modes of operation, accommodating multiple types of applications and communication devices. For example, the One-Wire Half-Duplex mode is useful in low pin-count applications. Communication is frame-based, and the frame format can be customized to support a wide range of standards.

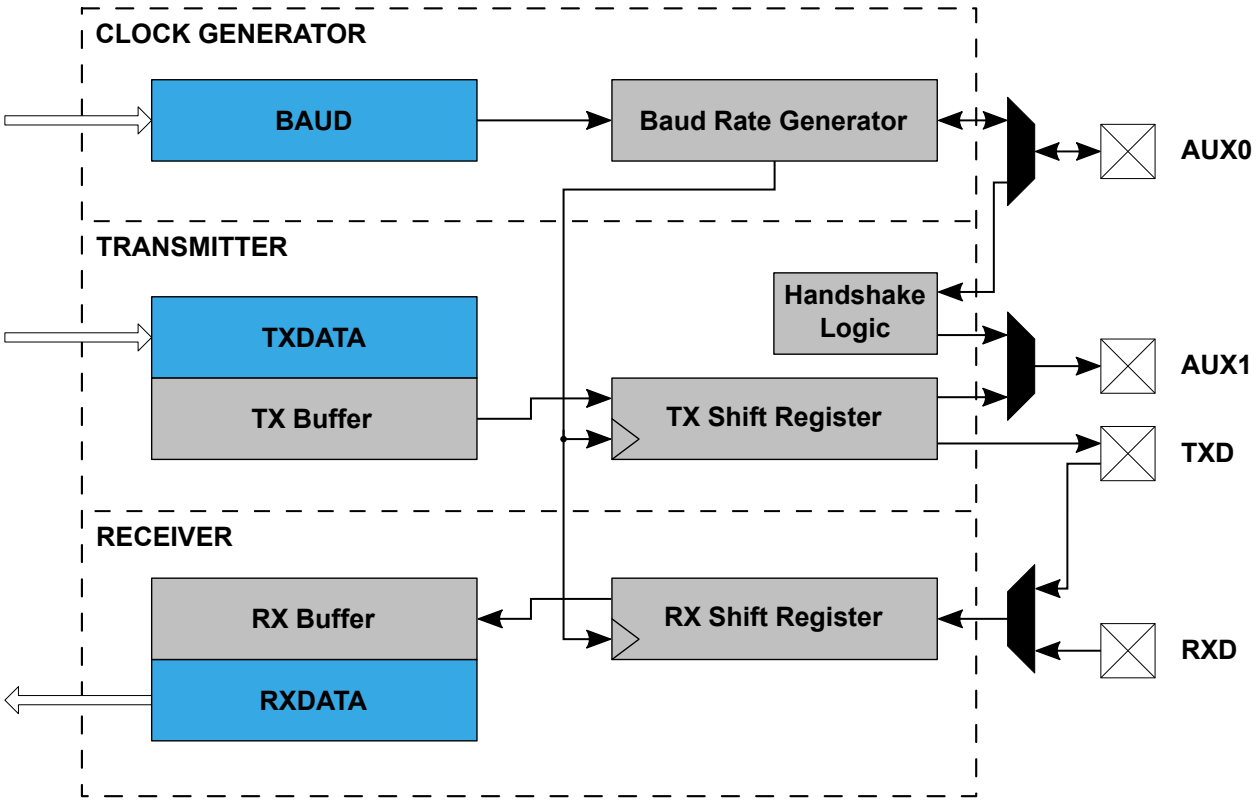
The USART is buffered in both directions, enabling continued data transmission without any delay between frames. Separate interrupts for receive and transmit completion allow fully interrupt-driven communication.

The transmitter consists of a two-level 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, and a separate error interrupt is available. Data and clock recovery units ensure robust synchronization and noise filtering during asynchronous data reception.

Interfaces such as RS-485 and LIN is supported in hardware. The USART also includes specialized hardware to accelerate encoding and decoding of various communication protocols, and includes sophisticated error handling mechanisms such as Collision Detection with automatic disabling of the transmitter.

25.2.1. Block Diagram

Figure 25-1. USART Block Diagram



25.2.2. Signal Description

Signal	Type	Description
AUX0	Output/input	Clock for synchronous operation (XCK), Clear to Send (CTS)
AUX1	Output	Transmit enable for RS-485 (XDIR), Request to Send (RTS)
TXD	Output/input	Transmitting line (and receiving line in One-Wire mode)
RXD	Input	Receiving line

25.3. Functional Description

25.3.1. Initialization

Full-Duplex Mode:

1. Set the baud rate (USARTn.BAUD).
2. Set the mode of operation (USARTn.CTRLA).
3. Enable the transmitter and the receiver (USARTn.CTRLB).
4. Configure the frame format (USARTn.CTRLC and USARTn.CTRLD).
5. Enable the USART (USARTn.CTRLA).

One-Wire Half-Duplex Mode:

1. Internally connect the TXD to the USART receiver (the LBME bit in the USARTn.CTRLE 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.CTRLE register).
4. Perform the initialization steps for Full-Duplex Mode as described above.

25.3.2. Operation

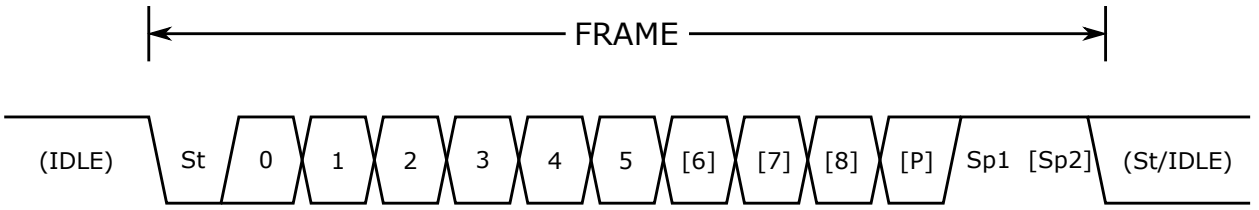
25.3.2.1. Frame Formats

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 25-2. Frame Formats



St Start bit, always low

(n) Data bits (0 to 8)

P Parity bit, may be odd or even

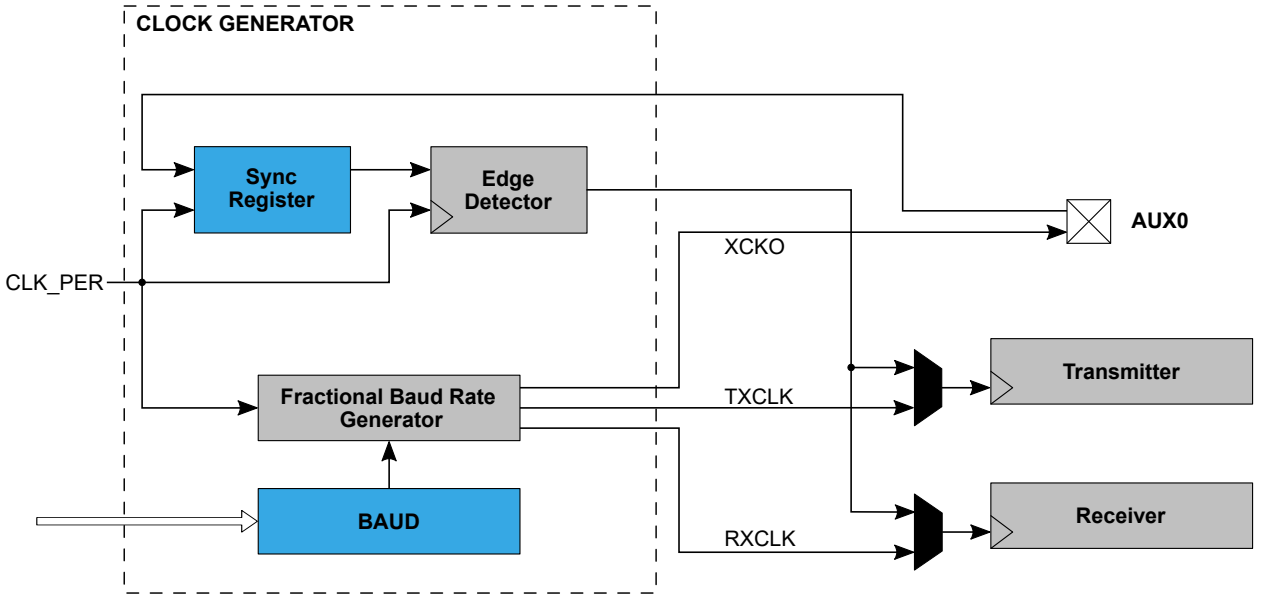
Sp Stop bit, always high

IDLE No transfer on the communication line (RxD or TxD). The Idle state is always high.

25.3.2.2. Clock Generation

The clock used for shifting and sampling data bits is generated either internally by the fractional baud rate generator or externally from the XCK input to the AUX0 pin.

Figure 25-3. Clock Generation Logic Block Diagram



25.3.2.2.1. 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 25-1. 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 (BAUD[15:6]) register 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 25-1. Equations for Calculating Baud Rate Register Setting

Operating Mode	Conditions	Baud Rate (Bits Per Seconds)	USART.BAUD Register Value Calculation
Asynchronous	$f_{BAUD} \leq \frac{f_{CLK\_PER}}{S}$ $USART.BAUD \geq 64$	$f_{BAUD} = \frac{64 \times f_{CLK\_PER}}{S \times BAUD}$	$BAUD = \frac{64 \times f_{CLK\_PER}}{S \times f_{BAUD}}$

Table 25-1. Equations for Calculating Baud Rate Register Setting (continued)

Operating Mode	Conditions	Baud Rate (Bits Per Seconds)	USART.BAUD Register Value Calculation
Synchronous Host	$f_{BAUD} \leq \frac{f_{CLK\_PER}}{S}$ $USART.BAUD \geq 64$	$f_{BAUD} = \frac{f_{CLK\_PER}}{S \times BAUD[15:6]}$	$BAUD[15:6] = \frac{f_{CLK\_PER}}{S \times f_{BAUD}}$

S is the number of samples per bit

- Asynchronous Normal mode: S = 16
- Asynchronous Double-Speed mode: S = 8
- Synchronous mode: S = 2

25.3.2.3. Data Transmission

The USART transmitter sends data by periodically driving the transmission line low. Data transmission is initiated by loading the Transmit Data (USARTn.TXDATAL and USARTn.TXDATAH) registers with the data to be sent. The data in the Transmit Data registers are moved to the TX Buffer once it is empty and onwards 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 Data registers or the TX Buffer, the Transmit Complete (TXC) interrupt flag in the Status (USARTn.INTFLAGS) register is set, and the interrupt is generated if it is enabled.

The Transmit Data registers can only be written when the Data Register Empty (DRE) interrupt flag in the Status (USARTn.INTFLAGS) register is set, indicating that the registers are empty and ready for new data.

When using frames with fewer than eight bits, the Most Significant bits (MSbs) written to the Transmit Data registers are ignored. When the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register is configured to 9-bit (low byte first), the Transmit Data Register Low Byte (TXDATAL) must be written before the Transmit Data Register High Byte (TXDATAH). When CHSIZE is configured to 9-bit (high byte first), TXDATAH must be written before TXDATAL.

25.3.2.3.1. Transmit Pipeline

Before writing data to the Transmit Data (USARTn.TXDATA) registers, it is important to check that the Data Register Empty (DRE) interrupt flag is set to '1' to avoid overwriting the buffer and losing data. Since the transmitter is double buffered, this can be handled in two different ways.

In cases where it is important that the data are transmitted back-to-back with no delay between the frames, the DRE interrupt flag should be polled or the corresponding interrupt should be enabled. When data are written immediately after DRE is set, the pipeline always has a new byte to transmit directly after the previous byte is shifted out. This ensures that each frame is transmitted with no delay between the stop bit of the previous frame and start bit of the next frame.

When delays between frames are acceptable and it is important that the CPU is not interrupted too often, the Transmit Complete (TXC) interrupt flag can be used. When the TXC flag is set to '1', the entire pipeline, including the shift register, is empty. At that point, software can alternate between checking DRE and writing data, and when the DRE is no longer '1', the pipeline is full and will subsequently transmit the bytes written.

Note: It may take up to one baud clock cycle for data to be shifted from the transmit buffer to the transmit shift register. This implies that two bytes may be written into an empty buffer back-to-back, while the third byte can be written only after the first byte has been loaded into the shift register.

25.3.2.3.2. Disabling the Transmitter

When disabling the transmitter, the operation does not take effect until ongoing and pending transmissions are completed. That is, when the transmit shift register, Transmit Data (USARTn.TXDATAL and USARTn.TXDATAH) registers, and TX 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 can then be used as a normal I/O pin with no port override from the USART.

25.3.2.4. Data Reception

The USART receiver samples the reception line to detect and interpret the received data. Therefore, the pin direction must be configured as an input by writing a '0' to the corresponding bit in the Data Direction (PORTx.DIR) register.

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 (RXC) interrupt flag in the Status (USARTn.INTFLAGS) register is then set, and an interrupt is generated if enabled.

The RXDATA registers are part of the double-buffered RX buffer that can be read by application software when the RXC flag is set. If only one frame has been received, the data and status bits for that frame are pushed directly to the RXDATA registers. If two frames are present in the RX buffer, the RXDATA registers contain the data for the frame that was received first.

The buffer shifts out data either when RXDATAL or RXDATAH is read, depending on the configuration. The register which does not lead to data being shifted must be read first to be able to read both bytes before shifting. When the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register is configured to 9-bit (low byte first), reading RXDATAH shifts the receive buffer. Otherwise, reading RXDATAL shifts the buffer.

25.3.2.4.1. Receiver Error Flags

The USART receiver features error detection mechanisms that uncover any 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 same flags are also present in the Status (USARTn.STATUS) register, but the flags here are set if any of the incoming bytes have an error. To know exactly which frame contains the error the Receiver Data High Byte (USARTn.RXDATAH) must be read.

25.3.2.4.2. Disabling the Receiver

When the receiver is disables, the operation takes effect immediately. The receiver buffer is flushed, and any data from ongoing receptions will be lost.

25.3.2.4.3. Flushing the Receive Buffer

If the RX buffer needs to be flushed during normal operation, repeatedly read the DATA locations (USARTn.RXDATAH and USARTn.RXDATAL registers) until the Receive Complete (RXC) interrupt flag in the Status (USARTn.RXDATAH) register is cleared.

25.3.3. Communication Modes

The USART is a flexible peripheral that supports multiple communication protocols. The available modes of operation are divided into two groups: Synchronous and asynchronous communication.

Synchronous communication relies on one device on the bus acting as the host, providing a clock signal to other devices through the XCK signal on the AUX0 pin. All devices use this common clock for both transmission and reception, elliminating the need for additional synchronization mechanism.

The device can be configured to operate as either a host or a client on the synchronous bus.

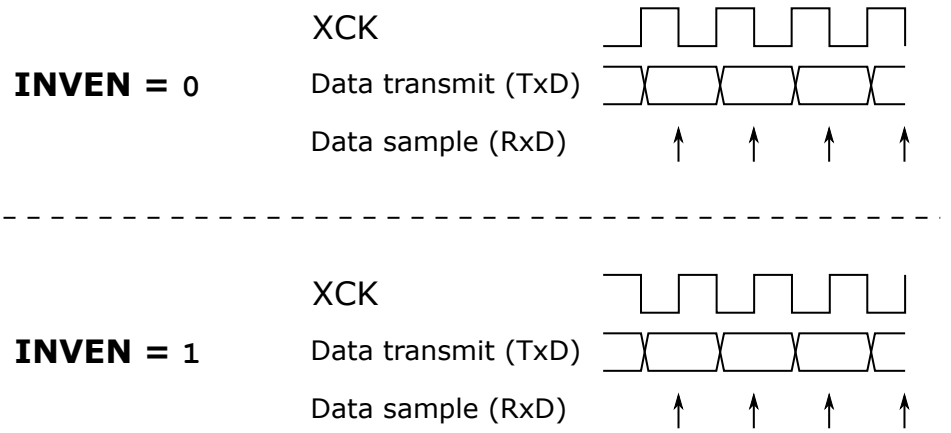
Asynchronous communication does not use a common clock signal. Instead, it relies on all communicating devices being configured with the same baud rate. When receiving a transmission, hardware synchronization mechanisms are used to align the incoming transmission with the receiving device peripheral clock.

25.3.3.1. Synchronous Operation

25.3.3.1.1. Clock Operation

The CMODE bit in the Control A (USARTn.CTRLA) register determines whether the transmission clock functions an input (Client mode) or an output (Host mode). The data input (on RXD) is sampled at the XCK clock edge opposite to the edge on which data are transmitted (on TXD), as illustrated in the figure below.

Figure 25-4. Synchronous Mode XCK Timing



The I/O pin can be inverted by setting the Inverted I/O Enable (INVEN) bit to '1' in the Pin n Control register of the port peripheral (PORTx.PINnCTRL). This setting configures which XCK clock edge is used for sampling RxD and transmitting on TxD. 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 on 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 on the rising XCK clock edge.

25.3.3.1.2. External Clock Limitations

When the USART is configured in Synchronous Client mode, the XCK signal must be provided externally by the host device. Since the clock is provided externally, configuring the BAUD register does not affect the transfer speed. For successful clock recovery, the clock signal must be sampled at least twice for each rising and falling edge. Therefore, the maximum XCK speed in Synchronous Operation mode (f_{Client_XCK}) is limited by:

$$f_{Client\_XCK} < \frac{f_{CLK\_PER}}{4}$$

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 at least twice for each edge.

25.3.3.1.3. USART in SPI Host 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 work as the host device. The SPI is a four-wire interface that enables a host device to communicate with one or multiple clients.

Frame Formats

In SPI Host mode, the USART serial frame 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 (DORD) bit in the Control C (USARTn.CTRLC) register.

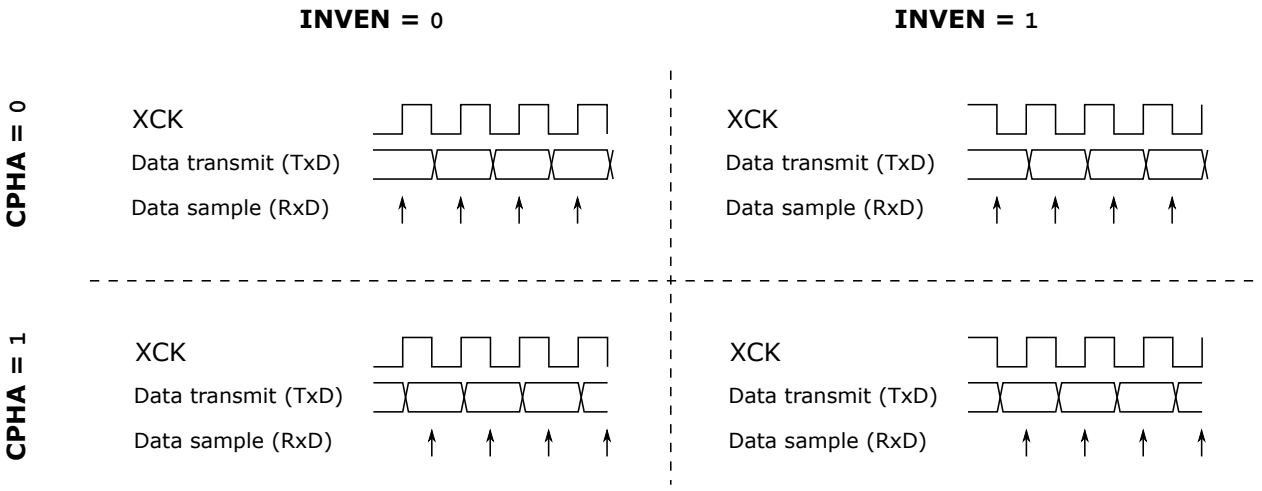
SPI does not use Start, Stop, or Parity bits, so the transmission frame only consist of the data bits.

Clock Generation

As the host device in a synchronous communication interface, the USART in SPI Host mode must generate the interface clock to be shared with the client devices. The interface clock is generated using the fractional Baud Rate Generator, as described in 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 samples each bit in the middle of the transmitter's hold period, as illustrated in the figure below. The timing scheme can be configured using the Inverted I/O Enable (INVEN) bit in the PORTx.PINnCTRL register and the Clock Phase (CPHA) bit in the USARTn.CTRLC register.

Figure 25-5. Data Transfer Timing Diagrams



The table below provides additional explanation for the figure above.

Table 25-2. Functionality of the INVEN and CPHA Bits

INVEN	CPHA	Leading Edge (1)	Trailing Edge (1)
0	0	Rising, sample	Falling, transmit
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, while the trailing edge is the last clock edge of a clock cycle.

Data Transmission

Data transmission in SPI Host mode is functionally identical to the general USART operation, as described in the *Operation* section. The transmitter interrupt flags and corresponding USART interrupts are also identical. For more details, see the Data Transmission section.

Data Reception

Data reception in SPI Host mode is identical in function to general USART operation as described in the *Operation* section. The receiver interrupt flags and corresponding USART interrupts are also identical, except for the receiver error flags that are not in use and always read as '0'. For more details, see the Data Reception section.

USART in SPI Host Mode vs. SPI

The USART in SPI Host mode is fully compatible with a stand-alone SPI peripheral as their data frame and timing configurations are identical. However, some SPI-specific special features are not supported when using the USART in SPI Host mode:

- Write Collision Flag Protection
- Double-Speed mode
- Multi-Host support

A comparison of the pins used by the USART in SPI Host mode and by a stand-alone SPI peripheral is shown in the table below.

Table 25-3. Comparison of USART in SPI Host Mode and SPI Pins

USART	SPI	Comment
TXD	MOSI	Host out
RXD	MISO	Host in
XCK	SCK	Functionally identical
-	$\overline{SS}$	Not supported by USART in SPI Host mode(1)

Note:

1. For a stand-alone SPI peripheral, this pin is used for the Multi-Host function or as a dedicated Client Select pin. The Multi-Host function is not available with the USART in SPI Host mode, and there is no dedicated Client Select pin.

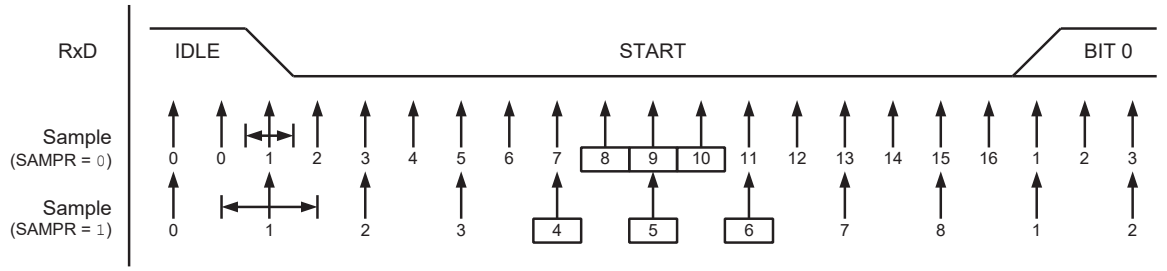
25.3.3.2. Asynchronous Operation

25.3.3.2.1. Clock Recovery

Since there is no common clock signal in Asynchronous mode, each communicating device generates separate clock signals. These clock signals must be configured to run at the same baud rate for communication to occur. 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 that synchronizes 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, showing the timing scheme for both Normal and Double-Speed mode. These modes are selected by configuring the Sample Rate (SAMPR) bit in the Control C (USARTn.CTRLC) register. 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 Double-Speed Operation for more details). The horizontal arrows indicate the maximum synchronization error. Note that the maximum synchronization error is larger in Double-Speed mode.

Figure 25-6. Start Bit Sampling

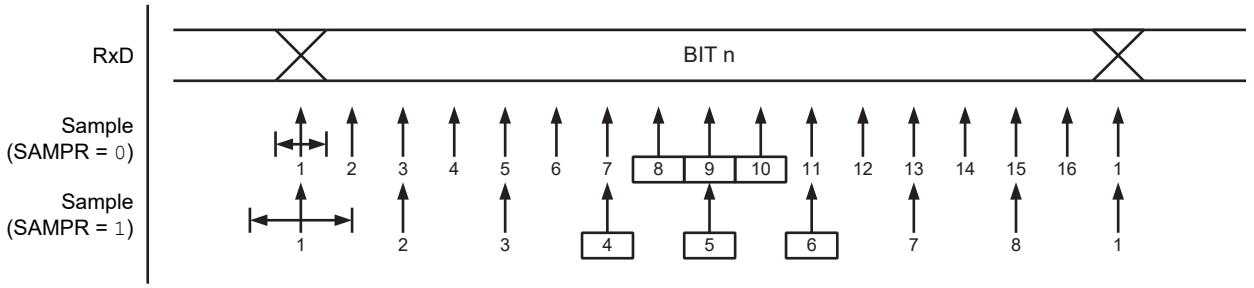


When the clock recovery logic detects a falling edge from the Idle (high) state to the Start bit (low), the Start bit detection sequence is initiated. In the figure above, sample 1 denotes the first sample reading '0'. The clock recovery logic then uses three subsequent samples (samples 8, 9, and 10 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.

25.3.3.2.2. Data Recovery

Similar to 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 illustrates the sampling process for reading a bit in a received frame.

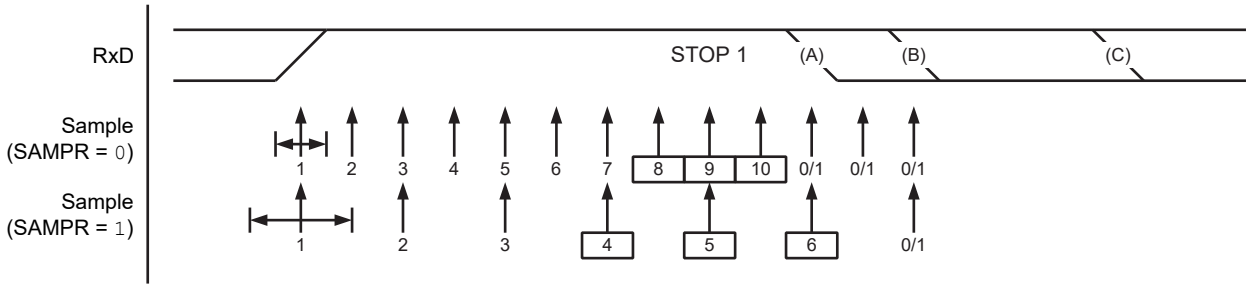
Figure 25-7. Sampling of Data and Parity Bits



A majority voting technique is, similar to that used in clock recovery, is applied to the three center samples to determine the logic level of each received bit. This process is repeated for each bit until a complete frame is received.

The data recovery unit receives only the first Stop bit and ignores any additional Stop bits. If the sampled Stop bit is read '0', the Frame Error flag will be set. The figure below illustrates the sampling of a Stop bit and shows the earliest possible beginning of the next frame's Start bit.

Figure 25-8. 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 appear at point (A) in the figure above. For Double-Speed mode, the first low-level sample must be delayed to point (B), which is the first sample after the majority vote samples. Point (C) marks a Stop bit of full length at the nominal baud rate.

25.3.3.2.3. Error Tolerance

The speed of the internally generated baud rate and the externally received data rate has to be identical, but, due to natural clock source error, this is usually not the case. The USART is tolerant of such errors, and the limits of this tolerance make up what is sometimes known as the Operational Range.

The following tables show the operational range of the USART, indicating 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 25-4. 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

Notes:

- D: The sum of the character size and parity size (D = 5 to 10 bits)
- R_{SLOW}: The ratio of the slowest incoming data rate that can be accepted relative to the receiver baud rate
- R_{FAST}: The ratio of the fastest incoming data rate that can be accepted relative to the receiver baud rate

Table 25-5. 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

Notes:

- D: The sum of the character size and parity size (D = 5 to 10 bits)
- R_{SLOW}: The ratio of the slowest incoming data rate that can be accepted relative to the receiver baud rate
- R_{FAST}: The ratio of the fastest incoming data rate that can be accepted relative to the receiver baud rate

The recommended 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 between 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 the 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 relative to the receiver baud rate
- R_{FAST}: The ratio of the fastest incoming data rate that can be accepted relative to the receiver baud rate

25.3.3.2.4. Double-Speed Operation

Double-speed operation allows for higher baud rates in asynchronous mode, even with lower peripheral clock frequencies. This mode is enabled by setting the Sample Rate (SAMPR) bit in the Control C (USARTn.CTRLC) register to '1'.

When enabled, the baud rate for a given asynchronous baud rate setting will be doubled, as shown in the equations in The Fractional Baud Rate Generator. In this mode, the receiver uses 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. For more information, see Error Tolerance.

25.3.4. Additional Features

25.3.4.1. 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

25.3.4.1.1. One-Wire Mode

One-Wire mode is enabled by setting the Loop-Back Mode Enable (LBME) bit in the USARTn.CTRLE register. This will enable an internal connection between the TXD pin and the USART receiver, allowing the TXD pin to function as a combined TxD/RxD line. The RXD pin will be disconnected from the USART receiver and can be controlled by a different peripheral.

In One-Wire mode, multiple devices can control the TxD/RxD line at the same time. In the case where one device drives the pin to a logical high level (V_{DD}), and another device pulls the line low (GND), a short circuit will occur. To prevent this, the USART features an Open-Drain mode (enabled by the ODME bit in the USARTn.CTRLE register), which restricts the transmitter from driving a pin to a logical high level, thereby constraining it to only be able to pull it low. When combined with the internal pull-up feature (enabled by the PULLUPEN bit in the PORTx.PINnCTRL register), the line is held high by a pull-up resistor, allowing any device to pull it low. If the line is pulled low, the current from V_{DD} to GND is limited by the pull-up resistor.

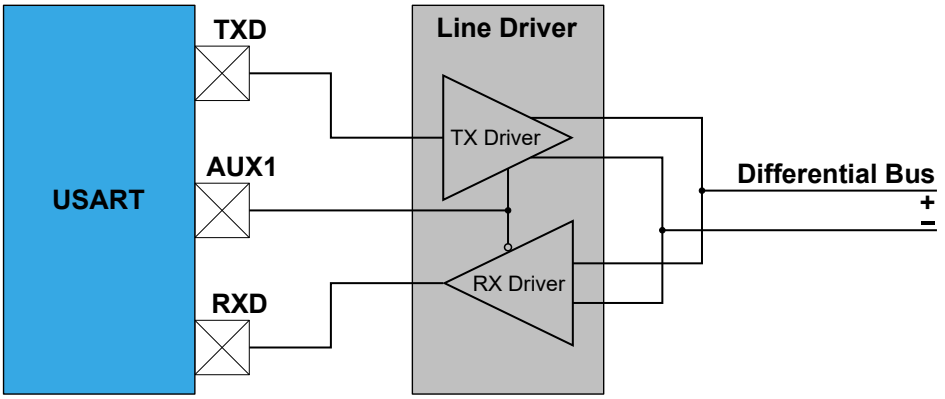
When the USART transmits on the TxD/RxD line, it also receives its own data. Overlapping transmissions are automatically detected, as detailed in the Collision Detection section.

25.3.4.1.2. 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, which makes communication robust against noise. RS-485 is enabled by setting the Communication Signals (CSIG) bit field in the Control A (USARTn.CTRLA) register to RS485.

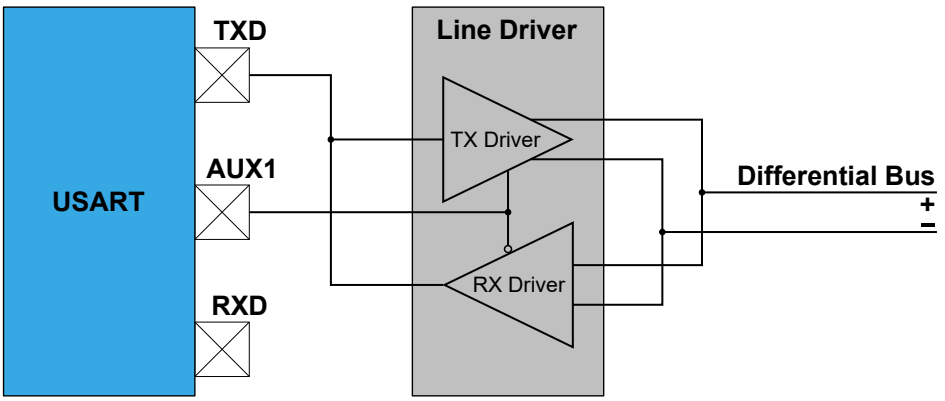
The RS-485 mode supports external line driver devices that convert a single USART transmission into corresponding differential pair signals. It implements automatic control of the AUX1 pin, which can be used to enable transmission or reception for the line driver device. The USART automatically drives the XDIR pin high while the USART is transmitting and pulls it low when transmission is complete. An example of circuit is shown in the figure below.

Figure 25-9. RS-485 Bus Connection



RS-485 mode is compatible with One-Wire mode. One-Wire mode enables an internal connection between the TXD pin and the USART receiver, allowing the TXD pin to function as a combined TxD/RxD line. The RXD pin will be disconnected from the USART receiver and may be controlled by another peripheral. An example of such a circuit is shown in the figure below.

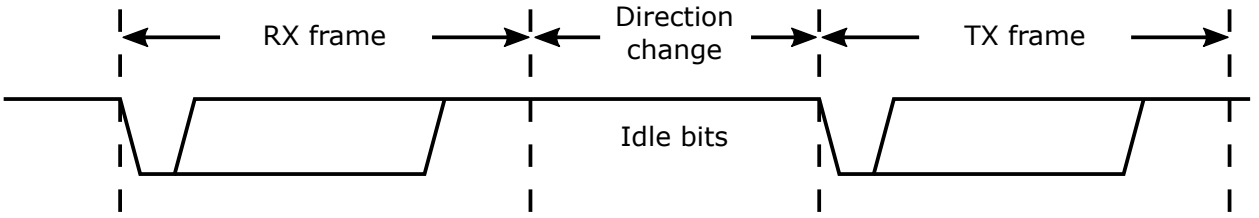
Figure 25-10. RS-485 with Loop-Back Mode Connection



Guard Time

To ensure enough turn-around time for line drivers when operating in RS485 mode, the USART 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 the 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 (GUARD) bit field in the Control F (USARTn.CTRLF) register. The duration of each idle bit is given by the baud rate used by the current transmission.

Figure 25-11. Guard Time

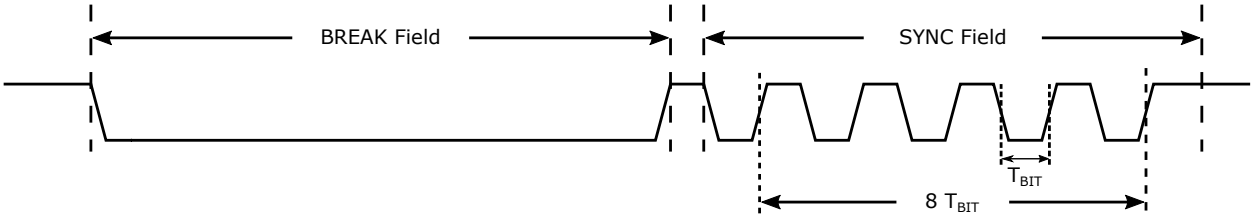


25.3.4.2. Auto-Baud

The auto-baud feature lets allows the USART to autonomously configure its BAUD register based on input from a communication device, which enables the communication with multiple devices operating at different baud rates or compensate for any baud rate drift. The USART peripheral features two auto-baud modes: Generic Auto-Baud mode and LIN Constrained Auto-Baud mode.

Both auto-baud modes use an auto-baud frame to synchronize the BAUD registers, as illustrated in the figure below.

Figure 25-12. Auto-Baud Timing



The break field is detected when 11 or more consecutive low cycles are sampled, signaling to the USART that a synchronization field will follow. 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. Once all eight bits are sampled, the counter is stopped. The resulting counter value is in effect the new BAUD register value.

When the Frame Format (FORM) bit field in the Control C (USARTn.CTRLC) register is set to auto-baud mode (AUTOBAUD), the Generic Auto-Baud mode is enabled. In this mode, the Wait For Break (WFBRK) bit in the Command (USARTn.COMMAND) register can optionally be set to allow detection of a break field of any length, including those 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.

25.3.4.3. LIN Constrained Auto-Baud

The USART features hardware to support LIN Client and LIN Host functionality.

25.3.4.3.1. LIN Client

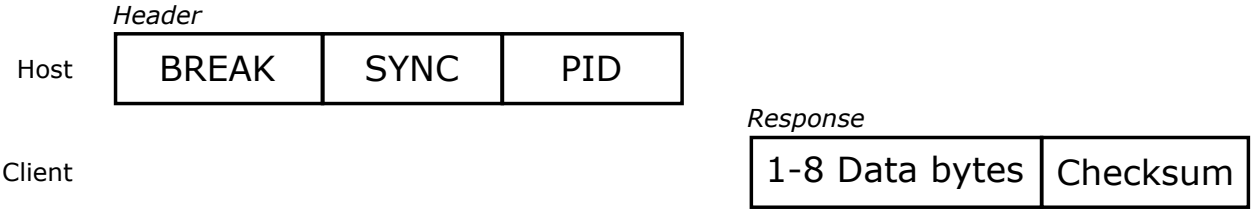
When the Frame Format (FORM) bit field is configured to LIN Client (LINCLIENT) mode, the USART can automatically detect incoming LIN Header transmissions. The LIN Header consists of a BREAK field followed by a SYNC field, similar to the Generic Auto-baud mode.

Upon receiving a BREAK field, the LIN Client expects a SYNC frame. The BREAK field must hold the transmission line low for at least 12 clock cycles, and the SYNC field must read 0x55. Additionally, the time between falling edges must adhere to the configuration of the Auto-baud Window size (ABW) bit field in the Control G (USARTn.CTRLG) register. If both the BREAK and SYNC fields fulfill these requirements, the BAUD register is updated accordingly. If not, the Inconsistent Sync Field (ISF) bit in the Status (USARTn.STATUS) register is set.

25.3.4.3.2. LIN Host

LIN Host mode is enabled by configuring the Frame Format (FORM) bit field to LIN Host (LINHOST) mode. In this mode, the USART can transmit LIN headers with a BREAK, SYNC and Protected Identifier (PID) field. The header can be controlled by software, or fully automated by hardware.

Figure 25-13. LIN Frame Format



Software-controlled header transmission is performed by first using the LIN hardware to transmit a BREAK field, and then to manually transmit the SYNC and PID field. To transmit the header using software, the following sequence must be followed:

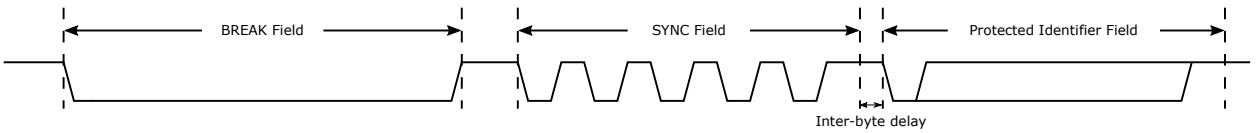
1. Set the LIN Command (LINCMD) bit field in the Command (USARTn.COMMAND) register to BREAK (`0x01`).
2. Write any value to the Transmit Data (USARTn.TXDATAL) register to trigger the transmission of the BREAK field.
3. Write `0x55` to the Transmit Data (USARTn.TXDATAL) register to transmit the SYNC field.
4. Write the PID value (including parity bits) to the Transmit Data (USARTn.TXDATAL) register.

Automatic header transmission is performed by configuring the mode and writing the identifier to the data register. The BREAK, SYNC and PID are all transmitted automatically. The PID is generated by calculating the parity bits from the identifier. To transmit the header automatically, the following sequence must be followed:

1. Set the LIN Command (LINCMD) bit field in the Command (USARTn.COMMAND) register to HEADER (`0x02`).
2. Write the identifier value to the Transmit Data (USARTn.TXDATAL) register.

In LIN Host mode, the length of the BREAK field is programmable using the Break Length (BRKLEN) bit field in the Control F (USARTn.CTRLF) register. When the automatic header command is used, the delay between the BREAK and SYNC fields is fixed to two bit times. When software controlled transmission is used, software controls the delay between the BREAK and SYNC fields.

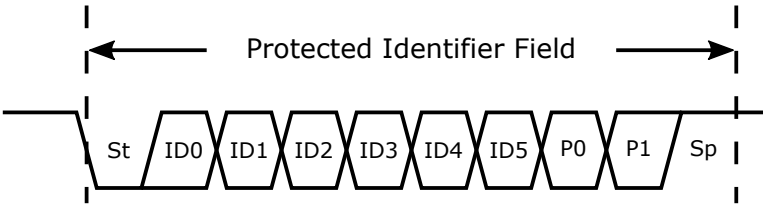
Figure 25-14. LIN Header



25.3.4.3.3. Protected Identifier Field

The Protected Identifier (PID) Field consists of five data bits and two parity bits, in addition to the start bit and stop bit.

Figure 25-15. Protected Identifier Field



The parity bits are calculated using the following equations.

$P0 = ID0 \text{ XOR } ID1 \text{ XOR } ID2 \text{ XOR } ID4$

$P1 = \text{ NOT } (ID1 \text{ XOR } ID3 \text{ XOR } ID4 \text{ XOR } ID5)$

When the Frame Format (FORM) bit field in the Control C (USARTn.CTRLC) register is set to LIN Client mode (LINCLIENT), the USART will automatically performs a parity check on the PID field. If a parity error is detected, the Parity Error (PERR) flag in the Status (USARTn.STATUS) register is set. No parity check is performed on any subsequent data bytes received within the same LIN frame.

When the USART is configured as a LIN host (with the FORM bit field in the USARTn.CTRLC register set to LINHOST) and the LIN Command (LINCMD) bit field in the Command (USARTn.COMMAND) register is set to HEADER, the PID is automatically calculated from the identifier placed in the Transmit Data (USARTn.TXDATAL) register before transmission. If LINCMD is set to BREAK, the PID is not automatically calculated and must be calculated manually and placed in the Transmit Data (USARTn.TXDATAL) register together with the identifier.

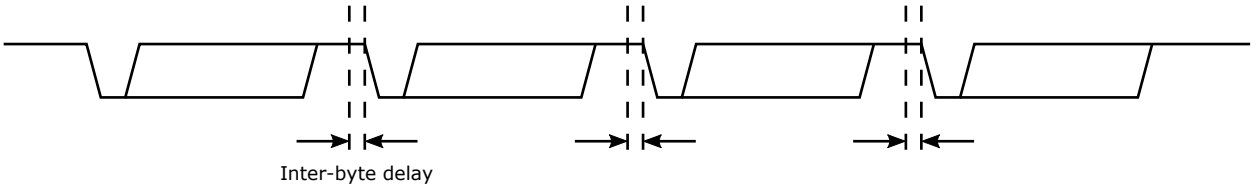
25.3.4.4. Inter-Byte Delay

During multi-byte transfers, the output data is transmitted in a continuous stream. In some applications, the data may be sent too quickly for the receiver, and there might not be enough time for the data to be processed before the next start bit arrives. The inter-byte delay mitigates this by inserting a fixed number of idle bits between bytes in multi-byte transfers.

The inter-byte delay feature can be enabled by writing to the Inter-Byte Delay (INTDLY) bit field in the Control F (USARTn.CTRLF) register.

In LIN Host mode, the inter-byte delay is also inserted after the SYNC field.

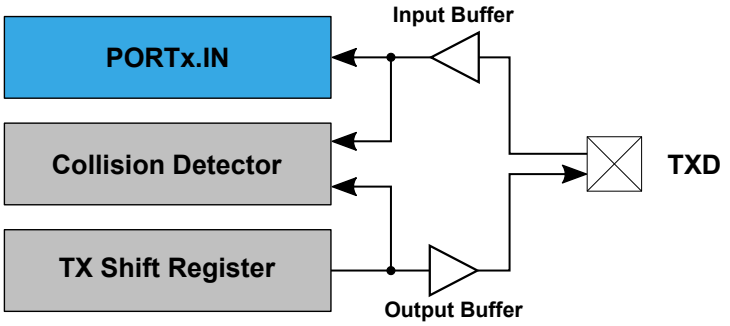
Figure 25-16. Inter-Byte Delay



25.3.4.5. Collision Detection

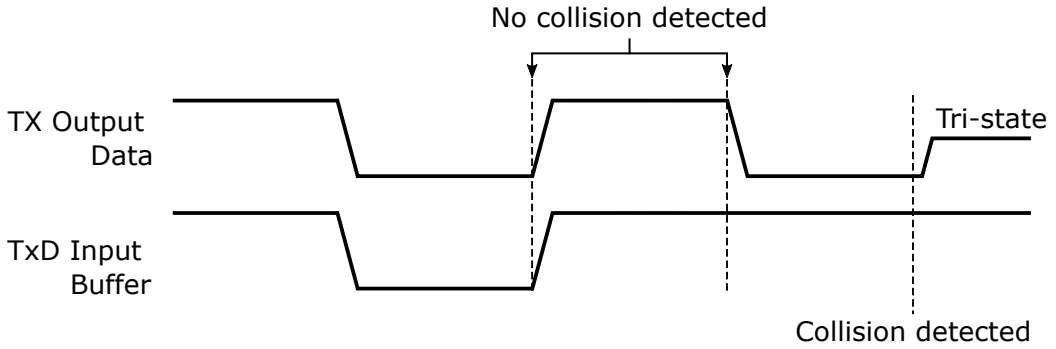
Transmit collisions can be detected by hardware by writing the Collision Detection Enable (COLDEN) bit to '1' in the Control E (USARTn.CTRLE) register. Collision detection is performed for each transmitted bit by comparing the expected transmit value with the value on the TxD pin input buffer, as illustrated in the figure below.

Figure 25-17. Collision Detection Block Diagram



The figure below shows an example of how the Collision Detection feature tristates the TxD pin when a collision occurs. Collision detection is performed on all transmitted data, including the Start bit, Stop bit(s), and Parity bit.

Figure 25-18. Collision Detected Example



When a collision is detected, the USART automatically takes the following actions:

- The current transfer is aborted
- The transmitter is disabled (TXEN in USARTn.CTRLB is set to '0')
  - This forces TxD to be tri-stated
- The transmit buffer is flushed, and the Transmit Complete (TXC) interrupt flag in the Interrupt Flags (USARTn.INTFLAGS) register is set
- The Collision Detected (COLL) bit in the Status (USARTn.STATUS) register is set to '1', and the Error (ERROR) interrupt flag in the Interrupt Flags (USARTn.INTFLAGS) register is also set.

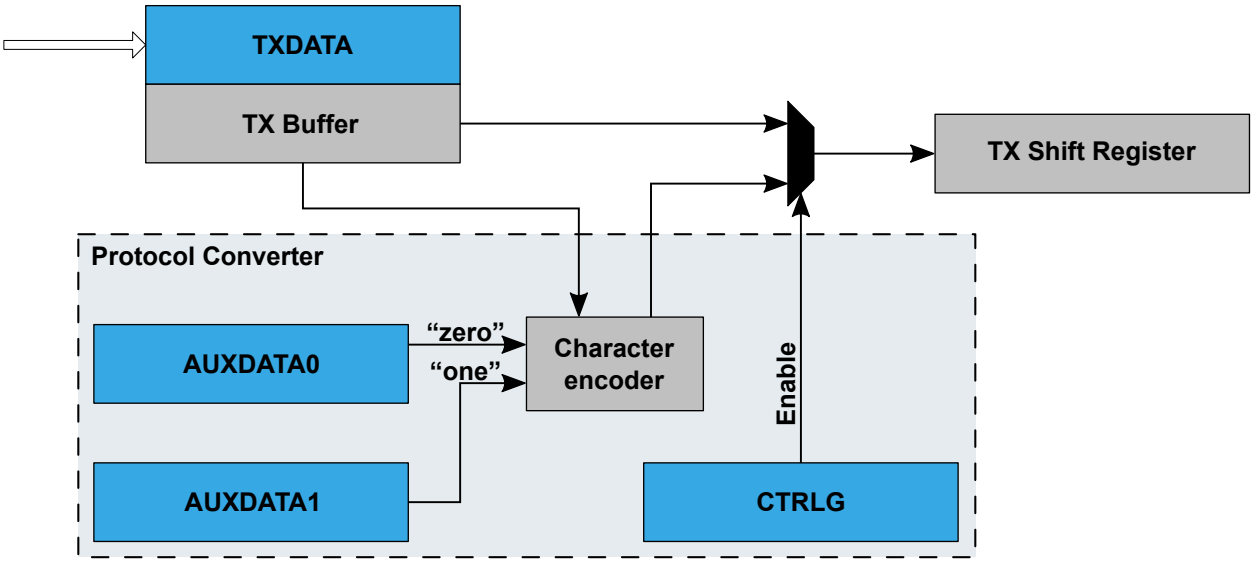
After a collision, software must manually enable the transmitter again before continuing.

25.3.4.6. Protocol Converter

The Protocol Converter is a hardware accelerator that supports encoding and decoding protocols based on UART or SPI, but use one frame to transfer a single bit. This approach is typically used for one-wire protocols, either unidirectional or where there is large baud rate difference between the host and client.

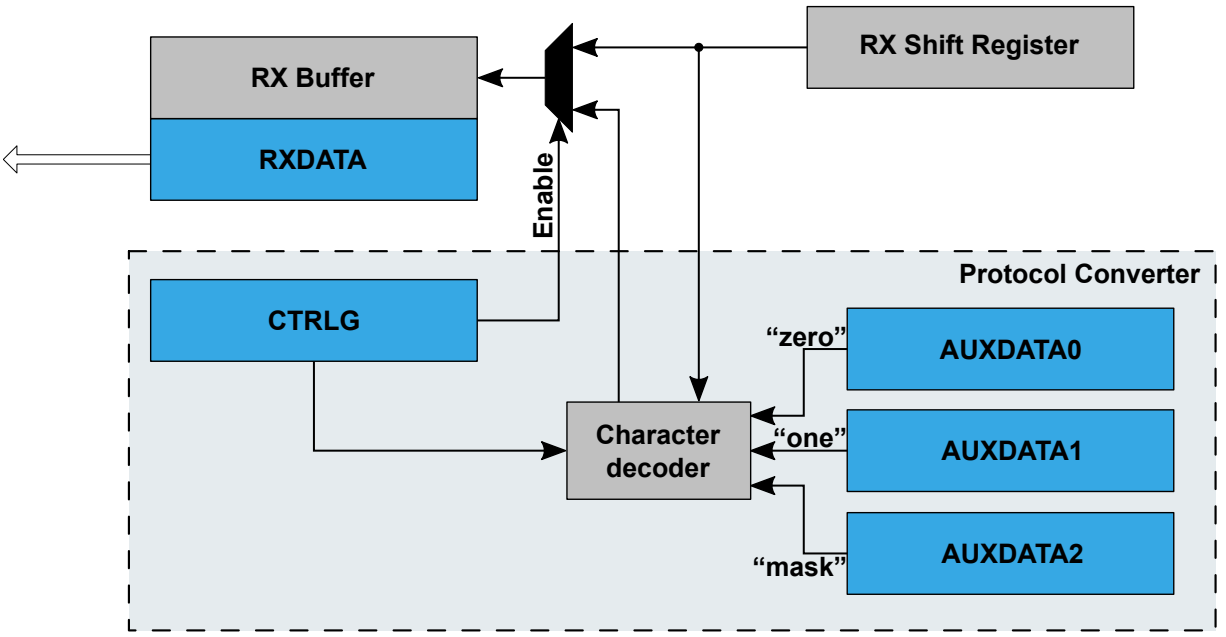
For encoding, the bits in the Transmit Data (TXDATA) register are shifted out one at a time and used to select the frame to be transmitted.

Figure 25-19. Protocol Encoding During Transmit



For decoding, the data from the Receive shift register are decoded and the result is shifted into the Receive Data (RXDATA) register one bit at a time. The Receive Complete (RXC) interrupt flag is set after reception of eight characters.

Figure 25-20. Protocol Decoding During Receive



The table below shows the available decoding schemes, which are selected by configuring the Protocol Converter (PROTCONV) bit field in the Control G (USARTn.CTRLG) register. For the FIXED decoding scheme, the Frame Error (FERR) flags in the Status (USARTn.STATUS) and Receiver Data High (USARTn.RXDATAH) registers is set if the received frame corresponds to neither a "one character" nor a "zero character". For all other decoding schemes, any frame is decoded as a "zero character" except frames that meet the requirements of a "one character".

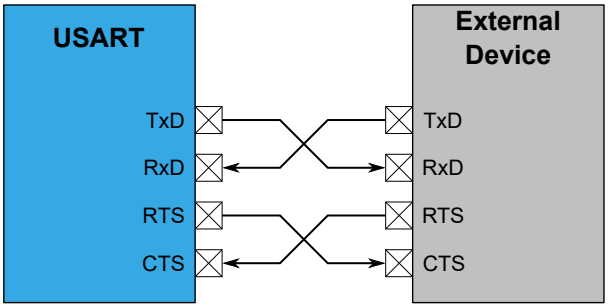
Table 25-6. Protocol Conversion Character Encoding

Decoding Scheme	Decoding "One character"	Decoding "Zero character"
FIXED	DATA = AUXDATA1	DATA = AUXDATA0
MASKED	(DATA & AUXDATA2) = (AUXDATA1 & AUXDATA2)	Not "One Character"
DUAL	(DATA = AUXDATA1) or (DATA = AUXDATA2)	Not "One Character"

25.3.4.7. Hardware Handshake

The USART features an out-of-band hardware handshaking mechanism, separate from the communication lines, to control transmission flow.

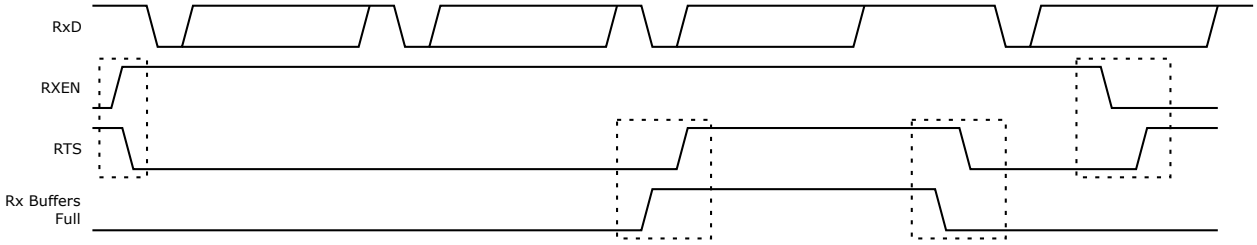
Figure 25-21. Hardware Handshake Connection



Hardware handshake is enabled by configuring the Communication Signals (CSIG) field in the Control A (USARTn.CTRLA) register to `0x02` (HANDSHAKE). Communication Mode (CMODE) in CTRLA must be configured to `0x00` (ASYNCHRONOUS).

The receiver drives its RTS pin high when it is unable to receive data, either because the receive buffers are full, or the receiver is disabled. This indicates to the remote device that it must stop transmitting after the ongoing transmission is complete, avoiding buffer overflow or lost data. The incoming data is stored in the shift register until the receive buffer is no longer full.

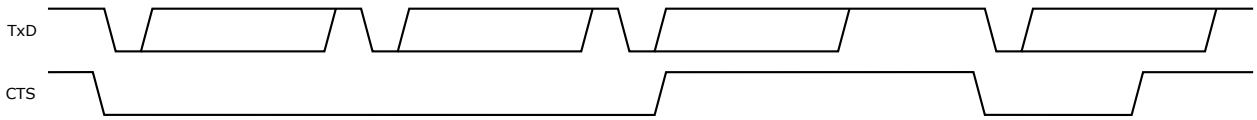
Figure 25-22. Receiver Behavior with Hardware Handshake



The transmitter automatically stops transmitting a new frame if the CTS pin is read as high. The current CTS level can be checked in the Status (USARTn.STATUS) register. When CTS changes from low to high, the transmitter will finish the ongoing transmission and then stop.

The figure below illustrates the transmitter behavior based on the state of the CTS pin.

Figure 25-23. Transmitter Behavior with Hardware Handshake



25.3.4.8. Parity

Parity bits can be used by the USART to verify 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 setting the Parity Mode (PMODE) bit field in the USARTn.CTRLD register. If even parity is selected, the Parity bit is set to '1' if the number of data bits with a value of '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 a value of '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.

25.3.4.9. 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 concerning the oscillator start-up time. The oscillators start-up time varies with supply voltage and temperature. For more information on oscillator start-up time characteristics, refer to the *Electrical Characteristics* section.

If a false Start bit is detected and no other wake-up source has been triggered, the device will return to Standby sleep mode.

The Start-of-Frame detection feature operates only in Asynchronous mode. It is enabled by setting the Start-of-Frame Detection Enable (SFDEN) bit in the Control E (USARTn.CTRLE) register. If a Start bit is detected while the device is in Standby sleep mode, the USART Receive Start (RXS) interrupt flag is set.

The USART Receive Complete (RXC) interrupt flag and the RXS bit share the same interrupt line, but each have dedicated interrupt settings. The table below shows the USART Start Frame Detection modes, depending on the interrupt setting.

Table 25-7. 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 frame reception. If interrupts are disabled and buffer overflow is ignored, all incoming frames will be lost.
1	Disabled	Enabled	System and all clocks are awakened when a Receive Complete interrupt occurs
1	Enabled	x	System and all clocks are awakened when a Start bit is detected

Note: The `SLEEP` instruction will not shut down the oscillator if communication is ongoing.

25.3.4.10. 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 E (USARTn.CTRLE) register. 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 client MCU has been addressed, it will receive the following data frames as usual, while the other client MCUs will ignore the frames until another address frame is received.

25.3.4.10.1. Using Multiprocessor Communication

Use the following procedure to exchange data in Multiprocessor Communication mode (MPCM):

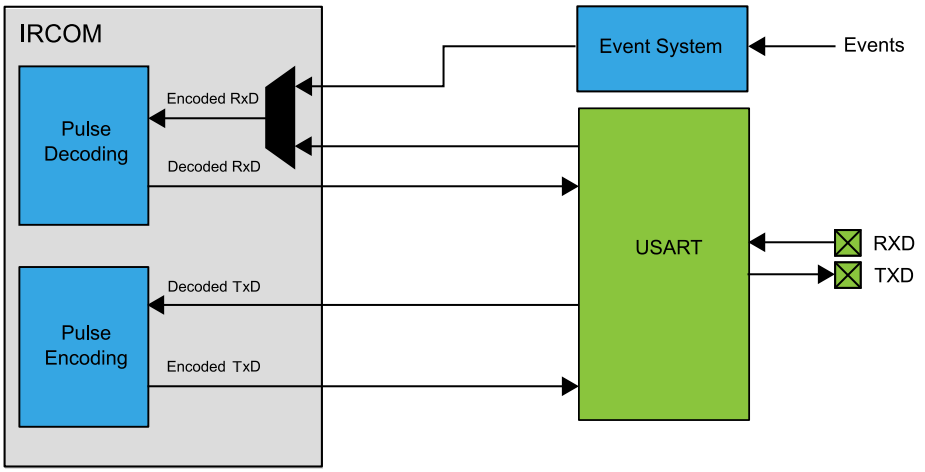
1. All client MCUs are in Multiprocessor Communication mode.
2. The host MCU sends an address frame, and all clients receive and read this frame.
3. Each client MCU determines if it has been selected.
4. The addressed MCU will disable MPCM and receive all data frames. The other client MCUs will ignore 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 host.

The process then repeats from step 2.

25.3.4.11. 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 25-24. Block Diagram



The USART is set in IRCOM mode by writing a '1' to the ENC bit in the Control G (USARTn.CTRLG) register. The data on the TXD/RXD pins are the inverted values of the transmitted/received infrared pulse. An event input can be used in place of the RX pin, allowing IRCOM to receive input from I/O pins or other sources instead of the corresonding RXD pin, which 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.

Double-Speed mode cannot be used for the USART when IRCOM mode is enabled.

25.3.5. Events

The USART can generate the following events:

Table 25-8. USART Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
USARTn	XCK	The clock signal in Synchronous Host mode and SPI Host mode	Level	CLK_PER	One XCK period

The conditions for generating an event are identical to those that will raise the corresponding flag in the Interrupt Flags (USARTn.INTFLAGS) register.

The USART has one event user for detecting and acting upon input events. The table below describes the event user and the associated functionality.

Table 25-9. USART Event Users and Available Event Actions

User Name		Description	Input Detection	Async/Sync
Peripheral	Event			
USARTn	RX	Receive data	Level	Async/Sync

25.3.6. Interrupts

Table 25-10. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
USARTn_ERROR	ERROR	• Auto-Baud Error/Invalid Sync Field (ISF) • Parity Error (PERR) • Frame Error (FERR) • Buffer Overflow (BUFOVF) • Transmit Collision Detected (COLL)	
USARTn_RXC	RXC	• There are unread data in the receive buffer (RXC) • Receive of Start-of-Frame detected (RXS) • Received Valid Break and Synchronization (RXBRK)	
USARTn_DRE	DRE	• The transmit buffer is empty/ready to receive new data (DRE) • Clear to Send Input Change (CTSIC)	
USARTn_TXC	TXC	The entire frame in the transmit shift register has been shifted out and there are no new data in the transmit buffer (TXC)	

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		CSIG[1:0]				CMODE[1:0]		ENABLE
0x01	CTRLB	7:0							RXEN	TXEN
0x02	CTRLC	7:0		DORD	CPHA		SAMPR	FORM[2:0]		
0x03	CTRLD	7:0			PMODE[1:0]		SBMODE	CHSIZE[2:0]		
0x04	CTRLE	7:0				COLDEN	SFDEN	ODME	LBME	MPCM
0x05	CTRLF	7:0	INTDLY[3:0]				GUARD[1:0]		BRKLEN[1:0]	
0x06	CTRLG	7:0	ABW[1:0]			ENC		PROTCONV[2:0]		
0x07	COMMAND	7:0						WFBRK	LINCMD[1:0]	
0x08	Reserved									
0x09	EVCTRL	7:0								RXEI
0x0A	BAUD	7:0	BAUD[7:0]							
		15:8	BAUD[15:8]							
0x0C	INTCTRL	7:0	ERROR	CTSIC		RXBRK	RXS	RXC	TXC	DRE
0x0D	INTFLAGS	7:0	ERROR	CTSIC		RXBRK	RXS	RXC	TXC	DRE
0x0E	STATUS	7:0	RXACTIVE	CTS	BUFOVF	FERR	PERR		COLL	ISF
0x0F	Reserved									
0x10	RXDATAL	7:0	DATA[7:0]							
0x11	RXDATAH	7:0			BUFOVF	FERR	PERR	RXC		DATA[8]
0x12	TXDATAL	7:0	DATA[7:0]							
0x13	TXDATAH	7:0								DATA[8]
0x14 ... 0x17	Reserved									
0x18	AUXDATA0	7:0	AUXDATA0[7:0]							
0x19	AUXDATA1	7:0	AUXDATA1[7:0]							
0x1A	AUXDATA2	7:0	AUXDATA2[7:0]							
0x1B ... 0x1E	Reserved									
0x1F	DBGCTRL	7:0								DBGRUN

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
		CSIG[1:0]				CMODE[1:0]		ENABLE
Access		R/W	R/W			R/W	R/W	R/W
Reset		0	0			0	0	0

Bits 6:5 – CSIG[1:0] Communication Signals
These bits select the pins used in communication. All input pins except for CTS are optional for all settings.

Value	Name	Description
0x00	NORMAL	Normal Asynchronous USART (TXD, RXD), Synchronous USART (TXD, RXD, XCK) or SPI host (MOSI, MISO, SCK)
0x01	RS485	Asynchronous USART in RS485 mode (TXD, RXD, XDIR).
0x02	HANDSHAKE	Asynchronous USART with Hardware Handshake (TXD, RXD, RTS, CTS).
0x03	CLKDIS	SPI Host or Synchronous USART host without clock output (MOSI, MISO or TXD, RXD)

Bits 2:1 – CMODE[1:0] Communication Mode
These bits select the communication mode of the USART.

Value	Name	Description
0x00	ASYNCHRONOUS	Asynchronous mode
0x01	SYNCCLIENT	Synchronous client mode
0x02	SYNCHOST	Synchronous host mode
0x03	SPIHOST	SPI Host

Bit 0 – ENABLE Enable
Writing this bit to '1' enables the USART.

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

25.5.2. Control B

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

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

Bit 1 – RXEN Receiver Enable
This bit controls whether the USART receiver is enabled. For more details, refer to the section Disabling the Receiver.

Value	Description
0	The USART receiver is disabled
1	The USART receiver is enabled

Bit 0 – TXEN Transmitter Enable
This bit controls whether the USART transmitter is enabled. The transmitter will override the TxD pin when enabled. For more details, refer to the sectionDisabling the Transmitter.

Value	Description
0	The USART transmitter is disabled
1	The USART transmitter is enabled

25.5.3. Control C

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

Bit	7	6	5	4	3	2	1	0
		DORD	CPHA		SAMPR	FORM[2:0]		
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

This bit indicates the data order when a character is shifted out from the Data register. Both the receiver and transmitter use the same setting. Changing the DORD (Data Order) bit takes effect immediately and will disrupt any communication for both the receiver and the transmitter, potentially corrupting data transfer.

Value	Description
0	LSb of the data frame is transmitted first
1	MSb of the data frame is transmitted first

Bit 5 – CPHA Clock Phase

This bit controls the phase of the interface clock. For more information, refer to the Clock Generation section. This setting only functions when the Communication mode (CMODE) in Control A (USARTn.CTRLA) is configured to SPIHOST.

Value	Description
0	Data are sampled on the leading (first) edge
1	Data are sampled on the trailing (last) edge

Bit 3 – SAMPR Sample Rate

This bit is used to set the transfer rate in asynchronous communication mode. For IrDA encoding and LIN frame formats, SAMPR must be set to '0'. In synchronous operation, SAMPR has no effect and must always be '0'. For more details, refer to the Double-Speed Operation section.

Value	Description
0	16x oversampling
1	8x oversampling (double-speed mode)

Bits 2:0 – FORM[2:0] Frame Format

This bit field defines the frame format. Configuring this a value other than NORMAL enables the transmitting, receiving or detecting special frames. Normal frames can be transmitted and received in all formats.

Value	Name	Description
0x00	NORMAL	Normal USART frame
0x01	AUTOBAUD	Auto-baud mode
0x02	-	Reserved
0x03	-	Reserved
0x04	LINHOST	LIN Host mode
0x05	LINCLIENT	LIN Client mode
0x06	-	Reserved
0x07	-	Reserved

25.5.4. Control D

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

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

Bits 5:4 – PMODE[1:0] Parity Mode
This bit field enables and selects the type of parity generation. For more information, refer to the section Parity.

Value	Name	Description
0x00	DISABLED	Disabled
0x01	-	Reserved
0x02	EVEN	Enabled, even parity
0x03	ODD	Enabled, odd parity

Bit 3 – SBMODE Stop Bit Mode
This bit selects the number of Stop bits to be inserted by the transmitter.

Value	Name	Description
0	1BIT	1 Stop bit
1	2BIT	2 Stop bits

Bits 2:0 – CHSIZE[2:0] Character Size
This bit field selects the number of data bits in each frame. Both the receiver and transmitter use the same setting. When the 9-bit character size (9BIT) is selected, the order of which byte to read or write first, low or high byte of RXDATA or TXDATA, can be configured.

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)

25.5.5. Control E

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

Bit	7	6	5	4	3	2	1	0
				COLDEN	SFDEN	ODME	LBME	MPCM
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – COLDEN Collision Detection Enable
This bit controls whether Collision Detection is enabled. For more information, refer to the Collision Detection section.

Value	Description
0	Collision Detection is disabled
1	Collision Detection is enabled

Bit 3 – SFDEN Start-of-Frame Detection Enable
This bit controls whether the Start-of-Frame Detection mode is enabled. For more information, refer to the Start-of-Frame Detection section.

Value	Description
0	The Start-of-Frame Detection mode is disabled
1	The Start-of-Frame Detection mode is enabled

Bit 2 – ODME Open Drain Mode Enable
This bit controls whether Open Drain mode is enabled. For more information, refer to the One-Wire Mode section.

Value	Description
0	Open Drain mode is disabled
1	Open Drain mode is enabled

Bit 1 – LBME Loop-Back Mode Enable
This bit controls whether the Loop-back mode is enabled. For more information, refer to the One-Wire Mode section.

Value	Description
0	Loop-back mode is disabled
1	Loop-back mode is enabled

Bit 0 – MPCM Multi-Processor Communication Mode
This bit controls whether the Multi-Processor Communication mode is enabled. For more information, refer to the Multiprocessor Communication section.

Value	Description
0	Multi-Processor Communication mode is disabled
1	Multi-Processor Communication mode is enabled

25.5.6. Control F

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

Bit	7	6	5	4	3	2	1	0
	INTDLY[3:0]				GUARD[1:0]		BRKLEN[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

Bits 7:4 – INTDLY[3:0] Inter-byte Delay
These bits define the delay added after a frame before a next byte can be transmitted. The value of the bitfield directly corresponds to the added number of clock cycles of delay. The delay applies to all communication modes.

Bits 3:2 – GUARD[1:0] Guard Time
This bitfield is used to control the guard time when communication signals (CSIG in USARTn.CTRLA) is set to RS485.
For RS485 mode, the guard time is programmable from 0-3 bit times and defines the time that the transmit enable pin (AUX1) remains high before the next Start bit of the first response byte is transmitted.

Bits 1:0 – BRKLEN[1:0] Break Length
These bits define the length of the break field transmitted when the Frame Format (FORM) in Control C (USARTn.CTRLC) is set to LIN Host mode (LINHOST).

Value	Name	Description
0x00	13BIT	Break field transmission is 13 bit times
0x01	17BIT	Break field transmission is 17 bit times
0x02	21BIT	Break field transmission is 21 bit times
0x03	26BIT	Break field transmission is 26 bit times

25.5.7. Control G

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

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

Bits 7:6 – ABW[1:0] Auto-baud Window Size
These bits control the tolerance for the difference between the baud rates between the two synchronizing devices when using LIN Constrained Auto-baud mode. The tolerance is determined by the number of baud samples between every two bits. When the baud rates are identical, there are 32 baud samples between each bit pair, as each bit is sampled 16 times.

Value	Name	Description
0x00	WDW0	32±6 (18% tolerance)
0x01	WDW1	32±5 (15% tolerance)
0x02	WDW2	32±7 (21% tolerance)
0x03	WDW3	32±8 (25% tolerance)

Bit 4 – ENC Encoding
This bit controls whether IrDA encoding and decoding is enabled. For more information, refer to the IRCOM Mode of Operation section.

Value	Description
0	IrDA encoding and decoding is disabled
1	IrDA encoding and decoding is enabled

Bits 2:0 – PROTCONV[2:0] Protocol Converter
This bit field is used to enable protocol conversion and defines how received characters are decoded to either a '0' or '1' bit. Protocol conversion cannot be used if IrDA encoding (ENC in USARTn.CTRLG) is enabled. Any configuration other than '0' enables protocol conversion for transmit.

Value	Name	Description
0x00	NONE	No protocol conversion
0x01	FIXED	Single character for both '0' and '1'
0x02	MASKED	AUXDATA2 works as a mask when decoding received data
0x03	DUAL	AUXDATA2 defines second value for '1' when decoding received data
0x04	-	Reserved
0x05	-	Reserved
0x06	-	Reserved
0x07	-	Reserved

25.5.8. COMMAND

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

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

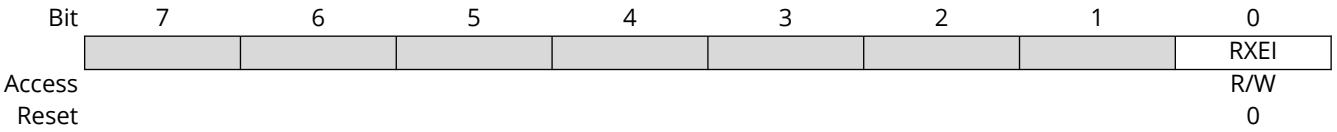
Bit 2 – WFBRK Wait For Break
Setting this bit enables the Wait For Break feature for the following incoming frame. After this frame is received, the feature is automatically disabled. For more information, refer to the Auto-Baud section.

Bits 1:0 – LINCMD[1:0] LIN Command
This bit field defines the LIN header transmission control. These settings are only valid when operating in LIN host mode. The configuration is automatically reset when the command is completed.

Value	Name	Description
0x00	NORMAL	Normal frame
0x01	BREAK	BREAK field transmitted when TXDATA is written
0x02	HEADER	BREAK, SYNC and PID are automatically transmitted when the identifier is written to TXDATA
0x03	-	Reserved

25.5.9. Event Control

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



Bit 0 – RXEI Receive Event Input Enable
This bit controls whether the event source for the receiver is enabled.

Value	Description
0	Receive Event input is disabled
1	Receive Event input is enabled

25.5.10. Baud Register

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

Ongoing transmissions by the transmitter and receiver will be corrupted if the baud rate is changed during communication. Writing to this register triggers an immediate update of the baud rate prescaler. For more information on how to set the baud rate, refer to Table 25-1.

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
This bit field holds the MSB of the 16-bit Baud register.

Bits 7:0 – BAUD[7:0] USART Baud Rate Low Byte
This bit field holds the LSB of the 16-bit Baud register.

25.5.11. Interrupt Control

Name: INTCTRL
Offset: 0x0C
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ERROR	CTSIC		RXBRK	RXS	RXC	TXC	DRE
Access	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	0	0	0	0

Bit 7 – ERROR Error Interrupt Enable
This bit controls whether the Error interrupt is enabled.

Value	Description
0	Error interrupt is disabled
1	Error interrupt is enabled

Bit 6 – CTSIC Clear to Send Input Change Interrupt Enable
This bit controls whether the Clear to Send Input Change interrupt is enabled.

Value	Description
0	Clear to Send Input Change interrupt is disabled
1	Clear to Send Input Change interrupt is enabled

Bit 4 – RXBRK Receive Break Interrupt Enable
This bit controls whether the Receive Break interrupt is enabled.

Value	Description
0	Receive Break interrupt is disabled
1	Receive Break interrupt is enabled

Bit 3 – RXS Receiver Start-of-Frame Interrupt Enable
This bit controls whether the Receiver Start-of-Frame interrupt is enabled.

Value	Description
0	Receiver Start-of-Frame interrupt is disabled
1	Receiver Start-of-Frame interrupt is enabled

Bit 2 – RXC Receive Complete Interrupt Enable
This bit controls whether the Receive Complete interrupt is enabled.

Value	Description
0	Receive Complete interrupt is disabled
1	Receive Complete interrupt is enabled

Bit 1 – TXC Transmit Complete Interrupt Enable
This bit controls whether the Transmit Complete interrupt is enabled.

Value	Description
0	Transmit Complete interrupt is disabled
1	Transmit Complete interrupt is enabled

Bit 0 – DRE Data Register Empty Interrupt Enable
This bit controls whether the Data Register Empty interrupt is enabled.

Value	Description
0	Data Register Empty interrupt is disabled
1	Data Register Empty interrupt is enabled

25.5.12. Interrupt Flags

Name: INTFLAGS
Offset: 0x0D
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ERROR	CTSIC		RXBRK	RXS	RXC	TXC	DRE
Access	R/W	R/W		R/W	R/W	R	R/W	R
Reset	0	0		0	0	0	0	0

Bit 7 – ERROR Error Interrupt Flag
This flag is set when an error occurrs during either transmission or reception. Errors that will set this flag have corresponding status flags in the Status (USARTn.STATUS) register. The specific errors that will set this flag are: PERR, FERR, ISF, COLL and BUFOVF. The flag can be cleared by writing a '1' to its bit location.

Bit 6 – CTSIC Clear to Send Input Change Interrupt Flag
The CTSIC flag is set when a change is detected in CTS pin. The flag can be cleared by writing a '1' to its bit location.

Bit 4 – RXBRK Receive Break Interrupt Flag
This interrupt flag is valid when the Frame Format (FORM) bitfield in the Control C (USARTn.CTRLC) register is configured to LINCLIENT or AUTOBAUD mode. The break detector uses a fixed threshold of 11 consecutive low bits to detect a Break condition. The RXBRK bit is set after a valid BREAK and SYNC character is detected. The bit is automatically cleared upon reception of the next data, and it can also be cleared manually by writing a '1' to its bit location.

Bit 3 – RXS Receiver Start-of-Frame Interrupt Flag
This flag is set when Start-of-Frame detection is enabled, the device is in Standby sleep mode, and a valid start bit is detected. This flag can be cleared by writing a '1' to its bit location. This flag is not used in SPI Host mode.

Bit 2 – RXC 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.

Bit 1 – TXC 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 (TXDATAL and TXDATAH) registers. The flag is cleared when the Transmit Data (TXDATA) registers are written to, or by writing a '1' to its bit location.

Bit 0 – DRE Data Register Empty Interrupt Flag
This flag is set when the Transmit Data (USARTn.TXDATAL and USARTn.TXDATAH) registers are empty, indicating they are ready to accept new data. The flag is cleared when these registers contain data that has not been moved into the transmit shift register.

25.5.13. USART Status Register

Name: STATUS
Offset: 0x0E
Reset: 0x20
Property: -

Bit	7	6	5	4	3	2	1	0
	RXACTIVE	CTS	BUFOVF	FERR	PERR		COLL	ISF
Access	R	R	R	R	R		R/W	R/W
Reset	0	0	0	0	0		0	0

Bit 7 – RXACTIVE Receive Active
This bit indicates the current state of the receiver. It is set when a start bit is detected and cleared when the Receive Complete (RXC) flag in the Interrupt Flags (USARTn.INTFLAGS) register is set.

Bit 6 – CTS Clear to Send
This bit indicates the current state of the CTS pin when flow control is enabled by configuring CSIG to HANDSHAKE in the Control A (USARTn.CTRLA) register.

Bit 5 – BUFOVF Buffer Overflow
This flag is set if a buffer overflow is detected. A buffer overflow occurs when the receive buffer is full, a new frame is waiting in the receive shift register, and a new Start bit is detected. A corresponding BUFOVF flag is set in any frame that has overwritten another frame. This flag remains set as long as a BUFOVF flag is set in any of the frames in the receive buffer.

Bit 4 – FERR Frame Error
This flag indicates that a received frame did not meet the requirements expected by the receiver. When the Protocol Converter is enabled by PROTCONV in the Control G (USARTn.CTRLG) register, this flag indicates that a frame corresponds to neither a '0' nor a '1'. In any other mode, this flag indicates that the stop bit was not received at the expected time. A corresponding FERR flag is set in any frame with a frame error, and this flag remains set as long as a FERR flag is set in any of the frames in the receive buffer.
This flag is not used in the SPI Host mode.

Bit 3 – PERR Parity Error
This flag is set when a parity error is detected in an incoming frame. A corresponding PERR flag is set in any frame which with a parity error. This flag remains set as long as a PERR flag is set in any of the frames in the receive buffer.

Bit 1 – COLL Collision Detected
This bit is set when collision detection is enabled by setting the Collision Detection Enable (COLDEN) bit in the Control E (USARTn.CTRLE) register and a collision is detected. This flag can be cleared by writing a '1' to its bit location.

Bit 0 – ISF Inconsistent SYNC Field
This bit is set when an auto-baud mode is enabled using the Frame Format (FORM) bit field in the Control C (USARTn.CTRLC) register and the SYNC field does not meet its requirements. When FORM is set to generic Auto-baud (AUTOBAUD) mode, the SYNC field is considered valid if it results in a valid baud setting. When FORM is set to LIN Client (LINCLIENT) mode, the SYNC field is valid only if it reads 0x55. This flag is cleared by writing a '1' to its bit location.

25.5.14. Receiver Data Register Low Byte

Name: RXDATAL
Offset: 0x10
Reset: 0x00
Property: -

This register contains the eight LSbs of the data received by the USART receiver. The USART receiver is double-buffered, and this register always holds the data for the oldest received frame. If only one frame of data is present in the receive buffer, this register contains that data.

The buffer shifts out the data when either RXDATAL or RXDATAH is read, depending on the configuration. The register which does not lead to data being shifted must be read first to be able to read both bytes before shifting.

When the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register is configured to 9-bit mode with low byte first, reading RXDATAH shifts the receive buffer. In all other configurations, reading RXDATAL shifts the buffer.

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

25.5.15. Receiver Data Register High Byte

Name: RXDATAH
Offset: 0x11
Reset: 0x00
Property: -

This register contains the MSb of the data received by the USART receiver, as well as status bits reflecting the status of the received data frame. The USART receiver is double-buffered, and this register always holds the data and status bits for the oldest received frame. If only one frame of data and status bits is present in the receive buffer, this register contains that information.

The buffer shifts out the data when either RXDATAL or RXDATAH is read, depending on the configuration. The register which does not lead to data being shifted must be read first to be able to read both bytes before shifting.

When the Character Size (CHSIZE) bits in the Control D (USARTn.CTRLD) register is configured to 9-bit mode with low byte first, reading RXDATAH shifts the receive buffer. In all other configurations, reading RXDATAL shifts the buffer.

Bit	7	6	5	4	3	2	1	0
			BUFOVF	FERR	PERR	RXC		DATA[8]
Access			R	R	R	R		R
Reset			0	0	0	0		0

Bit 5 – BUFOVF Buffer Overflow
This flag is set for any byte that has been overwritten due a buffer overflow. A buffer overflow occurs when the receive buffer is full, a new frame is waiting in the receive shift register, and another new frame is received.
This flag is not used in the SPI Host mode.

Bit 4 – FERR Frame Error
This flag indicates that the received frame did not meet the requirements expected by the receiver. When the Protocol Converter is enabled by PROTCONV in the Control G (USARTn.CTRLG) register, this flag indicates that the frame corresponds to neither a '0' nor a '1'. In other mode, this flag indicates that the stop bit was not received at the expected time.
This flag is not used in the SPI Host mode.

Bit 3 – PERR Parity Error
This flag is set if parity checking is enabled and the received data contains a parity error. For more details on how parity is calculated, refer to the Parity section.

Bit 2 – RXC Receive Complete
This flag is set when there are unread data in the receive buffer and cleared when the receive buffer becomes empty.

Bit 0 – DATA[8] Receiver Data Register
When using a 9-bit frame size, this bit holds the ninth bit (MSb) of the received data.
If the Frame Format (FORM) bit field in the Control C (USARTn.CTRLC) register is configured to LIN Client (LINCLIENT) mode, this bit indicates whether the received data are within the response space of a LIN frame. The bit is cleared if the received data are in the protected identifier field; otherwise, it is set.

25.5.16. Transmit Data Register Low Byte

Name: TXDATAL
Offset: 0x12
Reset: 0x00
Property: -

The data written to this register is automatically loaded into the TX buffer and then transferred to the dedicated shift register. The shift register outputs each of the bits serially to the TXD pin for transmission.

When using a 9-bit frame size, the ninth bit (MSb) must be written to the Transmit Data Register High Byte (USARTn.TXDATAH). In that case, the buffer shifts data when either the Transmit Data Register Low Byte (USARTn.TXDATAL) or the Transmit Data Register High Byte (USARTn.TXDATAH) is written, depending on the configuration. The register which does not lead to data being shifted must be written first to be able to write both registers before shifting.

When the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register is configured to 9-bit mode with low byte first, writing of the Transmit Data Register High Byte (TXDATAH) shifts the transmit buffer. In all other configurations,writing to the Transmit Data Register Low Byte (TXDATAL) shifts the buffer.

This register may only be written when the Data Register Empty (DRE) Interrupt Flag in the Interrupt Flags (USARTn.INTFLAGS) register is set.

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 Low Byte

25.5.17. Transmit Data Register High Byte

Name: TXDATAH
Offset: 0x13
Reset: 0x00
Property: -

Data written to this register is automatically loaded into the TX buffer and then transferred to the dedicated shift register. The shift register outputs each of the bits serially to the TXD pin for transmission.

When using a 9-bit frame size, the ninth bit (MSb) must be written to the Transmit Data Register High Byte (USARTn.TXDATAH). In that case, the buffer shifts data when either the Transmit Data Register Low Byte (USARTn.TXDATAL) or the Transmit Data Register High Byte (USARTn.TXDATAH) is written, depending on the configuration. The register, which does not lead to data being shifted, must be written first to be able to write both registers before shifting.

When the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register is configured to 9-bit mode with low byte first, writing to the Transmit Data Register High Byte (TXDATAH) shifts the transmit buffer. In all other configurations, writing to the Transmit Data Register Low Byte (TXDATAL) shifts the buffer.

This register may only be written when the Data Register Empty Interrupt Flag (DRE) Interrupt Flag in the Interrupt Flags (USARTn.INTFLAGS) register is set.

Bit	7	6	5	4	3	2	1	0
								DATA[8]
Access								R/W
Reset								0

Bit 0 – DATA[8] Transmit Data Register High Byte

25.5.18. Auxiliary Data 0

Name: AUXDATA0
Offset: 0x18
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	AUXDATA0[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 – AUXDATA0[7:0] Auxiliary Data 0

When protocol conversion is enabled by configuring the Protocol Converter (PROTCONV) bit field in the Control G (USARTn.CTRLG) register, this register holds the bit value for a "zero character". If the bit to be transmitted is '0' it is replaced with this character. The number of bits used for encoding and decoding is defined by the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register. Note that 9-bit character size is not supported for protocol conversion.

When IrDA Encoding (ENC) is enabled in the Control G (USARTn.CTRLG) register, this register sets the pulse modulation scheme for the transmitter. If the register value is set to '0', 3/16 of the baud rate period pulse modulation is used. Setting the value from `0x01` to `0xFE` provide a fixed pulse length coding, with 8-bit value sets the number of system clock periods for the pulse. The start of the pulse will be synchronized with the rising edge of the baud rate clock. Setting the value to `0xFF` disables pulse coding, allowing the RX and TX signals to pass through the IRCOM module unaltered.

AUXDATA0 must be configured before setting the Transmitter Enable (TXEN) or Receivere Enable (RXEN) bits in the Control B (USARTn.CTRLB) register.

25.5.19. Auxiliary Data 1

Name: AUXDATA1
Offset: 0x19
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	AUXDATA1[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 – AUXDATA1[7:0] Axillary Data 1

When protocol conversion is enabled by configuring the Protocol Converter (PROTCONV) bit field in the Control G (USARTn.CTRLG) register, this register holds the bit value for a "one character". If the bit to be transmitted is '1' it is replaced with this character. The number of bits used for encoding and decoding is defined by the Character Size (CHSIZE) bit field in the Control D (USARTn.CTRLD) register. Note that 9-bit character size is not supported for protocol conversion.

When IrDA Encoding (ENC) is enabled in the Control G (USARTn.CTRLG) register, this register sets the filter coefficient for the IRCOM transceiver. If register value is set to '0', filtering is disabled. Setting the value between `0x01` and `0xFF` enables filtering, requiring x+1 equal samples for a pulse to be accepted.

AUXDATA1 must be configured before setting the Transmitter Enable (TXEN) or Receivere Enable (RXEN) bits in the Control B (USARTn.CTRLB) register.

25.5.20. Auxiliary Data 2

Name: AUXDATA2
Offset: 0x1A
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	AUXDATA2[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 – AUXDATA2[7:0] Axillary Data 2
This register is used together with AUXDATA1 to define how a “one character” is decoded when protocol conversion is enabled by configuring the Protocol Converter (PROTCONV) bit field in the Control G (USARTn.CTRLG) register.

25.5.21. Debug Control Register

Name: DBGCTRL
Offset: 0x1F
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

26. SPI - Serial Peripheral Interface

26.1. Features

- Full Duplex, Three-Wire Synchronous Data Transfer
- Host or Client 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) Host SPI Mode

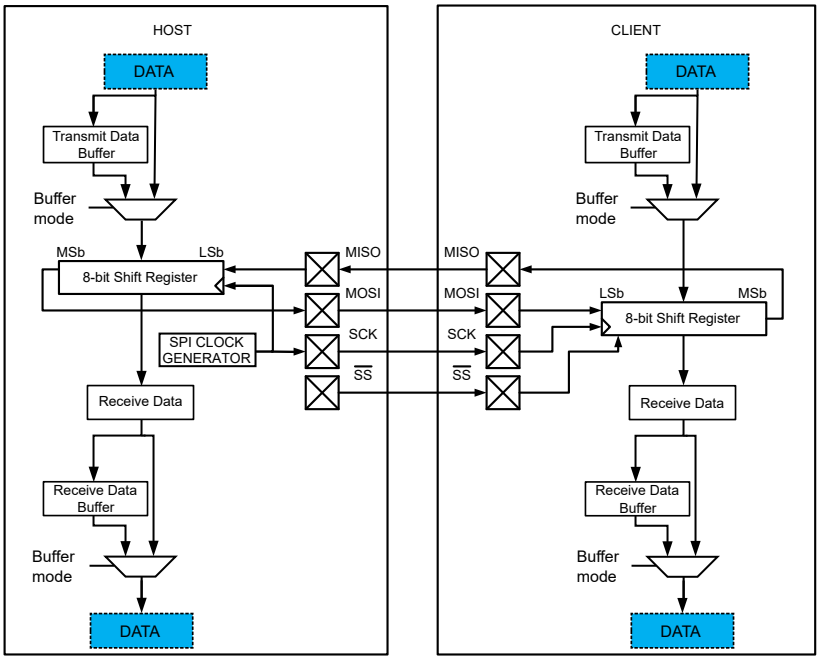
26.2. Overview

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 host or client. The host initiates and controls all data transactions.

The interconnection between host and client devices with SPI is shown in the block diagram below. The system consists of two shift registers and a server clock generator. The SPI host initiates the communication cycle by pulling the desired client's Client Select ($\overline{SS}$) signal low. The host and client prepare the data to be sent to their respective shift registers, and the host generates the required clock pulses on the SCK line to exchange data. Data are always shifted from host to client on the host output, client input (MOSI) line, and from client to host on the host input, client output (MISO) line.

26.2.1. Block Diagram

Figure 26-1. 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 (SPIn.DATA) register will write the shift register in Normal mode and the Transmit Buffer register in Buffer mode. Reading the Receive Data (SPIn.DATA) register will read the Receive Data register in Normal mode and the Receive Data Buffer in Buffer mode.

In Host mode, the SPI has a clock generator to generate the SCK clock. In Client mode, the received SCK clock is synchronized and sampled to trigger the shifting of data in the shift register.

26.2.2. Signal Description

Table 26-1. Signals in Host and Client Mode

Signal	Description	Pin Configuration	
		Host Mode	Client Mode
MOSI	Host Out Client In	User defined(1)	Input
MISO	Host In Client Out	Input	User defined(1,2)
SCK	Serial Clock	User defined(1)	Input
$\overline{SS}$	Client Select	User defined(1)	Input

Notes:

1. If the pin data direction is configured as output, the pin level is controlled by the SPI.
2. If the SPI is in Client 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 26-1 is overridden.

26.3. Functional Description

26.3.1. 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 the SPI host/client operation by writing the Host/Client Select (MASTER) bit in the Control A (SPIn.CTRLA) register.
3. In Host mode, select the clock speed by writing the Prescaler (PRESC) bits and the Clock Double (CLK2X) bit in SPIn.CTRLA.
4. Optional: Select the Data Transfer mode by writing to the MODE bits in the Control B (SPIn.CTRLB) register.
5. Optional: Write the Data Order (DORD) bit in SPIn.CTRLA.
6. Optional: Set up the Buffer mode by writing the BUFEN and BUFWR bits in the Control B (SPIn.CTRLB) register.
7. Optional: To disable the multi-host support in Host mode, write ‘1’ to the Client Select Disable (SSD) bit in SPIn.CTRLB.
8. Enable the SPI by writing a ‘1’ to the ENABLE bit in SPIn.CTRLA.

26.3.2. Operation

26.3.2.1. Host Mode Operation

When the SPI is configured in Host mode, a write to the SPIn.DATA register will start a new transfer. The SPI host can operate in two modes, Normal and Buffer, as explained below.

26.3.2.1.1. 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 (SPIn.DATA) register before the entire transfer has been completed. A premature write will cause corruption of the transmitted data, and the Write Collision (WRCOL) flag 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 the 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 been completed, the Interrupt Flag (IF) will be set in the Interrupt Flags (SPIn.INTFLAGS) register. 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 (SPIn.INTCTRL) register will enable the interrupt.

26.3.2.1.2. 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 does not affect Host 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 in the following ways:

1. New bytes can be written to the DATA (SPIn.DATA) register as long as the Data Register Empty Interrupt Flag (DREIF) in the Interrupt Flag (SPIn.INTFLAGS) register 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 (SPIn.INTFLAGS) register 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 (SPIn.INTCTRL) register enables the Transfer Complete Interrupt.

26.3.2.1.3. $\overline{SS}$ Pin Functionality in Host Mode - Multi-Host Support

In Host mode, the Client Select Disable (SSD) bit in the Control B (SPIn.CTRLB) register 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 Host to Client mode. This allows multiple SPI hosts 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. 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 Host SPI operation. A low level will be interpreted as another host is trying to take

control of the bus. This will switch the SPI into Client mode, and the hardware of the SPI will perform the following actions:

1. The Host (MASTER) bit in the SPI Control A (SPIn.CTRLA) register is cleared, and the SPI system becomes a client. The direction of the SPI pins will be switched when the conditions in Table 26-2 are met.
2. The Interrupt Flag (IF) bit in the Interrupt Flags (SPIn.INTFLAGS) register will be set. If the interrupt is enabled and the global interrupts are enabled, the interrupt routine will be executed.

Table 26-2. 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	Host activated (selected)
	Low	Host deactivated, switched to Client mode
Output	High	Host activated (selected)
	Low	

Note: If the device is in Host mode and it cannot be ensured that the $\overline{SS}$ pin will stay high between two transmissions, the status of the Host (MASTER) bit in SPIn.CTRLA has to be checked before a new byte is written. After the Host 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 Host mode.

26.3.2.2. Client Mode

In Client mode, the SPI peripheral receives the SPI clock and Client Select from a Host. Client mode supports three operational modes: One Normal mode and two configurations for the Buffered mode. In Client mode, the control logic will sample the incoming signal on the SCK pin.

26.3.2.2.1. Normal Mode

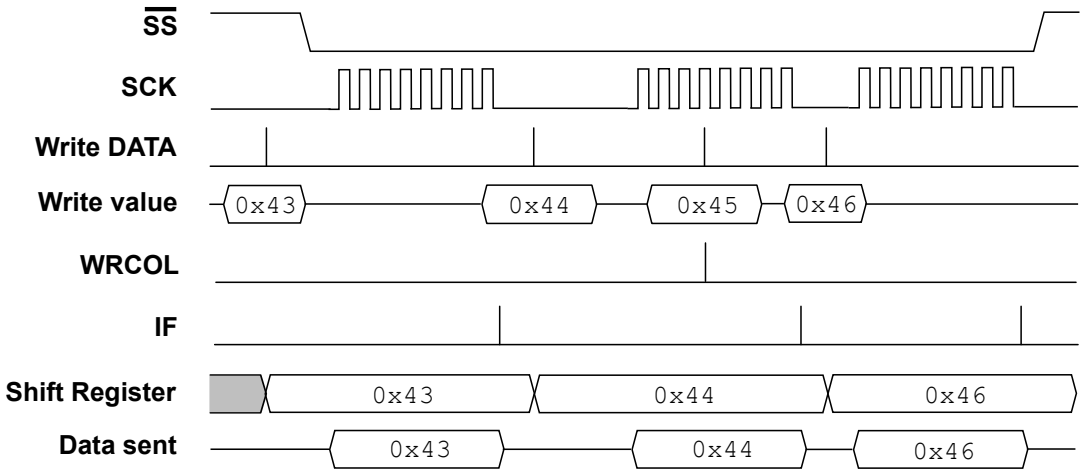
In Normal mode, the SPI peripheral will remain Idle as long as the $\overline{SS}$ pin is driven high. In this state, the software may update the contents of the DATA register, but the data will not be shifted out by incoming clock pulses on the SCK pin until the $\overline{SS}$ pin is driven low. If the $\overline{SS}$ pin is driven low, the client 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 been completed. A premature write will be ignored, and the hardware will set the Write Collision (WRCOL) flag in SPIn.INTFLAGS.

When the $\overline{SS}$ pin is driven high, the SPI logic is halted, and the SPI client will not receive any new data. Any partially received packet in the shift register will be lost.

Figure 26-2 shows a transmission sequence in Normal mode. Notice how the value `0x45` is written to the DATA register but never transmitted.

Figure 26-2. 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 (WRCOL) flag in SPIn.INTFLAGS is set.

26.3.2.2.2. Buffer Mode

The SPI peripheral can be configured in Buffered mode by writing a '1' to the Buffer Mode Enable (BUFEN) bit in the Control B (SPIn.CTRLB) register to avoid data collisions.

This mode will enable two receive buffers and one transmit buffer. Both will have separate interrupt flags, transmit complete and receive complete. Figure 26-1 shows the extra buffers.

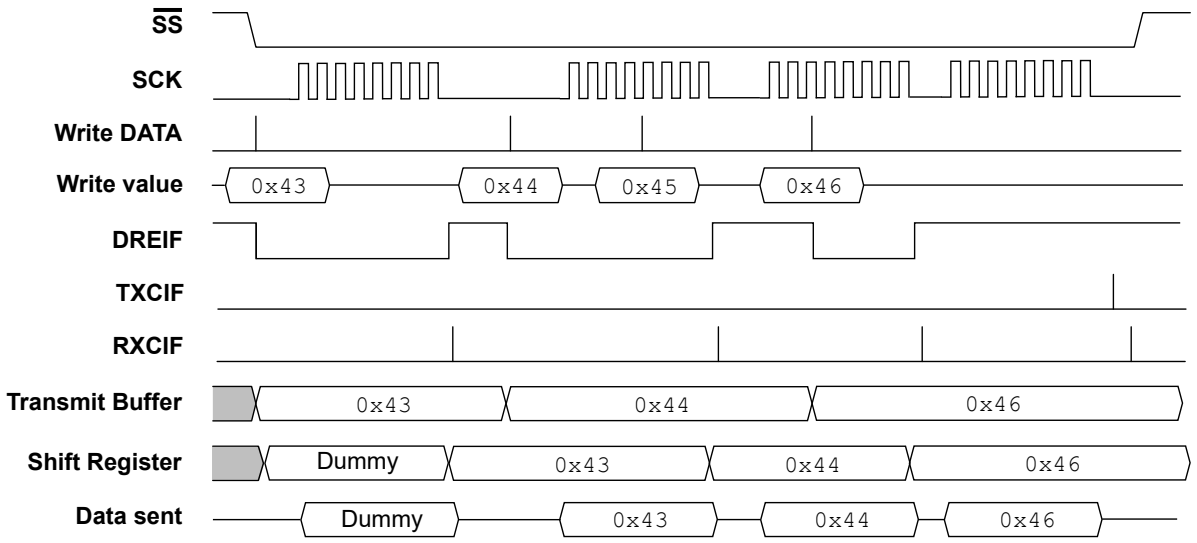
When Buffer mode is enabled it can work in two different ways. The Buffer Mode Wait for Receive (BUFWR) bit in the Control B (SPIn.CTRLB) register controls how the Buffer mode works. The details of how they work including timing diagrams are described below.

Note: When operating as a client in Buffered mode and the SPI clock is close to maximum frequency, the client may not be able to set up data in time for the first sample edge during back-to-back transfers. Refer to the *Electrical Characteristics - SPI* section for details.

Client Buffer Mode with Wait for Receive Bit Written to '0'

In Client mode, if the Buffer mode Wait for Receive (BUFWR) bit in SPIn.CTRLB is written to '0', a dummy byte will be sent before the transmission of user data starts. Figure 26-3 shows a transmission sequence with this configuration. Notice how the value 0x45 is written to the Data (SPIn.DATA) register but never transmitted.

Figure 26-3. SPI Timing Diagram in Buffer Mode with BUFWR in SPIn.CTRLB Written to '0'



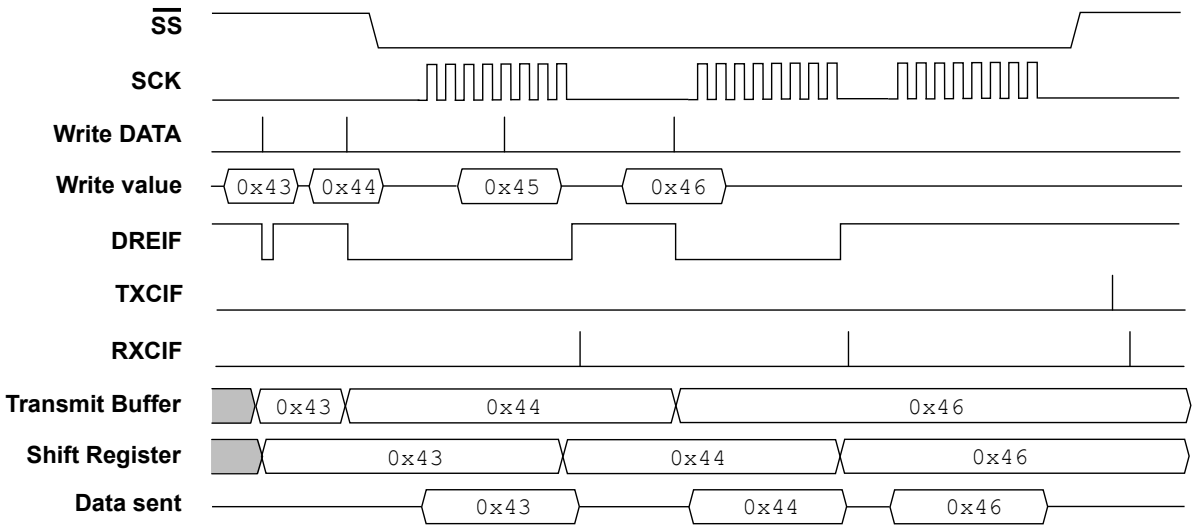
When the Wait for Receive (BUFWR) bit in SPIn.CTRLB is written to '0', all writes to the Data (SPIn.DATA) register go to the Transmit Data Buffer register. The figure above shows that the value 0x43 is written to the Data (SPIn.DATA) register but not immediately transferred to the shift register, so the first byte sent will be a dummy byte. The value of the dummy byte equals the values that were in the shift register at the same time. After the first dummy transfer is completed, the value 0x43 is transferred to the shift register. Then 0x44 is written to the Data (SPIn.DATA) register 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 (SPIn.DATA) register, 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 (SPIn.DATA) register, 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 in the Transmit Data Buffer register has been sent.

Client Buffer Mode with Wait for Receive Bit Written to '1'

In Client mode, if the Buffer Mode Wait for Receive (BUFWR) bit in SPIn.CRTLB is written to '1', the transmission of user data starts as soon as the $\overline{SS}$ pin is driven low. Figure 26-4 shows a transmission sequence with this configuration. Notice how the value 0x45 is written to the Data (SPIn.DATA) register but never transmitted.

Figure 26-4. SPI Timing Diagram in Buffer Mode with CTRLB.BUFWR Written to '1'



All writes to the Data (SPIn.DATA) register go to the Transmit Data Buffer register. The figure above shows that the value 0x43 is written to the Data (SPIn.DATA) register, and since the $\overline{SS}$ pin is high, it is copied to the shift register in the next cycle. 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 (SPIn.DATA) register, 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 identically to the Buffer Mode Wait for Receive (BUFWR) bit in SPIn.CTRLB set to '0'.

26.3.2.2.3. $\overline{SS}$ Pin Functionality in Client Mode

The Client Select ($\overline{SS}$) pin plays a central role in the operation of the SPI. Depending on the SPI mode 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 Client mode, the $\overline{SS}$, 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 an 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 an output, the MISO pin will be tri-stated. Table 26-3 shows an overview of the $\overline{SS}$ pin functionality.

Table 26-3. 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	Client deactivated (deselected)	Tri-stated	Input
	Low	Client activated (selected)	Output	Input

Note: In Client 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 Client bit counter synchronized with the host clock generator.

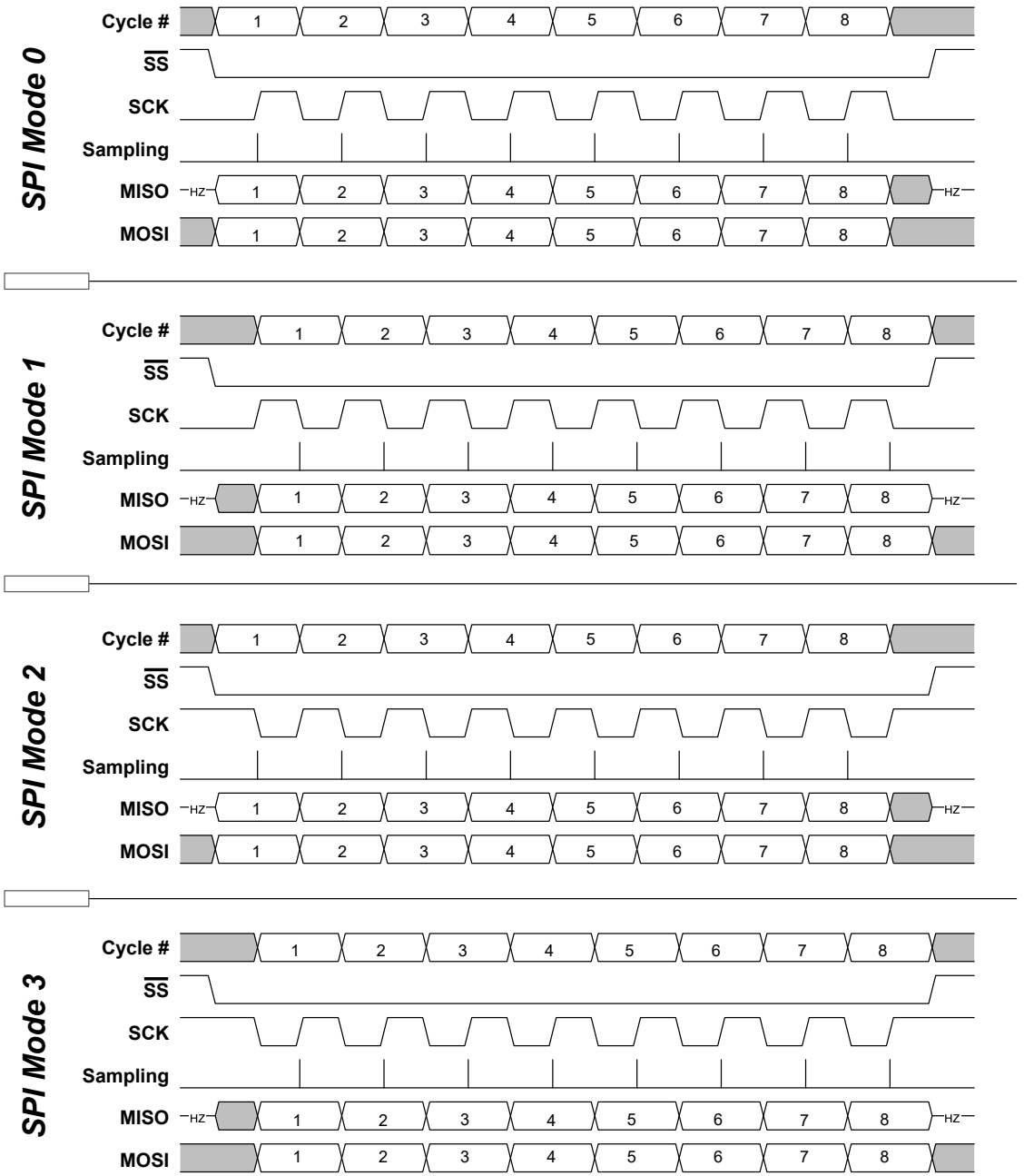
26.3.2.3. Data Modes

There are four combinations of SCK phase and polarity concerning the serial data. The desired combination is selected by writing to the MODE bits in the Control B (SPIn.CTRLB) register.

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 26-5. SPI Data Transfer Modes



26.3.2.4. Events

The SPI can generate the following events:

Table 26-4. Event Generators in SPI

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Module	Event				
SPIn	SCK	SPI Host clock	Level	CLK_PER	Minimum two CLK_PER periods

The SPI has no event users.

Refer to the *EVSYS - Event System* section for more details regarding event types and Event System configuration.

26.3.2.5. Interrupts

Table 26-5. 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: Client 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.

26.4. 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]							

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
		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 Host/Client 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 Host mode.
This behavior is controlled by the Client Select Disable (SSD) bit in SPIn.CTRLB.

Value	Description
0	SPI Client mode selected
1	SPI Host 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 Host mode.

Value	Description
0	SPI speed (SCK frequency) is not doubled
1	SPI speed (SCK frequency) is doubled in Host mode

Bits 2:1 – PRESC[1:0] Prescaler

This bit field controls the SPI clock rate configured in Host mode. These bits have no effect in Client 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

26.5.2. 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 Client Select Disable
If this bit is set when operating as SPI Host (MASTER = 1 in SPIn.CTRLA), $\overline{SS}$ does not disable Host mode.

Value	Description
0	Enable the Client Select line when operating as SPI host
1	Disable the Client Select line when operating as SPI host

Bits 1:0 – MODE[1:0] Mode
These bits select the Transfer mode. The four combinations of SCK phase and polarity concerning 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

26.5.3. 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 Client Select Trigger Interrupt Enable
In Buffer mode, this bit enables the Client 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.

26.5.4. 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 Host mode, this will also set this flag. The IF is cleared by writing a '1' to it. 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.

26.5.5. 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 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 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 Client Select Trigger Interrupt Flag
This flag indicates that the SPI has been in Host mode, and the $\overline{SS}$ pin has been pulled low externally, so the SPI is now working in Client mode. The flag will only be set if the Client 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.

26.5.6. 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 Host mode while preparing data for sending in Client 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.

27. TWI - Two-Wire Interface

27.1. 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
  - Client arbitration allows support for the Address Resolution Protocol (ARP) in software
  - Configurable SMBus Layer 1 time-outs in hardware
  - Independent time-outs for Dual mode
- Independent Host and Client Operation
  - Combined (same pins) or Dual mode (separate pins)
  - Single or multi-host bus operation with full arbitration support
- Hardware Support for Client 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

27.2. 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).

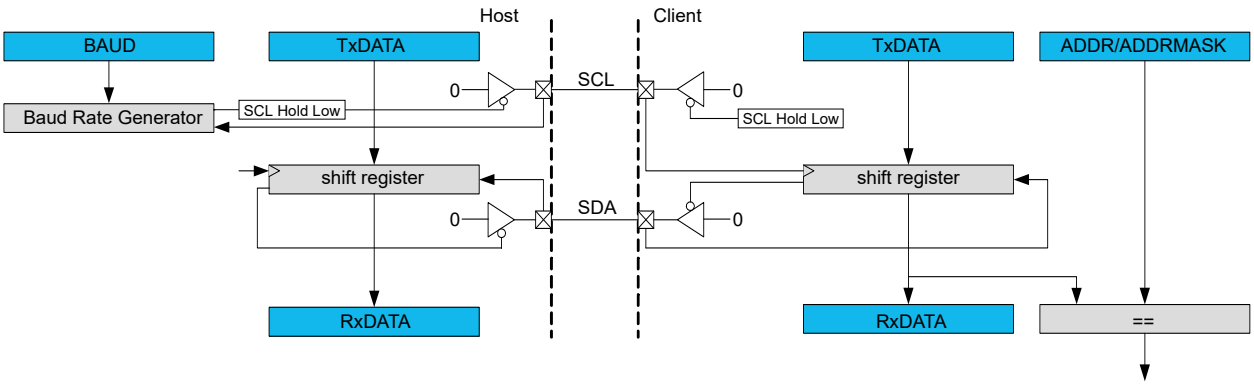
The TWI bus connects one or several client devices to one or several host devices. Any device connected to the bus can act as a host, a client, or both. The host generates the SCL using a Baud Rate Generator (BRG) and initiates data transactions by addressing one client and telling whether it wants to transmit or receive data. The BRG can generate the Standard mode (Sm) and Fast mode (Fm, Fm+) bus frequencies from 100 kHz 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 the Host and Client modes.

The TWI supports multi-host bus operations and arbitration. An arbitration scheme handles cases where more than one host tries to transmit data simultaneously. The TWI also supports Smart mode, which can auto-trigger operations and thus reduce software complexity. The TWI supports Dual mode with simultaneous host and client operations implemented as independent units with separate enabling and configuration. The TWI supports Quick Command mode, where the host can address a client without exchanging data.

27.2.1. Block Diagram

Figure 27-1. TWI Block Diagram



27.2.2. Signal Description

Signal	Description	Type
SCL	Serial Clock Line	Digital I/O
SDA	Serial Data Line	Digital I/O

27.3. Functional Description

27.3.1. 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 connecting multiple devices on a serial bus. A device connected to the bus can be a host or a client. Only host devices can control the bus and the bus communication.

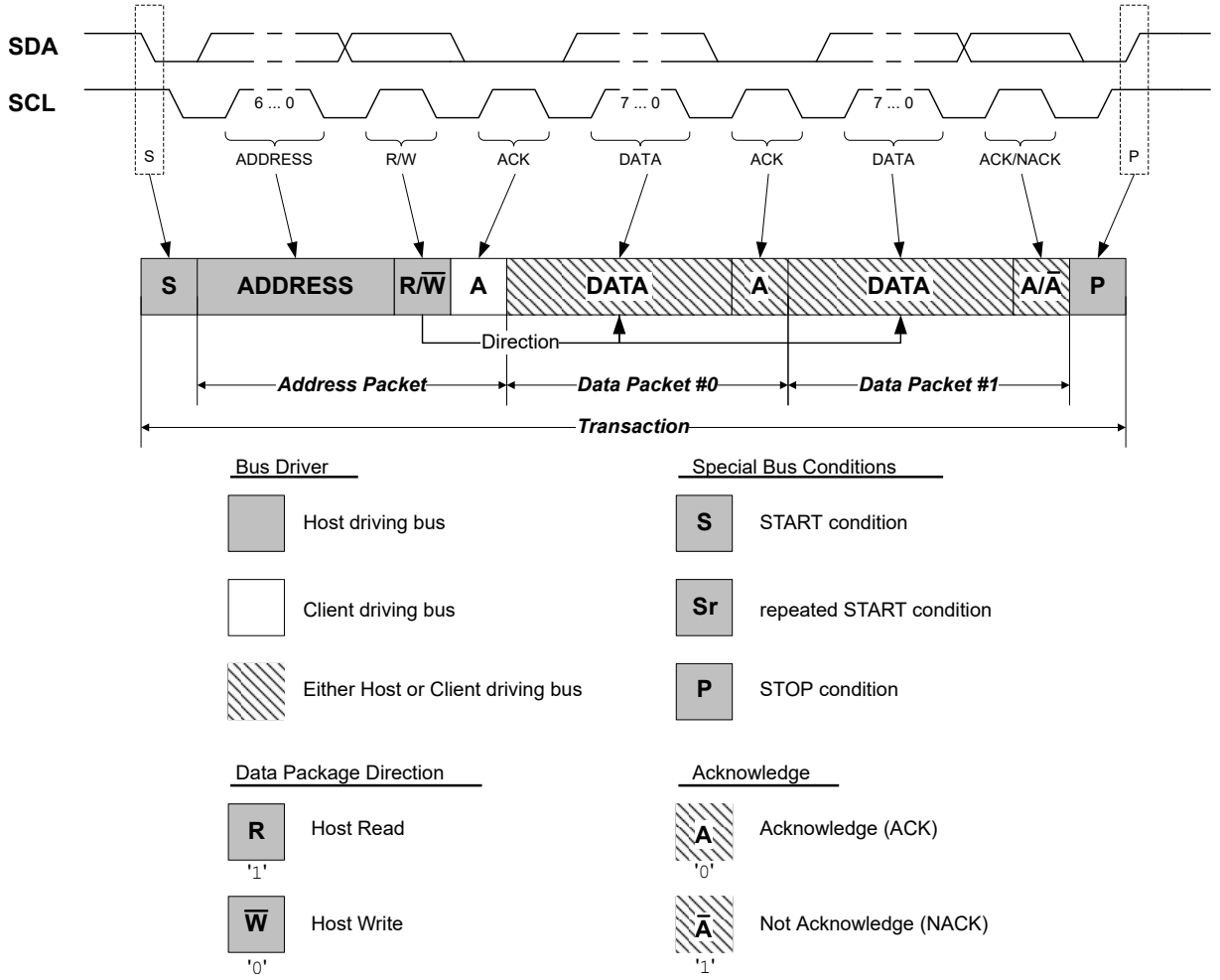
A unique address is assigned to each client device connected to the bus, and the host will use it to control the client and initiate a transaction. Several hosts can connect to the same bus, called a multi-host environment. An arbitration mechanism is provided for resolving bus ownership among hosts since only one host device may own the bus at any given time.

A host indicates the start of a transaction by issuing a Start condition (S) on the bus. The host provides the clock signal for the transaction. An address packet with a 7-bit client address (ADDRESS) and a direction bit, representing whether the host wishes to read or write data (R/$\overline{W}$), are then sent.

The addressed I^2C client will then acknowledge (ACK) the address, and data packet transactions can begin. Every 9-bit data packet consists of eight data bits followed by a 1-bit reply indicating whether the data was acknowledged or not by the receiver.

After transferring all the data packets (DATA), the host issues a Stop condition (P) on the bus to end the transaction.

Figure 27-2. Basic TWI Transaction Diagram Topology for a 7-Bit Address Bus



27.3.2. TWI Basic Operation

27.3.2.1. Initialization

If used, configure the following bits before enabling the TWI peripheral:

- 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

27.3.2.1.1. Host Initialization

Write the Host Baud Rate (TWIn.MBAUD) register to a value that will result in a valid TWI bus clock frequency. Writing a '1' to the Enable TWI Host (ENABLE) bit in the Host Control A (TWIn.MCTRLA) register will enable the TWI host. The Bus State (BUSSTATE) bit field from the Host Status (TWIn.MSTATUS) register must be set to 0x1 to force the bus state to Idle.

27.3.2.1.2. Client Initialization

Follow these steps to initialize the client:

1. Before enabling the TWI client, configure the SDA Setup Time (SDASETUP) bit in the Control A (TWIn.CTRLA) register.
2. Write the client address to the Client Address (TWIn.SADDR) register.
3. Write a '1' to the Enable TWI Client (ENABLE) bit in the Client Control A (TWIn.SCTRLA) register to enable the TWI client.

The TWI client will now wait for a host device to issue a Start condition and the matching client address.

27.3.2.2. TWI Host Operation

The TWI host is byte-oriented, with an optional interrupt after each byte. There are separate interrupt flags for the host write and read operation. Interrupt flags can also be used for polled operations. Dedicated status flags indicate 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, giving the host 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 automatically handling most conditions.

27.3.2.2.1. Clock Generation

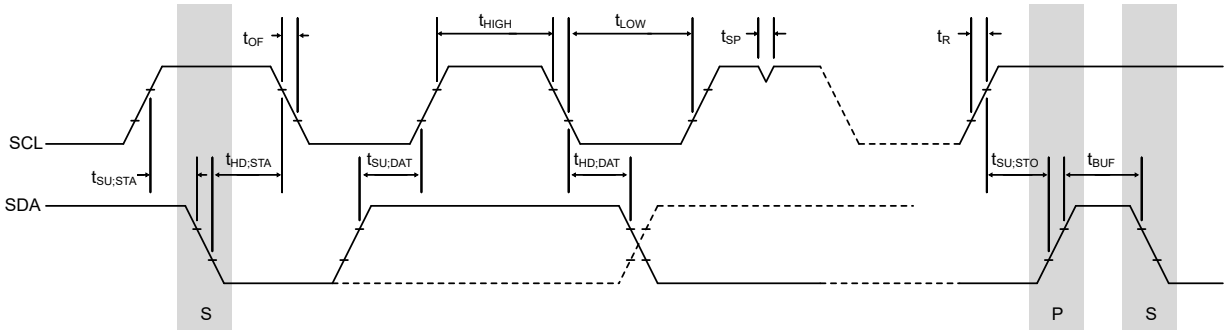
The TWI supports several transmission modes with different frequency limitations:

- Standard mode (Sm) up to 100 kHz
- Fast mode (Fm) up to 400 kHz
- Fast mode Plus (Fm+) up to 1 MHz

The Host Baud Rate (TWIn.MBAUD) register must be written to a value that will result in a TWI bus clock frequency equal to or less than those frequency limits, depending on the transmission mode.

The low (t_{LOW}) and high (t_{HIGH}) times are determined by the Host Baud Rate (TWIn.MBAUD) register, while the rise (t_R) and fall (t_{OF}) times are determined by the bus topology.

Figure 27-3. SCL Timing



- t_{LOW} is the low period of the SCL clock
- t_{HIGH} is the high period of the SCL clock
- t_R is determined by the bus impedance; for internal pull-ups. Refer to *Electrical Characteristics* for details.
- t_{OF} is the output fall time and 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:

$$f_{SCL} = \frac{1}{t_{LOW} + t_{HIGH} + t_{OF} + t_R}[Hz]$$

The SCL clock is designed to have a 50/50 duty cycle, where the low period of the duty cycle comprises t_{OF} and 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:

$$t_{LOW\_Fm} = t_{LOW\_Fm+} = \frac{BAUD + 6 + min(SCLDUTY, BAUD)}{f_{CLKPER}} - t_{OF} \qquad (3.1)$$

$$t_{LOW} = \frac{BAUD + 6}{f_{CLK\_PER}} - t_{OF} \qquad (3.2)$$

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:

$$BAUD = f_{CLK\_PER} \times (t_{LOW\_mode} + t_{OF}) - 3 \qquad (4)$$

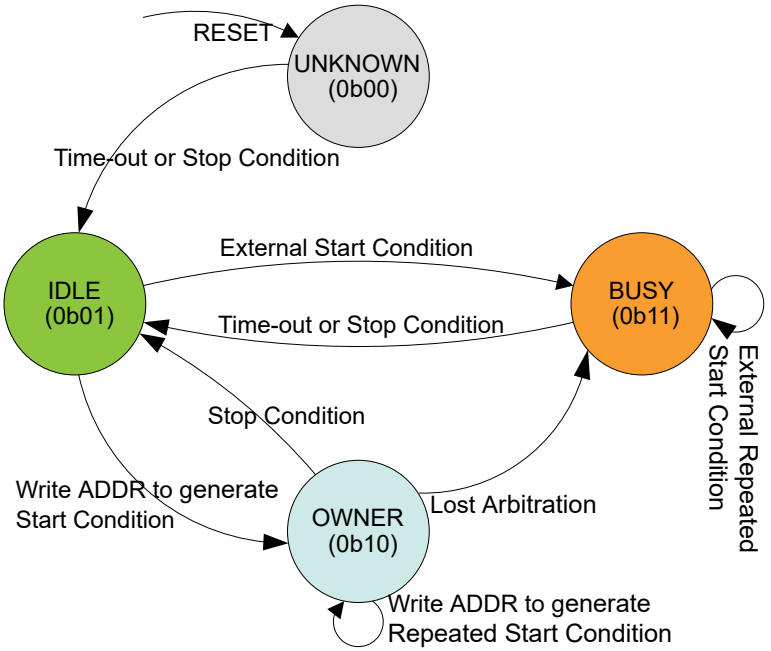
27.3.2.2.2. TWI Bus State Logic

The bus state logic continuously monitors the activity on the TWI bus when the host 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 Host 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 27-4. Bus State Diagram



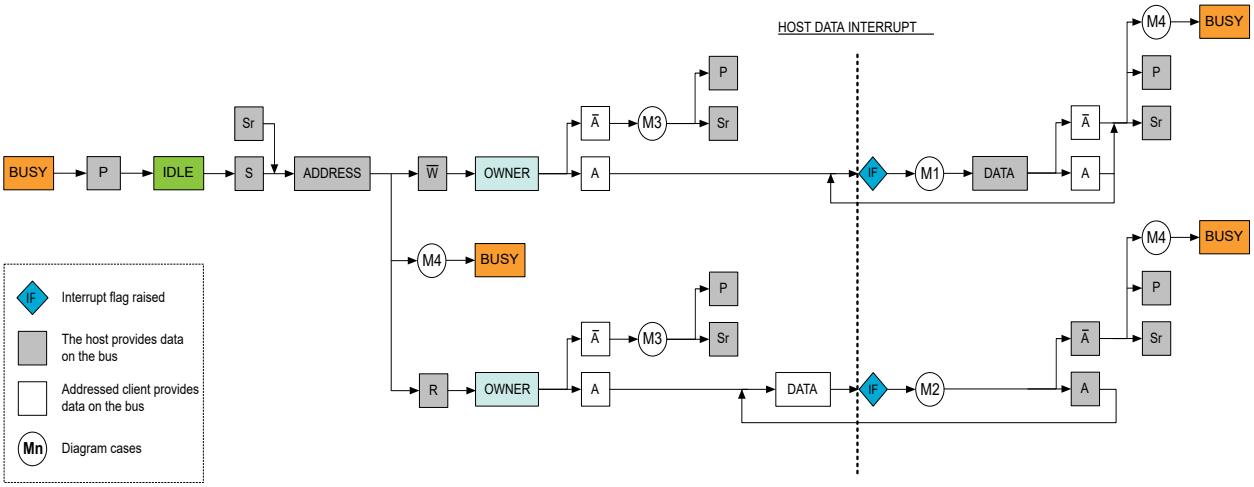
1. **Unknown**: The bus state machine is active when the TWI host is enabled. After enabling the TWI host, performing a system Reset, or disabling the TWI host, the bus state is Unknown.
2. **Idle**: The bus state machine can be forced to enter the Idle state by writing `0x1` to the Bus State (BUSSTATE) bit field. The bus state logic cannot be forced into any other state. If no state is set by the application software when the first Stop condition is detected, the bus state will become Idle. If the Inactive Bus Time-Out (TIMEOUT) bit field from the Host 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 host 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.

27.3.2.2.3. Transmitting Address Packets

The host starts performing a bus transaction when the Host Address (TWIn.MADDR) register is written with the client address and the R/$\overline{W}$ direction bit. The value of the MADDR register is then copied into the Host Data (TWIn.MDATA) register. If the bus state is Busy, the TWI host 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 27-5. TWI Host Operation



Case M1: Address Packet Transmit Complete - Direction Bit Set to '0'

If a client 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 Host Status (TWIn.MSTATUS) register.

The clock hold is active at this point, forcing the SCL low, which 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 client

Case M2: Address Packet Transmit Complete - Direction Bit Set to '1'

If a client device responds to the address packet with an ACK, the RXACK flag is set to '0', and the client can start sending data to the host without any delays because the client 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 client

Case M3: Address Packet Transmit Complete - Address not Acknowledged by Client

If no client 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 client is busy with other tasks or is in a sleep mode and cannot 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 Host Control B (TWIn.MCTRLB) register, which is the recommended action

Case M4: Arbitration Lost or Bus Error

If the arbitration is lost, the WIF and the Arbitration Lost (ARBLOST) flags in the Host 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 host 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 Host 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 Host Status (TWIn.MSTATUS) register

27.3.2.2.4. Transmitting Data Packets

Assuming the M1 case above, the TWI host can start transmitting data by writing to the Host Data (TWIn.MDATA) register, which also clears the Write Interrupt Flag (WIF). The host continuously monitors the bus for collisions and errors during the data transfer. After completing the data packet transfer, the WIF flag will be set to '1'.

If the transmission is successful and the host receives an ACK bit from the client, the Received Acknowledge (RXACK) flag will be set to '0', meaning that the client 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 Host Control B (TWIn.MCTRLB) register

If the transmission is successful and the host receives a NACK bit from the client, the RXACK flag will be set to '1', meaning that the client cannot 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 Host 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 host will lose the arbitration, the Arbitration Lost (ARBLOST) flag will be set to '1', and the bus state changes to Busy. An arbitration lost during the data packet transfer is treated the same way as the above M4 case.

The WIF, ARBLOST, BUSERR and RXACK flags are all located in the Host Status (TWIn.MSTATUS) register.

27.3.2.2.5. Receiving Data Packets

Assuming the M2 case above, the clock is released for one byte, allowing the client to put one byte of data on the bus. The host will receive one data byte from the client, 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 Host 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 execute successfully, as the arbitration can be lost during the transmission. If a collision is detected, the host loses the arbitration, the Arbitration Lost (ARBLOST) flag is set to '1', and the bus state changes to Busy. The Host 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 while transferring the data packet is treated as the above M4 case.

The RIF, CLKHOLD, ARBLOST and WIF flags are all located in the Host Status (TWIn.MSTATUS) register.

Note: The RIF and WIF flags are mutually exclusive and cannot be set simultaneously.

27.3.2.3. TWI Client Operation

The TWI client is byte-oriented with optional interrupts after each byte. There are separate interrupt flags for the client data and address/Stop recognition. Interrupt flags can also be used for polled operations. Dedicated status flags indicate ACK/NACK received, clock hold, collision, bus error, and R/$\overline{W}$ direction.

When an interrupt flag is set to '1', the SCL is forced low, giving the client 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 automatically handling most conditions.

The Address Recognition Mode (PMEN) bit in the Client Control A (TWIn.SCTRLA) register can be configured to allow the client to respond to all received addresses.

27.3.2.3.1. Receiving Address Packets

When the TWI is configured as a client, 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 client will ACK a correct address and store the address in the Client Data (TWIn.SDATA) register. If the received address is not a match, the client will not acknowledge or save the address but wait for a new Start condition.

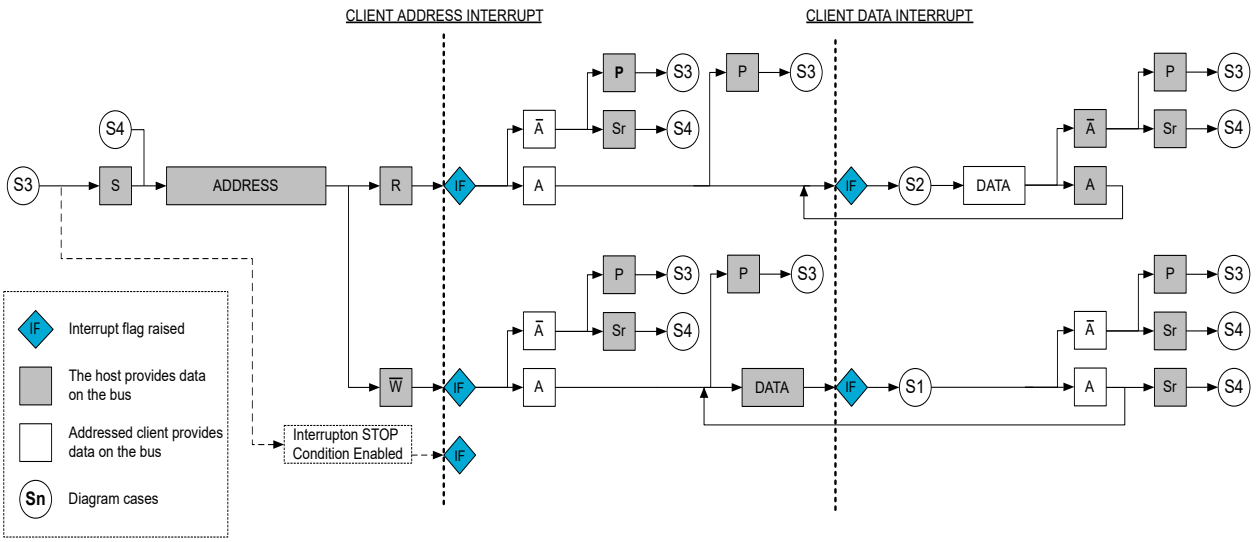
The Address or Stop Interrupt Flag (APIF) in the Client Status (TWIn.SSTATUS) register is set to '1' when a Start condition is followed by:

- A valid address matches the address stored in the Address (ADDR[7:1]) bit field in the Client Address (TWIn.SADDR) register
- The General Call Address (0x00) and the Address (ADDR[0]) bit in the Client Address (TWIn.SADDR) register is set to '1'
- A valid address matches the secondary address stored in the Address Mask (ADDRMASK) bit field, and the Address Mask Enable (ADDREN) bit is set to '1' in the Client Address Mask (TWIn.SADDRMASK) register
- Any address if the Address Recognition Mode (PMEN) bit in the Client Control A (TWIn.SCTRLA) register is set to '1'

A Start condition immediately followed by a Stop condition is an illegal operation, and the Bus Error (BUSERR) flag in the Client Status (TWIn.SSTATUS) register is set.

Depending on the Read/Write Direction (DIR) bit in the Client Status (TWIn.SSTATUS) register and the bus condition, one of four cases (S1 to S4) arises after the reception of the address packet.

Figure 27-6. TWI Client Operation



Case S1: Address Packet Accepted - Direction Bit Set to '0'

If an ACK is sent by the client after the address packet is received, and the Read/Write Direction (DIR) bit in the Client Status (TWIn.SSTATUS) register is set to '0', the host indicates a write operation.

The clock hold is active at this point, forcing the SCL low and stretching 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 host

Case S2: Address Packet Accepted - Direction Bit Set to '1'

If an ACK is sent by the client after the address packet is received, and the DIR bit is set to '1', the host indicates a read operation, and the Data Interrupt Flag (DIF) in the Client 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 host

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 Client Status (TWIn.SSTATUS) register.

The software can prepare to:

- Wait until a new address packet has been addressed to it

Case S4: Collision

If the client cannot send a high-level data bit or a NACK, the Collision (COLL) bit in the Client Status (TWIn.SSTATUS) register is set to '1'. The client will commence ordinary operation, except no low values will be shifted out on the SDA. The data and acknowledge output from the client 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.

27.3.2.3.2. Receiving Data Packets

Assuming the S1 case above, the client must be ready to receive data. When a data packet is received, the Data Interrupt Flag (DIF) in the Client Status (TWIn.SSTATUS) register is set to '1'. The action selected by the Acknowledge Action (ACKACT) bit in the Client 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 client is ready to receive more data
- Respond with a NACK by writing '1' to the ACKACT bit, indicating that the client cannot receive any more data and the host must issue a Stop or repeated Start condition

27.3.2.3.3. Transmitting Data Packets

Assuming the S2 case above, the client can start transmitting data by writing to the Client Data (TWIn.SDATA) register. When a data packet transmission is completed, the Data Interrupt Flag (DIF) in the Client Status (TWIn.SSTATUS) register is set to '1'.

The software can prepare to take one of the following actions:

- Check if the host responded with an ACK by reading the Received Acknowledge (RXACK) bit from the Client Status (TWIn.SSTATUS) register, and start transmitting new data packets
- Check if the host responded with a NACK by reading the RXACK bit and stop transmitting data packets. The host must send a Stop or repeated Start condition after the NACK.

27.3.3. Additional Features

27.3.3.1. SMBus

The Inactive Bus Time-Out (TIMEOUT) bit field from the Host Control A (TWIn.MCTRLA) register must be configured if the TWI is used in an SMBus environment. It is necessary to configure the TIMEBASE bit field in the CLKCTRL.TIMEBASE register to comply with the SMBus standard. Refer to the CLKCTRL - Clock Controller section for more information.

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.

27.3.3.1.1. Compliance to SMBus Specifications

Hardware-Specific Restrictions

Section 2 of the SMBus 2.0 specifications states that powered-down devices must provide no leakage path to ground. There are ESD diodes placed between V_{DD} and the pads used for SCL and SDA on this device. Assuming V_{DD} is equivalent to ground when powered down, these ESD diodes provide a path to ground.

Implementation in Software

The following elements of the SMBus 2.0 specifications are not implemented in hardware:

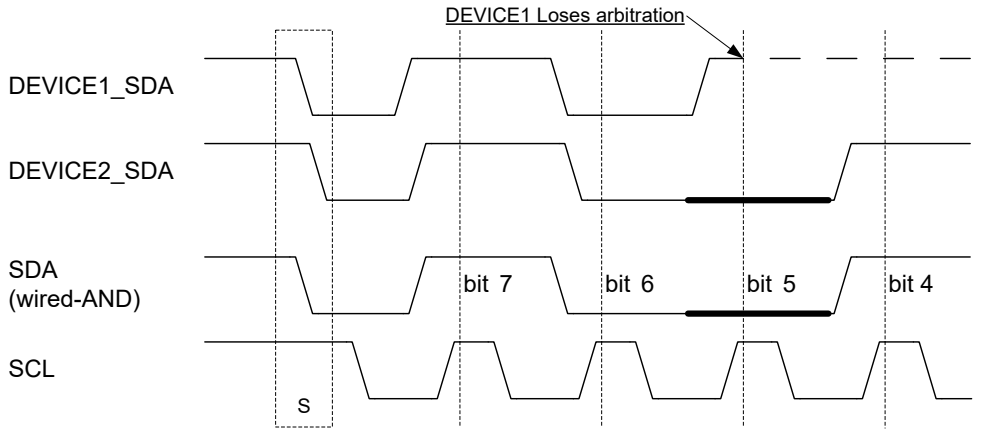
- Table 1 of the SMBus 2.0 specifications gives a maximum clock low timeout ($T_{timeout}$) of 25-35 ms, which can be implemented by connecting the SCL pin to the TCB peripheral using the Event System. Configure the TCB in Time-Out Check mode with the desired time-out value.
- Layer 3 (network layer) features such as packet error checking (PEC), address resolution protocol (ARP). These can be implemented in software if required.

27.3.3.2. Multi-Host

A host can start a bus transaction only if it has detected that the bus is in the Idle state. If multiple hosts are on the bus, other devices may try to initiate a transaction simultaneously, resulting in multiple hosts owning the bus. The TWI solves this problem by using an arbitration scheme where the host 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 Host Status (TWIn.MSTATUS) register will change to Busy. The hosts that lose the arbitration must wait until the bus becomes Idle before attempting to reacquire 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 27-7. TWI Arbitration



27.3.3.3. 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 host, the Smart mode will automatically send the ACK action as soon as the Host Data (TWIn.MDATA) register is read. This feature is only active when the Acknowledge Action (ACKACT) bit in the Host Control B (TWIn.MCTRLB) register is set to ACK. The TWI host will not generate a NACK after the MDATA register is read if the ACKACT bit is set to NACK. This feature is enabled when the Smart Mode Enable (SMEN) bit in the Host Control A (TWIn.MCTRLA) register is set to '1'.

For the TWI client, the Smart mode will automatically send the ACK action as soon as the Client Data (TWIn.SDATA) register is read. The Smart mode will automatically set the Data Interrupt Flag (DIF) to '0' in the Client 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 Client Control A (TWIn.SCTRLA) register is set to '1'.

27.3.3.4. Dual Mode

The TWI supports Dual mode operation where the host and the client will operate simultaneously and independently. In this case, the Control A (TWIn.CTRLA) register will configure the TWI host, and the Dual Mode Control (TWIn.DUALCTRL) register will configure the TWI client. See the Initialization section for more details about the host configuration.

If used, the following bits must be configured before enabling the TWI Dual mode:

- The SDA Hold Time (SDAHOLD) bit field in the DUALCTRL register
- The FM Plus Enable (FMPEN) bit from the DUALCTRL register

The Dual mode can be enabled by writing a '1' to the Dual Control Enable (ENABLE) bit in the DUALCTRL register.

27.3.3.5. Quick Command Mode

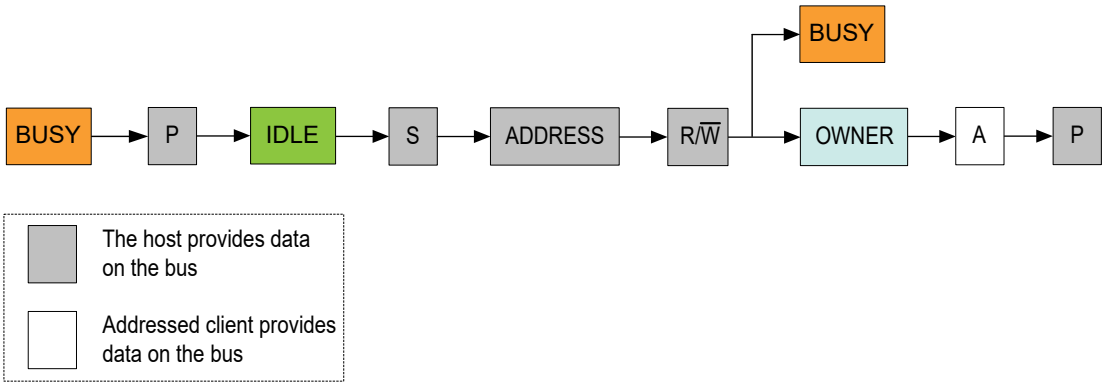
In Quick Command mode, the R/$\overline{W}$ bit from the address packet denotes the command. This mode is enabled by writing '1' to the Quick Command Enable (QCEN) bit in the Host Control A (TWIn.MCTRLA) register. There are no data sent or received.

The Quick Command mode is SMBus specific, using the R/$\overline{W}$ bit to turn a device function on/off or enable/disable a low-power Standby mode. This mode can be enabled to auto-trigger operations and reduce software complexity.

After the host receives an ACK from the client, 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 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 Host 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 Host Status (TWIn.MSTATUS) register.

Figure 27-8. Quick Command Frame Format



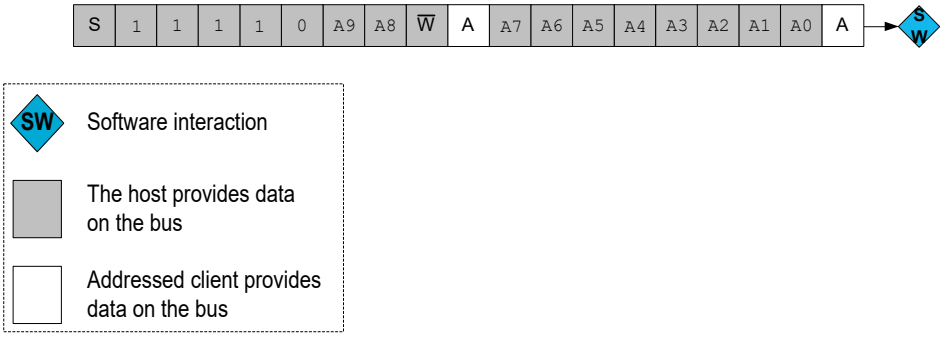
27.3.3.6. 10-Bit Address

Regardless of whether the transaction is a read or write, the host must start by sending the 10-bit address with the R/$\overline{W}$ direction bit set to '0'.

The client address match logic supports recognition of 7-bit addresses and General Call Address. The Client Address (TWIn.SADDR) register is used by the client address match logic to determine if a host device has addressed the TWI client.

The TWI client address match logic only supports the 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 Client Address (TWIn.SADDR) register are 0b11110. Thus, the first byte will consist of five indication bits, the two Most Significant bits (MSbs) of the 10-bits address, and the R/$\overline{W}$ direction bit. The Least Significant Byte (LSB) of the address that follows from the host will come in the form of a data packet.

Figure 27-9. 10-Bit Address Transmission



27.3.4. Interrupts

Table 27-1. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
Client	TWI Client 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'
Host	TWI Host 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 Host Status (TWIn.MSTATUS) register or the Client 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.

27.3.5. Sleep Mode Operation

The bus state logic and the address recognition hardware continue to operate in all sleep modes. If the TWI client is in a sleep mode and a Start condition followed by the client address is detected, clock stretching is active during the wake-up period until the main clock is available. The TWI host will stop operation in all sleep modes. When the Dual mode is active, the TWI peripheral will wake up only when the Start condition is received by the TWI client.

27.3.6. Debug Operation

During run-time debugging, the TWI will continue its ordinary operation. Halting the CPU in Debugging mode will stop the normal operation of the TWI. The TWI can be forced to operate with a 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 Host Data (TWIn.MDATA) register or the Client 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.

27.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		INPUTLVL		SDASETUP	SDAHOLD[1:0]		FMPEN	FMEN
0x01	DUALCTRL	7:0		INPUTLVL			SDAHOLD[1:0]		FMPEN	ENABLE
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

27.5. Register Description

27.5.1. Control A

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

Bit	7	6	5	4	3	2	1	0
		INPUTLVL		SDASETUP	SDAHOLD[1:0]		FMPEN	FMEN
Access		R/W		R/W	R/W	R/W	R/W	R/W
Reset		0		0	0	0	0	0

Bit 6 – INPUTLVL Input Voltage Transition Level
This bit selects between I^2C and SMBUS.

Value	Name	Description
0	I2C	I^2C input voltage transition level
1	SMBUS	SMBus 3.0 input voltage transition level

Bit 4 – SDASETUP SDA Setup Time
This bit controls the number of cycles the SCL is stretched to ensure sufficient setup time on the SDA out signal. This bit is used when operating in client mode.

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
0x2	300NS	Meets the SMBus 2.0 specifications under typical conditions
0x3	500NS	Meets the SMBus 2.0 across all corners

Bit 1 – FMPEN Fast-mode Plus Enable
Writing a '1' to this bit selects the 1 MHz bus speed for the TWI in default configuration or the TWI host in Dual mode configuration.

Value	Name	Description
0	OFF	Operating in Standard mode or Fast mode
1	ON	Operating in Fast mode Plus

Bit 0 – FMEN Fast-mode Enable
Writing this bit to '1' when operating in host mode adjusts the SCL duty cycle to comply with I^2C specification in Fast mode. This bit has no effect in client mode or if Fast-mode Plus is enabled.

Value	Name	Description
0	OFF	SCL duty cycle operating according to Sm specification
1	ON	SCL duty cycle operating according to Fm specification

27.5.2. Dual Mode Control Configuration

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

Bit	7	6	5	4	3	2	1	0
		INPUTLVL			SDAHOLD[1:0]		FMPEN	ENABLE
Access		R/W			R/W	R/W	R/W	R/W
Reset		0			0	0	0	0

Bit 6 – INPUTLVL Input Voltage Transition Level
This bit selects between I^2C and SMBUS. This bit is ignored if the Dual mode is not enabled.

Value	Name	Description
0	I2C	I^2C input voltage transition level
1	SMBUS	SMBus 3.0 input voltage transition level

Bits 3:2 – SDAHOLD[1:0] SDA Hold Time
This bit field selects the SDA hold time for the TWI client. See also the *Electrical Characteristics* section. This bit field is ignored if the Dual mode is not enabled.

Value	Name	Description
0x0	OFF	Hold time OFF
0x1	50NS	Short hold time
0x2	300NS	Meets the SMBus 2.0 specifications under typical conditions
0x3	500NS	Meets the SMBus 2.0 across all corners

Bit 1 – FMPEN FM Plus Enable
Writing a '1' to this bit selects the 1 MHz bus speed for the TWI client. This bit is ignored if the Dual mode is not enabled.

Value	Name	Description
0	OFF	Operating in Standard mode or Fast mode
1	ON	Operating in Fast mode Plus

Bit 0 – ENABLE Dual Control Enable
Writing a '1' to this bit will enable the Dual mode configuration.

27.5.3. 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
Refer to the *Debug Operation* section for 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

27.5.4. Host 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 host read interrupt will only be generated 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 Host Status (TWIn.MSTATUS) register. The RIF flag is set to '1' when the host read interrupt occurs.

Bit 6 – WIEN Write Interrupt Enable
A TWI host write interrupt will only be generated 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 Host Status (TWIn.MSTATUS) register. The WIF flag is set to '1' when the host write interrupt occurs.

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 client acknowledges the address, the corresponding Read Interrupt Flag (RIF) or Write Interrupt Flag (WIF) will be set depending on the value of the R/$\overline{W}$ bit.
The software must issue a Stop command by writing to the Command (MCMD) bit field in the Host Control B (TWIn.MCTRLB) register.

Bits 3:2 – TIMEOUT[1:0] Inactive Bus Time-Out
Setting this bit field to a non-zero 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
0x2	100US	100 µs
0x3	200US	200 µs

Bit 1 – SMEN Smart Mode Enable
Writing a '1' to this bit enables the Host Smart mode. When the Smart mode is enabled, the existing value in the Acknowledge Action (ACKACT) bit from the Host Control B (TWIn.MCTRLB) register is sent immediately after reading the Host Data (TWIn.MDATA) register.

Bit 0 – ENABLE Enable TWI Host
Writing a '1' to this bit enables the TWI as host.

27.5.5. Host 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 host and the bus states changes to Idle. The TWI will transmit invalid data if the Host Data (TWIn.MDATA) register is written before the Host Address (TWIn.MADDR) register. Writing to Host Address (TWIn.MADDR) and Host Data (TWIn.MDATA) after a Flush will cause a transaction to start as soon as hardware detects SCL bus free.
Writing a '1' to this bit generates a strobe for one clock cycle, disabling the host and then re-enabling the host. Writing a '0' to this bit has no effect.

Bit 2 – ACKACT Acknowledge Action

The ACKACT[1] bit represents the behavior in the Host mode under certain conditions defined by the bus state and the software interaction. If the Smart Mode Enable (SMEN) bit in the Host Control A (TWIn.MCTRLA) register is set to '1', the acknowledge action is performed when the Host Data (TWIn.MDATA) register is read. Otherwise a command must be written to the Command (MCDM) bit field in the Host Control B (TWIn.MCTRLB) register.
The acknowledge action is not performed when the Host Data (TWIn.MDATA) register is written since the host 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 host operation, as defined by the table below.

Table 27-2. 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

Notes:

1. The ACKACT bit and the MCMD bit field can be written simultaneously.
2. For a host write operation, the TWI will wait for new data to be written to the Host Data (TWIn.MDATA) register.

27.5.6. Host 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 host byte read operation is completed.
The RIF flag can generate a host read interrupt. Find more information in the description of the Read Interrupt Enable (RIEN) bit in the Host Control A (TWIn.MCTRLA) register.
This flag automatically clears when some TWI registers are accessed. Any of the following methods can be used to clear the RIF flag:

1. Writing a '1' to it.
2. Writing to the Host Address (TWIn.MADDR) register.
3. Writing/Reading the Host Data (TWIn.MDATA) register.
4. Writing to the Command (MCMD) bit field from the Host Control B (TWIn.MCTRLB) register.

Bit 6 – WIF Write Interrupt Flag
This flag is set to '1' when a host address transmit or byte write operation is completed, regardless of any occurrence of a bus error or arbitration lost condition.
The WIF flag can generate a host write interrupt. Find more information in the description of the Write Interrupt Enable (WIEN) bit in the Host Control A (TWIn.MCTRLA) register.
This flag can be cleared using any of the methods described above for the RIF flag.

Bit 5 – CLKHOLD Clock Hold
When this bit is read as '1', it indicates that the host currently holds the SCL low, stretching the TWI clock period.
This bit can be cleared using any of the methods described above 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 client was ACK, and the client is ready for more data.
When this flag is read as '1', it indicates that the most recent Acknowledge bit from the client was NACK, and the client 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 host 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 Host Address (TWIn.MADDR) register.

The TWI bus error detector is part of the TWI host circuitry. For bus errors to be detected, the TWI host 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. Writing 0x1 to this bit field will force the bus state to IDLE. All other values will be ignored.

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

27.5.7. Host 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 host is disabled. The host can be disabled by writing '0' to the Enable TWI Host (ENABLE) bit from the Host Control A (TWIn.MCTRLA) register.
Refer to the *Clock Generation* section for more information on how to calculate the frequency of the SCL.

27.5.8. Host 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 client 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 read access does not trigger the host logic to perform any bus protocol-related operations.

The host control logic uses bit 0 of this register as the R/$\overline{W}$ direction bit.

27.5.9. Host 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 host'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 to the MDATA register will command the host to perform a byte transmit operation on the bus, directly followed by receiving the Acknowledge bit from the client. This is independent of the Acknowledge Action (ACKACT) bit from the Host 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 Host Control A (TWIn.MCTRLA) register is set to '1', read access to the MDATA register will command the host to perform an acknowledge action. This is dependent on the setting of the Acknowledge Action (ACKACT) bit from the Host Control B (TWIn.MCTRLB) register.

Notes:

1. The WIF and RIF flags are automatically cleared 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 Host Status (TWIn.MSTATUS) register.

27.5.10. Client 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 Client Status (TWIn.SSTATUS) register.
A TWI client data interrupt will only be generated if this bit, the DIF flag, and the Global Interrupt Enable (I) bit in the 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 Client Status (TWIn.SSTATUS) register.
A TWI client address or stop interrupt will only be generated if this bit, the APIF flag, and the Global Interrupt Enable (I) bit in the Status (CPU.SREG) register are all '1'.
Notes:

1. The client stop interrupt shares the interrupt flag and vector with the client address interrupt.
2. The Stop Interrupt Enable (PIEN) bit in the Client 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 Client 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 Client Status (TWIn.SSTATUS) register to be set when a Stop condition occurs. The main clock frequency must be at least four times the SCL frequency to use this feature.

Bit 2 – PMEN Address Recognition Mode
If this bit is written to '1', the client address match logic responds to all received addresses.
If this bit is written to '0', the address match logic uses the Client Address (TWIn.SADDR) register to determine which address to recognize as the client's address.

Bit 1 – SMEN Smart Mode Enable
Writing this bit to '1' enables the client Smart mode. When the Smart mode is enabled, issuing a command by writing to the Command (SCMD) bit field in the Client Control B (TWIn.SCTRLB) register or accessing the Client Data (TWIn.SDATA) register resets the interrupt, and the operation continues. If the Smart mode is disabled, the client always waits for a new client command before continuing.

Bit 0 – ENABLE Enable TWI Client
Writing this bit to '1' enables the TWI client.

27.5.11. Client 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 TWI client under certain conditions defined by the bus protocol state and the software interaction. If the Smart Mode Enable (SMEN) bit in the Client Control A (TWIn.SCTRLA) register is set to '1', the acknowledge action is performed when the Client Data (TWIn.SDATA) register is read. Otherwise a command must be written to the Command (SCMD) bit field in the Client Control B (TWIn.SCTRLB) register.
The acknowledge action is not performed when the Client Data (TWIn.SDATA) register is written since the client 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 client operation as defined by the table below.

Table 27-3. Command Settings

Value	Name	DIR	Description	
0x0	NOACT	X	No action	
0x1	—	X	Reserved	
0x2	COMPTRANS	$\overline{W}$	Execute Acknowledge Action succeeded by waiting for any Start (S/Sr) condition	Used to complete a transaction
		R	Wait for any Start (S/Sr) condition	
0x3	RESPONSE	$\overline{W}$	Execute Acknowledge Action succeeded by the reception of the next byte	
		R	Used in response to an address interrupt (APIF): Execute Acknowledge Action succeeded by client data interrupt.	
			Used in response to a data interrupt (DIF): Execute a byte read operation followed by Acknowledge Action.	

Note: 1. The ACKACT bit and the SCMD bit field can be written simultaneously. The ACKACT will be updated before the command is triggered.

27.5.12. Client 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 client byte transmit or receive operation is completed without bus errors. This flag can be set to '1' with an unsuccessful transaction in case of collision detection. Find more information in the description of the Collision (COLL) bit.
The DIF flag can generate a client data interrupt. Find more information in the description of the Data Interrupt Enable (DIEN) bit in the Client Control A (TWIn.SCTRLA) register.
This flag automatically clears when some TWI registers are accessed. Any of the following methods can be used to clear the DIF flag:

1. Writing/Reading the Client Data (TWIn.SDATA) register.
2. Writing to the Command (SCMD) bit field in the Client Control B (TWIn.SCTRLB) register.

Bit 6 – APIF Address or Stop Interrupt Flag
This flag is set to '1' when the client address has been received or by a Stop condition.
The APIF flag can generate a client address or stop interrupt. Find more information in the description of the Address or Stop Interrupt Enable (APIEN) bit in the Client Control A (TWIn.SCTRLA) register.
This flag can be cleared using any of the methods described for the DIF flag.

Bit 5 – CLKHOLD Clock Hold
When this bit is read as '1', it indicates that the client 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 host was ACK.
When this flag is read as '1', it indicates that the most recent Acknowledge bit from the host was NACK.

Bit 3 – COLL Collision
When this bit is read as '1', it indicates that the client 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 because of the internal completion of an unsuccessful transaction.
2. Transmit the NACK bit. The collision occurs because the client 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 is 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 host circuitry. For the bus errors to be detected by the client, the TWI Dual mode or the TWI host must be enabled, and the main clock frequency must be at least four times the SCL frequency. The TWI Dual mode can be enabled by writing '1' to the ENABLE bit in the TWIn.DUALCTRL register. The TWI host can be enabled by writing '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 host TWI device.

When this bit is read as '1', it indicates that a host read operation is in progress.

When this bit is read as '0', it indicates that a host write operation is in progress.

Bit 0 – AP Address or Stop

When the TWI client 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

27.5.13. Client 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 Client Address (TWIn.SADDR) register is used by the client address match logic to determine if a host device has addressed the TWI client. If an address packet is received, the Address or Stop Interrupt Flag (APIF) and the Address or Stop (AP) bit in the Client Status (TWIn.SSTATUS) register are set to '1'.

The upper seven bits (ADDR[7:1]) of the TWIn.SADDR register represent the main client address. The TWIn.SADDR register's Least Significant bit (ADDR[0]) is used for recognition of the General Call Address (0x00) of the I^2C protocol. This feature is enabled when this bit is set to '1'.

27.5.14. Client 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 client data register.

Reading valid data or writing data to be transmitted can only be achieved when the SCL is held low by the client (i.e., when the client CLKHOLD bit is set to '1'). It is unnecessary to check the Clock Hold (CLKHOLD) bit from the Client 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 Client Control A (TWIn.SCTRLA) register is set to '1', read access to the SDATA register, when the clock hold is active, auto-triggers bus operations and commands the client to perform an acknowledge action. This is dependent on the setting of the Acknowledge Action (ACKACT) bit from the Client Control B (TWIn.SCTRLB) register.

27.5.15. Client 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 Client Address mask. Each of the bits in the Client Address Mask (TWIn.SADDRMASK) register can mask (disable) the corresponding address bits in the TWI Client 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 Client Address (TWIn.SADDR) register. In other words, masked bits will always match, making it possible to recognize the ranges of addresses.
If the ADDREN bit is written to '1', the Client Address Mask (TWIn.SADDRMASK) register can be loaded with a second client address in addition to the Client Address (TWIn.SADDR) register. In this mode, the client will have two unique addresses -- one in the Client Address (TWIn.SADDR) register and the other in the Client 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 client address match logic responds to the two unique addresses in the client TWIn.SADDR and TWIn.SADDRMASK registers.

28. CRCSCAN - Cyclic Redundancy Check Memory Scan

28.1. Features

- CRC-16-CCITT or CRC-32 (IEEE 802.3)
- Scan of the boot section, application code section, or application data section
- Scan of the Boot ROM during system start-up
- User-configurable scan of the boot section of the flash during system start-up
- Scanning is performed when the CPU is in IDLE sleep mode only
- Optional periodic interrupt after every 32 words scanned
- Optional interrupt when scanning is completed
- Selectable Non-Maskable Interrupt (NMI) Trigger on Failure
- Manual mode with application-supplied input data
- Error injection to test functionality

28.2. Overview

A Cyclic Redundancy Check (CRC) takes a data stream of bytes from the Flash (either the boot section, application code section, or application data 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 being checked must contain a correct pre-calculated 16- or 32-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 A (CRCSCAN.STATUSA) register will indicate that it failed. The CRCSCAN can be configured to generate a Non-Maskable Interrupt (NMI) if there is a checksum mismatch.

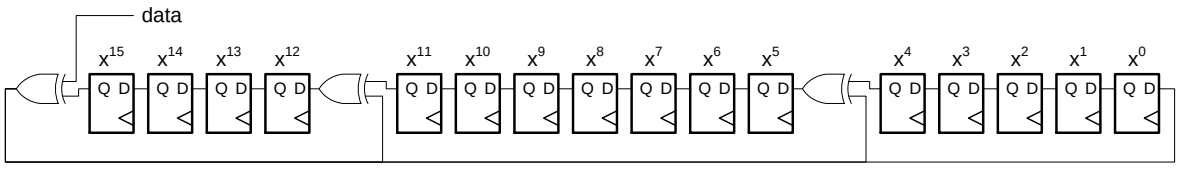
The CRC generator supports CRC-16-CCITT and CRC-32 (IEEE 802.3).

Polynomial:

- CRC-16-CCITT: $x^{16} + x^{12} + x^5 + 1$, initial value = `0xFFFF`
- CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$, initial value = `0xFFFF_FFFF`

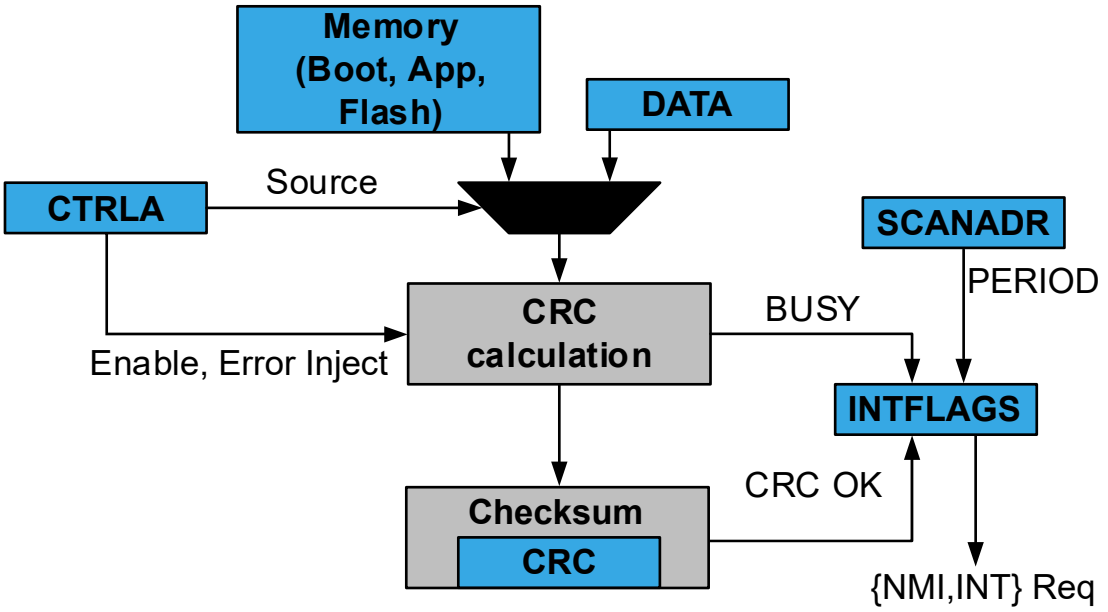
The CRC reads byte-by-byte the content of the section(s) it is set up to check, starting with byte 0, and generates a new checksum for each byte. The byte is sent through a shift register, as depicted below, starting with the most significant bit. The CRC passes if the last bytes in the section contain the correct checksum. See the *Checksum* section for instruction on how to place the checksum.

Figure 28-1. CRC Implementation Description



28.2.1. Block Diagram

Figure 28-2. Cyclic Redundancy Check Block Diagram



28.3. Functional Description

28.3.1. Initialization

To enable a CRC scan in software (or via the debugger):

Sleep Mode

1. Select between CRC32 and CRC16 modes with the CRC Mode Select (CRCSEL) bit in the Control A (CRCSCAN.CTRLA) register.
2. Write the CRC Source (SRC) bit field in the CRCSCAN.CTRLA register to select the desired region of the Flash to be scanned.
3. Enable the CRCSCAN by writing a '1' to the ENABLE bit in the CRCSCAN.CTRLA register. The CRC Busy (BUSY) bit in the Status A (CRCSCAN.STATUSA) register will be set.
4. The CRC will begin scanning when the CPU enters the Idle sleep mode.

Note: If sectioning the Flash and using interrupt-driven CRCSCAN, the interrupt vector table may need to be moved from its default location at the start of the Application Code (APPCODE) section of the Flash. For information on changing the expected location of the vector table, refer to the *CPU Interrupt Controller* section.

Manual Mode

1. Select CRC32 or CRC16 mode with the CRCSEL bit in the CRCSCAN.CTRLA register.
2. Write the SRC bit field in the CRCSCAN.CTRLA register to MANUAL.
3. Enable the CRCSCAN by writing a '1' to the ENABLE bit in the CRCSCAN.CTRLA register.
4. Write the byte to the Input Data (CRCSCAN.DATA) register for the CRC calculation.
5. Read the CRC result from the CRC Result (CRCSCAN.CRC) register.

28.3.2. Operation

28.3.2.1. Operation Modes

The CRCSCAN peripheral supports Scan-In-Sleep mode and the Manual mode operation. The mode is set in the CRC Source (SRC) bit field of the Control A (CRCSCAN.CTRLA) register. For Idle sleep mode operation, configure SRC with the desired section of the Flash to scan (BOOT, CODE or DATA). For the Manual mode operation, set SRC to MANUAL.

Note: The CRCSCAN can be set up to scan the BOOT section in Flash before the device exits Reset. If this scan fails, the CPU is not allowed to start normal code execution. This feature is enabled and controlled by the CRC Boot (CRCBOOT) and CRC Select (CRCSEL) bits in the Fuse System Configuration 0 (FUSE.SYSCFG0) fuse. Refer to the *Fuse* section for more information.

Scan-In-Sleep Mode

In Scan-In-Sleep mode, the CRCSCAN peripheral has access to the Flash only when the CPU is in Idle sleep mode. If a scan is enabled, the scan will start when the CPU enters Idle sleep mode, and the scan will pause when the CPU wakes up. If the CPU goes to sleep again, the scan will resume at the address where it left off. This operation will continue until the entire selected memory range has been scanned. An interrupt can optionally be triggered when the scan is complete.

The CRCSCAN peripheral has a built-in counter that, when enabled, will set an interrupt flag every 32 scanned words. This counter is enabled by writing to the Enable Periodic Timer (PEREN) bit in the CRCSCAN.CTRLA register. Its interrupt can be enabled by writing to the Scan Period Done Interrupt Enable (PERIOD) bit in the Interrupt Control (CRCSCAN.INTCTRL) register. If enabled, the interrupt will cause the CPU to leave Idle sleep and allow the application to do various housekeeping tasks between scanning. Continue scanning by clearing the Scan Period Done (PERIOD) bit in the Interrupt Flags (CRCSCAN.INTFLAGS) register and re-enter the Idle sleep mode.

Other sources can wake up the CPU even when the Period Interrupt is enabled. In this case, the periodic timer is frozen while the CPU is awake, and will continue where it froze after re-entering Idle sleep mode.

If the periodic timer is enabled without the period interrupt enabled, the CRC will pause scanning of more Flash words when the flag is set without waking up the CPU. Upon waking up from a different source clearing the PERIOD flag in INTFLAGS will cause the scan to continue.

A CRC error detected during the scan will set an NMI request if the Enable NMI Trigger (NMIEN) bit in the CRCSCAN.CTRLA register is '1'.

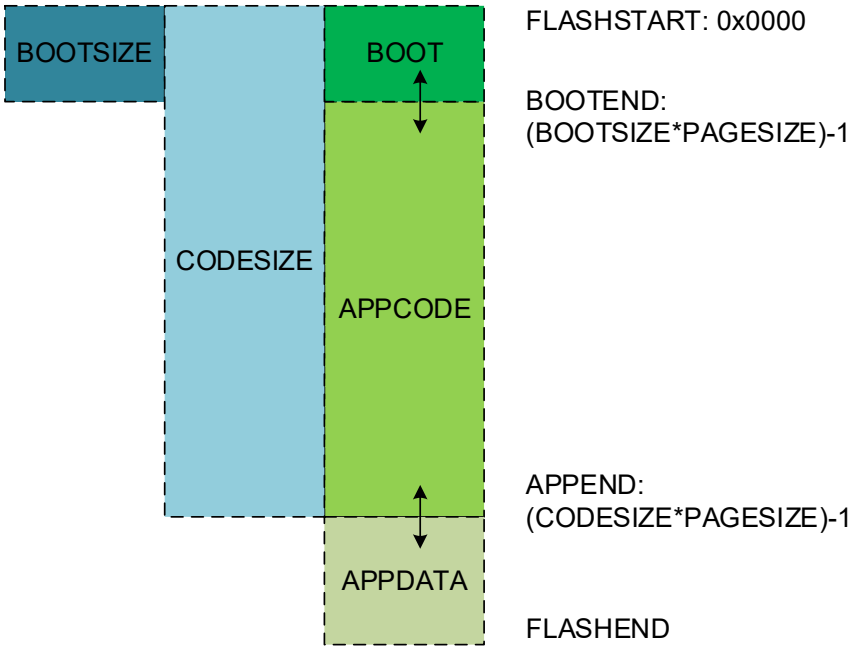
Manual Mode

The Manual mode allows the application to calculate CRC on a data stream by writing the bytes in the stream to the Input Data (CRCSCAN.DATA) register. CRC-16 or CRC-32 is used in the MANUAL mode, depending on the value of the CRC Source Select (CRCSEL) bit in the CRCSCAN.CTRLA register. One clock cycle is required to calculate CRC on a written byte, so data can be written continuously to the CRCSCAN.DATA register without polling the CRC Busy (BUSY) bit in the Status A (CRCSCAN.STATUSA) register between writes. The result of the CRC operation can be read from the CRC Result (CRCSCAN.CRC) register.

28.3.2.2. Configuration of Scan Region

The Flash can be divided into sections, as illustrated in the figure below.

Figure 28-3. Division of Flash Into Sections



Determine the section sizes in blocks corresponding to the Flash memory page size. The sizes of these sections are set by the Boot Size (FUSE.BOOTSIZE) and Code Size (FUSE.CODESIZE) fuses. Refer to the *Fuse* section for more information. The Boot Loader Code (BOOT) section stretches from FLASHSTART to BOOTEND, the Application Code (APPCODE) section stretches from BOOTEND to APPEND, and the remaining is the Application Data (APPDATA) section.

If FUSE.BOOTSIZE is written to '0', the entire Flash is considered the BOOT section. If FUSE.CODESIZE is written to '0' and FUSE.BOOTSIZE > 0, the APPCODE section runs from BOOTEND to the end of the Flash, meaning no APPDATA section.

Which region to scan is configured in the CRC Source (SRC) bit field in the CRCSCAN.CTRLA register.

Starting a scan of an unconfigured Flash region will set the ERROR bit in the CRCSCAN.STATUSA register. For starting a scan with SRC set to CODE or DATA when BOOTSIZE='0' (the whole Flash is BOOT) or SRC set to DATA when CODESIZE='0' (no DATA section) will both result in CRC failure.

Table 28-1. Configuration of Scan Regions

CRC Source	Scan Region	BOOTSIZE	CODESIZE
BOOT	BOOT	> 0	-
	The entire Flash	0	-
CODE	APPCODE	> 0	-
DATA	APPDATA	> 0	BOOTEND + APPEND < FLASHEND

28.3.2.3. Checksum

The pre-calculated checksum must be present at the end of the section to be scanned. The checksum must be stored in the last bytes of the BOOT section to check the BOOT section and similarly for the APPCODE and APPDATA sections. The checksum for CRC32 has opposite endianness compared to CRC16. *Table 1-2* and *Table 1-3* show explicitly how to store the checksum for the different sections. For additional information on partitioning the Flash section. Refer to the *Configuration of Scan Region* section.

Table 28-2. Placement of the Pre-Calculated Checksum for CRC16 in Flash

Section to Check	CHECKSUM[15:8]	CHECKSUM[7:0]
BOOT	BOOTEND - 1	BOOTEND
APPCODE	APPEND - 1	APPEND
APPDATA	FLASHEND - 1	FLASHEND

Table 28-3. Placement of the Pre-Calculated Checksum for CRC32 in Flash

Section to Check	CHECKSUM[7:0]	CHECKSUM[15:8]	CHECKSUM[23:16]	CHECKSUM[31:24]
BOOT	BOOTEND - 3	BOOTEND - 2	BOOTEND - 1	BOOTEND
APPCODE	APPEND - 3	APPEND - 2	APPEND - 1	APPEND
APPDATA	FLASHEND - 3	FLASHEND - 2	FLASHEND - 1	FLASHEND

Defining the Pre-Calculated Checksum

The pre-calculated checksum can be determined by using a CRC algorithm and the parameters specified in *Table 1-4*. The calculation should start at the region start and end at the address before the pre-calculated checksum's placement. Manual mode can be used to compare the checksum at the end of a region with known data, ensuring correctly configured calculation.

Table 28-4. Parameters for Checksum Calculation

CRC Selected	Polynomial	Initial Value	Width	XOR Value
CRC16	0x1021	0xFFFF	2	N/A
CRC32	0x4C11DB7	0xFFFF_FFFF	4	0xFFFF_FFFF

For more information on Checksum calculation and use, refer to the *Hexmate* section in the *MPLAB® XC8 C Compiler User's Guide for PIC® MCU* document.

28.3.2.4. Error Injection

The CRCSCAN error injection triggers a rapid CRC scan, resulting in a CRC error.

Writing a '1' to the Inject CRC Error (EINJ) bit in the Control B (CRCSCAN.CTRLB) register will trigger a scan that starts when the CPU enters Idle sleep mode.

No error injection will be performed if the BUSY bit in the Status A (CRCSCAN.STATUSA) register is '1' or if CRCSCAN is in the Manual mode (determined by the CRC Source (SRC) bit field in the Control A (CRCSCAN.CTRLA) register).

The error-injected scan is performed by only scanning the last four words in the section selected by the SRC bit in the CRCSCAN.CTRLA register and comparing the CRC with the pre-calculated checksum described in the *Checksum* section. As the checksums will differ, the ERROR bit in the CRCSCAN.STATUSA register will be set to '1' at the end of the scan.

The status flags in the Interrupt Flags (CRCSCAN.INTFLAGS) register are updated (as after an ordinary scan), and an interrupt is generated if the DONE bit in the Interrupt Control (CRCSCAN.INTCTRL) register is set. Error injection when the Enable NMI Trigger (NMIEN) bit in the CRCSCAN.CTRLA register is '1' will trigger an NMI request.

Error injection does not start in sleep unless the Enable CRCSCAN (ENABLE) bit in the CRCSCAN.CTRLA register is '1'. Writing a '1' to the Reset CRCSCAN (RESET) bit in the CRCSCAN.CTRLA register will abort the error injection, just as it would for an ordinary scan.

28.3.2.5. Data Bus Accesses

Access attempts to unused Special Function Register (SFR) addresses within the CRCSCAN peripheral's address space will be discarded and return Bus Error to the CPU.

Attempted writes to read-only SFRs will be discarded and return Bus Error. A bus error received during the scan will cause the CRC Scan to fail, and a parity error on read data will cause the CRC Scan to fail.

28.3.3. Configuration Change Protection

This peripheral has registers under Configuration Change Protection (CCP), a security mechanism to avoid unintentional changes to the CRCSCAN settings. To write to these, first write a specific key to the CPU.CCP register, followed by a write access to the protected bits within four CPU instructions.

Attempting to write to an SFR protected register without following the appropriate CCP unlock sequence leaves the protected register unchanged and returns a Bus Error response on the data bus.

The following register is under CCP:

Table 28-5. CRCSCAN - Registers under Configuration Change Protection

Register	Key
CRCSCAN.CTRLB	IOREG

28.3.4. Interrupts

Table 28-6. Available Interrupt Vectors and Sources

Interrupt Vector Name	Interrupt Source Name	Description	Condition
NMI	ERROR	Non-Maskable Interrupt from a CRC Scan failure	ERROR in CRCSCAN.STATUSA and NMIEN in CRCSCAN.CTRLA is '1'
CRCSCAN	PERIOD	The scan period is done	PERIOD in CRCSCAN.INTFLAGS and PEREN in CRCSCAN.CTRLA is '1'
CRCSCAN	DONE	The scan of an entire Flash region is done	DONE in CRCSCAN.INTFLAGS is '1'

When the CRCSCAN interrupt condition occurs, the DONE or PERIOD flag in the Interrupt Flags (CRCSCAN.INTFLAGS) register is set to '1'. The flag is cleared by writing a '1' to the bit position.

Enable the Enable Periodic Timer (PEREN) bit in the Control A (CRCSCAN.CTRLA) register to get a PERIOD interrupt.

A Non-Maskable Interrupt (NMI) is enabled by writing a '1' to the Enable NMI Trigger (NMIEN) bit in the CTRLA register. An NMI is generated when the CRC Error (ERROR) bit in the Status A (CRCSCAN.STATUSA) register is '1', and the NMIEN bit in the CRCSCAN.CTRLA register is '1'. The NMI request remains active until a System Reset or by writing a '1' to the Reset CRCSCAN (RESET) bit in the CRCSCAN.CTRLA register.

28.3.5. Scan-Sleep Mode Operation

In Scan-Sleep mode, the CRCSCAN scans while the CPU is in Idle mode. For more information, see the *Operation* section.

28.3.6. Debug Operation

When the debugger sends a break command, the CRC will finish an ongoing scan before releasing the CPU. So, when stepping through a sequence that starts the CRC, it will finish the CRC during one step.

28.4. Functional Safety

The CRCSCAN peripheral is designed for use in Functional Safety systems. Primarily to check the Flash and boot ROM at boot time, but also to do periodic scans, in systems both with and without ECC, as CRC scan detects other errors than ECC. Error injection is present to test correct behavior.

28.5. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RESET	SRC[1:0]		PEREN	CRCSEL		NMIEN	ENABLE
0x01	CTRLB	7:0								EINJ
0x02	INTCTRL	7:0							PERIOD	DONE
0x03	INTFLAGS	7:0							PERIOD	DONE
0x04	STATUSA	7:0						OK	BUSY	ERROR
0x05	SCANADR	7:0	SCANADR[7:0]							
0x06	DATA	7:0	DATA[7:0]							
0x07	Reserved									
0x08	CRC	7:0	CRC[7:0]							
		15:8	CRC[15:8]							
		23:16	CRC[23:16]							
		31:24	CRC[31:24]							

28.6. Register Description

28.6.1. Control A

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

The bits in this register are not writable when the CRC is busy (the BUSY bit in the Interrupt Flags (CRCSCAN.INTFLAGS) register is '1'), except for the RESET bit, which is writable at any time.

Bit	7	6	5	4	3	2	1	0
	RESET	SRC[1:0]		PEREN	CRCSEL		NMIEN	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 – RESET Reset CRCSCAN
Writing this bit to '1' resets the CRCSCAN peripheral. The CRCSCAN Control and Status registers (CTRLA, INTFLAGS, INTCTRL, STATUSA) will be cleared one clock cycle after the RESET bit is written to '1'.
The RESET bit is a strobe bit.

Bits 6:5 – SRC[1:0] CRC Source
The SRC bit field selects which section of the Flash the CRCSCAN peripheral should check. To set up section sizes, refer to the fuse description.
The CRCSCAN can be enabled during internal Reset initialization to verify the Boot section before letting the CPU start (see the *Fuses* section).

Value	Name	Description
0x0	BOOT	The CRC is performed on the boot section of the Flash
0x1	CODE	The CRC is performed on the application code section of the Flash
0x2	DATA	The CRC is performed on the application data section of the Flash
0x3	MANUAL	Manual mode, CRC is performed on data written to the Data (DATA) register

Bit 4 – PEREN Enable Periodic Timer
This bit enables the periodic timer mechanism, which sets the Scan Period Done (PERIOD) flag in the Interrupt Flags (CRCSCAN.INTFLAGS) register every 32 scanned Flash words.

Value	Name	Description
0	DISABLE	The periodic timer is disabled
1	ENABLE	The periodic timer is enabled

Bit 3 – CRCSEL CRC Mode Select
This bit determines whether to use CRC32 or CRC16.

Value	Name	Description
0	CRC16	The CRC is performed using CRC16
1	CRC32	The CRC is performed using CRC32

Bit 1 – NMIEN Enable NMI Trigger
When this bit is written to '1', a CRC error during the scan will trigger an NMI.

Value	Name	Description
0	DISABLE	The CRC error during scan will not trigger a NMI
1	ENABLE	The CRC CRC error during scan will trigger a NMI

Bit 0 – ENABLE Enable CRCSCAN

Writing this bit to '1' enables the CRCSCAN peripheral with the current settings. The ENABLE bit is cleared by hardware after a scan when the DONE bit in the CRCSCAN.INTFLAGS register is set.
Writing this bit to '0' has no effect.
To see whether the CRCSCAN peripheral is busy with an ongoing scan, poll the BUSY bit in the Status A (CRCSCAN.STATUSA) register.

28.6.2. Control B

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

Bit	7	6	5	4	3	2	1	0
								EINJ
Access								W
Reset								0

Bit 0 – EINJ Inject CRC Error
Writing this bit to '1' starts a short scan resulting in a CRC error. It is used to test the error system in a Functional Safety application.
This bit always reads as '0'.

28.6.3. Interrupt Control

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

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

Bit 1 – PERIOD Scan Period Done Interrupt Enable
Writing this bit to '1' enables the interrupt. The Periodic Interrupt Timer (PEREN) bit in the Control A (CRCSCAN.CTRLA) register needs to be enabled for the timer to run, which eventually causes the corresponding interrupt flag to be set.

Bit 0 – DONE Scan Done Interrupt Enable
Writing this bit to '1' enables the interrupt.

28.6.4. Interrupt Flags

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

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

Bit 1 – PERIOD Scan Period Done

This bit is set when a scan period has expired, i.e., 32 Flash words have been scanned. CRC scan is halted when the PERIOD bit in the CRCSCAN.INTFLAGS register is set, regardless of the value of the PERIOD bit in the CRCSCAN.INTCTRL register. Clear the PERIOD flag and enter the IDLE sleep mode to resume the scan. The flag is cleared by writing a '1' to the bit position. Writing a '0' to this bit has no effect.

Bit 0 – DONE Scan Done

This bit is set when the scan has completed, i.e., the entire region selected by SRC bit field in the Control A (CRCSCAN.CTRLA) register has been scanned. The result of the scan can be read from the ERROR bit in the CRCSCAN.STATUSA register. The flag is cleared by writing a '1' to the bit position. Writing a '0' to this bit has no effect.
The Boot ROM may be configured to execute a scan of the flash on start up in which case the application will read the flag as '1'. The DONE flag must then be cleared before starting a scan.

28.6.5. Status A

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

Bit	7	6	5	4	3	2	1	0
						OK	BUSY	ERROR
Access						R	R	R
Reset						1	0	0

Bit 2 – OK CRC OK
When this bit is read as '1', the previous CRC has completed successfully.
This bit is only valid when the BUSY bit is read as '0'.

Bit 1 – BUSY CRC Busy
When this bit is read as '1', the CRCSCAN peripheral is busy. BUSY will remain '1' until completion of the entire scan, i.e., all addresses from the start to the end address have been scanned. As long as the module is busy, the access to the control registers is limited.

Bit 0 – ERROR CRC Error
When this bit is read as '1', the previous CRC completed with failure, `ERROR = (!BUSY & !OK)`.
This bit is connected as an NMI source and is set when a scan results in an error, issuing an NMI request if the NMIEN bit in the Control A (CRCSCAN.CTRLA) register is set to '1'.

28.6.6. Scan Address

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

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

Bits 7:0 – SCANADR[7:0] Address Being Scanned
The least significant byte (LSB) of the address being scanned. This allows the software to monitor if a scan is ongoing by confirming the address changes.
The contents of this register are undefined if the CRC Source (SRC) bit field configuration of the Control A (CRCSCAN.CTRLA) register is MANUAL.
The SCANADR register contains byte addresses, i.e., a scan of 32 words will cause SCANADR to increment by 64.

28.6.7. Input Data

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

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

Bits 7:0 – DATA[7:0] Input Data
Input data written by the user application in manual mode (when the CRC Source (SRC) bit field configuration in the CRCSCAN.CTRLA register is MANUAL). Data can be written every clock cycle.

28.6.8. CRC Result

Name: CRC
Offset: 0x08
Reset: 0xFFFFFFFF
Property: -

Bit	31	30	29	28	27	26	25	24
	CRC[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	CRC[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	CRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

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

Bits 31:0 – CRC[31:0] CRC Result

The CRC result register represents the 32-bit result value. The low byte ([7:0], suffix 0) is accessible at the original offset, and the n higher bytes can be accessed at offset +n. The temporary registers do not buffer the CRC result register. The Most Significant bit (MSb) of the CRC result is bit 31. If CRCSCAN is in 16-bit CRC mode (the CRC Mode Select (CRCSEL) bit in the CRCSCAN.CTRLA register is '0'), only bit field [15:0] is used, and bit field [31:16] will read as '0'.

This register is the actual linear-feedback shift register (LFSR) used in the CRC calculation. This register is reset by writing a '1' to the RESET bit in the CRCSCAN.CTRLA register. Such a reset must be performed between each data block in MANUAL mode so that the CRC calculation on each block starts with the correct initial value.

This register is only readable if the CRC Source (SRC) bit field configuration in the CRCSCAN.CTRLA register is in MANUAL mode to prevent information leakage. Any other configuration of the SRC bit field will cause all reads to return '0x0000_0000'.

If reading the reset value of the SFR in the CRC32 mode (CRCSEL=CRC32 in CRCSCAN.CTRLA), the value read will be '0x0000_0000' due to post-processing of the read result.

29. CCL - Configurable Custom Logic

29.1. Features

- Glue Logic for General Purpose PCB Design
- Four Programmable Look-Up Tables (LUTs)
- Combinatorial Logic Functions: Any Logic Expression Which Is a Function of up to Three Inputs.
- Sequencer Logic Functions:
  - Gated D flip-flop
  - JK flip-flop
  - Gated D latch
  - RS latch
- Flexible LUT Input Selection:
  - I/Os
  - Events
  - Subsequent LUT output
  - Internal peripherals such as:
    - Analog comparator
    - Timers/Counters
    - USART
    - SPI
- Clocked by a System Clock or Other Peripherals
- Output Can Be Connected to I/O Pins or an Event System
- Optional Synchronizer, Filter, or Edge Detector Available on Each LUT Output
- Optional Interrupt Generation from Each LUT Output:
  - Rising edge
  - Falling edge
  - Both edges

29.2. Overview

The Configurable Custom Logic (CCL) is a programmable logic peripheral which can be connected to the device pins, to events, or to 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 (CIPs) to handle the most time-critical parts of the application independent of the CPU.

The CCL peripheral provides a number of Look-up Tables (LUTs). Each LUT consists of three inputs, a truth table, a synchronizer/filter, and an edge detector. Each LUT can generate an output as a user programmable logic expression with three inputs. The output is generated from the inputs using the combinatorial logic and can be filtered to remove spikes. The CCL can be configured to generate an interrupt request on changes in the LUT outputs.

Neighboring LUTs can be combined to perform specific operations. A sequencer can be used for generating complex waveforms.

29.2.1. Block Diagram

Figure 29-1. CCL Block Diagram

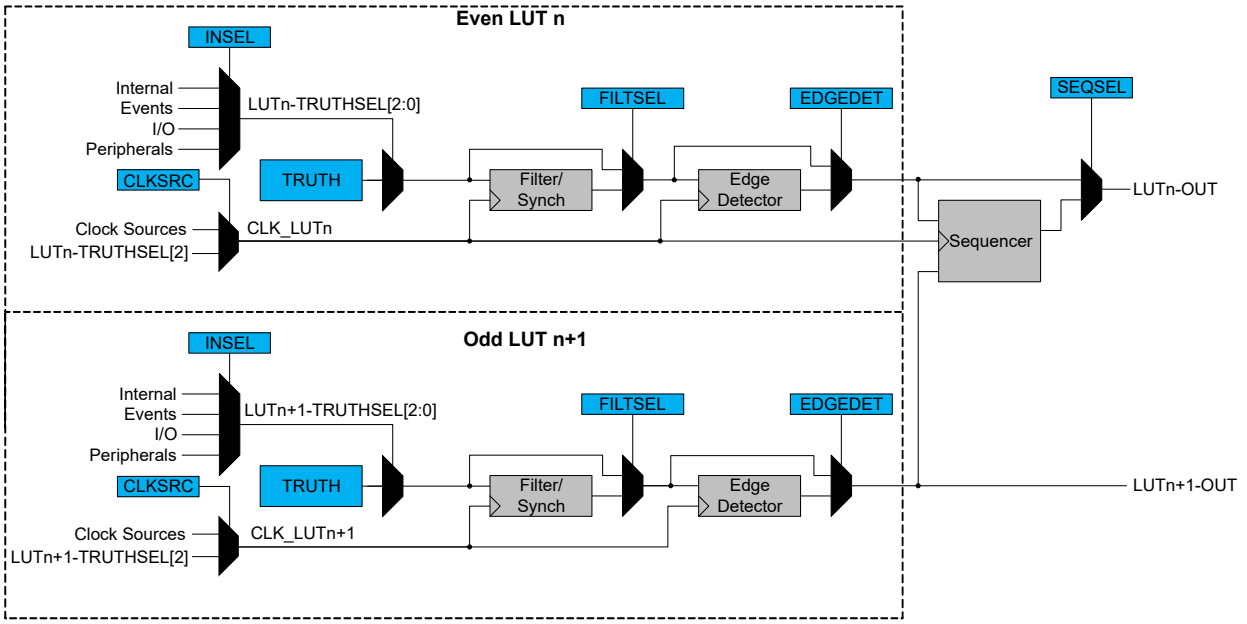


Table 29-2. Sequencer and LUT Connection

Sequencer	Even and Odd LUT
SEQ0	LUT0 and LUT1
SEQ1	LUT2 and LUT3

29.2.2. Signal Description

Name	Type	Description
LUTn-OUT	Digital output	Output from the Look-up Table
LUTn-IN[2:0]	Digital input	Input to the Look-up Table. LUTn-IN[2] can serve as CLK_LUTn.

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

29.2.2.1. CCL Input Selection MUX

The following peripherals outputs are available as inputs into the CCL LUT.

Value	Input source	INSEL0[3:0]	INSEL1[3:0]	INSEL2[3:0]
0x00	MASK	None		
0x01	FEEDBACK	LUTn		
0x02	LINK	LUT[n+1]		
0x03	EVENTA	EVENTA		
0x04	EVENTB	EVENTB		
0x05	IO	IN0	IN1	IN2
0x06	AC	AC0 OUT		
0x07	USART(1)	USART0 TXD		
0x08	SPI(2)	SPI0 MOSI	SPI0 MOSI	SPI0 SCK
0x09	TCE	WO0	WO1	WO2
0x0A	TCBA	TCB0 WO		

CCL Input Selection MUX (continued)				
Value	Input source	INSEL0[3:0]	INSEL1[3:0]	INSEL2[3:0]
0x0B	TCBB	TCB1 WO		

Notes:

1. USART connections to the CCL work only in Asynchronous/Synchronous USART Host mode.
2. SPI connections to the CCL work only in Host SPI mode.

29.3. Functional Description

29.3.1. Operation

29.3.1.1. Enable-Protected Configuration

The configuration of the LUTs and sequencers is enable-protected, meaning that they can only be configured when the corresponding even LUT is disabled (ENABLE = '0' in the LUT n Control A (CCL.LUTnCTRLA) register). This is a mechanism to suppress the undesired output from the CCL under (re-)configuration.

The following bits and registers are enable-protected:

- Sequencer Selection (SEQSEL) in the Sequencer Control n (CCL.SEQCTRLn) registers
- LUT n Control x (CCL.LUTnCTRLx) registers, except the ENABLE bit in the CCL.LUTnCTRLA register
- TRUTHn (CCL.TRUTHn) registers

The enable-protected bits in the CCL.LUTnCTRLx registers can be written at the same time as ENABLE in CCL.LUTnCTRLA is written to '1', but not at the same time as ENABLE is written to '0'.

The enable protection is denoted by the enable-protected property in the register description.

29.3.1.2. Enabling, Disabling, and Resetting

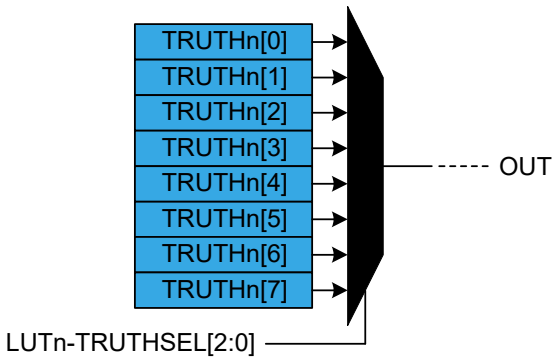
The CCL is enabled by writing a '1' to the ENABLE bit in the Control A (CCL.CTRLA) register. The CCL is disabled by writing a '0' to that ENABLE bit.

Each LUT is enabled by writing a '1' to the LUT Enable (ENABLE) bit in the CCL.LUTnCTRLA register. Each LUT is disabled by writing a '0' to the ENABLE bit in the CCL.LUTnCTRLA register.

29.3.1.3. Truth Table Logic

The truth table in each LUT unit can generate a combinational logic output as a function of up to three inputs (LUTn-TRUTHSEL[2:0]). It is possible to realize any 3-input Boolean logic function using one LUT.

Figure 29-2. Truth Table Output Value Selection of an LUT



Configure the truth table inputs (LUTn-TRUTHSEL[2:0]) by writing the Input Source Selection bit fields in the LUT Control registers:

- INSEL0 in CCL.LUTnCTRLB
- INSEL1 in CCL.LUTnCTRLB
- INSEL2 in CCL.LUTnCTRLC

Each combination of the input bits (LUTn-TRUTHSEL[2:0]) corresponds to one bit in the CCL.TRUTHn register, as shown in the table below:

Table 29-3. Truth Table of an LUT

LUTn-TRUTHSEL[2]	LUTn-TRUTHSEL[1]	LUTn-TRUTHSEL[0]	OUT
0	0	0	TRUTHn[0]
0	0	1	TRUTHn[1]
0	1	0	TRUTHn[2]
0	1	1	TRUTHn[3]
1	0	0	TRUTHn[4]
1	0	1	TRUTHn[5]
1	1	0	TRUTHn[6]
1	1	1	TRUTHn[7]

Important: Consider the unused inputs turned off (tied low) when logic functions are created.

Example 29-1. LUT Output for CCL.TRUTHn = `0x42`

If CCL.TRUTHn is configured to '`0x42`', the LUT output will be '`1`' when the inputs are '`0x1`' or '`0x6`', and will be '`0`' for any other combination of inputs.

29.3.1.4. Truth Table Inputs Selection

Input Overview

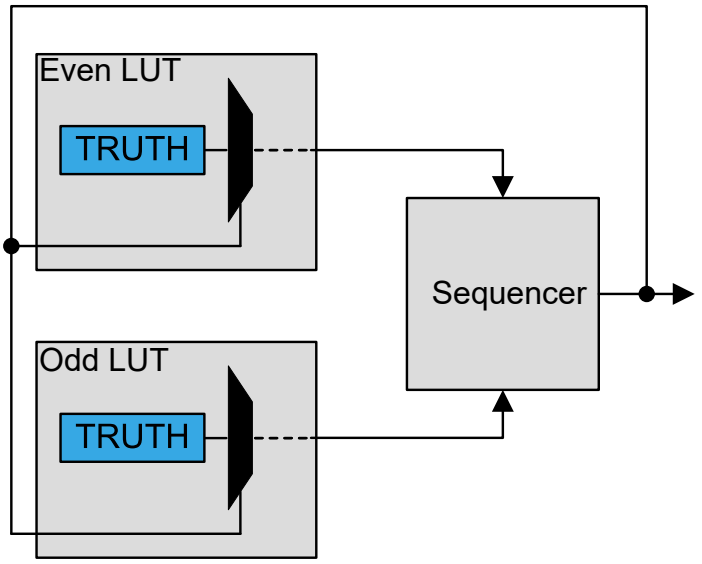
The inputs can be individually:

- OFF
- Driven by peripherals
- Driven by internal events from the Event System
- Driven by I/O pin inputs
- Driven by other LUTs

Internal Feedback Inputs (FEEDBACK)

The output from a sequencer can be used as an input source for the two LUTs it is connected to.

Figure 29-3. Feedback Input Selection



When selected (INSELy = FEEDBACK in LUTnCTRLx), the sequencer (SEQ) output is used as input for the corresponding LUTs.

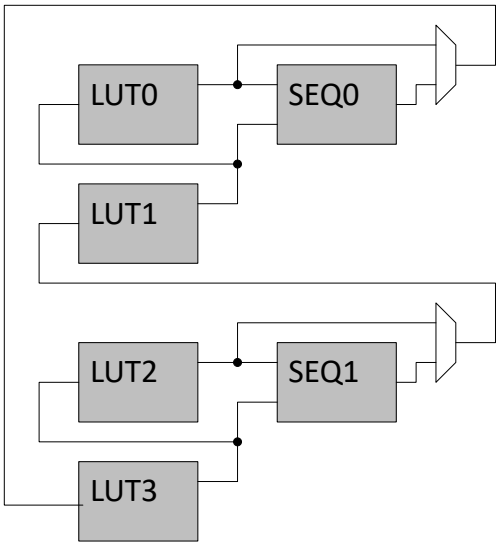
Linked LUT (LINK)

When selecting the LINK input option, the next LUT's direct output is used as LUT input. In general, LUT[n+1] is linked to the input of LUT[n]. LUT0 is linked to the input of the last LUT.

Example 29-2. Linking all LUTs on a Device with Four LUTs

- LUT1 is the input for LUT0
- LUT2 is the input for LUT1
- LUT3 is the input for LUT2
- LUT0 is the input for LUT3 (wrap-around)

Figure 29-4. Linked LUT Input Selection



Event Input Selection (EVENTx)

Events from the Event System can be used as inputs to the LUTs by writing to the INSELn bit groups in the LUT n Control B and C registers.

I/O Pin Inputs (IO)

When selecting the IO option, the LUT input will be connected to its corresponding I/O pin. Refer to the *I/O Multiplexing and Considerations* section in the data sheet for more details about where the LUTn-INy pins are located.

Peripherals

The different peripherals on the three input lines of each LUT are selected by writing to the Input Select (INSEL) bits in the LUT Control (LUTnCTRLB and LUTnCTRLC) registers.

29.3.1.5. 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.

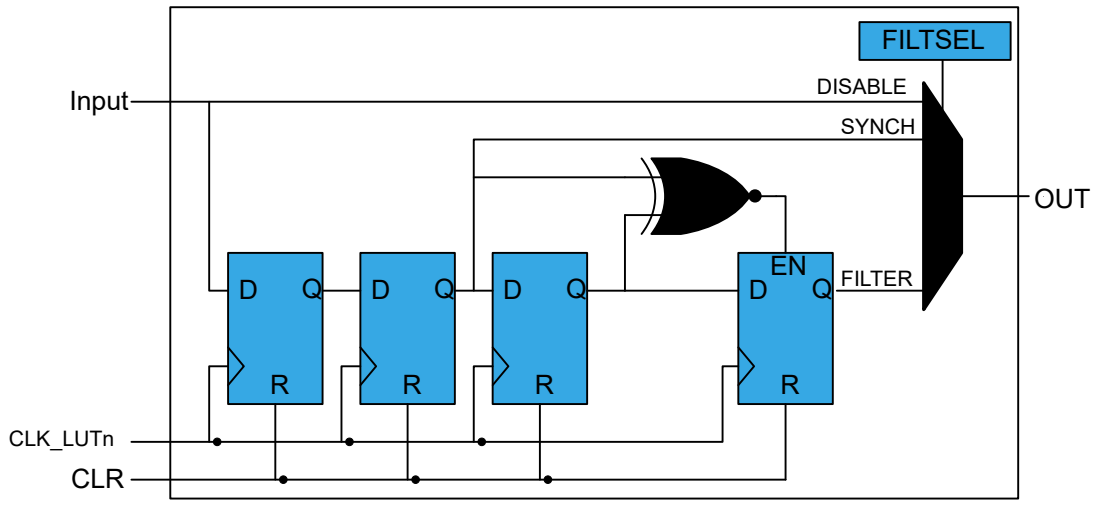
The Filter Selection (FILTSEL) bits in the LUT n Control A (CCL.LUTnCTRLA) registers define the digital filter options.

When FILTSEL = SYNCH, the output is synchronized with CLK_LUTn. The output will be delayed by two positive CLK_LUTn edges.

When FILTSEL = FILTER, only the input that is persistent for more than two positive CLK_LUTn edges will pass through the gated flip-flop to the output. The output will be delayed by four positive CLK_LUTn edges.

One clock cycle later, after the corresponding LUT is disabled, all internal filter logic is cleared.

Figure 29-5. Filter



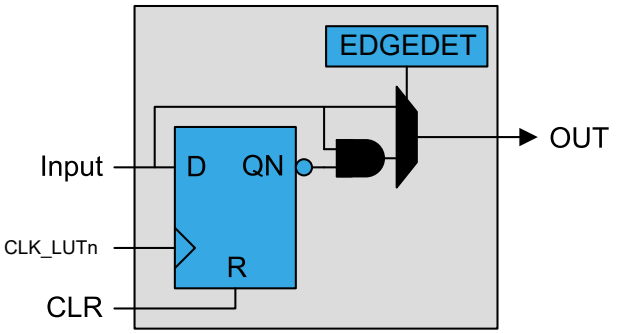
29.3.1.6. Edge Detector

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 '1' to the Edge Detection (EDGEDET) bit in the LUTn Control A (CCL.LUTnCTRLA) register. To avoid unpredictable behavior, a valid filter option must be enabled.

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 29-6. Edge Detector



29.3.1.7. Sequencer Logic

Each LUT pair can be connected to a sequencer. The sequencer can function as either gated D flip-flop, JK flip-flop, gated D latch, or RS latch. The function is selected by writing the Sequencer Selection (SEQSEL) bit group in the Sequencer Control (CCL.SEQCTRLn) register.

The sequencer receives its input from either the LUT, filter or edge detector, depending on the configuration.

A sequencer is clocked by the same clock as the corresponding even LUT. The clock source is selected by the Clock Source (CLKSRC) bit group in the LUT n Control A (CCL.LUTnCTRLA) register.

The flip-flop output (OUT) is refreshed on the rising edge of the clock. When the even LUT is disabled, the latch is cleared asynchronously. The flip-flop Reset signal (R) is kept enabled for one clock cycle.

Gated D Flip-Flop (GDFF)

The D input is driven by the even LUT output, and the G input is driven by the odd LUT output.

Figure 29-7. Gated D Flip-Flop

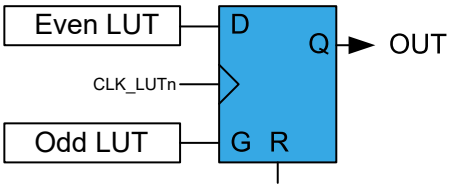


Table 29-4. GDFF Characteristics

R	G	D	OUT
1	X	X	0
0	1	1	1
0	1	0	0
0	0	X	Hold state (no change)

JK Flip-Flop (JK)

The J input is driven by the even LUT output, and the K input is driven by the odd LUT output.

Figure 29-8. JK Flip-Flop

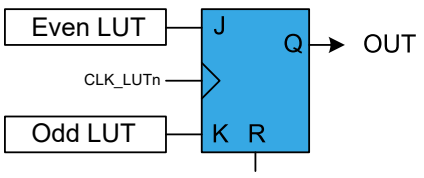


Table 29-5. JK Characteristics

R	J	K	OUT
1	X	X	0
0	0	0	Hold state (no change)
0	0	1	0
0	1	0	1
0	1	1	Toggle

Gated D Latch (DLATCH)

The D input is driven by the even LUT output, and the G input is driven by the odd LUT output.

Figure 29-9. D Latch

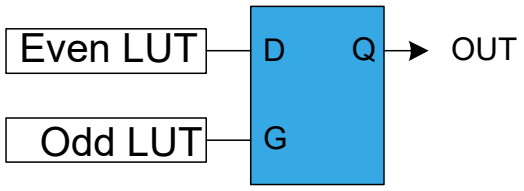


Table 29-6. D Latch Characteristics

G	D	OUT
0	X	Hold state (no change)
1	0	0
1	1	1

RS Latch (RS)

The S input is driven by the even LUT output, and the R input is driven by the odd LUT output.

Figure 29-10. RS Latch

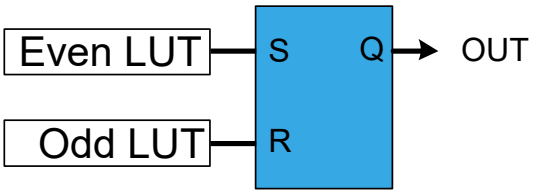


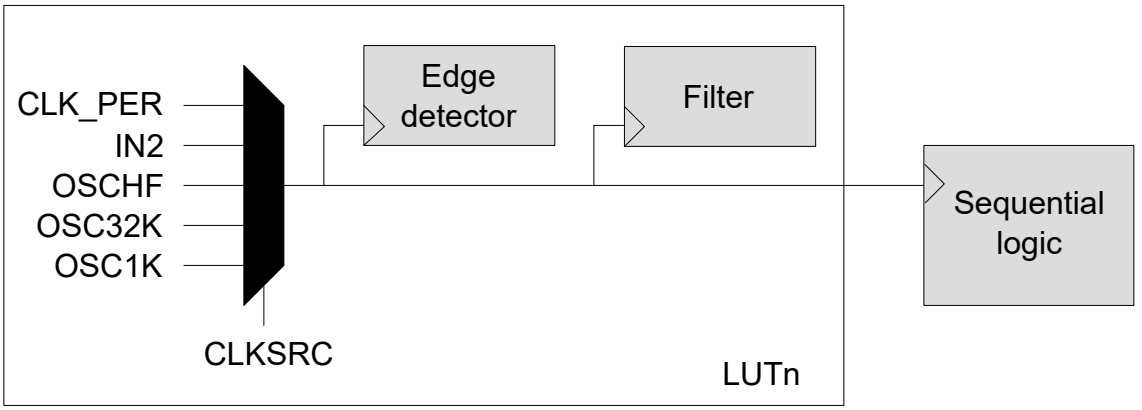
Table 29-7. RS Latch Characteristics

S	R	OUT
0	0	Hold state (no change)
0	1	0
1	0	1
1	1	Forbidden state

29.3.1.8. Clock Source Settings

The filter, edge detector, and sequencer are, by default, clocked by the peripheral clock (CLK_PER). It is also possible to use other clock inputs (CLK_LUTn) to clock these blocks. This is configured by writing the Clock Source (CLKSRC) bits in the LUT Control A register.

Figure 29-11. Clock Source Settings



When the Clock Source (CLKSRC) bit is written to `0x1`, LUTn-TRUTHSEL[2] is used to clock the corresponding filter and edge detector (CLK_LUTn). The sequencer is clocked by the CLK_LUTn of the even LUT in the pair. When CLKSRC is written to `0x1`, LUTn-TRUTHSEL[2] is treated as OFF (low) in the TRUTH table.

The CCL peripheral must be disabled while changing the clock source to avoid undefined outputs from the peripheral.

29.3.2. Interrupts

Table 29-8. Available Interrupt Vectors and Sources

Name	Vector Description	Conditions
CCL	CCL interrupt	INTn in INTFLAG is raised as configured by the INTMODEn bits in the CCL.INTCTRLn 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.

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.

29.3.3. Events

The CCL can generate the events shown in the table below.

Table 29-9. Event Generators in the CCL

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
CCL	LUTn	LUT output level	Level	Asynchronous	Depends on the CCL configuration

The CCL has the event users below for detecting and acting upon input events.

Table 29-10. Event Users in the CCL

User Name		Description	Input Detection	Async/Sync
Peripheral	Input			
CCL	LUTnx	LUTn input x or clock signal	No detection	Async

The event signals are passed directly to the LUTs without synchronization or input detection logic.

Two event users are available for each LUT. They can be selected as LUTn inputs by writing to the INSELn bit groups in the LUT n Control B and Control C (CCL.LUTnCTRLB or LUTnCTRLC) registers.

Refer to the *EVSYS - Event System* section for more details regarding the event types and the EVSYS configuration.

29.3.4. Sleep Mode Operation

Writing the Run In Standby (RUNSTDBY) bit in the Control A (CCL.CTRLA) register to '1' will allow the selected clock source to be enabled in Standby sleep mode.

If RUNSTDBY is '0', the peripheral clock will be disabled in Standby sleep mode. If the filter, edge detector, and/or sequencer are enabled, the LUT output will be forced to '0' in Standby sleep mode. In Idle sleep mode, the TRUTH table decoder will continue the 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 LUTn-TRUTHSEL[2] will always clock the filter, edge detector, and sequencer. The availability of the LUTn-TRUTHSEL[2] clock in sleep modes will depend on the sleep settings of the peripheral used.

29.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0		RUNSTDBY						ENABLE
0x01	SEQCTRL0	7:0					SEQSEL0[3:0]			
0x02	SEQCTRL1	7:0					SEQSEL1[3:0]			
0x03 ... 0x04	Reserved									
0x05	INTCTRL0	7:0	INTMODE3[1:0]		INTMODE2[1:0]		INTMODE1[1:0]		INTMODE0[1:0]	
0x06	Reserved									
0x07	INTFLAGS	7:0					INT3	INT2	INT1	INT0
0x08	LUT0CTRLA	7:0	EDGEDET	OUTEN	FILTSEL[1:0]		CLKSRC[2:0]			ENABLE
0x09	LUT0CTRLB	7:0	INSEL1[3:0]				INSEL0[3:0]			
0x0A	LUT0CTRLC	7:0					INSEL2[3:0]			
0x0B	TRUTH0	7:0	TRUTH[7:0]							
0x0C	LUT1CTRLA	7:0	EDGEDET	OUTEN	FILTSEL[1:0]		CLKSRC[2:0]			ENABLE
0x0D	LUT1CTRLB	7:0	INSEL1[3:0]				INSEL0[3:0]			
0x0E	LUT1CTRLC	7:0					INSEL2[3:0]			
0x0F	TRUTH1	7:0	TRUTH[7:0]							
0x10	LUT2CTRLA	7:0	EDGEDET	OUTEN	FILTSEL[1:0]		CLKSRC[2:0]			ENABLE
0x11	LUT2CTRLB	7:0	INSEL1[3:0]				INSEL0[3:0]			
0x12	LUT2CTRLC	7:0					INSEL2[3:0]			
0x13	TRUTH2	7:0	TRUTH[7:0]							
0x14	LUT3CTRLA	7:0	EDGEDET	OUTEN	FILTSEL[1:0]		CLKSRC[2:0]			ENABLE
0x15	LUT3CTRLB	7:0	INSEL1[3:0]				INSEL0[3:0]			
0x16	LUT3CTRLC	7:0					INSEL2[3:0]			
0x17	TRUTH3	7:0	TRUTH[7:0]							

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						ENABLE
Access		R/W						R/W
Reset		0						0

Bit 6 – RUNSTDBY Run in Standby
Writing this bit to '1' will enable the peripheral to run in Standby sleep mode.

Value	Description
0	The CCL will not run in Standby sleep mode
1	The CCL will run in Standby sleep mode

Bit 0 – ENABLE Enable

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

29.5.2. Sequencer Control 0

Name: SEQCTRL0
Offset: 0x01
Reset: 0x00
Property: Enable-Protected

Bit	7	6	5	4	3	2	1	0
					SEQSEL0[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 3:0 – SEQSEL0[3:0] Sequencer Selection
This bit group selects the sequencer configuration for LUT0 and LUT1.

Value	Name	Description
0x0	DISABLE	The sequencer is disabled
0x1	DFF	D flip-flop
0x2	JK	JK flip-flop
0x3	LATCH	D latch
0x4	RS	RS latch
Other	-	Reserved

29.5.3. Sequencer Control 1

Name: SEQCTRL1
Offset: 0x02
Reset: 0x00
Property: Enable-Protected

Bit	7	6	5	4	3	2	1	0
					SEQSEL1[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 3:0 – SEQSEL1[3:0] Sequencer Selection
This bit group selects the sequencer configuration for LUT2 and LUT3.

Value	Name	Description
0x0	DISABLE	The sequencer is disabled
0x1	DFF	D flip-flop
0x2	JK	JK flip-flop
0x3	LATCH	D latch
0x4	RS	RS latch
Other	-	Reserved

29.5.4. Interrupt Control 0

Name: INTCTRL0
Offset: 0x05
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	INTMODE3[1:0]		INTMODE2[1:0]		INTMODE1[1:0]		INTMODE0[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

Bits 0:1, 2:3, 4:5, 6:7 – INTMODEn

The bits in INTMODEn select the interrupt sense configuration for LUTn-OUT.

Value	Name	Description
0x0	INTDISABLE	Interrupt disabled
0x1	RISING	Sense rising edge
0x2	FALLING	Sense falling edge
0x3	BOTH	Sense both edges

29.5.5. Interrupt Flag

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

Bit	7	6	5	4	3	2	1	0
					INT3	INT2	INT1	INT0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 0, 1, 2, 3 – INT Interrupt Flag
The INTn flag is set when the LUTn output change matches the Interrupt Sense mode as defined in CCL.INTCTRLn. Writing a '1' to this flag's bit location will clear the flag.

29.5.6. LUT n Control A

Name: LUTnCTRLA
Offset: 0x08 + n*0x04 [n=0..3]
Reset: 0x00
Property: Enable-Protected

Bit	7	6	5	4	3	2	1	0
	EDGEDET	OUTEN	FILTSEL[1:0]		CLKSRC[2:0]			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 – EDGEDET Edge Detection

Value	Description
0	Edge detector is disabled
1	Edge detector is enabled

Bit 6 – OUTEN Output Enable
This bit enables the LUT output to the LUTn OUT 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

Bits 5:4 – FILTSEL[1:0] Filter Selection
These bits select the LUT output filter options.

Value	Name	Description
0x0	DISABLE	Filter disabled
0x1	SYNCH	Synchronizer enabled
0x2	FILTER	Filter enabled
0x3	-	Reserved

Bits 3:1 – CLKSRC[2:0] Clock Source Selection
This bit selects between various clock sources to be used as the clock (CLK_LUTn) for an LUT. The CLK_LUTn of the even LUT is used for clocking the sequencer of an LUT pair.

Value	Name	Description
0x0	CLKPER	CLK_PER is clocking the LUT
0x1	IN2	IN2 is clocking the LUT
0x2	-	Reserved
0x3	-	Reserved
0x4	OSCHF	Internal high-frequency oscillator before prescaler is clocking the LUT
0x5	OSC32K	Internal 32.786 kHz oscillator is clocking the LUT
0x6	OSC1K	Internal 32.786 kHz oscillator divided by 32 is clocking the LUT
0x7	-	Reserved

Bit 0 – ENABLE LUT Enable

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

29.5.7. LUT n Control B

Name: LUTnCTRLB
Offset: 0x09 + n*0x04 [n=0..3]
Reset: 0x00
Property: Enable-Protected

Notes:

1. SPI connections to the CCL work in Host SPI mode only.
2. USART connections to the CCL work only when the USART is in one of the following modes:
   - Asynchronous USART
   - Synchronous USART host

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
These bits select the source for input 1 of LUT n.

Value	Name	Description
0x00	MASK	Masked input
0x01	FEEDBACK	Feedback input
0x02	LINK	Output from LUT[n+1] as input source
0x03	EVENTA	Event A as input source
0x04	EVENTB	Event B as input source
0x05	IN1	IN1 as input source
0x06	AC0	AC0 OUT input source
0x07	USART0	USART0 TXD input source
0x08	SPI0	SPI0 MOSI input source
0x09	TCE0	TCE0 WO1 input source
0x0A	TCB0	TCB0 WO input source
0x0B	TCB1	TCB1 WO input source
Other	-	Reserved

Bits 3:0 – INSEL0[3:0] LUT n Input 0 Source Selection
These bits select the source for input 0 of LUT n.

Value	Name	Description
0x00	MASK	Masked input
0x01	FEEDBACK	Feedback input
0x02	LINK	Output from LUT[n+1] as input source
0x03	EVENTA	Event A as input source
0x04	EVENTB	Event B as input source
0x05	IN0	IN0 as input source
0x06	AC0	AC0 OUT input source
0x07	USART0	USART0 TXD input source
0x08	SPI0	SPI0 MOSI input source
0x09	TCE0	TCE0 WO0 input source
0x0A	TCB0	TCB0 WO input source
0x0B	TCB1	TCB1 WO input source
Other	-	Reserved

29.5.8. LUT n Control C

Name: LUTnCTRLC
Offset: 0x0A + n*0x04 [n=0..3]
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
These bits select the source for input 2 of LUT n.

Value	Name	Description
0x00	MASK	Masked input
0x01	FEEDBACK	Feedback input
0x02	LINK	Output from LUT[n+1] as input source
0x03	EVENTA	Event A as input source
0x04	EVENTB	Event B as input source
0x05	IN2	IN2 as input source
0x06	AC0	AC0 OUT input source
0x07	USART0	USART0 TXD input source
0x08	SPI0	SPI0 MOSI input source
0x09	TCE0	TCE0 WO2 input source
0x0A	TCB0	TCB0 WO input source
0x0B	TCB1	TCB1 WO input source
Other	-	Reserved

29.5.9. TRUTHn

Name: TRUTHn
Offset: 0x0B + n*0x04 [n=0..3]
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
These bits determine the output of LUTn according to the LUTn-TRUTHSEL[2:0] inputs.

Bit Name	Value	Description
TRUTH[0]	0	The output of LUTn is '0' when the inputs are '0x0'
	1	The output of LUTn is '1' when the inputs are '0x0'
TRUTH[1]	0	The output of LUTn is '0' when the inputs are '0x1'
	1	The output of LUTn is '1' when the inputs are '0x1'
TRUTH[2]	0	The output of LUTn is '0' when the inputs are '0x2'
	1	The output of LUTn is '1' when the inputs are '0x2'
TRUTH[3]	0	The output of LUTn is '0' when the inputs are '0x3'
	1	The output of LUTn is '1' when the inputs are '0x3'
TRUTH[4]	0	The output of LUTn is '0' when the inputs are '0x4'
	1	The output of LUTn is '1' when the inputs are '0x4'
TRUTH[5]	0	The output of LUTn is '0' when the inputs are '0x5'
	1	The output of LUTn is '1' when the inputs are '0x5'
TRUTH[6]	0	The output of LUTn is '0' when the inputs are '0x6'
	1	The output of LUTn is '1' when the inputs are '0x6'
TRUTH[7]	0	The output of LUTn is '0' when the inputs are '0x7'
	1	The output of LUTn is '1' when the inputs are '0x7'

30. AC - Analog Comparator

30.1. Features

- Selectable Hysteresis
- Analog Comparator Output Available on Pin
- Comparator Output Inversion Available
- Flexible Input Selection:
  - 2 positive pins
  - 3 negative pins
  - Internal reference voltage generator
- Sampled Mode for Low-Power Operation
- Interrupt Generation on:
  - Rising edge
  - Falling edge
  - Both edges
- Event Generation:
  - Comparator output
- Event User:
  - Sample

30.2. Overview

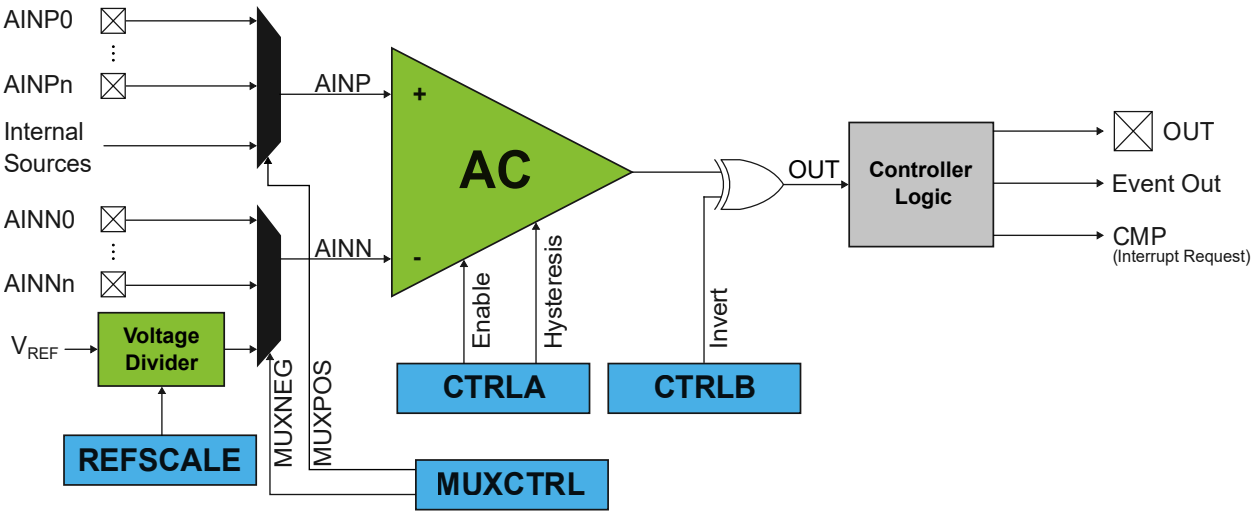
The Analog Comparator (AC) compares the voltage levels on two inputs and gives a digital output based on this comparison. The input selection includes analog port pins and internally generated inputs. The AC output can be configured to generate interrupt requests and/or events based on changes in the AC inputs, or it can be made available on the pin for external devices.

A hysteresis feature can adjust the dynamic behavior of the AC. The hysteresis can be customized to optimize the operation for each application.

The AC features a Sampled mode designed for low-power applications, wherein it periodically samples the inputs at a predefined frequency. Additionally, the AC includes an Event Trigger mode that enables input sampling at regular or irregular intervals, depending on the trigger source.

30.2.1. Block Diagram

Figure 30-1. AC Block Diagram



Note: Depending on the pin count for a device, the AC output may not be directly accessible via a pin. Refer to the *I/O Multiplexing and Considerations* section for more details. In such cases, an event system channel can route the AC output to a pin.

30.2.2. Signal Description

Signal	Description	Type
AINP	Positive input	Analog
AINN	Negative input	Analog
OUT	Comparator output	Digital

30.3. Functional Description

30.3.1. Initialization

Follow these steps for basic operation:

1. Configure the desired input pins as analog inputs. Refer to the *PORT - I/O Pin Configuration* section for more details.
2. Select the positive and negative input sources by writing to the Positive and Negative Input MUX Selection (MUXPOS and MUXNEG) bit fields in the MUX Control (ACn.MUXCTRL) register.
3. Optional: Select the desired mode by writing to the Operation Select (OPSEL) bit fields in the Control A (ACn.CTRLA) register. If Event Triggered mode is selected, configure the desired event source for ACn. Refer to the *EVSYS - Event System* section for more details.
4. Optional: Enable the output to pin by writing a '1' to the Output Enable (OUTEN) bit in the Control B (ACn.CTRLB) register.
5. Enable the AC by writing a '1' to the ENABLE bit in ACn.CTRLA.

Note: The OUT pin must be configured as an output to avoid the pin being tri-stated when the AC is disabled.

During the start-up time after enabling the AC, the INITVAL bit in the Control B (ACn.CTRLB) register determines the initial state of the AC output before the AC is ready. For details about the start-up time of the AC, refer to the *Electrical Characteristics* section.

30.3.2. Operation

The digital output of the comparator is '1' when the positive input voltage is greater than the negative input voltage and '0' otherwise. The output is available through a logic XOR gate, allowing inversion of the output when the Invert Comparator Output (INVERT) bit in the Control B (ACn.CTRLB) register is '1'.

30.3.2.1. Input and Reference Selection

The input selection to the AC is controlled by the Positive and Negative Input MUX Selection (MUXPOS and MUXNEG) bit fields in the Multiplexer Control (ACn.MUXCTRL) register. For positive AC input, an analog pin or internal connections can be selected. For negative input, the selection can be made between analog pins and the internal reference voltage.

When the negative input is an internal reference voltage, the generated voltage depends on the Scaled Reference Voltage (ACn.REFSCALE) register value and the reference voltage selected in the VREF peripheral and is calculated as:

$$V_{REFSCALE} = \frac{REFSCALE}{2^{SIZE\_REFSCALE}} \times V_{REF}$$

where SIZE_REFSCALE = 8.

Refer to the *VREF - Voltage Reference section* for details on available reference voltages.

The AC needs time to stabilize after changing inputs to the I/O pins or setting a new voltage reference. Refer to the *Electrical Characteristics* section for more details.

30.3.2.2. Input Hysteresis

Applying an input hysteresis helps prevent constant toggling of the output when the noise-afflicted input signals are close to each other.

The hysteresis is configured by writing to the Hysteresis Mode Select (HYSMODE) bit field in the Control A (ACn.CTRLA) register. For details about typical values of hysteresis levels, refer to the *Electrical Characteristics* section.

The hysteresis has no effect when operating in Sampled or Event Triggered mode.

30.3.2.3. Operation Modes

The AC can be operated in the desired mode based on application needs by writing to Operation Select (OPSEL) bit fields in the Control (ACn.CTRLA) register.

30.3.2.3.1. Continuous Mode

In this mode, the AC remains continuously on when enabled. While this mode offers the fastest response time, it also results in the highest power consumption.

30.3.2.3.2. Sampled Mode

In this mode, the AC is periodically enabled and disabled based on the selected sampling frequency, making it ideal for low-power operation. When activated, it evaluates the input pin states and then turns off. The output state of the comparator remains unchanged until the next comparison is performed. The AC utilizes the internal 32.768 kHz oscillator to support its sampling functionality.

30.3.2.3.3. Event Triggered Mode

In this mode, the comparator is activated by an incoming event. It turns on when the event occurs, performs the detection, and then automatically turns off. This mode allows sampling at predictable intervals or under specific conditions defined by the event generator. Note that there is a delay of 1-2 clock cycles on the internal 32.768 kHz oscillator between the occurrence of the event and the actual sampling.

30.3.3. Events

The AC can generate the following events:

Table 30-1. Event Generators in AC

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Module	Event				
ACn	OUT	Comparator output level	Level	Asynchronous	Given by AC output level

The AC has one event user for detecting and acting upon input events. The table below describes the event user and the associated functionality.

Table 30-2. AC Event Users and Available Event Actions

User Name		Description	Input Detection	Async/Sync
Peripheral	Input			
ACn	SAMPLE	Sample	Edge	Async

The AC can be configured to sample the inputs on the rising edge of an event signal by selecting the Event Triggered operating mode, which is done by configuring the Operation Select (OPSEL) bitfield in the Control A (ACn.CTRLA) register.

Refer to the *EVSYS - Event System* section for more information on event types and how to configure the Event System.

30.3.4. Interrupts

Table 30-3. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
ACn_AC	CMP	The output level matches the configuration	The Interrupt Mode (INTMODE) bit field in the Interrupt Control (ACn.INTCTRL) 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.

30.3.5. Sleep Mode Operation

The AC continues to operate normally in the Idle sleep mode.

In Standby sleep mode, the AC is disabled by default. However, if the Run in Standby (RUNSTDBY) bit in the Control A (ACn.CTRLA) register is written to '1', the AC will continue to operate as normal with an event, interrupt and AC output on the pin even if the CLK_PER is not running in Standby sleep mode.

In Power-Down sleep mode, the AC and the output to the pad are disabled.

30.4. Register Summary - AC

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY	OPSEL[1:0]				HYSMODE[1:0]		ENABLE
0x01	CTRLB	7:0			OUTEN	INVERT	INITVAL			
0x02	MUXCTRL	7:0			MUXPOS[2:0]			MUXNEG[2:0]		
0x03	REFSCALE	7:0	REFSCALE[7:0]							
0x04	INTCTRL	7:0			INTMODE[1:0]					CMP
0x05	INTFLAGS	7:0								CMP
0x06	STATUS	7:0								CMP

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
	RUNSTDBY	OPSEL[1:0]				HYSMODE[1:0]		ENABLE
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 Mode
This bit controls whether the AC continues operation in Standby sleep mode.
Since the clock is stopped, Interrupts and Status flags are not updated when the AC runs in Standby sleep mode.

Value	Description
0	The peripheral is halted in Standby sleep mode
1	The peripheral continues to operate in Standby sleep mode

Bits 6:5 – OPSEL[1:0] Operation Select
This bit field sets the operation mode and selects the sampling frequency for the sampled mode. When selecting event triggered or sampled mode the first sample is taken immediately when enabled.

Value	Name	Description
0x0	CONT	Continuous mode
0x1	EVENT	Event Triggered mode
0x2	SAMP1K	Sampled mode, 1 kHz sampling frequency
0x3	SAMP128	Sampled mode, 128 Hz sampling frequency

Bits 2:1 – HYSMODE[1:0] Hysteresis Mode Select
This bit field controls the Hysteresis mode for the AC input.
For more details on typical values of hysteresis levels, refer to the *Electrical Characteristics* section.

Value	Name	Description
0x0	NONE	No hysteresis
0x1	SMALL	Small hysteresis
0x2	MEDIUM	Medium hysteresis
0x3	LARGE	Large hysteresis

Bit 0 – ENABLE Enable AC
This bit controls whether the AC is enabled.

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

30.5.2. Control B

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

Bit	7	6	5	4	3	2	1	0
			OUTEN	INVERT	INITVAL			
Access			R/W	R/W	R/W			
Reset			0	0	0			

Bit 5 – OUTEN Output Enable
This bit controls whether the AC output is enabled on the pin.

Value	Description
0	Output to OUT pin disabled
1	Output to OUT pin enabled

Bit 4 – INVERT Invert Output
This bit controls whether the AC output is inverted.
When enabled, the inversion has to be considered when using the AC output signal as an input signal to other peripherals or parts of the system.

Value	Description
0	The OUT signal is not inverted
1	The OUT signal is inverted

Bit 3 – INITVAL Output Initial Value
This bit controls the initial value of the comparator output.
This initial value prevents the AC output from toggling before the comparator is ready.

Value	Name	Description
0	LOW	The OUT signal is initialized to '0'
1	HIGH	The OUT signal is initialized to '1'

30.5.3. MUX Control

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

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

Bits 5:3 – MUXPOS[2:0] Positive Input MUX Selection
This bit field controls the input signal to the positive input of the AC.

Value	Name	Description
0x0	ACINP0	Positive pin 0
0x1 – 0x2	-	Reserved
0x3	ACINP3	Positive pin 3
0x4 – 0x6	-	Reserved
0x7	VDDDIV10	VDD/10

Bits 2:0 – MUXNEG[2:0] Negative Input MUX Selection
This bit field controls the input signal to the negative input of the AC.

Value	Name	Description
0x0	ACINN0	Negative pin 0
0x1	ACINN1	Negative pin 1
0x2	ACINN2	Negative pin 2
0x3	-	Reserved
0x4	REFSCALE	Reference scaler
Other	-	Reserved

30.5.4. Reference Scaling

Name: REFSCALE
Offset: 0x03
Reset: 0xFF
Property: -

Bit	7	6	5	4	3	2	1	0
	REFSCALE[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 – REFSCALE[7:0] Scaled Reference Voltage
This bit field defines the scaling factor for the reference voltage scaler.

30.5.5. Interrupt Control

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

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

Bits 5:4 – INTMODE[1:0] Interrupt Mode
This bit field controls which edge(s) of the AC output triggers an interrupt request.

Interrupt Generation in Normal Mode

Value	Name	Description
0x0	BOTHEDGE	Positive and negative inputs crosses
0x1	-	Reserved
0x2	NEGEDGE	Positive input goes below negative input
0x3	POSEDGE	Positive input goes above negative input

Bit 0 – CMP AC Interrupt Enable
This bit controls whether the AC interrupt is enabled.
When enabled, the interrupt will be triggered when the CMP bit in the ACn.INTFLAGS register is set.

Value	Description
0	The AC interrupt is disabled
1	The AC interrupt is enabled

30.5.6. Interrupt Flags

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

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

Bit 0 – CMP AC Interrupt Flag
This flag is cleared by writing a '1' to it.
This flag is set when the OUT signal matches the Interrupt Mode (INTMODE) bit field, as defined in the Interrupt Control (ACn.INTCTRL) register.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the AC Interrupt Flag.

30.5.7. Status

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

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

Bit 0 – CMP Analog Comparator State
This bit shows the current status of the OUT signal from the Comparator. The status bit is up to three CLK_PER cycles delayed compared to the output value.

Value	Name	Description
0x0	LOW	The OUT signal is low
0x1	HIGH	The OUT signal is high

31. ADC - Analog-to-Digital Converter

31.1. Features

- 10-Bit Single-Ended Analog-to-Digital Converter (ADC)
  - Up to 15 bits with oversampling
- Conversion Rate Up to 166 ksps, at 10-bit Resolution
- Up to 20 I/O Pin Inputs, in Addition to Internal Inputs from Sensors and References
- Multiple Internal ADC Reference Voltages
  - V_{DD}
  - 0.512V
  - 1.024V
  - 2.048V
  - 2.500V
  - 4.096V
- External Reference Input
- Single and Free-Running Conversions
- Event Triggered Conversion
- Series and Burst Accumulation Modes
- Accumulation of Up to 1024 Conversions
- Left/Right Adjusted or Averaged Result
- Interrupts on Conversion Complete
- Configurable Window Comparator

31.2. Overview

The Analog-to-Digital Converter (ADC) peripheral is a single-ended ADC with a 10-bit resolution. It is connected to an analog input multiplexer, which allows selection between multiple inputs. The ADC measures the voltage between the selected input and 0V (GND). The ADC inputs can be internal (for example, the internal temperature sensor) or external analog input pins.

An ADC conversion can be started by software or by using the Event System (EVSYS) to route an event from other peripherals, which makes it possible to sample input signals periodically, trigger an ADC conversion on a particular condition, and trigger ADC conversions in Standby sleep mode. A window compare feature is available to monitor the input signal. It can be configured to trigger an interrupt if the sample is either below or above a user-defined threshold, or inside or outside a user-defined window.

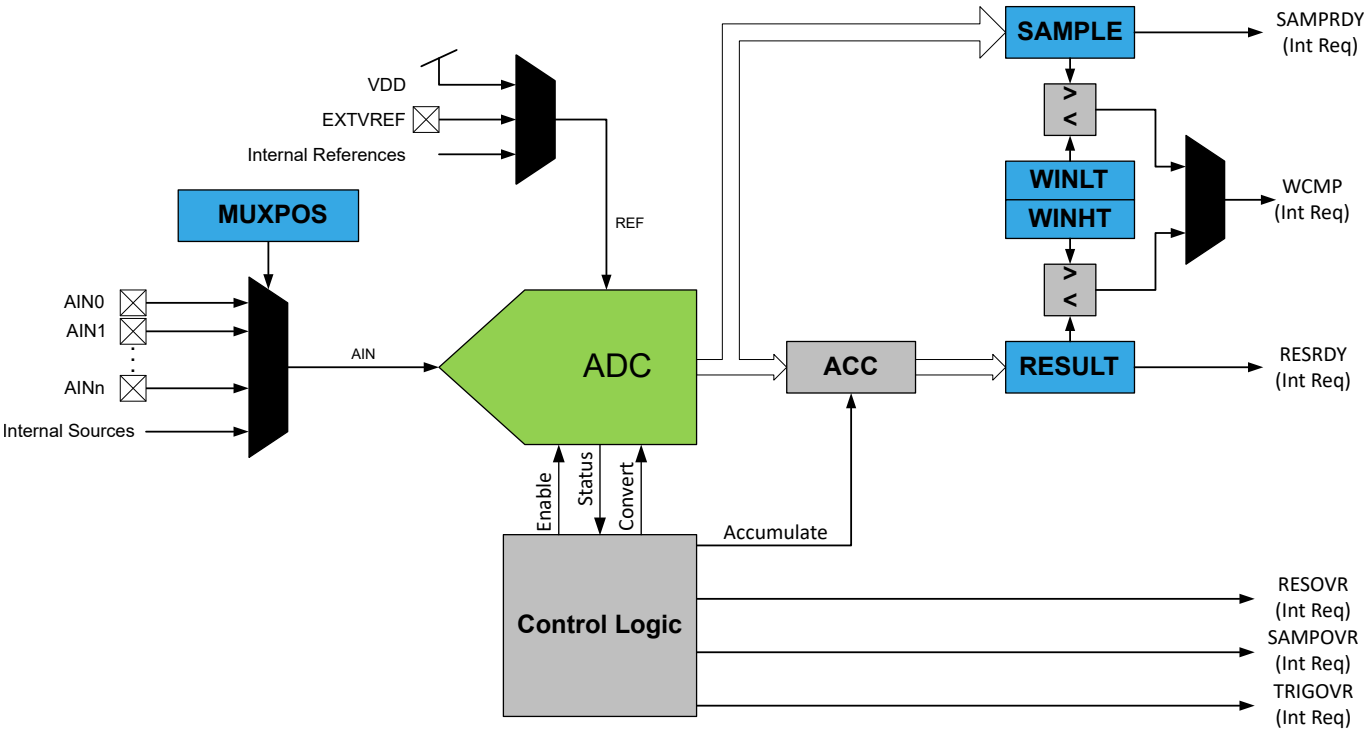
The ADC input signal is fed through a sample-and-hold circuit, which ensures that the input voltage to the ADC is kept at a constant level during conversion.

The ADC supports sampling in bursts, where a configurable number of samples are accumulated into a single ADC result (Sample Accumulation).

The ADC reference voltage can be either internal or supplied through the external analog reference pin (EXTVREF).

31.2.1. Block Diagram

Figure 31-1. Block Diagram



31.2.2. Signal Description

Signal	Type	Description
AIN	Analog input	Analog input
REF	Analog input	Voltage reference

31.3. Functional Description

31.3.1. Definitions

- Conversion: The operation in which analog values on the selected ADC inputs are transformed into a digital representation
- Sample: The value placed in the Sample (ADCn.SAMPLE) register, that is, the outcome of a conversion operation
- Result: The value placed in the Result (ADCn.RESULT) register. Depending on the ADC configuration, this value is a single sample or the sum of multiple accumulated samples.

31.3.2. Initialization

The following steps are recommended to initialize and run the ADC in basic operation:

1. Configure the timebase by writing to the TIMEBASE bit field in the Main Clock Timebase (MCLKTIMEBASE) register in the Clock Controller (CLKCTRL).
2. Configure the Prescaler (PRESC) bit field in the Control B (ADCn.CTRLB) register.
3. Configure the Reference Select (REFSEL) bit field in the Control C (ADCn.CTRLC) register.
4. Configure the Sample Duration (SAMPDUR) bit field in the Control E (ADCn.CTRLE) register.
5. Optional: Configure the number of samples to be accumulated by writing to the Sample Accumulation Number Select (SAMPNUM) bit field in the Control F (ADCn.CTRLF) register.

6. Optional: Enable the Free-Running mode by writing a '1' to the Free-Running (FREERUN) bit in the Control F (ADCn.CTRLF) register.
7. Configure a positive input by writing to the MUXPOS bit field in the Positive Input Multiplexer (ADCn.MUXPOS) register.
8. Enable the ADC by writing a '1' to the ENABLE bit in the Control A (ADCn.CTRLA) register.
9. Configure the mode of operation for the ADC by writing to the MODE bit field in the Command (ADCn.COMMAND) register.
10. Configure how an ADC conversion will start by writing to the START bit field in the Command (ADCn.COMMAND) register.

31.3.3. Operation

31.3.3.1. Operation Modes

The ADC supports four operation modes, which are configured in the Command (ADCn.COMMAND) register.

The operation modes can be split into three groups:

- Single mode - Single conversion per trigger, with 8- or 10-bit conversion output
- Series Accumulation mode - One conversion per trigger, with accumulation of n samples
- Burst Accumulation mode - A burst with n samples accumulated as fast as possible after a single trigger

Series and Burst modes utilize 10-bit conversions. The SAMPNUM bit field in the Control F (ADCn.CTRLF) register controls the number of samples to be accumulated. The accumulator is always reset to zero when a new Series or Burst accumulation is started.

The table below shows an overview of the available operation modes.

Table 31-1. Operation Modes

Operation Mode	COMMAND Mode	Conversions Per Trigger	RESULT Update
Single 8-bit	0	1	Every conversion
Single 10-bit	1		
Series Accumulation	2	1	After SAMPNUM conversions
Burst Accumulation	3	SAMPNUM	After SAMPNUM conversions

In addition to the ordinary operating modes, an Accumulator Test mode is available. See the Accumulator Test section for more information.

31.3.3.2. Conversion Triggers

A conversion is started by one of the following triggers, depending on the configuration of the START bit field in the Command (ADCn.COMMAND) register:

- Writing the IMMEDIATE value to the START bit field in the Command (ADCn.COMMAND) register
- Receiving an event input
- Writing to the Positive Input Multiplexer (ADCn.MUXPOS) register

Continuously repeating Single conversion or Burst accumulation can be enabled by writing a '1' to the Free-Running (FREERUN) bit in the Control F (ADCn.CTRLF) register before starting the first conversion. Free-Running mode is not supported in Series mode.

Attempting to trigger a new conversion before the ongoing conversion is finished will set the Trigger Overrun Interrupt (TRIGOVR) flag in the Interrupt Flags (ADCn.INTFLAGS) register, and the trigger will be ignored.

The Result Ready and Sample Ready (RESRDY and SAMPRDY) interrupt flags in the Interrupt Flags (ADCn.INTFLAGS) register indicate whether a conversion or an accumulation has finished. These

flags also trigger the corresponding interrupts if enabled in the Interrupt Control (ADCn.INTCTRL) register.

31.3.3.3. Aborting a Conversion

These actions will abort an ongoing conversion:

- Writing STOP to the START bit field in the Command (ADCn.COMMAND) register
- Writing a new value that changes the Reference Selection bit field (REFSEL) in the Control C (ADCn.CTRLC) register during a conversion
- Writing a new value that changes Positive Input Multiplexer (ADCn.MUXPOS) register

This will result in undefined values in the Result (ADCn.RESULT) and Sample (ADCn.SAMPLE) registers.

31.3.3.4. Output Formats

The resolution of an ADC output is given by the number of bits used in the conversion:

- An **8-bit** ADC means 256 discrete levels (from 0 to 255)
- A **10-bit** ADC means 1024 discrete levels (from 0 to 1023)

A **single-ended** conversion measures the voltage difference between one input voltage and ground. The result of an N-bit single-ended conversion is an unsigned (positive) integer between 0 and the maximum value 2^N - 1:

```
ADC result ∈ [0, 2^N-1]
```

The output from an ADC conversion is given by the equations below:

Equation	Explanation
Single-Ended 8-bit conversion:	
$RESULT = \frac{V_{IN}}{V_{REF}} \times 255$	• The multiplication factor is 255 (2^8 - 1 = 255) • The result (RES) will be an integer between 0 and 255
Single-Ended 10-bit conversion:	
$RESULT = \frac{V_{IN}}{V_{REF}} \times 1023$	• The multiplication factor is 1023 (2^{10} - 1 = 1023) • The result (RES) will be an integer between 0 and 1023

Explanation to the equations:

- $\mathbf{V_{IN}}$ is the input voltage
- $\mathbf{V_{REF}}$ is the selected reference voltage

The ADC has two output registers, the Sample (ADCn.SAMPLE) and Result (ADCn.RESULT) registers. The 16-bit Sample register will always be updated with the latest ADC conversion output (one sample). In Series and Burst accumulation mode, samples are added in an internal sample accumulator, which is configured by the Sample Accumulation Number Select (SAMPNUM) bit field in the Control F (ADCn.CTRLF) register. The accumulated result will automatically transferred to the 16-bit Result register when all samples are completed. When accumulating more than 64 samples, only the 16 MSBs of the accumulated result will be available in the Result register. In single conversion modes, the Result register will be updated with the latest sample, identical to the Sample register.

The Result Scaling (SCALING) bit field in the Control F (ADCn.CTRLF) register enables the left-shifting of the output data. If enabled, this will left-shift the output in both the Result and Sample registers. Left-adjusting the result will effectively scale the result such that however many samples are accumulated, the MSb of the result is always positioned at the leftmost bit position in the Result register.

The Result Scaling (SCALING) bit field in the Control F (ADCn.CTRLF) register determines how data is presented in the SAMPLE and RESULT registers, as shown in the following table.

Table 31-2. Scaling

SCALING bit field value	Description	SAMPLE	RESULT
0x0	None	LSb at bit 0	Accumulated value with LSb at bit 0
0x1	Left Adjust	MSb at bit 15	Accumulated value scaled corresponding to number of samples. Up to 16 bits available with MSb at bit 15.
0x2	Average	LSb at bit 0	Accumulated value divided by the number of samples, with LSb at bit 0

The data format for a sample is an unsigned number, where `0x000` represents zero and `0x3FF` represents the largest number (full scale). If the analog input is higher than the reference level of the ADC, the 10-bit ADC output will be equal to the maximum value of `0x3FF`. Likewise, if the input is below 0V, the ADC output will be `0x000`.

The following table shows the Result register output formats by mode of operation and left adjustment.

Table 31-3. RESULT Register

MODE(1)	SCALING	RESULT[15:10]	RESULT[9:8]	RESULT[7:0]
0	X(2)	0x00		Conversion[9:2]
1	0, 2	0x00	Conversion[9:0]	
	1	Conversion[9:0] << 6		
2, 3	0(3)	Accumulation[15:0]		
	1(3)	Left Adjusted Accumulation[15:0]		
	2	0x00	Averaged Accumulation[9:0]	

Notes:

1. See the Operation Modes section.
2. Left adjustment is not available in 8-bit mode.
3. When accumulating more than 64 samples (SAMPNUM > 6), only the 16 MSBs of the accumulated result will be available in the Result register, regardless of the value of the Result Scaling (SCALING) bit in the Control F (ADCn.CTRLF) register.

The following table shows the Sample register output formats by mode of operation and left adjustment.

Table 31-4. SAMPLE Register

MODE(1)	SCALING	SAMPLE[15:10]	SAMPLE[9:8]	SAMPLE[7:0]
0	X	0x00		Conversion[9:2]
Other	0, 2	0x00	Conversion[9:0]	
	1	Conversion[9:0] << 6		

Note:

1. See the Operation Modes section.

31.3.3.5. ADC Clock

The ADC clock (CLK_ADC) is down-scaled from the peripheral clock (CLK_PER), which can be configured using the Prescaler (PRESC) bit field in the Control B (ADCn.CTRLB) register. Refer to the ADC section of the Electrical Characteristics section for details on the minimum and maximum allowed values for the ADC clock (CLK_ADC) period.

Some of the internal timings in the ADC are independent of CLK_ADC. To ensure correct internal timing regardless of the ADC clock frequency, a 1 µs timebase, given in CLK_PER cycles, must be written to the Timebase (MCLKTIMEBASE) register in the Clock Controller (CLKCTRL) peripheral. For more details, refer to the MCLKTIMEBASE register description in the *CLKCTRL - Clock Controller* section.

31.3.3.6. Input and Reference Selection

The input selection for the ADC is controlled by the Positive Input Multiplexer (ADCn.MUXPOS) register.

The reference voltage for the ADC (V_{REF}) controls the conversion range of the ADC. V_{REF} can be selected by writing the Reference Selection (REFSEL) bit field in the Control C (ADCn.CTRLC) register. Except for V_{DD}, the internal reference voltages are generated from an internal band gap reference.

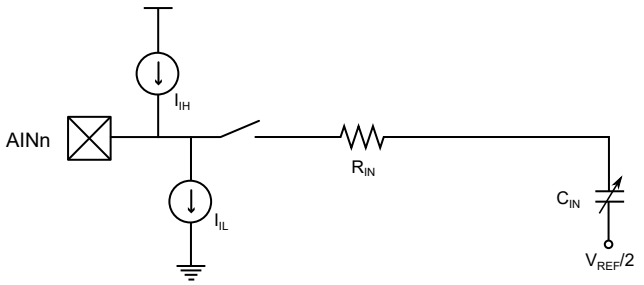
Changing the input or reference selections while a conversion is ongoing will corrupt the output. An ongoing conversion can be detected by reading the ADC Busy (ADCBUSY) bit in the Status (ADCn.STATUS) register. Writing STOP to the Start Conversion (START) bit field in the Command (ADCn.COMMAND) register will stop an ongoing conversion.

After switching the input or reference, the ADC requires time to settle. Refer to the *Electrical Characteristics* section for further details.

31.3.3.6.1. Analog Input Circuit

The figure below illustrates the analog input circuit. An analog source connected to an analog input (AINn) is subject to the pin capacitance and input leakage of that pin (represented by I_H and I_L). When the input is selected, the source must also drive the Sample-Hold capacitor (C_{IN}) through the combined resistance of the input path (represented by R_{IN}). Refer to the *Electrical Characteristics* section for details on the input characteristics of the ADC.

Figure 31-2. Analog Input Schematic



If a source with high impedance is used, the sampling time may need to be increased. The required sample time will depend on how long the source needs to charge the C_{IN} capacitor and can be configured using the Sample Duration (SAMPDUR) bit field in the Control E (ADCn.CTRLE) register.

31.3.3.7. Conversion Timing

Some of the analog modules in the ADC are disabled between conversions and require time to initialize before the conversion starts. Only the modules used by the current ADC configuration are enabled, and since the initializations run in parallel, the limiting factor is the module with the slowest initialization time.

By writing the Low Latency (LOWLAT) bit in the Control A (ADCn.CTRLA) register to '1', the latency of the ADC peripheral can be reduced. This will keep the configured modules continuously enabled, effectively removing all initialization time at the start of a conversion. Initialization time is still needed when enabling the ADC for the first time and reconfiguring the ADC to use an input or reference that requires initialization. The ADC Busy (ADCBUSY) bit in the Status (ADCn.STATUS) register can be used to check if initialization is ongoing.

The following table shows the initialization times required by the different analog modules.

Table 31-5. ADC Initialization Timing

Analog Module	LOWLAT	Initialization Time (µs)
ADC	0	10
	1(1)	0
Settling of internal references	0	60
	1(1)	2(2)
Settling of internal references after changing from one internal reference voltage to another	X	35
Settling of external reference or VDD as the reference	X	2
Settling of internal Temperature Sensor input	0	60
	1(1)	0
Settling of internal Analog Comparator Reference DAC input	0	35
	1(1)	0
Selecting VDD/10 as the input	X	2

Notes:

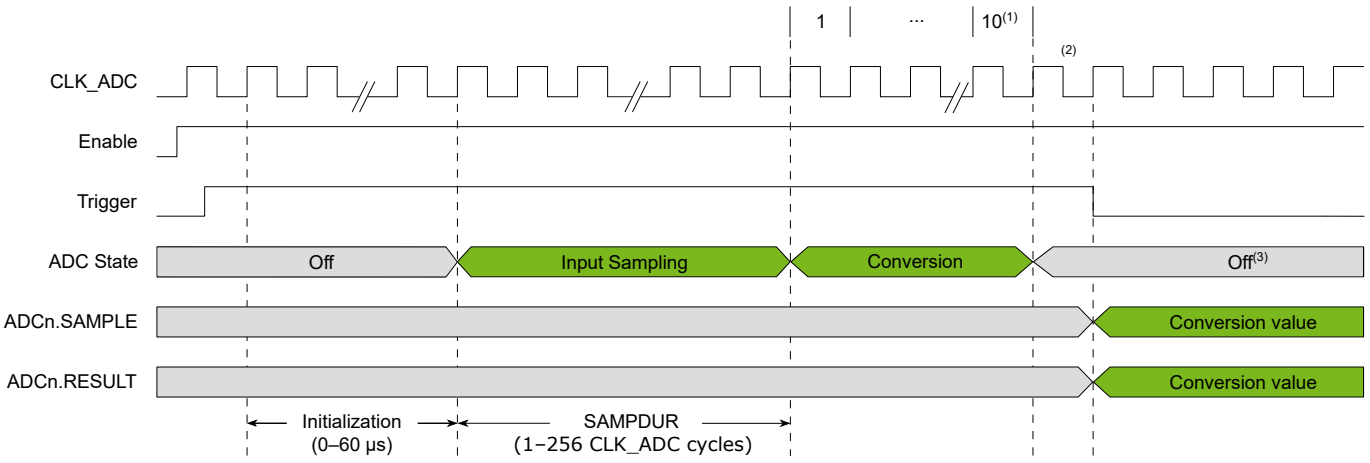
1. The LOWLAT timing values are valid between two conversions that both have the LOWLAT bit written to '1'.
2. When changing from one internal reference voltage to another internal reference voltage, the full 60 µs delay will be used if changing from a non-internal reference to an internal reference.

The sampling period of the input to the ADC is configured through the Sample Duration (SAMPDUR) bit field in the Control E (ADCn.CTRLE) register as *SAMPDUR* + *1* CLK_ADC cycles.

31.3.3.7.1. Single Conversion

The figure below illustrates the timing diagram for the ADC when running in Single 8- or 10-bit mode.

Figure 31-3. Timing Diagram - Single Conversion



Notes:

1. In Single 8-bit mode, the length of the Conversion state is eight CLK_ADC cycles. In all other modes, it is ten cycles.
2. The time from when the conversion has finished to when the outputs are available in the registers is 0.5 CLK_ADC cycles followed by one CLK_PER cycle. With a minimum prescaler of two, this sums up to a maximum of one CLK_ADC cycle.
3. If the Low Latency (LOWLAT) bit is set to '1' in the Control A (ADCn.CTRLA) register, the analog modules in the ADC will not turn OFF at the end of the conversion, which eliminates the initialization time when triggering the next conversion.

The total conversion time for a single result is calculated as follows:

$$t_{conv}\,(\text{10-bit}) = t_{initialization} + \frac{\text{SAMPDUR}+1+10+1}{f_{\text{CLK\_ADC}}}$$

$$t_{conv}\,(\text{8-bit}) = t_{initialization} + \frac{\text{SAMPDUR}+1+8+1}{f_{\text{CLK\_ADC}}}$$

If the Free-Running (FREERUN) bit is set to '1' in the Control F (ADCn.CTRLF) register, a new conversion will start immediately after a result is available in the Result (ADCn.RESULT) register. The Free-Running conversion rate (f_{conv}) is calculated as follows:

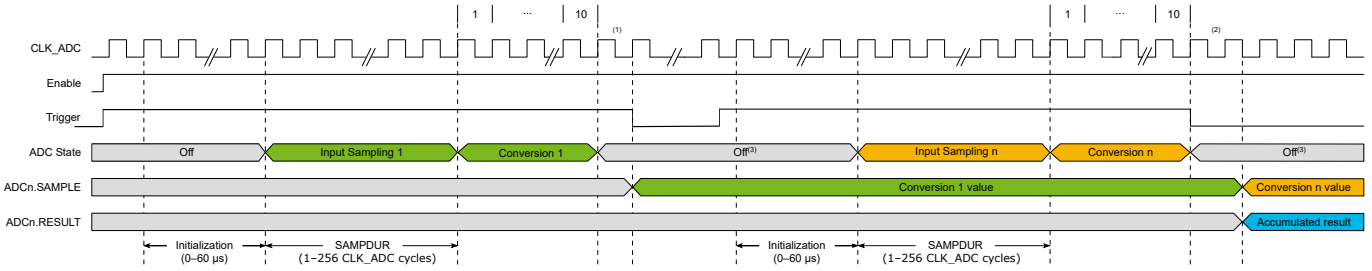
$$f_{\text{conv}}\,(\text{10-bit}) = \frac{f_{\text{CLK\_ADC}}}{\text{SAMPDUR}+1+10+1}$$

$$f_{\text{conv}}\,(\text{8-bit}) = \frac{f_{\text{CLK\_ADC}}}{\text{SAMPDUR}+1+8+1}$$

31.3.3.7.2. Series Accumulation

The figure below illustrates the timing diagram for the ADC when running in Series Accumulation mode.

Figure 31-4. Timing Diagram - Series Accumulation



Notes:

1. The time from when the conversion has finished to when the outputs are available in the registers is 0.5 CLK_ADC cycles followed by one CLK_PER cycle. With a minimum prescaler of two, this sums up to a maximum of one CLK_ADC cycle.
2. The time from when the final conversion has finished to when the Result (ADCn.RESULT) register is updated is 0.5 CLK_ADC cycles followed by two CLK_PER cycles. With a minimum prescaler of two, this sums up to the maximum of 1.5 CLK_ADC cycles.
3. If the Low Latency (LOWLAT) bit is set to '1' in the Control A (ADCn.CTRLA) register, the analog modules in the ADC will not turn OFF at the end of the conversion, which eliminates the initialization time when triggering the next conversion.

The number of samples to accumulate is set by the Sample Number (SAMPNUM) bit field in the Control F (ADCn.CTRLF) register.

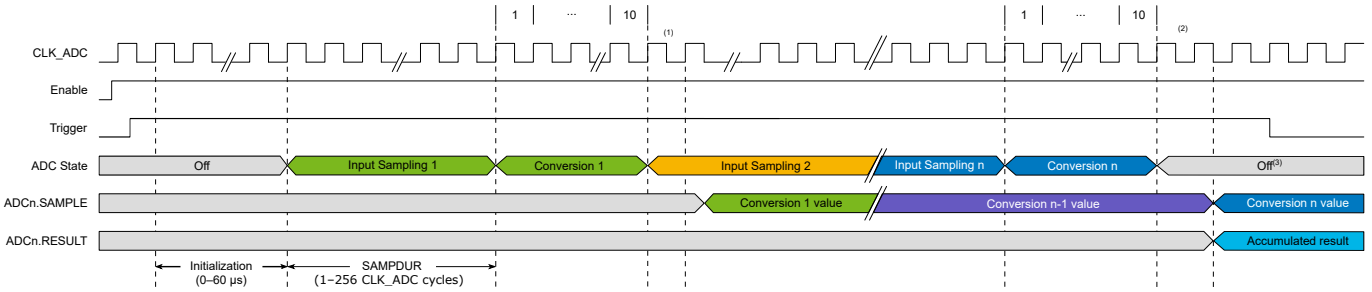
The total conversion time for each separate sample is calculated as follows:

$$t_{conv} = t_{initialization} + \frac{\text{SAMPDUR}+1+10+1}{f_{\text{CLK\_ADC}}}$$

31.3.3.7.3. Burst Accumulation

The figure below illustrates the timing diagram for the ADC when running in Burst Accumulation mode.

Figure 31-5. Timing Diagram - Burst Accumulation



Notes:

1. The time from when the conversion has finished to when the outputs are available in the registers is 0.5 CLK_ADC cycles followed by one CLK_PER cycle. With a minimum prescaler of two, this sums up to a maximum of one CLK_ADC cycle.
2. The time from when the final conversion has finished to when the Result (ADCn.RESULT) register is updated is 0.5 CLK_ADC cycles followed by two CLK_PER cycles. With a minimum prescaler of two, this sums up to a maximum of 1.5 CLK_ADC cycles.
3. If the Low Latency (LOWLAT) bit is set to '1' in the Control A (ADCn.CTRLA) register, the analog modules in the ADC will not turn OFF at the end of the conversion, which eliminates the initialization time when triggering the next conversion.

The number of samples to accumulate is set by the Sample Number (SAMPNUM) bit field in the Control F (ADCn.CTRLF) register.

The total conversion time for a Burst Accumulation is calculated as follows:

$$t_{conv} = t_{initialization} + \frac{(\text{SAMPDUR}+1+10) \times \text{SAMPNUM} \ + \ 1.5}{f_{\text{CLK\_ADC}}}$$

The Burst Accumulation conversion rate (f_{conv}) is calculated as follows:

$$f_{\text{conv}} = \frac{f_{\text{CLK\_ADC}}}{\text{SAMPDUR}+1+10}$$

31.3.3.8. Temperature Measurement

An on-chip temperature sensor is available. To perform a temperature measurement, follow these steps:

1. Configure the voltage reference to internal 1.024V by writing to the Reference Selection (REFSEL) bit field in the Control C (ADCn.CTRLC) register.
2. Select the temperature sensor as the input in the Positive Input Multiplexer (ADCn.MUXPOS) register.
3. Configure the ADC Sample Duration by writing a value ≥ 60 µs × $f_{\text{CLK\_ADC}}$ to the Sample Duration (SAMPDUR) bit field in the Control E (ADCn.CTRLE) register.
4. Acquire the temperature sensor output voltage by running a 10-bit Single-Ended conversion.

5. Process the measurement result as described below.

The measured voltage has a linear relationship to temperature. The relationship between the ADC conversion result and temperature in Kelvin is tested in production and is available in the Signature Row:

- SIGROW.TEMPSENSE0: gain/slope correction
- SIGROW.TEMPSENSE1: offset correction

Use the following equation to calculate the temperature in Kelvin:

$$T = \frac{(\text{ADC Result} + \text{Offset}) \times \text{Slope}}{1024}$$

It is possible to perform further calibration to achieve even more accurate results. For example, multi-point calibration can be performed by imposing known temperatures on the device and recording the corresponding ADC results. This helps create a precise calibration curve, which can be implemented in firmware to adjust the output result.

31.3.3.9. Window Comparator

The ADC features a window comparator, which automatically compares the conversion result to two configurable thresholds. The comparison result is indicated by the WCMP bit in the Interrupt Flags (ADCn.INTFLAGS) register and can optionally trigger the corresponding interrupt. The available modes are:

- The value is above a threshold
- The value is below a threshold
- The value is inside a window (above the lower threshold and below the upper threshold)
- The value is outside a window (either below the lower threshold or above the upper threshold)

The thresholds are set by writing to the Window Comparator Low and High Threshold (ADCn.WINLT and ADCn.WINHT) registers. The Window mode to used is selected by the Window Comparator Mode (WINCM) bit field in the Control D (ADCn.CTRLD) register.

The Window Mode Source (WINSRC) bit in the Control D (ADCn.CTRLD) register selects if the comparison is done on the Result (ADCn.RESULT) register or the Sample (ADCn.SAMPLE) register. If an interrupt request is enabled for the WCMP flag, WINSRC selects which interrupt vector to request: RESRDY or SAMPRDY.

When accumulating multiple samples, if the Window Comparator source is the Result register, the comparison between the result and the threshold(s) occurs after the last conversion is complete. If the source is the Sample register, the comparison occurs after every conversion.

Assuming the ADC is already configured to run, follow these steps to use the Window Comparator mode:

1. Set the required threshold(s) by writing to the Window Comparator Low and High Threshold (ADCn.WINLT and ADCn.WINHT) registers.
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 by writing the WINSRC bit field and writing a non-zero value to the WINCM bit field in the Control D (ADCn.CTRLD) register.

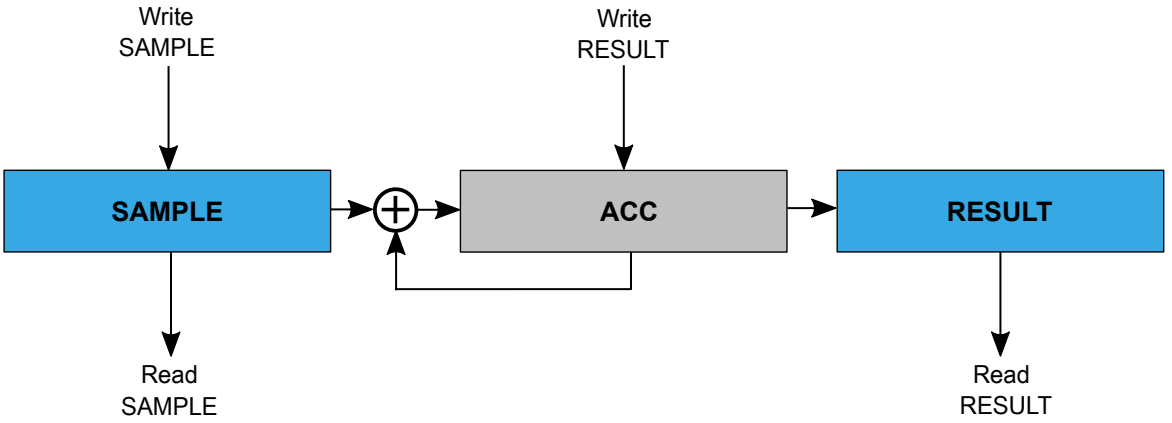
31.3.3.10. Accumulator Test

The Accumulator Test mode aids in diagnostics of the accumulator and result formatting hardware, and can test the integrity of the logic. It is enabled by configuring the MODE bit field in the Command (ADCn.COMMAND) register to ACCTEST.

The Accumulator Test mode is used as follows:

- Writing to the Result (ADCn.RESULT) register writes to the internal 20-bit accumulator
- Writing to the Sample (ADCn.SAMPLE) register accumulates the 10 LSb of the written value in the internal 20-bit accumulator
- Reading the Result register will read the 16 bits of the accumulator

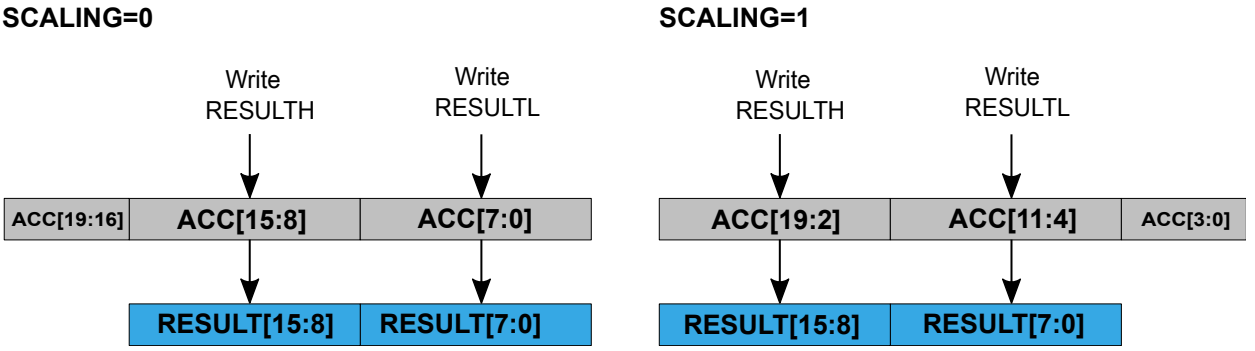
Figure 31-6. Accumulator Test



Notes:

1. If the Result Scaling (SCALING) bit field in the Control F (ADCn.CTRLF) register is '1', a value written to the accumulator will be left-adjusted by 4 bits.
2. If the SCALING bit field is '1', the 16 MSb of the 20-bit internal accumulator will be read; if SCALING bit is '0', the 16 LSb of the 20-bit internal accumulator will be read.

Figure 31-7. Register Access and Adjustment



31.3.4. Events

The ADC can generate the following events:

Table 31-6. ADC Event Generators

Generator Name		Description	Event Type	Generating Clock Domain	Length of Event
Peripheral	Event				
ADCn	RESRDY	Result ready	Pulse	CLK_PER	One CLK_PER period
ADCn	SAMPRDY	Sample ready	Pulse	CLK_PER	One CLK_PER period
ADCn	WCMP	Window compare match	Pulse	CLK_PER	One CLK_PER period

The conditions for generating an event are identical to those that will raise the corresponding flag in the Interrupt Flags (ADCn.INTFLAGS) register.

The ADC has one event user for detecting and acting upon input events. The table below describes the event user and the associated functionality.

Table 31-7. ADC Event Users and Available Event Actions

User Name		Description	Input Detection	Async/Sync
Peripheral	Event			
ADCn	START	ADC start on event	Edge	Async

The START event action can be triggered if the EVENT_TRIGGER setting is written to the START bit field in the Command (ADCn.COMMAND) register.

31.3.5. Interrupts

Table 31-8. Available Interrupt Vectors and Sources

Vector Name	Source Name	Condition	Dependency
ADCn_ERROR	TRIGOVR	A new conversion is triggered while another is in progress	
	SAMPOVR	A new conversion overwrites an unread sample in the Sample (ADCn.SAMPLE) register	
	RESOVR	A new conversion or accumulation overwrites an unread result in the Result (ADCn.RESULT) register	
ADCn_RESRDY	RESRDY	A new result is available in the Result (ADCn.RESULT) register	
	WCMP	A conversion or accumulation matches the conditions set by the window comparator	The Window Source (WINSRC) bit in the Control D (ADCn.CTRLD) register is '0'
ADCn_SAMPRDY	SAMPRDY	A new sample is available in the Sample (ADCn.SAMPLE) register	
	WCMP	A sample matches the conditions set by the window comparator	The WINSRC bit in ADCn.CTRLD is '1'

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.

31.3.6. Sleep Mode Operation

The ADC can start conversions in Idle sleep mode if the START bit field in the Command (ADCn.COMMAND) register is configured to start a conversion on an event trigger. This is also possible in Standby sleep mode if the RUNSTDBY bit is set in the Control A (ADCn.CTRLA) register. For both cases, the ADC will finish any ongoing conversion or burst accumulation when transitioning to sleep.

If both the LOWLAT and RUNSTDBY bits in the Control A register are set, the ADC will keep all required modules ON during Standby sleep mode to start a conversion faster, at the expense of increased power consumption during sleep. A clock startup delay may be experienced as described in the *SLPCTRL - Sleep Controller* and *Electrical Characteristics* sections.

When the system enters POWERDOWN, the ADC will abort an ongoing conversion and enter sleep mode immediately. Make sure conversions have completed before entering Power-Down mode.

31.3.7. Debug Operation

If the Run in Debug mode (DBGRUN) bit in the Debug Control (ADCn.DBGCTRL) register is written to '1', the ADC will continue operating when the CPU is halted in Debug mode.

If DBGRUN is '0' when the CPU halts, an ongoing conversion will finish before the ADC halts.

31.4. Register Summary

Offset	Name	Bit Pos.	7	6	5	4	3	2	1	0
0x00	CTRLA	7:0	RUNSTDBY		LOWLAT					ENABLE
0x01	CTRLB	7:0				PRESC[4:0]				
0x02	CTRLC	7:0						REFSEL[2:0]		
0x03	CTRLD	7:0					WINSRC	WINCM[2:0]		
0x04	CTRLE	7:0	SAMPDUR[7:0]							
0x05	CTRLF	7:0		FREERUN	SCALING[1:0]		SAMPNUM[3:0]			
0x06	INTCTRL	7:0			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
0x07	INTFLAGS	7:0			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
0x08	STATUS	7:0								ADCBUSY
0x09	DBGCTRL	7:0								DBGRUN
0x0A	COMMAND	7:0		MODE[2:0]				START[2:0]		
0x0B	MUXPOS	7:0		MUXPOS[6:0]						
0x0C	RESULT	7:0	RESULT[7:0]							
		15:8	RESULT[15:8]							
0x0E	SAMPLE	7:0	SAMPLE[7:0]							
		15:8	SAMPLE[15:8]							
0x10	WINLT	7:0	WINLT[7:0]							
		15:8	WINLT[15:8]							
0x12	WINHT	7:0	WINHT[7:0]							
		15:8	WINHT[15:8]							
0x14	TEMP	7:0	TEMP[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		LOWLAT					ENABLE
Access	R/W		R/W					R/W
Reset	0		0					0

Bit 7 – RUNSTDBY Run in Standby
This bit controls whether the ADC will run in Standby sleep mode or not.

Value	Description
0	The ADC will not run in Standby sleep mode. An ongoing conversion will finish before the ADC enters sleep mode.
1	The ADC will run in Standby sleep mode. The main clock will be requested when the ADC is triggered to perform a conversion.

Bit 5 – LOWLAT Low Latency
This bit controls whether the analog modules required by the ADC are enabled continuously or only when needed.

Value	Description
0	The ADC enables the required analog modules only when starting a conversion, which reduces the overall power consumption of the ADC and increases the initialization time when starting an ADC conversion.
1	The analog modules stay enabled when selected as input to the ADC. Using this setting will minimize the initialization time of the ADC.

Bit 0 – ENABLE ADC Enable
This bit controls whether the ADC is enabled or not.

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

31.5.2. Control B

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

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

Bits 4:0 – PRESC[4:0] Prescaler
This bit field controls the division factor from the peripheral clock (CLK_PER) to the ADC clock (CLK_ADC).
The generated frequency is f_{CLK_PER} / (PRESC+2)

Value	Name	Description
0x0	DIV2	CLK_PER divided by 2
0x1	DIV3	CLK_PER divided by 3
0x2	DIV4	CLK_PER divided by 4
0x3	DIV5	CLK_PER divided by 5
0x4	DIV6	CLK_PER divided by 6
0x5	DIV7	CLK_PER divided by 7
0x6	DIV8	CLK_PER divided by 8
0x7	DIV9	CLK_PER divided by 9
0x8	DIV10	CLK_PER divided by 10
0x9	DIV11	CLK_PER divided by 11
0xA	DIV12	CLK_PER divided by 12
0xB	DIV13	CLK_PER divided by 13
0xC	DIV14	CLK_PER divided by 14
0xD	DIV15	CLK_PER divided by 15
0xE	DIV16	CLK_PER divided by 16
0xF	DIV17	CLK_PER divided by 17
0x10	DIV18	CLK_PER divided by 18
0x11	DIV19	CLK_PER divided by 19
0x12	DIV20	CLK_PER divided by 20
0x13	DIV21	CLK_PER divided by 21
0x14	DIV22	CLK_PER divided by 22
0x15	DIV23	CLK_PER divided by 23
0x16	DIV24	CLK_PER divided by 24
0x17	DIV25	CLK_PER divided by 25
0x18	DIV26	CLK_PER divided by 26
0x19	DIV27	CLK_PER divided by 27
0x1A	DIV28	CLK_PER divided by 28
0x1B	DIV29	CLK_PER divided by 29
0x1C	DIV30	CLK_PER divided by 30
0x1D	DIV31	CLK_PER divided by 31
0x1E	DIV32	CLK_PER divided by 32
0x1F	-	RESERVED

31.5.3. Control C

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

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

Bits 2:0 – REFSEL[2:0] Reference Selection
This bit field controls the voltage reference for the ADC. Changing to one of the internal references will require a 40 µs initialization time for the first conversion.

Value	Name	Description
0x0	VDD	V_{DD}
0x1	-	Reserved
0x2	EXTVREF	External Reference VREFA for ADC0
0x3	0V512	Internal reference 0.512V
0x4	1V024	Internal reference 1.024V
0x5	2V048	Internal reference 2.048V
0x6	4V096	Internal reference 4.096V
0x7	2V500	Internal reference 2.500V

Note: The internal references can be used only if lower than V_{DD} - 0.5V.

31.5.4. Control D

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

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

Bit 3 – WINSRC Window Mode Source
This bit controls which source is used by the Window Comparator.

Value	Name	Description
0	RESULT	ADCn.RESULT is used as the Window Comparator source
1	SAMPLE	ADCn.SAMPLE is used as the Window Comparator source

Bits 2:0 – WINCM[2:0] Window Comparator Mode
This bit field controls whether the Window Comparator is enabled and which thresholds will set the Window Comparator (WCMP) interrupt flag.
In the table below, OUTPUT is the 16-bit result or sample selected by WINSRC. WINLT and WINHT are the 16-bit low and high threshold registers, respectively.

Value	Name	Description
0x0	NONE	Window Comparator disabled
0x1	BELOW	*OUTPUT < WINLT*
0x2	ABOVE	*OUTPUT > WINHT*
0x3	INSIDE	*WINLT < OUTPUT < WINHT*
0x4	OUTSIDE	*OUTPUT < WINLT or OUTPUT >WINHT*
Other	-	Reserved

31.5.5. Control E

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

Bit	7	6	5	4	3	2	1	0
	SAMPDUR[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 – SAMPDUR[7:0] Sample Duration
This bit field selects the ADC sample duration in ADC clock (CLK_ADC) cycles. The sample duration is *SAMPDUR* + *1* CLK_ADC cycles.

31.5.6. Control F

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

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

Bit 6 – FREERUN Free-Running
This bit controls whether the ADC Free-Running mode is enabled.
Free-Running mode is not supported in Series mode.

Value	Name	Description
0x0	DISABLE	The ADC Free-Running mode is disabled
0x1	ENABLE	The ADC Free-Running mode is enabled. The first conversion is started by writing the IMMEDIATE setting to the START bit field in the Command (ADCn.COMMAND) register. In Free-Running mode, a new conversion is started as soon as the previous conversion or accumulation is completed, which is signaled by the RESRDY flag in the Interrupt Flags (ADCn.INTFLAGS) register.

Bits 5:4 – SCALING[1:0] Result Scaling
This bit field is used to control the scaling of the digital value in the SAMPLE and RESULT registers. The NORMAL and LEFTADJ modes are truncated when accumulating more than 64 samples (result exceeds 16 bits), while the AVERAGE is rounded.

Value	Name	Description
0x0	NORMAL	The ADC output of sample or accumulated result is right-adjusted. If accumulating more than 64 samples, result is left-adjusted.
0x1	LEFTADJ	ADC output left-adjusted
0x2	AVERAGE	The accumulated ADC result is averaged and right-adjusted

Bits 3:0 – SAMPNUM[3:0] Sample Accumulation Number Select
This bit field selects the number of consecutive ADC samples that are automatically accumulated into the ADC Result (ADCn.RESULT) register. The most recent sample will be available in the ADC Sample (ADCn.SAMPLE) register.

Value	Name	Description
0x0	NONE	No accumulation, single sample per conversion result
0x1	ACC2	2 samples accumulated
0x2	ACC4	4 samples accumulated
0x3	ACC8	8 samples accumulated
0x4	ACC16	16 samples accumulated
0x5	ACC32	32 samples accumulated
0x6	ACC64	64 samples accumulated
0x7	ACC128	128 samples accumulated
0x8	ACC256	256 samples accumulated
0x9	ACC512	512 samples accumulated
0xA	ACC1024	1024 samples accumulated
Other	-	Reserved

31.5.7. Interrupt Control

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

Bit	7	6	5	4	3	2	1	0
			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TRIGOVR Trigger Overrun Interrupt Enable
This bit controls whether the interrupt for a trigger overrun is enabled.

Value	Description
0	The Trigger Overrun interrupt is disabled
1	The Trigger Overrun interrupt is enabled

Bit 4 – SAMPOVR Sample Overwrite Interrupt Enable
This bit controls whether the interrupt for a sample overwrite is enabled.

Value	Description
0	The Sample Overwrite interrupt is disabled
1	The Sample Overwrite interrupt is enabled

Bit 3 – RESOVR Result Overwrite Interrupt Enable
This bit controls whether the interrupt for a result overwrite is enabled.

Value	Description
0	The Result Overwrite interrupt is disabled
1	The Result Overwrite interrupt is enabled

Bit 2 – WCMP Window Comparator Interrupt Enable
This bit controls whether the interrupt for the Window Comparator is enabled.

Value	Description
0	The Window Comparator interrupt is disabled
1	The Window Comparator interrupt is enabled

Bit 1 – SAMPRDY Sample Ready Interrupt Enable
This bit controls whether the Sample Ready interrupt is enabled.

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

Bit 0 – RESRDY Result Ready Interrupt Enable
This bit controls whether the Result Ready interrupt is enabled.

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

31.5.8. Interrupt Flags

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

Bit	7	6	5	4	3	2	1	0
			TRIGOVR	SAMPOVR	RESOVR	WCMP	SAMPRDY	RESRDY
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – TRIGOVR Trigger Overrun Interrupt Flag
Clear this flag by writing a '1' to it.
This flag is set when a start trigger is received while a conversion is ongoing.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Trigger Overrun interrupt flag.

Bit 4 – SAMPOVR Sample Overwrite Interrupt Flag
Clear this flag by writing a '1' to it.
This flag is set when an unread sample is overwritten in the Sample (ADCn.SAMPLE) register.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Sample Overwrite interrupt flag.

Bit 3 – RESOVR Result Overwrite Interrupt Flag
Clear this flag by writing a '1' to it.
This flag is set when an unread result is overwritten in the Result (ADCn.RESULT) register.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Result Overwrite interrupt flag.

Bit 2 – WCMP Window Comparator Interrupt Flag
Clear this flag by writing a '1' to it.
This flag is set when the conversion or accumulation is complete and the thresholds match the selected window comparator source and mode, as set by WINSRC and WINCM in the Control D (ADCn.CTRLD) register.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Window Comparator interrupt flag.

Bit 1 – SAMPRDY Sample Ready Interrupt Flag
Clear this flag by writing a '1' to it or by reading the Sample (ADCn.SAMPLE) register.
This flag is set when a conversion is complete and a new sample is ready.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Sample Ready interrupt flag.

Bit 0 – RESRDY Result Ready Interrupt Flag
Clear this flag by writing a '1' to it or by reading the Result (ADCn.RESULT) register.
This flag is set when a conversion or accumulation is complete and a new result is ready.
Writing a '0' to this bit has no effect.
Writing a '1' to this bit will clear the Result Ready interrupt flag.

31.5.9. Status

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

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

Bit 0 – ADCBUSY ADC Busy

This bit is cleared when an ADC conversion is complete and settling times related to configuration changes have finished.
This bit is set when the ADC is performing a conversion or waiting for settling times related to configuration changes.

31.5.10. Debug Control

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

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

Bit 0 – DBGRUN Run in Debug Mode
This bit controls whether the ADC continues to operate when in Debug mode and the CPU is halted.

Value	Description
0	The ADC will not continue operating in Debug mode when the CPU is halted. Any ongoing conversion or burst accumulation will finish before the ADC stops.
1	The ADC will continue operating in Debug mode when the CPU is halted

31.5.11. Command

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

Bit	7	6	5	4	3	2	1	0
		MODE[2:0]				START[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 – MODE[2:0] Mode
This bit field controls the conversion mode for the ADC. Switching from one of the accumulation modes to Single mode will reset the accumulator.
For more information on the operation modes, see the *Operation Modes* and *Accumulator Test* sections.

Value	Name	Description
0x0	SINGLE_8BIT	Single conversion with 8-bit resolution
0x1	SINGLE_10BIT	Single conversion with 10-bit resolution
0x2	SERIES	Series with accumulation, with a separate trigger for each 10-bit conversion
0x3	BURST	Burst with accumulation. One trigger will run SAMPNUM 10-bit conversions in a single sequence.
0x4 - 0x6	-	Reserved
0x7	ACCTEST	Accumulator diagnostics mode for Functional Safety applications
Other	-	Reserved

Bits 2:0 – START[2:0] Start Conversion
This bit field starts or stops an ADC conversion, or controls how an ADC conversion will start.

Value	Name	Description
0x0	STOP	Stop an ongoing conversion
0x1	IMMEDIATE	Start a conversion immediately. The bit field value will be set back to STOP after the conversion is completed, unless Free-Running mode is enabled.
0x2	MUXPOS_WRITE	Start when a write to the MUXPOS register is done
0x4	EVENT_TRIGGER	Start when an event is received by the ADC
Other	-	Reserved

Note: If the ENABLE bit in ADCn.CTRLA is '0' when the START bit field is written to IMMEDIATE, it will automatically be set to STOP.

31.5.12. Positive Input Multiplexer

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

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

Bits 6:0 – MUXPOS[6:0] Positive Input Multiplexer
This bit field controls which analog input is connected to the positive input of the ADC. Changing this setting may require some settling time. Refer to the *Electrical Characteristics* section for further details.

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
0x08-0x0F	-	Reserved
0x10	AIN16	ADC input pin 16
0x11	AIN17	ADC input pin 17
0x12	AIN18	ADC input pin 18
0x13	AIN19	ADC input pin 19
0x14	AIN20	ADC input pin 20
0x15	AIN21	ADC input pin 21
0x16	AIN22	ADC input pin 22
0x17	AIN23	ADC input pin 23
0x18	AIN24	ADC input pin 24
0x19	AIN25	ADC input pin 25
0x1A	AIN26	ADC input pin 26
0x1B	AIN27	ADC input pin 27
0x1C-0x28	-	Reserved
0x29	TEMPSENSE	Temperature sensor (for 1V reference)
0x2A	VDDDIV10	V_{DD} divided by 10
0x2B	AC0REFSCALE	AC0 Reference Scaler
0x2C-0x3D	-	Reserved
0x3E	PTC	Peripheral Touch Controller
Other	-	Reserved

31.5.13. Result

Name: RESULT
Offset: 0x0C
Reset: 0x00
Property: -

The ADCn.RESULTL and ADCn.RESULTH registers together represent the 16-bit value, ADCn.RESULT. 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 is also used and writable in the Functional Safety test mode, ACCTEST.

Refer to the *Output Formats* section for details on the output from this register.

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

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

Bits 15:8 – RESULT[15:8] Result byte 1
This bit field constitutes the MSB of the ADCn.RESULT register.

Bits 7:0 – RESULT[7:0] Result byte 0
This bit field constitutes the LSB of the ADCn.RESULT register.

31.5.14. Sample

Name: SAMPLE
Offset: 0x0E
Reset: 0x0000
Property: -

The ADCn.SAMPLEL and ADCn.SAMPLEH register pair represents the 16-bit value, ADCn.SAMPLE. 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`.

Refer to the *Output Formats* section for details on the output from this register.

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

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

Bits 15:8 – SAMPLE[15:8] Sample high byte
This bit field constitutes the MSB of the 16-bit register.

Bits 7:0 – SAMPLE[7:0] Sample low byte
This bit field constitutes the LSB of the 16-bit register.

31.5.15. Window Comparator Low Threshold

Name: WINLT
Offset: 0x10
Reset: 0x00
Property: -

This register is the 16-bit Low Threshold for the digital comparator monitoring the ADC Result or Sample (ADCn.RESULT or ADCn.SAMPLE) registers. The data format must be according to Conversion mode and left adjustment setting.

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 monitoring the ADC Result register, in an accumulation mode, the window comparator thresholds are applied to the result after all accumulation and, optionally, scaling is done.

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
This bit field holds the MSB of the 16-bit register.

Bits 7:0 – WINLT[7:0] Window Comparator Low Threshold low byte
This bit field holds the LSB of the 16-bit register.

31.5.16. Window Comparator High Threshold

Name: WINHT
Offset: 0x12
Reset: 0x00
Property: -

This register is the 16-bit High Threshold for the digital comparator monitoring the ADC Result or Sample (ADCn.RESULT or ADCn.SAMPLE) registers. The data format must be according to Conversion mode and left adjustment setting.

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`.

When monitoring the ADC Result register, in an accumulation mode, the window comparator thresholds are applied to the result after all accumulation and, optionally, scaling is done.

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
This bit field holds the MSB of the 16-bit register.

Bits 7:0 – WINHT[7:0] Window Comparator High Threshold low byte
This bit field holds the LSB of the 16-bit register.

31.5.17. Temporary

Name: TEMP
Offset: 0x14
Reset: 0x00
Property: -

Temporary register for read/write operations in the ADCn.RESULT and ADCn.SAMPLE registers. 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 *Memories* 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 bit field for read/write operations in 16-bit registers.

32. PTC - Peripheral Touch Controller

32.1. Features

- Low-Power, High-Sensitivity, Environmentally Robust Capacitive Touch Buttons, Sliders, Wheels and 2D Surface
- Supports Mix-and-Match Mutual Capacitance and Self-Capacitance Sensing
  - Supports 11/17/23/27 PTC pins for the 14-/20-/28-/32- pin variants, respectively
- Supports Lumped Mode that allows the Integration of Multiple Sensors, Configuring them to function as a Single Sensor
- One Pin per Electrode – No External Components
- Load Compensating Charge Sensing:
  - Parasitic capacitance compensation
  - Adjustable gain for superior sensitivity
- Zero Drift Over the Temperature and V_{DD} Range:
  - Auto-calibration and recalibration of sensors
- Hardware Noise Filtering and Noise Signal Desynchronization for High-Conducted Immunity
- Supports Analog Accumulation. Digital accumulation supported with ADC interface
- Driven Shield+ for Superior Noise Immunity and Moisture Tolerance
  - Any PTC X/Y line can be used for the driven shield
- Supports External Integration Capacitors (EXTCINT) for Large Capacitance Sensors
- Boost Mode for Doubled Signal-to-Noise Ratio or 4x Faster Response Time in Mutual Capacitance Systems
- Uses ADC Peripheral for Signal Conversion and Acquisition
- Supported by the MPLAB® Code Configurator (MCC) Touch Configurator Development Tools

32.2. Overview

The Peripheral Touch Controller (PTC) detects touch inputs on capacitive sensors. The external capacitive touch sensor is typically formed on a printed circuit board (PCB) or a transparent conductive substrate made of a transparent or translucent material, such as indium tin oxide (ITO) or poly (3,4-ethylenedioxythiophene) (PEDOT). An increasingly popular implementation is printing the sensor electrodes directly on the backside of the touch surface using conductive inks. The sensor electrodes are connected to the PTC through the I/O pins in the device. The PTC supports both mutual and self-capacitance sensors. Figure 32-1 shows some examples of the supported touch sensors .

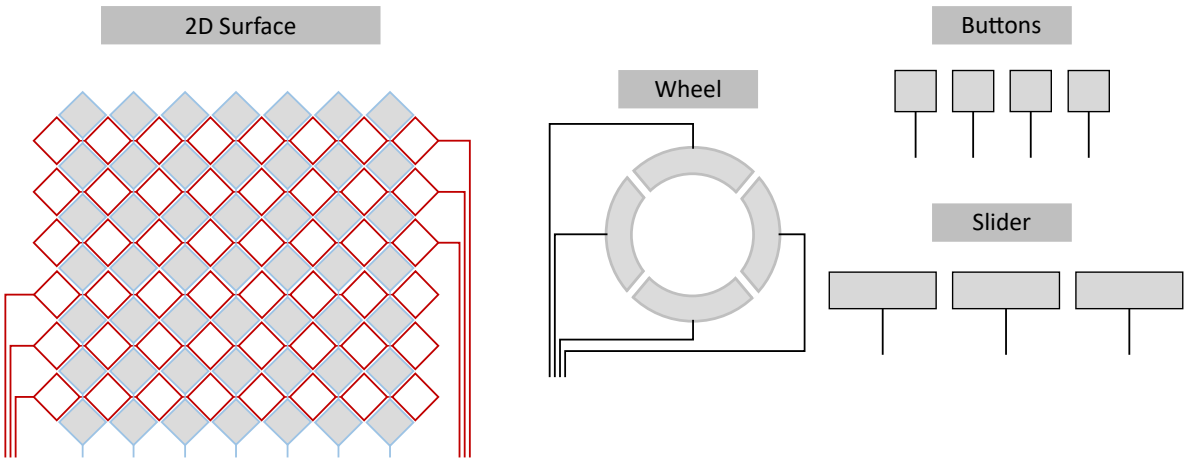
A self-capacitance sensor consists of a conductor plate with one wire connected directly to an I/O pad. In Self-Capacitance mode, the PTC requires one pin (Y-line) for each touch sensor. See the Figure 32-2.

A mutual capacitance sensor consists of two conductor plates, each connected to an I/O pad. Driving one plate, and sensing the other one measures the mutual capacitance between the two plates. In Mutual Capacitance mode, sensing is performed using capacitive touch matrices supporting various X-Y configurations. The PTC requires one pin per X-line and one pin per Y-line. See the Figure 32-3.

For a capacitive matrix array where only a single touch point is needed, self-capacitance sensing on both vertical and horizontal lines can effectively identify the touch point by locating the intersection of the touched lines. However, for multiple touch points, self-capacitance scanning alone is insufficient for accurately determining their exact positions. To precisely detect multiple touch points, mutual-capacitance sensing is required.

The number of available pins and the assignment of X- and Y-lines depend on both package type and device configuration.

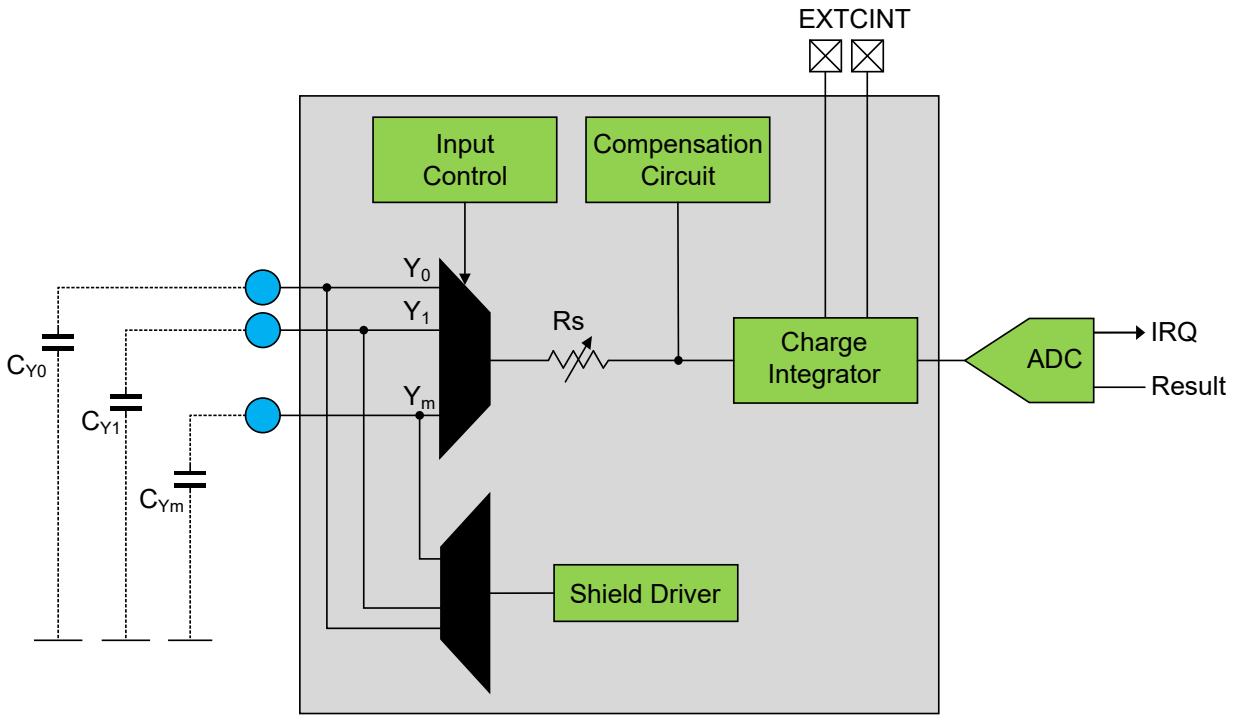
Figure 32-1. Example of Supported Touch Sensors



For more information about designing the touch sensor, refer to the AN2934, "Capacitive Touch Sensor Design Guide (DS00002934)".

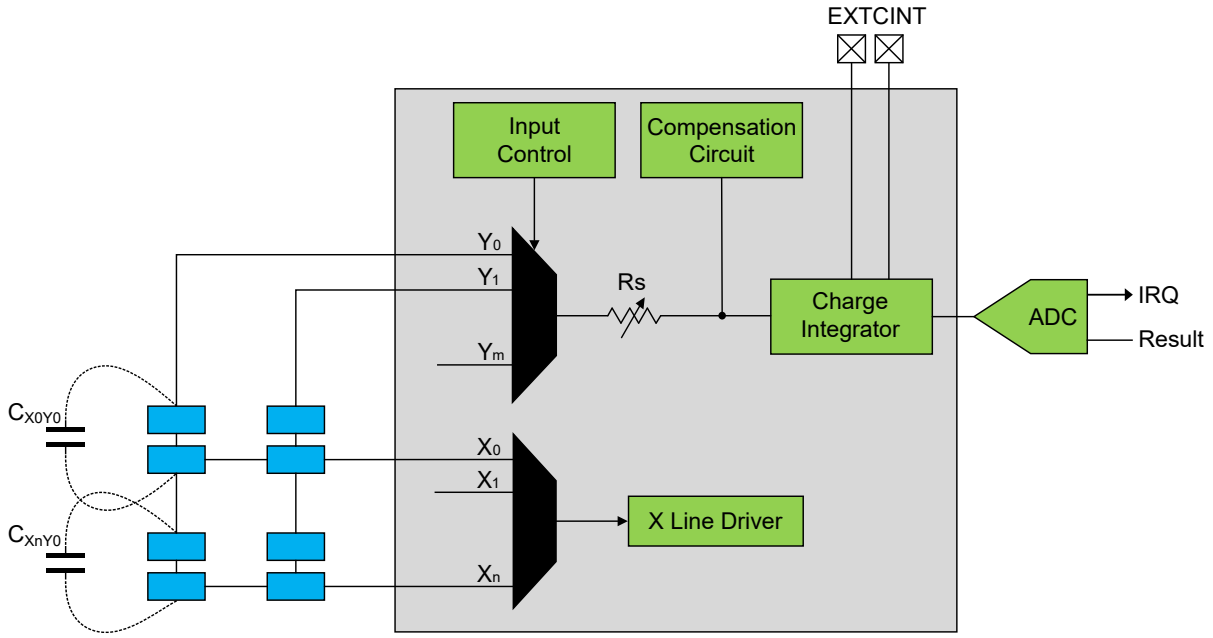
32.3. Block Diagram

Figure 32-2. PTC Block Diagram in Self-Capacitance Mode



Note: The internal series resistor, R_S, has a limited effect in Self-Capacitance mode. It is always recommended to add an external series resistor.

Figure 32-3. PTC Block Diagram in Mutual Capacitance Mode



32.4. Signal Description

Table 32-1. Signal Description

Name	Type	Description
Y[m:0]	Analog	Y-line (Input/Output)
X[n:0]	Digital	X-line (Output)
EXTCINT[1:0]	Analog	External Integration Capacitor (Input/Output)

Note: The EXTCINT pins are used to connect an external capacitor to the charge integrator when working with sensors that have high capacitance.

For available pins and functionalities, refer to the *I/O Multiplexing and Considerations* section.

32.5. System Dependencies

To use this peripheral, configure the other components of the system as described in the following sections.

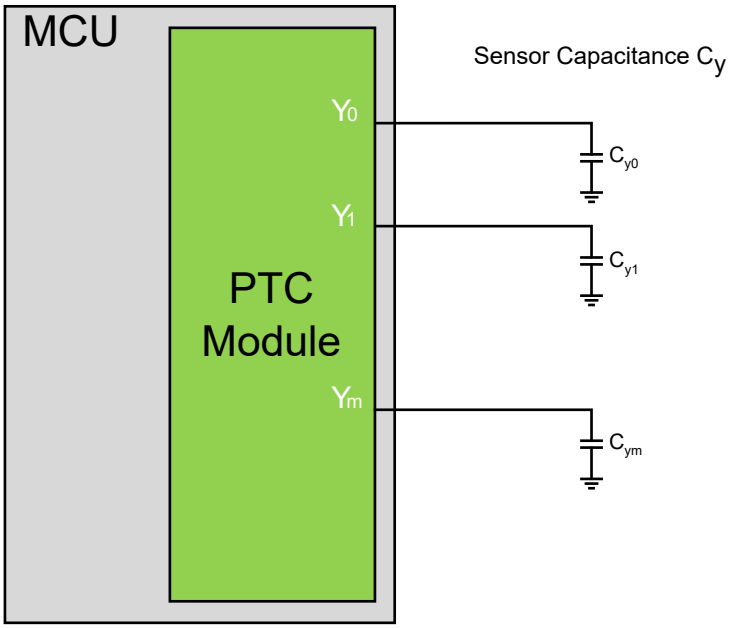
32.5.1. I/O Lines

The I/O lines used for analog X- and Y-lines must be connected to external capacitive touch sensor electrodes. External components are not necessary for normal operation. However, to reduce the effects of electromagnetic interference, add a series resistor of up to 100 kΩ to the Y-lines. In Mutual Capacitance mode, add a series resistor of up to 10 kΩ to the X-lines to reduce EMC emissions.

32.5.1.1. Self Capacitance Sensor Arrangement

A self-capacitance sensor is connected to a single pin on the PTC through the Y-electrode. The sense electrode capacitance is measured by the PTC.

Figure 32-4. Sensor Capacitance Sensor Arrangement

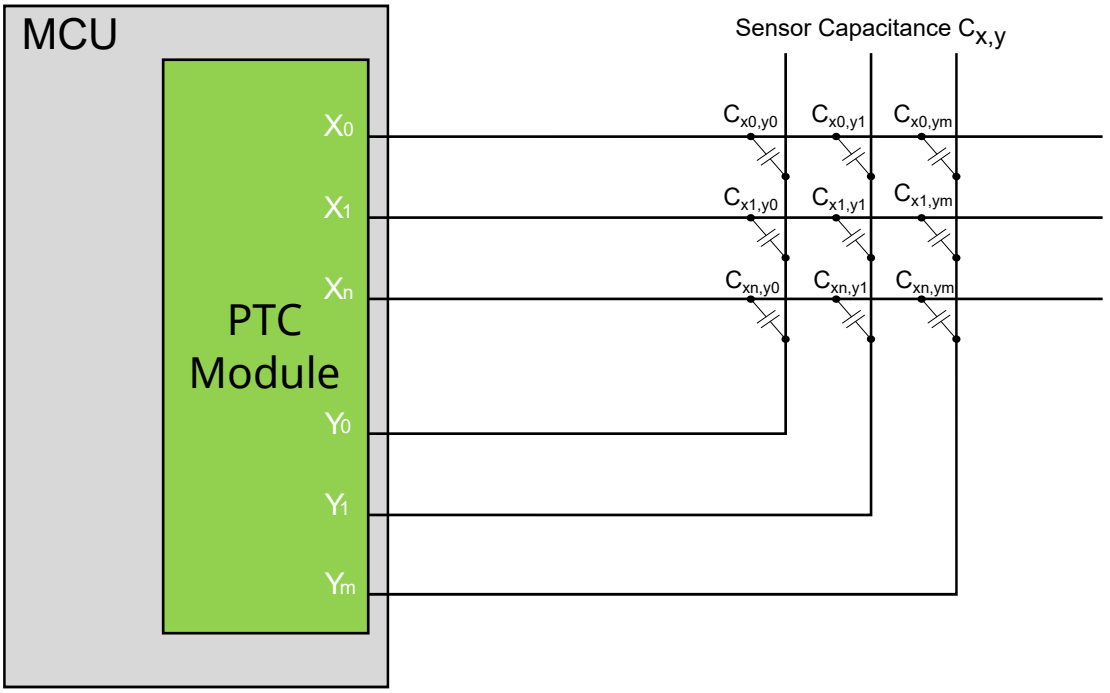


32.5.1.2. Mutual Capacitance Sensor Arrangement

A mutual capacitance sensor is formed between two I/O lines - an X-electrode for transmitting and a Y-electrode for sensing. The mutual capacitance between the X- and Y-electrodes is measured by the PTC.

A mutual capacitance sensor array is a good way to reduce the number of I/O lines needed to connect a large number of sensors. It is constructed by connecting the capacitor plates either horizontally or vertically, where horizontal and vertical lines do not physically connect, forming a mutual-capacitive sensor at each intersection.

Figure 32-5. Mutual Capacitance Sensor Arrangement



32.5.2. Clocks

The PTC is clocked by the CLK_PER clock signal. Refer to the Clock Controller (CLKCTRL) section for details on configuring CLK_PER. The PTC clock can run at the same speed as the peripheral clock or at a slower speed, depending on the prescaler settings. PTC clock is configured using MCC Touch Configurator.

32.5.3. Analog-to-Digital Converter (ADC)

The PTC utilizes the in-built 10-bit ADC for signal acquisition and conversion. The ADC must be enabled and configured appropriately to allow the correct behavior of the PTC, which is managed by the MCC Touch library. Refer to the *Analog-to-Digital Converter (ADC)* section for further details.

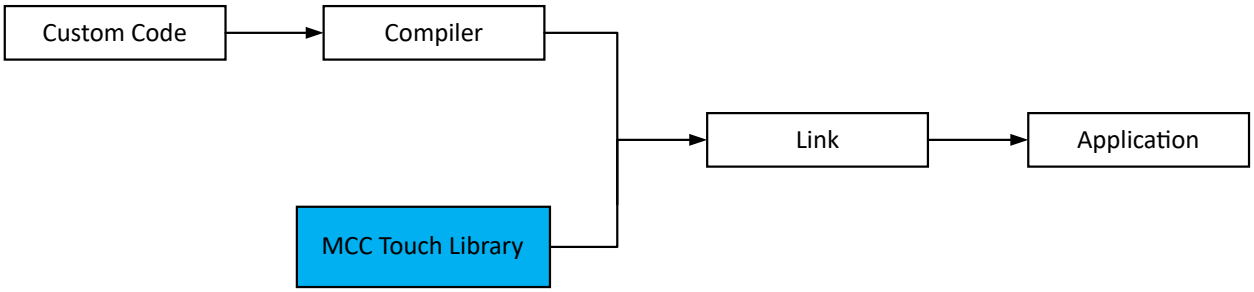
When the PTC is enabled, it takes control of ADC. The MCC Touch library provides APIs to facilitate multiplexing of the ADC between the PTC and non-touch applications if the application also requires the ADC for purposes other than touch sensing.

When the PTC is disabled, the ADC becomes available for normal operation.

32.6. Functional Description

To access the PTC, the *MCC Touch Configurator* must be used to configure the *Touch Library* and link it to the application software. The MCC Touch Library can be used to implement buttons, sliders and wheels in various combinations on a single interface.

Figure 32-6. MCC Touch Library Usage



For more information about MCC Touch Library, refer to the "QTouch® Modular Library Peripheral Touch Controller User's Guide (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 to enable the UPDI function, and one UPDI pin of the device for programming
  - 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
- Programming and Debug Interface Disable (PDID) Security Functionality
  - UPDI access can be limited by PDICFG and LOCK fuses
  - Bootloader still negotiating firmware updates

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 the 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 a dedicated UPDI pin for data reception and transmission. The clocking of the PHY layer is done by a dedicated internal oscillator.

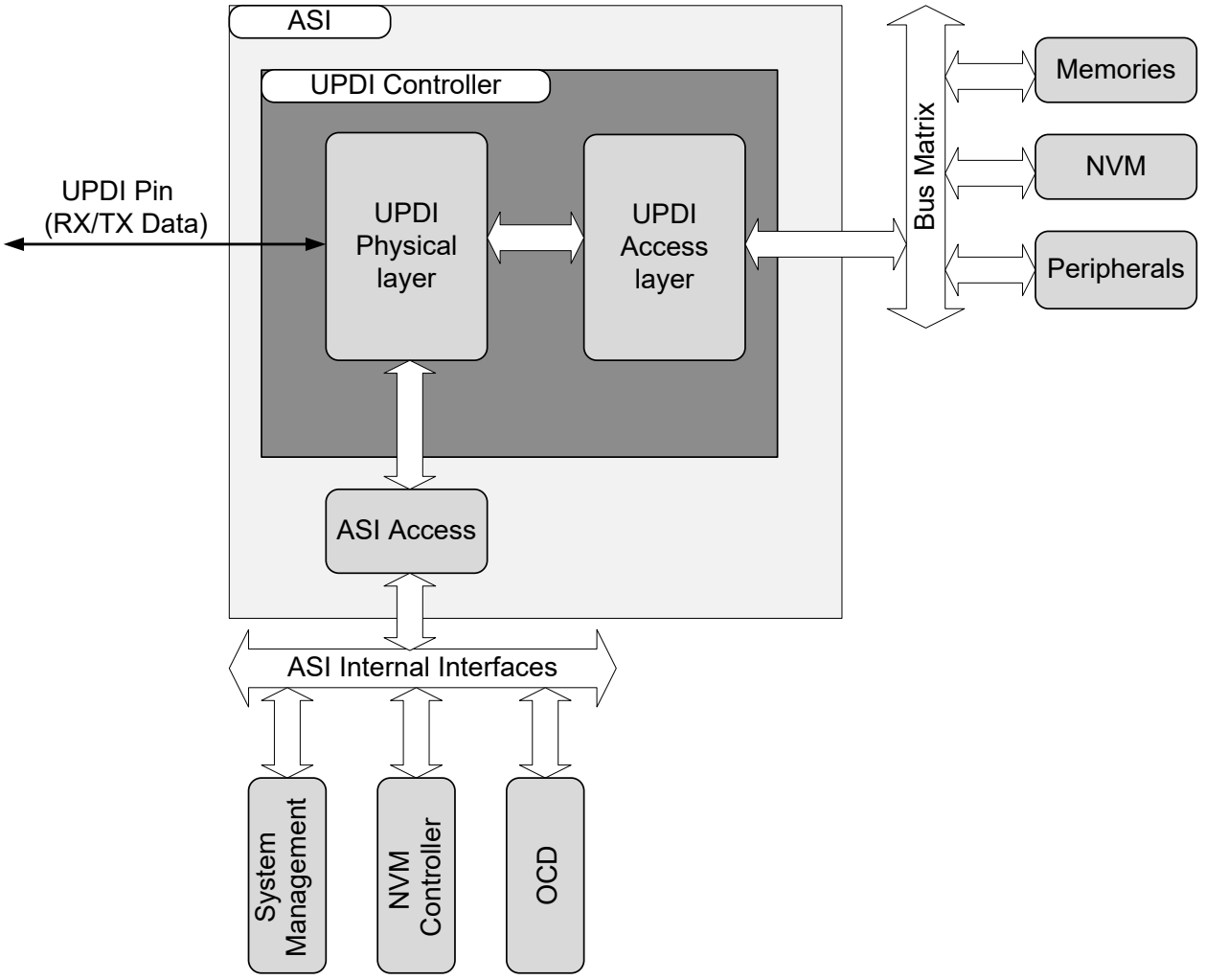
The ACC layer is the interface between the UPDI and the connected bus matrix. This layer grants access via the UPDI interface to the bus matrix with memory-mapped access to system blocks such as memories, NVM, and peripherals.

The Asynchronous System Interface (ASI) provides direct interface access to select features in the OCD, NVM, and System Management systems, which gives the debugger direct access to system information without requesting bus access.

The UPDI access can be limited by PDICFG and LOCK fuses, while the bootloader can still negotiate firmware updates.

33.2.1. Block Diagram

Figure 33-1. UPDI Block Diagram

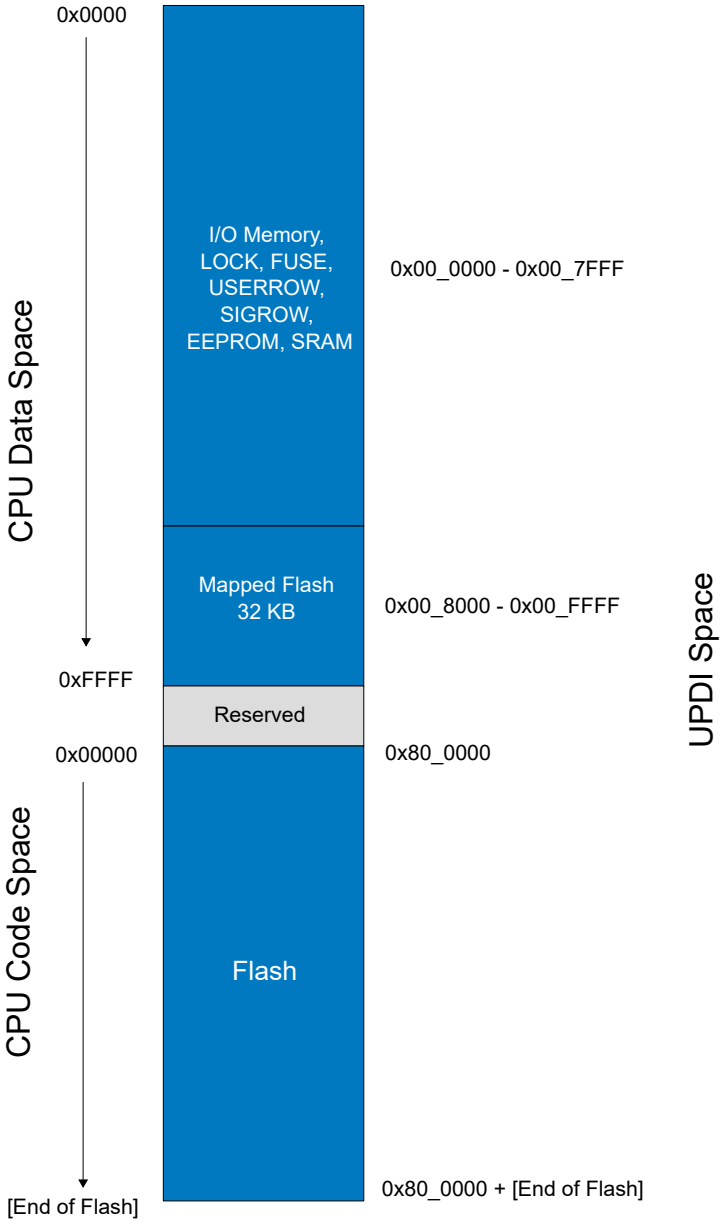


33.2.2. Addressing the Program Memory Space

In the CPU data space, the I/O memory, the fuses, EEPROM and SRAM are located at addresses from 0x0000 to 0x7FFF. In addition, a section of the Flash memory (up to 32 KB) can be mapped into the addresses from 0x8000 to 0xFFFF. These addresses (0x0000 - 0xFFFF) are also valid for access by the UPDI peripheral.

The CPU code space, i.e., the *entire* Flash memory, can be accessed by the CPU using the `LPM/SPM` instructions, starting at the relative address 0x0000. For access by UPDI, the CPU data space and the CPU code space are virtually one continuous address space, and the code space always starts at the offset address 0x80_0000.

Figure 33-2. Memory Map, As Seen From The UPDI



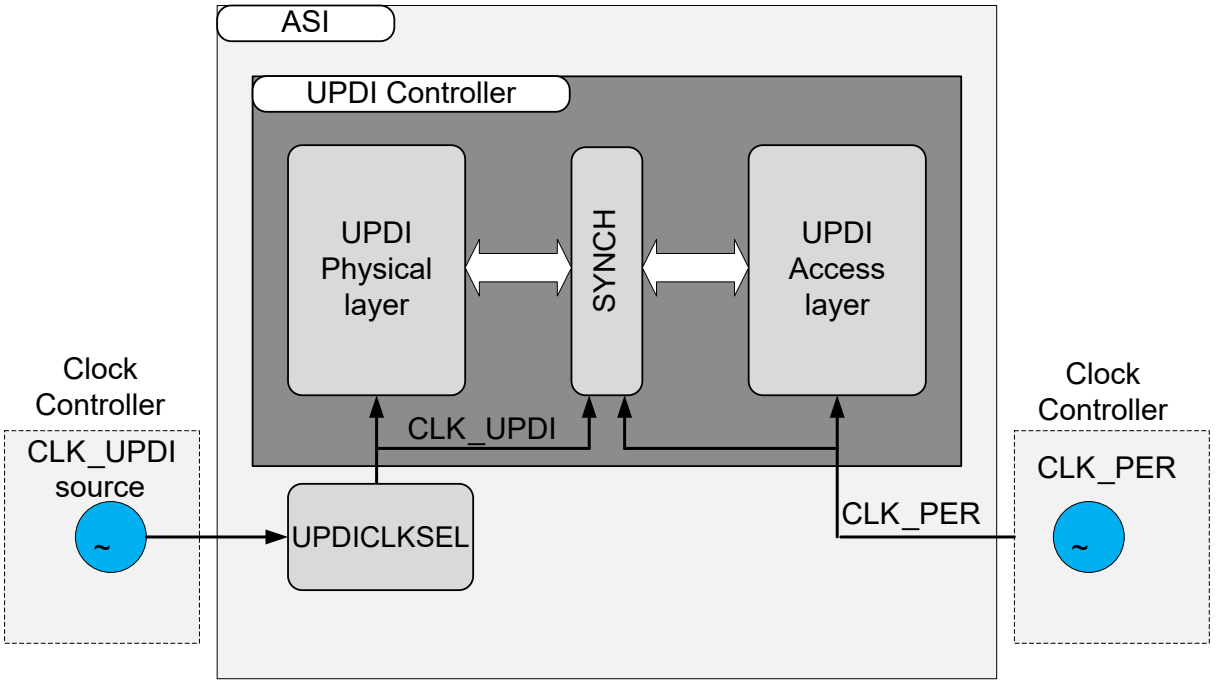
See the *Memories* sections for more details and exact addresses of the memory areas in a given device.

33.2.3. 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 (UPDICLKSEL) bit field in the ASI Control A (UPDI.ASI_CTRLA) register.

Figure 33-3. UPDI Clock Domains



33.2.4. 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

33.2.5. I/O Lines and Connections

The UPDI uses the UPDI pin for half-duplex UART communication.

The $\overline{\text{RESET}}$ pin can be used to invoke and maintain UPDI operation.

The $\overline{\text{RESET}}$ pin can be used as $\overline{\text{RESET}}$ or input, depending on the configuration of the RSTPINCFG bit in the Pin Configuration (FUSE.PINCFG) fuse. The UPDI pin can be used as GPIO or for UPDI function, depending on the configuration of the UPDIPINCFG bit in the Pin Configuration (FUSE.PINCFG) fuse. The high voltage applied to the $\overline{\text{RESET}}$ pin can be used to invoke and maintain UPDI operation, thus overriding both pins configuration in the Pin Configuration (FUSE.PINCFG) fuse.

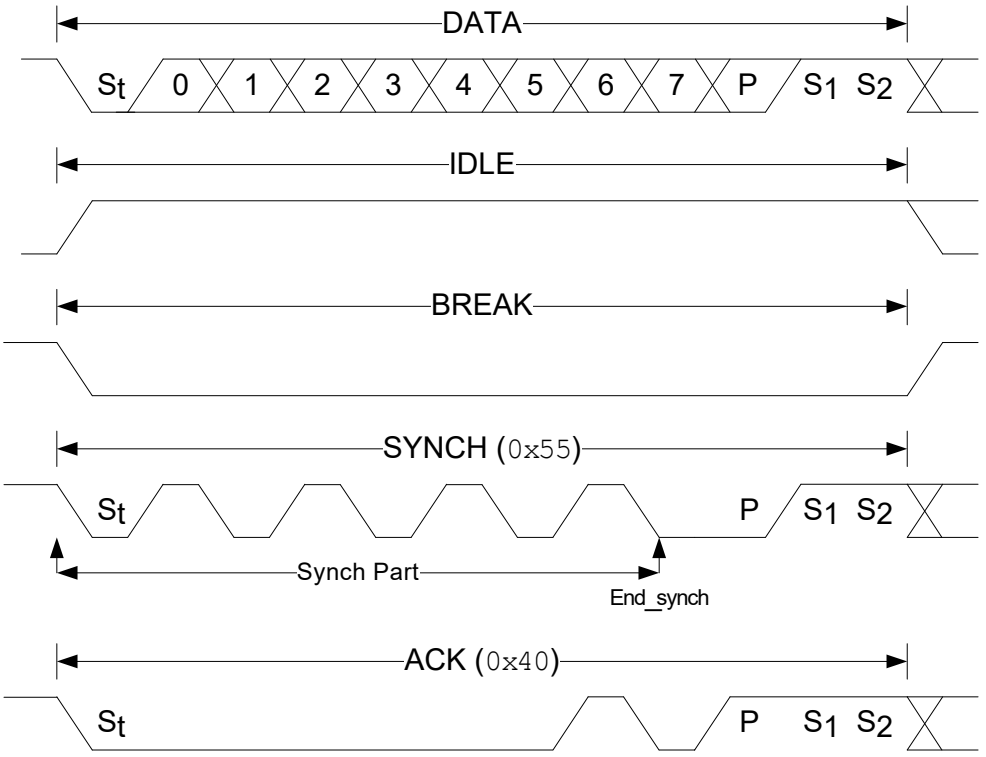
Refer to the UPDI Enabling section for details about the required and optional pin configurations.

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 33-4. 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	IDLE is a specific frame that consists of at least 12 high bits, which is the same as keeping the transmission line in an Idle state
BREAK	BREAK is a specific frame that consists of at least 12 low bits. It is used to reset the UPDI back to its default state and is typically used for error recovery.
SYNCH	The Baud Rate Generator uses the SYNCH frame 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 `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 debugger/programmer side. 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 *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 (UPDICLKSEL) 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. It is recommended that the clock frequency does not run faster than the required frequency for the desired baud rate. The default UPDICLKSEL 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.

Table 33-1. Recommended UART Baud Rate Based on UPDICLKSEL Setting

UPDICLKSEL[1:0]	Max. Recommended Baud Rate	Min. Recommended Baud Rate
0x0 (32 MHz)	1.8 Mbps	0.600 kbps
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 Table 33-2.

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 an 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, which changes the baud rate range of the UPDI, according to the *Recommended UART Baud Rate Based on UPDICLKSEL Setting* table above.

33.3.1.2.1. BREAK in One-Wire Mode

In One-Wire 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
$65535 \times 250\ \text{ns} = 16.4\ \text{ms/byte} = 16.4\ \text{ms}/8\ \text{bits} = 2.05\ \text{ms/bit}.$

This gives a worst-case BREAK frame duration of $2.05\ \text{ms} \times 12\text{bits} \approx 24.6\ \text{ms}$ 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 Table 33-3.

Table 33-3. Recommended BREAK Character Duration

UPDICLKSEL[1:0]	Recommended BREAK Character Duration
0x1 (16 MHz)	6.15 ms
0x2 (8 MHz)	12.30 ms
0x3 (4 MHz)	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 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.

33.3.1.4. Special Considerations for Programming

Any reset source will also reset the UPDI on a locked device. The reset may be triggered during boot of the device, which results in a repeating reset loop. Use a UPDI handshake mechanism to access a device in this state.

Important: Both the UPDI pin and the $\overline{\text{RESET}}$ pin must be connected to the programmer/debugger to use the UPDI handshake mechanism.

The UPDI handshake procedure:

1. Drive the UPDI line low and hold it low.
2. Pull the $\overline{\text{RESET}}$ line low for the minimum reset pulse width and release it.
3. Wait for at least 40 ms so the device can reach an initial state.
4. Release the UPDI line to its UART IDLE state.
5. Issue a negative pulse to the UPDI line.
6. Send a CHIP ERASE key to unlock the device and poll for successful completion.
7. Toggle power on the device.

Important: Steps 4 to 6 must be completed within a 64 ms time window.

NOTICE After performing the chip erase, the lock bits on the device will be cleared. However, any reset source will continue to reset the UPDI until the next Power-On Reset happens.

33.3.2. Operation

The UPDI must be enabled before the UART communication can start.

However, PDICFG and LOCK fuses can limit the UPDI access, while the bootloader can still negotiate firmware updates.

33.3.2.1. UPDI Enabling

Depending on how the application has configured the UPDI pin, one of the following two methods can be used to enable the UPDI:

- **One-Wire Enable** - This method requires the UPDI pin to be configured in UPDI function mode by setting the UPDIPINCFG bit in the Pin Configuration (FUSE.PINCFG) register to '1'.
- **HV Override of UPDI Pin** - This method is used if the UPDI pin is configured in GPIO mode (the UPDIPINCFG bit in Pin Configuration (FUSE.PINCFG) is set to '0'). Applying an HV pulse on $\overline{\text{RESET}}$ will reconfigure the UPDI pin to UPDI function mode, thereby overriding the configuration in the Pin Configuration (FUSE.PINCFG) fuse.

33.3.2.1.1. One-Wire Enable

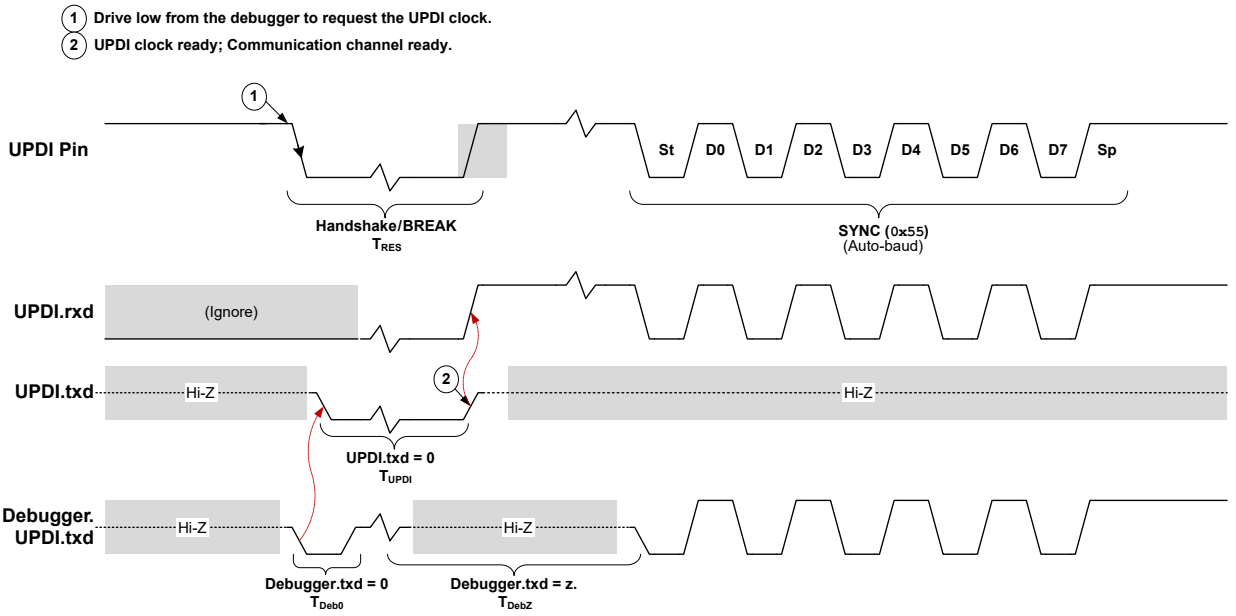
The UPDI pin has a constant pull-up when enabled. As a prerequisite, the UPDI pin must be configured in UPDI function mode, either by setting the UPDIPINCFG bit in Pin Configuration (FUSE.PINCFG) fuse to '1' or by applying an HV pulse on the $\overline{\text{RESET}}$ pin, thus overriding the configuration of UPDIPINCFG.

Follow this sequence to enable the UPDI:

1. Drive the UPDI pin low for more than 200 ns, and release it.
   By this, a connected programmer will initiate the start-up sequence:
   - An edge detector starts driving the UPDI pin low, so when the programmer releases the line, it will stay low
   - The UPDI clock is started. The UPDI will continue to drive the line low until the clock is stable and ready for the UPDI to use
     The expected arrival time for the clock will depend on the oscillator implementation regarding the accuracy, overshoot, and readout of the oscillator calibration
   - The data line will be released by the UPDI and pulled high when the oscillator is ready and stable
2. Poll the UPDI pin to detect when the pin transitions to high again.
   This transition indicates that the edge detector has released the pin (pull-up), and the UPDI can receive a SYNCH character.
3. Send a SYNCH character `0x55`.
   After a successful SYNCH character transmission, the first instruction frame can be transmitted.
4. Send the NVMPROG key using the `KEY` instruction.
   Sending this key clears the lock bits, and the Programming Start (PROGSTART) bit in the ASI_SYS_STATUS is set. The device is now prepared for programming.
5. 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.
   Disabling the UPDI and hence, the accompanying clock request, will reduce power consumption.

The timing of the enable sequence is shown in Figure 33-5, 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.

Figure 33-5. UPDI Enable Sequence



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.

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. If the timing is violated, the UPDI is disabled to avoid unintentional enabling of the UPDI. See Disable During Start-Up for more details.

Note: The actual values for T_{RES}, T_{UPDI}, T_{Deb0}, and T_{DebZ} can be found in the *Electrical Characteristics* section.

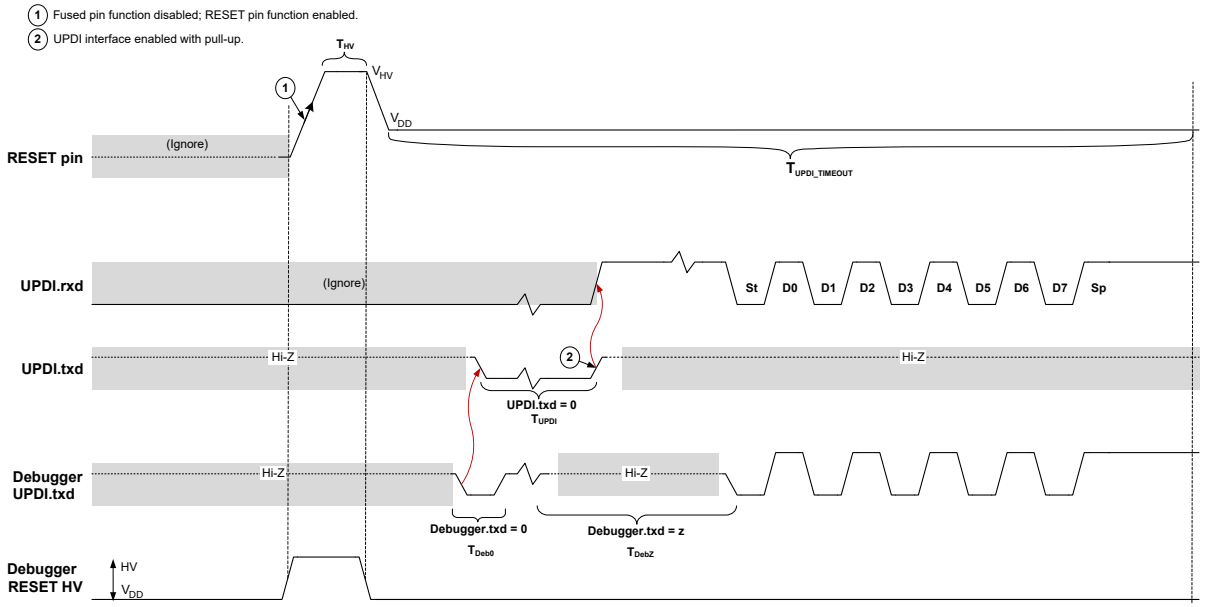
33.3.2.1.2. UPDI Enable with High-Voltage Override of UPDI Pin

An application can configure the UPDI pin as an I/O pin. In this case, the regular one-wire enable sequence cannot be used.

An HV pulse applied to the $\overline{\text{RESET}}$ pin will switch the pin functionality of the UPDI pin to the UPDI function.

1. Apply the HV signal, as described in Figure 33-6 and the *Electrical Characteristics* section. The HV must be applied after the POR has been released.
   This will override the pin configuration of the UPDI pin. The HV detection circuitry will trigger a device Reset. The CPU will remain halted until the reception of a valid UPDI key, or the expiration of $T_{UPDI\_TIMEOUT}$. If no such key is received, the device will be reset and the UPDI pin will have the configuration specified by the fuses.
2. Follow the regular one-wire enable sequence as described in One-Wire Enable. A valid UPDI key must be sent before $T_{UPDI\_TIMEOUT}$.
3. When the UPDI is enabled by an HV pulse, only a POR will disable the override of the UPDI pin and restore the settings as configured by the fuses.

Figure 33-6. UPDI Enable Sequence by High-Voltage (HV) Programming



Notes:

1. 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.
2. The actual threshold voltage for the UPDI HV activation depends on V_{DD}. See the *Electrical Characteristics* section for more details.
3. See the *Electrical Characteristics* section for the value of $T_{UPDI\_TIMEOUT}$.

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, which 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.

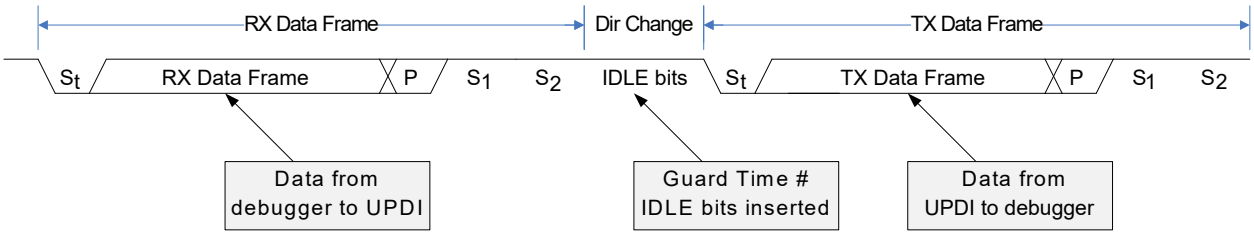
Whenever the UPDI detects an error, it will immediately enter an internal Error state to avoid unwanted system communication. In the Error state, the UPDI will ignore all incoming data requests, except when a BREAK character is received. The following procedure must always be applied when recovering from an Error condition.

1. Send a BREAK character. See the *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 the 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. When the synchronization time plus the data bus accessing time is longer than the guard time, the Idle time before a transmission will be more than the expected guard time.

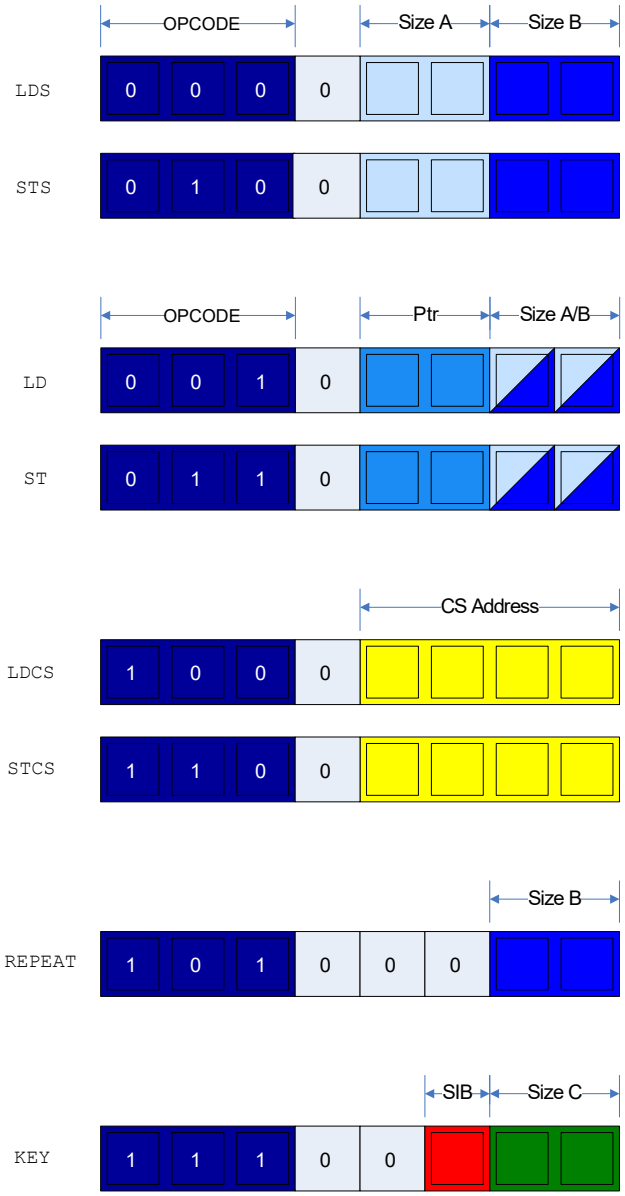
It is recommended to always use the insertion of a minimum of two Guard Time bits on the UPDI side and one guard time cycle insertion from the debugger side.

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 the *UPDI UART* section for information about setting the baud rate for the transmission. Figure 33-8 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	1 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	*(ptr--)

Size B - Data Size		
0	0	1 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

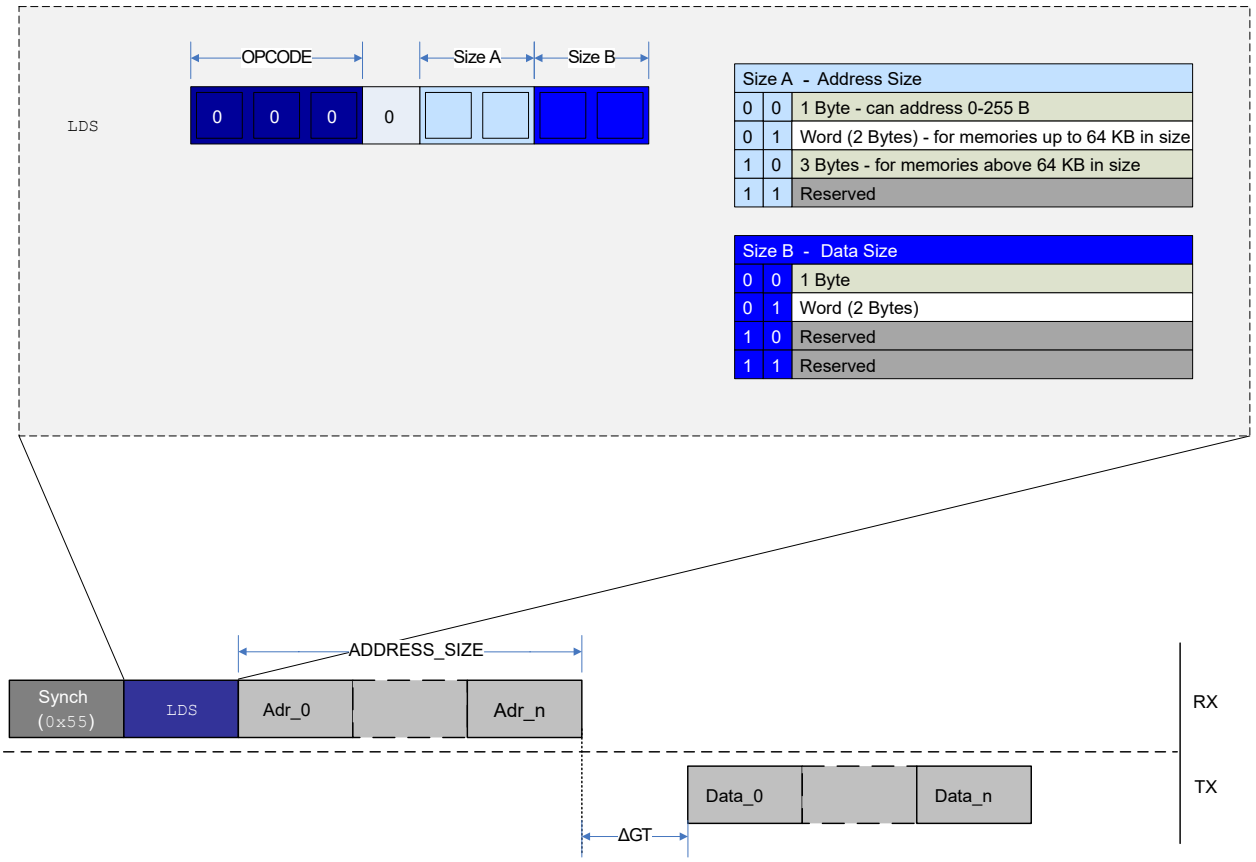
SIB - System Information Block Sel.	
0	Receive KEY
1	Send SIB

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 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



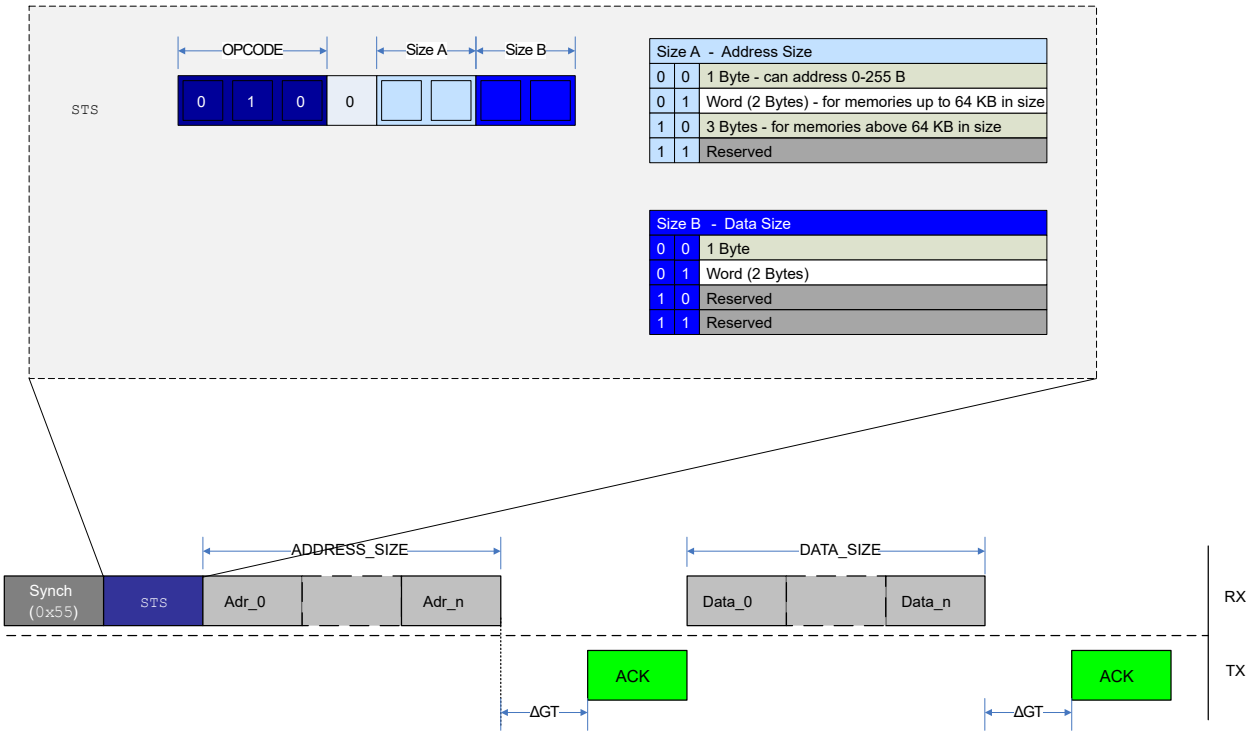
When the instruction is decoded and the address byte(s) are received as dictated by the decoded instruction, the DL layer will synchronize all required information to the ACC layer. This will handle the bus request and synchronize data buffered from the bus back to the DL layer, which will create a synchronization 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 is given by the size fields presented in Figure 33-10. 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 Figure 33-10, 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

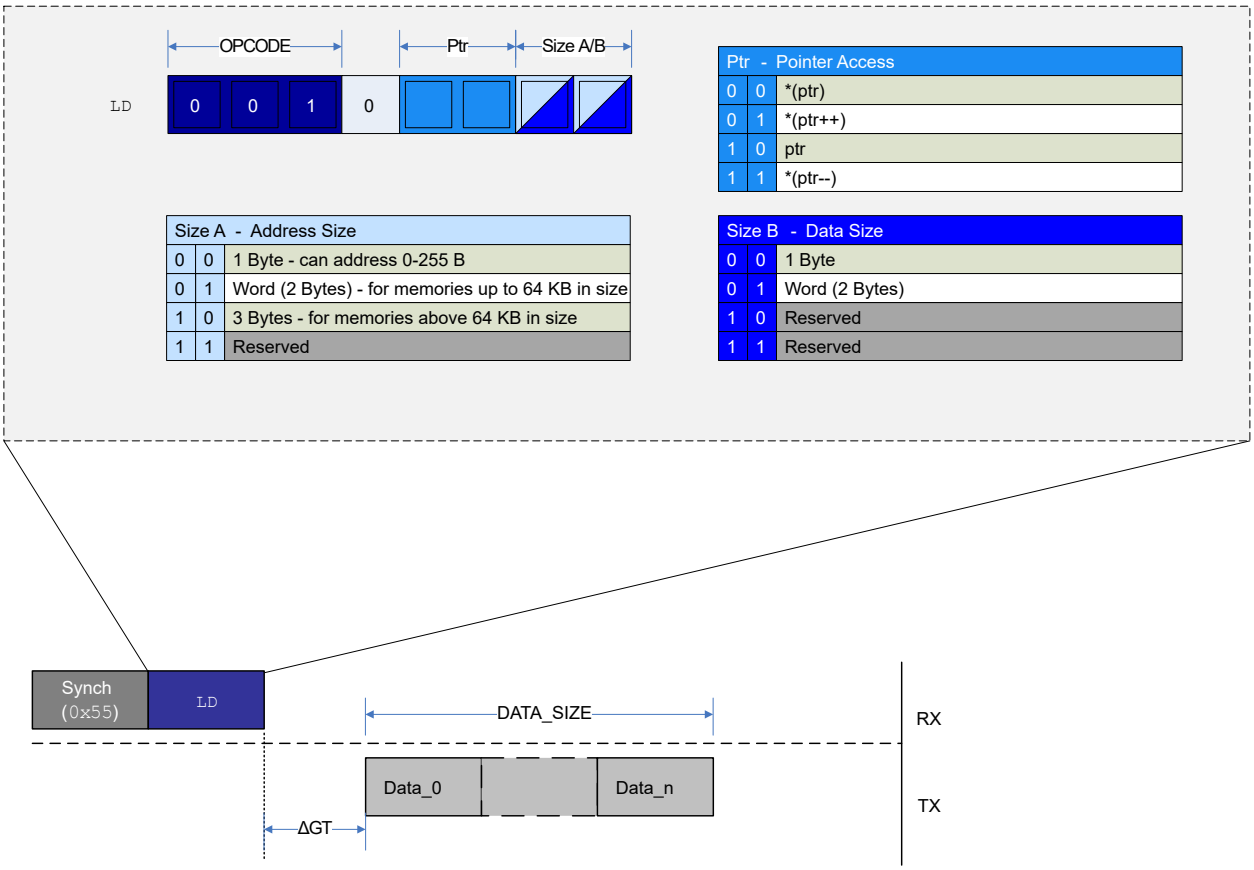


Figure 33-11 shows an example of a typical LD sequence, where the data are received after the Guard Time (GT) period. Loading data from the UPDI Pointer register follows the same transmission protocol.

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

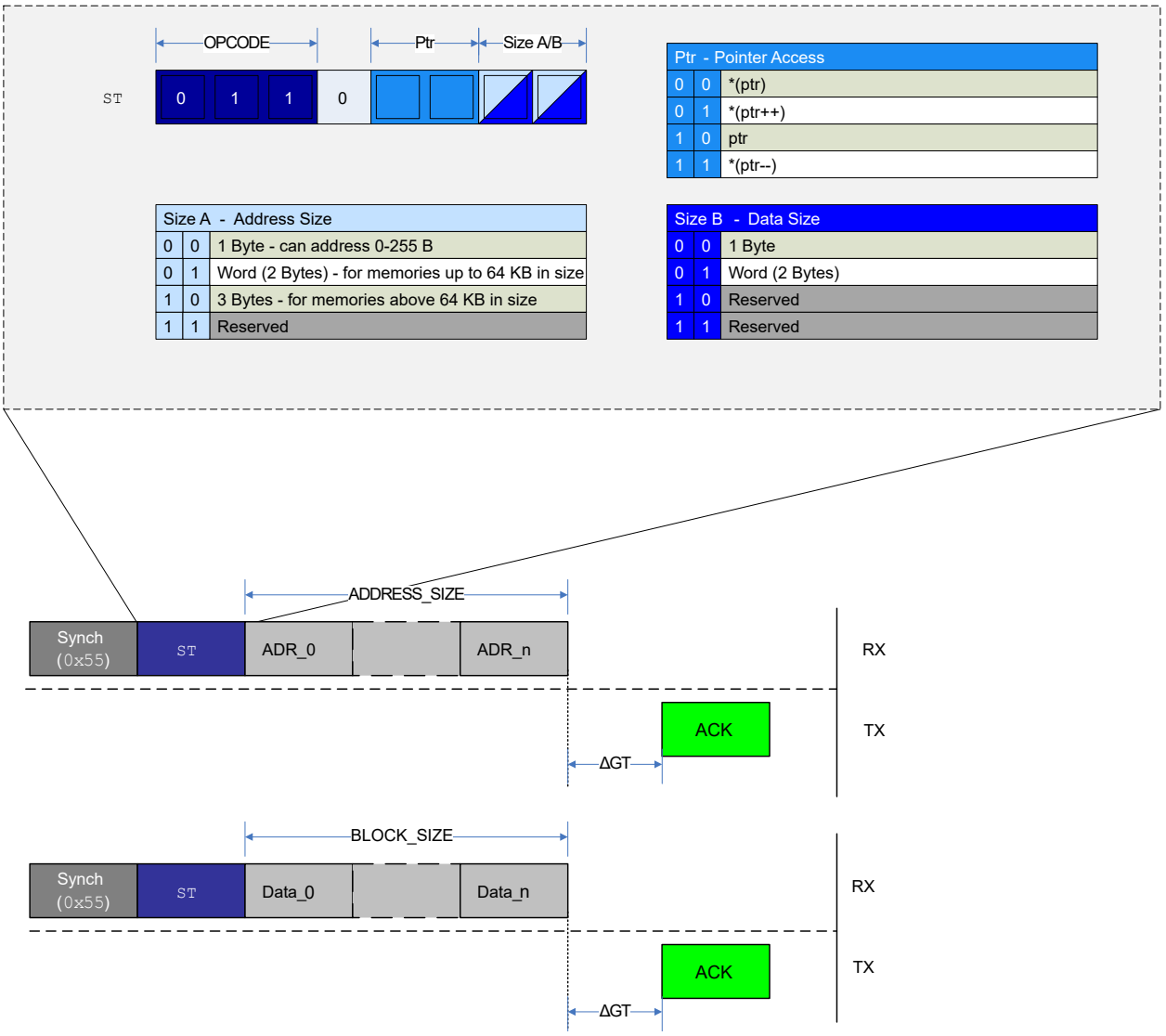


Figure 33-12 gives an example of an ST instruction to the UPDI Pointer register and the storage of regular data. A SYNCH character is sent before each instruction. In both cases, an Acknowledge (ACK) is sent back by the UPDI if the ST instruction was successful.

The next procedure has to be followed to write the UPDI Pointer register:

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

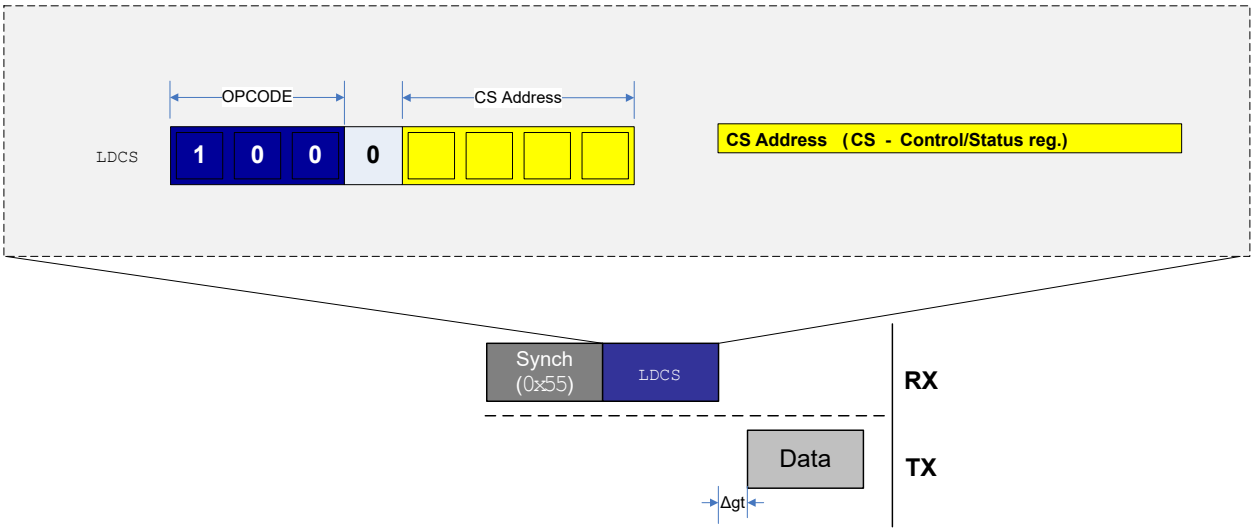


Figure 33-13 shows a typical example of LDCS data transmission. A data byte from the LDCS is transmitted from the UPDI after the guard time is completed.

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 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

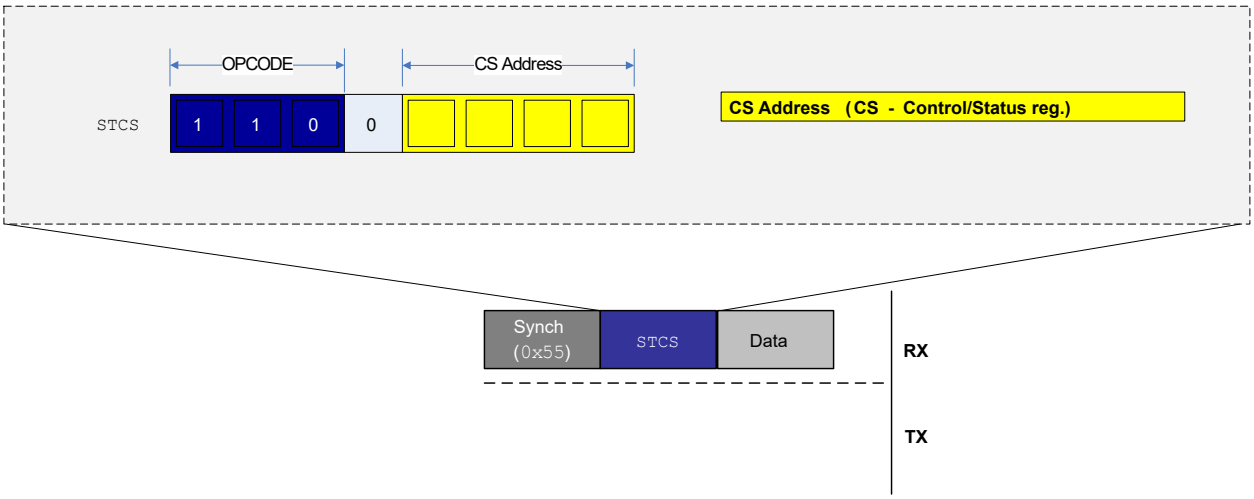


Figure 33-14 shows the data frame transmitted after the SYNCH character and the instruction frames. The STCS instruction byte can immediately be followed by the data byte. There is no response generated from the STCS instruction, as is the case for the ST and STS instructions.

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 for 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

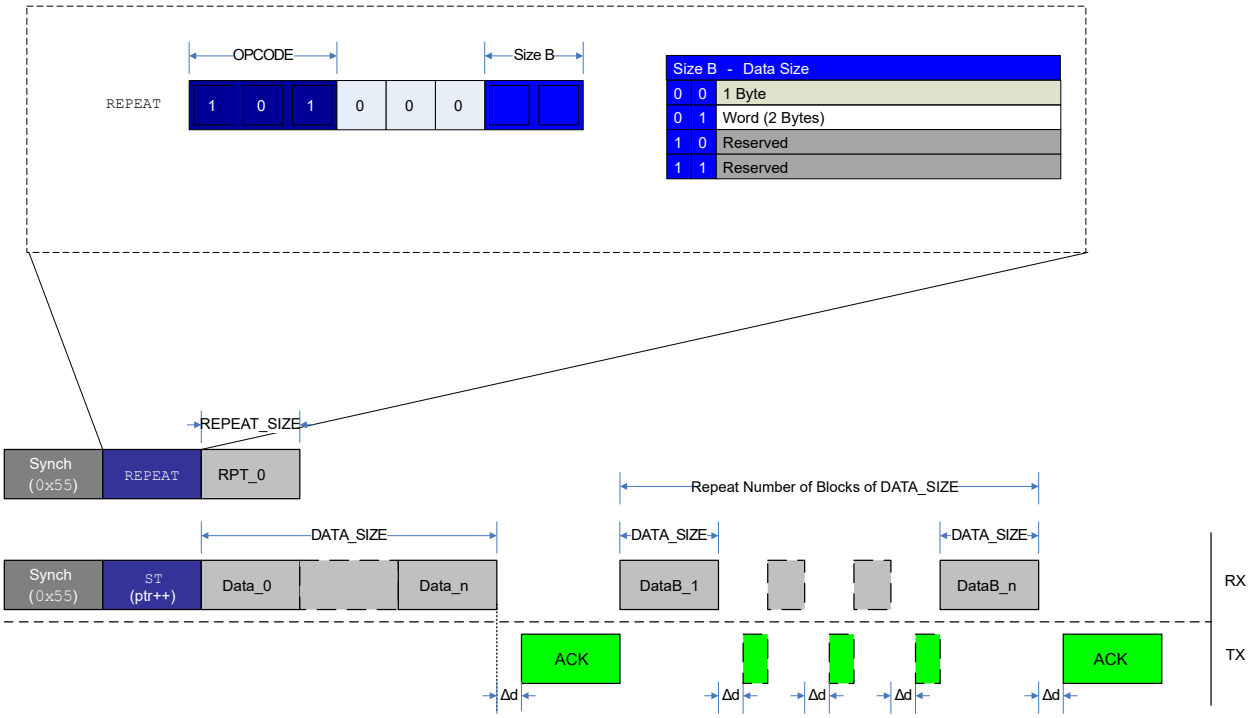
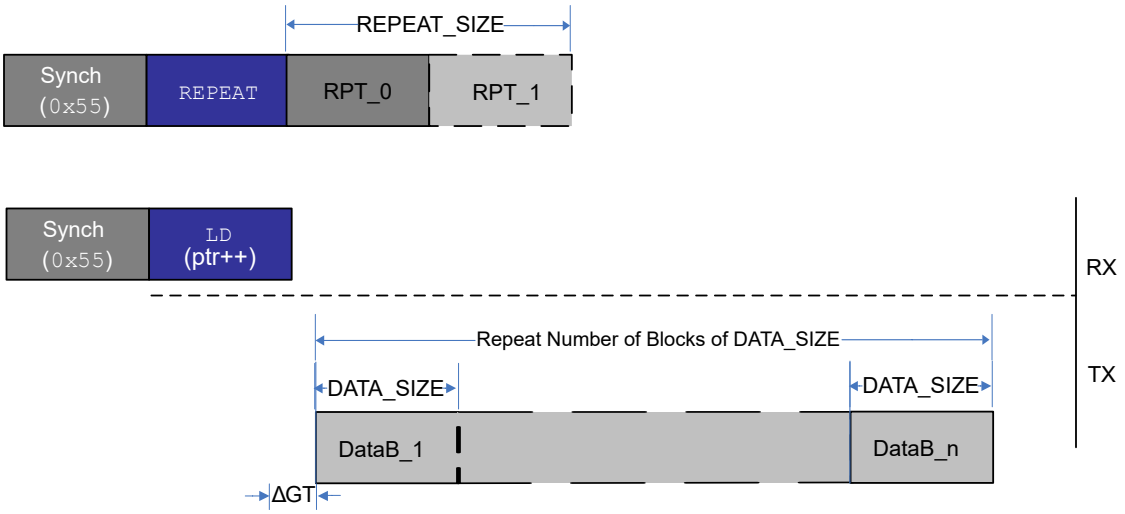


Figure 33-15 gives an example of a repeat operation with an ST instruction using pointer post-increment operation. After the REPEAT instruction is sent with RPT_0 = *n*, the first ST instruction is issued with SYNCH and instruction frame. The next *n* ST instructions are executed by only sending data 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 the *Key Activation Overview* table in the *Enabling of Key Protected Interfaces* section 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

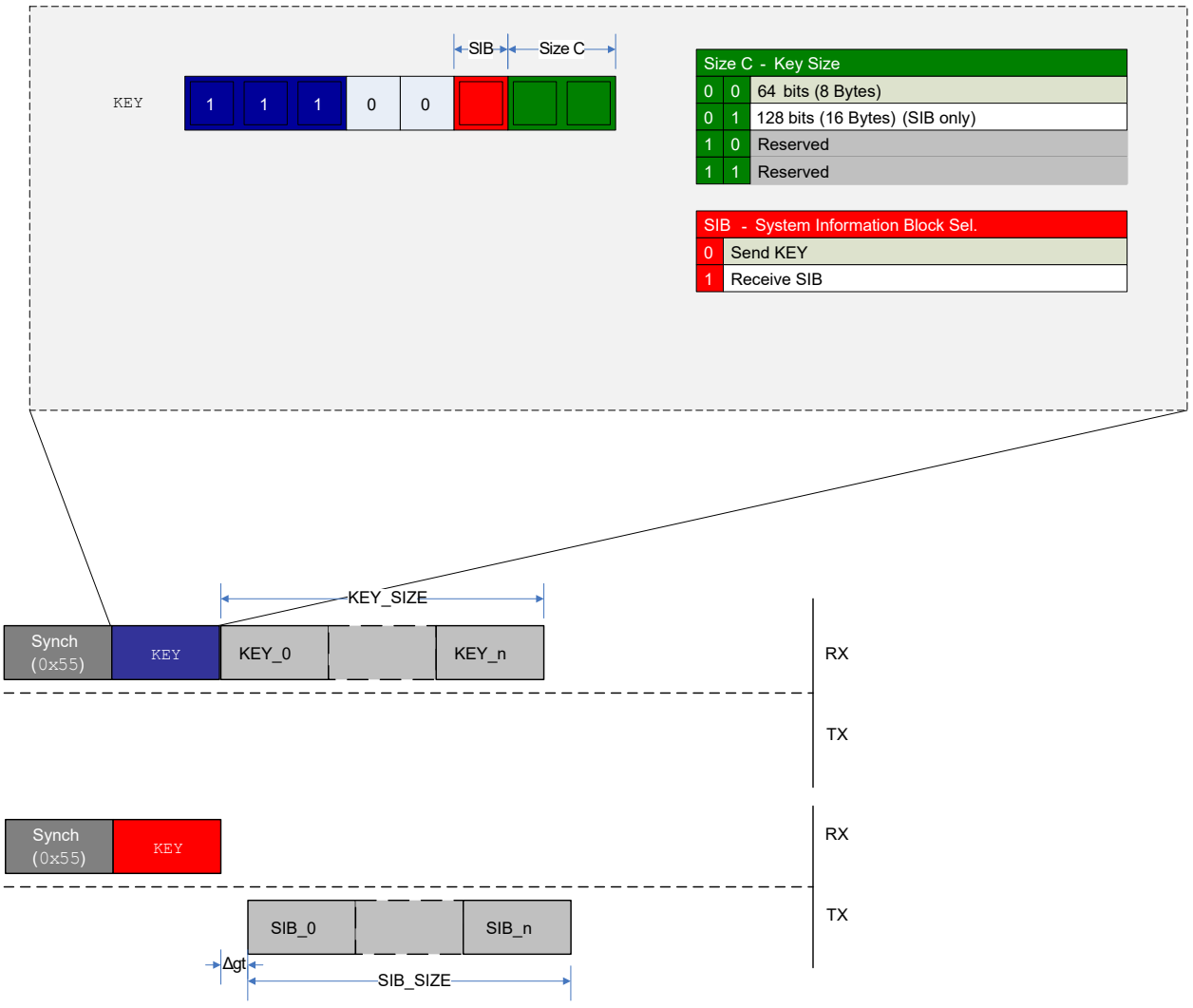


Figure 33-17 shows the transmission of a key and the reception of a SIB. In both cases, the Size C (SIZE_C) field in the operand determines the number of frames being sent or received. There is no response after sending a KEY to the UPDI. When requesting the SIB, data will be transmitted from the UPDI 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

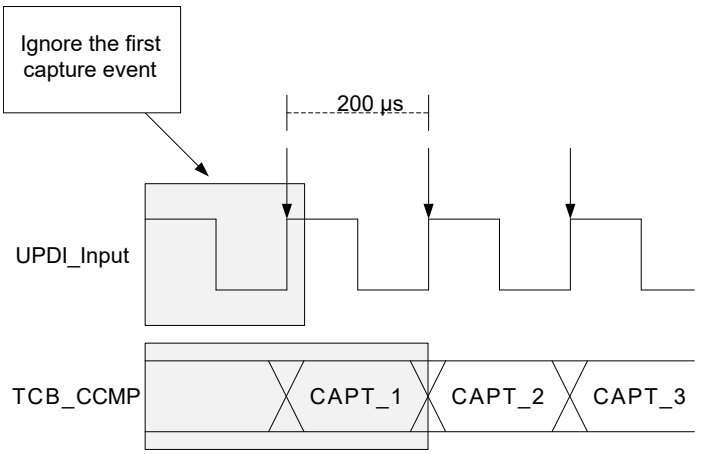
from the ASI_CRC_STATUS register. Refer to the *CRCSCAN - Cyclic Redundancy Check Memory Scan* section in the device data sheet for more information on this feature.

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 of 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 must be ignored. The second captured value based on the input event must be used for the measurement. See Figure 33-18 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. For more details, refer to the *TCB - 16-bit Timer/Counter Type B* section.

Figure 33-18. UPDI System Clock Measurement Events



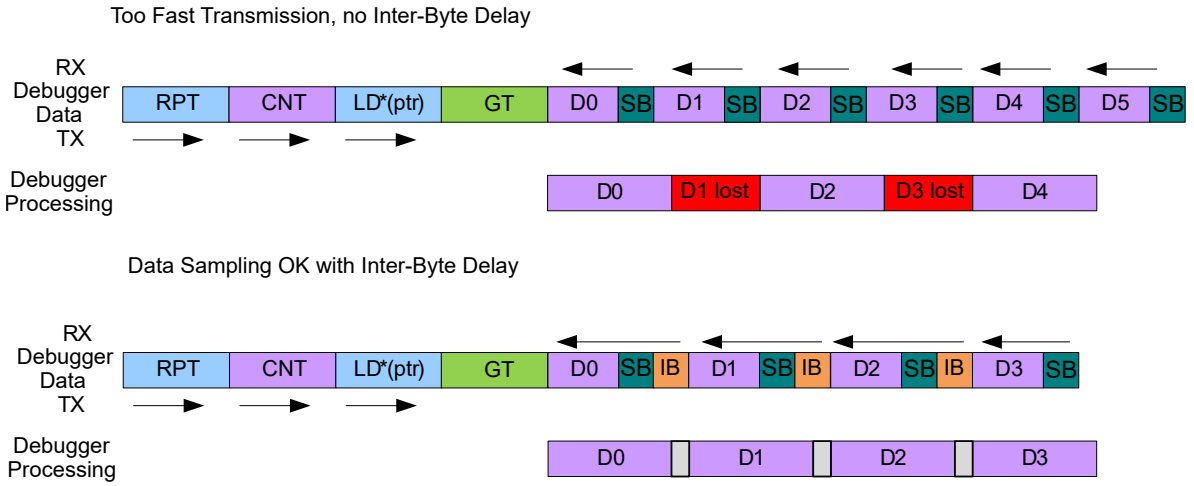
33.3.6. Inter-Byte Delay

When performing a multibyte 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, the data might come out too fast on the receiver side, 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 multibyte 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`



Notes:

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 the *KEY - Set Activation Key or Send System Information Block* section. 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 Figure 33-20 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 KEY - Set Activation Key or Send System Information Block. Table 33-4 describes the available keys and the condition required for starting the operation after the key has been loaded.

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.PROGSTART set.	Programming done/UPDI Reset
USERROW-Write	Program the user row on the locked device	ASI_SYS_STATUS.UROWSTART set	Write to key Status bit/UPDI Reset

Table 33-5 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 next steps must be followed to issue a chip erase:

1. Enter the Chip Erase key by using the `KEY` instruction. See the *Key Activation Signatures* table for the CHIPERASE signature.
2. Enter the NVM Programming key by using the `KEY` instruction. See the *Key Activation Signatures* table for the NVMPROG signature. This will prevent a freshly erased device from failing the CRC (if activated).
3. Read the ASI Key Status (UPDI.ASI_KEY_STATUS) register to verify that both the Chip Erase Key Status (CHER) and the NVM Programming Key Status (NVMPROG) bits are set.
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 Lock Status (LOCKSTATUS) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
7. The chip erase is done when the LOCKSTATUS bit is '0'. If the LOCKSTATUS bit is '1', return to step 5.
8. Check the Chip Erase Key Failed (ERASEFAIL) bit in the ASI System Status (UPDI.ASI_SYS_STATUS) register to verify if the chip erase was successful.
9. If the ERASEFAIL 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 - Brown-out Detector* 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 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 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 Start NVM Programming (PROGSTART) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
7. NVM programming can start when the PROGSTART bit is '1'. If the PROGSTART 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 next 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 (UROWWR) 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 (UROWSTART) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.
6. The user row programming can start when the UROWSTART bit is '1'. If UROWSTART 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 the 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 (UROWSTART) bit from the ASI System Status (UPDI.ASI_SYS_STATUS) register.

10. The user row programming is completed when the UROWSTART bit is '0'. If the UROWSTART bit is '1', return to step 9.
11. Write to the User Row Write Key Status (UROWWR) 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 *EVSYS - 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 cannot 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's 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			UROWWR	NVMPROG	CHER			
0x08	ASI_RESET_REQ	7:0	RSTREQ[7:0]							
0x09	ASI_CTRLA	7:0							UPDICLKSEL[1:0]	
0x0A	ASI_SYS_CTRLA	7:0							UROWDONE	CLKREQ
0x0B	ASI_SYS_STATUS	7:0	BDEF	ERASEFAIL	SYSRST	INSLEEP	PROGSTART	UROWSTART	BOOTDONE	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	1	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 multibyte 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 and 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, a loss of data may occur.

Bits 2:0 – GTVAL[2:0] Guard Time Value
This bit field selects the guard time value 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 contention detection. Writing a '0' to this bit enables 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
			UROWWR	NVMPROG	CHER			
Access			R/W	R	R			
Reset			0	0	0			

Bit 5 – UROWWR 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 PROGSTART bit in ASI_SYS_STATUS is set.

Bit 3 – CHER 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 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	-	Reserved

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 Divider Select
Writing these bits selects the UPDI clock output frequency. The default setting after Reset and enable is 4 MHz. See the *Electrical Characteristics* section for more information on possible UPDI oscillator frequencies.

Value	Description
0x0	32 MHz UPDI clock
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
Property: -

Bit	7	6	5	4	3	2	1	0
							UROWDONE	CLKREQ
Access	-	-	-	-	-	-	R/W	R/W
Reset							0	0

Bit 1 – UROWDONE User Row Programming Done

Write this bit when the user row data is 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's sleep modes. This makes it possible for the UPDI to access the ACC layer even if the system is in a sleep mode.
Writing a '0' to this bit will lower the clock request.
This bit is set by default when the UPDI is enabled in any mode (Fuse, HV).

33.5.9. ASI System Status

Name: ASI_SYS_STATUS
Offset: 0x0B
Reset: 0x01
Property: -

Bit	7	6	5	4	3	2	1	0
	BDEF	ERASEFAIL	SYSRST	INSLEEP	PROGSTART	UROWSTART	BOOTDONE	LOCKSTATUS
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	1

Bit 7 – BDEF Boot Sequence Done or Chip Erase Failed
This bit is set to '1' if the chip erase has failed (ERASEFAIL bit is '1') or the boot sequence is complete (BOOTDONE bit is '1').

Bit 6 – ERASEFAIL Chip Erase Key Failed
This bit is set to '1' if the chip erase has failed. This bit is set to '0' on Reset. A Reset held from the ASI Reset Request (ASI_RESET_REQ) register will also affect this bit.

Bit 5 – SYSRST 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 – PROGSTART Start NVM Programming
When this bit is set to '1', NVM programming can start from the UPDI.
When the UPDI is done, the system must be reset through the ASI Reset Request (ASI_RESET_REQ) register.

Bit 2 – UROWSTART Start User Row Programming
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 1 – BOOTDONE Boot Sequence Done
This bit is set to '1' when the CPU is done with the boot sequence. The UPDI will not have access to the ACC layer until this bit is set to '1'.
Check also that SYSRST is '0' before proceeding.

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 is 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. Electrical Characteristics

35.1. Disclaimer

Unless otherwise specified, all typical values are measured at T_A = 25°C and V_{DD} = 3.0V. All minimum and maximum values are valid across the operating temperature and voltage ranges unless otherwise specified.

The given typical values must be considered for design guidance only, and part variation around the values is expected.

35.2. Absolute Maximum Ratings

Stresses beyond those listed in this section can cause permanent damage to the device and are for stress rating purposes only. The device's functional operation 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 35-1. Absolute Maximum Ratings

Parameter	Rating	Units	Condition
Ambient temperature under bias	-40 to +125	°C	
Maximum junction temperature	+140	°C	
Storage temperature	-65 to +150	°C	
Voltage on Pins with Respect to GND			
• On the V_{DD} pins	-0.3 to +6.0	V	
• On the $\overline{\text{RESET}}$ pin	-0.3 to +9.0	V	
• On all other pins	-0.3 to (V_{DD} + 0.3)	V	
Maximum Current			
• On the GND pins[1]	200	mA	-40°C ≤ T_A ≤ +85°C
	100	mA	+85°C < T_A ≤ +125°C
• On the V_{DD} pins[1]	200	mA	-40°C ≤ T_A ≤ +85°C
	100	mA	+85°C < T_A ≤ +125°C
• On any standard I/O pin	±40	mA	
Clamp current, I_K (V_{PIN} < 0 or V_{PIN} > V_{DD})	±20	mA	
Total power dissipation[2]	800	mW	

Notes:

1. The maximum current rating requires even load distribution across I/O pins. The maximum current rating may be limited by the device package power dissipation characterizations. See the *Thermal Characteristics* section to calculate device specifications.
2. Calculate power dissipation as follows: $P_{DIS} = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma \{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma (V_{OL} \times I_{OL})$

35.3. Standard Operating Conditions

For all other device characteristics to be valid, the device must operate within the ratings listed in this section.

Table 35-2. General Operating Conditions:

Operating Voltage:	$V_{DDMIN} \le V_{DD} \le V_{DDMAX}$
Operating Temperature:	$T_{A\_MIN} \le T_A \le T_{A\_MAX}$

The standard operating conditions for any device are defined as follows:

Table 35-3. Standard Operating Conditions

Parameter		Ratings	Units
V_{DD} — Operating Supply Voltage[1]			
Industrial and Extended Temperature	V_{DDMIN}	+1.62	V
	V_{DDMAX}	+5.5	V
T_A — Operating Ambient Temperature Range			
Industrial Temperature	T_{A_MIN}	-40	°C
	T_{A_MAX}	+85	°C
Extended Temperature	T_{A_MIN}	-40	°C
	T_{A_MAX}	+125	°C

Note:

1. Refer to the Supply Voltage parameter in the *Supply Voltage* section.

35.4. Supply Voltage

Table 35-4. Supply Voltage

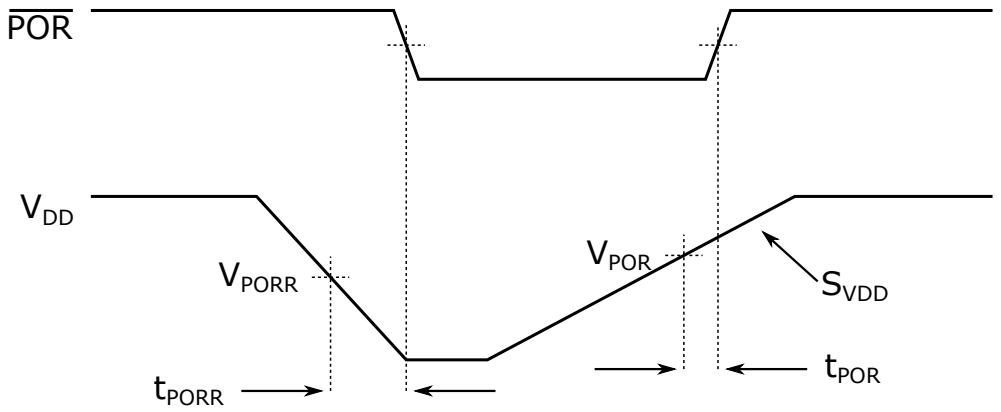
Symbol	Min.	Typ. †	Max.	Units	Conditions
Supply Voltage					
V_{DD}	1.62[1]	—	5.5	V	
RAM Data Retention[2]					
V_{DR}	1.62	—	—	V	
Power-on Reset Release Voltage[4]					
V_{POR}	—	1.5	1.62	V	BOD disabled[3]
Power-on Reset Re-Arm Voltage[4]					
V_{PORR}	—	1.1	—	V	BOD disabled[3]
V_{DD} Slope[5]					
S_{VDD}	—	—	0.2	V/µs	BOD disabled[3]

† Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified, . These parameters are not tested and are for design guidance only.

Notes:

1. Operation is ensured down to 1.62V or BOD triggering level V_{BOD} when BOD is active.
2. This is the limit to which V_{DD} can be lowered in Sleep mode without losing RAM data.
3. Refer to RSTCTRL and BOD for BOD trip point information.
4. Refer to Figure 35-1.
5. For design guidance only and not tested in production.

Figure 35-1. $\overline{POR}$ and $\overline{PORR}$ with Slow Rising V_{DD}



Note: When $\overline{POR}$ is low, the device is held in Reset.

35.5. Power Consumption

Table 35-5. Power Consumption in Active and Sleep Mode

Operating conditions: V_{DD} = 3.0V T_A = 25°C System power consumption measured with peripherals disabled and I/O ports driven low with inputs disabled								
Symbol	**Description**	**Min.**	**Typ.†**	**Max. 25°C**	**Max. 85°C**	**Max. 125°C**	**Unit**	**Conditions**
I_{DD}	Active power consumption	—	10	—	—	—	mA	OSCHF = 20 MHz, V_{DD} = 5V
		—	3.1	—	—	—	mA	OSCHF = 10 MHz (OSCHF/2)
		—	1.7	—	—	—	mA	OSCHF = 5 MHz (OSCHF/4)
		—	25	—	—	—	µA	OSC32K = 32.768 kHz
		—	25	—	—	—	µA	XOSC32K = 32.768 kHz, XOSC32KCTRLA.PMODE = 0x0
I_{DD_IDLE}	Idle power consumption	—	3.8	—	—	—	mA	OSCHF = 20 MHz, V_{DD} = 5V
		—	1.2	—	—	—	mA	OSCHF = 10 MHz (OSCHF/2)
		—	0.9	—	—	—	mA	OSCHF = 5 MHz (OSCHF/4)
		—	3.6	—	—	—	µA	OSC32K = 32.768 kHz
		—	3.3	—	—	—	µA	XOSC32K = 32.768 kHz, XOSC32KCTRLA.PMODE = 0x0
I_{DD_BASE}	Minimum power consumption in different sleep modes	—	0.06	0.2	—	—	µA	Power-Down or Standby mode, All peripherals disabled. V_{DD} = 1.8V
		—	0.07	0.3	3	15	µA	Power-Down or Standby mode, All peripherals disabled. V_{DD} = 3.0V
† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V, unless otherwise specified. These parameters are not tested and are for design guidance only.								

35.6. Peripherals Power Consumption

The table below shows how to calculate the additional current consumption for the different I/O peripherals in the various operating modes. Some peripherals request that the clock be enabled when operating in STANDBY. Refer to the peripheral section for further information.

Table 35-6. Peripherals Power Consumption[1]

Operating conditions: V_{DD} = 3.0V T_A = 25°C OSCHF at 20 MHz with a prescaler division factor of six is used as clock source Device in Standby sleep mode							
Symbol	**Description**	**Min.**	**Typ.†**	**Max. 85°C**	**Max. 125°C**	**Units**	**Conditions**
I_{DD_WDT}	Watchdog Timer (WDT)	—	0.68	—	—	uA	Including I_{DD_OSC32K}
I_{DD_BG}	Bandgap	—	13	—	—	µA	
I_{DD_VREF}	Voltage Reference (V_{REF})	—	40	—	—	µA	ACREF enabled, V_{REF} = 2.048V Excluding I_{DD_BG}
I_{DD_BOD}	Brown-out Detector (BOD)	—	4.7	—	—	µA	Continuous mode Excluding I_{DD_BG}, OSCHF = 16 MHz
		—	0.5	—	—	µA	Sampling mode @128 Hz Including I_{DD_OSC32K} Excluding I_{DD_BG}, OSCHF = 16 MHz
		—	0.5	—	—	µA	Sampling mode @32 Hz Including I_{DD_OSC32K} Excluding I_{DD_BG}, OSCHF = 16 MHz
I_{DD_TCB}	16-bit Timer/Counter Type B (TCB)	—	20	—	—	µA	CLKSEL = DIV1 (CLK_PER) Periodic Interrupt mode
I_{DD_TCE}	16-bit Timer/Counter Type E (TCE)	—	40	—	—	µA	CLKSEL = DIV1 (fTCE = fCLK_PER) WGMODE = NORMAL
I_{DD_RTC}	Real-Time Counter (RTC)	—	0.81	—	—	µA	Including I_{DD_OSC32K} CTRLA.PRESCALER = 0x5 (DIV32)
I_{DD_OSC32K}	32.768 kHz Internal Oscillator (OSC32K)	—	0.42	—	—	uA	
$I_{DD\_XOSC32K}$	32.768 kHz Crystal Oscillator (XOSC32K)	—	1.1	—	—	µA	XOSC32KCTRLA.PMODE = 0x0 ESR = 50 KΩ C_L = 8 pF
		—	1.3	—	—	µA	XOSC32KCTRLA.PMODE = 0x1 ESR = 70 KΩ C_L = 12.5 pF
		—	5.6	—	—	µA	XOSC32KCTRLA.PMODE = 0x2 ESR = 100 KΩ C_L = 24 pF
I_{DD_OSCHF}	Internal High Frequency Oscillator (OSCHF)	—	92	—	—	µA	
I_{DD_ADC}[2]	Analog-to-Digital Converter (ADC)	—	210	—	—	µA	CTRLC.REFSEL = 0x0 (V_{DD}) Excluding I_{DD_BG}
		—	265	—	—	µA	CTRLC.REFSEL = 0x5 (Internal reference 2.048V) Including I_{DD_BG}
I_{DD_AC}	Analog Comparator (AC)	—	44	—	—	µA	Continuous mode CTRLA.HYSMODE = 0x2
		—	4	—	—	µA	Event mode CTRLA.HYSMODE = 0x2
		—	6.5	—	—	µA	Sampled mode = 1 KHz CTRLA.HYSMODE = 0x2 Including I_{DD_OSC32K}
		—	4.2	—	—	µA	Sampled mode = 128 Hz CTRLA.HYSMODE = 0x2 Including I_{DD_OSC32K}

Table 35-6. Peripherals Power Consumption[1] (continued)

Operating conditions:

V_{DD} = 3.0V

T_A = 25°C

OSCHF at 20 MHz with a prescaler division factor of six is used as clock source

Device in Standby sleep mode

Symbol	Description	Min.	Typ.†	Max. 85°C	Max. 125°C	Units	Conditions
I_{DD_PTC}	Peripheral Touch Controller (PTC)	—	50	—	—	µA	Device current consumption in Standby mode with PTC sampling every 20 ms, CLK_ADC 2 MHz, 16x accumulation
I_{DD_USART}	Universal Synchronous and Asynchronous Receiver and Transmitter (USART)	—	63	—	—	µA	USART Enabled @9600 Baud, Device in Idle sleep mode
I_{DD_SPI}	Serial Peripheral Interface (SPI)	—	14	—	—	µA	SPI Host @100 kHz, Device in Idle sleep mode
I_{DD_TWI}	Two-Wire Interface (TWI)	—	56	—	—	µA	TWI Host @100 kHz, Device in Idle sleep mode
		—	46	—	—	µA	TWI Client @100 kHz, Device in Idle sleep mode
$I_{DD\_NVM\_ERASE}$	Flash Programming Erase	—	7	—	—	mA	
$I_{DD\_NVM\_WRITE}$	Flash Programming Write	—	10	—	—	mA	

† Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V, unless otherwise specified. These parameters are not tested and are for design guidance only.

Notes:

1. The module's current consumption only. To calculate the total internal power consumption of the microcontroller, add the power consumption values of all the peripherals and the clock sources used to the base power consumption given in Power Consumption.
2. Average power consumption with the following conditions:
   - Single-ended 10-bit ADC
   - ADC active in Free Running mode
   - CLK_ADC = 1 MHz
   - Device in Idle sleep mode

35.7. I/O Pins

Table 35-7. I/O Pin Specifications

Symbol	Description	Min.	Typ.†	Max.	Unit	Conditions
Input Low Voltage						
V_{IL}	I/O PORTS	—	—	0.2×V_{DD}	V	PINnCTRL.INLVL = 0x0
	$\overline{\text{RESET}}$ Pin	—	—	0.2×V_{DD}	V	
Input High Voltage						
V_{IH}	I/O PORTS	0.8×V_{DD}	—	—	V	PINnCTRL.INLVL = 0x0
	$\overline{\text{RESET}}$ pin	0.8×V_{DD}	—	—	V	
Input Leakage Current[1]						
I_{IL}	I/O PORTS	—	±5	±500	nA	GND ≤ VPIN ≤ VDD Pin at high-impedance TA= 85°C
	$\overline{\text{RESET}}$ pin[2]	—	±5	±500	nA	GND ≤ VPIN ≤ VDD Pin at high-impedance TA= 85°C

Table 35-7. I/O Pin Specifications (continued)

Symbol	Description	Min.	Typ.†	Max.	Unit	Conditions
Pull-up Resistance						
R_P		—	30	—	kΩ	
Output Low Voltage						
V_{OL}		—	—	0.4	V	I_{OL} = 6 mA V_{DD} = 3.0V
Output High Voltage						
V_{OH}		2.6	—	—	V	I_{OH} = -6 mA V_{DD} = 3.0V
I/O Slew Rate						
t_{SR}	Rising	—	45	—	ns	PORTCTRL.SRL = 0x01
		—	22	—	ns	PORTCTRL.SRL = 0x00
	Falling	—	30	—	ns	PORTCTRL.SRL = 0x01
		—	16	—	ns	PORTCTRL.SRL = 0x00
Pin Capacitance						
C_{IO}	EXTVREF0 pin	—	14	—	pF	
	XTAL32K1 pin	—	9	—	pF	
	XTAL32K2 pin	—	29	—	pF	
	PD6 pin	—	6	—	pF	
	Other pins	—	4	—	pF	

† Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V, unless otherwise specified. These parameters are not tested and are for design guidance only.

Notes:

1. The negative current is defined as the current sourced by the pin.
2. The leakage current on the $\overline{\text{RESET}}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. A higher leakage current may occur at different input voltages.

35.8. Memory Programming Specifications

Table 35-8. Memory Programming Specifications

Symbol	Description	Min.	Typ.†	Max.	Unit	Conditions
V_{P_REW}	V_{DD} for Read or Erase/Write Operation	V_{DDMIN}	—	V_{DDMAX}	V	
Data EEPROM Memory Specifications						
E_D*	Data EEPROM byte endurance	100k	—	—	Erase/Write cycles	
t_{D_RET}	Characteristic retention	—	40	—	Year	T_A = 55°C
t_{D_CE}	Full EEPROM Erase time	—	4	—	ms	
t_{D_BPW}	Byte/Page Write time	—	2	—	ms	
t_{D_BPE}	Byte/Page Erase time	—	2	—	ms	
t_{D_BPEW}	Atomic Byte/Page Erase/Write time	—	4	—	ms	
Program Flash Memory Specifications						
E_P*	Flash memory cell endurance	10k	—	—	Erase/Write cycles	
t_{P_RET}	Characteristic retention	—	40	—	Year	T_A = 55°C
t_{P_CE}	Chip Erase time	—	10	—	ms	
t_{P_UPDICE}	UPDI Chip Erase time	—	50	—	ms	Flash size = 16 KB
		—	70	—	ms	Flash size = 32 KB
t_{P_PE}	Page/Multipage Erase time	—	6	—	ms	

Table 35-8. Memory Programming Specifications (continued)

Symbol	Description	Min.	Typ.†	Max.	Unit	Conditions
t_{P_PW}	Page Write time	—	4	—	ms	
t_{P_PEW}	Atomic Page Erase/Write time	—	10	—	ms	

† Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V, unless otherwise specified. These parameters are not tested and are for design guidance only.

***** These parameters are characterized but not tested in production.

35.9. Thermal Specifications

Table 35-9. Thermal Specifications

Symbol	Description	Typ.	Unit	Conditions
θ_{JA}	Thermal Resistance Junction to Ambient	72	°C/W	14-pin TSSOP package (ST)
		62	°C/W	14-pin SOIC package (SL)
		60	°C/W	20-pin SSOP package (SS)
		54	°C/W	20-pin VQFN package (2LX)
		49	°C/W	28-pin PDIP package (SP)
		51	°C/W	28-pin SSOP package (SS)
		34	°C/W	28-pin VQFN package (3LW)
		31	°C/W	32-pin VQFN package (QZB)
		56	°C/W	32-pin TQFP package (PT)

Notes:

1. Calculate power dissipation as follows:
   $P_{DIS} = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma \{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma (V_{OI} \times I_{OL})$
2. Calculate internal power dissipation as follows:
   $P_{INTERNAL} = I_{DD} \times V_{DD}$,
   where I_{DD} is the current running the chip alone without driving any load on the output pins
3. Calculate derated power as follows:
   $P_{DER} = PD_{MAX} (T_J - T_A)/\theta_{JA}$,
   where T_A is the ambient temperature and T_J is the junction temperature

35.10. CLKCTRL

35.10.1. Internal Oscillators

Table 35-10. Internal Oscillators Specifications[(1)]

Operating Conditions: $V_{DDIO} = V_{DDMIN}$ to V_{DDMAX}, Temperature = -40 °C to +125 °C						
Symbol	**Description**	**Min.**	**Typ. †**	**Max.**	**Unit**	**Conditions**
f_{OSCHF}	Precision-calibrated OSCHF frequency	—	16	—	MHz	OSCCFG.OSCHFFRQ = 0x1
		—	20	—	MHz	OSCCFG.OSCHFFRQ = 0x0
f_{CAL}	Frequency tune range	—	±10	—	%	OSCCFG.OSCHFFRQ = 0x1
		—	±10	—	%	OSCCFG.OSCHFFRQ = 0x0
$\%_{CAL}$	OSCHF tune step size	—	0.4	—	%	

Table 35-10. Internal Oscillators Specifications[1] (continued)

Operating Conditions:
$V_{DDIO} = V_{DDMIN}$ to V_{DDMAX}
Temperature = -40 °C to +125 °C

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions
E_{OSCHF_TOTAL}	Total error with 16 MHz frequency selection[3]	-2	—	2	%	T_A = 25°C, V_{DD} = 3.0V
		-4	—	4	%	Full operation range
	Total error with 20 MHz frequency selection[4]	-1.5	—	1.5	%	T_A = 25°C, V_{DD} = 3.0V
		-3	—	3	%	Full operation range
D_{OSCHF}	OSCHF Duty cycle	—	50	—	%	
t_{OSCHF_ST}[2]	OSCHF Start-up time	—	10	—	µs	Within 2% accuracy
f_{OSC32K}	Internal OSC32K frequency	—	32.768	—	kHz	
E_{OSC32K_TOTAL}	Total error from target frequency	-4	—	4	%	T_A = 25°C, V_{DD} = 3.0V
		-15	—	15	%	T_A < 85°C
		-20	—	20	%	Full operation range
D_{OSC32K}	OSC32K duty cycle	—	50	—	%	
t_{OSC32K_ST}[2]	OSC32K Start-up time	—	220	—	µs	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Notes:

1. To ensure these oscillator frequency tolerances, V_{DD} and GND must be decoupled as close to the device as possible. 100 nF and 0.1 µF values in parallel are recommended.
2. Wake-up times are measured from the wake-up event to code execution.
3. The CLKCTRL.OSCHFTUNE is loaded from the Signature Row OSCHF16TUNE.
4. The CLKCTRL.OSCHFTUNE is loaded from the Signature Row OSCHF20TUNE.

35.10.2. XOSC32K

Table 35-11. 32.768 kHz Crystal Oscillator (XOSC32K) Specifications

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions
f_{XOSC32}	Frequency	—	32.768	—	kHz	
$C_{XTAL1/XTAL2}$	Parasitic pin capacitance	—	5	—	pF	
C_L	Crystal load capacitance	—	24	—	pF	XOSC32KCTRLA.PMODE = 0x2
		—	12.5	—	pF	XOSC32KCTRLA.PMODE = 0x1
		—	8	—	pF	XOSC32KCTRLA.PMODE = 0x0
ESR	Equivalent Series Resistance	—	100	—	kΩ	XOSC32KCTRLA.PMODE = 0x2
		—	70	—	kΩ	XOSC32KCTRLA.PMODE = 0x1
		—	50	—	kΩ	XOSC32KCTRLA.PMODE = 0x0
t_{XOSC32_ST}*	XOSC32 Start-up time	—	< 200	—	ms	XOSC32KCTRLA.PMODE = 0x2
		—	< 1000	—	ms	XOSC32KCTRLA.PMODE = 0x1
		—	< 1000	—	ms	XOSC32KCTRLA.PMODE = 0x0

† Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

* Dependent on crystal type and external components.

35.10.3. External Clock

Figure 35-2. External Clock Waveform

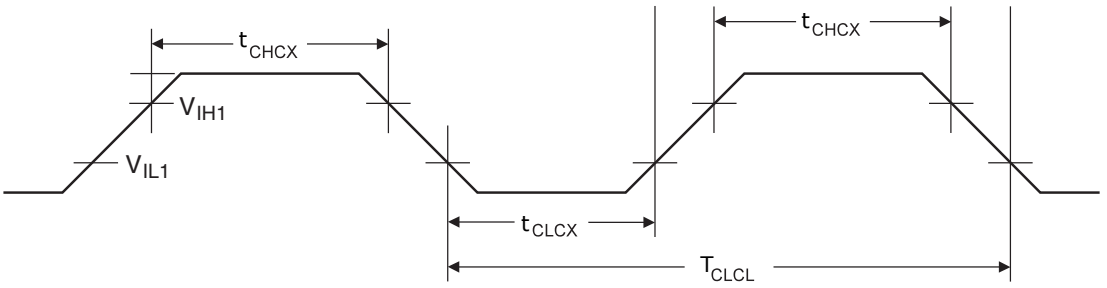


Table 35-12. External Clock Specifications

Symbol	Description	Min.	Typ. †	Max.	Units	Conditions
f_{CLCL}*	Clock frequency	—	—	20	MHz	
T_{CLCL}	Clock period	50	—	—	ns	
t_{CHCX}	High time	—	40	—	%	
t_{CLCX}	Low time	—	40	—	%	
ΔT_{CLCL}	Change in period from cycle to cycle time	—	20	—	%	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

***** The system clock is derived from the external clock and must be prescaled to comply with the system clock timing characterstics.

35.10.4. System Clock

Table 35-13. System Clock Timing Characteristics

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions
f_{CLK_CPU} f_{CLK_PER}	System clock frequency, industrial temperature range(1)	0	—	4	MHz	-40°C ≤ T_A ≤ +85°C 1.62V ≤ V_{DD} < 1.8V
		0	—	5	MHz	-40°C ≤ T_A ≤ +85°C 1.8V ≤ V_{DD} < 2.7V
		0	—	10	MHz	-40°C ≤ T_A ≤ +85°C 2.7V ≤ V_{DD} < 4.5V
		0	—	20	MHz	-40°C ≤ T_A ≤ +85°C V_{DD} ≥ 4.5V
	System clock frequency, extended temperature range(1)	0	—	3.4	MHz	-40°C ≤ T_A ≤ +125°C 1.62V ≤ V_{DD} < 1.8V
		0	—	4	MHz	-40°C ≤ T_A ≤ +125°C 1.8V ≤ V_{DD} < 2.7V
		0	—	8	MHz	-40°C ≤ T_A ≤ +125°C 2.7V ≤ V_{DD} < 4.5V
		0	—	16	MHz	-40°C ≤ T_A ≤ +125°C V_{DD} ≥ 4.5V
f_{CY}	Instruction clock frequency	—	f_{CLK_CPU}	—	MHz	
T_{CY}	Instruction cycle period(2)	50	$1/f_{CY}$	—	ns	

Table 35-13. System Clock Timing Characteristics (continued)

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Notes:

1. The system clock frequency (CLK_CPU) is configured by the Clock Select (CLKSEL) bit field, as described in the *CLKCTRL - Clock Controller* section.
2. Instruction Cycle Period (T_{CY}) is equal to the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type, under standard operating conditions with the device executing code. Exceeding these specified limits may result in incorrect code execution and/or higher than expected current consumption. All devices are tested to operate at 'Min.' values with an external clock applied to the EXTCLK pin. When using an external clock input, the 'Max.' cycle time limit is 'DC' (no clock) for all devices.

The maximum CPU clock frequency depends on V_{DD}. As shown in the following figure, the maximum frequency vs. V_{DD} is linear between 1.62V ≤ V_{DD} < 1.8V, 1.8V ≤ V_{DD} < 2.7V, and 2.7V ≤ V_{DD} < 4.5V.

Figure 35-3. Maximum Frequency vs. V_{DD} for [-40, 85]°C, Industrial Temperature

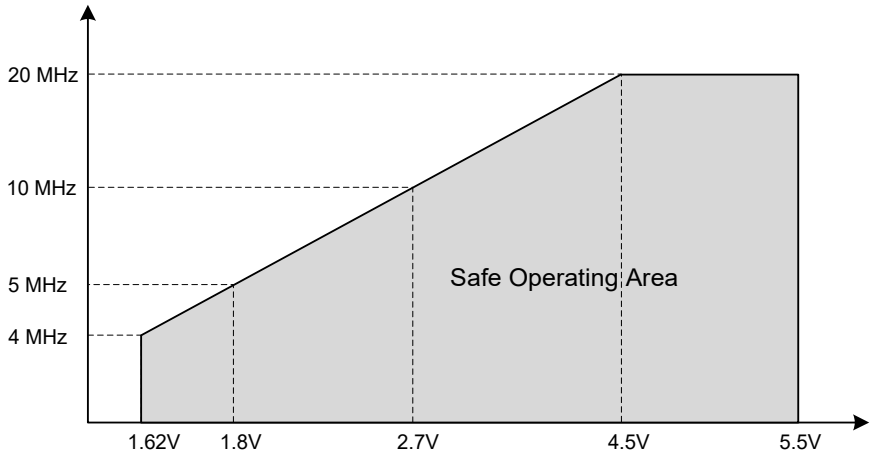
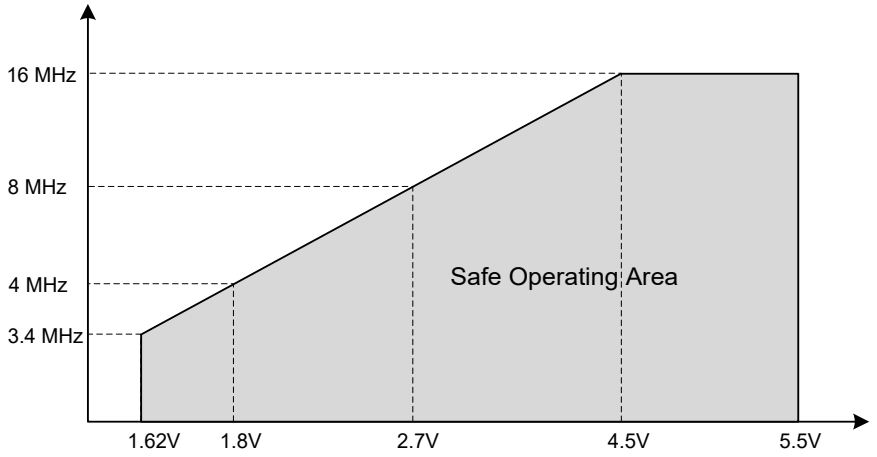


Figure 35-4. Maximum Frequency vs. V_{DD} for [-40, 125]°C, Extended Temperature



35.11. RSTCTRL and BOD

Table 35-14. Reset, WDT, Oscillator Start-up Timer, Power-up Timer, Brown-out Detector Specifications

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions
t_{RST} *	$\overline{\text{RESET}}$ pin pulse-width low to ensure a Reset	2.5	—	—	µs	
R_{RST_UP} *	$\overline{\text{RESET}}$ pin pull-up resistor	—	30	—	kΩ	

Table 35-14. Reset, WDT, Oscillator Start-up Timer, Power-up Timer, Brown-out Detector Specifications (continued)

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions
T_{OST}[1]	Oscillator Start-up timer period	—	1024	—	cycles	
V_{BOD+}[2]	Brown-out Detect voltage, rising slope	—	1.7	1.74	V	BODLEVEL0
		—	1.93	2.00	V	BODLEVEL1
		—	2.65	2.70	V	BODLEVEL2
		—	4.34	4.50	V	BODLEVEL3
V_{BOD-}[2]	Brown-out Detect voltage, falling slope	1.62	1.67	—	V	BODLEVEL0
		1.80	1.91	—	V	BODLEVEL1
		2.50	2.62	—	V	BODLEVEL2
		4.10	4.32	—	V	BODLEVEL3
V_{BOD_HYS}	Brown-out Detect hysteresis	—	25	—	mV	
t_{BOD_ST}	Brown-out Detect start-up time from sleep	—	40	—	µs	
t_{BOD_128Hz}	BOD response time Sampling mode @128 Hz	—	7.8	—	ms	CTRLA.SAMPFREQ = 0x0
t_{BOD_32Hz}	BOD response time Sampling mode @32 Hz	—	31.3	—	ms	CTRLA.SAMPFREQ = 0x1
t_{BOD_RST}	Brown-out Reset response time	—	5	—	µs	$V_{DD} = V_{BOD}$ - 0.1V

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.
***** These parameters are characterized but not tested in production.
Notes:

1. By design, the Oscillator Start-up Timer (T_{OST}) counts the first 1024 cycles, independent of frequency.
2. V_{DD} and GND must be capacitively decoupled as close to the device as possible to ensure these voltage tolerances. Recommended values are 0.1 µF and 0.01 µF in parallel.

Table 35-15. Voltage Level Monitor Threshold Specifications

Symbol	Description	Min.	Typ. †	Max.	Unit	Conditions
V_{DET} *	Voltage Detection Threshold	—	5	—	% above BOD threshold	VLMCTRLA.VLMLVL = 0x01
		—	17	—		VLMCTRLA.VLMLVL = 0x02
		—	27	—		VLMCTRLA.VLMLVL = 0x03

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are for design guidance only and are not tested.

***** These parameters are characterized but not tested in production.

35.12. VREF

Table 35-16. V_{REF} Specifications

Standard Operating conditions: T_A = -40°C to +125°C (unless otherwise specified)						
Symbol	**Description**	**Min.**	**Typ.†**	**Max.**	**Unit**	**Conditions**
VDD_{VREF_0V512}	V_{DD} for ADC internal VREF 0.512V	1.62	—	5.5	V	
VDD_{VREF_1V024}	V_{DD} for AC/ADC internal VREF 1.024V	1.62	—	5.5	V	
VDD_{VREF_2V048}	V_{DD} for AC/ADC internal VREF 2.048V	2.55	—	5.5	V	
VDD_{VREF_4V096}	V_{DD} for AC/ADC internal VREF 4.096V	4.6	—	5.5	V	
VDD_{VREF_2V5}	V_{DD} for AC/ADC internal VREF 2.5V	3.0	—	5.5	V	
VREF_0V512	ADC Internal reference 0.512V	—	0.512	—	V	
VREF_1V024	AC/ADC Internal reference 1.024V	—	1.024	—	V	
VREF_2V048	AC/ADC Internal reference 2.048V	—	2.048	—	V	
VREF_4V096	AC/ADC Internal reference 4.096V	—	4.096	—	V	

Table 35-16. V_{REF} Specifications (continued)

Standard Operating conditions: T_A = -40°C to +125°C (unless otherwise specified)						
Symbol	**Description**	**Min.**	**Typ.†**	**Max.**	**Unit**	**Conditions**
VREF_2V5	AC/ADC Internal reference 2.5V	—	2.5	—	V	
V_{VREF_0V512}(1)	ADC Internal voltage reference 0.512V	-1	—	1	%	T_A = 25°C
		-2	—	2	%	
V_{VREF_1V024}(1)	ADC Internal voltage reference 1.024V	-1	—	1	%	T_A = 25°C
		-2	—	2	%	
V_{VREF_2V048}(1)	ADC Internal voltage reference 2.048V	-1	—	1	%	T_A = 25°C
		-2	—	2	%	
V_{VREF_4V096}(1)	ADC Internal voltage reference 4.096V	-1	—	1	%	T_A = 25°C
		-2	—	2	%	
V_{VREF_2V5}(1)	ADC Internal voltage reference 2.5V	-1	—	1	%	T_A = 25°C
		-2	—	2	%	
V_{VREF_1V024}(1)	AC Internal voltage reference 1.024V	-2	—	2	%	
V_{VREF_2V048}(1)	AC Internal voltage reference 2.048V	-2	—	2	%	
V_{VREF_4V096}(1)	AC Internal voltage reference 4.096V	-2	—	2	%	
V_{VREF_2V5}(1)	AC Internal voltage reference 2.5V	-2	—	2	%	
t_{VREF_ST} *	Internal VREF Start-up Time	—	< 130	—	µs	
$V_{EXTVREF0}$	VREFA external reference input pin voltage	1.024	—	V_{DD}	V	

† Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

* These parameters are characterized but not tested in production.

Note:

1. The V_{VREF_xVxxx} symbol refers to the respective values of the REFSEL bit fields in the ADC0.CTRLC and VREF.AC0REF registers.

35.13. USART

Figure 35-5. USART in SPI Mode - Timing Requirements in Host Mode

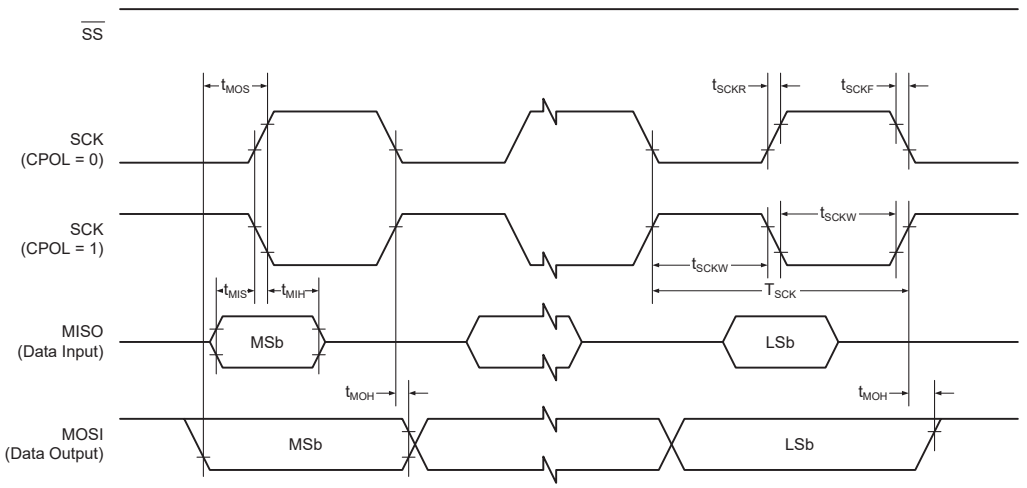


Table 35-17. USART in SPI Host Mode - Timing Characteristics

Symbol	Description	Min.	Typ. †	Max.	Units	Conditions
f_{SCK} (1)	SCK clock frequency	—	—	f_{CLK_PER}/2	MHz	

Table 35-17. USART in SPI Host Mode - Timing Characteristics (continued)

Symbol	Description	Min.	Typ. †	Max.	Units	Conditions
T_{SCK}[1]	SCK period	$2\times T_{CLK\_PER}$	—	—	ns	
t_{SCKW}	SCK high/low width	—	$0.5\times T_{SCK}$	—	ns	
t_{MOS}	MOSI valid before SCK	—	$0.5\times T_{SCK}$	—	ns	
t_{MOH}	MOSI hold after SCK	—	$0.5\times T_{SCK}$	—	ns	
t_{MIS}	MISO setup to SCK	—	T_{CLK_PER}	—	ns	
t_{MIH}	MISO hold after SCK	—	0	—	ns	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Note:

1. These parameters are characterized but not tested in production.

35.14. SPI

Figure 35-6. SPI - Timing Requirements in Host Mode

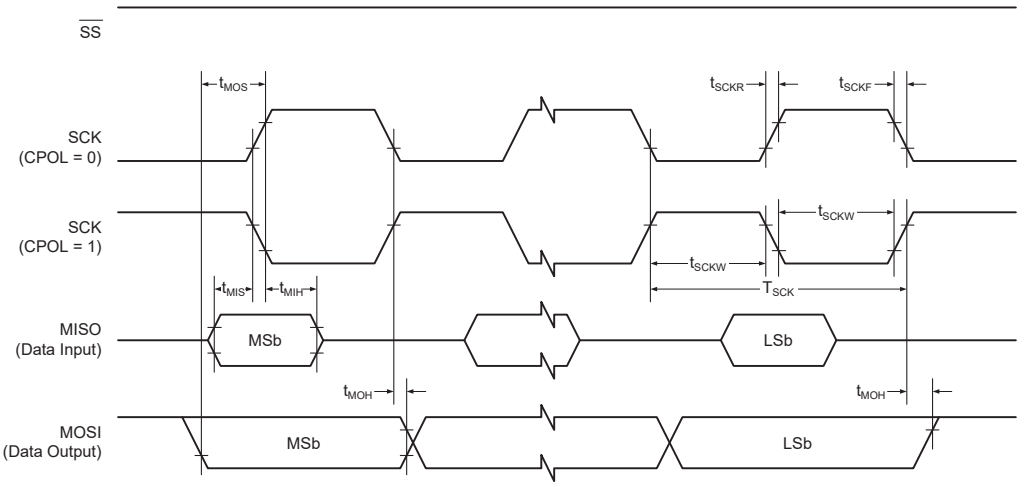


Table 35-18. SPI - Timing Specifications in Host Mode

Symbol	Description	Min.	Typ. †	Max.	Units	Conditions
f_{SCK}[1]	SCK clock frequency	—	—	$f_{CLK\_PER}/2$	MHz	
T_{SCK}[1]	SCK period	$2\times T_{CLK\_PER}$	—	—	ns	
t_{SCKW}	SCK high/low width	—	$0.5\times T_{SCK}$	—	ns	
t_{MOS}	MOSI valid before SCK	—	$0.5\times T_{SCK}$	—	ns	
t_{MOH}	MOSI hold after SCK	—	$0.5\times T_{SCK}$	—	ns	
t_{MIS}	MISO setup to SCK	—	T_{CLK_PER}	—	ns	
t_{MIH}	MISO hold after SCK	—	0	—	ns	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Note:

1. These parameters are characterized but not tested in production.

Figure 35-7. SPI - Timing Requirements in Client Mode

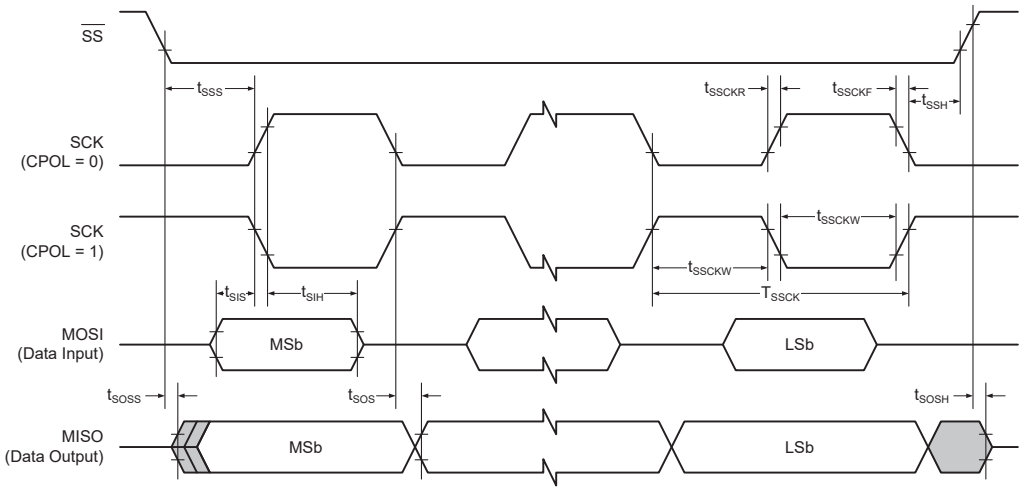


Table 35-19. SPI - Timing Specifications in Client Mode

Symbol	Description	Min.	Typ. †	Max.	Units	Conditions
f_{SSCK}[1]	Client SCK clock frequency	—	—	$f_{CLK\_PER}/6$	MHz	
T_{SSCK}[1]	Client SCK period	$6\times T_{CLK\_PER}$	—	—	ns	
t_{SSCKW}[1]	SCK high/low width	$3\times T_{CLK\_PER}$	—	—	ns	
t_{SIS}[1]	MOSI setup to SCK	0	—	—	ns	
t_{SIH}[1]	MOSI hold after SCK	$3\times T_{CLK\_PER}$	—	—	ns	
t_{SSS}[1]	$\overline{SS}$ low before SCK	—	T_{CLK_PER}	—	ns	
t_{SSH}[1]	$\overline{SS}$ high after SCK	—	T_{CLK_PER}	—	ns	
t_{SOS}	MISO Valid after SCK	—	t_{SR}[2]	—	ns	
t_{SOSS}	MISO setup after $\overline{SS}$ low	—	t_{SR}[2]	—	ns	
t_{SOSH}	MISO hold after $\overline{SS}$ high	—	t_{SR}[2]	—	ns	
t_{SDLY}	Interbyte delay	$5 - f_{CLK\_PER}/(2*f_{SSCK})$	—	—	ns	$f_{SSCK}<f_{CLK\_PER}/10$
		0	—	—	ns	$f_{SSCK}\geq f_{CLK\_PER}/10$

†Data in the "Typ." column is at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Notes:

1. These parameters are characterized but not tested in production.
2. t_{SR} is the I/O pin rise/fall time.

35.15. TWI

Figure 35-8. TWI - Timing Requirements

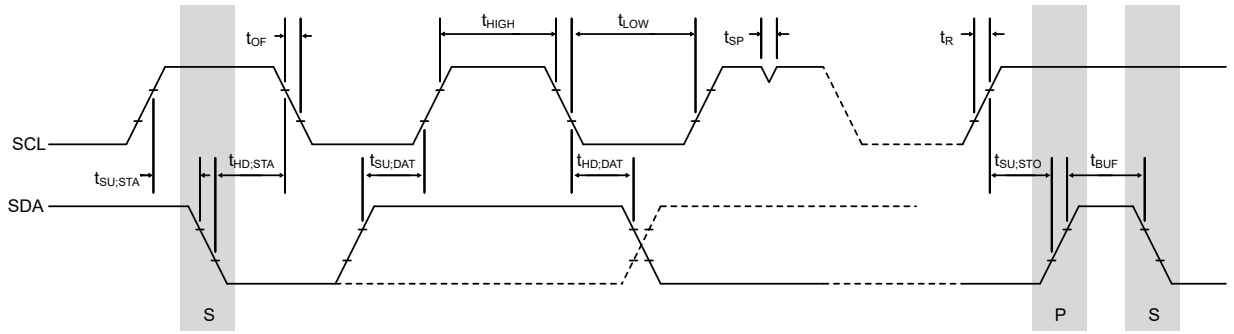


Table 35-20. TWI - Timing Specifications

Symbol	Description	Min.	Typ.†	Max.	Unit	Condition
f_{SCL}	SCL clock frequency (1)	—	—	1000	kHz	Fast-mode Plus (F_{m+})
		—	—	400	kHz	Fast-mode (F_m)
		—	—	100	kHz	Standard-mode (S_m)
V_{IL}	Low level input voltage	-0.5	—	$0.3 \times V_{DD}$	V	INPUTLVL = 0x0 (I^2C)
		-0.5	—	0.8		INPUTLVL = 0x1 (SMBUS) V_{DD} > 2.5V
V_{IH}	High level input voltage (3)	$0.7 \times V_{DD}$	—	V_{DD}+0.5V	V	INPUTLVL = 0x0 (I^2C)
		1.35	—	V_{DD}+0.5V		INPUTLVL = 0x1 (SMBUS) 0°C ≤ T_A ≤ +125°C
		1.5	—	V_{DD}+0.5V		INPUTLVL = 0x1 (SMBUS) -40°C ≤ T_A ≤ +125°C
V_{HYS}	Hysteresis of Schmitt Trigger inputs	$0.05 \times V_{DD}$	—	—	V	INPUTLVL = 0x0 (I^2C)
		—	0.1	—		INPUTLVL = 0x1 (SMBUS)
V_{OL}	Output low voltage (2)	—	—	0.6	V	I_{load} = 6 mA, V_{DD} > 2.5V
		—	—	0.4		I_{load} = 3 mA, V_{DD} > 2V
		—	—	$0.2 \times V_{DD}$		I_{load} = 2 mA, V_{DD} ≤ 2V
t_{SP} *	Spike pulse width suppressed by the input filter	0	—	t_{SPM}	ns	
t_{SPM} *	Input filter delay	50	—	200	ns	
t_{HD_DAT} *	Data hold time	—	0	—	ns	SDAHOLD[1:0] = 0x0
		—	50	—		SDAHOLD[1:0] = 0x1
		—	300	—		SDAHOLD[1:0] = 0x2
		300	500	900		SDAHOLD[1:0] = 0x3

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

***** These parameters are not tested and are for design guidance only.

Notes:

1. System clock frequency needs to be at least ten times faster than the TWI bus clock ($f_{CLK\_PER} \geq 10x\ f_{SCL}$) but additional limitations may apply depending on baud rate. Refer to the *TWI* chapter for more detailed information.
2. Fast-mode Plus full 20 mA drive capability is not supported.
3. If TWI pins are above V_{DD}+0.5V, the current will flow from the TWI pin(s) to V_{DD}.

35.16. AC

Table 35-21. Analog Comparator Specifications

Operating conditions: V_{DD} = 3.0V, T_A = 25°C						
Symbol	**Description**	**Min.**	**Typ.†**	**Max.**	**Unit**	**Conditions**
V_{IN} *	Input voltage range	-0.2	—	V_{DD}	V	
I_L	Input leakage current	—	5	—	nA	
V_{OFF} *	Input offset voltage	-30	±10	30	mV	0.1V < V_{IN} < (V_{DD} - 0.1V) CTRLA.HYSMODE = 0x0
CMRR	Common Mode Rejection Ratio	—	60	—	dB	V_{CM} = V_{DD} / 2
V_{HYST}	Hysteresis	—	5	—	mV	CTRLA.HYSMODE = 0x1
		—	10	—		CTRLA.HYSMODE = 0x2
		—	15	—		CTRLA.HYSMODE = 0x3
t_{RESP}	Response time, rising edge	—	180	—	ns	V_{CM} = V_{DD} / 2
	Response time, falling edge	—	240	—	ns	
t_{START}	AC Start-up time	—	10	—	us	OSC32K enabled and running

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are for design guidance only and are not tested.

***** These parameters are characterized but not tested in production.

Table 35-22. REFSCALE Specifications

Symbol	Description	Min.	Typ.†	Max.	Unit	Conditions
$V_{REFSCALE}$	Reference Scaler Max Voltage	—	V_{REF} - REFSCALE_LSB	—	V	REFSCALE = 0xFF
REFSCALE_RNG	Reference Scaler Range	0	—	$V_{REFSCALE}$	V	
REFSCALE_LSB	Reference Scaler Resolution	—	1/ (256*V_{REF})	—	V	
$E_{REFSCALE}$ (1)	Reference Scaling Error	-2	—	+2	%	Excluding error from V_{REF}
t_{READY} *	Ready Time	—	—	100	µs	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are for design guidance only and are not tested.

***** These parameters are characterized but not tested in production.

Note:

1. Error given as a percentage of the selected REFSCALE reference voltage. Calculate the total accuracy of the comparator trip point when using a scaled reference voltage as follows:
   $Error_{Min}$(negative) = $VREF_{Min}$ * (1 + 0.01*$E_{REFSCALE\_Min}$) - $VREF_{Typ}$ - | VOFF |
   $Error_{Max}$ (positive) = $VREF_{Max}$ * (1 + 0.01*$E_{REFSCALE\_Max}$) - $VREF_{Typ}$ + | VOFF |
   $VREF_{Min}$ and $VREF_{Max}$ represent the minimum and maximum values of the reference voltage. If V_{REF} is obtained from the device's internal V_{REF} source, refer to the VREF section to calculate the minimum and maximum V_{REF} values. If V_{REF} is obtained externally, the characteristics of the external V_{REF} source must be used to determine the minimum and maximum V_{REF} values.
   When input hysteresis is enabled, add ±V_{HYST}/2 to the rising and falling trip point.

35.17. ADC

Table 35-23. Power Supply, Reference and Input Range

Operating conditions: V_{DD} = 3.0V, T_A = 25°C, Applies to all allowed combinations of V_{REF} selections and sample rates, unless otherwise specified						
Symbol	**Description**	**Min.**	**Typ.†**	**Max.**	**Unit**	**Conditions**
V_{DD}	Supply voltage	1.62	—	5.5	V	
V_{REF}	Reference voltage	0.512	—	V_{DD}	V	
C_{IN}	Input capacitance	—	3	—	pF	
R_{IN}	Input resistance	—	3	—	kΩ	
V_{IN}	Input voltage range	0	—	V_{REF}	V	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Table 35-24. Clock and Timing Characteristics

Operating conditions: Applies to all allowed combinations of V_{REF} selections and sample rates, unless otherwise specified						
Symbol	**Description**	**Min.**	**Typ.†**	**Max.**	**Unit**	**Conditions**
CLK_ADC	ADC clock frequency	300	—	2000	kHz	REFSEL = Internal Reference
		300	—	2000	kHz	REFSEL = External Reference REFSEL = V_{DD}

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Table 35-25. Accuracy Characteristics (1)

Operating conditions: V_{DD} = 3.0V, V_{REF} = External 3.0V, T_A = 25°C, f_{CLK_ADC} = 1 MHz, CPU in Sleep mode						
Symbol	**Description**	**Min.**	**Typ.†**	**Max.**	**Unit**	**Conditions**
Res	Resolution	—	—	10	bit	
E_{INL}	Integral nonlinearity error	—	0.3	—	LSb	
E_{DNL}	Differential nonlinearity error	—	0.3	—	LSb	
E_{OFF}	Offset error	—	0.3	—	LSb	
E_{GAIN}	Gain error	—	2	—	LSb	
E_T	Total unadjusted error	—	3.5	—	LSb	

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.

Note:

1. These values are based on characterization and are not covered by production test limits.

35.18. PTC

Table 35-26. Peripheral Touch Controller Specifications

Symbol	Description	Min.	Typ.†	Max.	Unit	Conditions
V_{DD}	Supply voltage	1.62	—	5.5	V	
C_{LOAD} *	Maximum Load, Mutual Capacitance mode	—	—	27	pF	Without external compensation capacitor
	Maximum Load, Self-Capacitance mode (Y1)	—	—	25	pF	
	Maximum Load, Self-Capacitance mode (Y0, Y17, Y18)	—	—	40	pF	
	Maximum Load, Self-Capacitance mode (Other PTC pins)	—	—	50	pF	
C_{LOAD}*	Maximum Load, Mutual Capacitance mode	—	—	C_{EXT_CC}	nF	With external compensation capacitor
	Maximum Load, Self-Capacitance mode	—	—	C_{EXT_CC}	nF	
C_{EXT_IC}(1)*	Size of External Integration Capacitor	—	—	10	nF	
C_{EXT_CC}(1)*	Size of External Compensation Capacitor	—	—	1	nF	
C_{SH}*	Driven shield capacitive drive	—	—	1	nF	Per driven shield pin
CLK_{ADC}	Supported ADC Clock frequency	300	—	2000	kHz	
CLK_{PTC}	PTC Clock frequency	—	—	f_{CLK_PER}	MHz	
EXT_R_S_Y *	External Series Resistor (Y-line)	0	10	100	KΩ	For EMC, Self-Capacitance mode
EXT_R_S_X *	External Series Resistor (X-line)	0	1	10	KΩ	For EMC, Mutual Capacitance mode

† Data in the "Typ." column is measured at T_A = 25°C and V_{DD} = 3.0V unless otherwise specified. These parameters are not tested and are for design guidance only.
***** These Parameters are characterized but not tested in production.
Note:

1. Refer to the PTC library documentation for sizing.

35.19. UPDI

Figure 35-9. UPDI Enable Sequence with Dedicated UPDI Pin

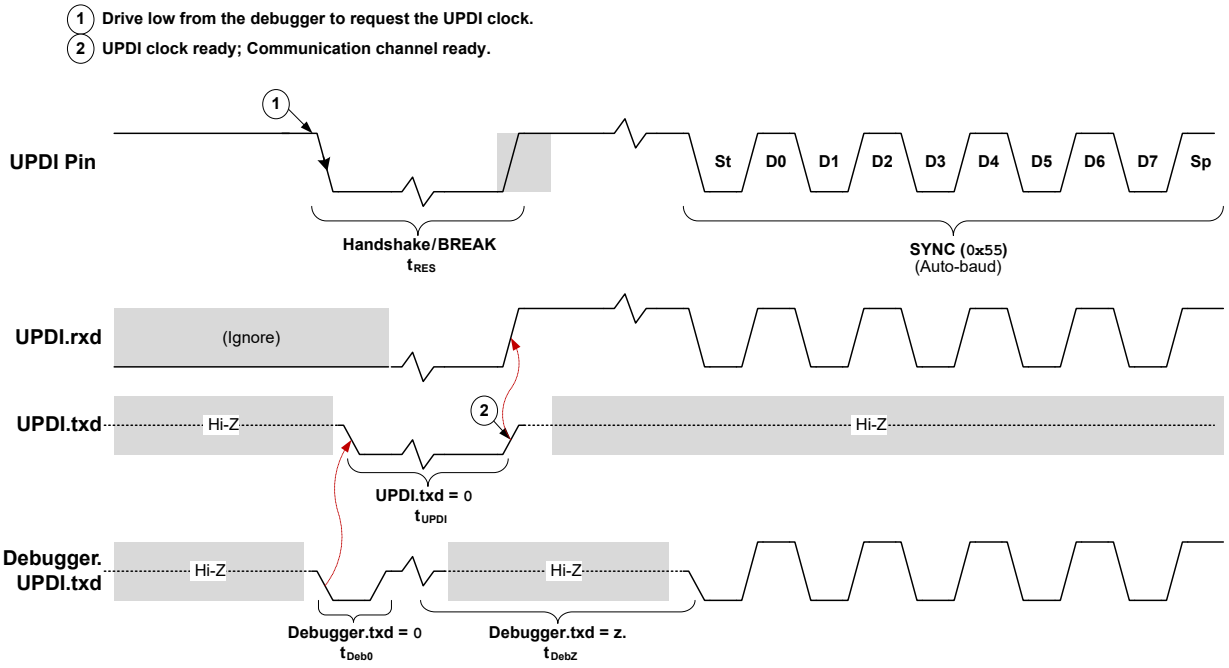


Table 35-27. UPDI Timing Specifications

Symbol	Description	Min.	Max.	Unit	Conditions	
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		
f_{UPDI} *	UPDI clock frequency	—	4	MHz	$V_{DDMIN} \le V_{DD} \le V_{DDMAX}$	$T_{A\_MIN} \le T_A \le T_{A\_MAX}$
		—	8	MHz	$2.7V \le V_{DD} \le 5.5V$	
		—	16	MHz	$4.5V \le V_{DD} \le 5.5V$	
		—	8	MHz	$V_{DDMIN} \le V_{DD} \le V_{DDMAX}$	$0°C \le T_A \le +50°C$
		—	16	MHz	$2.7V \le V_{DD} \le 5.5V$	
		—	32	MHz	$4.5V \le V_{DD} \le 5.5V$	

***** These parameters are for deign guidance only and are not tested.

Figure 35-10. UPDI Enable Sequence by High-Voltage (HV) Programming

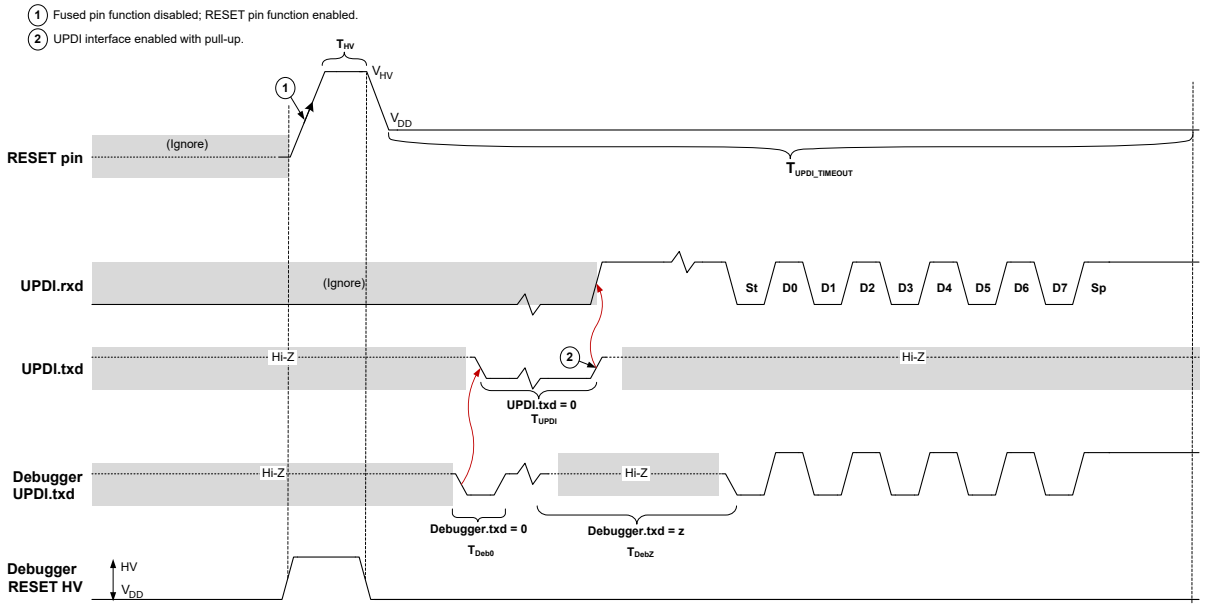


Table 35-28. UPDI HV Pulse Specifications

Symbol	Description	Min.	Typ.	Max.	Unit	Conditions
V_{HV} *	Debugger RESET HV signal level	V_{DD}+2	7.5	8.5	V	
T_{HV} *	Debugger RESET HV signal duration	10	—	—	µs	
$T_{UPDI\_TIMEOUT}$ *	Time to receive valid key after HV pulse	—	65	—	ms	

***** These parameters are for design guidance only and are not tested.

36. Characteristics Graphs

Characteristics Graphs are not available at this time.

37. Ordering Information

- Available Ordering Options Can Be Found By:
  - Clicking on one of the following product page links:
    - AVR16LA14
    - AVR16LA20
    - AVR16LA28
    - AVR16LA32
  - Searching by product name at www.microchipdirect.com
  - Contacting your local sales representative

Note: Automotive-grade ordering codes (VAO suffix) are set up upon request for devices that are labelled “Recommended for Automotive Designs” on the product page. To request VAO ordering codes not listed on the respective product page, contact your local Microchip sales representative.

Table 37-1. Available Product Numbers

CPN	Flash/SRAM	Pin Count	Package Type	Temperature Range	Carrier Type
AVR16LA14-I/SL	16 KB/2 KB	14	SOIC	-40°C to +85°C	Tube
AVR16LA14T-I/SL	16 KB/2 KB	14	SOIC	-40°C to +85°C	Tape & Reel
AVR16LA14-E/SL	16 KB/2 KB	14	SOIC	-40°C to +125°C	Tube
AVR16LA14T-E/SL	16 KB/2 KB	14	SOIC	-40°C to +125°C	Tape & Reel
AVR16LA14-I/ST	16 KB/2 KB	14	TSSOP	-40°C to +85°C	Tube
AVR16LA14T-I/ST	16 KB/2 KB	14	TSSOP	-40°C to +85°C	Tape & Reel
AVR16LA14-E/ST	16 KB/2 KB	14	TSSOP	-40°C to +125°C	Tube
AVR16LA14T-E/ST	16 KB/2 KB	14	TSSOP	-40°C to +125°C	Tape & Reel
AVR16LA20-I/SS	16 KB/2 KB	20	SSOP	-40°C to +85°C	Tube
AVR16LA20T-I/SS	16 KB/2 KB	20	SSOP	-40°C to +85°C	Tape & Reel
AVR16LA20-E/SS	16 KB/2 KB	20	SSOP	-40°C to +125°C	Tube
AVR16LA20T-E/SS	16 KB/2 KB	20	SSOP	-40°C to +125°C	Tape & Reel
AVR16LA20-I/2LX	16 KB/2 KB	20	VQFN WF	-40°C to +85°C	Tray
AVR16LA20T-I/2LX	16 KB/2 KB	20	VQFN WF	-40°C to +85°C	Tape & Reel
AVR16LA20-E/2LX	16 KB/2 KB	20	VQFN WF	-40°C to +125°C	Tray
AVR16LA20T-E/2LX	16 KB/2 KB	20	VQFN WF	-40°C to +125°C	Tape & Reel
AVR16LA28-I/SP	16 KB/2 KB	28	SPDIP	-40°C to +85°C	Tube
AVR16LA28-E/SP	16 KB/2 KB	28	SPDIP	-40°C to +125°C	Tube
AVR16LA28-I/SS	16 KB/2 KB	28	SSOP	-40°C to +85°C	Tube
AVR16LA28T-I/SS	16 KB/2 KB	28	SSOP	-40°C to +85°C	Tape & Reel
AVR16LA28-E/SS	16 KB/2 KB	28	SSOP	-40°C to +125°C	Tube
AVR16LA28T-E/SS	16 KB/2 KB	28	SSOP	-40°C to +125°C	Tape & Reel
AVR16LA28-I/3LW	16 KB/2 KB	28	VQFN WF	-40°C to +85°C	Tube
AVR16LA28T-I/3LW	16 KB/2 KB	28	VQFN WF	-40°C to +85°C	Tape & Reel
AVR16LA28-E/3LW	16 KB/2 KB	28	VQFN WF	-40°C to +125°C	Tube
AVR16LA28T-E/3LW	16 KB/2 KB	28	VQFN WF	-40°C to +125°C	Tape & Reel
AVR16LA32-I/QZB	16 KB/2 KB	32	VQFN WF	-40°C to +85°C	Tray
AVR16LA32T-I/QZB	16 KB/2 KB	32	VQFN WF	-40°C to +85°C	Tape & Reel
AVR16LA32-E/QZB	16 KB/2 KB	32	VQFN WF	-40°C to +125°C	Tray
AVR16LA32T-E/QZB	16 KB/2 KB	32	VQFN WF	-40°C to +125°C	Tape & Reel

Table 37-1. Available Product Numbers (continued)

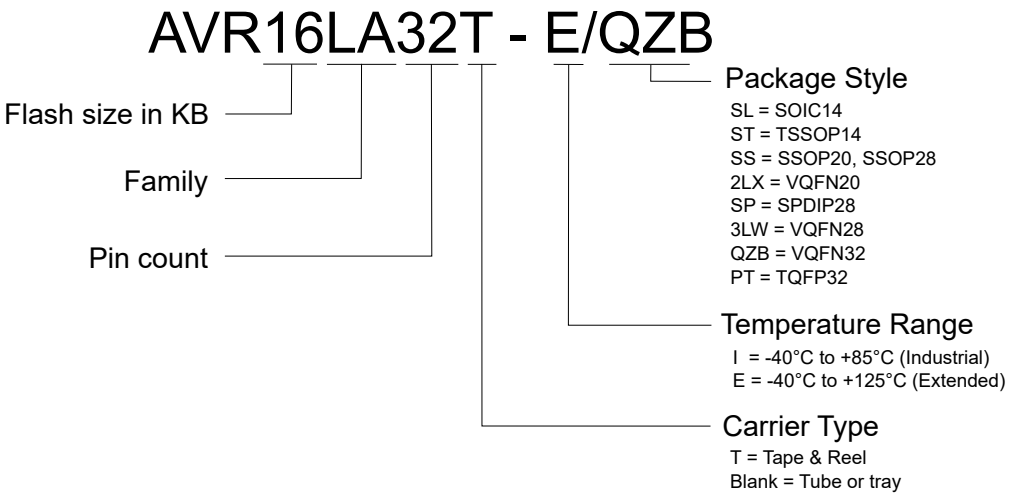
CPN	Flash/SRAM	Pin Count	Package Type	Temperature Range	Carrier Type
AVR16LA32-I/PT	16 KB/2 KB	32	TQFP	-40°C to +85°C	Tray
AVR16LA32T-I/PT	16 KB/2 KB	32	TQFP	-40°C to +85°C	Tape & Reel
AVR16LA32-E/PT	16 KB/2 KB	32	TQFP	-40°C to +125°C	Tray
AVR16LA32T-E/PT	16 KB/2 KB	32	TQFP	-40°C to +125°C	Tape & Reel

Notes:

1. Pb-free packing complies with the European Directive for Restrictions of Hazardous Substances (RoHS directive). Also halide-free and completely green.
2. Package outline drawings can be found in the *Package Drawings* section.

Figure 37-1. Product Identification System

To order or obtain information, such as pricing or delivery details, 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.

38. Package Drawings

38.1. Online package drawings

For the most recent package drawings:

1. Go to 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 drawings.

Table 38-1. Drawing Numbers

Pin Count	Package Type	Drawing Number	Style	Package Code
14	SOIC	C04-00065	SL	D3X
14	TSSOP	C04-00087	ST	D4X
20	SSOP	C04-00072	SS	G3X
20	VQFN WF(1)	C04-00476	2LX	2LX
28	SSOP	C04-00073	SS	N2X
28	SPDIP	C04-00070	SP	M3X
28	VQFN WF(1)	C04-00565	3LW	3LW
32	TQFP	C04-00074	PT	T5X
32	VQFN WF(1)	C04-21511	QZB	QZB

Note:

1. This package type has wettable flanks (WF).

38.2. Package Marking Information

Legend:	XX...X	Customer-specific information or Microchip part number
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC® designator for Matte Tin (Sn)
Note:	In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.	

38.2.1. 14-Pin SOIC

Figure 38-1. General

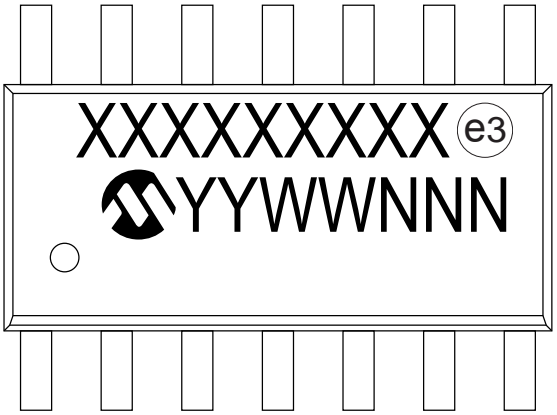
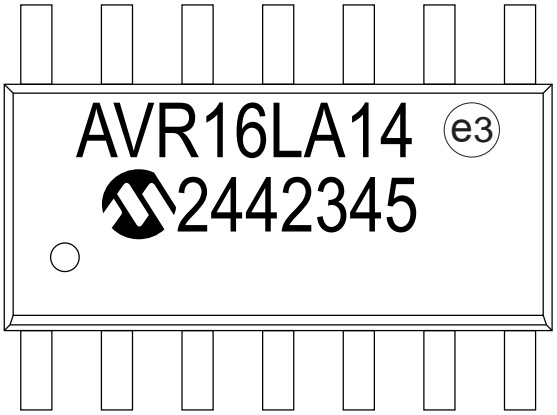


Figure 38-2. Example



38.2.2. 14-Pin TSSOP

Figure 38-3. General

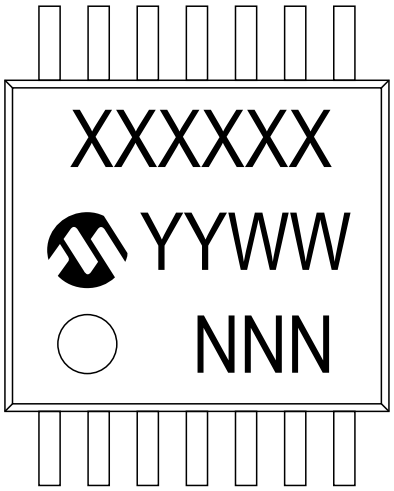
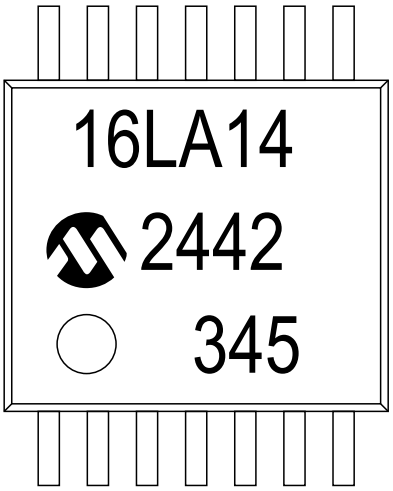


Figure 38-4. Example



38.2.3. 20-Pin SSOP

Figure 38-5. General

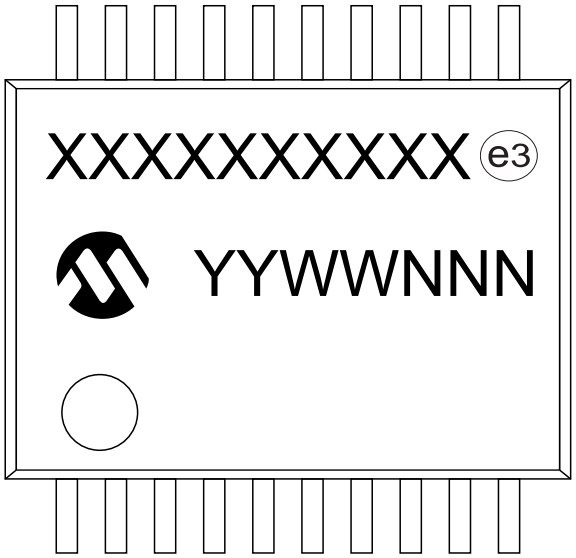
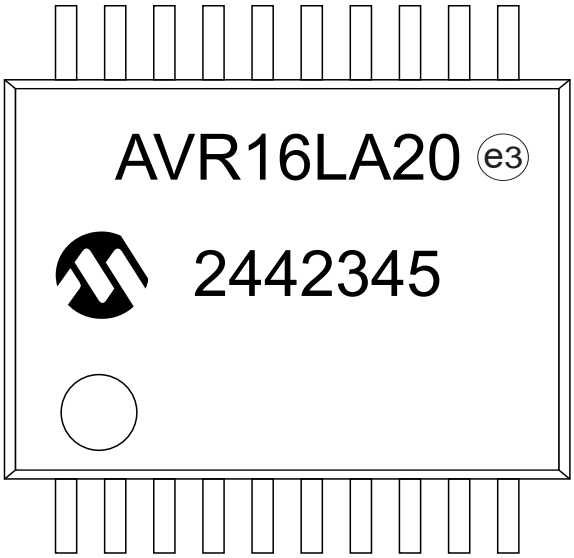


Figure 38-6. Example



38.2.4. 20-Pin VQFN Wettable Flanks

Figure 38-7. General

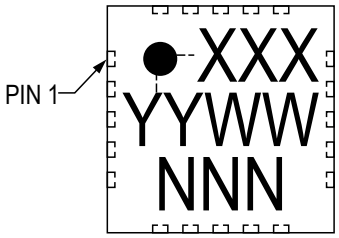
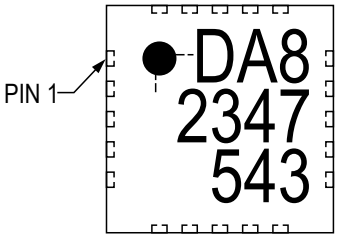


Figure 38-8. Example



38.2.5. 28-Pin SSOP

Figure 38-9. General

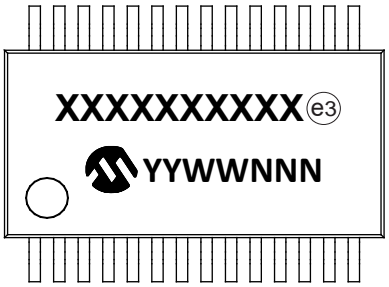
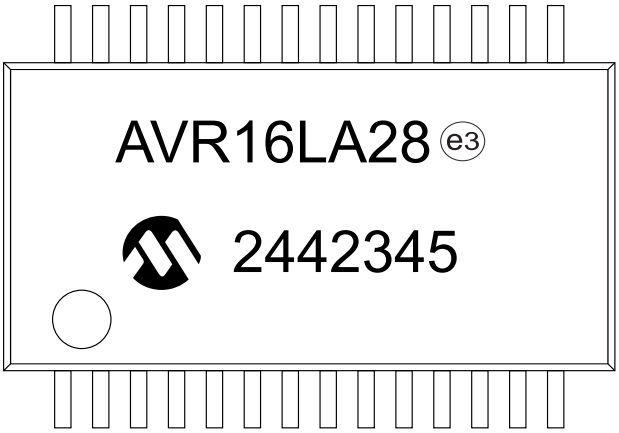


Figure 38-10. Example



38.2.6. 28-Pin SPDIP

Figure 38-11. General

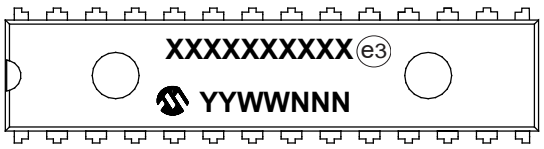


Figure 38-12. Example

38.2.7. 28-Pin VQFN Wettable Flanks

Figure 38-13. General

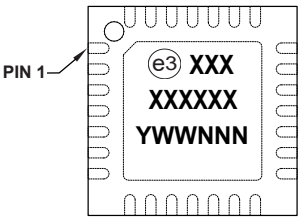
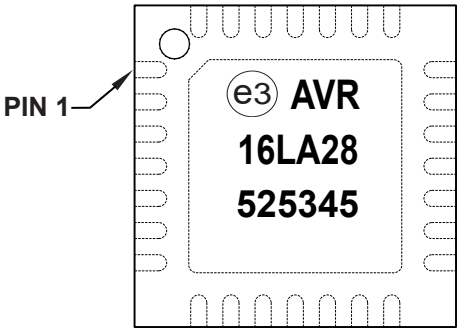


Figure 38-14. Example



38.2.8. 32-Pin TQFP

Figure 38-15. General

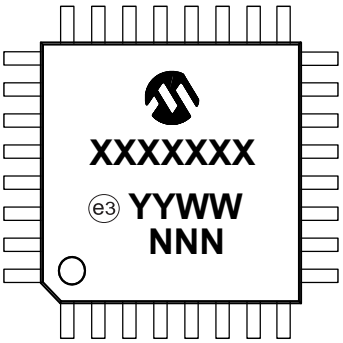
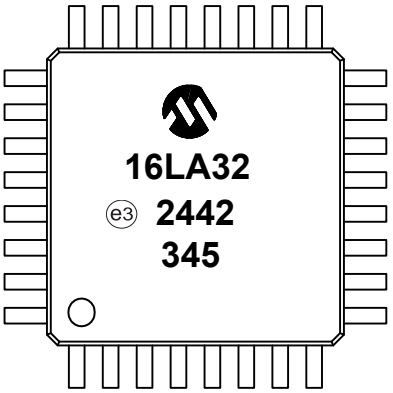


Figure 38-16. Example



38.2.9. 32-Pin VQFN Wettable Flanks

Figure 38-17. General

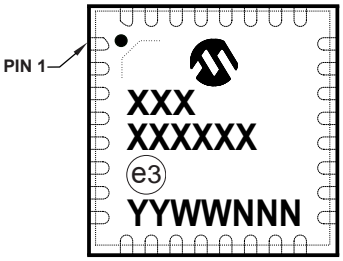
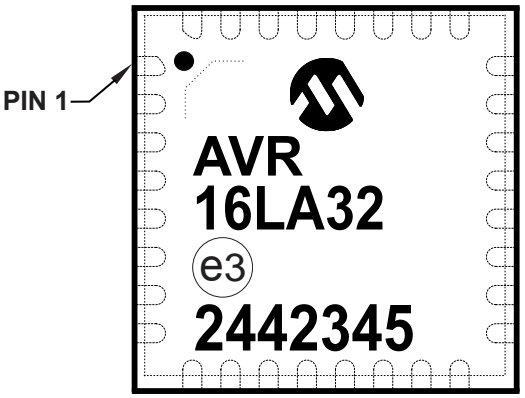


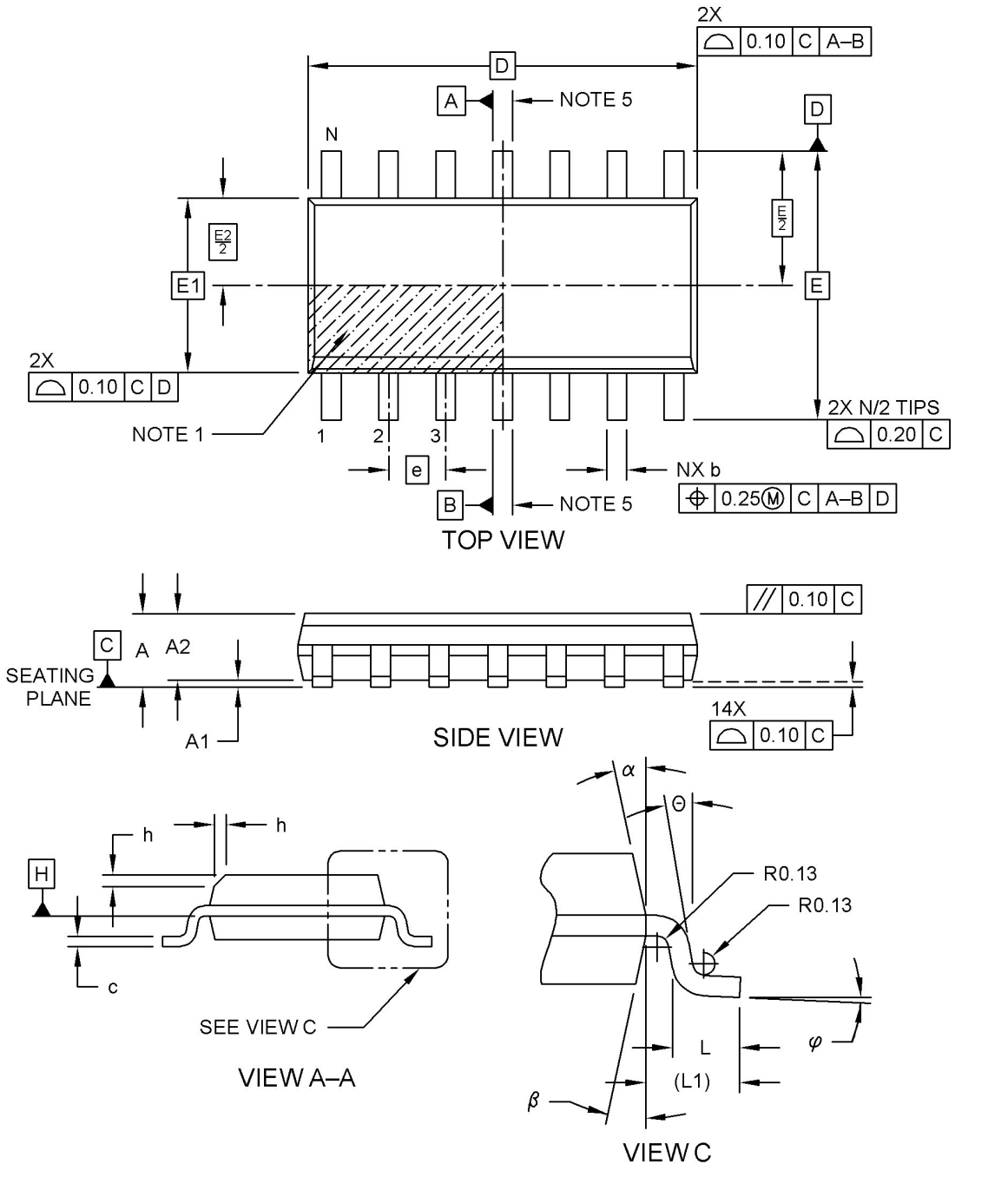
Figure 38-18. Example



38.3. 14-Pin SOIC

14-Lead Plastic Small Outline (D3X) - Narrow, 3.90 mm Body [SOIC]

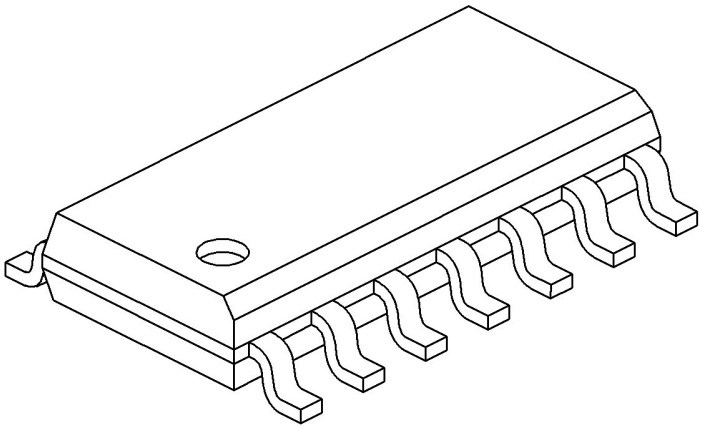
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing No. C04-065-D3X Rev E Sheet 1 of 2

14-Lead Plastic Small Outline (D3X) - Narrow, 3.90 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		14	
Pitch	e		1.27 BSC	
Overall Height	A	-	-	1.75
Molded Package Thickness	A2	1.25	-	-
Standoff §	A1	0.10	-	0.25
Overall Width	E		6.00 BSC	
Molded Package Width	E1		3.90 BSC	
Overall Length	D		8.65 BSC	
Chamfer (Optional)	h	0.25	-	0.50
Foot Length	L	0.40	-	1.27
Footprint	L1		1.04 REF	
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.10	-	0.25
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

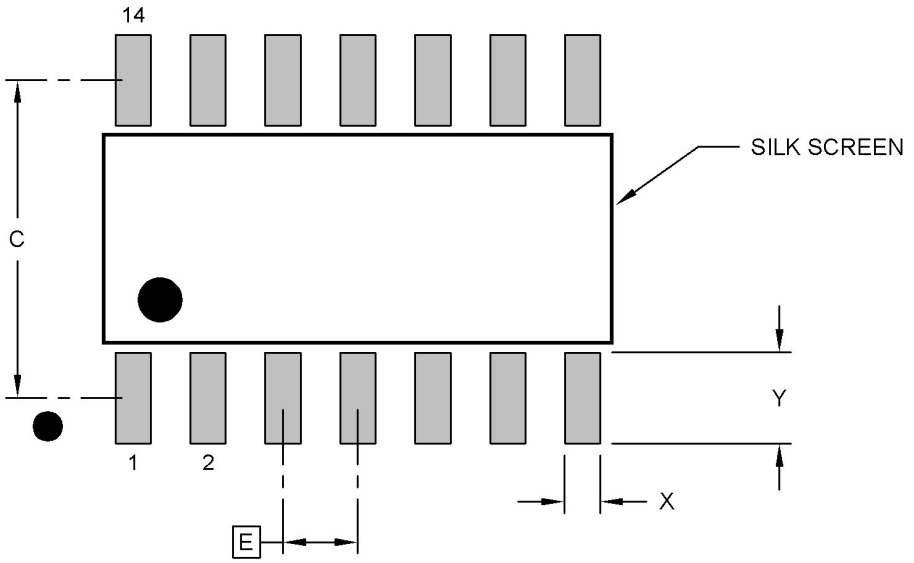
Notes:

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. C04-065-D3X Rev E Sheet 2 of 2

14-Lead Plastic Small Outline (D3X) - Narrow, 3.90 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		5.40	
Contact Pad Width (X14)	X			0.60
Contact Pad Length (X14)	Y			1.55

Notes:

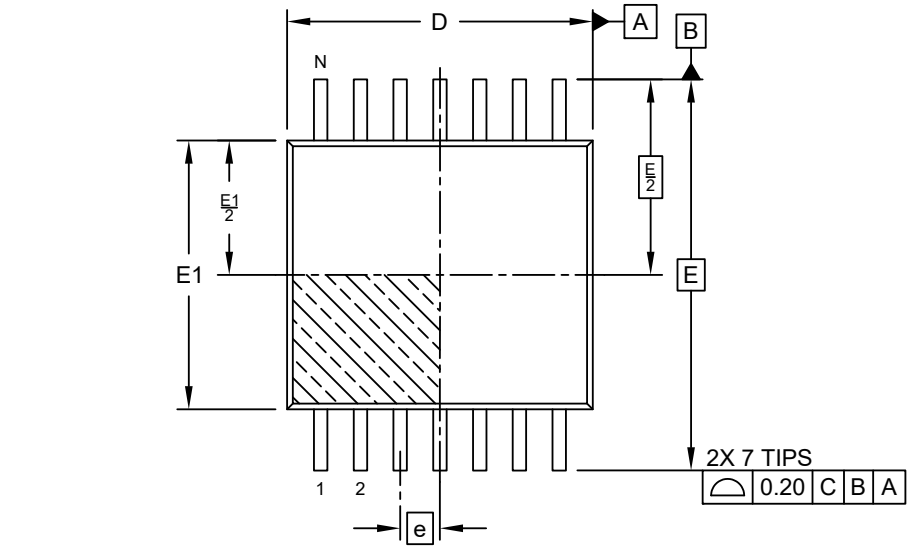
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2065-D3X Rev E

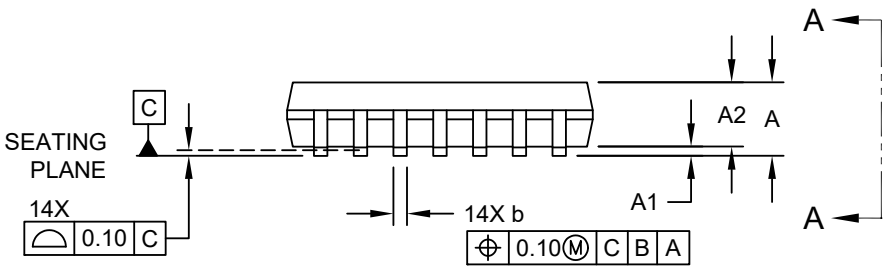
38.4. 14-Pin TSSOP

14-Lead Plastic Thin Shrink Small Outline Package [ST] - 4.4 mm Body [TSSOP]

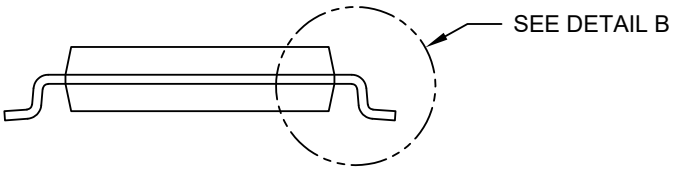
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW



SIDE VIEW

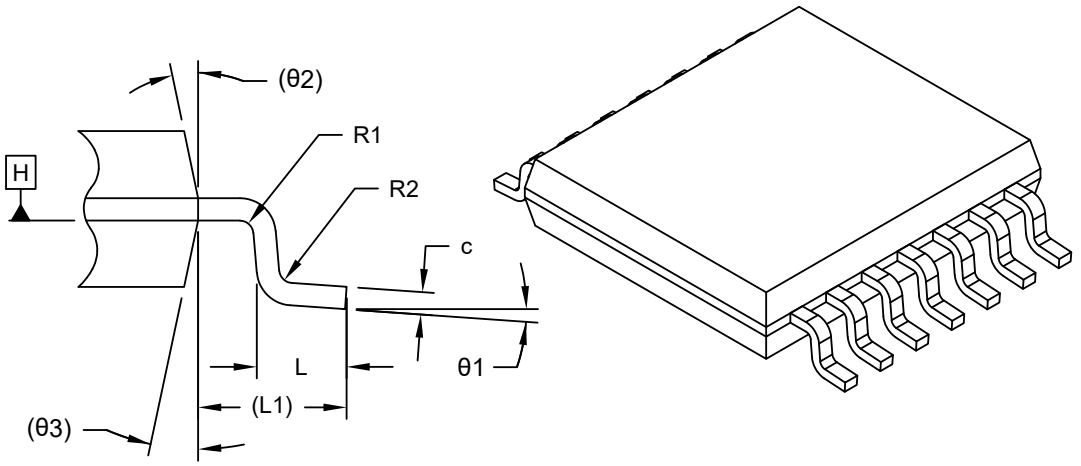


VIEW A–A

Microchip Technology Drawing C04-087-ST Rev F Sheet 1 of 2

14-Lead Plastic Thin Shrink Small Outline Package [ST] - 4.4 mm Body [TSSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



DETAIL B

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		14	
Pitch	e		0.65 BSC	
Overall Height	A	–	–	1.20
Standoff	A1	0.05	–	0.15
Molded Package Thickness	A2	0.80	1.00	1.05
Overall Length	D	4.90	5.00	5.10
Overall Width	E		6.40 BSC	
Molded Package Width	E1	4.30	4.40	4.50
Terminal Width	b	0.19	–	0.30
Terminal Thickness	c	0.09	–	0.20
Terminal Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Lead Bend Radius	R1	0.09	–	–
Lead Bend Radius	R2	0.09	–	–
Foot Angle	θ1	0°	–	8°
Mold Draft Angle	θ2	–	12° REF	–
Mold Draft Angle	θ3	–	12° REF	–

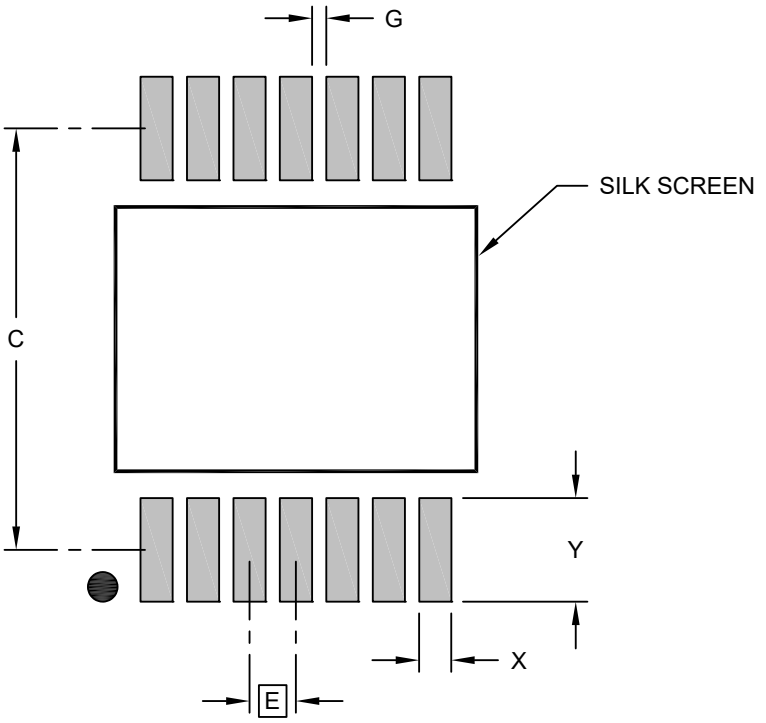
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-087-ST Rev F Sheet 2 of 2

14-Lead Plastic Thin Shrink Small Outline Package [ST] – 4.4 mm Body [TSSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Contact Pad Spacing	C		5.90	
Contact Pad Width (X14)	X			0.45
Contact Pad Length (X14)	Y			1.45
Contact Pad to Contact Pad (X12)	G	0.20		

Notes:

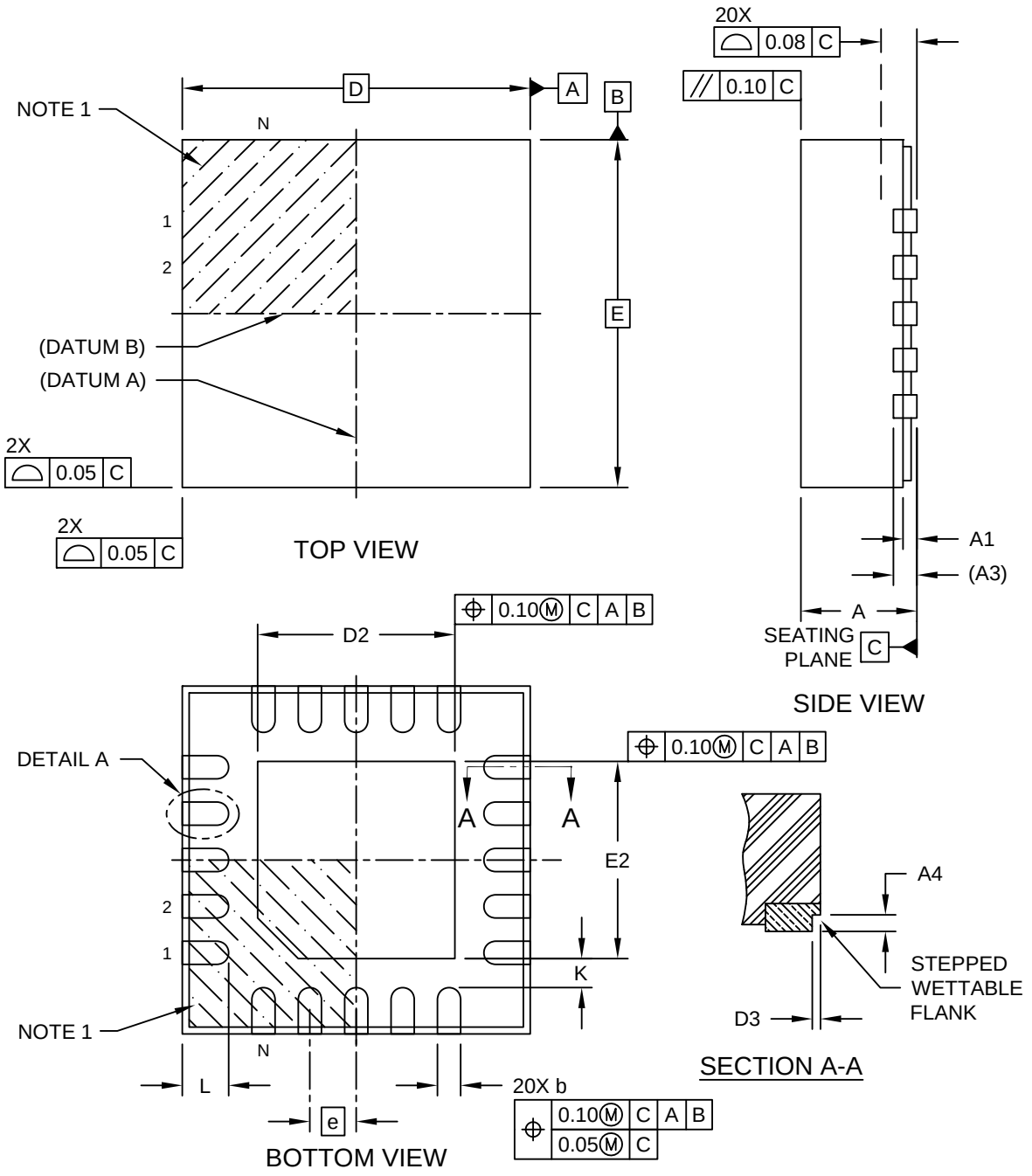
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2087-ST Rev F

38.5. 20-Pin VQFN Wettable Flanks

20-Lead Very Thin Plastic Quad Flat, No Lead Package (2LX) - 3x3 mm Body [VQFN] With 1.7 mm Exposed Pad and Stepped Wettable Flanks

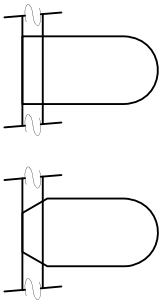
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



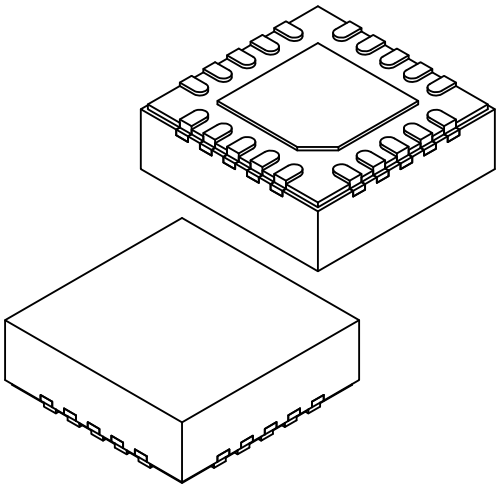
Microchip Technology Drawing C04-476-2LX Rev C Sheet 1 of 2

20-Lead Very Thin Plastic Quad Flat, No Lead Package (2LX) - 3x3 mm Body [VQFN] With 1.7 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



DETAIL 1
ALTERNATE TERMINAL CONFIGURATIONS



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		20	
Pitch	e		0.40 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		3.00 BSC	
Exposed Pad Length	D2	1.60	1.70	1.80
Overall Width	E		3.00 BSC	
Exposed Pad Width	E2	1.60	1.70	1.80
Terminal Width	b	0.15	0.20	0.25
Terminal Length	L	0.35	0.40	0.45
Terminal-to-Exposed-Pad	K	0.20	-	-
Wettable Flank Step Length	D3	-	-	0.085
Wettable Flank Step Height	A4	0.10	-	0.19

Dimensions D3 and A4 above apply to all new products released after November 1, and all products shipped after January 1, 2019, and supersede dimensions D3 and A4 below.
No physical changes are being made to any package; this update is to align cosmetic and tolerance variations from existing suppliers.

Wettable Flank Step Length	D3	0.035	0.06	0.085
Wettable Flank Step Height	A4	0.10	-	0.19

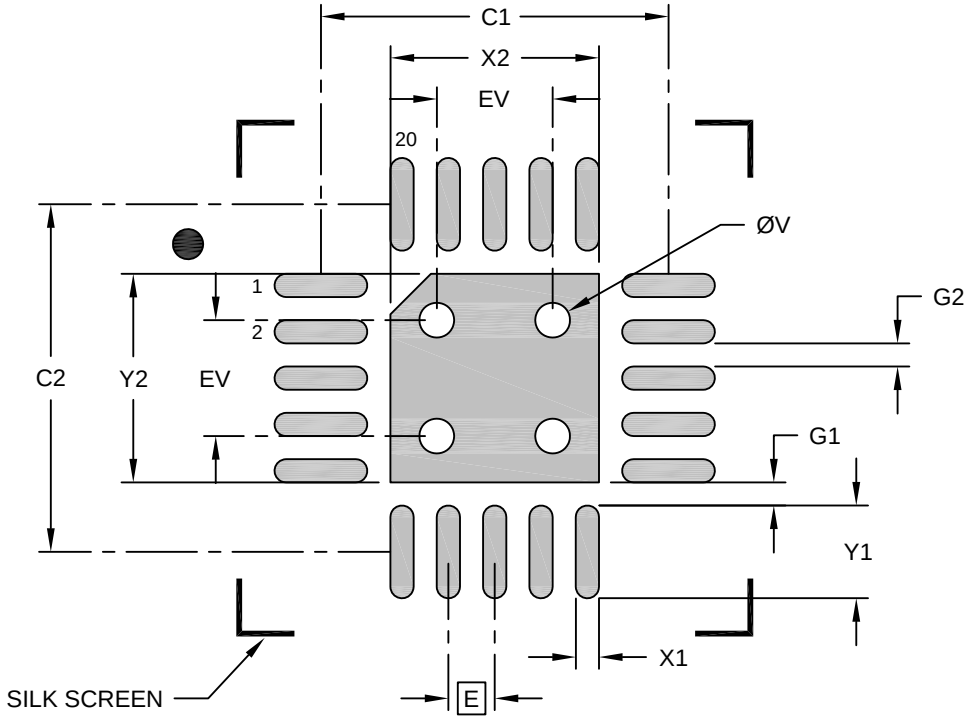
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-476-2LX Rev C Sheet 2 of 2

20-Lead Very Thin Plastic Quad Flat, No Lead Package (2LX) - 3x3 mm Body [VQFN] With 1.7 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.40 BSC	
Optional Center Pad Width	X2			1.80
Optional Center Pad Length	Y2			1.80
Contact Pad Spacing	C1		3.00	
Contact Pad Spacing	C2		3.00	
Contact Pad Width (X20)	X1			0.20
Contact Pad Length (X20)	Y1			0.80
Contact Pad to Center Pad (X20)	G1	0.20		
Contact Pad to Contact Pad (X16)	G2	0.20		
Thermal Via Diameter	V		0.30	
Thermal Via Pitch	EV		1.00	

Notes:

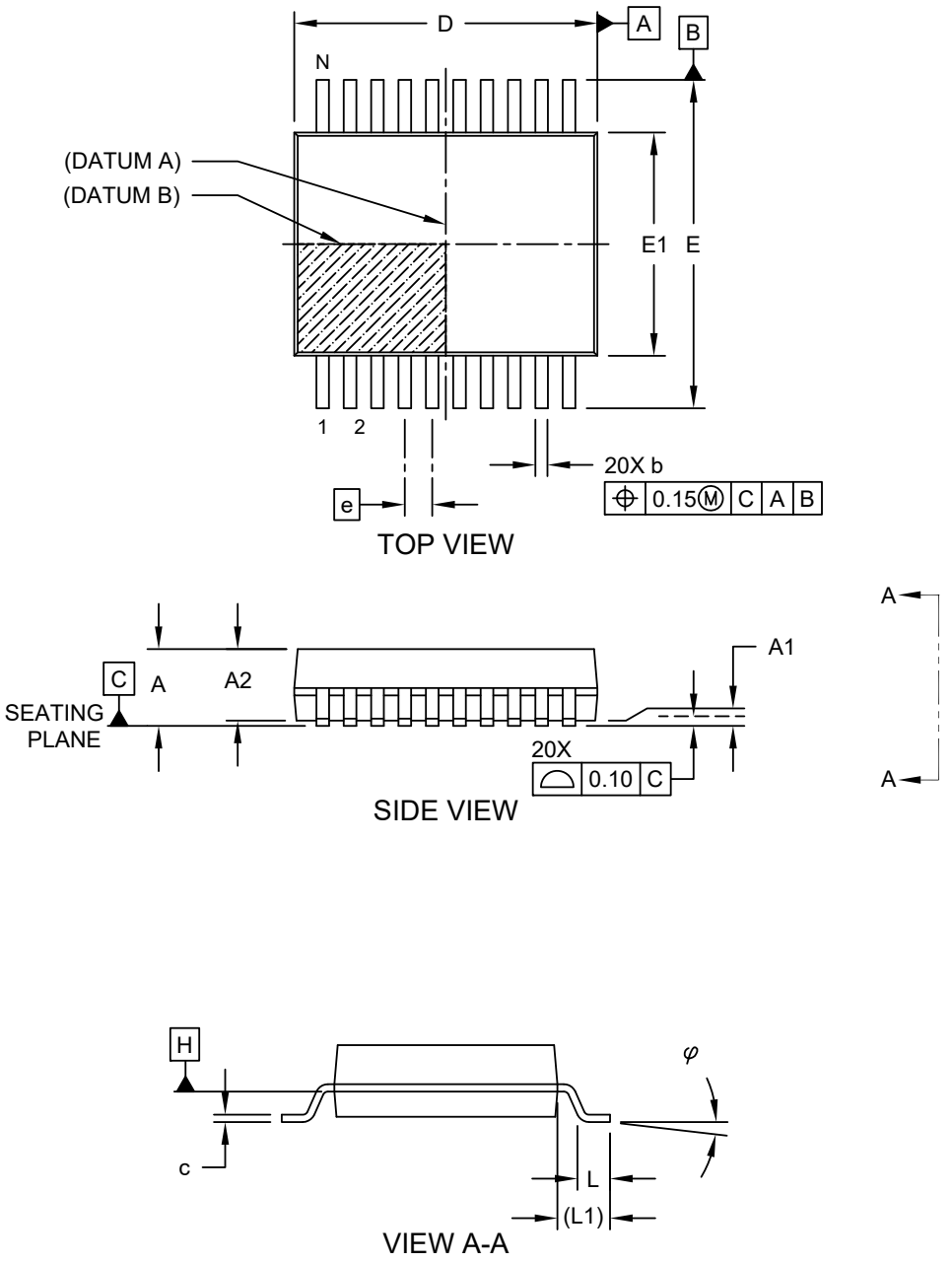
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2476-2LX Rev C

38.6. 20-Pin SSOP

20-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

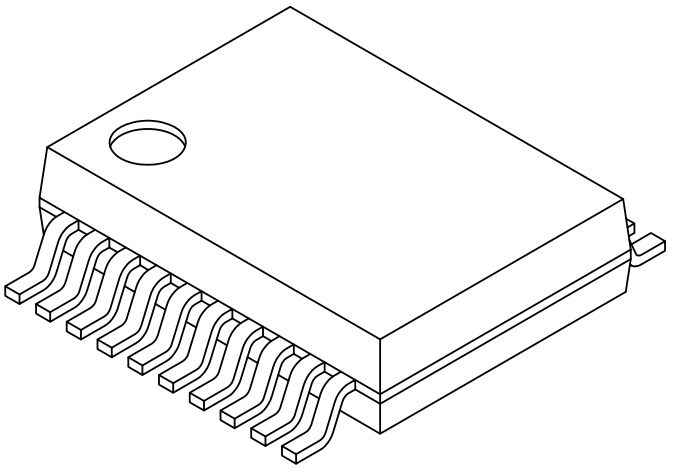
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-072 Rev C Sheet 1 of 2

20-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		20	
Pitch	e		0.65 BSC	
Overall Height	A	-	-	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	-	-
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	6.90	7.20	7.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1		1.25 REF	
Lead Thickness	c	0.09	-	0.25
Foot Angle	φ	0°	4°	8°
Lead Width	b	0.22	-	0.38

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M

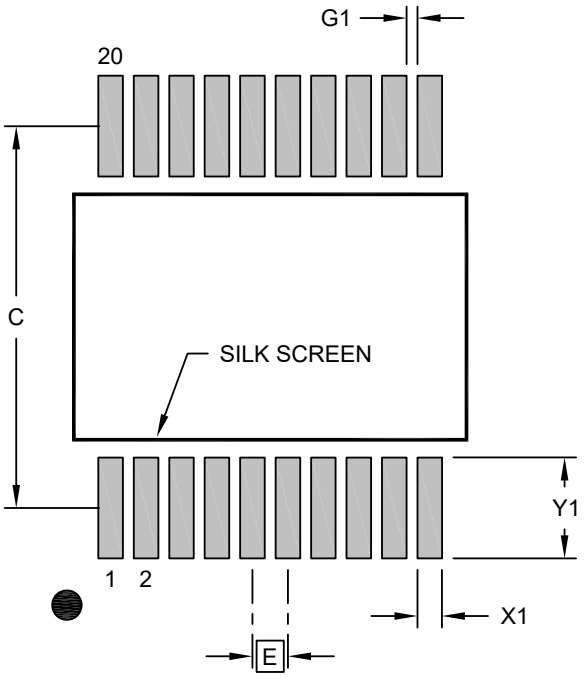
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-072 Rev C Sheet 2 of 2

20-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Contact Pad Spacing	C		7.00	
Contact Pad Width (X20)	X1			0.45
Contact Pad Length (X20)	Y1			1.85
Contact Pad to Center Pad (X18)	G1	0.20		

Notes:

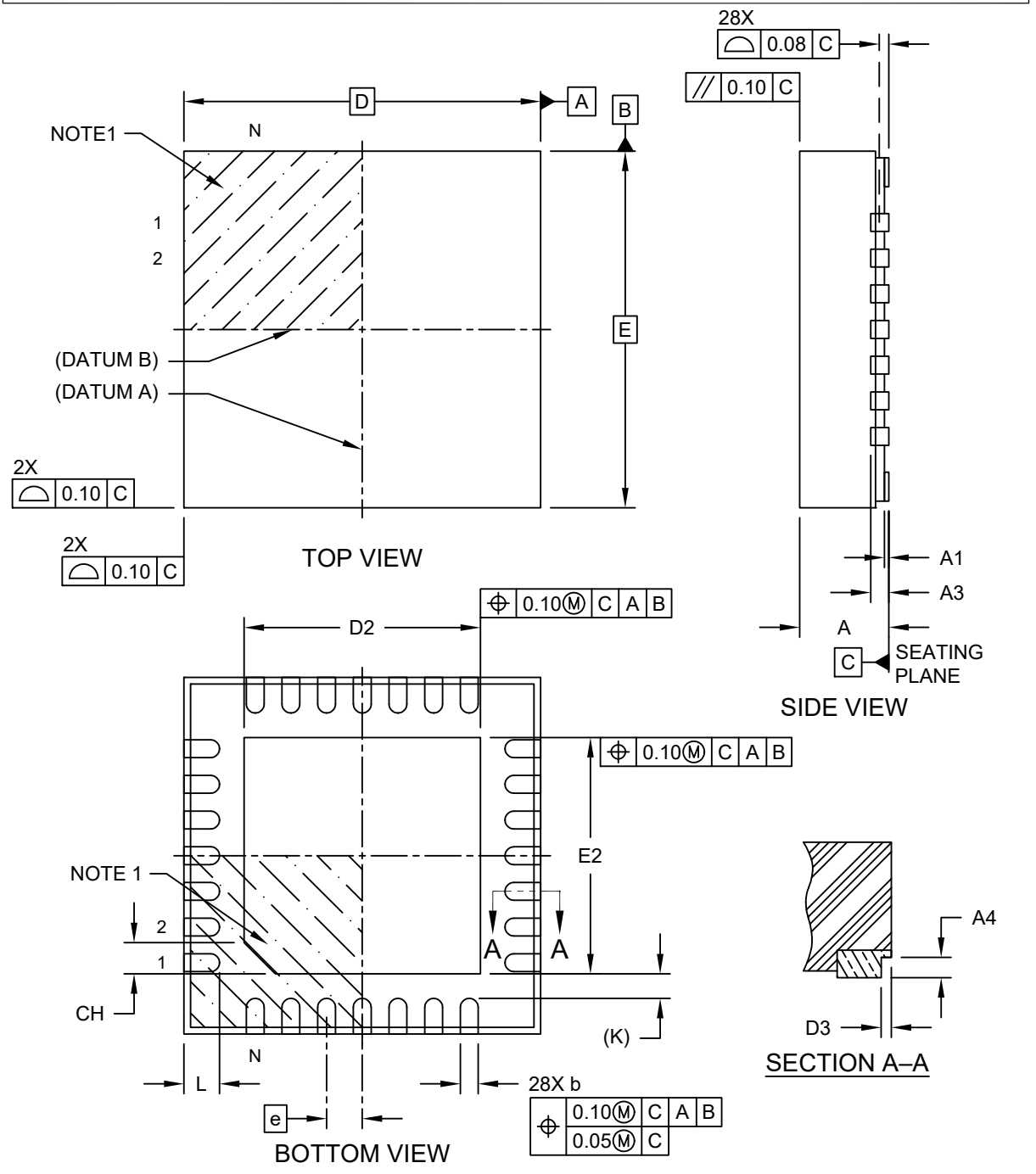
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2072 Rev C

38.7. 28-Pin VQFN Wettable Flanks

28-Lead Very Thin Plastic Quad Flat, No Lead Package (3LW) – 4x4x1.0 mm Body [VQFN] With 2.65 mm Exposed Pad and Stepped Wettable Flanks

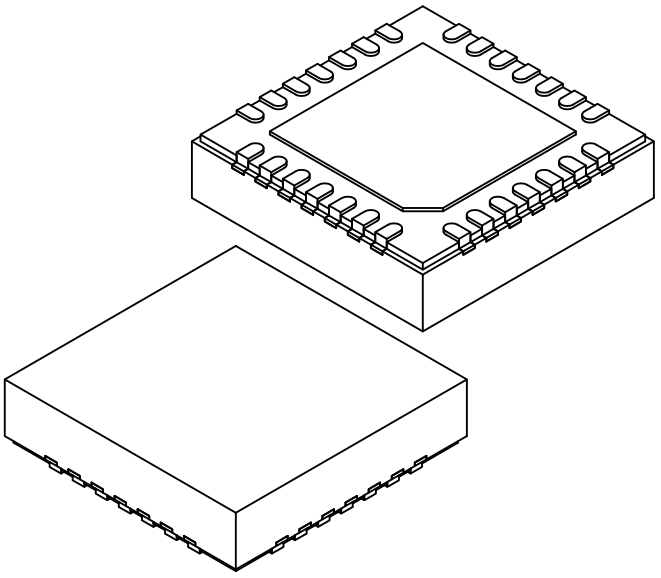
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-565-3LW Rev B Sheet 1 of 2

28-Lead Very Thin Plastic Quad Flat, No Lead Package (3LW) – 4x4x1.0 mm Body [VQFN] With 2.65 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		28	
Pitch	e		0.40 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		4.00 BSC	
Exposed Pad Length	D2	2.55	2.65	2.75
Overall Width	E		4.00 BSC	
Exposed Pad Width	E2	2.55	2.65	2.75
Exposed Pad Index Chamfer	CH		0.35 REF	
Terminal Width	b	0.15	0.20	0.25
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K		0.275 REF	
Wettable Flank Step Cut Length	D3	–	–	0.085
Wettable Flank Step Cut Height	A4	0.10	–	0.19

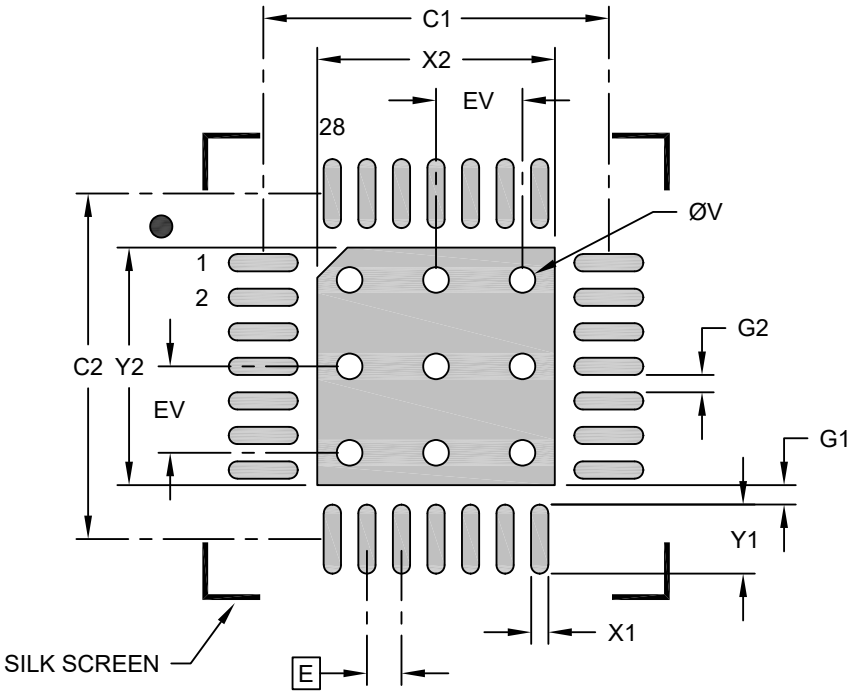
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-565-3LW Rev B Sheet 2 of 2

28-Lead Very Thin Plastic Quad Flat, No Lead Package (3LW) – 4x4x1.0 mm Body [VQFN] With 2.65 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.40 BSC	
Center Pad Width	X2			2.75
Center Pad Length	Y2			2.75
Contact Pad Spacing	C1		4.00	
Contact Pad Spacing	C2		4.00	
Contact Pad Width (X28)	X1			0.20
Contact Pad Length (X28)	Y1			0.80
Contact Pad to Center Pad (X28)	G1	0.23		
Contact Pad to Contact Pad (X24)	G2	0.20		
Thermal Via Diameter	V		0.30	
Thermal Via Pitch	EV		1.00	

Notes:

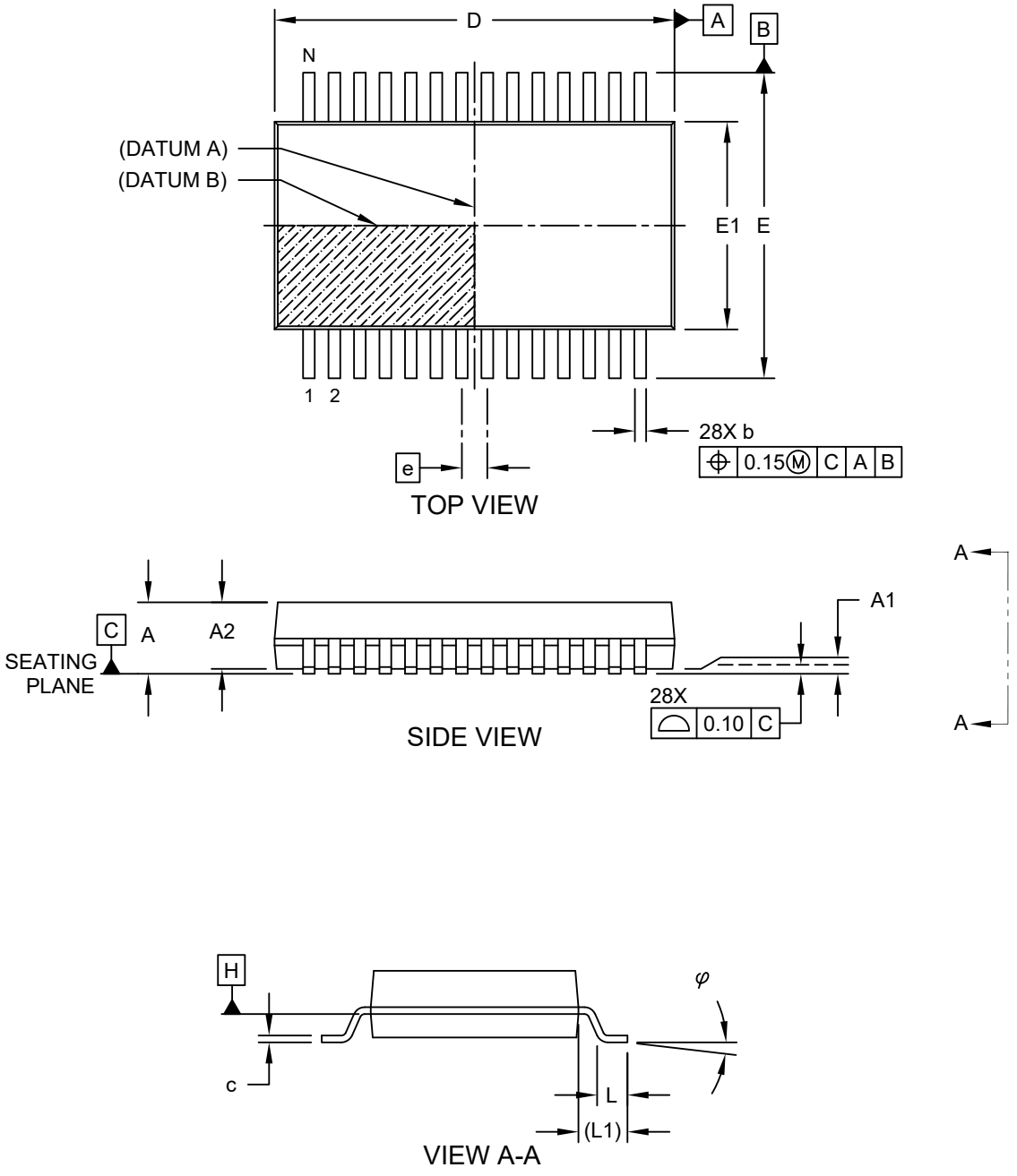
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2565-3LW Rev B

38.8. 28-Pin SSOP

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

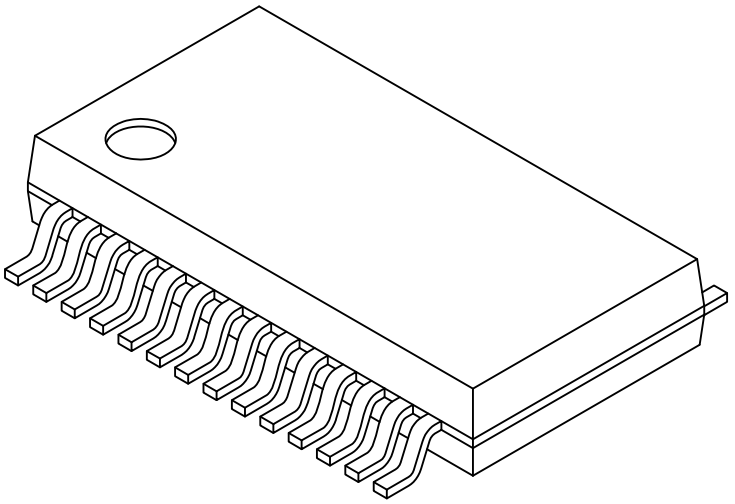
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-073 Rev C Sheet 1 of 2

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	-	-	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	-	-
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1		1.25 REF	
Lead Thickness	c	0.09	-	0.25
Foot Angle	φ	0°	4°	8°
Lead Width	b	0.22	-	0.38

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M

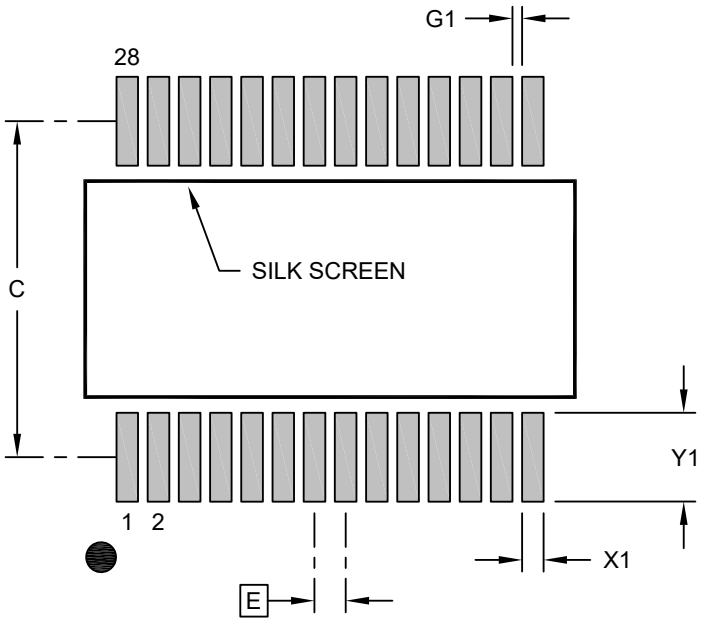
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073 Rev C Sheet 2 of 2

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Contact Pad Spacing	C		7.00	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.85
Contact Pad to Center Pad (X26)	G1	0.20		

Notes:

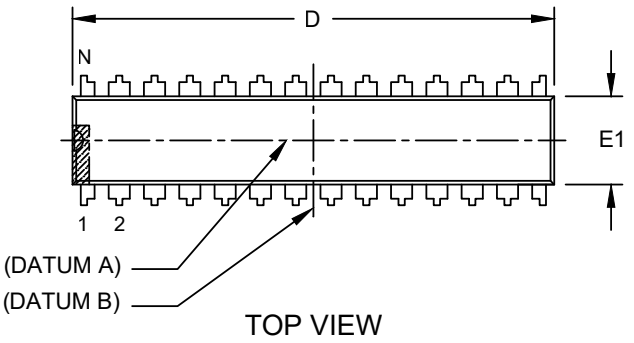
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-2073 Rev B

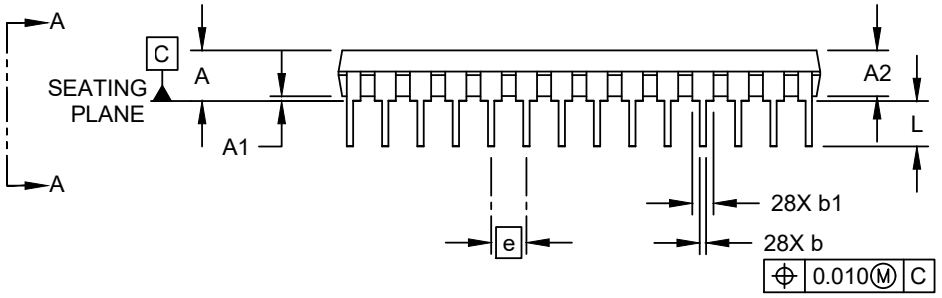
38.9. 28-Pin SPDIP

28-Lead Skinny Plastic Dual In-Line (SP) - 300 mil Body [SPDIP]

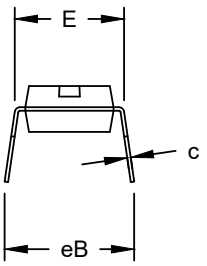
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW



SIDE VIEW

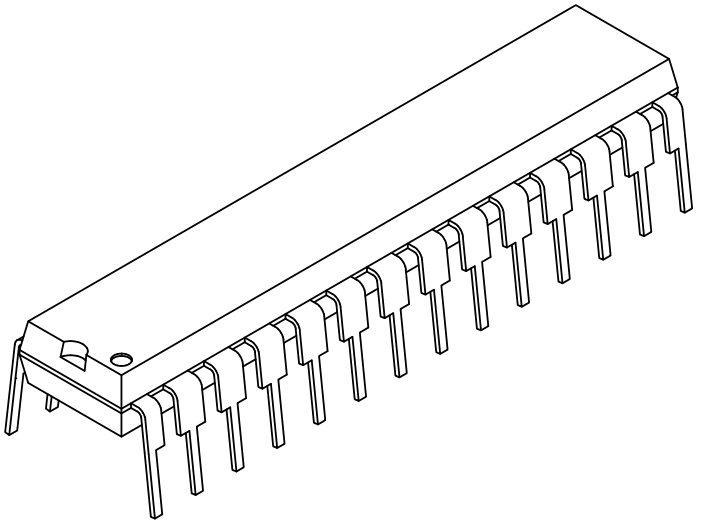


VIEW A-A

Microchip Technology Drawing C04-00070 Rev D (SP) Sheet 1 of 2

28-Lead Skinny Plastic Dual In-Line (SP) - 300 mil Body [SPDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		INCHES			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	N		28			28	
Pitch	e		.100 BSC			2.54 BSC	
Top to Seating Plane	A	-	-	.200	-	-	5.08
Molded Package Thickness	A2	.120	.135	.150	3.05	3.43	3.81
Base to Seating Plane	A1	.015	-	-	0.38	-	-
Shoulder to Shoulder Width	E	.290	.310	.335	7.37	7.87	8.51
Molded Package Width	E1	.240	.285	.295	6.10	7.24	7.49
Overall Length	D	1.345	1.365	1.400	34.16	34.67	35.56
Tip to Seating Plane	L	.110	.130	.150	2.79	3.30	3.81
Lead Thickness	c	.008	.010	.015	0.203	0.254	0.381
Upper Lead Width	b1	.045	.050	.070	1.14	1.27	1.78
Lower Lead Width	b	.014	.018	.022	0.36	0.46	0.56
Overall Row Spacing §	eB	-	-	.430	-	-	10.92

Notes:

1. The Pin 1 visual index feature may vary, but it must be located within the hatched area.
2. § Significant Characteristic
3. Dimensions D and E do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .005" per side.
4. Dimensioning and tolerancing per ASME Y14.5M

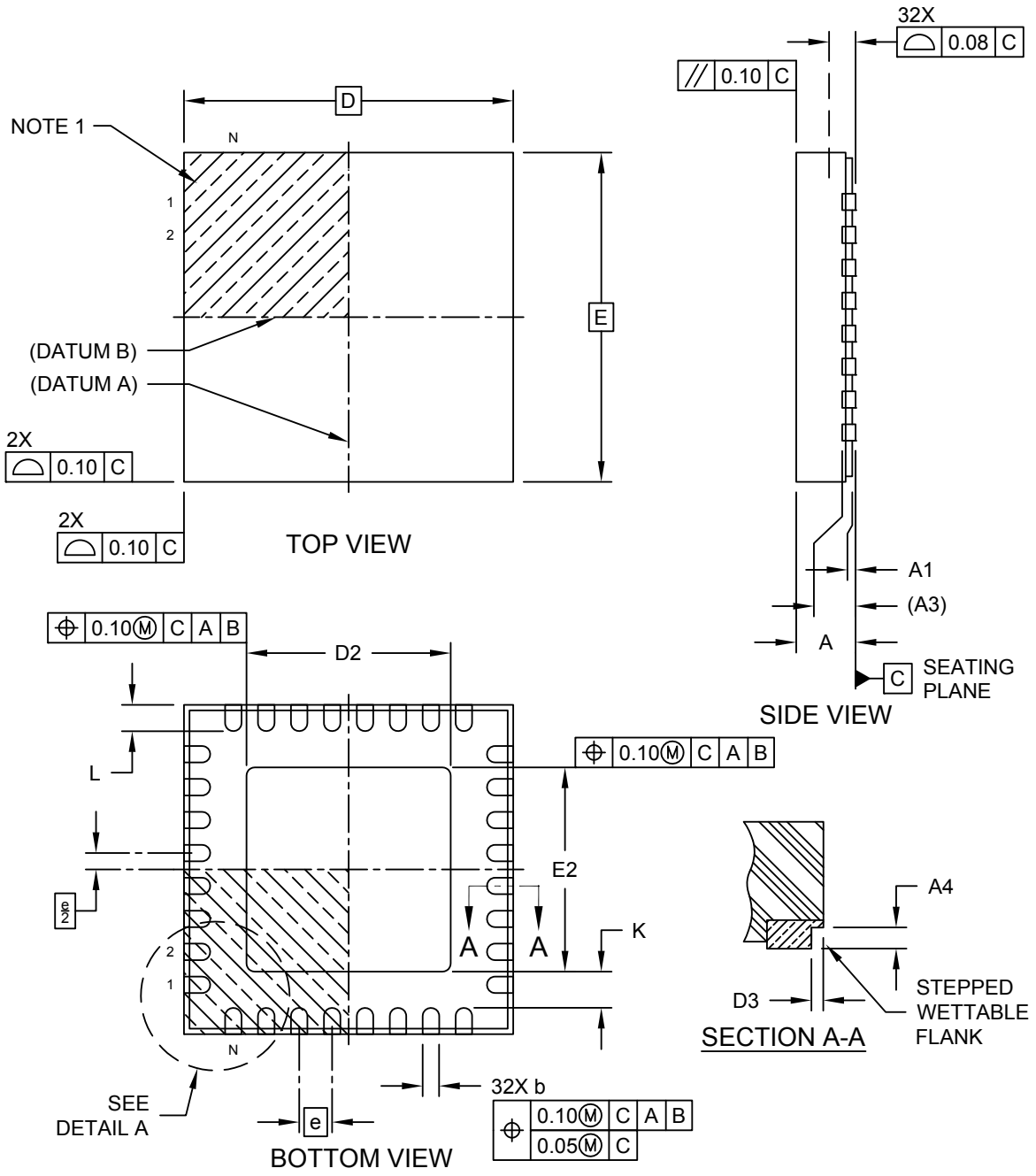
BSC: Basic Dimension. The theoretically exact value is shown without tolerances.

Microchip Technology Drawing C04-00070 Rev D (SP) Sheet 2 of 2

38.10. 32-Pin VQFN Wettable Flanks

32-Lead Ultra Thin Plastic Quad Flat, No Lead Package (QZB) - 5x5x0.9 mm Body [VQFN] With 3.1 mm Exposed Pad and Stepped Wettable Flanks

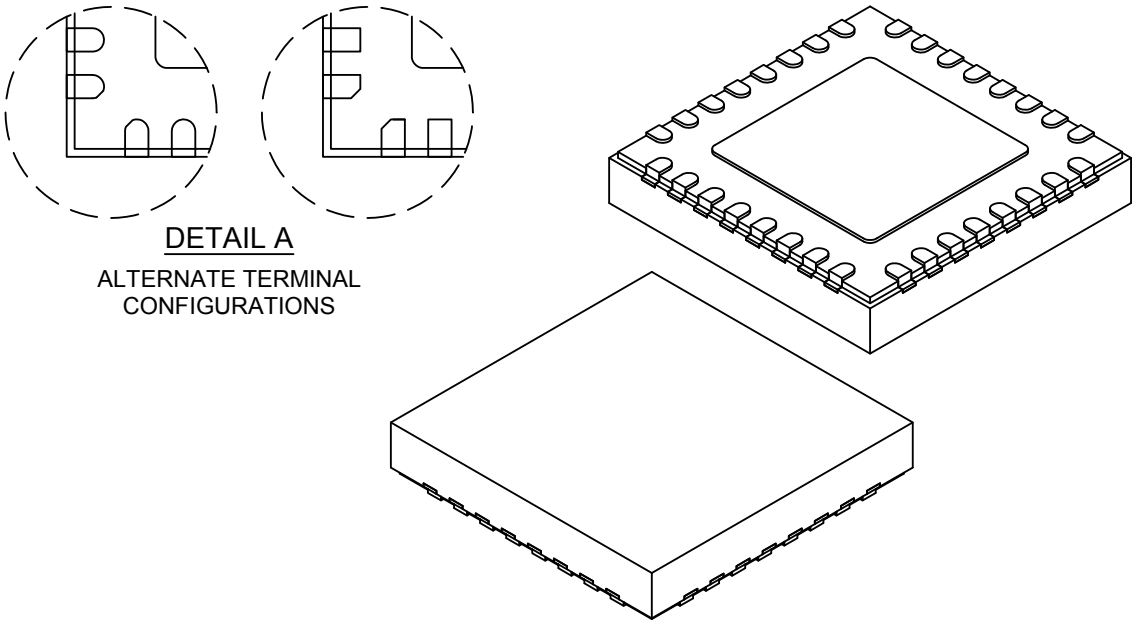
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-21511 Rev A Sheet 1 of 2

32-Lead Ultra Thin Plastic Quad Flat, No Lead Package (QZB) - 5x5x0.9 mm Body [VQFN] With 3.1 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



DETAIL A

ALTERNATE TERMINAL CONFIGURATIONS

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Terminals	N		32	
Pitch	e		0.50 BSC	
Overall Height	A	0.80	0.85	0.90
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.203 REF	
Overall Length	D		5.00 BSC	
Exposed Pad Length	D2	3.00	3.10	3.20
Overall Width	E		5.00 BSC	
Exposed Pad Width	E2	3.00	3.10	3.20
Terminal Width	b	0.18	0.25	0.30
Terminal Length	L	0.30	0.40	0.50
Terminal-to-Exposed-Pad	K	0.20	-	-
Wettable Flank Step Length	D3	-	-	0.085
Wettable Flank Step Height	A4	0.10	-	0.19

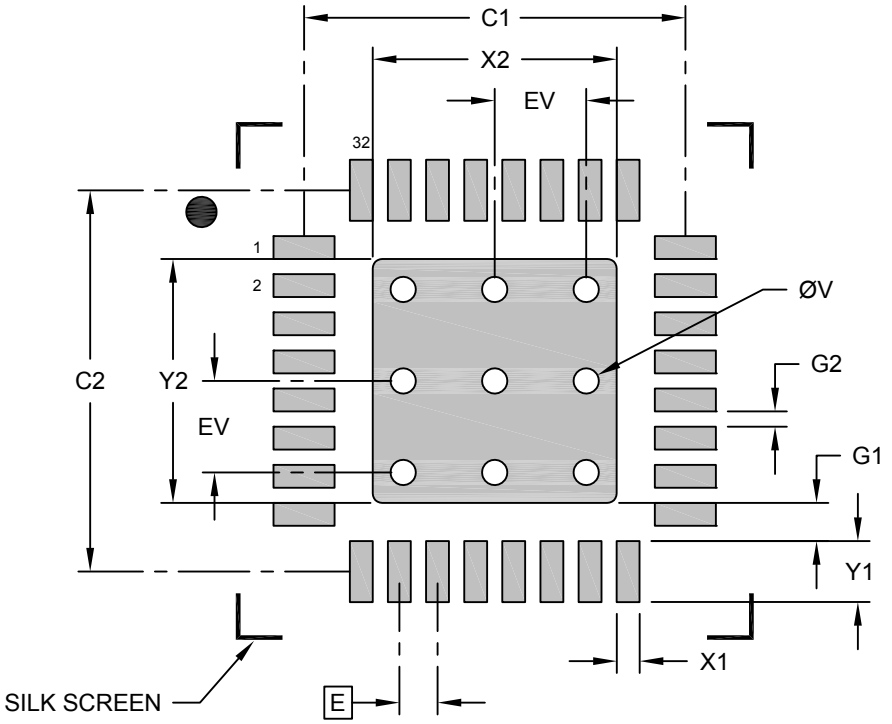
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21511 Rev A Sheet 1 of 2

32-Lead Ultra Thin Plastic Quad Flat, No Lead Package (QZB) - 5x5x0.9 mm Body [VQFN] With 3.1 mm Exposed Pad and Stepped Wettable Flanks

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Center Pad Width	X2			3.20
Center Pad Length	Y2			3.20
Contact Pad Spacing	C1		5.00	
Contact Pad Spacing	C2		5.00	
Contact Pad Width (X32)	X1			0.30
Contact Pad Length (X32)	Y1			0.80
Contact Pad to Center Pad (X32)	G1	0.20		
Contact Pad to Contactr Pad (X28)	G2	0.20		
Thermal Via Diameter	V		0.33	
Thermal Via Pitch	EV		1.20	

Notes:

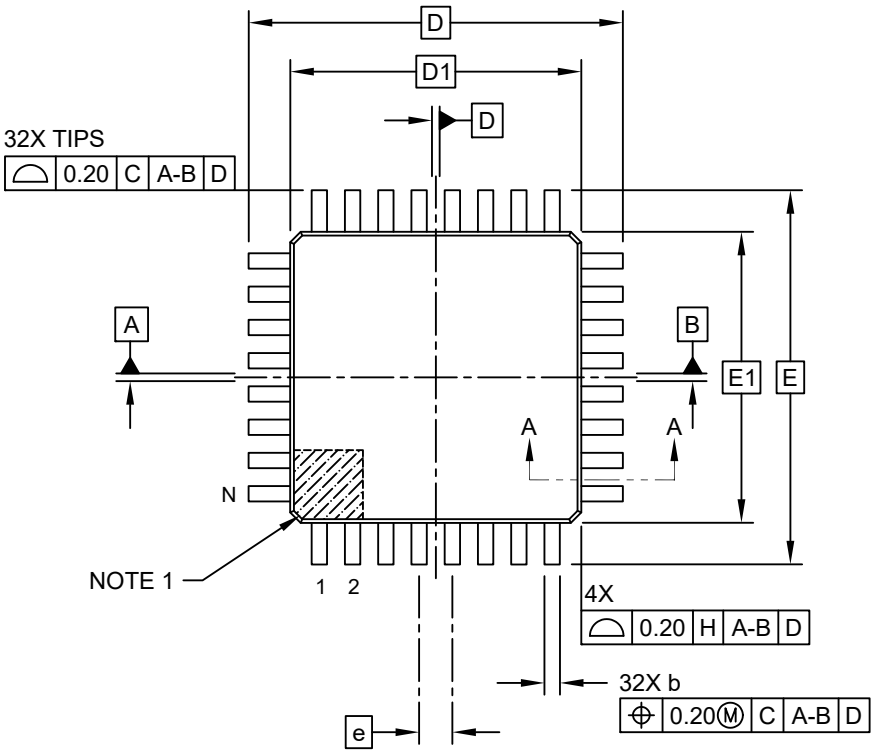
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
2. For best soldering results, thermal vias, if used, should be filled or tented to avoid solder loss during reflow process

Microchip Technology Drawing C04-23511 Rev A

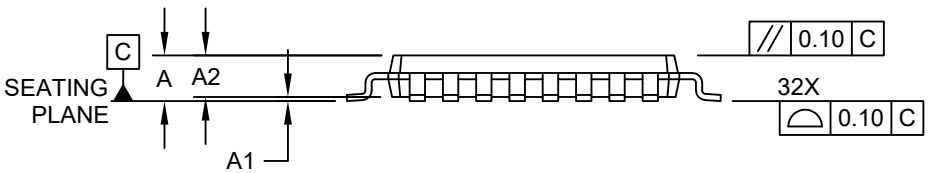
38.11. 32-Pin TQFP

32-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]
2.00 mm Footprint; Also Atmel Legacy Global Package Code AUT

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

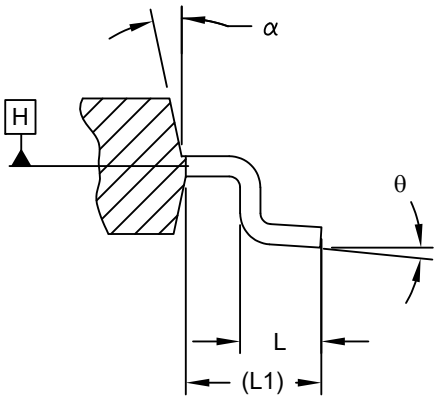


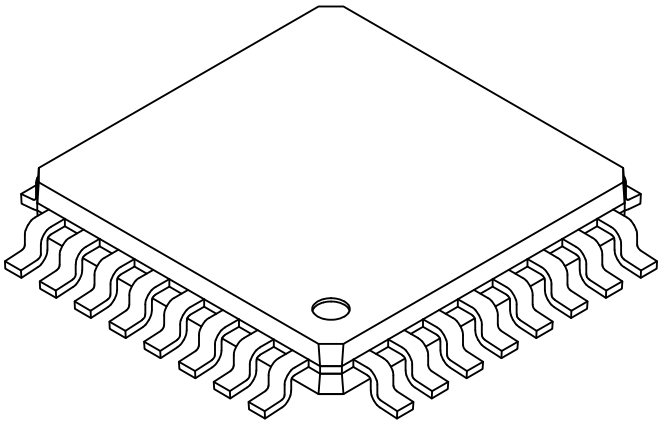
SIDE VIEW

Microchip Technology Drawing C04-074-PT Rev D Sheet 1 of 2

32-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP] 2.00 mm Footprint; Also Atmel Legacy Global Package Code AUT

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging





SECTION A-A

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		32	
Lead Pitch	e		0.80 BSC	
Overall Height	A	-	-	1.20
Standoff	A1	0.05	-	0.15
Molded Package Thickness	A2	0.95	1.00	1.05
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	θ	0°	-	7°
Overall Width	E		9.00 BSC	
Overall Length	D		9.00 BSC	
Molded Package Width	E1		7.00 BSC	
Molded Package Length	D1		7.00 BSC	
Lead Width	b	0.30	0.37	0.45
Mold Draft Angle Top	α	11°	-	13°

Notes:

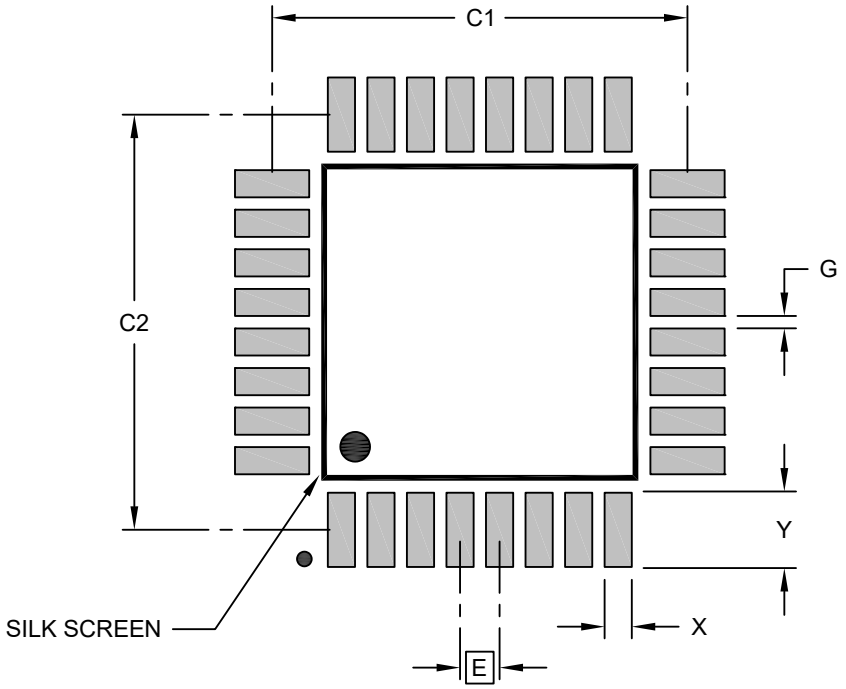
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-074-PT Rev D Sheet 2 of 2

32-Lead Plastic Thin Quad Flatpack (PT) - 7x7x1.0 mm Body [TQFP]
2.00 mm Footprint; Also Atmel Legacy Global Package Code AUT

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.80 BSC	
Contact Pad Spacing	C1		8.40	
Contact Pad Spacing	C2		8.40	
Contact Pad Width (X32)	X			0.55
Contact Pad Length (X32)	Y			1.55
Contact Pad to Contact Pad (X28)	G	0.25		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2074-PT Rev D

39. 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).

39.1. Revision History

Doc. Rev.	Date	Comments
A	05/2026	Initial release of preliminary data sheet

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 979-8-3371-3256-3

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Product Page Links

AVR16LA14, AVR16LA20, AVR16LA28, AVR16LA32