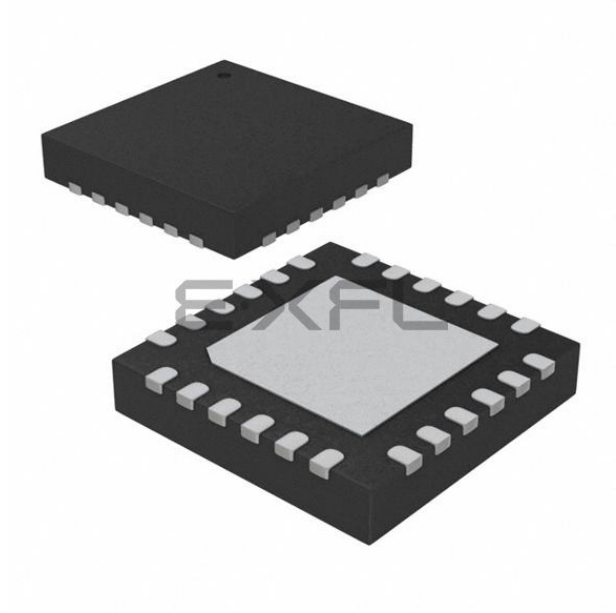


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	22
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 24x10b; D/A 3x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	24-VFQFN Exposed Pad
Supplier Device Package	24-VQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny1617-mn

7. Peripherals and Architecture

7.1 Peripheral Module Address Map

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

Table 7-1. Peripheral Module Address Map

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

8.5.7.2 Sequence for Execution of Self-Programming

9.4 Register Summary - NVMCTRL

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

9.5 Register Description

14.5.3 Asynchronous Channel n Generator Selection

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

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

Bits 7:0 – ASYNCCH[7:0] Asynchronous Channel Generator Selection

Table 14-2. Asynchronous Channel Generator Selection

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

17.5.4 Interrupt Control

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

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

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

Value	Description
0x0	Voltage crosses VLM threshold from above
0x1	Voltage crosses VLM threshold from below
0x2	Either direction is triggering an interrupt request
Other	Reserved

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

19. WDT - Watchdog Timer

19.1 Features

- Issues a System Reset if the Watchdog Timer is not Cleared Before its Time-out Period
- Operating Asynchronously from System Clock Using an Independent Oscillator
- Using the 1 KHz Output of the 32 KHz Ultra Low-Power Oscillator (OSCULP32K)
- 11 Selectable Time-out Periods, from 8 ms to 8s
- Two Operation modes:
  - Normal mode
  - Window mode
- Configuration Lock to Prevent Unwanted Changes
- Closed Period Timer Activation After First WDT Instruction for Easy Setup

19.2 Overview

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

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

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

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

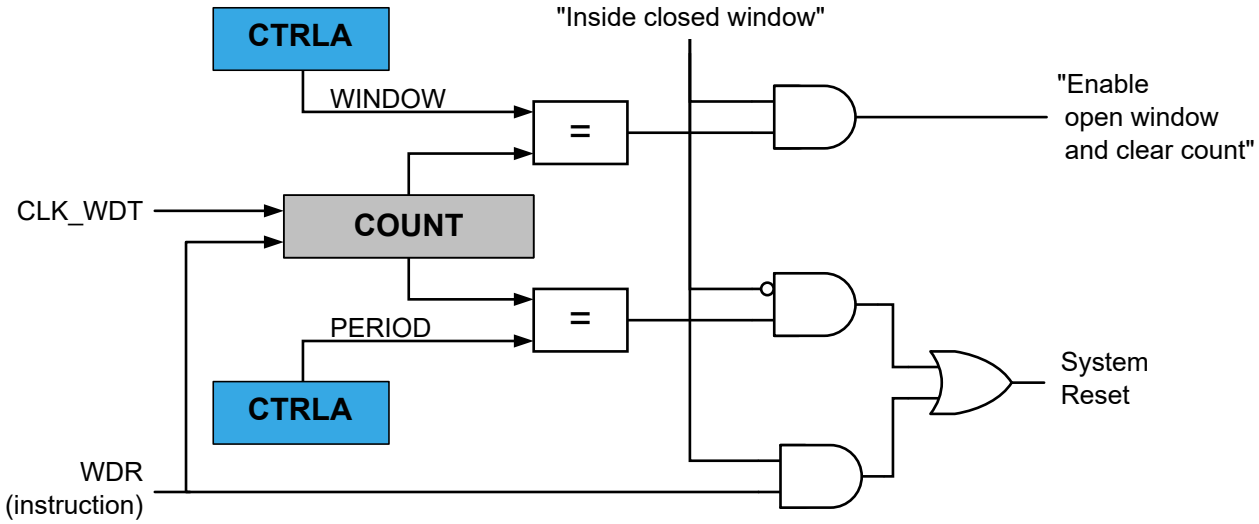
The CCP mechanism ensures that the WDT settings cannot be changed by accident. For increased safety, a configuration for locking the WDT settings is available.

Related Links

8.5.7 Configuration Change Protection (CCP)

19.2.1 Block Diagram

Figure 19-1. WDT Block Diagram



19.2.2 Signal Description

Not applicable.

19.2.3 System Dependencies

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

Table 19-1. WDT System Dependencies

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

Related Links

19.2.3.1 Clocks

19.2.3.5 Debug Operation

19.2.3.1 Clocks

A 1 KHz Oscillator Clock (CLK_WDT_OSC) is sourced from the internal Ultra Low-Power Oscillator, OSCULP32K. Due to the ultra low-power design, the oscillator is not very accurate, and so the exact time-out period may vary from device to device. This variation must be kept in mind when designing software that uses the WDT to ensure that the time-out periods used are valid for all devices.

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

19.2.3.2 I/O Lines and Connections

Not applicable.

20.3.2 Initialization

To start using the timer/counter in a basic mode, follow these steps:

- Write a TOP value to the Period register (TCAn.PER)
- Enable the peripheral by writing a '1' to the ENABLE bit in the Control A register (TCAn.CTRLA). The counter will start counting clock ticks according to the prescaler setting in the Clock Select bit field (CLKSEL) in TCAn.CTRLA.
- Optional: By writing a '1' to the Enable Count on Event Input bit (CNTEI) in the Event Control register (TCAn.EVCTRL), event inputs are counted instead of clock ticks.
- The counter value can be read from the Counter bit field (CNT) in the Counter register (TCAn.CNT).

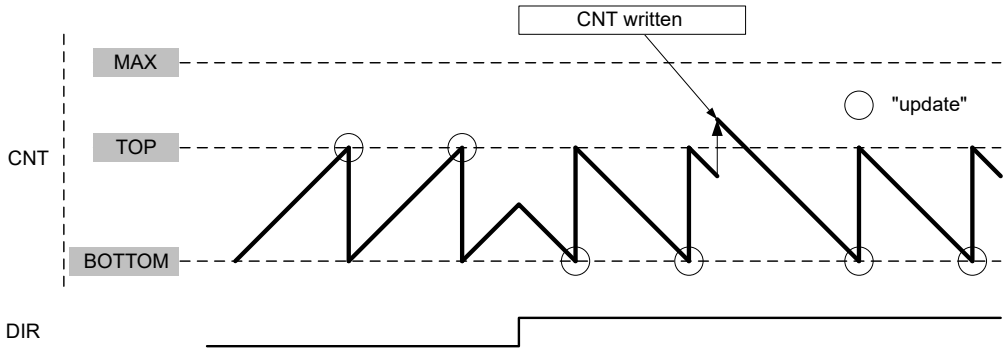
20.3.3 Operation

20.3.3.1 Normal Operation

In normal operation, the counter is counting clock ticks in the direction selected by the Direction bit (DIR) in the Control E register (TCAn.CTRLE), until it reaches TOP or BOTTOM. The clock ticks are from the peripheral clock CLK_PER, optionally prescaled, depending on the Clock Select bit field (CLKSEL) in the Control A register (TCAn.CTRLA).

When up-counting and TOP are reached, the counter will wrap to zero at the next clock tick. When down-counting, the counter is reloaded with the Period register value (TCAn.PER) when BOTTOM is reached.

Figure 20-4. Normal Operation



It is possible to change the counter value in the Counter register (TCAn.CNT) when the counter is running. The write access to TCAn.CNT has higher priority than count, clear, or reload, and will be immediate. The direction of the counter can also be changed during normal operation by writing to DIR in TCAn.CTRLE.

20.3.3.2 Double Buffering

The Period register value (TCAn.PER) and the Compare n register values (TCAn.CMPn) are all double-buffered (TCAn.PERBUF and TCAn.CMPnBUF).

Each buffer register has a Buffer Valid flag (PERBV, CMPnBV) in the Control F register (TCAn.CTRLF), which indicates that the buffer register contains a valid, i.e. new, value that can be copied into the corresponding Period or Compare register. When the Period register and Compare n registers are used for a compare operation, the BV flag is set when data is written to the buffer register and cleared on an UPDATE condition. This is shown for a Compare register (CMPn) below.

20.5.14 Counter Register - Normal Mode

Name: CNT
Offset: 0x20
Reset: 0x00
Property: -

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

CPU and UPDI write access has priority over internal updates of the register.

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

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

Bits 15:8 – CNT[15:8] Counter High Byte
These bits hold the MSB of the 16-bit counter register.

Bits 7:0 – CNT[7:0] Counter Low Byte
These bits hold the LSB of the 16-bit counter register.

20.7.13 High Byte Period Register - Split Mode

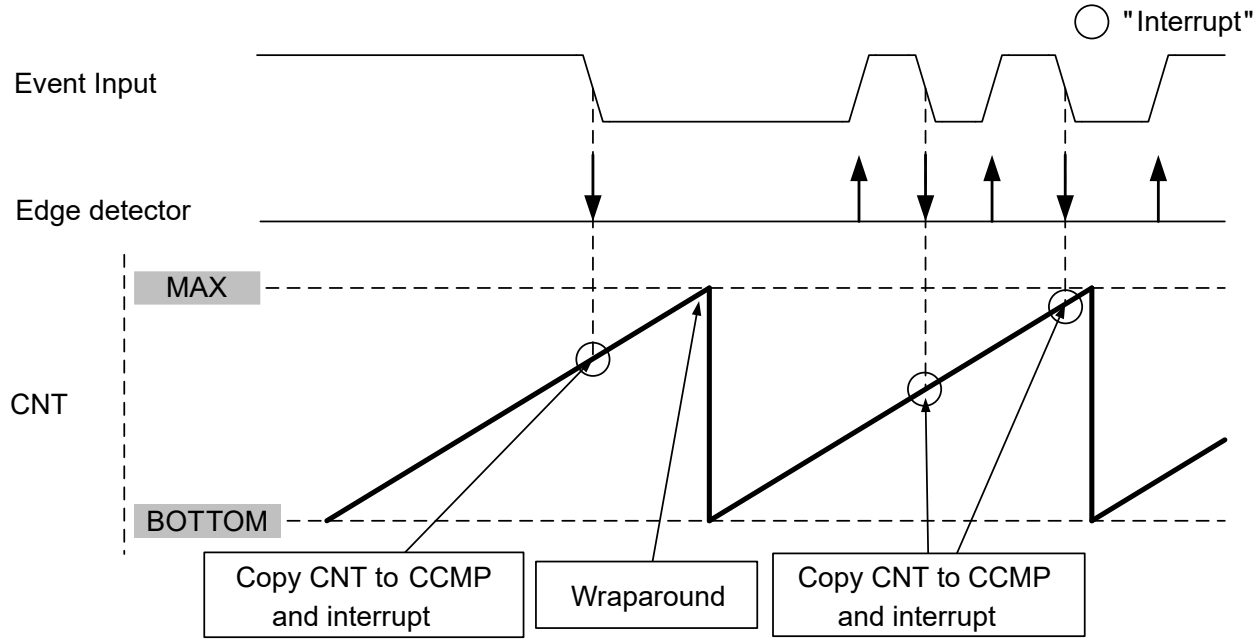
Name: HPER
Offset: 0x27
Reset: 0x00
Property: -

The TCAn.HPER register contains the TOP value of high byte timer.

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

Bits 7:0 – HPER[7:0] Period Value High Byte Timer
These bits hold the TOP value of high byte timer.

Figure 21-4. Input Capture on Event



It is recommended to write '0' to the TCBn.CNT register when entering this mode from any other mode.

21.3.3.1.4 Input Capture Frequency Measurement Mode

In this mode, the TCB captures the counter value and restarts on either a positive or negative edge of the event input signal.

The interrupt flag is automatically cleared after the high byte of the Compare/Capture register (TCBn.CCMP) has been read, and an interrupt request is generated.

The figure below illustrates this mode when configured to act on rising edge.

Figure 21-5. Input Capture Frequency Measurement

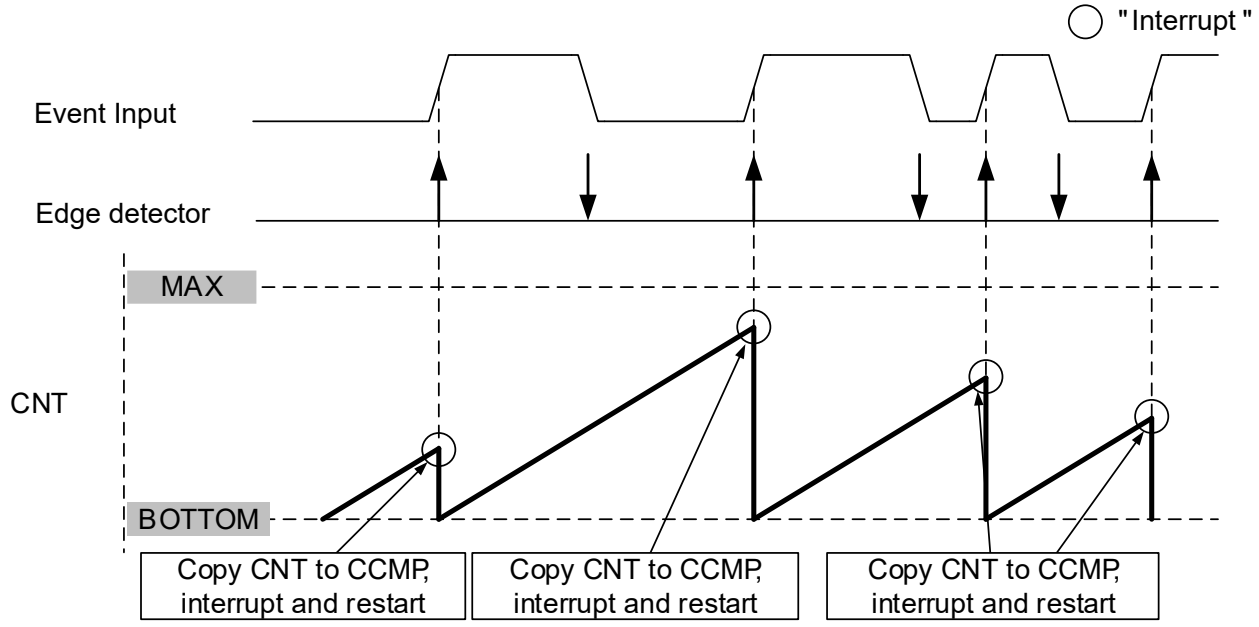
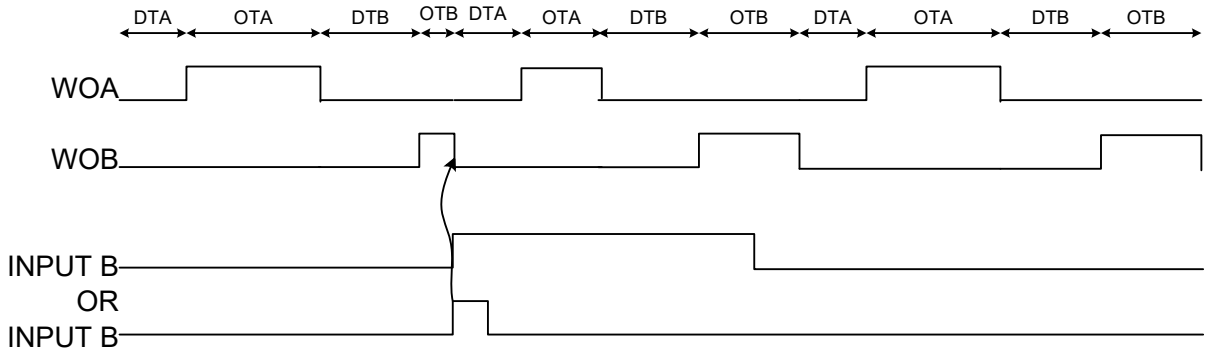


Figure 22-21. Input Mode 8 on Input B

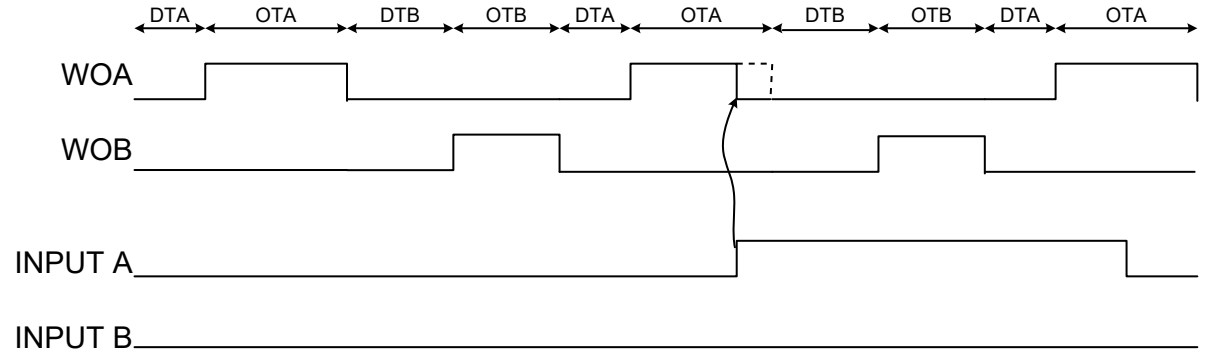


Input Mode 9: Stop Output at Level, Maintain Frequency

In Input mode 9 a positive edge on the input event while the corresponding output is ON will cause the output to stop during the rest of the on-time. The TCD counter will not be affected by the event, only the output.

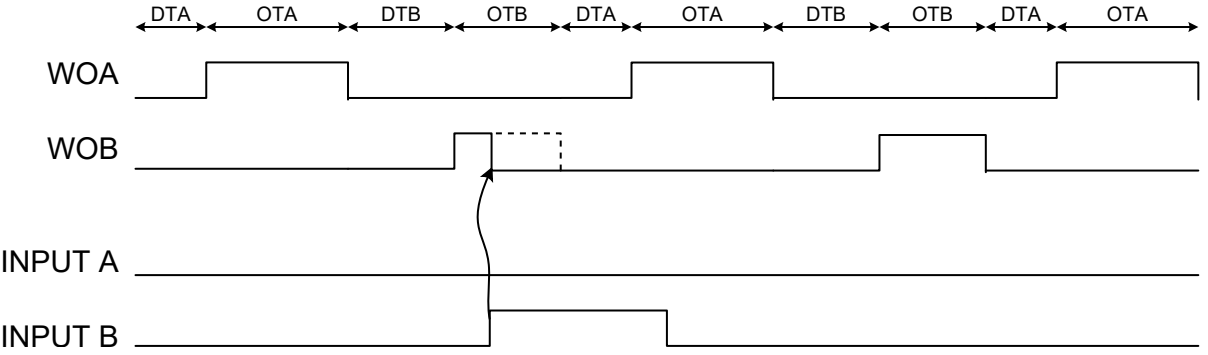
If Input mode 9 is used on input A and a positive input event occurs while in On-time A, the output will be OFF for the rest of the on-time.

Figure 22-22. Input Mode 9 on Input A



If Input mode 9 is used on input B and a positive input event occurs while in On-time B, the output will be OFF for the rest of the on-time.

Figure 22-23. Input Mode 9 on Input B



Input Mode 10: Stop Output on Edge, Maintain Frequency

In Input mode 10 the input event will cause the corresponding output to stop as long as the input is active. If the input goes low while there should have been an on-time on the corresponding output, the output will be deactivated for the rest of the on-time, too. The TCD counter is not affected by the event, only the output.

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.

Related Links

13. CPUINT - CPU Interrupt Controller

8.7.3 SREG

22.3.5 Sleep Mode Operation

The TCD operates in Idle Sleep mode and is stopped when entering Standby and Power-Down Sleep modes.

22.3.6 Synchronization

The TCD has two different clock domains and needs to synchronize the communication between the domains. See the Initialization section for details on how the synchronization of values from the I/O clock domain to the TCD clock domain is done. See the Capture section for details on how the synchronization of values from the TCD clock domain to the I/O clock domain is done.

Related Links

22.3.1 Initialization and Disabling

22.3.2.6 TCD Counter Capture

22.3.7 Configuration Change Protection

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

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

The following registers are under CCP:

Table 22-9. TCD - Registers under Configuration Change Protection

Register	Key
FAULTCTRL	IOREG

23. RTC - Real-Time Counter

23.1 Features

- 16-Bit Resolution
- Selectable Clock Source:
  - 32.768 kHz external crystal (XOSC32K)
  - External clock
  - 32 KHz internal ULP oscillator (OSCULP32K)
  - OSCULP32K divided by 32
- Programmable 15-Bit Clock Prescaling
- One Compare Register
- One Period Register
- Clear Timer On Period Overflow
- Optional Interrupt/Event on Overflow and Compare Match
- Periodic Interrupt and Event

23.2 Overview

The RTC peripheral offers two timing functions: the Real-Time Counter (RTC) and a Periodic Interrupt Timer (PIT).

The PIT functionality can be enabled independently of the RTC functionality.

RTC - Real-Time Counter

The RTC counts (prescaled) clock cycles in a Counter register, and compares the content of the Counter register to a Period register and a Compare register.

The RTC can generate both interrupts and events on compare match or overflow. It will generate a compare interrupt and/or event at the first count after the counter equals the Compare register value, and an overflow interrupt and/or event at the first count after the counter value equals the Period register value. The overflow will also 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 Ultra Low-Power Oscillator (OSCULP32K), or the OSCULP32K divided by 32.

The RTC peripheral includes a 15-bit programmable prescaler that can scale down the reference clock before it reaches the counter. A wide range of resolutions and time-out periods can be configured for the RTC. With a 32.768 kHz clock source, the maximum resolution is 30.5 µs, and time-out periods can be up to two seconds. With a resolution of 1s, the maximum time-out period is more than 18 hours (65536 seconds). The RTC can give a compare interrupt and/or event when the counter equals the compare register value, and an overflow interrupt and/or event when it equals the period register value.

23.11.6 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

27.2.2.4 Events

Not applicable.

27.2.2.5 Debug Operation

Whenever the debugger accesses the device, for instance, reading or writing a peripheral or memory location, the CRCSCAN peripheral will be disabled.

If the CRCSCAN is busy when the debugger accesses the device, the CRCSCAN will restart the ongoing operation when the debugger accesses an internal register or when the debugger disconnects.

The BUSY bit in the Status register (CRCSCAN.STATUS) will read '1' if the CRCSCAN was busy when the debugger caused it to disable, but it will not actively check any section as long as the debugger keeps it disabled. There are synchronized CRC Status bits in the debugger's internal register space, which can be read by the debugger without disabling the CRCSCAN. Reading the debugger's internal CRC status bits will make sure that the CRCSCAN is enabled.

It is possible to write the CRCSCAN.STATUS register directly from the debugger:

- BUSY bit in CRCSCAN.STATUS:
  - Writing the BUSY bit to '0' will stop the ongoing CRC operation (so that the CRCSCAN does not restart its operation when the debugger allows it).
  - Writing the BUSY bit to '1' will make the CRC start a single check with the settings in the Control B register (CRCSCAN.CTRLB), but not until the debugger allows it.

  As long as the BUSY bit in CRCSCAN.STATUS is '1', CRCSCAN.CRCTRLB and the Non-Maskable Interrupt Enable bit (NMIEN) in the Control A register (CRCSCAN.CTRLA) cannot be altered.
- OK bit in CRCSCAN.STATUS:
  - Writing the OK bit to '0' can trigger a Non-Maskable Interrupt (NMI) if the NMIEN bit in CRCSCAN.CTRLA is '1'. If an NMI has been triggered, no writes to the CRCSCAN are allowed.
  - Writing the OK bit to '1' will make the OK bit read as '1' when the BUSY bit in CRCSCAN.STATUS is '0'.

Writes to CRCSCAN.CTRLA and CRCSCAN.CTRLB from the debugger are treated in the same way as writes from the CPU.

Related Links

33. UPDI - Unified Program and Debug Interface
27.5.1 CTRLA
27.5.2 CTRLB

27.3 Functional Description

27.3.1 Initialization

To enable a CRC in software (or via the debugger):

1. Write the Source (SRC) bit field of the Control B register (CRCSCAN.CTRLB) to select the desired source settings. Ensure that the MODE bit field in CRCSCAN.CTRLB is 0x0.
2. Enable the CRCSCAN by writing a '1' to the ENABLE bit in the Control A register (CRCSCAN.CTRLA).
3. The CRC will start after three cycles, and the CPU will continue executing during these three cycles.

The output from an internal sequential module can be used as input source for the LUT, see the figure below for an example for LUT0 and LUT1. The sequential selection for each LUT follows the formula:

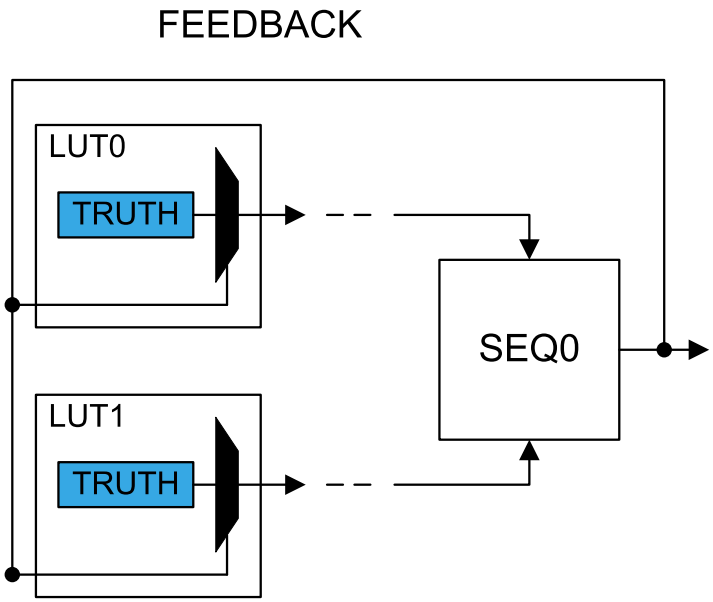
$\mathrm{IN}[2\mathrm{N}][i] = \mathrm{SEQ}[N]$

$\mathrm{IN}[2\mathrm{N}+1][i] = \mathrm{SEQ}[N]$

With *N* representing the sequencer number and *i*=0,1 representing the LUT input index.

For details, refer to 28.3.2.6 Sequential Logic.

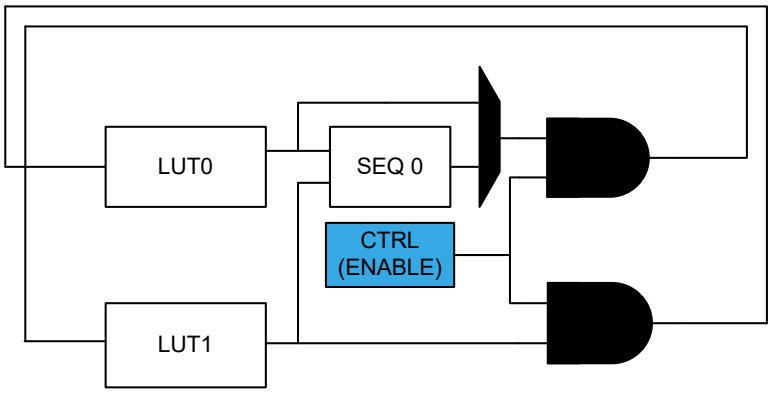
Figure 28-2. Feedback Input Selection



Linked LUT (LINK)

When selecting the LINK input option, the next LUT's direct output is used as the LUT input. In general, LUT[n+1] is linked to the input of LUT[n]. As example, LUT1 is the input for LUT0.

Figure 28-3. Linked LUT Input Selection



Internal Events Inputs Selection (EVENT)

Asynchronous events from the Event System can be used as input to the LUT. Two event input lines (EVENT0 and EVENT1) are available, and can be selected as LUT input. Before selecting the EVENT

- AINN1
- AINP0
- AINP1
- AINP2
- AINP3

29.3.2.2.2 Internal Inputs

Two internal inputs are available for the analog comparator:

- Output from the DAC
- DAC and AC voltage reference

29.3.2.3 Low-Power Mode

For power sensitive applications, the AC provides a Low-Power mode with reduced power consumption and increased propagation delay.

This mode is enabled by writing a '1' to the Low-Power Mode bit (LPMODE) in the Control A register (AC.CTRLA).

29.3.3 Events

The AC will generate the following event automatically when the AC is enabled:

- The digital output from the AC (OUT in the block diagram) is available as an Event System source. The events from the AC are asynchronous to any clocks in the device.

The AC has no event inputs.

29.3.4 Interrupts

Table 29-2. Available Interrupt Vectors and Sources

Offset	Name	Vector Description	Conditions
0x00	COMP0	Analog comparator interrupt	AC output is toggling as configured by INTMODE in AC.CTRLA

When an interrupt condition occurs, the corresponding interrupt flag is set in the STATUS register (AC.STATUS).

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

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 AC.STATUS register description for details on how to clear interrupt flags.

29.3.5 Sleep Mode Operation

In Idle Sleep mode, the AC will continue to operate as normal.

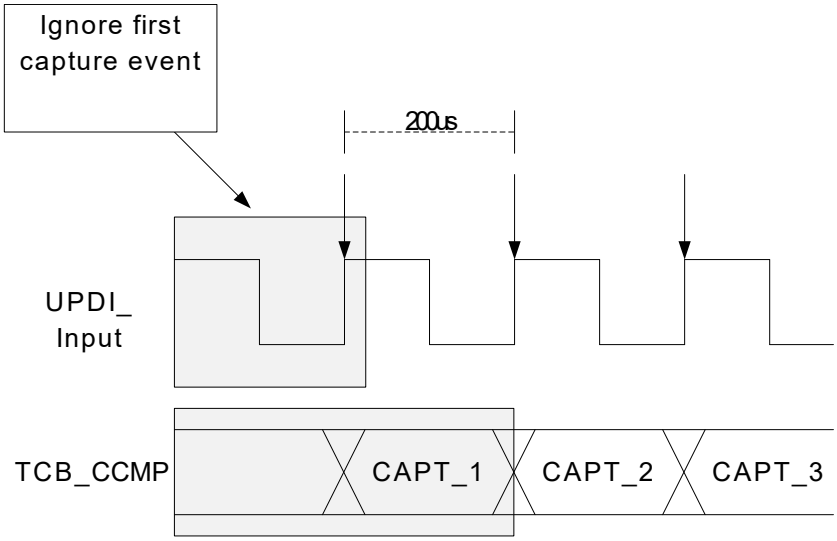
In Standby Sleep mode, the AC is disabled by default. If the Run in Standby Sleep mode bit (RUNSTDBY) in the Control A register (AC.CTRLA) is written to '1', the AC will continue to operate, but the Status register will not be updated, and no Interrupts are generated if no other modules request the CLK_PER, but events and the pad output will be updated.

In Power-Down Sleep mode, the AC and the output to the pad are disabled.

on the input event should be used for the measurement. See the figure below for an example using 10 kbps UPDI SYNCH character pulses, giving a capture window of 200 µs for the timer.

- It is possible to read out the captured value directly after the SYNCH character by reading the TCBn.CCMP register or the value can be written to memory by the CPU once the capture is done.

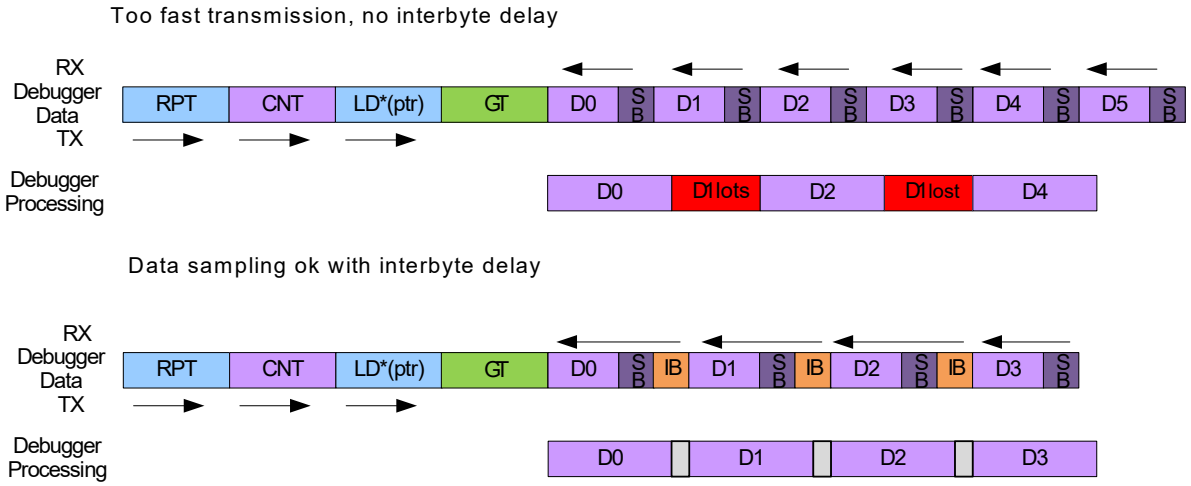
Figure 33-17. UPDI System Clock Measurement Events



33.3.5 Interbyte Delay

When loading data with the UPDI, or reading out the System Information Block, the output data will normally appear with two IDLE bits between each transmitted byte for a multibyte transfer. Depending on the application on the receiver side, data might be coming out too fast when there are no extra IDLE bits between each byte. By enabling the IBDLY feature in UPDI.CTRLB, two extra Stop bits will be inserted between each byte to relax the sampling time for the debugger. Interbyte delay works in the same way as a guard time, by inserting extra IDLE bits, but only a fixed number of IDLE bits and only for multibyte transfers. The first transmitted byte after a direction change will be subject to the regular Guard Time before it is transmitted, and the interbyte delay is not added to this time.

Figure 33-18. Interbyte Delay Example with LD and RPT



GT denotes the Guard Time insertion, SB is for Stop bit and IB is the inserted interbyte delay. The rest of the frames are data and instructions.

Mnemonic	Operands	Description		Op		Flags	#Clocks
			PC(21:16)	←	0		
CALL	k	Call Subroutine	PC	←	k	None	3 / 4
RET		Subroutine Return	PC	←	STACK	None	4 / 5
RETI		Interrupt Return	PC	←	STACK	I	4 / 5
CPSE	Rd,Rr	Compare, skip if Equal	if (Rd = Rr) PC	←	PC + 2 or 3	None	1 / 2 / 3
CP	Rd,Rr	Compare	Rd - Rr			Z,C,N,V,S,H	1
CPC	Rd,Rr	Compare with Carry	Rd - Rr - C			Z,C,N,V,S,H	1
CPI	Rd,K	Compare with Immediate	Rd - K			Z,C,N,V,S,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b) = 0) PC	←	PC + 2 or 3	None	1 / 2 / 3
SBRS	Rr, b	Skip if Bit in Register Set	if (Rr(b) = 1) PC	←	PC + 2 or 3	None	1 / 2 / 3
SBIC	A, b	Skip if Bit in I/O Register Cleared	if (I/O(A,b) = 0) PC	←	PC + 2 or 3	None	1 / 2 / 3
SBIS	A, b	Skip if Bit in I/O Register Set	If (I/O(A,b) =1) PC	←	PC + 2 or 3	None	1 / 2 / 3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC	←	PC + k + 1	None	1 / 2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC	←	PC + k + 1	None	1 / 2
BREQ	k	Branch if Equal	if (Z = 1) then PC	←	PC + k + 1	None	1 / 2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC	←	PC + k + 1	None	1 / 2
BRCS	k	Branch if Carry Set	if (C = 1) then PC	←	PC + k + 1	None	1 / 2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC	←	PC + k + 1	None	1 / 2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC	←	PC + k + 1	None	1 / 2
BRLO	k	Branch if Lower	if (C = 1) then PC	←	PC + k + 1	None	1 / 2
BRMI	k	Branch if Minus	if (N = 1) then PC	←	PC + k + 1	None	1 / 2
BRPL	k	Branch if Plus	if (N = 0) then PC	←	PC + k + 1	None	1 / 2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC	←	PC + k + 1	None	1 / 2
BRLT	k	Branch if Less Than, Signed	if (N ⊕ V= 1) then PC	←	PC + k + 1	None	1 / 2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC	←	PC + k + 1	None	1 / 2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC	←	PC + k + 1	None	1 / 2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC	←	PC + k + 1	None	1 / 2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC	←	PC + k + 1	None	1 / 2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC	←	PC + k + 1	None	1 / 2