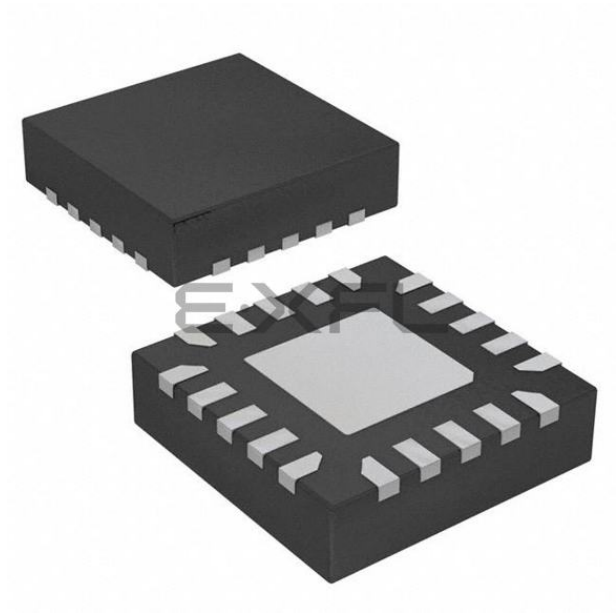


Welcome to E-XFL.COM

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	16MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, WDT
Number of I/O	18
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b; D/A 1x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	20-VFQFN Exposed Pad
Supplier Device Package	20-VQFN (3x3)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny816-mf

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

Inter-Section Write Protection

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

- Code in the BOOT section can write to APPCODE and APPDATA.
- Code in APPCODE can write to APPDATA
- Code in APPDATA cannot write to Flash or EEPROM.

Boot Section Lock and Application Code Section Write Protection

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

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

9.3.1.2 EEPROM

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

9.3.1.3 User Row

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

9.3.2 Memory Access

9.3.2.1 Read

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

9.3.2.2 Page Buffer Load

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

- a device reset
- any page write or erase operation
- a clear page buffer command
- the device wakes up from any sleep mode

9.3.2.3 Programming

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

Before programming a flash page with the data in the page buffer, the flash page must be erased. The page buffer is also erased when the device enter sleep mode. Programming an un-erased flash page will corrupt its content.

Value	ASYNCCH0	ASYNCCH1	ASYNCCH2	ASYNCCH3
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		PIT_DIV128
0x11	PORTA_PIN7	PORTB_PIN7		PIT_DIV64
0x12	UPDI		-	
Other	-	-	-	-

14.5.4 Synchronous Channel n Generator Selection

Name: SYNCCH0, SYNCCH1
Offset: 0x0A + n*0x01 [n=0..1]
Reset: 0x00
Property: -

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

Bits 7:0 – SYNCCH[7:0]: Synchronous Channel Generator Selection

Table 14-3. Synchronous Channel Generator selection

Value	SYNCCH0	SYNCCH1
0x00	OFF	
0x01	TCB0	
0x02	TCA0_OVF_LUNF	
0x03	TCA0_HUNF	

- Window Period bits in Control A register (CTRLA.WINDOW)
- Lock bit in Status register (STATUS.LOCK)

Related Links

CCP - Configuration Change Protection

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

CCP

20. TCA - 16-bit Timer/Counter Type A

20.1 Features

- 16-bit timer/counter
- Three compare channels
- Double buffered timer period setting
- Double buffered compare channels
- Waveform generation:
  - Frequency generation
  - Single-slope PWM (pulse width modulation)
  - Dual-slope PWM
- Count on event
- Timer overflow interrupts/events
- One compare match per compare channel
- Two 8-bit timer/counters in Split Mode

20.2 Overview

The flexible 16-bit PWM Timer/Counter type A (TCA) provides accurate program execution timing, frequency and waveform generation, and command execution.

A TCA consists of a base counter and a set of compare channels. The base counter can be used to count clock cycles or events, or let events control how it counts clock cycles. It has direction control and period setting that can be used for timing. The compare channels can be used together with the base counter to do compare match control, frequency generation, and pulse width waveform modulation.

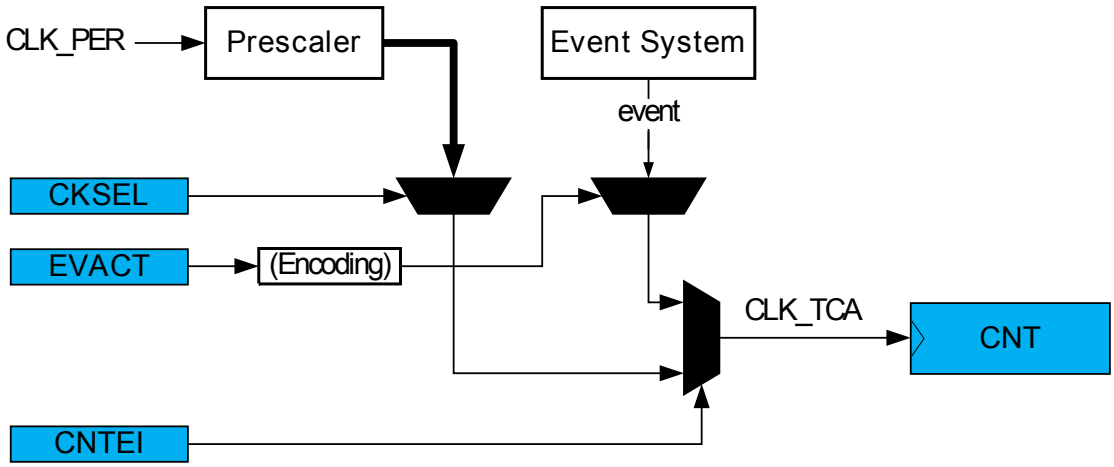
Depending on the mode of operation, the counter is cleared, reloaded, incremented, or decremented at each timer/counter clock or event input.

A timer/counter can be clocked and timed from the peripheral clock with optional prescaling or from the event system. The event system can also be used for direction control or to synchronize operations.

By default, the TCA is a 16-bit timer/counter. The timer/Counter has a Split mode feature that splits it into two 8-bit timer/counters with three compare channels each.

A block diagram of the 16-bit timer/counter with closely related peripheral modules (in grey) is shown below.

Figure 20-3. Timer/Counter Clock Logic



20.2.2 Signal Description

Signal	Description	Type
WO[2:0]	Digital output	Waveform output
WO[5:3]	Digital output	Waveform output - Split Mode only

20.2.3 System Dependencies

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

Table 20-1. TCA System Dependencies

Dependency	Applicable	Peripheral
Clocks	Yes	CLKCTRL
I/O Lines and Connections	Yes	WO[5:0]
Interrupts	Yes	CPUINT
Events	Yes	EVSYS
Debug	Yes	UPDI

Related Links

Clocks
Debug Operation
Interrupts
Events

20.2.3.1 Clocks

This peripheral uses the system clock CLK_PER, and has its own prescaler.

Related Links

CLKCTRL - Clock Controller

20.2.3.2 I/O Lines and Connections

Using the I/O lines of the peripheral requires configuration of the I/O pins.

20.6 Register Summary - TCA in Split Mode (CTRLD.SPLITM=1)

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0					CLKSEL[2:0]			ENABLE
0x01	CTRLB	7:0		HCMP2EN	HCMP1EN	HCMP0EN		LCMP2EN	LCMP1EN	LCMP0EN
0x02	CTRLC	7:0		HCMP2OV	HCMP1OV	HCMP0OV		LCMP2OV	LCMP1OV	LCMP0OV
0x03	CTRLD	7:0								SPLITM
0x04	CTRLECLR	7:0					CMD[1:0]			
0x05	CTRLESET	7:0					CMD[1:0]			
0x06 ... 0x09	Reserved									
0x0A	INTCTRL	7:0		LCMP2	LCMP1	LCMP0			HUNF	LUNF
0x0B	INTFLAGS	7:0		LCMP2	LCMP1	LCMP0			HUNF	LUNF
0x0C ... 0x0D	Reserved									
0x0E	DBGCTRL	7:0								DBGRUN
0x0F ... 0x1F	Reserved									
0x20	LCNT	7:0	LCNT[7:0]							
0x21	HCNT	7:0	HCNT[7:0]							
0x22 ... 0x25	Reserved									
0x26	LPER	7:0	LPER[7:0]							
0x27	HPER	7:0	HPER[7:0]							
0x28	LCMP0	7:0	LCMP[7:0]							
0x29	HCMP0	7:0	HCMP[7:0]							
0x2A	LCMP1	7:0	LCMP[7:0]							
0x2B	HCMP1	7:0	HCMP[7:0]							
0x2C	LCMP2	7:0	LCMP[7:0]							
0x2D	HCMP2	7:0	HCMP[7:0]							

20.7 Register Description - Split Mode

20.7.1 Control A

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

Value	Description
0x2	Use CLK_TCA from TCA0
0x3	Reserved

Bit 0 – ENABLE: Enable
Writing this bit to '1' enables the Timer/Counter type B peripheral.

21.5.2 Control B

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

Bit	7	6	5	4	3	2	1	0
		ASYNC	CCMPINIT	CCMPEN		CNTMODE[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 6 – ASYNC: Asynchronous Enable
Writing this bit to '1' will allow asynchronous updates of the TCB output signal in single shot mode

Value	Description
0	The output will go HIGH when the counter actually starts
1	The output will go HIGH when an Event arrives

Bit 5 – CCMPINIT: Compare/Capture Pin Initial Value
This bit is used to set the initial output value of the pin when a pin output is used.

Value	Description
0	Initial pin state is LOW
1	Initial pin state is HIGH

Bit 4 – CCMPEN: Compare/Capture Output Enable
This bit is used to set the output value of the Compare/Capture Output.

Value	Description
0	Compare/Capture Output is zero
1	Compare/Capture Output has a valid value

Bits 2:0 – CNTMODE[2:0]: Timer Mode
Writing these bits selects the timer mode.

Value	Description
0x0	Periodic interrupt mode
0x1	Timeout check mode
0x2	Input capture on event mode
0x3	Input capture frequency measurement mode
0x4	Input capture pulse width measurement mode
0x5	Input capture frequency and pulse width measurement mode

Count Mode	EDGE	Positive Edge	Negative Edge
		2nd Negative: stop, interrupt	
Single Shot Mode	0	Start counter	Not Applicable
	1	Start counter	Start counter
8-bit PWM Mode	0	Not Applicable	Not Applicable
	1	Not Applicable	Not Applicable

Bit 0 – CAPTEI: Capture Event Input Enable
Writing this bit to '1' enables the event input capture.

21.5.4 Interrupt Control

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

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

Bit 0 – CAPT: Capture Interrupt Enable
Writing this bit to '1' enables the Capture interrupt.

21.5.5 Interrupt Flags

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

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

Bit 0 – CAPT: Interrupt Flag
This bit is set when an interrupt occurs. The interrupt conditions are dependent on the Counter mode (CNTMODE) in TCB.CTRLB.

This bit is cleared by writing a '1' to it or when the Capture register is read in capture mode.

INPUTMODE	Trigger → Output affected	Fault On/Active	Fault Release/Inactive
0x05	Input A→{WOA, WOB}	Execute dead-time only	
	Input B→{WOA, WOB}		
0x06	Input A→{WOA, WOB}	End on-time and wait	Start with dead-time for other compare
	Input B→{WOA, WOB}		
0x07	Input A→{WOA, WOB}	End on-time and wait for software action	Start with dead-time for current compare
	Input B→{WOA, WOB}		
0x08	Input A→WOA	End current on-time and continue with other off-time	
	Input B→WOB		
0x09	Input A→WOA	Block current on-time and continue sequence	
	Input B→WOB		
0x0A	Input A→WOA	Deactivate on-time until end of sequence while trigger is active	
	Input B→WOB		
other	-	-	-

22.3.2.5 Dithering

If it is not possible to achieve the desired frequency because of pre-scaler/period selection limitations, Dithering can be used to approximate the desired frequency and reduce waveform drift.

Dither accumulates the fractional error of the counter clock for each cycle. When the fractional error overflows, an additional cycle is added to the selected part of the cycle.

Example

If the timer clock frequency is 10MHz, it will give the timer a resolution of 100ns. Then, the output frequency should be 75 kHz. Here 75 kHz means a period of 13333ns, and that is not possible to achieve with a constant period with a 100ns resolution because it equals 133.33 cycles. The output period can be set to either 133 cycles (75.188kHz) or 134 cycles (74.626 kHz).

It is possible to change the period between the two frequencies manually in the firmware to get an average output frequency of 75 kHz. (change the every 3rd period to 134 cycles) The dither can do this automatically by accumulating the error (0.33 cycles). The accumulator calculate when the accumulated error is larger than one clock cycle and when that happens it adds an additional cycle to the timer period.

Offset	Name	Bit Pos.								
0x2B		15:8					CMPCLR[11:8]			
0x2C	CMPBSET	7:0	CMPSET[7:0]							
0x2D		15:8					CMPSET[11:8]			
0x2E	CMPBCLR	7:0	CMPCLR[7:0]							
0x2F		15:8					CMPCLR[11:8]			

22.5 Register Description

22.5.1 Control A

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

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

Bits 6:5 – CLKSEL[1:0]: Clock Select
The clock select bits select the clock source of the TCD clock.

Value	Description
0x0	OSC20M
0x1	Reserved
0x2	External clock
0x3	System clock

Bits 4:3 – CNTPRES[1:0]: Counter Prescaler
The Counter Prescaler bits select the division factor of the TCD counter clock.

Value	Description
0x0	Division factor 1
0x1	Division factor 4
0x2	Division factor 32
0x3	Reserved

Bits 2:1 – SYNCPRES[1:0]: Synchronization Prescaler
The synchronization prescaler bits select the division factor of the TCD clock.

Value	Description
0x0	Division factor 1
0x1	Division factor 2
0x2	Division factor 4
0x3	Division factor 8

Table 26-3. SDA Hold Time

SDAHOLD[1:0]	Nominal Hold Time	Hold Time Range across All Corners (ns)	Description
0x0	OFF	0	Hold time off.
0x1	50ns	36 - 131	Backward compatible setting.
0x2	300ns	180 - 630	Meets SMBus specification under typical conditions.
0x3	500ns	300 - 1050	Meets SMBus specification across all corners.

Bit 1 – FMPEN: FM Plus Enable
Writing these bits selects the 1MHz bus speed (Fast mode plus, Fm+) for the TWI in default configuration.

Value	Description
0	Fm+ disabled
1	Fm+ enabled

26.5.2 Debug Control

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

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

Bit 0 – DBGRUN: Debug Run

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.

26.5.3 Master Control A

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

27.4 Register Summary - CRCSCAN

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0	RESET						NMIEN	ENABLE
0x01	CTRLB	7:0			MODE[1:0]				SRC[1:0]	
0x02	STATUS	7:0							OK	BUSY

27.5 Register Description

27.5.1 Control A

If an NMI has been triggered, this register is not writable.

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

Bit	7	6	5	4	3	2	1	0
	RESET						NMIEN	ENABLE
Access	R/W						R/W	R/W
Reset	0						0	0

Bit 7 – RESET: Reset CRCSCAN
Writing this bit to '1' resets the CRCSCAN peripheral: The CRCSCAN Control registers and Status register (CTRLA, CTRLB, STATUS) will be cleared one clock cycle after the RESET bit was written to '1'.

If NMIEN is '0', this bit is writable both when the CRCSCAN is busy (BUSY bit in CRCSCAN.STATUS is '1') and not busy (BUSY bit is '0'), and will take effect immediately.

If NMIEN is '1', this bit is only writable when the CRCSCAN is not busy (BUSY bit in CRCSCAN.STATUS is '0').

The RESET bit is a strobe bit.

Bit 1 – NMIEN: Enable NMI Trigger
When this bit is written to '1', any CRC failure will trigger an NMI.

This can only be cleared by a system Reset - it is not cleared by a write to the RESET bit.

This bit can only be written to '1' when the CRCSCAN is not busy (BUSY bit in CRCSCAN.STATUS is '0').

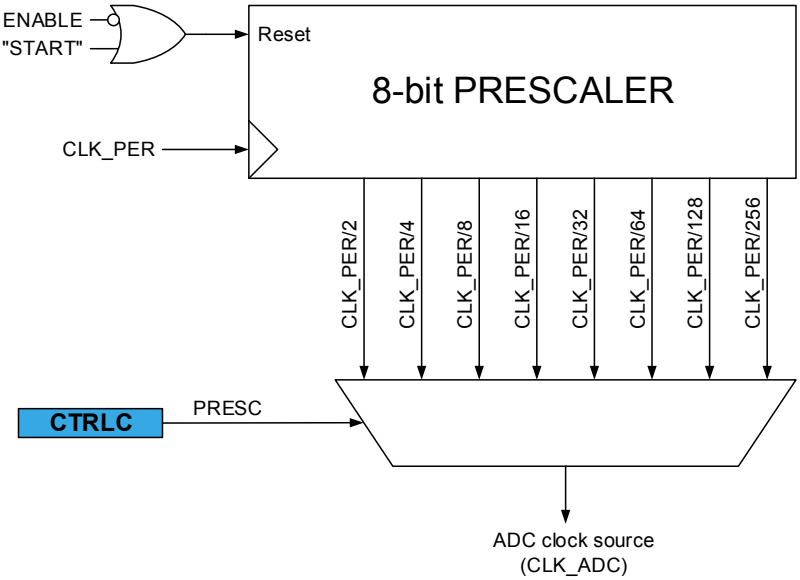
Bit 0 – ENABLE: Enable CRCSCAN
Writing this bit to '1' enables the CRCSCAN peripheral with the current settings. It will stay '1' even after a CRC check has completed, but writing it to 1 again will start a new check.

Writing the bit to '0' will disable the CRCSCAN after the ongoing check is completed (after reaching the end of the section it is set up to check). This is the preferred way to stop a continuous background check. A failure in the ongoing check will still be detected and can cause an NMI if the NMIEN bit is '1'.

The CRCSCAN can be enabled during the internal reset initialization to verify Flash sections before letting the CPU start normal code execution (see device datasheet fuse description). If the CRCSCAN is

30.3.2.2 Clock generation

Figure 30-6. ADC Prescaler



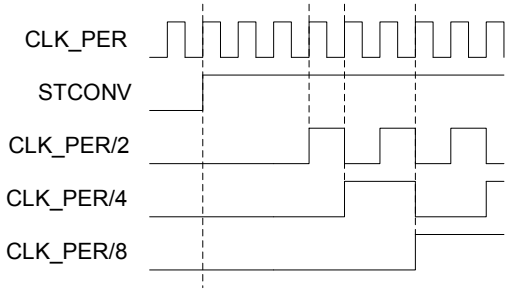
The ADC requires an input clock frequency between 50kHz and 1.5MHz for maximum resolution. If a lower resolution than 10 bits is selected, the input clock frequency to the ADC can be higher than 1.5MHz to get a higher sample rate.

The ADC module contains a prescaler which generates the ADC clock (CLK_ADC) from any CPU clock (CLK_PER) above 100kHz. The prescaling is selected by writing to the Prescaler bits (PRESC) in the Control C register (ADC.CTRLC). The prescaler starts counting from the moment the ADC is switched on by writing a '1' to the ENABLE bit in ADC.CTRLA. The prescaler keeps running as long as the ENABLE bit is one, the prescaler counter is reset to zero when the ENABLE bit is zero.

When initiating a conversion by writing a '1' to the Start Conversion bit (STCONV) in the Command register (ADC.COMMAND) or from event, the conversion starts at the following rising edge of the CLK_ADC clock cycle. The prescaler is kept reset as long as there is no ongoing conversion. This assures a fixed delay from the trigger to the actual start of a conversion in CLK_PER cycles as:

$$\text{StartDelay} = \frac{\text{PRESC}_{\text{factor}}}{2} + 2$$

Figure 30-7. Start conversion and clock generation



30.3.2.3 Conversion timing

A normal conversion takes 13 CLK_ADC cycles. The actual sample-and-hold takes place 2 CLK_ADC cycles after the start of a conversion. Start of conversion is initiated by writing a '1' to the STCONV bit in ADC.COMMAND. When a conversion is complete, the result is available in the Result register (ADC.RES), and the Result Ready interrupt flag is set (RESRDY in ADC.INTFLAG). The interrupt flag will

32. PTC - Peripheral Touch Controller

32.1 Overview

The Peripheral Touch Controller (PTC) acquires signals in order to detect touch on capacitive sensors. The external capacitive touch sensor is typically formed on a PCB, and the sensor electrodes are connected to the analog front end of the PTC through the I/O pins in the device. The PTC supports both self- and mutual-capacitance sensors.

In mutual-capacitance mode, sensing is done using capacitive touch matrices in various X-Y configurations, including indium tin oxide (ITO) sensor grids. The PTC requires one pin per X-line and one pin per Y-line.

In self-capacitance mode, the PTC requires only one pin (Y-line) for each touch sensor.

The number of available pins and the assignment of X- and Y-lines is depending on both package type and device configuration. Refer to the Configuration Summary and I/O Multiplexing table for details.

Related Links

Configuration Summary

I/O Multiplexing and Considerations

32.2 Features

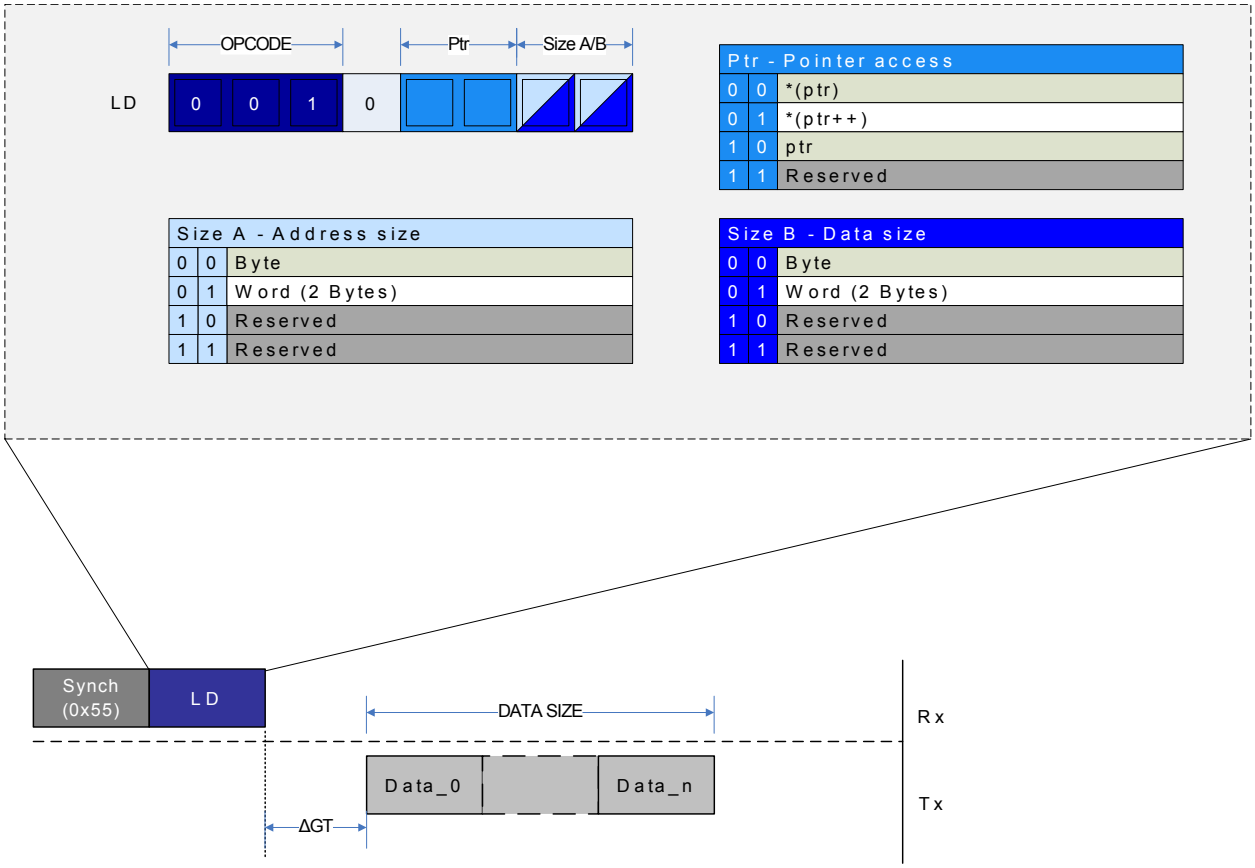
- Low-power, high-sensitivity, environmentally robust capacitive touch buttons, sliders, wheels
- Supports wake-up on touch from Sleep mode
- Supports mutual capacitance and self-capacitance sensing
  - Mix-and-match mutual-and self-capacitance sensors
- One pin per electrode – no external components
- Load compensating charge sensing
  - Parasitic capacitance compensation and adjustable gain for superior sensitivity
- Zero drift over the temperature and V_{DD} range
  - Auto calibration and re-calibration of sensors
- Single-shot and free-running charge measurement
- Hardware noise filtering and noise signal de-synchronization for high conducted immunity
- Driven shield for better noise immunity and moisture tolerance
- Selectable channel change delay allows choosing the settling time on a new channel, as required
- Acquisition-start triggered by command or through auto-triggering feature
- Low CPU utilization through interrupt on acquisition-complete
- Using ADC peripheral for signal conversion and acquisition
- Supported by Atmel|START and Atmel Studio documentation.

Related Links

Configuration Summary

I/O Multiplexing and Considerations

Figure 33-11. LD Instruction Operation

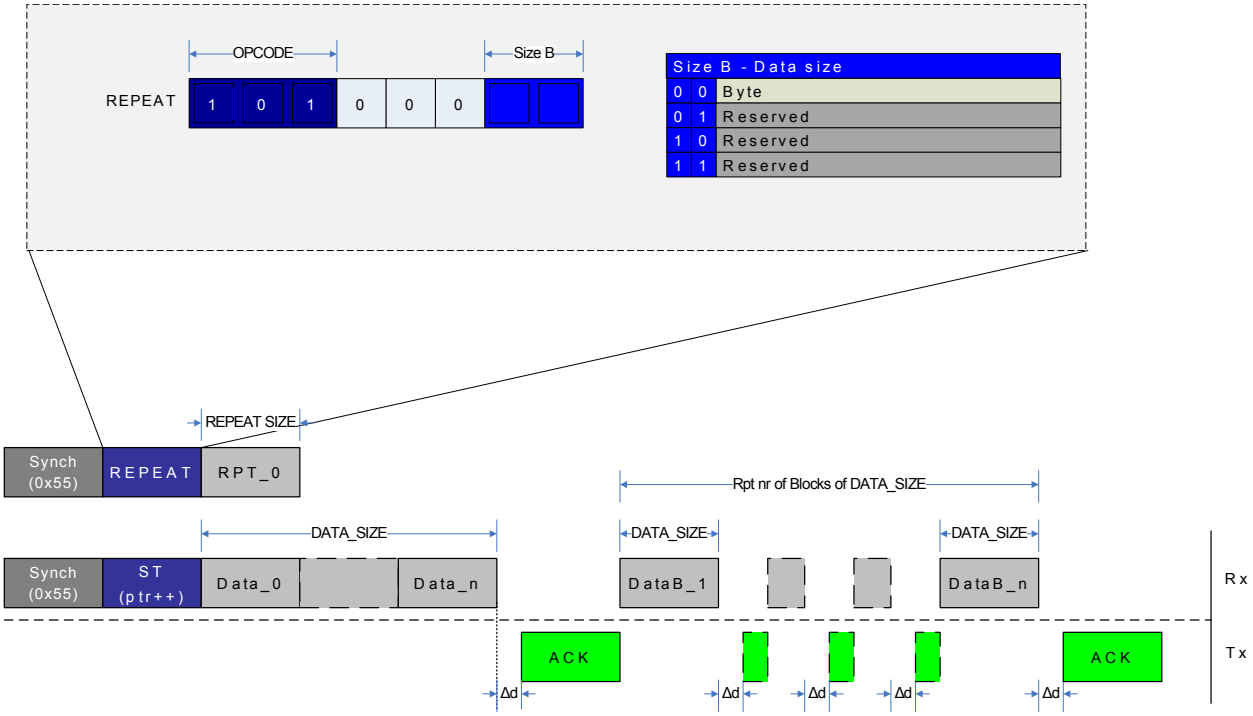


The Figure above shows an example of a typical LD sequence, where data is received after the Guard Time period. Loading data from the UPDI pointer register follows the same transmission protocol.

33.3.3.4 ST - Store Data from Data Space Using Indirect Addressing

The ST instruction is used to store data that is shifted serially into the PHY layer to the bus matrix address space. The ST instruction is based on indirect addressing, which means that the address pointer in the UPDI needs to be written prior to bus matrix access. Automatic pointer post increment operation is supported, and is useful when the ST instruction is used with REPEAT. ST is also used to store the UPDI address pointer into the pointer register. Maximum supported size for storing address and data is 16 bit.

Figure 33-15. `REPEAT` Instruction Operation



The Figure above gives an example of repeat operation with a `ST` instruction using pointer post increment operation. After the `REPEAT` instruction is sent with `RPT_0 =` ***n***, the first `ST` instruction is issued with SYNCH- and Instruction frame, while the next ***n*** `ST`-instructions are executed by only sending in data bytes according to the `ST` operand `DATA_SIZE`, and maintaining the Acknowledge (ACK) handshake protocol.

If using indirect addressing instructions `(LD/ST)` it is recommended to always use the pointer post increment option when combined with `REPEAT` . 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

The `KEY` instruction is used for communicating KEY bytes to the UPDI, opening up for executing protected features on the device. See Table 33-5for an overview over functions that are activated by KEYs. For the `KEY` instruction, only 64bit KEY size is supported. If the System Information Block (SIB) field of the `KEY` instruction is set, the `KEY` instruction returns the SIB instead of expecting incoming KEY bytes. Maximum supported size for SIB is 128bits.

Table 34-25. Power Supply, Reference, and Input Range

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
V_{DD}	Supply Voltage		1.8	-	5.5	V
V_{REF}	Reference Voltage	REFSEL = Internal reference	0.55	-	V_{DD}-0.5	V
		REFSEL = V_{DD}	1.8	-	5.5	
C_{IN}	Input capacitance	SAMPCAP=5pF	-	5	-	pF
		SAMPCAP=10pF	-	10	-	
V_{IN}	Input Voltage Range		0	-	V_{REF}	V
I_{BAND}	Input Bandwidth	1.1V≤V_{REF}	-	-	57.5	kHz

Table 34-26. Clock and Timing Characteristics

Symbol	Description	Conditions	Min.	Typ.	Max.	Unit
f_{ADC}	Sample Rate	1.1V≤V_{REF}	15	-	115	ksps
		1.1V≤V_{REF} (8 bit resolution)	15	-	150	
		V_{REF}=0.55V (10 bit)	7.5	-	20	
CLK_{ADC}	Clock frequency	V_{REF}=0.55V (10 bit)	100	-	260	kHz
		1.1V≤V_{REF} (10 bit)	200	-	1500	
		1.1V≤V_{REF} (8 bit resolution)	200	-	2000	
Ts	Sampling time		2	2	33	CLK_{ADC} cycles
T_{CONV}	Conversion time (latency)	Sampling time = 2CLK_{ADC}	8.7	-	50	µs
T_{START}	Start-up time	Internal V_{REF}	-	22	-	µs

Table 34-27. Accuracy Characteristics

Symbol	Description	Conditions		Min.	Typ.	Max.	Unit
Res	Resolution			-	10	-	bit
INL	Integral Non Linearity	REFSEL = Internal reference or V_{DD}	0.55V≤V_{REF}≤V_{DD} f_{ADC}=15ksps	-	1.0	-	LSB
			1.1V≤V_{REF}≤2.5 f_{ADC}=115ksps	-	1.0	-	
			2.5V<V_{REF}≤V_{DD} f_{ADC}=115ksps	-	1.3	-	
DNL(1)	Dynamic Non Linearity	REFSEL = Internal reference or V_{DD}	V_{REF}=0.55V f_{ADC}=15ksps	-	0.4	-	LSB
			1.1V≤V_{REF}≤V_{DD} f_{ADC}=15ksps	-	0.3	-	

35.1.6 Supply Currents in Idle Mode for ATtiny816

Figure 35-21. ATtiny816: Idle Supply Current vs. Frequency (1-20MHz) at T=25°C

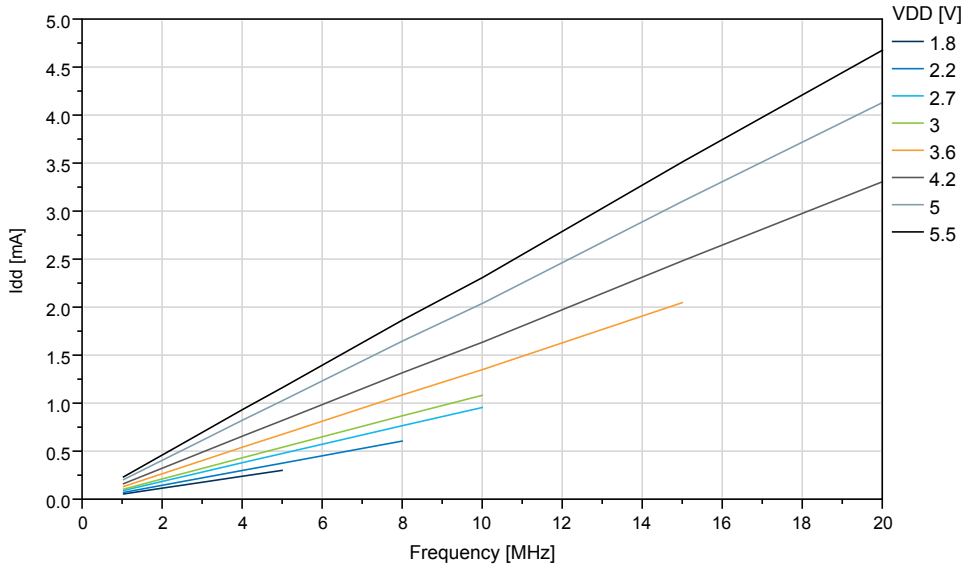
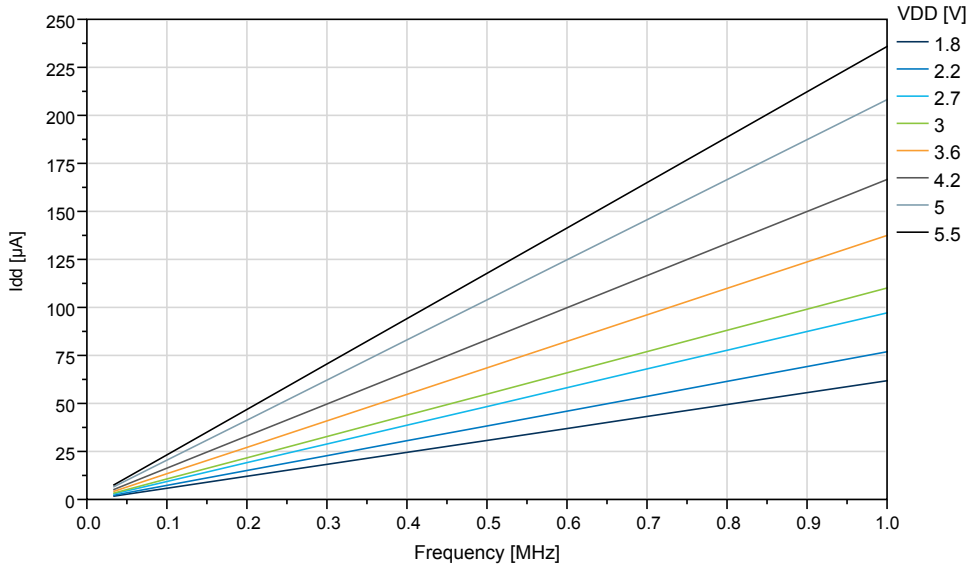


Figure 35-22. ATtiny816: Idle Supply Current vs. Low Frequency (0.1-1.0MHz) at T=25°C



40. Errata

40.1 Errata - ATtiny416

40.1.1 Die Revision A

40.1.1.1 AC

1 –

False triggers may occur under certain conditions.

False triggers may occur on falling input pin:

- For common mode voltage below 0.5V
- For common mode voltage above 0.5V if slew rate is greater than 1V/us

Fix/Workaround:

None.

2 –

AC interrupt flag not set unless interrupt is enabled

ACn.STATUS.CMP is not set if the ACn.INTCTRL.CMP is not set.

Fix/Workaround:

Enable ACn.INTCTRL.CMP or use ACn.STATUS.STATE for polling.

40.1.1.2 ADC

1 –

One extra measurement performed after disabling ADC free running mode

The ADC may perform one additional measurement after clearing ADCn.CTRLA.FREERUN

Fix/Workaround:

Write ADCn.CTRLA.ENABLE to zero to stop the free running mode immediately.

2 –

Changing ADC control bits during free running mode not working

If control signals are changed during free running mode, the new configuration is not properly taken into account in the next measurement. This is valid for registers ADC.CTRLB, ADC.CTRLC, ADC.SAMPCTRL and ADC.MUXPOS, ADC.WINLT and ADC.WINHT.

Fix/Workaround:

Disable ADC free running mode before update of ADC.CTRLB, ADC.CTRLC, ADC.SAMPCTRL, ADC.MUXPOS, ADC.WINLT or ADC.WINHT.

3 –

SAMPDLY and ASDV does not work together with SAMPLEN

Using SAMPCTRL.SAMPLEN at the same time as CTRLD.SAMPDLY or CTRLD.ASDV will cause an unpredictable sampling length.

Fix/Workaround:

When setting SAMPCTRL.SAMPLEN greater than zero the CTRLD.SAMPDLY and CTRLD.ASDV must be cleared.