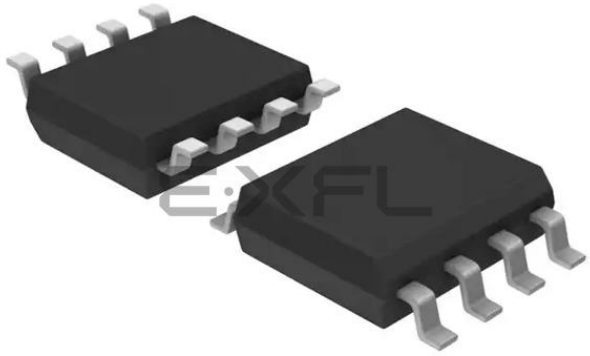


Welcome to E-XFL.COM

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"

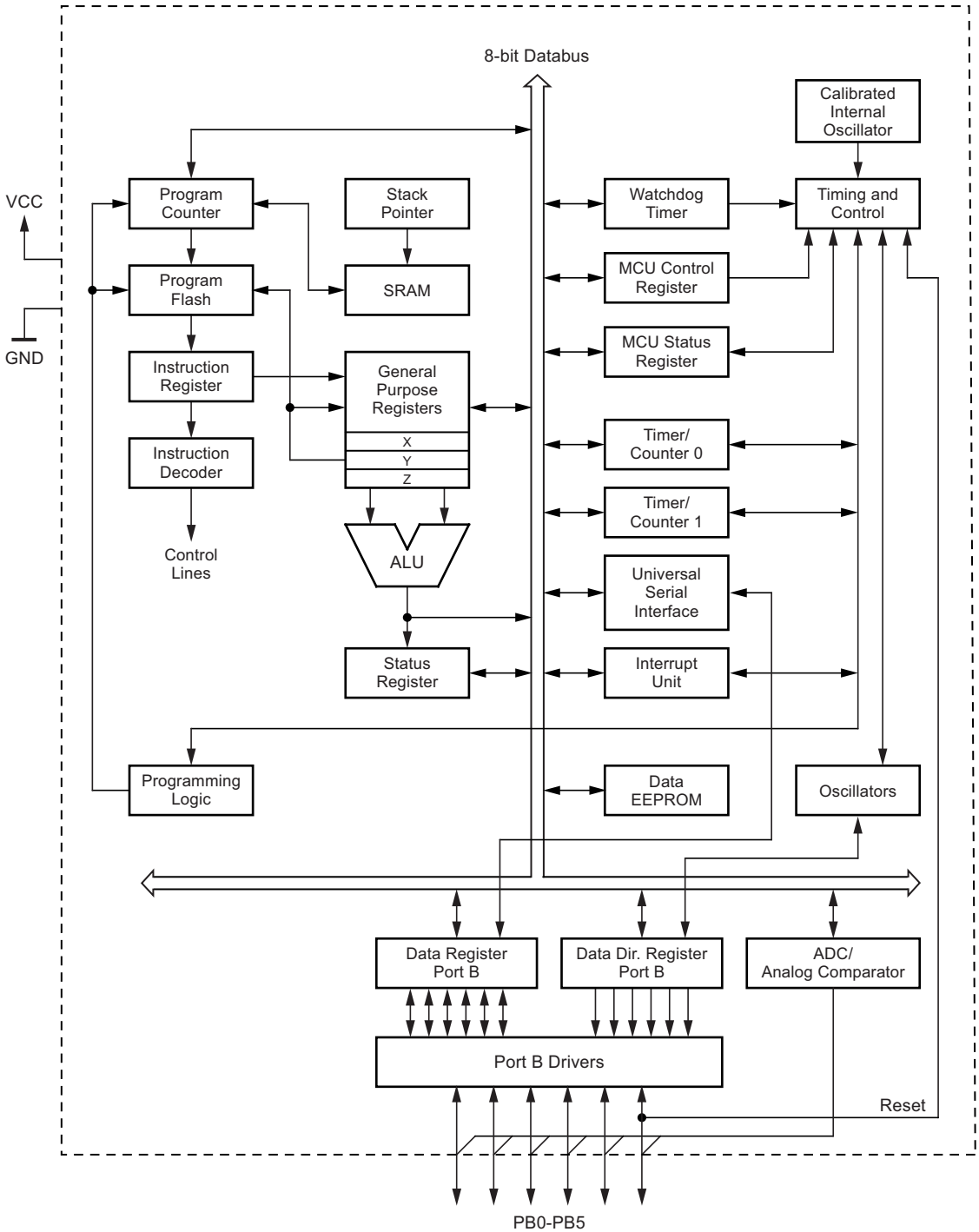
Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	USI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	6
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 4x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	8-SOIC (0.209", 5.30mm Width)
Supplier Device Package	8-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny85-15st

1. Overview

The Atmel® ATtiny25/45/85 is a low-power CMOS 8-bit microcontroller based on the AVR® enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATtiny25/45/85 achieves throughputs approaching 1MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

1.1 Block Diagram

Figure 1-1. Block Diagram



6.2 Idle Mode

When the SM1..0 bits are written to 00, the SLEEP instruction makes the MCU enter idle mode, stopping the CPU but allowing analog comparator, ADC, Timer/Counter, watchdog, and the interrupt system to continue operating. This sleep mode basically halts clk_{CPU} and clk_{FLASH}, while allowing the other clocks to run.

Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the timer overflow. If wake-up from the analog comparator interrupt is not required, the analog comparator can be powered down by setting the ACD bit in the analog comparator control and status register – ACSR. This will reduce power consumption in Idle mode. If the ADC is enabled, a conversion starts automatically when this mode is entered.

6.3 ADC Noise Reduction Mode

When the SM1..0 bits are written to 01, the SLEEP instruction makes the MCU enter ADC noise reduction mode, stopping the CPU but allowing the ADC, the external interrupts, and the watchdog to continue operating (if enabled). This sleep mode halts $clk_{I/O}$, clk_{CPU}, and clk_{FLASH}, while allowing the other clocks to run.

This improves the noise environment for the ADC, enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically when this mode is entered. Apart form the ADC conversion complete interrupt, only an external reset, a watchdog reset, a brown-out reset, an SPM/EEPROM ready interrupt, an external level interrupt on INT0 or a pin change interrupt can wake up the MCU from ADC noise reduction mode.

6.4 Power-down Mode

When the SM1..0 bits are written to 10, the SLEEP instruction makes the MCU enter power-down mode. In this mode, the oscillator is stopped, while the external interrupts, and the watchdog continue operating (if enabled). Only an external reset, a watchdog reset, a brown-out reset, an external level interrupt on INT0, or a pin change interrupt can wake up the MCU. This sleep mode halts all generated clocks, allowing operation of asynchronous modules only.

Note that if a level triggered interrupt is used for wake-up from power-down mode, the changed level must be held for some time to wake up the MCU. Refer to Section 10. “External Interrupts” on page 54 for details.

Table 6-2. Active Clock Domains and Wake-up Sources in the Different Sleep Modes

	Active Clock Domains					Oscillators	Wake-up Sources					
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{PCK}	Main Clock Source Enabled	INT0 and Pin Change	SPM/ EEPROM Ready	USI Start Condition	ADC	Other I/O	Watchdog Interrupt
Idle			X	X	X	X	X	X	X	X	X	X
ADC Noise Reduction				X		X	X(1)	X	X	X		X
Power-down							X(1)		X			X

Note: 1. For INT0, only level interrupt.

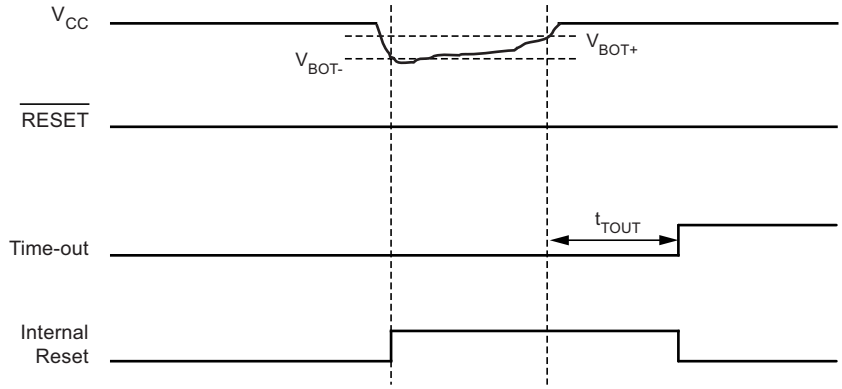
6.5 Limitations

BOD disable functionality has been implemented in the following devices, only:

- ATtiny25, revision D, and newer
- ATtiny45, revision D, and newer
- ATtiny85, revision C, and newer

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than t_{BOD} given in Table 7-1 on page 34.

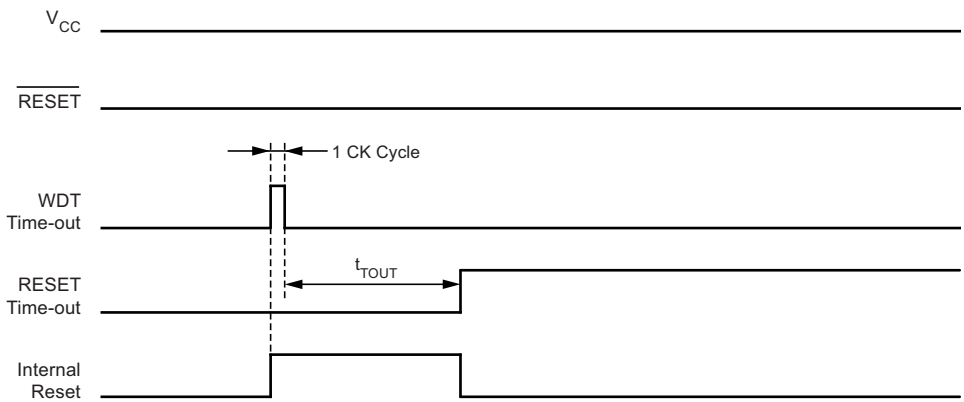
Figure 7-5. Brown-out Reset During Operation



7.6 Watchdog Reset

When the watchdog times out, it will generate a short reset pulse of one CK cycle duration. On the falling edge of this pulse, the delay timer starts counting the time-out period t_{TOUT}. Refer to Section 7.9 “Watchdog Timer” on page 38 for details on operation of the watchdog timer.

Figure 7-6. Watchdog Reset During Operation



8. Interrupts

This section describes the specifics of the interrupt handling as performed in Atmel® ATtiny25/45/85. For a general explanation of the AVR® interrupt handling, refer to Section 3.8 “Reset and Interrupt Handling” on page 10.

8.1 Interrupt Vectors in ATtiny25/45/85

Table 8-1. Reset and Interrupt Vectors

Vector No.	Program Address	Source	Interrupt Definition
1	0x0000	RESET	External pin, power-on reset, brown-out reset, watchdog reset
2	0x0001	INT0	External interrupt request 0
3	0x0002	PCINT0	Pin change interrupt request 0
4	0x0003	TIM1_COMPA	Timer/Counter1 compare match A
5	0x0004	TIM1_OVF	Timer/Counter1 overflow
6	0x0005	TIM0_OVF	Timer/Counter0 overflow
7	0x0006	EE_RDY	EEPROM ready
8	0x0007	ANA_COMP	Analog comparator
9	0x0008	ADC	ADC conversion complete
10	0x0009	TIM1_COMPB	Timer/Counter1 compare match B
11	0x000A	TIM0_COMPA	Timer/Counter0 compare match A
12	0x000B	TIM0_COMPB	Timer/Counter0 compare match B
13	0x000C	WDT	Watchdog time-out
14	0x000D	USI_START	USI START
15	0x000E	USI_OVF	USI overflow

If the program never enables an interrupt source, the interrupt vectors are not used, and regular program code can be placed at these locations. The most typical and general program setup for the reset and interrupt vector addresses in Atmel ATtiny25/45/85 is:

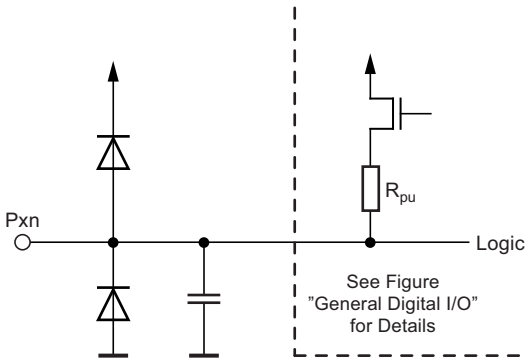
```
Address      Labels Code                 Comments
0x0000              rjmp   RESET        ; Reset Handler
0x0001              rjmp   EXT_INT0     ; IRQ0 Handler
0x0002              rjmp   PCINT0       ; PCINT0 Handler
0x0003              rjmp   TIM1_COMPA   ; Timer1 CompareA Handler
0x0004              rjmp   TIM1_OVF     ; Timer1 Overflow Handler
0x0005              rjmp   TIM0_OVF     ; Timer0 Overflow Handler
0x0006              rjmp   EE_RDY       ; EEPROM Ready Handler
0x0007              rjmp   ANA_COMP     ; Analog Comparator Handler
0x0008              rjmp   ADC          ; ADC Conversion Handler
0x0009              rjmp   TIM1_COMPB   ; Timer1 CompareB Handler
0x000A              rjmp   TIM0_COMPA   ;
0x000B              rjmp   TIM0_COMPB   ;
0x000C              rjmp   WDT          ;
0x000D              rjmp   USI_START    ;
0x000E              rjmp   USI_OVF      ;
0x000F       RESET: ldi    r16, low(RAMEND); Main program start
0x0010              ldi    r17, high(RAMEND); Tiny85 has also SPH
0x0011              out    SPL, r16     ; Set Stack Pointer to top of RAM
0x0012              out    SPH, r17     ; Tiny85 has also SPH
0x0013              sei                 ; Enable interrupts
0x0014       <instr> xxx
...    ...    ...    ...
```

9. I/O Ports

9.1 Introduction

All AVR® ports have true read-modify-write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both V_{CC} and ground as indicated in Figure 9-1. Refer to Section 21. “Electrical Characteristics” on page 137 for a complete list of parameters.

Figure 9-1. I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case “x” represents the numbering letter for the port, and a lower case “n” represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. For example, PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn. The physical I/O registers and bit locations are listed in Section 9.4 “Register Description for I/O-Ports” on page 53.

Three I/O memory address locations are allocated for each port, one each for the data register – PORTx, data direction register – DDRx, and the port input Pins – PINx. The port input pins I/O location is read only, while the data register and the data direction register are read/write. However, writing a logic one to a bit in the PINx register, will result in a toggle in the corresponding bit in the data register. In addition, the pull-up disable – PUD bit in MCUCR disables the pull-up function for all pins in all ports when set.

Using the I/O port as general digital I/O is described in Section 9.2 “Ports as General Digital I/O” on page 44. Most port pins are multiplexed with alternate functions for the peripheral features on the device. How each alternate function interferes with the port pin is described in Section 9.3 “Alternate Port Functions” on page 48. Refer to the individual module sections for a full description of the alternate functions.

Note that enabling the alternate function of some of the port pins does not affect the use of the other pins in the port as general digital I/O.

Table 9-2 summarizes the function of the overriding signals. The pin and port indexes from Figure 9-5 on page 48 are not shown in the succeeding tables. The overriding signals are generated internally in the modules having the alternate function.

Table 9-2. Generic Description of Overriding Signals for Alternate Functions

Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up override value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD register bits.
DDOE	Data direction override enable	If this signal is set, the output driver enable is controlled by the DDOV signal. If this signal is cleared, the output driver is enabled by the DDxn register bit.
DDOV	Data direction override value	If DDOE is set, the output driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn register bit.
PVOE	Port value override enable	If this signal is set and the output driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the output driver is enabled, the port value is controlled by the PORTxn register bit.
PVOV	Port value override value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn register bit.
PTOE	Port toggle override enable	If PTOE is set, the PORTxn register bit is inverted.
DIEOE	Digital input enable override enable	If this bit is set, the digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU state (Normal mode, sleep mode).
DIEOV	Digital input enable Override value	If DIEOE is set, the digital input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, sleep mode).
DI	Digital input	This is the digital input to alternate functions. In the figure, the signal is connected to the output of the schmitt-trigger but before the synchronizer. Unless the digital input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog input/output	This is the analog input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

The following subsections shortly describe the alternate functions for each port, and relate the overriding signals to the alternate function. Refer to the alternate function description for further details.

9.3.1 MCU Control Register – MCUCR

Bit	7	6	5	4	3	2	1	0	
	BODS	PUD	SE	SM1	SM0	BODSE	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 6 – PUD: Pull-up Disable**

When this bit is written to one, the pull-ups in the I/O ports are disabled even if the DDxn and PORTxn registers are configured to enable the pull-ups ({DDxn, PORTxn} = 0b01). See Section 9.2.1 “Configuring the Pin” on page 45 for more details about this feature.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC0x pins. Setting the COM0x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM0x1:0 to three: Setting the COM0A1:0 bits to one allows the AC0A pin to toggle on compare matches if the WGM02 bit is set. This option is not available for the OC0B pin (See Table 11-3 on page 67). The actual OC0x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC0x register at the compare match between OCR0x and TCNT0, and clearing (or setting) the OC0x register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk\_I/O}}{N \times 256}$$

The *N* variable represents the prescale factor (1, 8, 64, 256, or 1024).

The extreme values for the OCR0A register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR0A is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle.

Setting the OCR0A equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM0A1:0 bits.)

A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC0x to toggle its logical level on each compare match (COM0x1:0 = 1). The waveform generated will have a maximum frequency of $f_{OC0} = f_{clk\_I/O}/2$ when OCR0A is set to zero. This feature is similar to the OC0A toggle in CTC mode, except the double buffer feature of the output compare unit is enabled in the fast PWM mode.

11.6.4 Phase Correct PWM Mode

The phase correct PWM mode (WGM02:0 = 1 or 5) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is based on a dual-slope operation. The counter counts repeatedly from BOTTOM to TOP and then from TOP to BOTTOM. TOP is defined as 0xFF when WGM2:0 = 1, and OCR0A when WGM2:0 = 5.
In non-inverting compare output mode, the output compare (OC0x) is cleared on the compare match between TCNT0 and OCR0x while up counting, and set on the compare match while down-counting. In inverting output compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

In phase correct PWM mode the counter is incremented until the counter value matches TOP. When the counter reaches TOP, it changes the count direction. The TCNT0 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 11-7 on page 64. The TCNT0 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT0 slopes represent compare matches between OCR0x and TCNT0.

Table 11-3 shows the COM0A1:0 bit functionality when the WGM01:0 bits are set to fast PWM mode.

Table 11-3. Compare Output Mode, Fast PWM Mode(1)

COM01	COM00	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A disconnected. WGM02 = 1: Toggle OC0A on compare match.
1	0	Clear OC0A on compare match, set OC0A at TOP
1	1	Set OC0A on compare match, clear OC0A at TOP

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 11.6.3 “Fast PWM Mode” on page 62 for more details.

Table 11-4 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 11-4. Compare Output Mode, Phase Correct PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal port operation, OC0A disconnected. WGM02 = 1: Toggle OC0A on compare match.
1	0	Clear OC0A on compare match when up-counting. Set OC0A on compare match when down-counting.
1	1	Set OC0A on compare match when up-counting. Clear OC0A on compare match when down-counting.

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 11.6.4 “Phase Correct PWM Mode” on page 63 for more details.

- **Bits 5:4 – COM0B1:0: Compare Match Output B Mode**

These bits control the output compare pin (OC0B) behavior. If one or both of the COM0B1:0 bits are set, the OC0B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the data direction register (DDR) bit corresponding to the OC0B pin must be set in order to enable the output driver.

When OC0B is connected to the pin, the function of the COM0B1:0 bits depends on the WGM02:0 bit setting. Table 11-2 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 11-5. Compare Output Mode, non-PWM Mode

COM01	COM00	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Toggle OC0B on compare match
1	0	Clear OC0B on compare match
1	1	Set OC0B on compare match

12. Timer/Counter Prescaler

The Timer/Counter can be clocked directly by the system clock (by setting the CSn2:0 = 1). This provides the fastest operation, with a maximum Timer/Counter clock frequency equal to system clock frequency ($f_{CLK\_I/O}$). Alternatively, one of four taps from the prescaler can be used as a clock source. The prescaled clock has a frequency of either $f_{CLK\_I/O}/8$, $f_{CLK\_I/O}/64$, $f_{CLK\_I/O}/256$, or $f_{CLK\_I/O}/1024$.

12.1 Prescaler Reset

The prescaler is free running, i.e., operates independently of the clock select logic of the Timer/Counter. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler (6 > CSn2:0 > 1). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

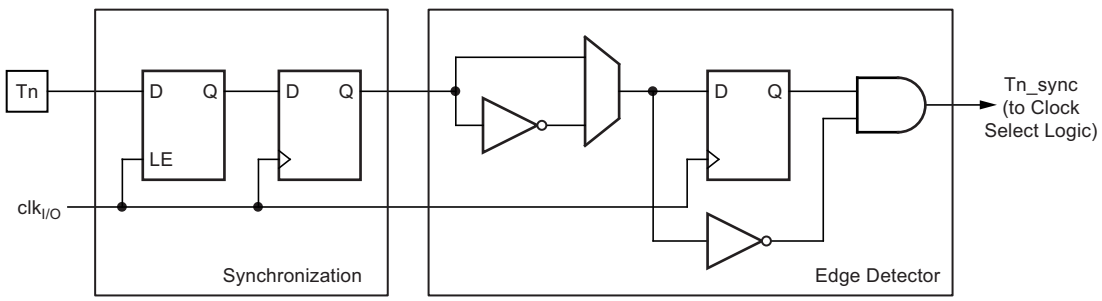
It is possible to use the prescaler reset for synchronizing the Timer/Counter to program execution.

12.2 External Clock Source

An external clock source applied to the T0 pin can be used as Timer/Counter clock (clk_{T0}). The T0 pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 12-1 shows a functional equivalent block diagram of the T0 synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ($clk_{I/O}$). The latch is transparent in the high period of the internal system clock.

The edge detector generates one clk_{T0} pulse for each positive (CSn2:0 = 7) or negative (CSn2:0 = 6) edge it detects.

Figure 12-1. T0 Pin Sampling



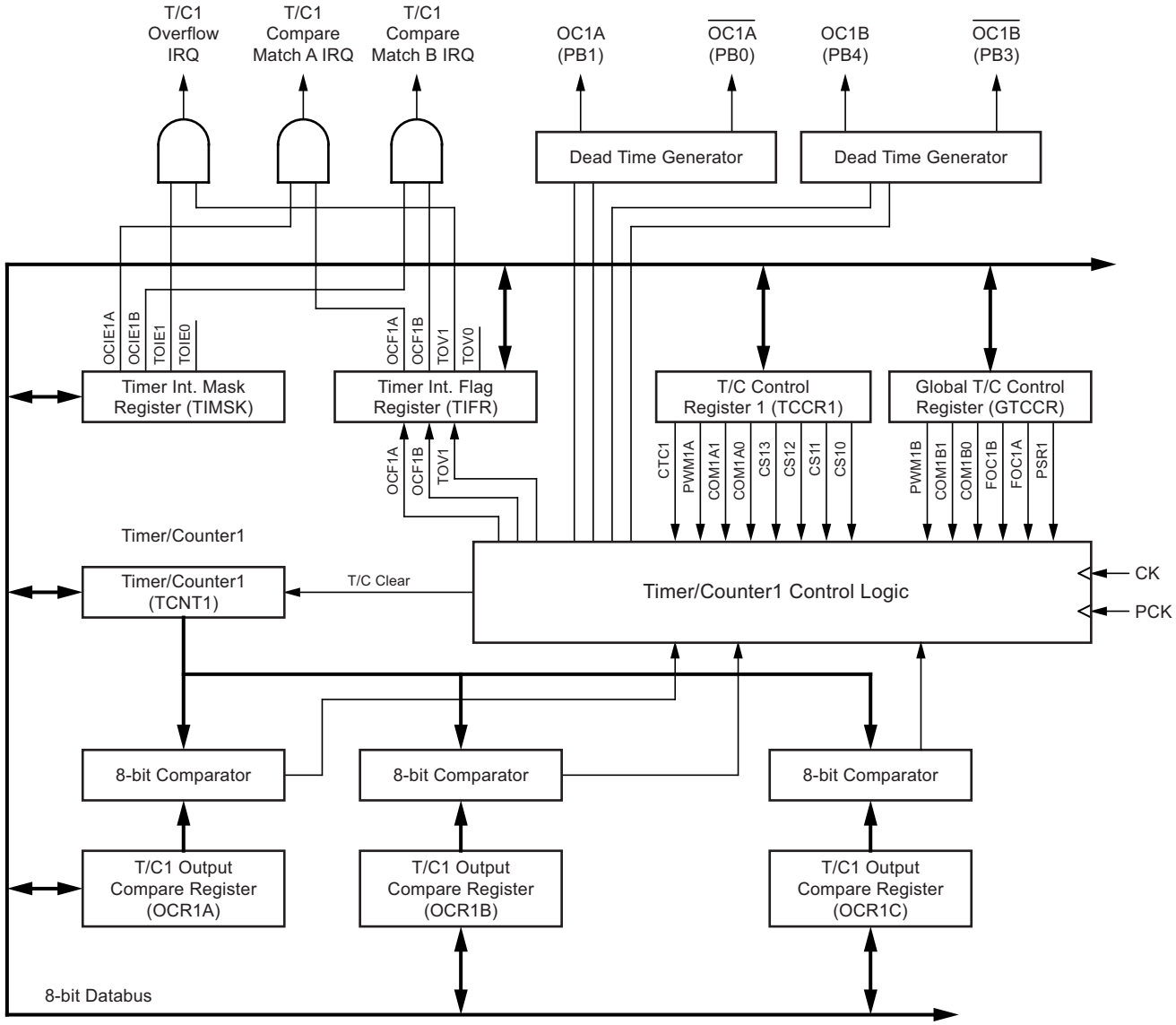
The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the T0 pin to the counter is updated.

Enabling and disabling of the clock input must be done when T0 has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

Each half period of the external clock applied must be longer than one system clock cycle to ensure correct sampling. The external clock must be guaranteed to have less than half the system clock frequency ($f_{ExtClk} < f_{clk\_I/O}/2$) given a 50/50% duty cycle. Since the edge detector uses sampling, the maximum frequency of an external clock it can detect is half the sampling frequency (nyquist sampling theorem). However, due to variation of the system clock frequency and duty cycle caused by oscillator source (crystal, resonator, and capacitors) tolerances, it is recommended that maximum frequency of an external clock source is less than $f_{clk\_I/O}/2.5$.

An external clock source can not be prescaled.

Figure 13-3. Timer/Counter1 Block Diagram



Three status flags (overflow and compare matches) are found in the Timer/Counter interrupt flag register - TIFR. control signals are found in the Timer/Counter control registers TCCR1 and GTCCR. The interrupt enable/disable settings are found in the Timer/Counter interrupt mask register - TIMSK.

The Timer/Counter1 contains three output compare registers, OCR1A, OCR1B, and OCR1C as the data source to be compared with the Timer/Counter1 contents. In normal mode the output compare functions are operational with all three output compare registers. OCR1A determines action on the OC1A pin (PB1), and it can generate timer1 OC1A interrupt in normal mode and in PWM mode. Likewise, OCR1B determines action on the OC1B pin (PB3) and it can generate timer1 OC1B interrupt in normal mode and in PWM mode. OCR1C holds the Timer/Counter maximum value, i.e. the clear on compare match value. In the normal mode an overflow interrupt (TOV1) is generated when Timer/Counter1 counts from $FF to $00, while in the PWM mode the overflow interrupt is generated when Timer/Counter1 counts either from $FF to $00 or from OCR1C to $00. The inverted PWM outputs $\overline{OC1A}$ and $\overline{OC1B}$ are not connected in normal mode.

In PWM mode, OCR1A and OCR1B provide the data values against which the timer counter value is compared. Upon compare match the PWM outputs (OC1A, $\overline{OC1A}$, OC1B, $\overline{OC1B}$) are generated. In PWM mode, the timer counter counts up to the value specified in the output compare register OCR1C and starts again from $00. This feature allows limiting the counter “full” value to a specified value, lower than $FF. Together with the many prescaler options, flexible PWM frequency selection is provided. Table 13-6 on page 84 lists clock selection and OCR1C values to obtain PWM frequencies from 20kHz to 250kHz in 10kHz steps and from 250kHz to 500kHz in 50kHz steps. Higher PWM frequencies can be obtained at the expense of resolution.

13.1.1 Timer/Counter1 Control Register - TCCR1

Bit	7	6	5	4	3	2	1	0	
$30 ($50)	CTC1	PWM1A	COM1A1	COM1A0	CS13	CS12	CS11	CS10	TCCR1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7- CTC1: Clear Timer/Counter on Compare Match**

When the CTC1 control bit is set (one), Timer/Counter1 is reset to $00 in the CPU clock cycle after a compare match with OCR1C register value. If the control bit is cleared, Timer/Counter1 continues counting and is unaffected by a compare match.

- **Bit 6- PWM1A: Pulse Width Modulator A Enable**

When set (one) this bit enables PWM mode based on comparator OCR1A in Timer/Counter1 and the counter value is reset to $00 in the CPU clock cycle after a compare match with OCR1C register value.

- **Bits 5,4 - COM1A1, COM1A0: Comparator A Output Mode, Bits 1 and 0**

The COM1A1 and COM1A0 control bits determine any output pin action following a compare match with compare register A in Timer/Counter1. Output pin actions affect pin PB1 (OC1A). Since this is an alternative function to an I/O port, the corresponding direction control bit must be set (one) in order to control an output pin. Note that $\overline{\text{OC1A}}$ is not connected in normal mode.

Table 13-1. Comparator A Mode Select

COM1A1	COM1A0	Description
0	0	Timer/Counter comparator A disconnected from output pin OC1A.
0	1	Toggle the OC1A output line.
1	0	Clear the OC1A output line.
1	1	Set the OC1A output line

In PWM mode, these bits have different functions. Refer to Table 13-4 on page 83 for a detailed description.

- **Bits 3..0 - CS13, CS12, CS11, CS10: Clock Select Bits 3, 2, 1, and 0**

The clock select bits 3, 2, 1, and 0 define the prescaling source of Timer/Counter1.

The three-wire mode timing is shown in Figure 15-3 on page 89 At the top of the figure is a USCK cycle reference. One bit is shifted into the USI shift register (USIDR) for each of these cycles. The USCK timing is shown for both external clock modes. In external clock mode 0 (USICS0 = 0), DI is sampled at positive edges, and DO is changed (data register is shifted by one) at negative edges. External clock mode 1 (USICS0 = 1) uses the opposite edges versus mode 0, i.e., samples data at negative and changes the output at positive edges. The USI clock modes corresponds to the SPI data mode 0 and 1.

Referring to the timing diagram (Figure 15-3 on page 89), a bus transfer involves the following steps:

1. The slave device and master device sets up its data output and, depending on the protocol used, enables its output driver (mark A and B). The output is set up by writing the data to be transmitted to the serial data register. Enabling of the output is done by setting the corresponding bit in the port data direction register. Note that point A and B does not have any specific order, but both must be at least one half USCK cycle before point C where the data is sampled. This must be done to ensure that the data setup requirement is satisfied. The 4-bit counter is reset to zero.
2. The master generates a clock pulse by software toggling the USCK line twice (C and D). The bit value on the slave and master's data input (DI) pin is sampled by the USI on the first edge (C), and the data output is changed on the opposite edge (D). The 4-bit counter will count both edges.
3. Step 2. is repeated eight times for a complete register (byte) transfer.
4. After eight clock pulses (i.e., 16 clock edges) the counter will overflow and indicate that the transfer is completed. The data bytes transferred must now be processed before a new transfer can be initiated. The overflow interrupt will wake up the processor if it is set to Idle mode. Depending of the protocol used the slave device can now set its output to high impedance.

15.2.2 SPI Master Operation Example

The following code demonstrates how to use the USI module as a SPI master:

```
SPITransfer:
       sts    USIDR,r16
       ldi    r16,(1<<USIOIF)
       sts    USISR,r16
       ldi    r16,(1<<USIWM0)|(1<<USICS1)|(1<<USICLK)|(1<<USITC)
SPITransfer_loop:
       sts    USICR,r16
       lds    r16, USISR
       sbrs   r16, USIOIF
       rjmp   SPITransfer_loop
       lds    r16,USIDR
       ret
```

The code is size optimized using only eight instructions (+ ret). The code example assumes that the DO and USCK pins are enabled as output in the DDRE register. The value stored in register r16 prior to the function is called is transferred to the slave device, and when the transfer is completed the data received from the slave is stored back into the r16 register.

The second and third instructions clears the USI counter overflow flag and the USI counter value. The fourth and fifth instruction set three-wire mode, positive edge shift register clock, count at USITC strobe, and toggle USCK. The loop is repeated 16 times.

Figure 17-4. ADC Timing Diagram, First Conversion (Single Conversion Mode)

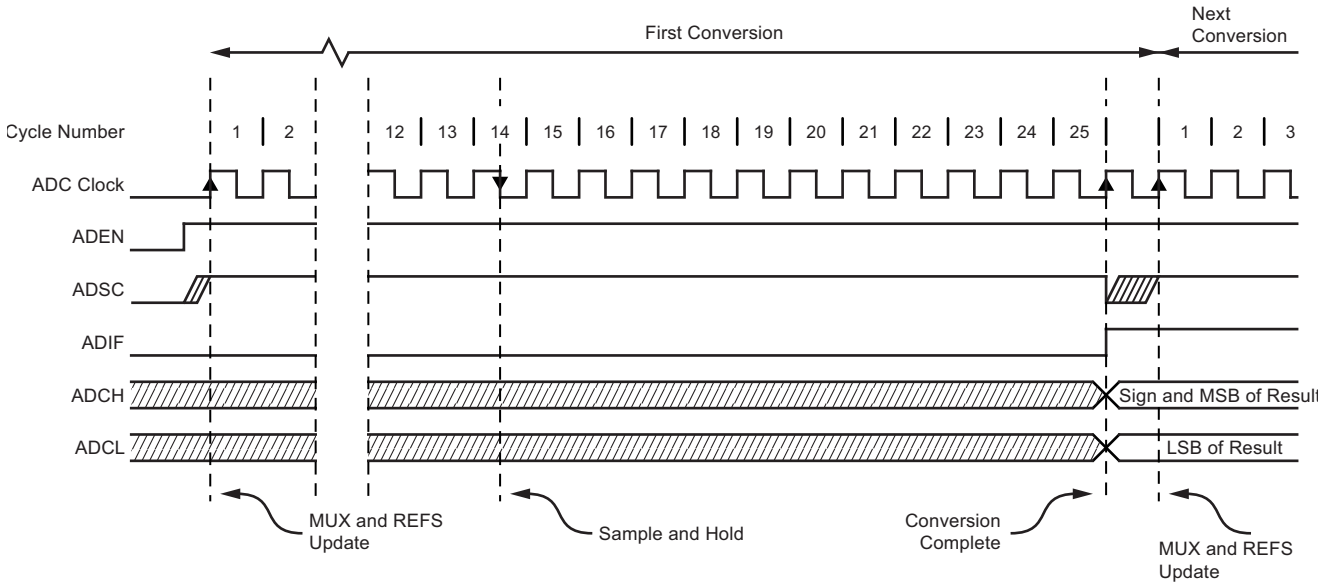
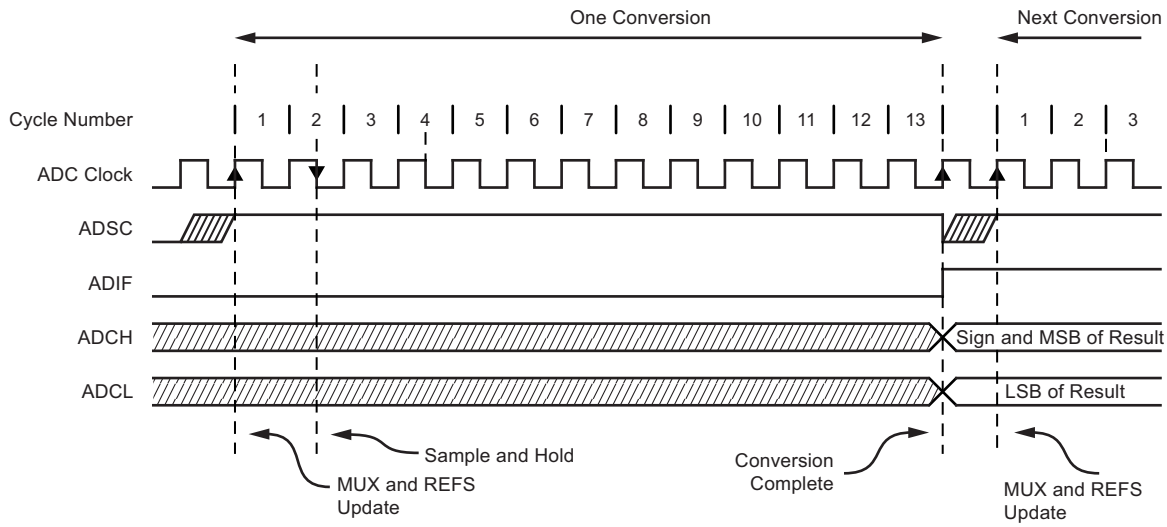


Figure 17-5. ADC Timing Diagram, Single Conversion



17.5 Changing Channel or Reference Selection

The MUX3..0 and REFS2..0 bits in the ADMUX register are single buffered through a temporary register to which the CPU has random access. This ensures that the channels and voltage reference selection only takes place at a safe point during the conversion. The channel and voltage reference selection is continuously updated until a conversion is started. Once the conversion starts, the channel and voltage reference selection is locked to ensure a sufficient sampling time for the ADC. Continuous updating resumes in the last ADC clock cycle before the conversion completes (ADIF in ADCSRA is set). Note that the conversion starts on the following rising ADC clock edge after ADSC is written. The user is thus advised not to write new channel or voltage reference selection values to ADMUX until one ADC clock cycle after ADSC is written.

If auto triggering is used, the exact time of the triggering event can be indeterministic. Special care must be taken when updating the ADMUX register, in order to control which conversion will be affected by the new settings.

If both ADATE and ADEN is written to one, an interrupt event can occur at any time. If the ADMUX register is changed in this period, the user cannot tell if the next conversion is based on the old or the new settings. ADMUX can be safely updated in the following ways:

a. When ADATE or ADEN is cleared.
b. During conversion, minimum one ADC clock cycle after the trigger event.
c. After a conversion, before the Interrupt Flag used as trigger source is cleared.

When updating ADMUX in one of these conditions, the new settings will affect the next ADC conversion.

17.5.1 ADC Input Channels

When changing channel selections, the user should observe the following guidelines to ensure that the correct channel is selected:

In single conversion mode, always select the channel before starting the conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the conversion to complete before changing the channel selection.

In free running mode, always select the channel before starting the first conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the first conversion to complete, and then change the channel selection. Since the next conversion has already started automatically, the next result will reflect the previous channel selection. Subsequent conversions will reflect the new channel selection.

17.5.2 ADC Voltage Reference

The voltage reference for the ADC (V_{REF}) indicates the conversion range for the ADC. Single ended channels that exceed V_{REF} will result in codes close to 0x3FF. V_{REF} can be selected as either V_{CC}, or internal 1.1V/2.56V voltage reference, or external AREF pin. The first ADC conversion result after switching voltage reference source may be inaccurate, and the user is advised to discard this result.

17.6 ADC Noise Canceler

The ADC features a noise canceler that enables conversion during sleep mode to reduce noise induced from the CPU core and other I/O peripherals. The noise canceler can be used with ADC noise reduction and Idle mode. To make use of this feature, the following procedure should be used:

a. Make sure that the ADC is enabled and is not busy converting. Single conversion mode must be selected and the ADC conversion complete interrupt must be enabled.
b. Enter ADC noise reduction mode (or idle mode). The ADC will start a conversion once the CPU has been halted.
c. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the CPU and execute the ADC conversion complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC conversion complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

Note that the ADC will not be automatically turned off when entering other sleep modes than Idle mode and ADC noise reduction mode. The user is advised to write zero to ADEN before entering such sleep modes to avoid excessive power consumption.

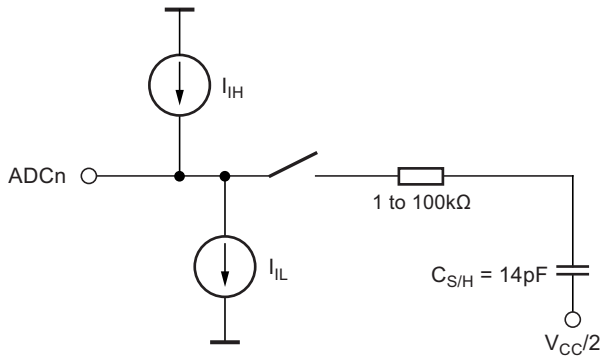
17.6.1 Analog Input Circuitry

The analog input circuitry for single ended channels is illustrated in Figure 17-8. An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10 kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long time the source needs to charge the S/H capacitor, with can vary widely. The user is recommended to only use low impedant sources with slowly varying signals, since this minimizes the required charge transfer to the S/H capacitor.

Signal components higher than the nyquist frequency ($f_{ADC}/2$) should not be present to avoid distortion from unpredictable signal convolution. The user is advised to remove high frequency components with a low-pass filter before applying the signals as inputs to the ADC.

Figure 17-8. Analog Input Circuitry



17.6.2 Analog Noise Canceling Techniques

Digital circuitry inside and outside the device generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

a. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.

b. Use the ADC noise canceler function to reduce induced noise from the CPU.

c. If any port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress.

19. Self-Programming the Flash

The device provides a self-programming mechanism for downloading and uploading program code by the MCU itself. The self-programming can use any available data interface and associated protocol to read code and write (program) that code into the program memory.

The program memory is updated in a page by page fashion. Before programming a page with the data stored in the temporary page buffer, the page must be erased. The temporary page buffer is filled one word at a time using SPM and the buffer can be filled either before the page erase command or between a page erase and a page write operation:

Alternative 1, fill the buffer before a page erase

- Fill temporary page buffer
- Perform a page erase
- Perform a page write

Alternative 2, fill the buffer after page erase

- Perform a page erase
- Fill temporary page buffer
- Perform a page write

If only a part of the page needs to be changed, the rest of the page must be stored (for example in the temporary page buffer) before the erase, and then be re-written. When using alternative 1, the boot loader provides an effective read-modify-write feature which allows the user software to first read the page, do the necessary changes, and then write back the modified data. If alternative 2 is used, it is not possible to read the old data while loading since the page is already erased. The temporary page buffer can be accessed in a random sequence. It is essential that the page address used in both the page erase and page write operation is addressing the same page.

19.1 Performing Page Erase by SPM

To execute page erase, set up the address in the Z-pointer, write “00000011” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE in the Z-register. Other bits in the Z-pointer will be ignored during this operation.

- The CPU is halted during the page erase operation.

19.2 Filling the Temporary Buffer (Page Loading)

To write an instruction word, set up the address in the Z-pointer and data in R1:R0, write “00000001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The content of PCWORD in the Z-register is used to address the data in the temporary buffer. The temporary buffer will auto-erase after a Page Write operation or by writing the CTPB bit in SPMCSR. It is also erased after a system reset. Note that it is not possible to write more than one time to each address without erasing the temporary buffer.

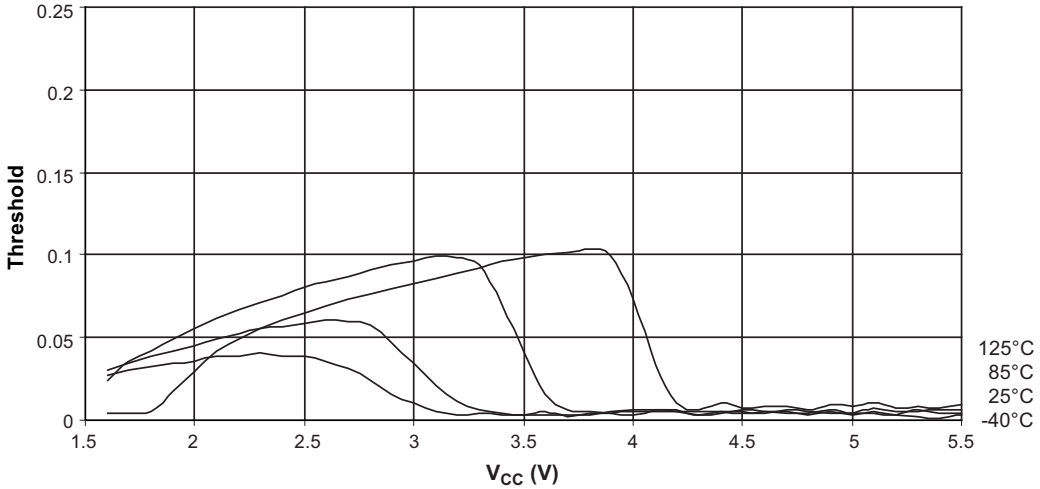
If the EEPROM is written in the middle of an SPM page load operation, all data loaded will be lost.

19.3 Performing a Page Write

To execute page write, set up the address in the Z-pointer, write “00000101” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

- The CPU is halted during the page write operation.

Figure 22-30. Reset Input Pin Hysteresis versus V_{CC}



22.7 BOD Thresholds and Analog Comparator Offset

Figure 22-31. BOD Thresholds versus Temperature (BODLEVEL Is 4.3V)

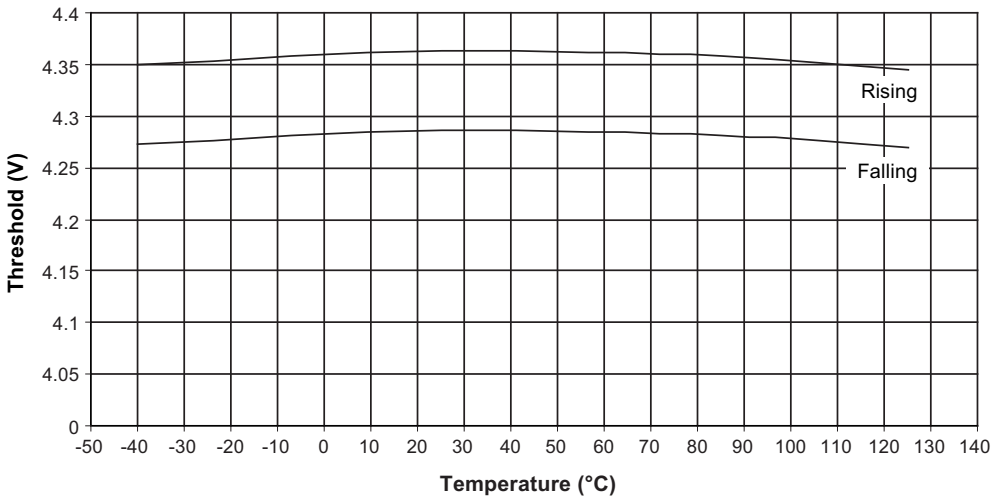


Figure 22-36. Calibrated 8MHz RC Oscillator Frequency versus Temperature

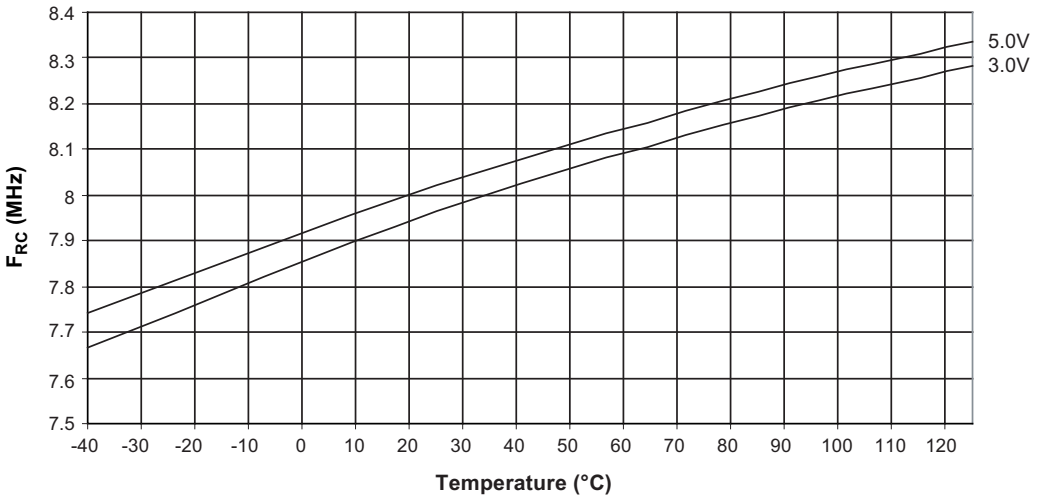


Figure 22-37. Calibrated 8MHz RC Oscillator Frequency versus V_{CC}

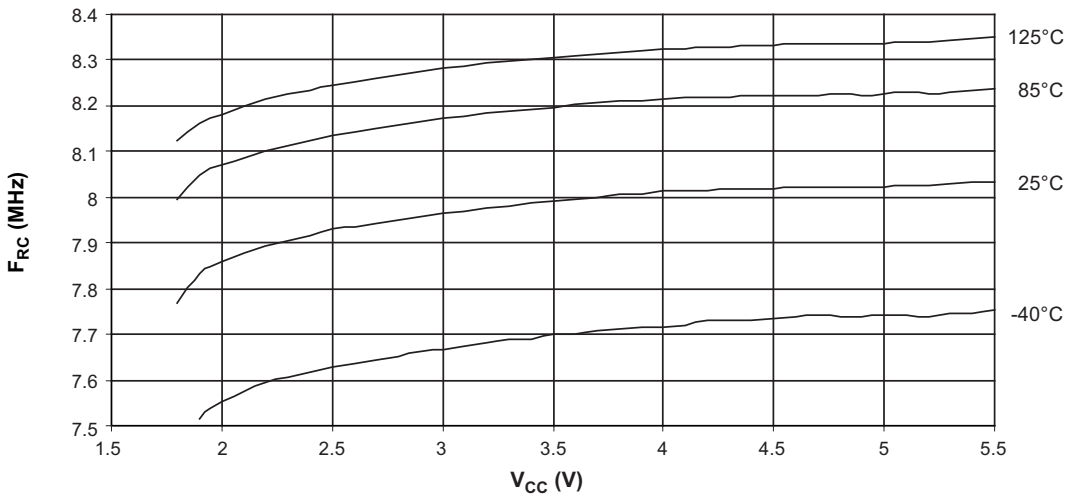
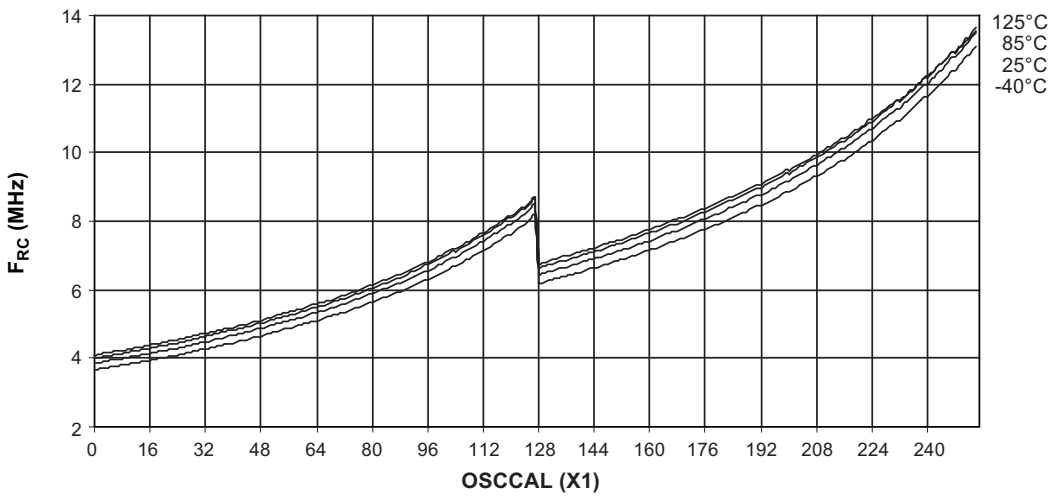


Figure 22-38. Calibrated 8MHz RC Oscillator Frequency versus OSCCAL Value



22.10 Current Consumption in Reset and Reset Pulse width

Figure 22-41. Reset Supply Current versus V_{CC} (0.1 to 1.0MHz, Excluding Current through the Reset Pull-up)

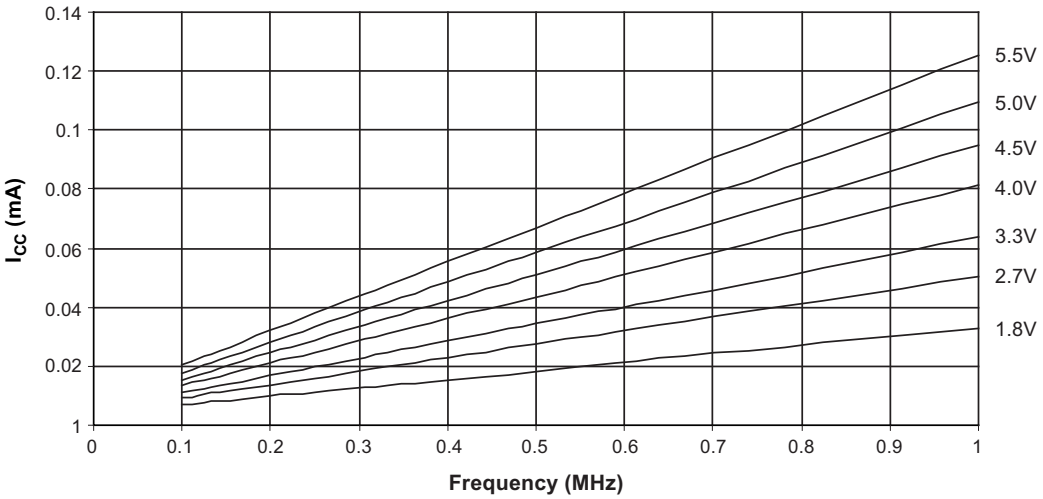


Figure 22-42. Reset Supply Current versus V_{CC} (1 to 24MHz, Excluding Current through the Reset Pull-up)