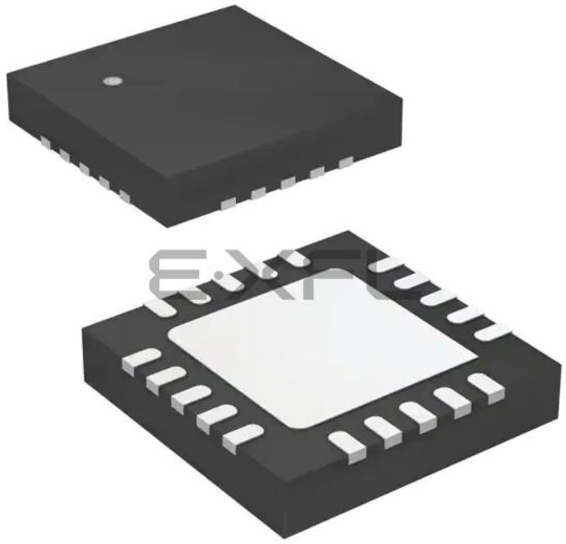


What is "Embedded - Microcontrollers"?

"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	20-WFQFN Exposed Pad
Supplier Device Package	20-QFN-EP (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny85-15mt

2. About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

3. AVR CPU Core

3.1 Introduction

This section discusses the AVR® core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

3.2 Architectural Overview

Figure 3-1. Block Diagram of the AVR Architecture

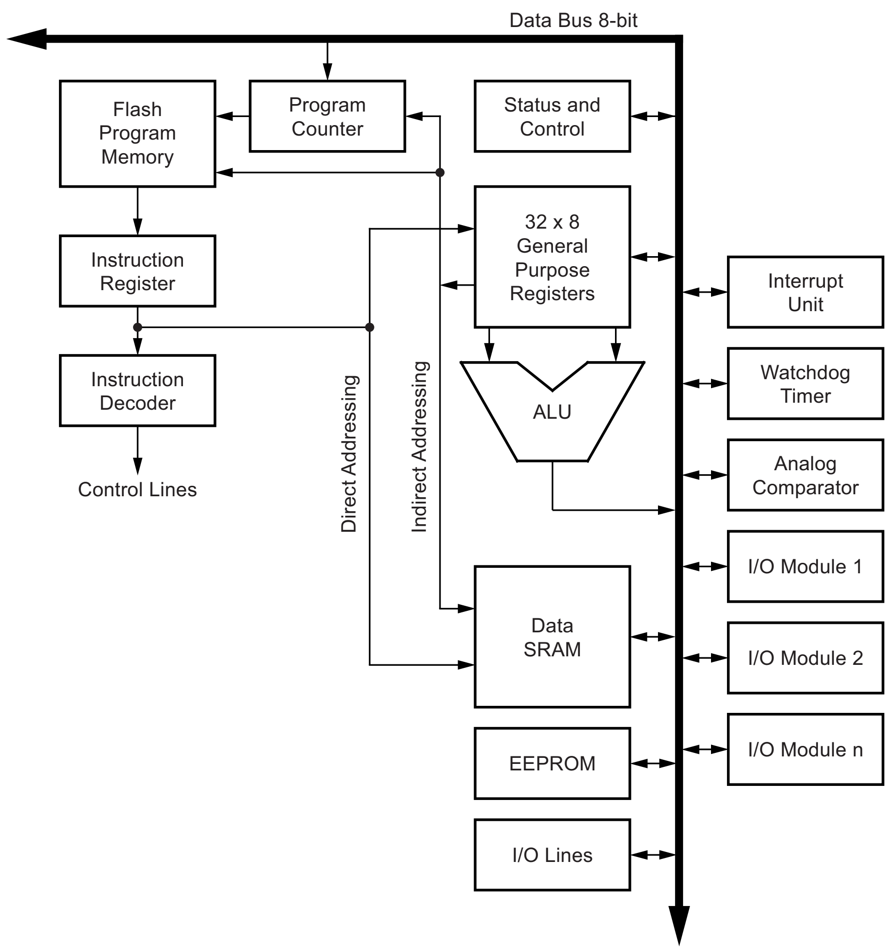


Figure 3-2 shows the structure of the 32 general purpose working registers in the CPU.

Figure 3-2. AVR CPU General Purpose Working Registers

	7 — 0	Addr.	
	R0	0x00	
	R1	0x01	
	R2	0x02	
	…		
	R13	0x0D	
General	R14	0x0E	
Purpose	R15	0x0F	
Working	R16	0x10	
Registers	R17	0x11	
	…		
	R26	0x1A	X-register low byte
	R27	0x1B	X-register high byte
	R28	0x1C	Y-register low byte
	R29	0x1D	Y-register high byte
	R30	0x1E	Z-register low byte
	R31	0x1F	Z-register high byte

Most of the instructions operating on the register file have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 3-2, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user data space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y- and Z-pointer registers can be set to index any register in the file.

3.5.1 The X-register, Y-register, and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers X, Y, and Z are defined as described in Figure 3-3.

Figure 3-3. The X-, Y-, and Z-registers

	15	XH	XL	0
X-register	7	0	7	0
	R27 (0x1B)		R26 (0x1A)	
	15	YH	YL	0
Y-register	7	0	7	0
	R29 (0x1D)		R28 (0x1C)	
	15	ZH	ZL	0
Z-register	7	0	7	0
	R31 (0x1F)		R30 (0x1E)	

In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement (see the instruction set reference for details).

4.3.8 Erase

To erase a byte, the address must be written to EEAR. If the EEPMn bits are 0b01, writing the EEPE (within four cycles after EEMPE is written) will trigger the erase operation only (programming time is given in Table 19-1 on page 122). The EEPE bit remains set until the erase operation completes. While the device is busy programming, it is not possible to do any other EEPROM operations.

4.3.9 Write

To write a location, the user must write the address into EEAR and the data into EEDR. If the EEPMn bits are 0b10, writing the EEPE (within four cycles after EEMPE is written) will trigger the write operation only (programming time is given in Table 19-1 on page 122). The EEPE bit remains set until the write operation completes. If the location to be written has not been erased before write, the data that is stored must be considered as lost. While the device is busy with programming, it is not possible to do any other EEPROM operations.

The calibrated oscillator is used to time the EEPROM accesses. Make sure the oscillator frequency is within the requirements described in Section 5.6.1 "Oscillator Calibration Register – OSCCAL" on page 24.

The following code examples show one assembly and one C function for erase, write, or atomic write of the EEPROM. The examples assume that interrupts are controlled (e.g., by disabling interrupts globally) so that no interrupts will occur during execution of these functions

Assembly Code Example

```
EEPROM_write:
        ; Wait for completion of previous write
        sbic    EECR,EEPE
        rjmp    EEPROM_write
        ; Set Programming mode
        ldi    r16, (0<<EEPM1)|(0<<EEPM0)
        out    EECR, r16
        ; Set up address (r17) in address register
        out     EEARL, r17
        ; Write data (r16) to data register
        out     EEDR,r16
        ; Write logical one to EEMWE
        sbi     EECR,EEMWE
        ; Start eeprom write by setting EEWE
        sbi     EECR,EEWE
        ret
```

C Code Example

```
void EEPROM_write(unsigned char ucAddress, unsigned char ucData)
{
        /* Wait for completion of previous write */
        while(EECR & (1<<EEPE))
        ;
        /* Set Programming mode */
        EECR = (0<<EEPM1)|(0>>EEPM0)
        /* Set up address and data registers */
        EEARL = ucAddress;
        EEDR = ucData;
        /* Write logical one to EEMWE */
        EECR |= (1<<EEMWE);
        /* Start eeprom write by setting EEWE */
        EECR |= (1<<EEWE);
}
```

5.2 Clock Sources

The device has the following clock source options, selectable by flash fuse bits as shown below. The clock from the selected source is input to the AVR® clock generator, and routed to the appropriate modules.

Table 5-1. Device Clocking Options Select[1]

Device Clocking Option	CKSEL3..0
External clock	0000
PLL clock	0001
Calibrated internal RC oscillator 8.0MHz	0010
Watchdog oscillator 128kHz	0100
External low-frequency crystal	0110
External crystal/ceramic resonator	1000-1111
Reserved	0101, 0111, 0011

Note: 1. For all fuses “1” means unprogrammed while “0” means programmed.

The various choices for each clocking option is given in the following sections. When the CPU wakes up from power-down or power-save, the selected clock source is used to time the start-up, ensuring stable oscillator operation before instruction execution starts. When the CPU starts from reset, there is an additional delay allowing the power to reach a stable level before commencing normal operation. The watchdog oscillator is used for timing this real-time part of the start-up time. The number of WDT oscillator cycles used for each time-out is shown in Table 5-2.

Table 5-2. Number of Watchdog Oscillator Cycles

Typ Time-out	Number of Cycles
4ms	512
64ms	8K (8,192)

5.3 Default Clock Source

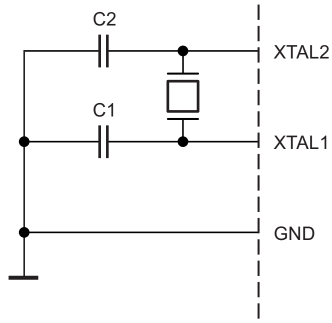
The device is shipped with CKSEL = “0010”, SUT = “10”, and CKDIV8 programmed. The default clock source setting is therefore the internal RC oscillator running at 8MHz with longest start-up time and an initial system clock prescaling of 8. This default setting ensures that all users can make their desired clock source setting using an in-system or high-voltage programmer.

5.4 Crystal Oscillator

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 5-3. Either a quartz crystal or a ceramic resonator may be used.

C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 5-3. For ceramic resonators, the capacitor values given by the manufacturer should be used.

Figure 5-3. Crystal Oscillator Connections



The oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1 as shown in Table 5-3.

Table 5-3. Crystal Oscillator Operating Modes

CKSEL3..1	Frequency Range (MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals (pF)
100(1)	0.4 to 0.9	–
101	0.9 to 3.0	12 to 22
110	3.0 to 8.0	12 to 22
111	8.0 –	12 to 22

Note: 1. This option should not be used with crystals, only with ceramic resonators.

The CKSEL0 fuse together with the SUT1..0 fuses select the start-up times as shown in Table 5-4.

Table 5-4. Start-up Times for the Crystal Oscillator Clock Selection

CKSEL0	SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
0	00	258 CK(1)	14CK + 4ms	Ceramic resonator, fast rising power
0	01	258 CK(1)	14CK + 64ms	Ceramic resonator, slowly rising power
0	10	1KCK(2)	14CK	Ceramic resonator, BOD enabled
0	11	1KCK(2)	14CK + 4ms	Ceramic resonator, fast rising power
1	00	1KCK(2)	14CK + 64ms	Ceramic resonator, slowly rising power
1	01	16KCK	14CK	Crystal oscillator, BOD enabled
1	10	16KCK	14CK + 4ms	Crystal oscillator, fast rising power
1	11	16KCK	14CK + 64ms	Crystal oscillator, slowly rising power

Notes: 1. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.

2. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

7. System Control and Reset

7.1 Resetting the AVR

During reset, all I/O registers are set to their initial values, and the program starts execution from the reset vector. The instruction placed at the reset vector must be a RJMP – relative jump – instruction to the reset handling routine. 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 circuit diagram in Figure 7-1 on page 33 shows the reset logic. Table on page 34 defines the electrical parameters of the reset circuitry.

The I/O ports of the AVR® are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset. This allows the power to reach a stable level before normal operation starts. The time-out period of the delay counter is defined by the user through the SUT and CKSEL fuses. The different selections for the delay period are presented in Section 5.2 “Clock Sources” on page 21.

7.2 Reset Sources

The Atmel® ATtiny25/45/85 has four sources of reset:

- Power-on reset. The MCU is reset when the supply voltage is below the power-on reset threshold (V_{POT}).
- External reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length.
- Watchdog reset. The MCU is reset when the watchdog timer period expires and the watchdog is enabled.
- Brown-out reset. The MCU is reset when the supply voltage V_{CC} is below the brown-out reset threshold (V_{BOT}) and the brown-out detector is enabled.

7.7 MCU Status Register – MCUSR

The MCU status register provides information on which reset source caused an MCU reset.

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	WDRF	BORF	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bits 7..4 – Res: Reserved Bits**

These bits are reserved bits in the Atmel® ATtiny25/45/85 and will always read as zero.

- **Bit 3 – WDRF: Watchdog Reset Flag**

This bit is set if a watchdog reset occurs. The bit is reset by a power-on reset, or by writing a logic zero to the flag.

- **Bit 2 – BORF: Brown-out Reset Flag**

This bit is set if a brown-out reset occurs. The bit is reset by a power-on reset, or by writing a logic zero to the flag.

- **Bit 1 – EXTRF: External Reset Flag**

This bit is set if an external reset occurs. The bit is reset by a power-on reset, or by writing a logic zero to the flag.

- **Bit 0 – PORF: Power-on Reset Flag**

This bit is set if a power-on reset occurs. The bit is reset only by writing a logic zero to the flag.

To make use of the reset flags to identify a reset condition, the user should read and then reset the MCUSR as early as possible in the program. If the register is cleared before another reset occurs, the source of the reset can be found by examining the reset flags.

7.8 Internal Voltage Reference

Atmel ATtiny25/45/85 features an internal bandgap reference. This reference is used for brown-out detection, and it can be used as an input to the analog comparator or the ADC.

7.8.1 Voltage Reference Enable Signals and Start-up Time

The voltage reference has a start-up time that may influence the way it should be used. The start-up time is given in Table 7-4. To save power, the reference is not always turned on. The reference is on during the following situations:

1. When the BOD is enabled (by programming the BODLEVEL [2..0] fuse bits).
2. When the bandgap reference is connected to the analog comparator (by setting the ACBG bit in ACSR).
3. When the ADC is enabled.

Thus, when the BOD is not enabled, after setting the ACBG bit or enabling the ADC, the user must always allow the reference to start up before the output from the analog comparator or ADC is used. To reduce power consumption in power-down mode, the user can avoid the three conditions above to ensure that the reference is turned off before entering power-down mode.

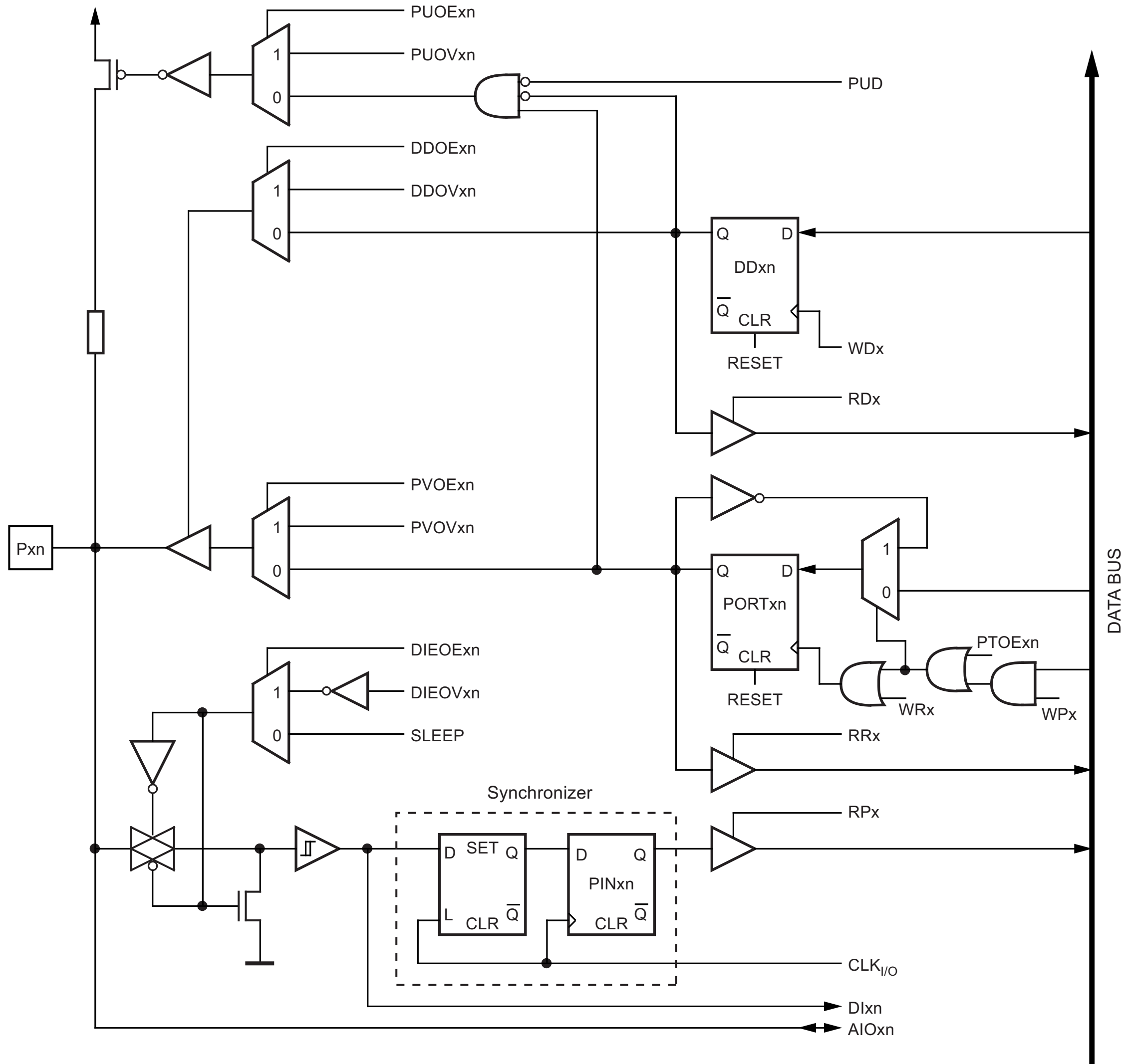
Table 7-4. Internal Voltage Reference Characteristics

Parameter	Condition	Symbol	Min	Typ	Max	Units
Bandgap reference voltage	V_{CC} = 1.1V/2.7V, T_A = 25°C	V_{BG}	1.0	1.1	1.2	V
Bandgap reference start-up time	V_{CC} = 2.7V, T_A = 25°C	t_{BG}		40	70	µs
Bandgap reference current consumption	V_{CC} = 2.7V, T_A = 25°C	I_{BG}		15		µA

9.3 Alternate Port Functions

Most port pins have alternate functions in addition to being general digital I/Os. Figure 9-5 shows how the port pin control signals from the simplified Figure 9-2 can be overridden by alternate functions. The overriding signals may not be present in all port pins, but the figure serves as a generic description applicable to all port pins in the AVR® microcontroller family.

Figure 9-5. Alternate Port Functions[1]



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE	PUD:	PULL-UP DISABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE	WDx:	WRITE DDRx
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE	RDx:	READ DDRx
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RRx:	READ PORTx REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WRx:	WRITE PORTx
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	RPx:	READ PORTx PIN
DIEOExn:	Pxn DIGITAL INPUT ENABLE OVERRIDE ENABLE	WPx:	WRITE PINx
DIEOVxn:	Pxn DIGITAL INPUT ENABLE OVERRIDE VALUE	$CLK_{I/O}$:	I/O CLOCK
SLEEP:	SLEEP CONTROL	DIxn:	DIGITAL INPUT PIN n ON PORTx
PTOExn:	Pxn, PORT TOGGLE OVERRIDE ENABLE	AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

11. 8-bit Timer/Counter0 with PWM

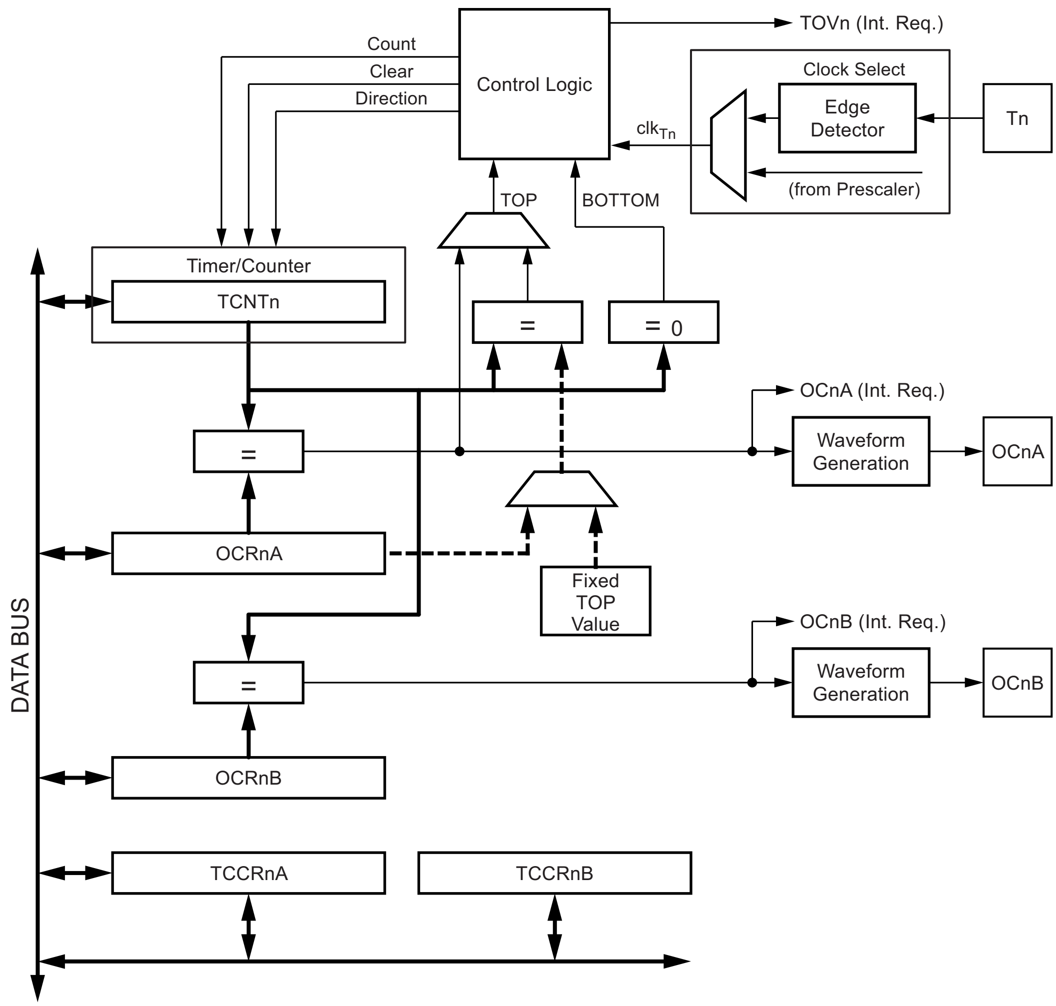
Timer/Counter0 is a general purpose 8-bit Timer/Counter module, with two independent output compare units, and with PWM support. It allows accurate program execution timing (event management) and wave generation. The main features are:

- Two independent output compare units
- Double buffered output compare registers
- Clear timer on compare match (auto reload)
- Glitch free, phase correct pulse width modulator (PWM)
- Variable PWM period
- Frequency generator
- Three independent interrupt sources (TOV0, OCF0A, and OCF0B)

11.1 Overview

A simplified block diagram of the 8-bit Timer/Counter is shown in Figure 11-1. For the actual placement of I/O pins, refer to Figure 1 on page 2. CPU accessible I/O registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O register and bit locations are listed in Section 11.8 "8-bit Timer/Counter Register Description" on page 66.

Figure 11-1. 8-bit Timer/Counter Block Diagram



The OCR0x registers are double buffered when using any of the pulse width modulation (PWM) modes. For the normal and clear timer on compare (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the update of the OCR0x compare registers to either top or bottom of the counting sequence. The synchronization prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR0x register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR0x buffer register, and if double buffering is disabled the CPU will access the OCR0x directly.

11.4.1 Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a one to the force output compare (FOC0x) bit. Forcing compare match will not set the OCF0x flag or reload/clear the timer, but the OC0x pin will be updated as if a real compare match had occurred (the COM0x1:0 bits settings define whether the OC0x pin is set, cleared or toggled).

11.4.2 Compare Match Blocking by TCNT0 Write

All CPU write operations to the TCNT0 register will block any compare match that occur in the next timer clock cycle, even when the timer is stopped. This feature allows OCR0x to be initialized to the same value as TCNT0 without triggering an interrupt when the Timer/Counter clock is enabled.

11.4.3 Using the Output Compare Unit

Since writing TCNT0 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT0 when using the output compare unit, independently of whether the Timer/Counter is running or not. If the value written to TCNT0 equals the OCR0x value, the compare match will be missed, resulting in incorrect waveform generation. Similarly, do not write the TCNT0 value equal to BOTTOM when the counter is down-counting.

The setup of the OC0x should be performed before setting the data direction register for the port pin to output. The easiest way of setting the OC0x value is to use the force output compare (FOC0x) strobe bits in normal mode. The OC0x registers keep their values even when changing between waveform generation modes.

Be aware that the COM0x1:0 bits are not double buffered together with the compare value. Changing the COM0x1:0 bits will take effect immediately.

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.

15.4.4 USI Control Register – USICR

Bit	7	6	5	4	3	2	1	0	
	USISIE	USIOIE	USIWM1	USIWM0	USICS1	USICS0	USICLK	USITC	USICR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	W	W	
Initial Value	0	0	0	0	0	0	0	0	

The control register includes interrupt enable control, wire mode setting, clock select setting, and clock strobe.

- **Bit 7 – USISIE: Start Condition Interrupt Enable**

Setting this bit to one enables the start condition detector interrupt. If there is a pending interrupt when the USISIE and the global interrupt enable flag is set to one, this will immediately be executed. Refer to the USISIF bit description on page 95 for further details.

- **Bit 6 – USIOIE: Counter Overflow Interrupt Enable**

Setting this bit to one enables the counter overflow interrupt. If there is a pending interrupt when the USIOIE and the global interrupt enable flag is set to one, this will immediately be executed. Refer to the USIOIF bit description on page 95 for further details.

- **Bit 5..4 – USIWM1..0: Wire Mode**

These bits set the type of wire mode to be used. Basically only the function of the outputs are affected by these bits. Data and clock inputs are not affected by the mode selected and will always have the same function. The counter and shift register can therefore be clocked externally, and data input sampled, even when outputs are disabled. The relations between USIWM1..0 and the USI operation is summarized in Table 15-1 on page 96.

Table 15-1. Relations between USIWM1..0 and the USI Operation

USIWM1	USIWM0	Description
0	0	Outputs, clock hold, and start detector disabled. Port pins operates as normal.
0	1	Three-wire mode. Uses DO, DI, and USCK pins. The *Data Output* (DO) pin overrides the corresponding bit in the PORT Register in this mode. However, the corresponding DDR bit still controls the data direction. When the port pin is set as input the pins pull-up is controlled by the PORT bit. The *Data Input* (DI) and *Serial Clock* (USCK) pins do not affect the normal port operation. When operating as master, clock pulses are software generated by toggling the PORT Register, while the data direction is set to output. The USITC bit in the USICR Register can be used for this purpose.
1	0	Two-wire mode. Uses SDA (DI) and SCL (USCK) pins(1). The *Serial Data* (SDA) and the *Serial Clock* (SCL) pins are bi-directional and uses open-collector output drives. The output drivers are enabled by setting the corresponding bit for SDA and SCL in the DDR Register. When the output driver is enabled for the SDA pin, the output driver will force the line SDA low if the output of the Shift Register or the corresponding bit in the PORT Register is zero. Otherwise the SDA line will not be driven (i.e., it is released). When the SCL pin output driver is enabled the SCL line will be forced low if the corresponding bit in the PORT Register is zero, or by the start detector. Otherwise the SCL line will not be driven. The SCL line is held low when a start detector detects a start condition and the output is enabled. Clearing the Start Condition Flag (USISIF) releases the line. The SDA and SCL pin inputs is not affected by enabling this mode. Pull-ups on the SDA and SCL port pin are disabled in Two-wire mode.
1	1	Two-wire mode. Uses SDA and SCL pins. Same operation as for the Two-wire mode described above, except that the SCL line is also held low when a counter overflow occurs, and is held low until the Counter Overflow Flag (USIOIF) is cleared.

Note: 1. The DI and USCK pins are renamed to serial data (SDA) and serial clock (SCL) respectively to avoid confusion between the modes of operation.

- **Bit 6 – ACBG: Analog Comparator Bandgap Select**

When this bit is set an internal 1.1V/2.56V reference voltage replaces the positive input to the analog comparator. The selection of the internal voltage reference is done by writing the REFS2..0 bits in ADMUX register. When this bit is cleared, AIN0 is applied to the positive input of the analog comparator.

- **Bit 5 – ACO: Analog Comparator Output**

The output of the analog comparator is synchronized and then directly connected to ACO. The synchronization introduces a delay of 1 - 2 clock cycles.

- **Bit 4 – ACI: Analog Comparator Interrupt Flag**

This bit is set by hardware when a comparator output event triggers the interrupt mode defined by ACIS1 and ACIS0. The analog comparator interrupt routine is executed if the ACIE bit is set and the I-bit in SREG is set. ACI is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ACI is cleared by writing a logic one to the flag.

- **Bit 3 – ACIE: Analog Comparator Interrupt Enable**

When the ACIE bit is written logic one and the I-bit in the status register is set, the analog comparator interrupt is activated. When written logic zero, the interrupt is disabled.

- **Bit 2 – Res: Reserved Bit**

This bit is a reserved bit in the Atmel® ATtiny25/45/85 and will always read as zero.

- **Bits 1, 0 – ACIS1, ACIS0: Analog Comparator Interrupt Mode Select**

These bits determine which comparator events that trigger the analog comparator interrupt. The different settings are shown in Table 16-1.

Table 16-1. ACIS1/ACIS0 Settings

ACIS1	ACIS0	Interrupt Mode
0	0	Comparator interrupt on output toggle.
0	1	Reserved
1	0	Comparator interrupt on falling output edge.
1	1	Comparator interrupt on rising output edge.

When changing the ACIS1/ACIS0 bits, the analog comparator interrupt must be disabled by clearing its interrupt enable bit in the ACSR register. Otherwise an interrupt can occur when the bits are changed.

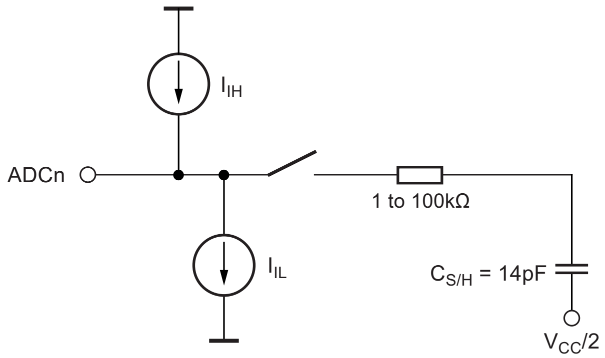
17.6.1 Analog Input Circuitry

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.

20.9 High-voltage Serial Programming Characteristics

Figure 20-7. High-voltage Serial Programming Timing

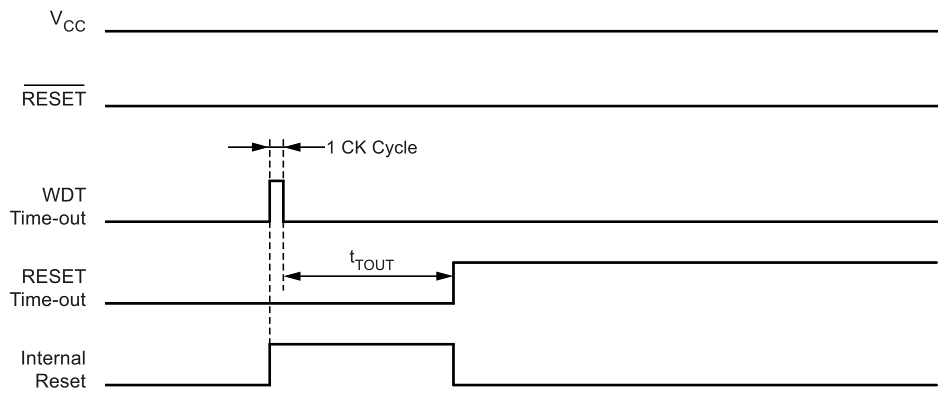


Table 20-17. High-voltage Serial Programming Characteristics T_A = 25°C ±10%, V_{CC} = 5.0V ±10% (Unless otherwise noted)

Parameter	Symbol	Min	Typ	Max	Units
SCI (PB3) pulse width high	t_{SHSL}	110			ns
SCI (PB3) pulse width low	t_{SLSH}	110			ns
SDI (PB0), SII (PB1) valid to SCI (PB3) high	t_{IVSH}	50			ns
SDI (PB0), SII (PB1) hold after SCI (PB3) high	t_{SHIX}	50			ns
SCI (PB3) high to SDO (PB2) valid	t_{SHOV}		16		ns
Wait after Instr. 3 for write fuse bits	t_{WLWH_PFB}		2.5		ms

Table 21-1. DC Characteristics T_A = – 40°C to 125°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)(1) (Continued)

Parameter	Condition	Symbol	Min.(2)	Typ.	Max.(3)	Units
Input leakage Current I/O pin except RESET	Vcc = 5.5V, pin low (absolute value)	I_{IL}			50	nA
Input leakage current I/O pin except RESET	Vcc = 5.5V, pin high (absolute value)	I_{IH}			50	nA
Reset pull-up resistor		R_{RST}	30		60	kΩ
I/O pin pull-up resistor		R_{pu}	20		50	kΩ
Power supply current	Active 4MHz, V_{CC} = 3V	I_{CC}(6)		1.25	3	mA
	Active 8MHz, V_{CC} = 5V			5	10	mA
	Active 16MHz, V_{CC} = 5V			10	15	mA
	Idle 4MHz, V_{CC} = 3V			0.4	0.5	mA
	Idle 8MHz, V_{CC} = 5V			1.2	2	mA
	Idle 16MHz, V_{CC} = 5V			2.5	5	mA
Power-down mode(7)	WDT enabled, V_{CC} = 3V			5	30	µA
	WDT disabled, V_{CC} = 3V			2	24	µA
	WDT enabled, V_{CC} = 5V			9	50	µA
	WDT disabled, V_{CC} = 5V			3	36	µA

Notes:
1. All DC characteristics contained in this data sheet result from actual silicon characterization.
2. “Max” means the highest value where the pin is guaranteed to be read as low.
3. “Min” means the lowest value where the pin is guaranteed to be read as high.
4. Although each I/O port can sink more than the test conditions (8mA at V_{CC} = 5V, 5mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   1] The sum of all IOL, for all ports, should not exceed 60mA.
   If IOL exceeds the test condition, VOL may exceed the related specification. pins are not guaranteed to sink current greater than the listed test condition.
5. Although each I/O port can source more than the test conditions (8mA at V_{CC} = 5V, 5mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   1] The sum of all IOH, for all ports, should not exceed 60mA.
   If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.
6. All I/O modules are turned off (PRR = 0xFF) for all I_{CC} values.
7. Brown-out detection (BOD) disabled.

21.2 External Clock Drive Waveforms

Figure 21-1. External Clock Drive Waveforms

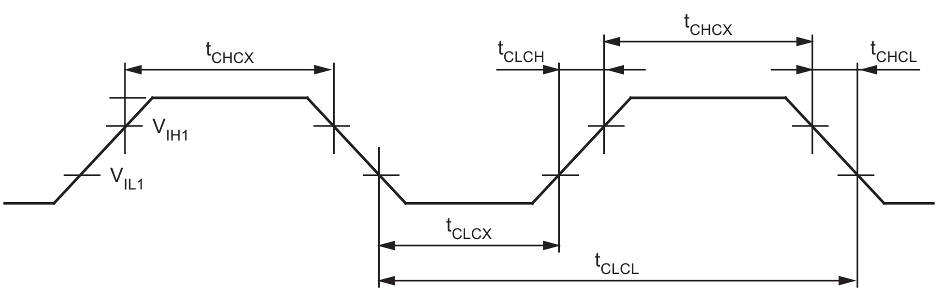
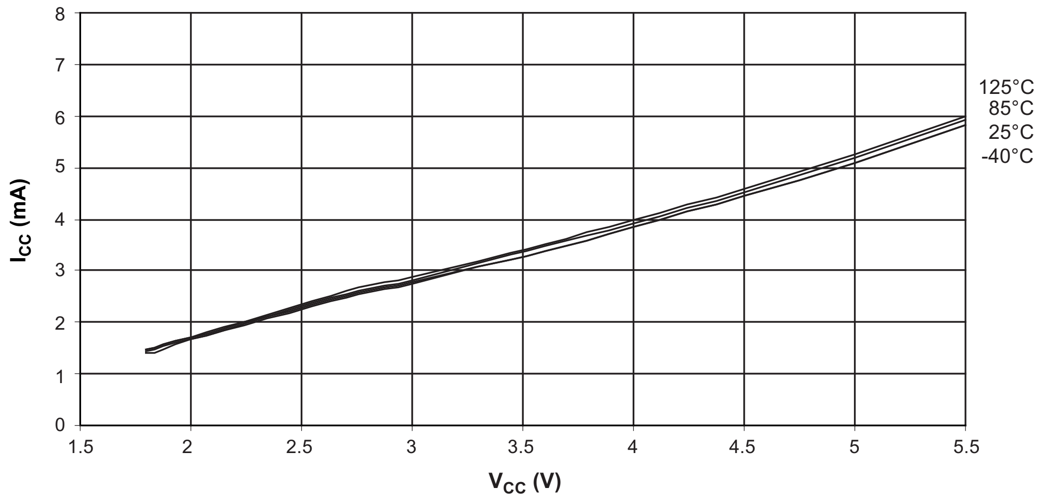


Figure 22-5. Active Supply Current versus V_{CC} (Internal RC Oscillator, 8MHz)



22.2 Idle Supply Current

Figure 22-6. Idle Supply Current versus Frequency (0.1 to 1.0MHz)

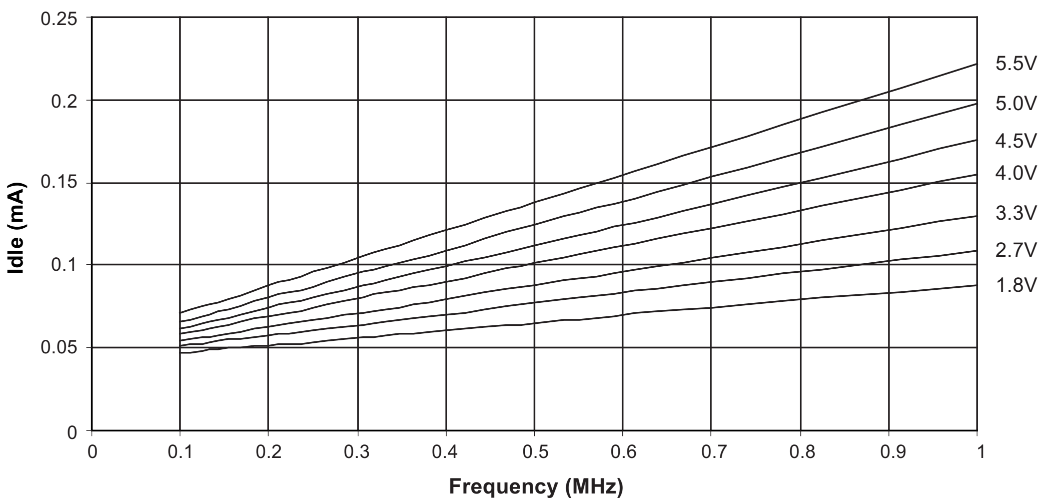
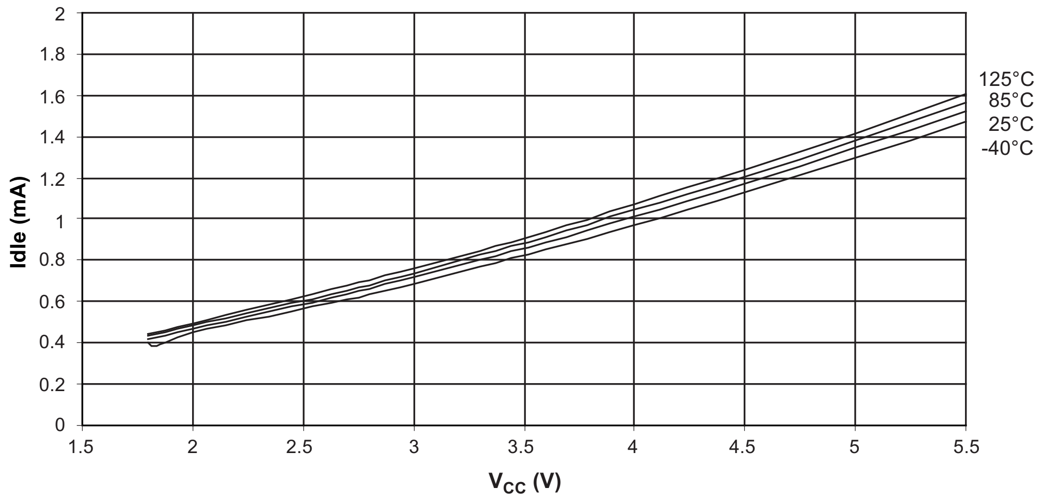


Figure 22-10. Idle Supply Current versus V_{CC} (Internal RC Oscillator, 8MHz)



22.2.1 Using the Power Reduction Register

The tables and formulas below can be used to calculate the additional current consumption for the different I/O modules in active and idle mode. The enabling or disabling of the I/O modules are controlled by the power reduction register. See Section 6.6 "Power Reduction Register" on page 30 for details.

Table 22-1. Additional Current Consumption for the different I/O modules (absolute values)

PRR bit	Typical numbers		
	V_{CC} = 2V, F = 1MHz	V_{CC} = 3V, F = 4MHz	V_{CC} = 5V, F = 8MHz
PRTIM1	43µA	270µA	1090µA
PRTIM0	5.0µA	28µA	116µA
PRUSI	4.0µA	25µA	102µA
PRADC	13µA	84µA	351µA

Table 22-2. Additional Current Consumption (percentage) in Active and Idle mode

PRR bit	Additional Current consumption compared to Active with external clock (see Figure 22-1 and Figure 22-2)	Additional Current consumption compared to Idle with external clock (see Figure 22-6 and Figure 22-7)
PRTIM1	17.3%	68.4%
PRTIM0	1.8%	7.3%
PRUSI	1.6%	6.4%
PRADC	5.4%	21.4%

It is possible to calculate the typical current consumption based on the numbers from Table 22-2 for other V_{CC} and frequency settings than listed in Table 22-1.

22.8 Internal Oscillator Speed

Figure 22-34. Watchdog Oscillator Frequency versus V_{CC})

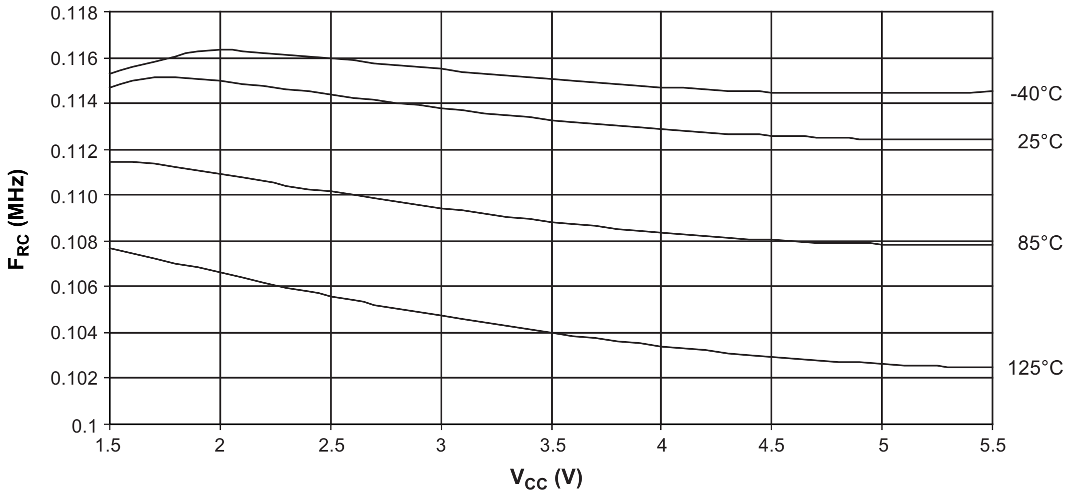


Figure 22-35. Watchdog Oscillator Frequency versus Temperature