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	Obsolete
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	23
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-TQFP
Supplier Device Package	32-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/atmel/atmega168-15at

- **Bit 5 - PRTIM0: Power Reduction Timer/Counter0**

Writing a logic one to this bit shuts down the Timer/Counter0 module. When the Timer/Counter0 is enabled, operation will continue like before the shutdown.

- **Bit 4 - Res: Reserved bit**

This bit is reserved in Atmel® ATmega48/88/168 and will always read as zero.

- **Bit 3 - PRTIM1: Power Reduction Timer/Counter1**

Writing a logic one to this bit shuts down the Timer/Counter1 module. When the Timer/Counter1 is enabled, operation will continue like before the shutdown.

- **Bit 2 - PRSPI: Power Reduction Serial Peripheral Interface**

If using debugWIRE on-chip debug system, this bit should not be written to one. Writing a logic one to this bit shuts down the serial peripheral interface by stopping the clock to the module. When waking up the SPI again, the SPI should be re-initialized to ensure proper operation.

- **Bit 1 - PRUSART0: Power Reduction USART0**

Writing a logic one to this bit shuts down the USART by stopping the clock to the module. When waking up the USART again, the USART should be re-initialized to ensure proper operation.

- **Bit 0 - PRADC: Power Reduction ADC**

Writing a logic one to this bit shuts down the ADC. The ADC must be disabled before shut down. The analog comparator cannot use the ADC input MUX when the ADC is shut down.

7.8 Minimizing Power Consumption

There are several possibilities to consider when trying to minimize the power consumption in an AVR® controlled system. In general, sleep modes should be used as much as possible, and the sleep mode should be selected so that as few as possible of the device's functions are operating. All functions not needed should be disabled. In particular, the following modules may need special consideration when trying to achieve the lowest possible power consumption.

7.8.1 Analog to Digital Converter

If enabled, the ADC will be enabled in all sleep modes. To save power, the ADC should be disabled before entering any sleep mode. When the ADC is turned off and on again, the next conversion will be an extended conversion. Refer to Section 21. “Analog-to-Digital Converter” on page 206 for details on ADC operation.

7.8.2 Analog Comparator

When entering idle mode, the analog comparator should be disabled if not used. When entering ADC noise reduction mode, the analog comparator should be disabled. In other sleep modes, the analog comparator is automatically disabled. However, if the analog comparator is set up to use the internal voltage reference as input, the analog comparator should be disabled in all sleep modes. Otherwise, the internal voltage reference will be enabled, independent of sleep mode. Refer to Section 20. “Analog Comparator” on page 203 for details on how to configure the analog comparator.

7.8.3 Brown-out Detector

If the brown-out detector is not needed by the application, this module should be turned off. If the brown-out detector is enabled by the BODLEVEL fuses, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to Section 8.5 “Brown-out Detection” on page 41 for details on how to configure the brown-out detector.

10.2.2 Toggling the Pin

Writing a logic one to PINxn toggles the value of PORTxn, independent on the value of DDRxn. Note that the SBI instruction can be used to toggle one single bit in a port.

10.2.3 Switching Between Input and Output

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled {DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedant environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the MCUCR register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b11) as an intermediate step.

Table 10-1 summarizes the control signals for the pin value.

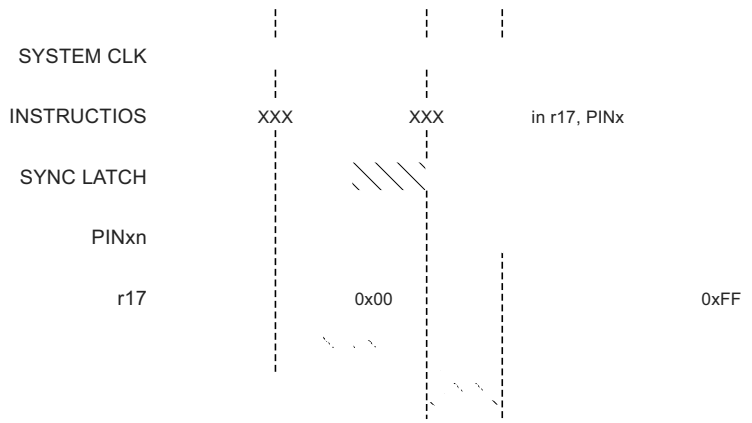
Table 10-1. Port Pin Configurations

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (hi-Z)
1	0	X	Output	No	Output low (sink)
1	1	X	Output	No	Output high (source)

10.2.4 Reading the Pin Value

Independent of the setting of data direction bit DDxn, the port pin can be read through the PINxn register bit. As shown in Figure 10-2 on page 58, the PINxn register bit and the preceding latch constitute a synchronizer. This is needed to avoid metastability if the physical pin changes value near the edge of the internal clock, but it also introduces a delay. Figure 10-3 shows a timing diagram of the synchronization when reading an externally applied pin value. The maximum and minimum propagation delays are denoted $t_{pd,max}$ and $t_{pd,min}$ respectively.

Figure 10-3. Synchronization when Reading an Externally Applied Pin value



Consider the clock period starting shortly after the first falling edge of the system clock. The latch is closed when the clock is low, and goes transparent when the clock is high, as indicated by the shaded region of the “SYNC LATCH” signal. The signal value is latched when the system clock goes low. It is clocked into the PINxn register at the succeeding positive clock edge. As indicated by the two arrows tpd,max and tpd,min, a single signal transition on the pin will be delayed between ½ and 1½ system clock period depending upon the time of assertion.

- **Bit 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 Bit 1 and Bit 0**

The external interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 11-2. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 11-2. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

11.2 External Interrupt Mask Register – EIMSK

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	–	–	INT1	INT0	EIMSK
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..2 – Res: Reserved Bits**

These bits are unused bits in the Atmel® ATmega48/88/168, and will always read as zero.

- **Bit 1 – INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the status register (SREG) is set (one), the external pin interrupt is enabled. The interrupt sense control1 bits 1/0 (ISC11 and ISC10) in the external interrupt control register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT1 pin or level sensed. Activity on the pin will cause an interrupt request even if INT1 is configured as an output. The corresponding interrupt of external interrupt request 1 is executed from the INT1 interrupt vector.

- **Bit 0 – INT0: External Interrupt Request 0 Enable**

When the INT0 bit is set (one) and the I-bit in the status register (SREG) is set (one), the external pin interrupt is enabled. The interrupt sense control0 bits 1/0 (ISC01 and ISC00) in the external interrupt control register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of external interrupt request 0 is executed from the INT0 interrupt vector.

11.3 External Interrupt Flag Register – EIFR

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	–	–	INTF1	INTF0	EIFR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..2 – Res: Reserved Bits**

These bits are unused bits in the Atmel ATmega48/88/168, and will always read as zero.

14. 16-bit Timer/Counter1 with PWM

The 16-bit Timer/Counter unit allows accurate program execution timing (event management), wave generation, and signal timing measurement. The main features are:

- True 16-bit Design (i.e., allows 16-bit PWM)
- Two independent output compare units
- Double buffered output compare registers
- One input capture unit
- Input capture noise canceler
- Clear timer on compare match (auto reload)
- Glitch-free, phase correct pulse width modulator (PWM)
- Variable PWM period
- Frequency generator
- External event counter
- Four independent interrupt sources (TOV1, OCF1A, OCF1B, and ICF1)

14.1 Overview

Most register and bit references in this section are written in general form. A lower case “n” replaces the Timer/Counter number, and a lower case “x” replaces the output compare unit channel. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT1 for accessing Timer/Counter1 counter value and so on.

A simplified block diagram of the 16-bit Timer/Counter is shown in Figure 14-1 on page 95. For the actual placement of I/O pins, refer to Section 1-1 “Pinout ATmega48/88/168” on page 3. 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 the Section 14.10 “16-bit Timer/Counter Register Description” on page 113.

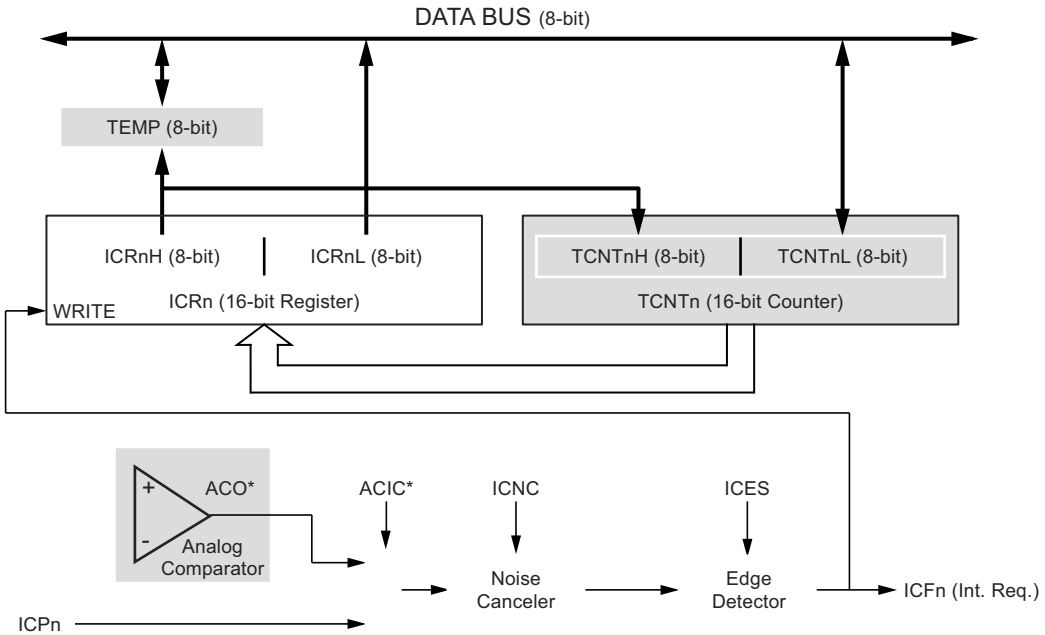
The PRTIM1 bit in Section 7.7.1 “Power Reduction Register - PRR” on page 35 must be written to zero to enable Timer/Counter1 module.

14.5 Input Capture Unit

The Timer/Counter incorporates an input capture unit that can capture external events and give them a time-stamp indicating time of occurrence. The external signal indicating an event, or multiple events, can be applied via the ICP1 pin or alternatively, via the analog-comparator unit. The time-stamps can then be used to calculate frequency, duty-cycle, and other features of the signal applied. Alternatively the time-stamps can be used for creating a log of the events.

The input capture unit is illustrated by the block diagram shown in Figure 14-3. The elements of the block diagram that are not directly a part of the input capture unit are gray shaded. The small “n” in register and bit names indicates the Timer/Counter number.

Figure 14-3. Input Capture Unit Block Diagram



When a change of the logic level (an event) occurs on the input capture pin (ICP1), alternatively on the analog comparator output (ACO), and this change confirms to the setting of the edge detector, a capture will be triggered. When a capture is triggered, the 16-bit value of the counter (TCNT1) is written to the input capture register (ICR1). The input capture flag (ICF1) is set at the same system clock as the TCNT1 value is copied into ICR1 Register. If enabled (ICIE1 = 1), the Input Capture flag generates an input capture interrupt. The ICF1 flag is automatically cleared when the interrupt is executed. Alternatively the ICF1 flag can be cleared by software by writing a logical one to its I/O bit location.

Reading the 16-bit value in the input capture register (ICR1) is done by first reading the low byte (ICR1L) and then the high byte (ICR1H). When the low byte is read the high byte is copied into the high byte temporary register (TEMP). When the CPU reads the ICR1H I/O location it will access the TEMP register.

The ICR1 register can only be written when using a waveform generation mode that utilizes the ICR1 register for defining the counter’s TOP value. In these cases the waveform generation mode (WGM13:0) bits must be set before the TOP value can be written to the ICR1 register. When writing the ICR1 register the high byte must be written to the ICR1H I/O location before the low byte is written to ICR1L.

For more information on how to access the 16-bit registers refer to Section 14.2 “Accessing 16-bit Registers” on page 96.

The OCR2x register is 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 OCR2x compare register 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 OCR2x register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR2x buffer register, and if double buffering is disabled the CPU will access the OCR2x directly.

15.4.1 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 (FOC2x) bit. Forcing compare match will not set the OCF2x flag or reload/clear the timer, but the OC2x pin will be updated as if a real compare match had occurred (the COM2x1:0 bits settings define whether the OC2x pin is set, cleared or toggled).

15.4.2 Compare Match Blocking by TCNT2 Write

All CPU write operations to the TCNT2 register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR2x to be initialized to the same value as TCNT2 without triggering an interrupt when the Timer/Counter clock is enabled.

15.4.3 Using the Output Compare Unit

Since writing TCNT2 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT2 when using the output compare channel, independently of whether the Timer/Counter is running or not. If the value written to TCNT2 equals the OCR2x value, the compare match will be missed, resulting in incorrect waveform generation. Similarly, do not write the TCNT2 value equal to BOTTOM when the counter is downcounting.

The setup of the OC2x should be performed before setting the data direction register for the port pin to output. The easiest way of setting the OC2x value is to use the force output compare (FOC2x) strobe bit in normal mode. The OC2x register keeps its value even when changing between waveform generation modes.

Be aware that the COM2x1:0 bits are not double buffered together with the compare value. Changing the COM2x1:0 bits will take effect immediately.

15.5 Compare Match Output Unit

The compare output mode (COM2x1:0) bits have two functions. The waveform generator uses the COM2x1:0 bits for defining the output compare (OC2x) state at the next compare match. Also, the COM2x1:0 bits control the OC2x pin output source. Figure 15-4 on page 123 shows a simplified schematic of the logic affected by the COM2x1:0 bit setting.
The I/O registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O port control registers (DDR and PORT) that are affected by the COM2x1:0 bits are shown. When referring to the OC2x state, the reference is for the internal OC2x register, not the OC2x pin.

15.8.6 Timer/Counter2 Interrupt Mask Register – TIMSK2

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	–	OCIE2B	OCIE2A	TOIE2	TIMSK2
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 2 – OCIE2B: Timer/Counter2 Output Compare Match B Interrupt Enable**

When the OCIE2B bit is written to one and the I-bit in the status register is set (one), the Timer/Counter2 compare match B interrupt is enabled. The corresponding interrupt is executed if a compare match in Timer/Counter2 occurs, i.e., when the OCF2B bit is set in the Timer/Counter 2 interrupt flag register – TIFR2.

- **Bit 1 – OCIE2A: Timer/Counter2 Output Compare Match A Interrupt Enable**

When the OCIE2A bit is written to one and the I-bit in the status register is set (one), the Timer/Counter2 compare match A interrupt is enabled. The corresponding interrupt is executed if a compare match in Timer/Counter2 occurs, i.e., when the OCF2A bit is set in the Timer/Counter 2 interrupt flag register – TIFR2.

- **Bit 0 – TOIE2: Timer/Counter2 Overflow Interrupt Enable**

When the TOIE2 bit is written to one and the I-bit in the status register is set (one), the Timer/Counter2 overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter2 occurs, i.e., when the TOV2 bit is set in the Timer/Counter2 interrupt flag register – TIFR2.

15.8.7 Timer/Counter2 Interrupt Flag Register – TIFR2

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	–	OCF2B	OCF2A	TOV2	TIFR2
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 2 – OCF2B: Output Compare Flag 2 B**

The OCF2B bit is set (one) when a compare match occurs between the Timer/Counter2 and the data in OCR2B – output compare register2. OCF2B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF2B is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE2B (Timer/Counter2 compare match interrupt enable), and OCF2B are set (one), the Timer/Counter2 compare match interrupt is executed.

- **Bit 1 – OCF2A: Output Compare Flag 2 A**

The OCF2A bit is set (one) when a compare match occurs between the Timer/Counter2 and the data in OCR2A – output compare register2. OCF2A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF2A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE2A (Timer/Counter2 compare match interrupt enable), and OCF2A are set (one), the Timer/Counter2 compare match interrupt is executed.

- **Bit 0 – TOV2: Timer/Counter2 Overflow Flag**

The TOV2 bit is set (one) when an overflow occurs in Timer/Counter2. TOV2 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV2 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE2A (Timer/Counter2 overflow interrupt enable), and TOV2 are set (one), the Timer/Counter2 overflow interrupt is executed. In PWM mode, this bit is set when Timer/Counter2 changes counting direction at 0x00.

- **Bit 5..1 – Res: Reserved Bits**

These bits are reserved bits in the Atmel® ATmega48/88/168 and will always read as zero.

- **Bit 0 – SPI2X: Double SPI Speed Bit**

When this bit is written logic one the SPI speed (SCK frequency) will be doubled when the SPI is in master mode (see Table 16-4 on page 143). This means that the minimum SCK period will be two CPU clock periods. When the SPI is configured as Slave, the SPI is only guaranteed to work at $f_{osc}/4$ or lower.

The SPI interface on the Atmel ATmega48/88/168 is also used for program memory and EEPROM downloading or uploading. See Section 25.8 “Serial Downloading” on page 256 for serial programming and verification.

16.1.5 SPI Data Register – SPDR

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	SPDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	X	X	X	X	X	X	X	X	Undefined

The SPI data register is a read/write register used for data transfer between the register file and the SPI shift register. Writing to the register initiates data transmission. Reading the register causes the shift register receive buffer to be read.

16.2 Data Modes

There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 16-3 on page 145 and Figure 16-4 on page 145. Data bits are shifted out and latched in on opposite edges of the SCK signal, ensuring sufficient time for data signals to stabilize. This is clearly seen by summarizing Figure 16-2 on page 142 and Table 16-3 on page 143, as done below.

Table 16-5. CPOL Functionality

	Leading Edge	Trailing eDge	SPI Mode
CPOL=0, CPHA=0	Sample (rising)	Setup (falling)	0
CPOL=0, CPHA=1	Setup (rising)	Sample (falling)	1
CPOL=1, CPHA=0	Sample (falling)	Setup (rising)	2
CPOL=1, CPHA=1	Setup (falling)	Sample (rising)	3

17. USART0

The universal synchronous and asynchronous serial receiver and transmitter (USART) is a highly flexible serial communication device. The main features are:

- Full duplex operation (independent serial receive and transmit registers)
- Asynchronous or synchronous operation
- Master or slave clocked synchronous operation
- High resolution baud rate generator
- Supports serial frames with 5, 6, 7, 8, or 9 data bits and 1 or 2 stop bits
- Odd or even parity generation and parity check supported by hardware
- Data overrun detection
- Framing error detection
- Noise filtering includes false start bit detection and digital low pass filter
- Three separate interrupts on TX complete, TX data register empty and RX complete
- Multi-processor communication mode
- Double speed asynchronous communication mode

The USART can also be used in master SPI mode, see Section 18. “USART in SPI Mode” on page 168. The power reduction USART bit, PRUSART0, in Section 7.7.1 “Power Reduction Register - PRR” on page 35 must be disabled by writing a logical zero to it.

By writing the TWEA bit to zero, the device can be virtually disconnected from the 2-wire serial bus temporarily. Address recognition can then be resumed by writing the TWEA bit to one again.

- **Bit 5 – TWSTA: TWI START Condition Bit**

The application writes the TWSTA bit to one when it desires to become a master on the 2-wire serial bus. The TWI hardware checks if the bus is available, and generates a START condition on the bus if it is free. However, if the bus is not free, the TWI waits until a STOP condition is detected, and then generates a new START condition to claim the bus master status. TWSTA must be cleared by software when the START condition has been transmitted.

- **Bit 4 – TWSTO: TWI STOP Condition Bit**

Writing the TWSTO bit to one in master mode will generate a STOP condition on the 2-wire serial bus. When the STOP condition is executed on the bus, the TWSTO bit is cleared automatically. In slave mode, setting the TWSTO bit can be used to recover from an error condition. This will not generate a STOP condition, but the TWI returns to a well-defined unaddressed Slave mode and releases the SCL and SDA lines to a high impedance state.

- **Bit 3 – TWWC: TWI Write Collision Flag**

The TWWC bit is set when attempting to write to the TWI data register – TWDR when TWINT is low. This flag is cleared by writing the TWDR register when TWINT is high.

- **Bit 2 – TWEN: TWI Enable Bit**

The TWEN bit enables TWI operation and activates the TWI interface. When TWEN is written to one, the TWI takes control over the I/O pins connected to the SCL and SDA pins, enabling the slew-rate limiters and spike filters. If this bit is written to zero, the TWI is switched off and all TWI transmissions are terminated, regardless of any ongoing operation.

- **Bit 1 – Res: Reserved Bit**

This bit is a reserved bit and will always read as zero.

- **Bit 0 – TWIE: TWI Interrupt Enable**

When this bit is written to one, and the I-bit in SREG is set, the TWI interrupt request will be activated for as long as the TWINT flag is high.

19.6.3 TWI Status Register – TWSR

Bit	7	6	5	4	3	2	1	0	
	TWS7	**TWS6**	**TWS5**	**TWS4**	**TWS3**	**–**	**TWPS1**	**TWPS0**	**TWSR**
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	1	1	1	1	1	0	0	0	

- **Bits 7..3 – TWS: TWI Status**

These 5 bits reflect the status of the TWI logic and the 2-wire serial bus. The different status codes are described later in this section. Note that the value read from TWSR contains both the 5-bit status value and the 2-bit prescaler value. The application designer should mask the prescaler bits to zero when checking the status bits. This makes status checking independent of prescaler setting. This approach is used in this datasheet, unless otherwise noted.

- **Bit 2 – Res: Reserved Bit**

This bit is reserved and will always read as zero.

- **Bits 1..0 – TWPS: TWI Prescaler Bits**

These bits can be read and written, and control the bit rate prescaler.

21.6.4 ADC Control and Status Register B – ADCSRB

Bit	7	6	5	4	3	2	1	0	
	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7, 5:3 – Res: Reserved Bits**

These bits are reserved for future use. To ensure compatibility with future devices, these bist must be written to zero when ADCSRB is written.

- **Bit 2:0 – ADTS2:0: ADC Auto Trigger Source**

If ADATE in ADCSRA is written to one, the value of these bits selects which source will trigger an ADC conversion. If ADATE is cleared, the ADTS2:0 settings will have no effect. A conversion will be triggered by the rising edge of the selected interrupt flag. Note that switching from a trigger source that is cleared to a trigger source that is set, will generate a positive edge on the trigger signal. If ADEN in ADCSRA is set, this will start a conversion. Switching to free running mode (ADTS[2:0]=0) will not cause a trigger event, even if the ADC interrupt flag is set**.**

Table 21-5. ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free running mode
0	0	1	Analog comparator
0	1	0	External interrupt request 0
0	1	1	Timer/Counter0 compare match A
1	0	0	Timer/Counter0 overflow
1	0	1	Timer/Counter1 compare match B
1	1	0	Timer/Counter1 overflow
1	1	1	Timer/Counter1 capture event

21.6.5 Digital Input Disable Register 0 – DIDR0

Bit	7	6	5	4	3	2	1	0	
	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – Res: Reserved Bits**

These bits are reserved for future use. To ensure compatibility with future devices, these bits must be written to zero when DIDR0 is written.

- **Bit 5..0 – ADC5D..ADC0D: ADC5..0 Digital Input Disable**

When this bit is written logic one, the digital input buffer on the corresponding ADC pin is disabled. The corresponding PIN register bit will always read as zero when this bit is set. When an analog signal is applied to the ADC5..0 pin and the digital input from this pin is not needed, this bit should be written logic one to reduce power consumption in the digital input buffer.

Note that ADC pins ADC7 and ADC6 do not have digital input buffers, and therefore do not require digital input disable bits.

22.4 Software Break Points

debugWIRE supports program memory break points by the AVR® break instruction. Setting a break point in AVR Studio® will insert a BREAK instruction in the program memory. The instruction replaced by the BREAK instruction will be stored. When program execution is continued, the stored instruction will be executed before continuing from the program memory. A break can be inserted manually by putting the BREAK instruction in the program.

The flash must be re-programmed each time a break point is changed. This is automatically handled by AVR Studio through the debugWIRE interface. The use of break points will therefore reduce the flash data retention. Devices used for debugging purposes should not be shipped to end customers.

22.5 Limitations of debugWIRE

The debugWIRE communication pin (dW) is physically located on the same pin as external reset (RESET). An external reset source is therefore not supported when the debugWIRE is enabled.

The debugWIRE system accurately emulates all I/O functions when running at full speed, i.e., when the program in the CPU is running. When the CPU is stopped, care must be taken while accessing some of the I/O registers via the debugger (AVR Studio).

A programmed DWEN fuse enables some parts of the clock system to be running in all sleep modes. This will increase the power consumption while in sleep. Thus, the DWEN fuse should be disabled when debugWire is not used.

22.6 debugWIRE Related Register in I/O Memory

The following section describes the registers used with the debugWire.

22.6.1 debugWire Data Register – DWDR

Bit	7	6	5	4	3	2	1	0	
	DWDR[7:0]								DWDR
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	

The DWDR register provides a communication channel from the running program in the MCU to the debugger. This register is only accessible by the debugWIRE and can therefore not be used as a general purpose register in the normal operations.

Flash corruption can easily be avoided by following these design recommendations (one is sufficient):

1. If there is no need for a boot loader update in the system, program the boot loader lock bits to prevent any boot loader software updates.
2. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal brown-out detector (BOD) if the operating voltage matches the detection level. If not, an external low V_{CC} reset protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.
3. Keep the AVR® core in power-down sleep mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the SPMCSR register and thus the flash from unintentional writes.

24.7.11 Programming Time for Flash when Using SPM

The calibrated RC oscillator is used to time flash accesses. Table 24-5 shows the typical programming time for flash accesses from the CPU.

Table 24-5. SPM Programming Time

Symbol	Min Programming Time	Max Programming Time
Flash write (page erase, page write, and write lock bits by SPM)	3.7ms	4.5ms

24.7.12 Simple Assembly Code Example for a Boot Loader

```
        ;-the routine writes one page of data from RAM to Flash
        ; the first data location in RAM is pointed to by the Y pointer
        ; the first data location in Flash is pointed to by the Z-pointer
        ;-error handling is not included
        ;-the routine must be placed inside the Boot space
        ; (at least the Do_spm sub routine). Only code inside NRWW section can
        ; be read during Self-Programming (Page Erase and Page Write).
        ;-registers used: r0, r1, temp1 (r16), temp2 (r17), looplo (r24),
        ; loophi (r25), spmcrval (r20)
        ; storing and restoring of registers is not included in the routine
        ; register usage can be optimized at the expense of code size
        ;-It is assumed that either the interrupt table is moved to the Boot
        ; loader section or that the interrupts are disabled.
.equ   PAGESIZEB = PAGESIZE*2     ;PAGESIZEB is page size in BYTES, not words
.org SMALLBOOTSTART
Write_page:
        ;Page Erase
        ldi    spmcrval, (1<<PGERS) | (1<<SELFPRGEN)
        call   Do_spm

        ;      re-enable the RWW section
        ldi    spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
        call   Do_spm

        ;      transfer data from RAM to Flash page buffer
        ldi    looplo, low(PAGESIZEB)     ;init loop variable
        ldi    loophi, high(PAGESIZEB)    ;not required for PAGESIZEB<=256
Wrloop:
        ld     r0, Y+
        ld     r1, Y+
        ldi    spmcrval, (1<<SELFPRGEN)
        call   Do_spm
        adiw   ZH:ZL, 2
        sbiw   loophi:looplo, 2    ;use subi for PAGESIZEB<=256
```

25.2.1 Latching of Fuses

The fuse values are latched when the device enters programming mode and changes of the fuse values will have no effect until the part leaves programming mode. This does not apply to the EESAVE fuse which will take effect once it is programmed. The fuses are also latched on power-up in normal mode.

25.3 Signature Bytes

All Atmel® microcontrollers have a three-byte signature code which identifies the device. This code can be read in both serial and parallel mode, also when the device is locked. The three bytes reside in a separate address space.

25.3.1 ATmega48 Signature Bytes

1. 0x000: 0x1E (indicates manufactured by Atmel).
2. 0x001: 0x92 (indicates 4KB flash memory).
3. 0x002: 0x05 (indicates Atmel ATmega48 device when 0x001 is 0x92).

25.3.2 ATmega88 Signature Bytes

1. 0x000: 0x1E (indicates manufactured by Atmel).
2. 0x001: 0x93 (indicates 8KB flash memory).
3. 0x002: 0x0A (indicates ATmega88 device when 0x001 is 0x93).

25.3.3 ATmega168 Signature Bytes

1. 0x000: 0x1E (indicates manufactured by Atmel).
2. 0x001: 0x94 (indicates 16KB flash memory).
3. 0x002: 0x06 (indicates Atmel ATmega168 device when 0x001 is 0x94).

25.4 Calibration Byte

The Atmel ATmega48/88/168 has a byte calibration value for the internal RC oscillator. This byte resides in the high byte of address 0x000 in the signature address space. During reset, this byte is automatically written into the OSCCAL register to ensure correct frequency of the calibrated RC oscillator.

25.5 Parallel Programming Parameters, Pin Mapping, and Commands

This section describes how to parallel program and verify flash program memory, EEPROM data memory, memory lock bits, and fuse bits in the Atmel ATmega48/88/168. Pulses are assumed to be at least 250ns unless otherwise noted.

25.5.1 Signal Names

In this section, some pins of the ATmega48/88/168 are referenced by signal names describing their functionality during parallel programming, see Figure 25-1 on page 246 and Table 25-8 on page 246. Pins not described in the following table are referenced by pin names.

The XA1/XA0 pins determine the action executed when the XTAL1 pin is given a positive pulse. The bit coding is shown in Table 25-10 on page 246.

When pulsing $\overline{WR}$ or $\overline{OE}$, the command loaded determines the action executed. The different Commands are shown in Table 25-11 on page 247.

27. 2-wire Serial Interface Characteristics

Table 27-1 describes the requirements for devices connected to the 2-wire serial bus. The Atmel® ATmega48/88/168 2-wire serial interface meets or exceeds these requirements under the noted conditions.

Timing symbols refer to Figure 27-1.

Table 27-1. 2-wire Serial Bus Requirements

Parameter	Condition	Symbol	Min	Max	Unit
Input low-voltage		V_{IL}	–0.5	0.3 V_{CC}	V
Input high-voltage		V_{IH}	0.7 V_{CC}	V_{CC} + 0.5	V
Hysteresis of schmitt trigger inputs		V_{hys}(1)	0.05 V_{CC}(2)	–	V
Output low-voltage	3mA sink current	V_{OL}(1)	0	0.4	V
Rise time for both SDA and SCL		t_r(1)	20 + 0.1C_b(3)(2)	300	ns
Output fall time from V_{IHmin} to V_{ILmax}	10pF < C_b < 400pF(3)	t_{of}(1)	20 + 0.1C_b(3)(2)	250	ns
Spikes suppressed by input filter		t_{SP}(1)	0	50(2)	ns
Input current each I/O pin	0.1V_{CC} < V_i < 0.9V_{CC}	I_i	–10	10	µA
Capacitance for each I/O Pin		C_i(1)	–	10	pF
SCL clock frequency	f_{CK}(4) > max(16f_{SCL}, 250kHz)(5)	f_{SCL}	0	400	kHz
Value of pull-up resistor	f_{SCL} ≤ 100kHz	Rp	$\frac{V_{CC}-0,4V}{3mA}$	$\frac{1000ns}{C_b}$	Ω
	f_{SCL} > 100kHz		$\frac{V_{CC}-0,4V}{3mA}$	$\frac{300ns}{C_b}$	Ω
Hold time (repeated) START Condition	f_{SCL} ≤ 100kHz	$t_{HD;STA}$	4.0	–	µs
	f_{SCL} > 100kHz		0.6	–	µs
Low period of the SCL clock	f_{SCL} ≤ 100kHz(6)	t_{LOW}	4.7	–	µs
	f_{SCL} > 100kHz(7)		1.3	–	µs
High period of the SCL clock	f_{SCL} ≤ 100kHz	t_{HIGH}	4.0	–	µs
	f_{SCL} > 100kHz		0.6	–	µs
Set-up time for a repeated START condition	f_{SCL} ≤ 100kHz	$t_{SU;STA}$	4.7	–	µs
	f_{SCL} > 100kHz		0.6	–	µs
Data hold time	f_{SCL} ≤ 100kHz	$t_{HD;DAT}$	0	3.45	µs
	f_{SCL} > 100kHz		0	0.9	µs

Notes:
1. In Atmel ATmega48/88/168, this parameter is characterized and not 100% tested.
2. Required only for f_{SCL} > 100kHz.
3. C_b = capacitance of one bus line in pF.
4. f_{CK} = CPU clock frequency
5. This requirement applies to all ATmega48/88/168 2-wire serial interface operation. Other devices connected to the 2-wire serial bus need only obey the general f_{SCL} requirement.
6. The actual low period generated by the Atmel ATmega48/88/168 2-wire serial interface is ($1/f_{SCL} – 2/f_{CK}$), thus f_{CK} must be greater than 6MHz for the low time requirement to be strictly met at f_{SCL} = 100kHz.
7. The actual low period generated by the ATmega48/88/168 2-wire serial interface is ($1/f_{SCL} – 2/f_{CK}$), thus the low time requirement will not be strictly met for f_{SCL} > 308kHz when f_{CK} = 8MHz. Still, ATmega48/88/168 devices connected to the bus may communicate at full speed (400kHz) with other ATmega48/88/168 devices, as well as any other device with a proper t_{LOW} acceptance margin.

Table 27-1. 2-wire Serial Bus Requirements (Continued)

Parameter	Condition	Symbol	Min	Max	Unit
Data setup time	$f_{SCL} \leq 100kHz$	$t_{SU;DAT}$	250	–	ns
	$f_{SCL} > 100kHz$		100	–	ns
Setup time for STOP condition	$f_{SCL} \leq 100kHz$	$t_{SU;STO}$	4.0	–	µs
	$f_{SCL} > 100kHz$		0.6	–	µs
Bus free time between a STOP and START condition	$f_{SCL} \leq 100kHz$	t_{BUF}	4.7	–	µs
	$f_{SCL} > 100kHz$		1.3	–	µs

Notes:
1. In Atmel ATmega48/88/168, this parameter is characterized and not 100% tested.
2. Required only for $f_{SCL} > 100kHz$.
3. C_b = capacitance of one bus line in pF.
4. f_{CK} = CPU clock frequency
5. This requirement applies to all ATmega48/88/168 2-wire serial interface operation. Other devices connected to the 2-wire serial bus need only obey the general f_{SCL} requirement.
6. The actual low period generated by the Atmel ATmega48/88/168 2-wire serial interface is $(1/f_{SCL} – 2/f_{CK})$, thus f_{CK} must be greater than 6MHz for the low time requirement to be strictly met at f_{SCL} = 100kHz.
7. The actual low period generated by the ATmega48/88/168 2-wire serial interface is $(1/f_{SCL} – 2/f_{CK})$, thus the low time requirement will not be strictly met for $f_{SCL} > 308kHz$ when f_{CK} = 8MHz. Still, ATmega48/88/168 devices connected to the bus may communicate at full speed (400kHz) with other ATmega48/88/168 devices, as well as any other device with a proper t_{LOW} acceptance margin.

Figure 27-1. 2-wire Serial Bus Timing

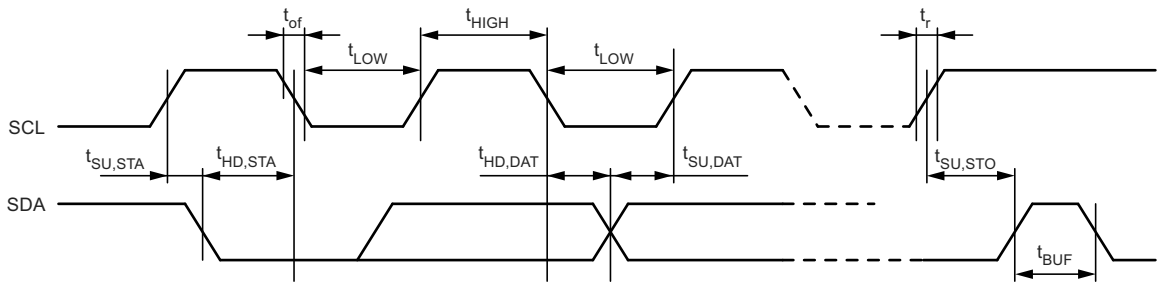


Figure 28-21. Calibrated 8MHz RC Oscillator Frequency versus V_{CC}

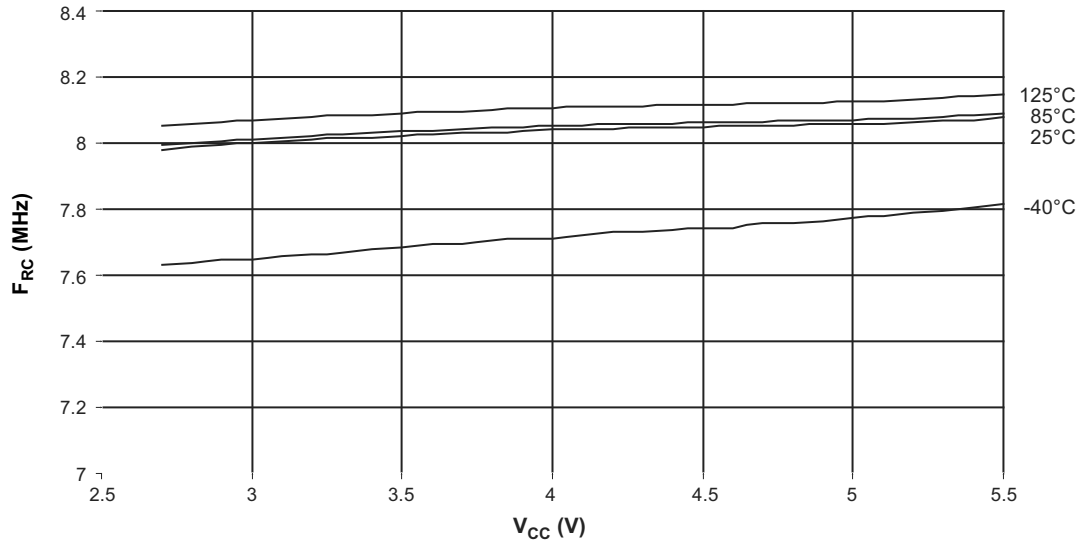


Figure 28-22. Calibrated 8MHz RC Oscillator Frequency versus OSCCAL Value (for ATmega48-15AZ and ATmega168-15AZ)

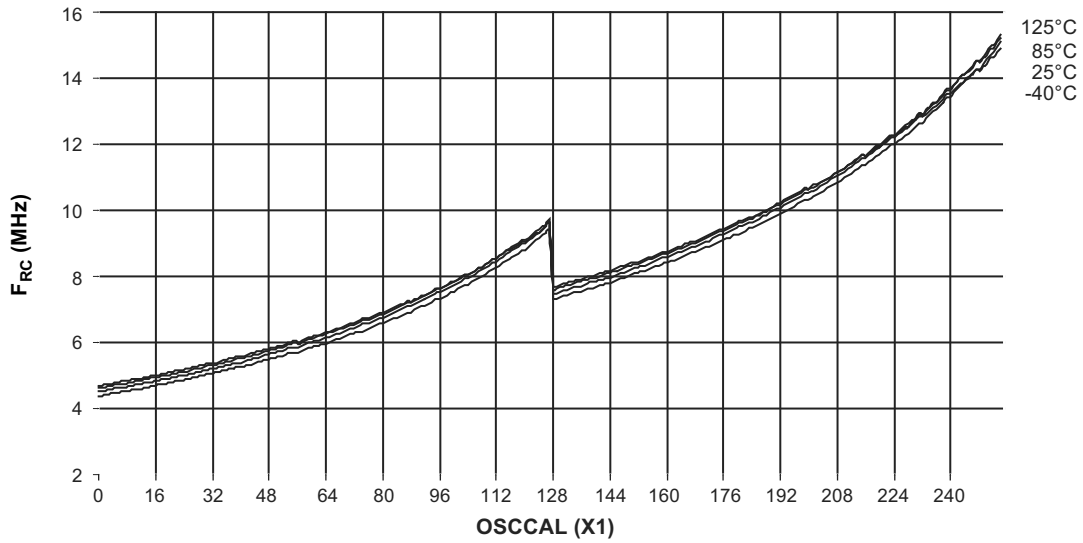
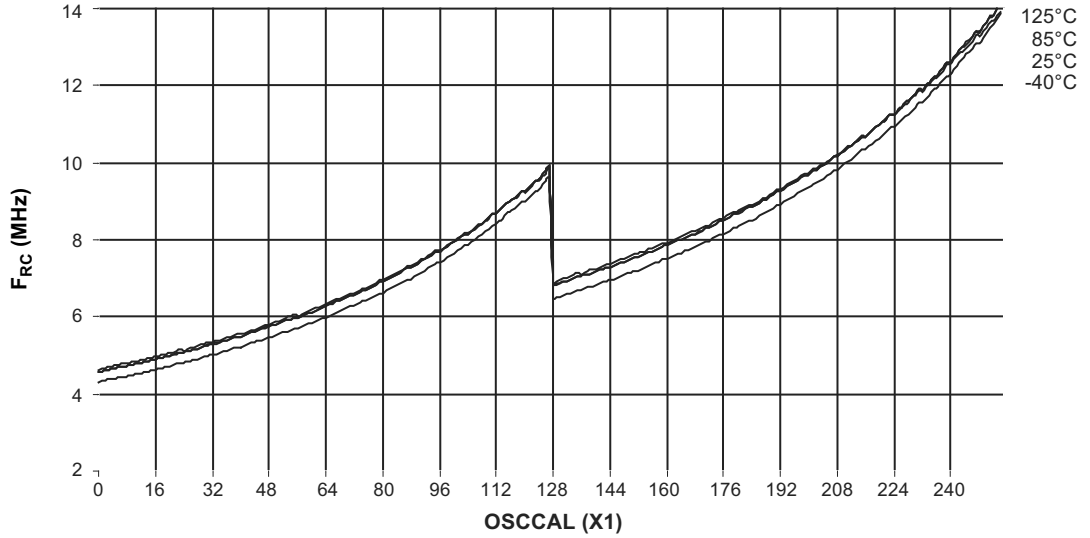
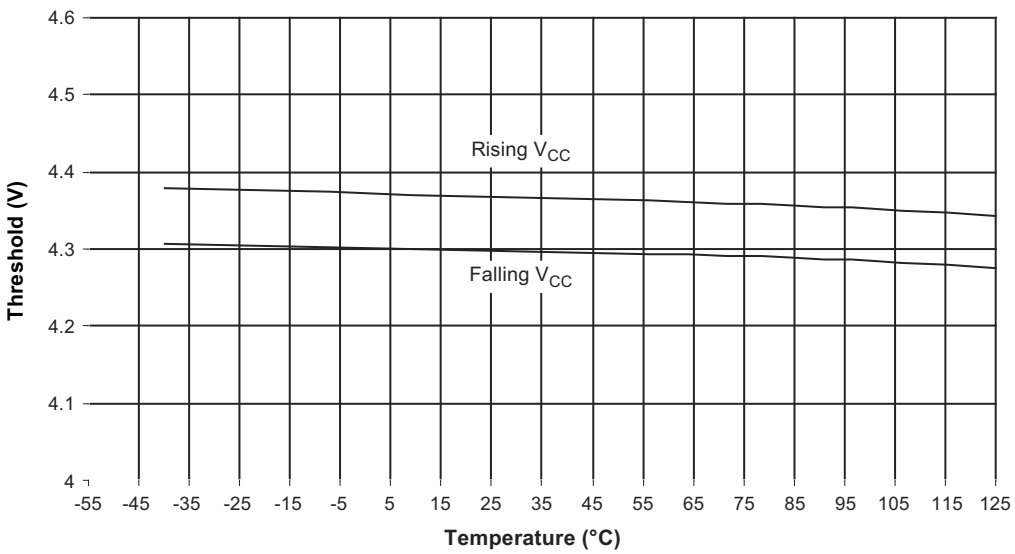


Figure 28-23. Calibrated 8MHz RC Oscillator Frequency versus OSCCAL Value (for ATmega88-15AZ only)



28.1.6 BOD Thresholds and Analog Comparator Offset

Figure 28-24. BOD Threshold versus Temperature (BODLEVEL is 4.0V)



28.1.7 Peripheral Units

Figure 28-27. Analog to Digital Converter GAIN versus V_{CC}

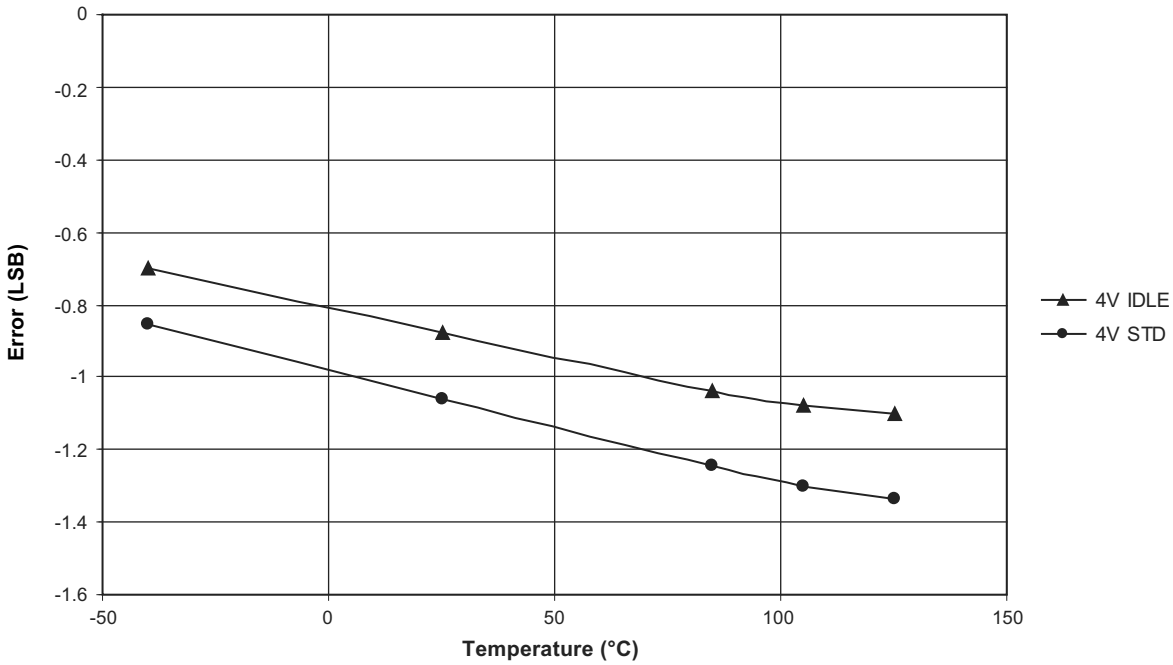
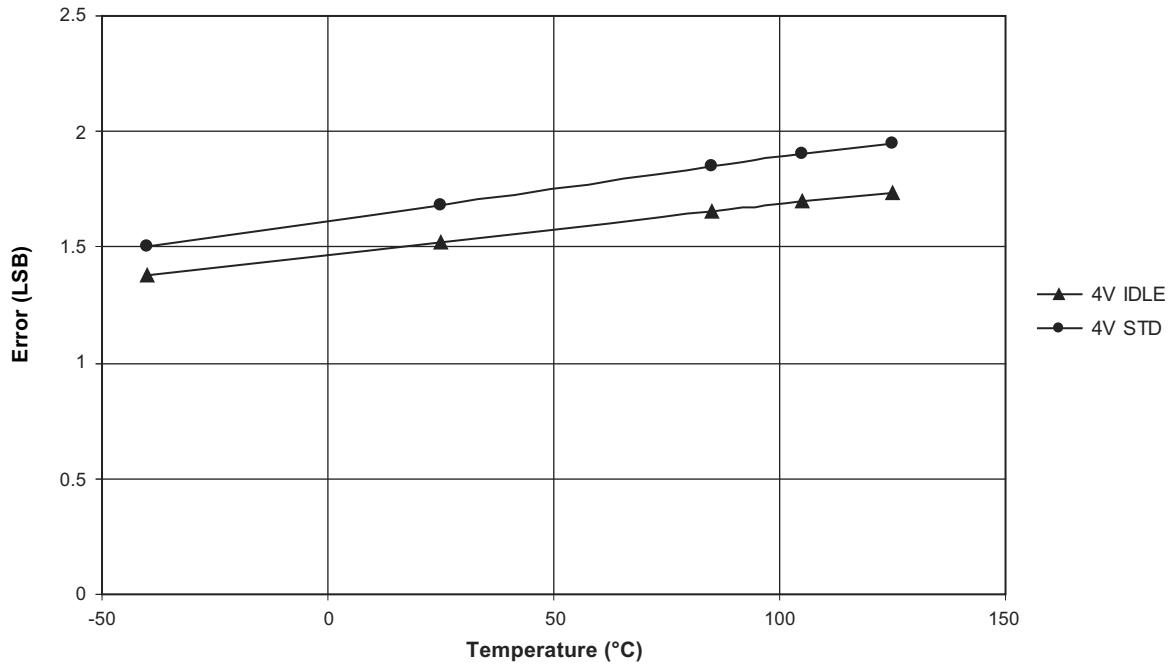


Figure 28-28. Analog to Digital Converter OFFSET versus V_{CC}



33. Errata ATmega48

The revision letter in this section refers to the revision of the Atmel® ATmega48 device.

33.1 Rev. E

- Interrupts may be lost when writing the timer registers in the asynchronous timer

1. Interrupts may be lost when writing the timer registers in the asynchronous timer

If one of the timer registers which is synchronized to the asynchronous Timer2 clock is written in the cycle before an overflow interrupt occurs, the interrupt may be lost.

Problem Fix/Workaround

Always check that the Timer2 Timer/Counter register, TCNT2, does not have the value 0xFF before writing the Timer2 control register, TCCR2, or output compare register, OCR2.

33.2 Rev. D

- Interrupts may be lost when writing the timer registers in the asynchronous timer
- POR sensitivity with Vcc ramp up from a very low supply voltage

1. Interrupts may be lost when writing the timer registers in the asynchronous timer

If one of the timer registers which is synchronized to the asynchronous Timer2 clock is written in the cycle before an overflow interrupt occurs, the interrupt may be lost.

Problem Fix/Workaround

Always check that the Timer2 Timer/Counter register, TCNT2, does not have the value 0xFF before writing the Timer2 control register, TCCR2, or output compare register, OCR2.

2. POR sensitivity with Vcc ramp up from a very low supply voltage

If Vcc ramp up from a stable 150mV to 300mV plateau, the power on reset (POR) may not reset the device properly.

Problem Fix/Workaround

None.

33.3 Rev. A

- Interrupts may be lost when writing the timer registers in the asynchronous timer
- POR sensitivity with Vcc ramp up from a very low supply voltage

1. Interrupts may be lost when writing the timer registers in the asynchronous timer

If one of the timer registers which is synchronized to the asynchronous Timer2 clock is written in the cycle before an overflow interrupt occurs, the interrupt may be lost.

Problem Fix/Workaround

Always check that the Timer2 Timer/Counter register, TCNT2, does not have the value 0xFF before writing the Timer2 control register, TCCR2, or output compare register, OCR2.

2. POR sensitivity with Vcc ramp up from a very low supply voltage

If Vcc ramp up from a stable 150mV to 300mV plateau, the power on reset (POR) may not reset the device properly.

Problem Fix/Workaround

None.

Note: Please note from datasheet 7530F-AVR-09/07 we introduce a new errata numbering scheme (errata Rev E of datasheet 7530E-AVR-03/07 is equivalent to errata rev D of datasheet 7530F-AVR-09/07)