



Welcome to [E-XFL.COM](https://www.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 | I ² C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 32 |
| 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) | 4.5V ~ 5.5V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-VQFN Exposed Pad |
| Supplier Device Package | 44-VQFN (7x7) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/atmega16-16mi |

Resources

A comprehensive set of development tools, application notes and datasheets are available for download on <http://www.atmel.com/avr>.

Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

General Purpose Register File

The Register File is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register File:

- One 8-bit output operand and one 8-bit result input
- Two 8-bit output operands and one 8-bit result input
- Two 8-bit output operands and one 16-bit result input
- One 16-bit output operand and one 16-bit result input

Figure 4 shows the structure of the 32 general purpose working registers in the CPU.

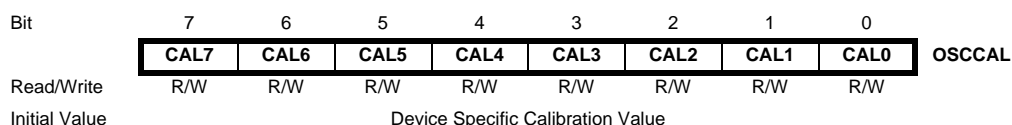
Figure 4. AVR CPU General Purpose Working Registers

| | | | | |
|--|-----|---|-------|----------------------|
| | 7 | 0 | Addr. | |
| | R0 | | \$00 | |
| | R1 | | \$01 | |
| | R2 | | \$02 | |
| | ... | | | |
| | R13 | | \$0D | |
| | R14 | | \$0E | |
| | R15 | | \$0F | |
| General Purpose Working Registers | R16 | | \$10 | |
| | R17 | | \$11 | |
| | ... | | | |
| | R26 | | \$1A | X-register Low Byte |
| | R27 | | \$1B | X-register High Byte |
| | R28 | | \$1C | Y-register Low Byte |
| | R29 | | \$1D | Y-register High Byte |
| | R30 | | \$1E | Z-register Low Byte |
| | R31 | | \$1F | 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 4, 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.

Oscillator Calibration Register – OSCCAL



• Bits 7..0 – CAL7..0: Oscillator Calibration Value

Writing the calibration byte to this address will trim the Internal Oscillator to remove process variations from the Oscillator frequency. This is done automatically during Chip Reset. When OSCCAL is zero, the lowest available frequency is chosen. Writing non-zero values to this register will increase the frequency of the Internal Oscillator. Writing \$FF to the register gives the highest available frequency. The calibrated Oscillator is used to time EEPROM and Flash access. If EEPROM or Flash is written, do not calibrate to more than 10% above the nominal frequency. Otherwise, the EEPROM or Flash write may fail. Note that the Oscillator is intended for calibration to 1.0 MHz, 2.0 MHz, 4.0 MHz, or 8.0 MHz. Tuning to other values is not guaranteed, as indicated in [Table 11](#).

Table 11. Internal RC Oscillator Frequency Range.

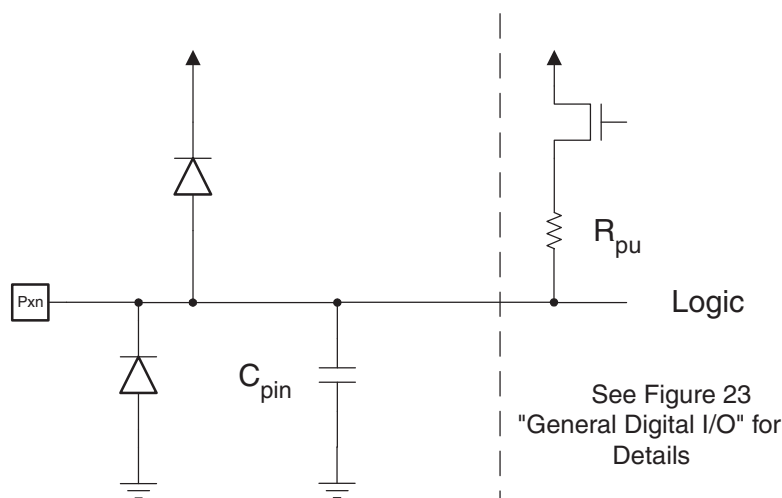
| OSCCAL Value | Min Frequency in Percentage of Nominal Frequency (%) | Max Frequency in Percentage of Nominal Frequency (%) |
|--------------|--|--|
| \$00 | 50 | 100 |
| \$7F | 75 | 150 |
| \$FF | 100 | 200 |

I/O Ports

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 22. Refer to “Electrical Characteristics” on page 291 for a complete list of parameters.

Figure 22. 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, that is, PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn. The physical I/O Registers and bit locations are listed in “Register Description for I/O Ports” on page 66.

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. In addition, the Pull-up Disable – PUD bit in SFIOR 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 “Ports as General Digital I/O” on page 50. 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 “Alternate Port Functions” on page 55. 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.

Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 23 shows a functional description of one I/O-port pin, here generically called Pxn.

The following code example shows how to set port B pins 0 and 1 high, 2 and 3 low, and define the port pins from 4 to 7 as input with pull-ups assigned to port pins 6 and 7. The resulting pin values are read back again, but as previously discussed, a *nop* instruction is included to be able to read back the value recently assigned to some of the pins.

Assembly Code Example⁽¹⁾

```
...
; Define pull-ups and set outputs high
; Define directions for port pins
ldi r16, (1<<PB7) | (1<<PB6) | (1<<PB1) | (1<<PB0)
ldi r17, (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0)
out PORTB,r16
out DDRB,r17
; Insert nop for synchronization
nop
; Read port pins
in r16,PINB
...
```

C Code Example⁽¹⁾

```
unsigned char i;
...
/* Define pull-ups and set outputs high */
/* Define directions for port pins */
PORTB = (1<<PB7) | (1<<PB6) | (1<<PB1) | (1<<PB0);
DDRB = (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0);
/* Insert nop for synchronization*/
_NOP();
/* Read port pins */
i = PINB;
...
```

Note: 1. For the assembly program, two temporary registers are used to minimize the time from pull-ups are set on pins 0, 1, 6, and 7, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

Digital Input Enable and Sleep Modes

As shown in [Figure 23](#), the digital input signal can be clamped to ground at the input of the schmitt-trigger. The signal denoted SLEEP in the figure, is set by the MCU Sleep Controller in Power-down mode, Power-save mode, Standby mode, and Extended Standby mode to avoid high power consumption if some input signals are left floating, or have an analog signal level close to $V_{CC}/2$.

SLEEP is overridden for port pins enabled as External Interrupt pins. If the External Interrupt Request is not enabled, SLEEP is active also for these pins. SLEEP is also overridden by various other alternate functions as described in [“Alternate Port Functions” on page 55](#).

If a logic high level (“one”) is present on an Asynchronous External Interrupt pin configured as “Interrupt on Rising Edge, Falling Edge, or Any Logic Change on Pin” while the External Interrupt is *not* enabled, the corresponding External Interrupt Flag will be set when resuming from the above mentioned sleep modes, as the clamping in these sleep modes produces the requested logic change.

Table 24. Overriding Signals for Alternate Functions in PA3..PA0

| Signal Name | PA3/ADC3 | PA2/ADC2 | PA1/ADC1 | PA0/ADC0 |
|-------------|------------|------------|------------|------------|
| PUOE | 0 | 0 | 0 | 0 |
| PUOV | 0 | 0 | 0 | 0 |
| DDOE | 0 | 0 | 0 | 0 |
| DDOV | 0 | 0 | 0 | 0 |
| PVOE | 0 | 0 | 0 | 0 |
| PVOV | 0 | 0 | 0 | 0 |
| DIEOE | 0 | 0 | 0 | 0 |
| DIEOV | 0 | 0 | 0 | 0 |
| DI | — | — | — | — |
| AIO | ADC3 INPUT | ADC2 INPUT | ADC1 INPUT | ADC0 INPUT |

Alternate Functions of Port B

The Port B pins with alternate functions are shown in [Table 25](#).

Table 25. Port B Pins Alternate Functions

| Port Pin | Alternate Functions |
|----------|---|
| PB7 | SCK (SPI Bus Serial Clock) |
| PB6 | MISO (SPI Bus Master Input/Slave Output) |
| PB5 | MOSI (SPI Bus Master Output/Slave Input) |
| PB4 | \overline{SS} (SPI Slave Select Input) |
| PB3 | AIN1 (Analog Comparator Negative Input) OC0 (Timer/Counter0 Output Compare Match Output) |
| PB2 | AIN0 (Analog Comparator Positive Input) INT2 (External Interrupt 2 Input) |
| PB1 | T1 (Timer/Counter1 External Counter Input) |
| PB0 | T0 (Timer/Counter0 External Counter Input) XCK (USART External Clock Input/Output) |

The alternate pin configuration is as follows:

• SCK – Port B, Bit 7

SCK: Master Clock output, Slave Clock input pin for SPI channel. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB7. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB7. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB7 bit.

• MISO – Port B, Bit 6

MISO: Master Data input, Slave Data output pin for SPI channel. When the SPI is enabled as a Master, this pin is configured as an input regardless of the setting of DDB6. When the SPI is enabled as a Slave, the data direction of this pin is controlled by DDB6. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB6 bit.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC0 pin. Setting the COM01:0 bits to 2 will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM01:0 to 3 (See [Table 40 on page 84](#)). The actual OC0 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 OC0 Register at the compare match between OCR0 and TCNT0, and clearing (or setting) the OC0 Register at the timer clock cycle the counter is cleared (changes from MAX to BOTTOM).

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

$$f_{OCnPWM} = \frac{f_{clk_I/O}}{N \cdot 256}$$

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

The extreme values for the OCR0 Register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR0 is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle. Setting the OCR0 equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM01:0 bits.)

PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnPCPWM} = \frac{f_{clk_I/O}}{N \cdot 510}$$

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

The extreme values for the OCR0 Register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR0 is set equal to BOTTOM, the output will be continuously low and if set equal to MAX the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

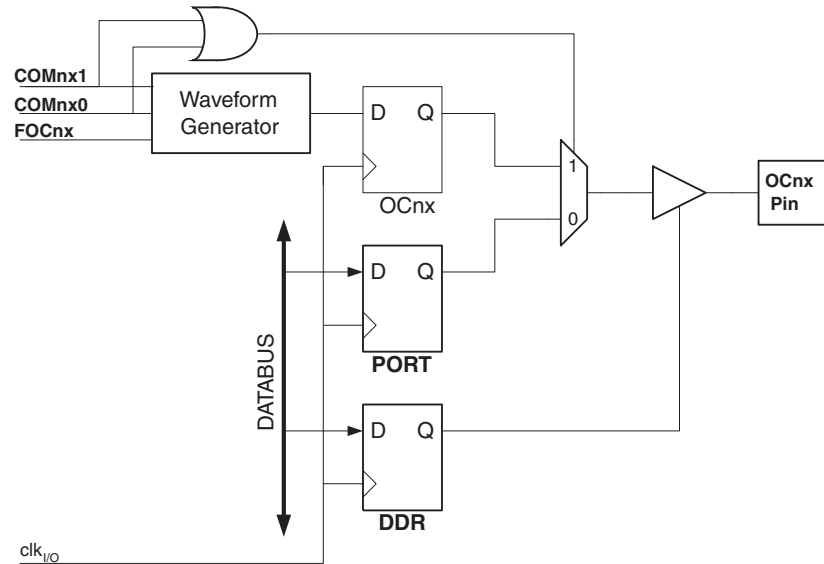
At the very start of Period 2 in [Figure 33](#) OCn has a transition from high to low even though there is no Compare Match. The point of this transition is to guarantee symmetry around BOTTOM. There are two cases that give a transition without Compare Match:

- OCR0A changes its value from MAX, like in [Figure 33](#). When the OCR0A value is MAX the OCn pin value is the same as the result of a down-counting Compare Match. To ensure symmetry around BOTTOM the OCn value at MAX must correspond to the result of an up-counting Compare Match.
- The Timer starts counting from a value higher than the one in OCR0A, and for that reason misses the Compare Match and hence the OCn change that would have happened on the way up.

Compare Match Output Unit

The *Compare Output mode* (COM1x1:0) bits have two functions. The Waveform Generator uses the COM1x1:0 bits for defining the Output Compare (OC1x) state at the next compare match. Secondly the COM1x1:0 bits control the OC1x pin output source. [Figure 44](#) shows a simplified schematic of the logic affected by the COM1x1:0 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 COM1x1:0 bits are shown. When referring to the OC1x state, the reference is for the internal OC1x Register, not the OC1x pin. If a System Reset occur, the OC1x Register is reset to “0”.

Figure 44. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the Output Compare (OC1x) from the Waveform Generator if either of the COM1x1:0 bits are set. However, the OC1x pin direction (input or output) is still controlled by the *Data Direction Register* (DDR) for the port pin. The Data Direction Register bit for the OC1x pin (DDR_OC1x) must be set as output before the OC1x value is visible on the pin. The port override function is generally independent of the Waveform Generation mode, but there are some exceptions. Refer to [Table 44](#), [Table 45](#) and [Table 46](#) for details.

The design of the output compare pin logic allows initialization of the OC1x state before the output is enabled. Note that some COM1x1:0 bit settings are reserved for certain modes of operation. See “[16-bit Timer/Counter Register Description](#)” on page 110.

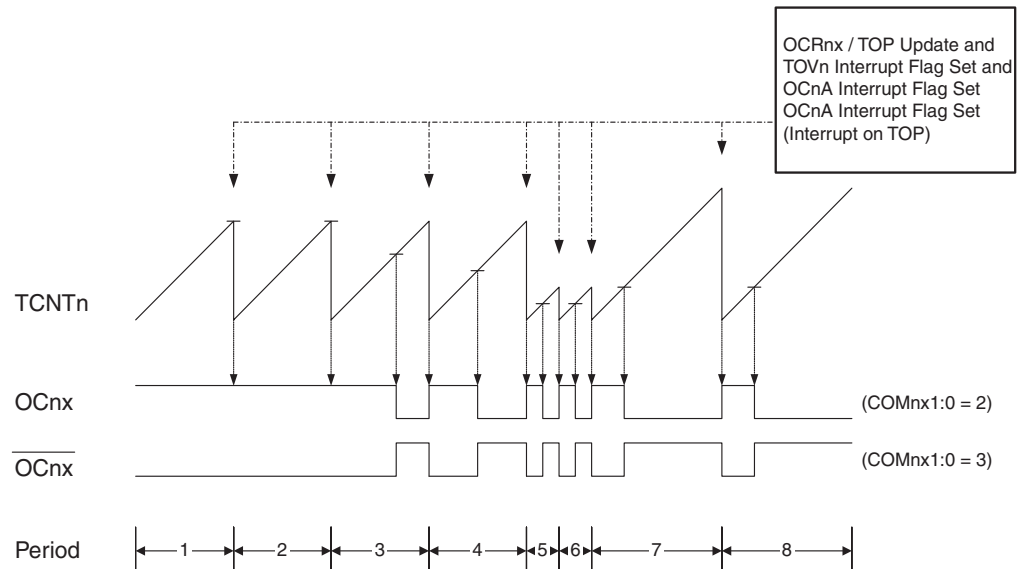
The COM1x1:0 bits have no effect on the Input Capture unit.

Compare Output Mode and Waveform Generation

The Waveform Generator uses the COM1x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM1x1:0 = 0 tells the Waveform Generator that no action on the OC1x Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to [Table 44 on page 110](#). For fast PWM mode refer to [Table 45 on page 111](#), and for phase correct and phase and frequency correct PWM refer to [Table 46 on page 111](#).

A change of the COM1x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC1x strobe bits.

Figure 46. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV1) is set each time the counter reaches TOP. In addition the OC1A or ICF1 Flag is set at the same timer clock cycle as TOV1 is set when either OCR1A or ICR1 is used for defining the TOP value. If one of the interrupts are enabled, the interrupt handler routine can be used for updating the TOP and compare values.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x. Note that when using fixed TOP values the unused bits are masked to zero when any of the OCR1x Registers are written.

The procedure for updating ICR1 differs from updating OCR1A when used for defining the TOP value. The ICR1 Register is not double buffered. This means that if ICR1 is changed to a low value when the counter is running with none or a low prescaler value, there is a risk that the new ICR1 value written is lower than the current value of TCNT1. The result will then be that the counter will miss the compare match at the TOP value. The counter will then have to count to the MAX value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. The OCR1A Register however, is double buffered. This feature allows the OCR1A I/O location to be written anytime. When the OCR1A I/O location is written the value written will be put into the OCR1A Buffer Register. The OCR1A Compare Register will then be updated with the value in the Buffer Register at the next timer clock cycle the TCNT1 matches TOP. The update is done at the same timer clock cycle as the TCNT1 is cleared and the TOV1 Flag is set.

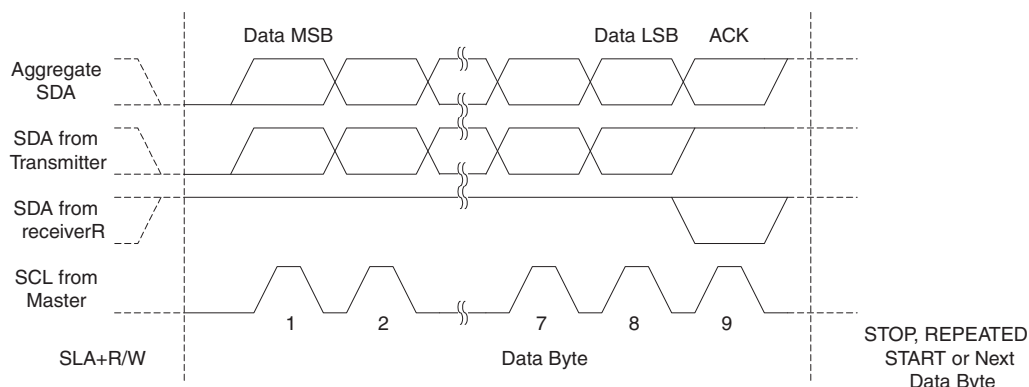
Using the ICR1 Register for defining TOP works well when using fixed TOP values. By using ICR1, the OCR1A Register is free to be used for generating a PWM output on OC1A. However, if the base PWM frequency is actively changed (by changing the TOP value), using the OCR1A as TOP is clearly a better choice due to its double buffer feature.

In fast PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to 2 will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to 3 (See [Table 44 on page 110](#)). The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x Register at the compare match between OCR1x and TCNT1, and clearing (or setting) the OC1x Register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

Data Packet Format

All data packets transmitted on the TWI bus are nine bits long, consisting of one data byte and an acknowledge bit. During a data transfer, the Master generates the clock and the START and STOP conditions, while the receiver is responsible for acknowledging the reception. An Acknowledge (ACK) is signalled by the receiver pulling the SDA line low during the ninth SCL cycle. If the receiver leaves the SDA line high, a NACK is signalled. When the receiver has received the last byte, or for some reason cannot receive any more bytes, it should inform the transmitter by sending a NACK after the final byte. The MSB of the data byte is transmitted first.

Figure 80. Data Packet Format



Combining Address and Data Packets into a Transmission

A transmission basically consists of a START condition, a SLA+R/W, one or more data packets and a STOP condition. An empty message, consisting of a START followed by a STOP condition, is illegal. Note that the wired-ANDing of the SCL line can be used to implement handshaking between the Master and the Slave. The Slave can extend the SCL low period by pulling the SCL line low. This is useful if the clock speed set up by the Master is too fast for the Slave, or the Slave needs extra time for processing between the data transmissions. The Slave extending the SCL low period will not affect the SCL high period, which is determined by the Master. As a consequence, the Slave can reduce the TWI data transfer speed by prolonging the SCL duty cycle.

[Figure 81](#) shows a typical data transmission. Note that several data bytes can be transmitted between the SLA+R/W and the STOP condition, depending on the software protocol implemented by the application software.

Figure 81. Typical Data Transmission

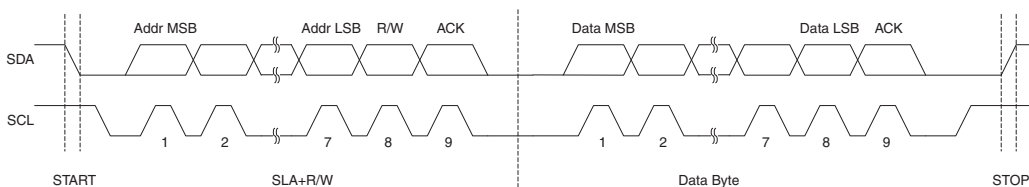
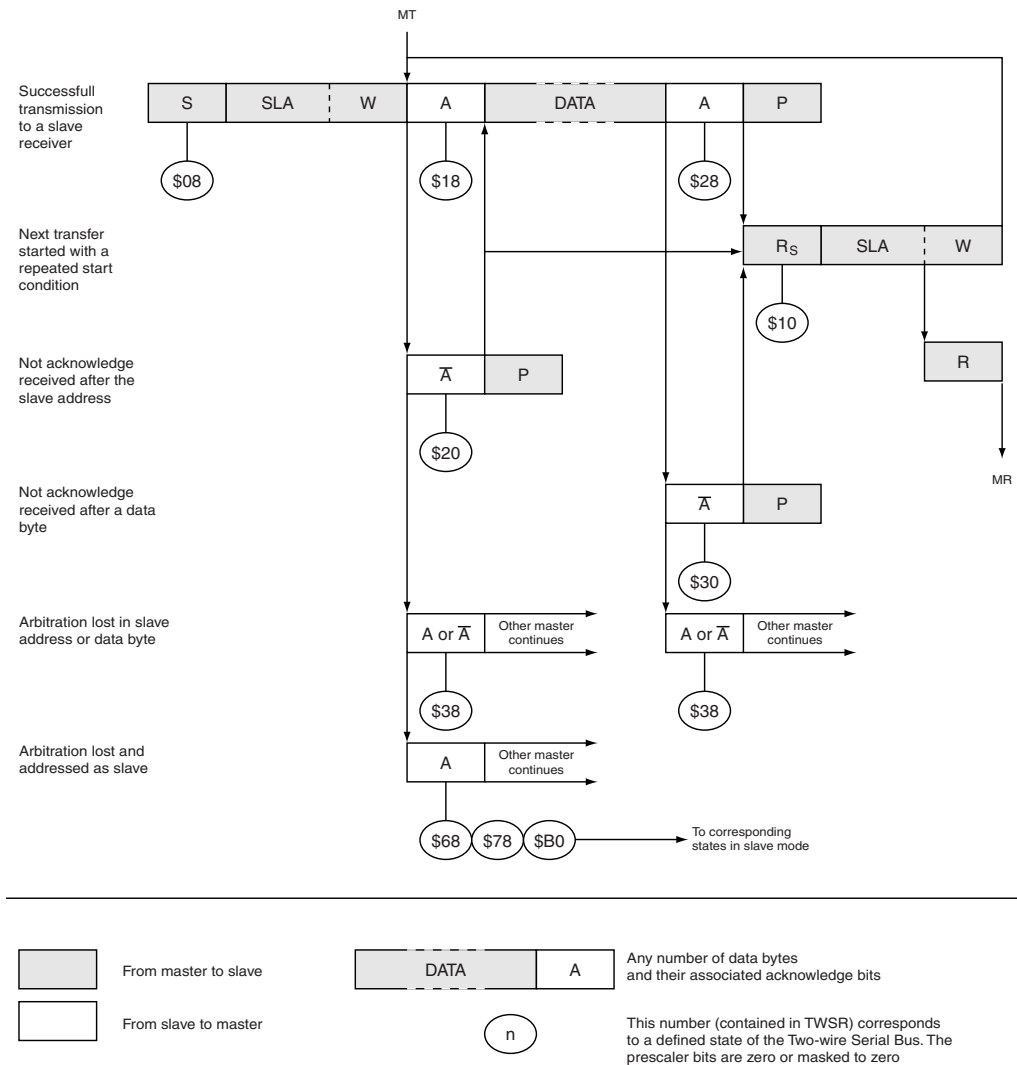


Figure 87. Formats and States in the Master Transmitter Mode



Master Receiver Mode

In the Master Receiver mode, a number of data bytes are received from a Slave Transmitter (see [Figure 88](#)). In order to enter a Master mode, a START condition must be transmitted. The format of the following address packet determines whether Master Transmitter or Master Receiver mode is to be entered. If SLA+W is transmitted, MT mode is entered, if SLA+R is transmitted, MR mode is entered. All the status codes mentioned in this section assume that the prescaler bits are zero or are masked to zero.

Note that the Two-wire Serial Interface Data Register – TWDR does not reflect the last byte present on the bus when waking up from these sleep modes.

Boundary-scan Chain

The Boundary-scan chain has the capability of driving and observing the logic levels on the digital I/O pins, as well as the boundary between digital and analog logic for analog circuitry having Off-chip connection.

Scanning the Digital Port Pins

Figure 116 shows the Boundary-scan Cell for a bi-directional port pin with pull-up function. The cell consists of a standard Boundary-scan cell for the Pull-up Enable – PUE_{xn} – function, and a bi-directional pin cell that combines the three signals Output Control – OC_{xn}, Output Data – OD_{xn}, and Input Data – ID_{xn}, into only a two-stage Shift Register. The port and pin indexes are not used in the following description.

The Boundary-scan logic is not included in the figures in the datasheet. Figure 117 shows a simple digital Port Pin as described in the section “I/O Ports” on page 50. The Boundary-scan details from Figure 116 replaces the dashed box in Figure 117.

When no alternate port function is present, the Input Data – ID – corresponds to the PIN_{xn} Register value (but ID has no synchronizer), Output Data corresponds to the PORT Register, Output Control corresponds to the Data Direction – DD Register, and the Pull-up Enable – PUE_{xn} – corresponds to logic expression $PUD \cdot DD_{xn} \cdot PORT_{xn}$.

Digital alternate port functions are connected outside the dotted box in Figure 117 to make the scan chain read the actual pin value. For Analog function, there is a direct connection from the external pin to the analog circuit, and a scan chain is inserted on the interface between the digital logic and the analog circuitry.

Figure 116. Boundary-scan Cell for Bidirectional Port Pin with Pull-up Function.

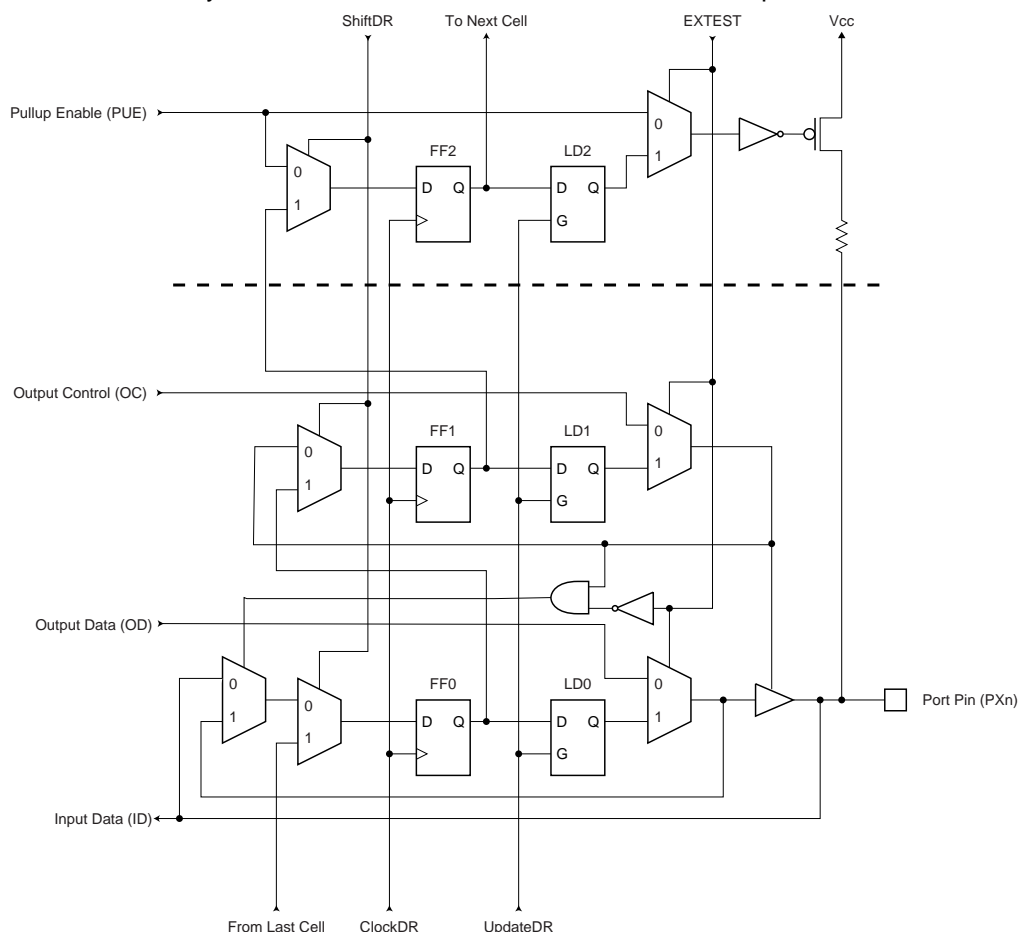


Table 94. ATmega16 Boundary-scan Order (Continued)

| Bit Number | Signal Name | Module |
|------------|-------------------|--------|
| 111 | MUXEN_6 | |
| 110 | MUXEN_5 | |
| 109 | MUXEN_4 | |
| 108 | MUXEN_3 | |
| 107 | MUXEN_2 | |
| 106 | MUXEN_1 | |
| 105 | MUXEN_0 | |
| 104 | NEGSEL_2 | |
| 103 | NEGSEL_1 | |
| 102 | NEGSEL_0 | |
| 101 | PASSEN | |
| 100 | PRECH | |
| 99 | SCTEST | |
| 98 | ST | |
| 97 | VCCREN | |
| 96 | PB0.Data | Port B |
| 95 | PB0.Control | |
| 94 | PB0.Pullup_Enable | |
| 93 | PB1.Data | |
| 92 | PB1.Control | |
| 91 | PB1.Pullup_Enable | |
| 90 | PB2.Data | |
| 89 | PB2.Control | |
| 88 | PB2.Pullup_Enable | |
| 87 | PB3.Data | |
| 86 | PB3.Control | |
| 85 | PB3.Pullup_Enable | |
| 84 | PB4.Data | |
| 83 | PB4.Control | |
| 82 | PB4.Pullup_Enable | |
| 81 | PB5.Data | |
| 80 | PB5.Control | |
| 79 | PB5.Pullup_Enable | |
| 78 | PB6.Data | |
| 77 | PB6.Control | |
| 76 | PB6.Pullup_Enable | |

Performing Page Erase by SPM

To execute Page Erase, set up the address in the Z-pointer, write “X0000011” to SPMCR and execute SPM within four clock cycles after writing SPMCR. 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 must be written zero during this operation.

- Page Erase to the RWW section: The NRWW section can be read during the page erase.
- Page Erase to the NRWW section: The CPU is halted during the operation.

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 SPMCR and execute SPM within four clock cycles after writing SPMCR. 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 RWWSRE bit in SPMCR. 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.

Note: If the EEPROM is written in the middle of an SPM Page Load operation, all data loaded will be lost.

Performing a Page Write

To execute Page Write, set up the address in the Z-pointer, write “X0000101” to SPMCR and execute SPM within four clock cycles after writing SPMCR. 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 zero during this operation.

- Page Write to the RWW section: The NRWW section can be read during the Page Write.
- Page Write to the NRWW section: The CPU is halted during the operation.

Using the SPM Interrupt

If the SPM interrupt is enabled, the SPM interrupt will generate a constant interrupt when the SPMEN bit in SPMCR is cleared. This means that the interrupt can be used instead of polling the SPMCR Register in software. When using the SPM interrupt, the Interrupt Vectors should be moved to the BLS section to avoid that an interrupt is accessing the RWW section when it is blocked for reading. How to move the interrupts is described in [“Interrupts” on page 45](#).

Consideration while Updating BLS

Special care must be taken if the user allows the Boot Loader section to be updated by leaving Boot Lock bit11 unprogrammed. An accidental write to the Boot Loader itself can corrupt the entire Boot Loader, and further software updates might be impossible. If it is not necessary to change the Boot Loader software itself, it is recommended to program the Boot Lock bit11 to protect the Boot Loader software from any internal software changes.

Prevent Reading the RWW Section during Self-Programming

During Self-Programming (either Page Erase or Page Write), the RWW section is always blocked for reading. The user software itself must prevent that this section is addressed during the Self-Programming operation. The RWWSB in the SPMCR will be set as long as the RWW section is busy. During self-programming the Interrupt Vector table should be moved to the BLS as described in [“Interrupts” on page 45](#), or the interrupts must be disabled. Before addressing the RWW section after the programming is completed, the user software must clear the RWWSB by writing the RWWSRE. See [“Simple Assembly Code Example for a Boot Loader” on page 256](#) for an example.

Reset Register

The Reset Register is a Test Data Register used to reset the part during programming. It is required to reset the part before entering programming mode.

A high value in the Reset Register corresponds to pulling the external Reset low. The part is reset as long as there is a high value present in the Reset Register. Depending on the Fuse settings for the clock options, the part will remain reset for a Reset Time-out Period (refer to [“Clock Sources” on page 25](#)) after releasing the Reset Register. The output from this Data Register is not latched, so the reset will take place immediately, as shown in [Figure 115 on page 230](#).

Programming Enable Register

The Programming Enable Register is a 16-bit register. The contents of this register is compared to the programming enable signature, binary code 1010_0011_0111_0000. When the contents of the register is equal to the programming enable signature, programming via the JTAG port is enabled. The register is reset to 0 on Power-on Reset, and should always be reset when leaving Programming mode.

Figure 140. Programming Enable Register

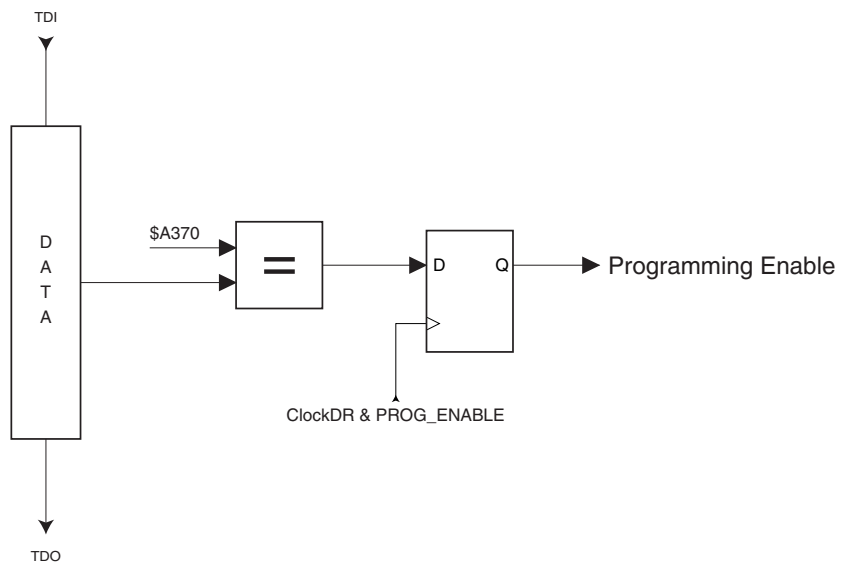


Figure 200. Calibrated 2 MHz RC Oscillator Frequency vs. Temperature

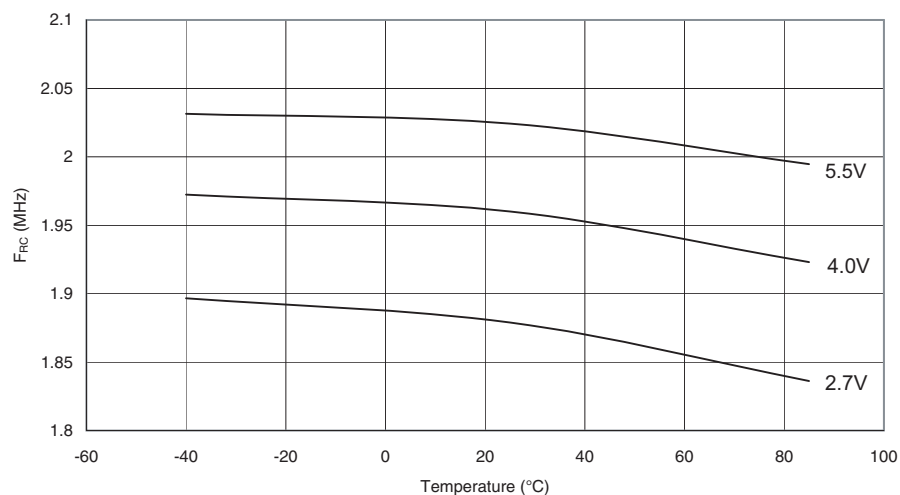
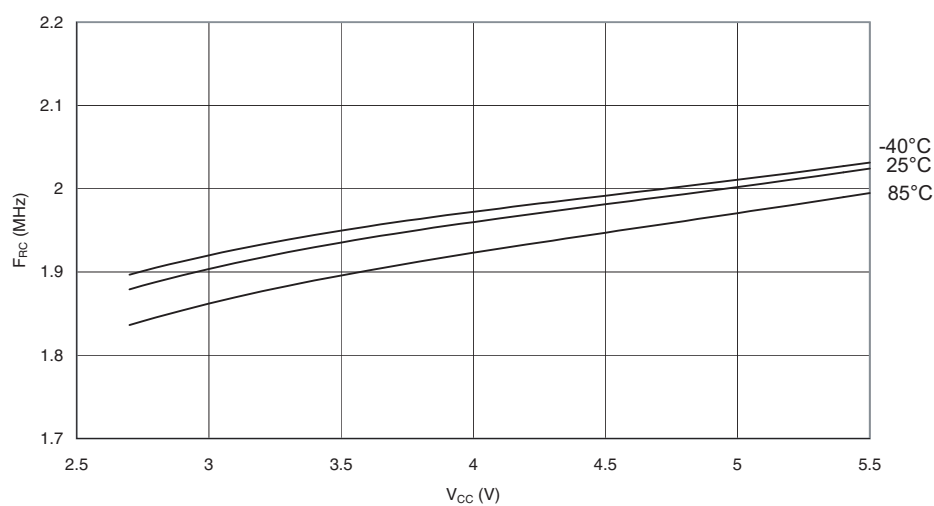


Figure 201. Calibrated 2 MHz RC Oscillator Frequency vs. V_{CC}



Errata

The revision letter in this section refers to the revision of the ATmega16 device.

ATmega16(L) Rev. M

- First Analog Comparator conversion may be delayed
- Interrupts may be lost when writing the timer registers in the asynchronous timer
- IDCODE masks data from TDI input
- Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request

1. First Analog Comparator conversion may be delayed

If the device is powered by a slow rising V_{CC} , the first Analog Comparator conversion will take longer than expected on some devices.

Problem Fix/Workaround

When the device has been powered or reset, disable then enable the Analog Comparator before the first conversion.

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

The interrupt will be lost if a timer register that is synchronized to the asynchronous timer clock is written when the asynchronous Timer/Counter register(TCNTx) is 0x00.

Problem Fix / Workaround

Always check that the asynchronous Timer/Counter register neither have the value 0xFF nor 0x00 before writing to the asynchronous Timer Control Register(TCCRx), asynchronous Timer Counter Register(TCNTx), or asynchronous Output Compare Register(OCRx).

3. IDCODE masks data from TDI input

The JTAG instruction IDCODE is not working correctly. Data to succeeding devices are replaced by all-ones during Update-DR.

Problem Fix / Workaround

- If ATmega16 is the only device in the scan chain, the problem is not visible.
- Select the Device ID Register of the ATmega16 by issuing the IDCODE instruction or by entering the Test-Logic-Reset state of the TAP controller to read out the contents of its Device ID Register and possibly data from succeeding devices of the scan chain. Issue the BYPASS instruction to the ATmega16 while reading the Device ID Registers of preceding devices of the boundary scan chain.
- If the Device IDs of all devices in the boundary scan chain must be captured simultaneously, the ATmega16 must be the first device in the chain.

4. Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request.

Reading EEPROM by using the ST or STS command to set the EERE bit in the EECR register triggers an unexpected EEPROM interrupt request.

Problem Fix / Workaround

Always use OUT or SBI to set EERE in EECR.

ATmega16(L) Rev. L

- First Analog Comparator conversion may be delayed
- Interrupts may be lost when writing the timer registers in the asynchronous timer
- IDCODE masks data from TDI input
- Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request

1. First Analog Comparator conversion may be delayed

If the device is powered by a slow rising V_{CC} , the first Analog Comparator conversion will take longer than expected on some devices.

