



Welcome to [E-XFL.COM](https://www.e-xfl.com)

### What is "[Embedded - Microcontrollers](#)"?

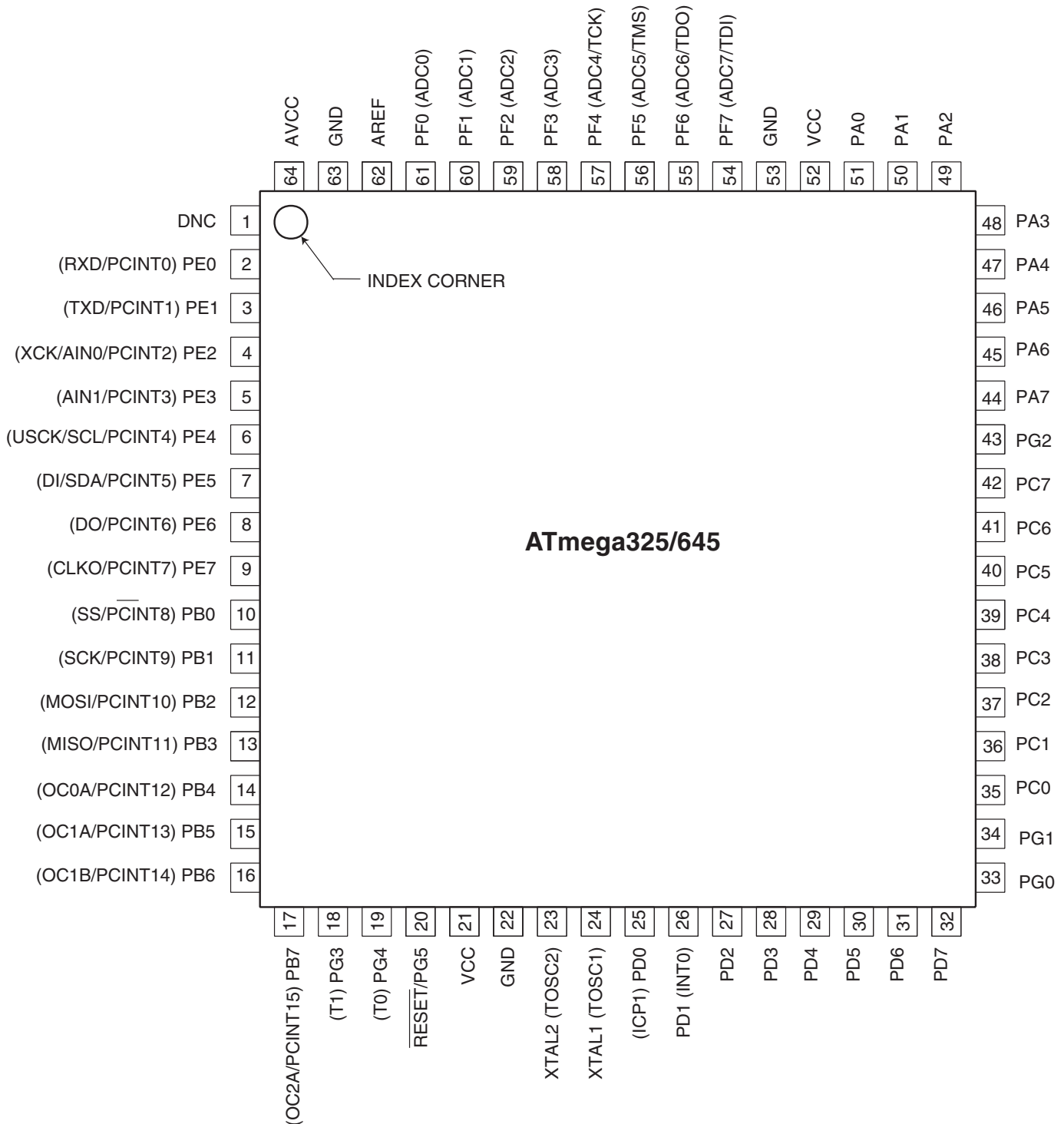
"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	8MHz
Connectivity	SPI, UART/USART, USI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	53
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-QFN (9x9)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/atmega645v-8mi">https://www.e-xfl.com/product-detail/microchip-technology/atmega645v-8mi</a>

**Figure 1-2.** Pinout ATmega325/645



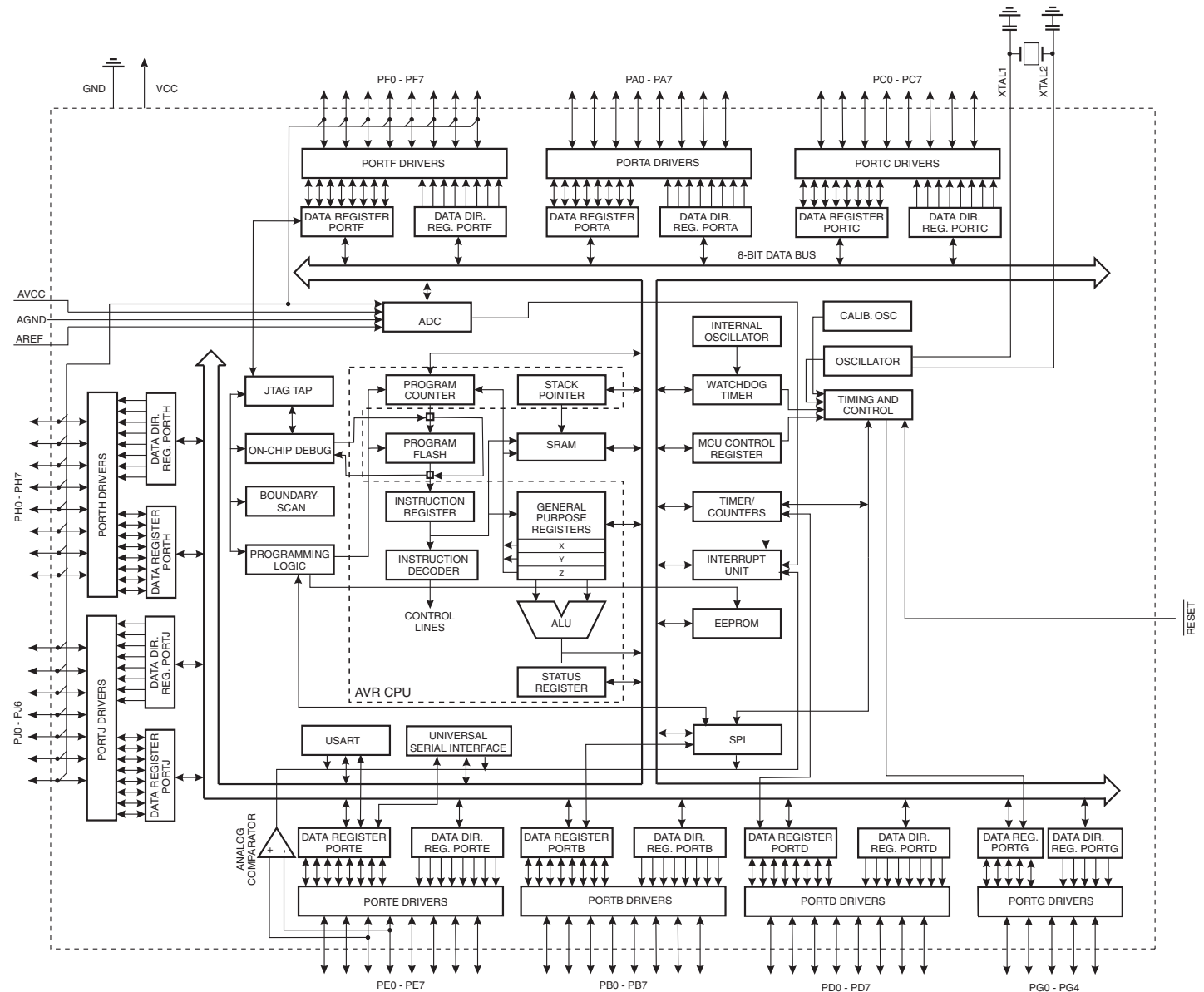
**Note:** The large center pad underneath the QFN/MLF packages is made of metal and internally connected to GND. It should be soldered or glued to the board to ensure good mechanical stability. If the center pad is left unconnected, the package might loosen from the board.

## 2. Overview

The Atmel ATmega325/3250/645/6450 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the Atmel ATmega325/3250/645/6450 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

### 2.1 Block Diagram

Figure 2-1. Block Diagram



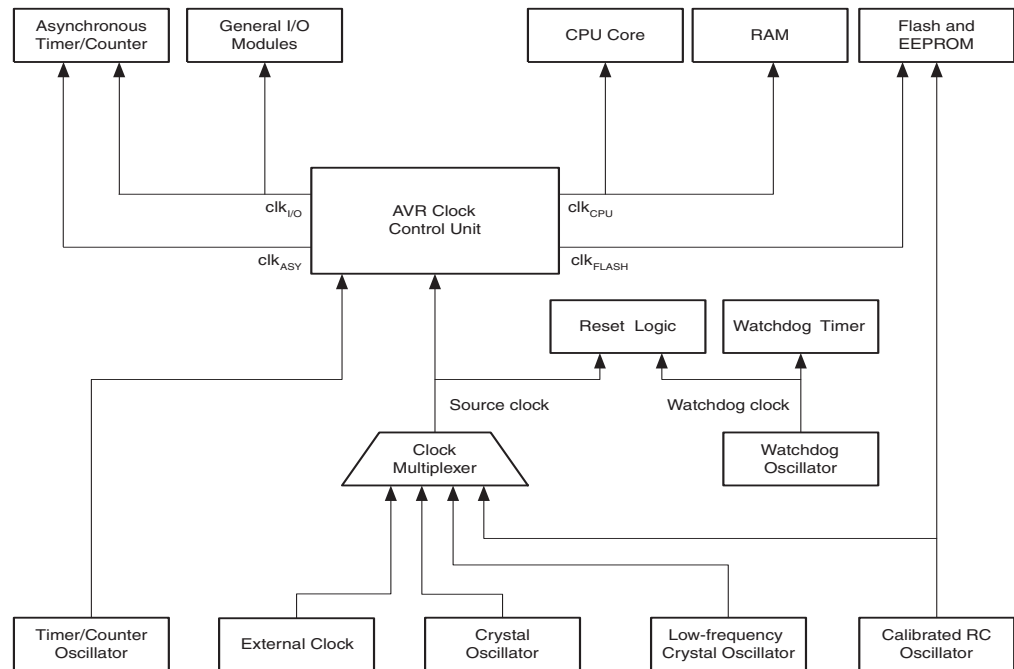
The Atmel®AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The

## 9. System Clock and Clock Options

### 9.1 Clock Systems and their Distribution

Figure 9-1 on page 26 presents the principal clock systems in the AVR and their distribution. All of the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes, as described in “Power Management and Sleep Modes” on page 35. The clock systems are detailed below.

**Figure 9-1.** Clock Distribution



#### 9.1.1 CPU Clock – $\text{clk}_{\text{CPU}}$

The CPU clock is routed to parts of the system concerned with operation of the AVR core. Examples of such modules are the General Purpose Register File, the Status Register and the data memory holding the Stack Pointer. Halting the CPU clock inhibits the core from performing general operations and calculations.

#### 9.1.2 I/O Clock – $\text{clk}_{\text{I/O}}$

The I/O clock is used by the majority of the I/O modules, like Timer/Counters, SPI, and USART. The I/O clock is also used by the External Interrupt module, but note that some external interrupts are detected by asynchronous logic, allowing such interrupts to be detected even if the I/O clock is halted. Also note that start condition detection in the USI module is carried out asynchronously when  $\text{clk}_{\text{I/O}}$  is halted, enabling USI start condition detection in all sleep modes.

#### 9.1.3 Flash Clock – $\text{clk}_{\text{FLASH}}$

The Flash clock controls operation of the Flash interface. The Flash clock is usually active simultaneously with the CPU clock.

in the Application section and Boot Lock bit BLB12 is programmed, interrupts are disabled while executing from the Boot Loader section. Refer to the section [“Boot Loader Support – Read-While-Write Self-Programming” on page 251](#) for details on Boot Lock bits.

- **Bit 0 – IVCE: Interrupt Vector Change Enable**

The IVCE bit must be written to logic one to enable change of the IVSEL bit. IVCE is cleared by hardware four cycles after it is written or when IVSEL is written. Setting the IVCE bit will disable interrupts, as explained in the IVSEL description above. See Code Example below.

## Assembly Code Example

```
Move_interrupts:
;Get MCUCR
in r16, MCUCR
mov r17, r16
; Enable change of Interrupt Vectors
ori r16, (1<<IVCE)
out MCUCR, r16
; Move interrupts to Boot Flash section
ori r17, (1<<IVSEL)
out MCUCR, r17
ret
```

## C Code Example

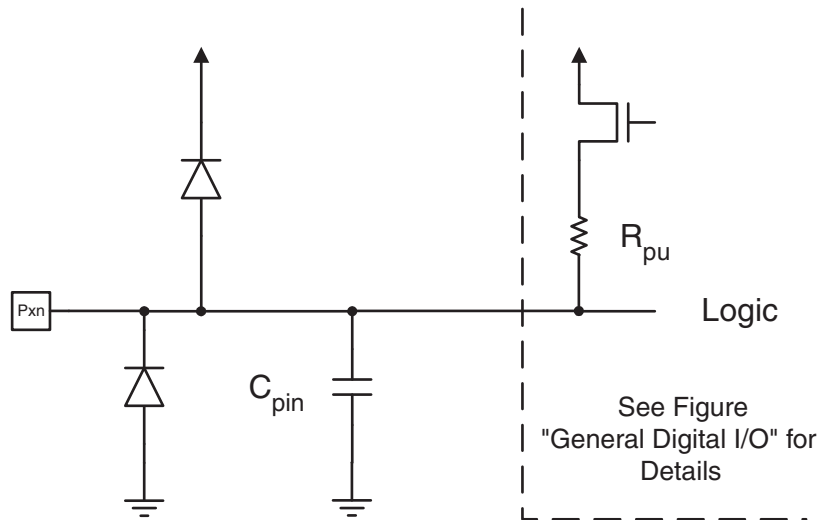
```
void Move_interrupts(void)
{
/* Enable change of Interrupt Vectors */
MCUCR |= (1<<IVCE);
/* Move interrupts to Boot Flash section */
MCUCR |= (1<<IVSEL);
}
```

## 14. I/O-Ports

### 14.1 Overview

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. Port B has a higher pin driver strength than the other ports, but all the pin drivers are 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 14-1. Refer to “Electrical Characteristics” on page 297 for a complete list of parameters. If exceeding the pin voltage “Absolute Maximum Ratings”, resulting currents can harm the device if not limited accordingly. For segment pins used as general I/O, the same situation can also influence the LCD voltage level.

**Figure 14-1.** I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case “x” represents the numbering letter for the port, and a lower case “n” represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. For example, PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn. The physical I/O Registers and bit locations are listed in “Register Description” on page 81.

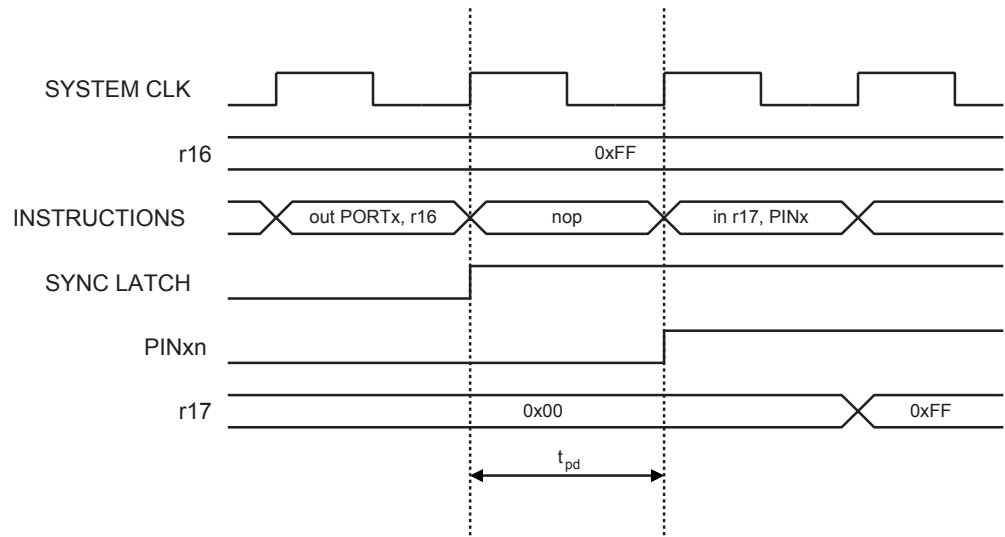
Three I/O memory address locations are allocated for each port, one each for the Data Register – PORTx, Data Direction Register – DDRx, and the Port Input Pins – PINx. The Port Input Pins I/O location is read only, while the Data Register and the Data Direction Register are read/write. However, writing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

Using the I/O port as General Digital I/O is described in “Ports as General Digital I/O” on page 61. 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

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  $t_{pd,max}$  and  $t_{pd,min}$ , a single signal transition on the pin will be delayed between  $\frac{1}{2}$  and  $1\frac{1}{2}$  system clock period depending upon the time of assertion.

When reading back a software assigned pin value, a nop instruction must be inserted as indicated in Figure 14-4. The out instruction sets the “SYNC LATCH” signal at the positive edge of the clock. In this case, the delay  $t_{pd}$  through the synchronizer is 1 system clock period.

**Figure 14-4.** Synchronization when Reading a Software Assigned Pin Value



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.

- **TDO, ADC6 – Port F, Bit 6**

ADC6, Analog to Digital Converter, Channel 6.

TDO, JTAG Test Data Out: Serial output data from Instruction Register or Data Register. When the JTAG interface is enabled, this pin can not be used as an I/O pin. In TAP states that shift out data, the TDO pin drives actively. In other states the pin is pulled high.

- **TMS, ADC5 – Port F, Bit 5**

ADC5, Analog to Digital Converter, Channel 5.

TMS, JTAG Test mode Select: This pin is used for navigating through the TAP-controller state machine. When the JTAG interface is enabled, this pin can not be used as an I/O pin.

- **TCK, ADC4 – Port F, Bit 4**

ADC4, Analog to Digital Converter, Channel 4.

TCK, JTAG Test Clock: JTAG operation is synchronous to TCK. When the JTAG interface is enabled, this pin can not be used as an I/O pin.

- **ADC3 - ADC0 – Port F, Bit 3:0**

Analog to Digital Converter, Channel 3-0.

**Table 14-12.** Overriding Signals for Alternate Functions in PF7:PF4

Signal Name	PF7/ADC7/TDI	PF6/ADC6/TDO	PF5/ADC5/TMS	PF4/ADC4/TCK
PUOE	JTAGEN	JTAGEN	JTAGEN	JTAGEN
PUOV	1	1	1	1
DDOE	JTAGEN	JTAGEN	JTAGEN	JTAGEN
DDOV	0	SHIFT_IR + SHIFT_DR	0	0
PVOE	0	JTAGEN	0	0
PVOV	0	TDO	0	0
PTOE	–	–	–	–
DIEOE	JTAGEN	JTAGEN	JTAGEN	JTAGEN
DIEOV	0	0	0	0
DI	–	–	–	–
AIO	TDI ADC7 INPUT	ADC6 INPUT	TMS ADC5 INPUT	TCK ADC4 INPUT



The definitions in [Table 15-1](#) are also used extensively throughout the document.

**Table 15-1.** Definitions of Timer/Counter values.

BOTTOM	The counter reaches the BOTTOM when it becomes 0x00.
MAX	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR0A Register. The assignment is dependent on the mode of operation.

## 15.2.2 Registers

The Timer/Counter (TCNT0) and Output Compare Register (OCR0A) are 8-bit registers. Interrupt request (abbreviated to Int.Req. in the figure) signals are all visible in the Timer Interrupt Flag Register (TIFR0). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSK0). TIFR0 and TIMSK0 are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T0 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk<sub>T0</sub>).

The double buffered Output Compare Register (OCR0A) is compared with the Timer/Counter value at all times. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pin (OC0A). See [“Output Compare Unit” on page 87](#). for details. The compare match event will also set the Compare Flag (OCF0A) which can be used to generate an Output Compare interrupt request.

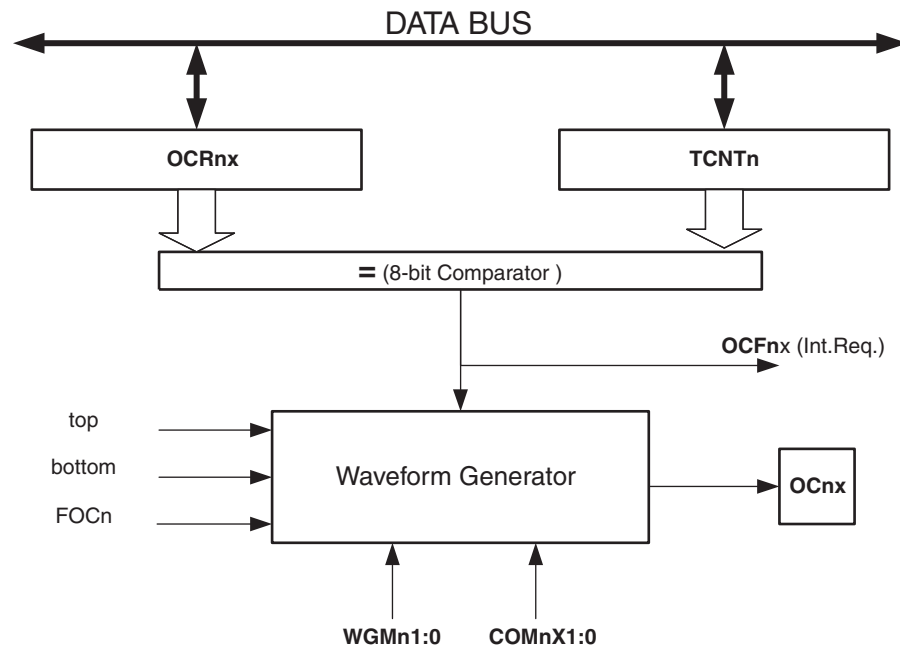
## 15.3 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the Clock Select logic which is controlled by the Clock Select (CS02:0) bits located in the Timer/Counter Control Register (TCCR0A). For details on clock sources and prescaler, see [“Timer/Counter0 and Timer/Counter1 Prescalers” on page 99](#).

Waveform Generator uses the match signal to generate an output according to operating mode set by the WGM01:0 bits and Compare Output mode (COM0A1:0) bits. The max and bottom signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (See “Modes of Operation” on page 90.).

Figure 15-3 shows a block diagram of the Output Compare unit.

**Figure 15-3.** Output Compare Unit, Block Diagram



- 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.

## 15.8 Timer/Counter Timing Diagrams

The Timer/Counter is a synchronous design and the timer clock ( $\text{clk}_{T0}$ ) is therefore shown as a clock enable signal in the following figures. The figures include information on when Interrupt Flags are set. Figure 15-8 contains timing data for basic Timer/Counter operation. The figure shows the count sequence close to the MAX value in all modes other than phase correct PWM mode.

**Figure 15-8.** Timer/Counter Timing Diagram, no Prescaling

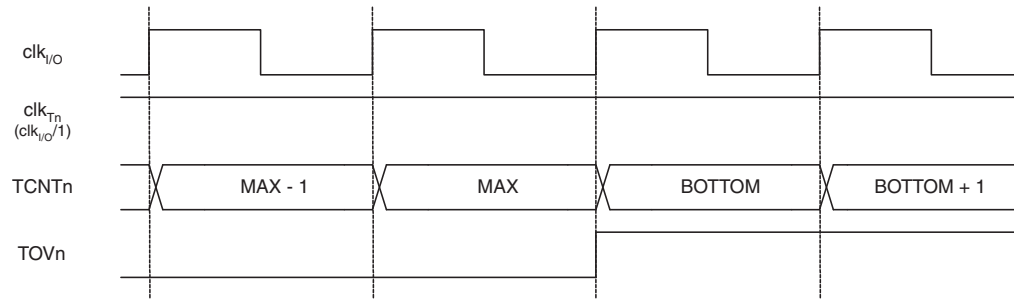


Figure 15-9 shows the same timing data, but with the prescaler enabled.

**Figure 15-9.** Timer/Counter Timing Diagram, with Prescaler ( $f_{\text{clk}_{I/O}/8}$ )

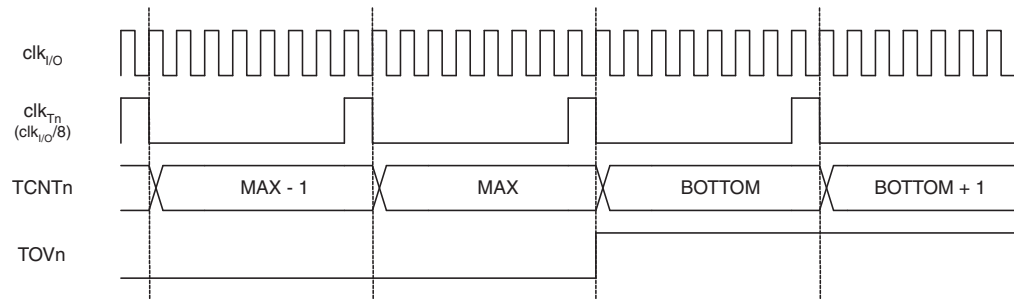
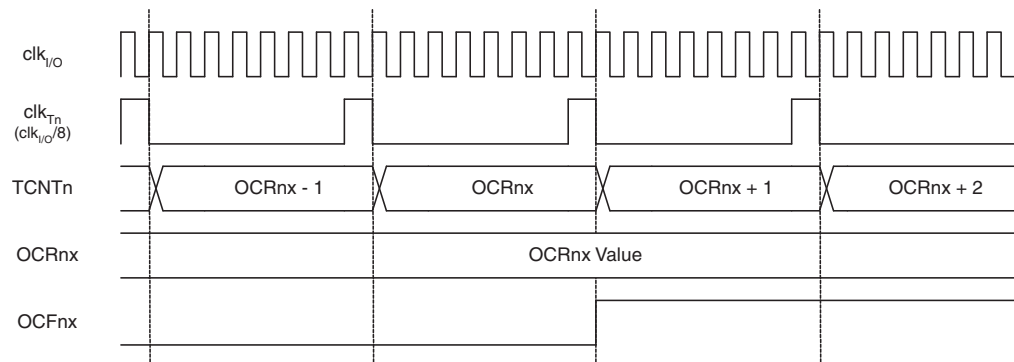


Figure 15-10 shows the setting of OCF0A in all modes except CTC mode.

**Figure 15-10.** Timer/Counter Timing Diagram, Setting of OCF0A, with Prescaler ( $f_{\text{clk}_{I/O}/8}$ )



## 16.0.1 Internal Clock Source

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

## 16.0.2 Prescaler Reset

The prescaler is free running, i.e., operates independently of the Clock Select logic of the Timer/Counter, and it is shared by Timer/Counter1 and Timer/Counter0. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler ( $6 > CSn2:0 > 1$ ). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

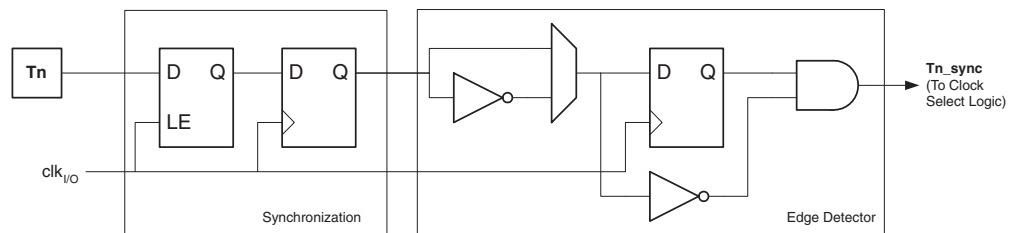
It is possible to use the prescaler reset for synchronizing the Timer/Counter to program execution. However, care must be taken if the other Timer/Counter that shares the same prescaler also uses prescaling. A prescaler reset will affect the prescaler period for all Timer/Counter it is connected to.

## 16.0.3 External Clock Source

An external clock source applied to the T1/T0 pin can be used as Timer/Counter clock ( $clk_{T1}/clk_{T0}$ ). The T1/T0 pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 1 shows a functional equivalent block diagram of the T1/T0 synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ( $clk_{I/O}$ ). The latch is transparent in the high period of the internal system clock.

The edge detector generates one  $clk_{T1}/clk_{T0}$  pulse for each positive ( $CSn2:0 = 7$ ) or negative ( $CSn2:0 = 6$ ) edge it detects.

**Figure 1. T1/T0 Pin Sampling**



The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the T1/T0 pin to the counter is updated.

Enabling and disabling of the clock input must be done when T1/T0 has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

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

cleared by software (writing a logical one to the I/O bit location). For measuring frequency only, the clearing of the ICF1 Flag is not required (if an interrupt handler is used).

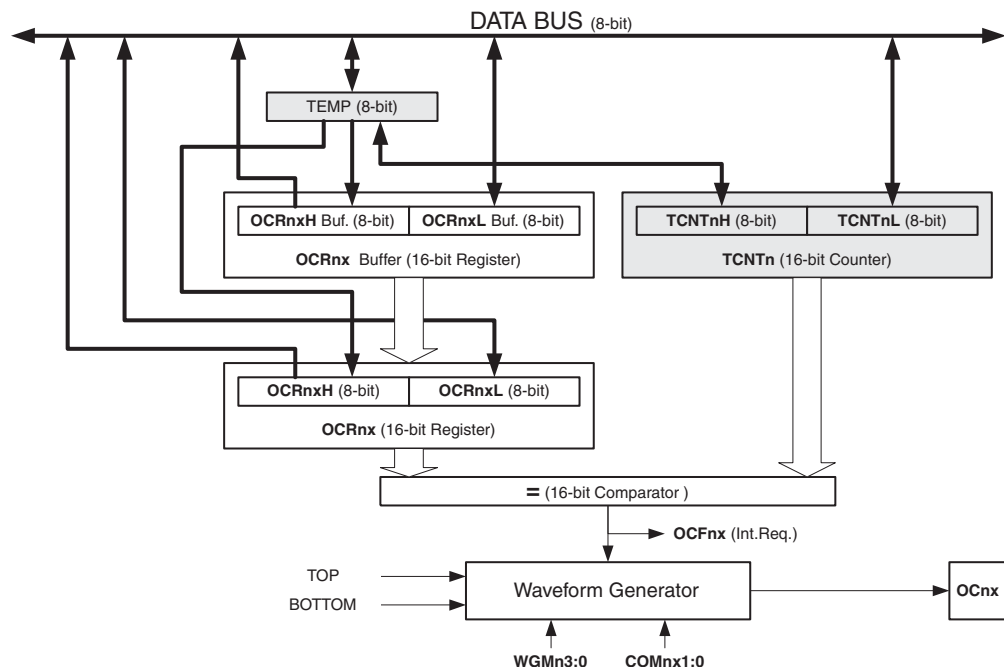
## 17.7 Output Compare Units

The 16-bit comparator continuously compares TCNT1 with the *Output Compare Register* (OCR1x). If TCNT equals OCR1x the comparator signals a match. A match will set the *Output Compare Flag* (OCF1x) at the next timer clock cycle. If enabled (OCIE1x = 1), the Output Compare Flag generates an Output Compare interrupt. The OCF1x Flag is automatically cleared when the interrupt is executed. Alternatively the OCF1x Flag can be cleared by software by writing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the *Waveform Generation mode* (WGM13:0) bits and *Compare Output mode* (COM1x1:0) bits. The TOP and BOTTOM signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (See “Modes of Operation” on page 114.)

A special feature of Output Compare unit A allows it to define the Timer/Counter TOP value (i.e., counter resolution). In addition to the counter resolution, the TOP value defines the period time for waveforms generated by the Waveform Generator.

Figure 17-4 shows a block diagram of the Output Compare unit. The small “n” in the register and bit names indicates the device number (n = 1 for Timer/Counter 1), and the “x” indicates Output Compare unit (A/B). The elements of the block diagram that are not directly a part of the Output Compare unit are gray shaded.

**Figure 17-4.** Output Compare Unit, Block Diagram



The OCR1x Register is double buffered when using any of the twelve *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 OCR1x Compare Register to either TOP or BOTTOM of the counting sequence. The synchronization

When the ICR1 is used as TOP value (see description of the WGM13:0 bits located in the TCCR1A and the TCCR1B Register), the ICP1 is disconnected and consequently the Input Capture function is disabled.

- **Bit 5 – Reserved Bit**

This bit is reserved for future use. For ensuring compatibility with future devices, this bit must be written to zero when TCCR1B is written.

- **Bit 4:3 – WGM13:2: Waveform Generation Mode**

See TCCR1A Register description.

- **Bit 2:0 – CS12:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter, see [Figure 17-10](#) and [Figure 17-11](#).

**Table 17-6.** Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk <sub>I/O</sub> /1 (No prescaling)
0	1	0	clk <sub>I/O</sub> /8 (From prescaler)
0	1	1	clk <sub>I/O</sub> /64 (From prescaler)
1	0	0	clk <sub>I/O</sub> /256 (From prescaler)
1	0	1	clk <sub>I/O</sub> /1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter1, transitions on the T1 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

### 17.11.3 TCCR1C – Timer/Counter1 Control Register C

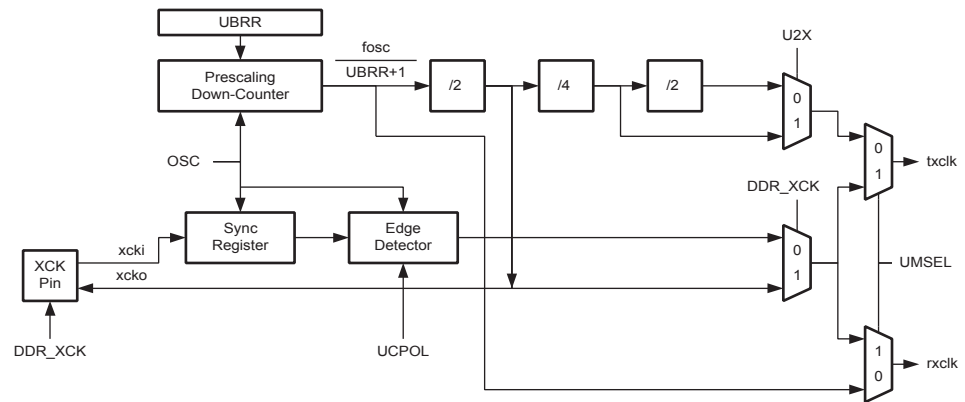
Bit	7	6	5	4	3	2	1	0	
(0x82)	<b>FOC1A</b>	<b>FOC1B</b>	–	–	–	–	–	–	<b>TCCR1C</b>
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC1A: Force Output Compare for Unit A**

- **Bit 6 – FOC1B: Force Output Compare for Unit B**

The FOC1A/FOC1B bits are only active when the WGM13:0 bits specifies a non-PWM mode. However, for ensuring compatibility with future devices, these bits must be set to zero when TCCR1A is written when operating in a PWM mode. When writing a logical one to the FOC1A/FOC1B bit, an immediate compare match is forced on the Waveform Generation unit. The OC1A/OC1B output is changed according to its COM1x1:0 bits setting. Note that the FOC1A/FOC1B bits are implemented as strobes. Therefore it is the value present in the COM1x1:0 bits that determine the effect of the forced compare.

**Figure 20-2.** Clock Generation Logic, Block Diagram



Signal description:

<b>txclk</b>	Transmitter clock (Internal Signal).
<b>rxclk</b>	Receiver base clock (Internal Signal).
<b>xcki</b>	Input from XCK pin (internal Signal). Used for synchronous slave operation.
<b>xcko</b>	Clock output to XCK pin (Internal Signal). Used for synchronous master operation.
<b>fosc</b>	XTAL pin frequency (System Clock).

## 20.3.1 Internal Clock Generation – The Baud Rate Generator

Internal clock generation is used for the asynchronous and the synchronous master modes of operation. The description in this section refers to [Figure 20-2](#).

The USART Baud Rate Register (UBRR) and the down-counter connected to it function as a programmable prescaler or baud rate generator. The down-counter, running at system clock ( $f_{osc}$ ), is loaded with the UBRR value each time the counter has counted down to zero or when the UBRR Register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output ( $= f_{osc}/(UBRR+1)$ ). The Transmitter divides the baud rate generator clock output by 2, 8 or 16 depending on mode. The baud rate generator output is used directly by the Receiver's clock and data recovery units. However, the recovery units use a state machine that uses 2, 8 or 16 states depending on mode set by the state of the UMSELn, U2Xn and DDR\_XCK bits.

[Table 20-1](#) contains equations for calculating the baud rate (in bits per second) and for calculating the UBRR value for each mode of operation using an internally generated clock source.

- 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.

## 26.8.4 Using the SPM Interrupt

If the SPM interrupt is enabled, the SPM interrupt will generate a constant interrupt when the SPMEN bit in SPMCSR is cleared. This means that the interrupt can be used instead of polling the SPMCSR 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 49](#).

## 26.8.5 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.

## 26.8.6 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 SPMCSR 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 49](#), 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 260](#) for an example.

## 26.8.7 Setting the Boot Loader Lock Bits by SPM

To set the Boot Loader Lock bits and general Lock bits, write the desired data to R0, write “X0001001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR.

Bit	7	6	5	4	3	2	1	0
R0	1	1	BLB12	BLB11	BLB02	BLB01	LB2	LB1

See [Table 26-2](#) and [Table 26-3](#) for how the different settings of the Boot Loader bits affect the Flash access.

If bits 5..0 in R0 are cleared (zero), the corresponding Lock bit will be programmed if an SPM instruction is executed within four cycles after BLBSET and SPMEN are set in SPMCSR. The Z-pointer is don't care during this operation, but for future compatibility it is recommended to load the Z-pointer with 0x0001 (same as used for reading the Lock bits). For future compatibility it is also recommended to set bits 7, and 6 in R0 to “1” when writing the Lock bits. When programming the Lock bits the entire Flash can be read during the operation.

## 26.8.8 EEPROM Write Prevents Writing to SPMCSR

Note that an EEPROM write operation will block all software programming to Flash. Reading the Fuses and Lock bits from software will also be prevented during the EEPROM write operation. It is recommended that the user checks the status bit (EWE) in the EECR Register and verifies that the bit is cleared before writing to the SPMCSR Register.



**Table 27-15. Serial Programming Instruction Set**

Instruction/Operation	Instruction Format			
	Byte 1	Byte 2	Byte 3	Byte4
Read Lock bits	\$58	\$00	\$00	data byte out
Read Signature Byte	\$30	\$00	0000 000aa	data byte out
Read Fuse bits	\$50	\$00	\$00	data byte out
Read Fuse High bits	\$58	\$08	\$00	data byte out
Read Extended Fuse Bits	\$50	\$08	\$00	data byte out
Read Calibration Byte	\$38	\$00	\$00	data byte out
<b>Write Instructions</b>				
Write Program Memory Page	\$4C	adr MSB	adr LSB	\$00
Write EEPROM Memory	\$C0	0000 00aa / 0000 0aaa	aaaa aaaa	data byte in
Write EEPROM Memory Page (page access)	\$C2	0000 00aa / 0000 0aaa	aaaa aa00 / aaaa a000	\$00
Write Lock bits	\$AC	\$E0	\$00	data byte in
Write Fuse bits	\$AC	\$A0	\$00	data byte in
Write Fuse High bits	\$AC	\$A8	\$00	data byte in
Write Extended Fuse Bits	\$AC	\$A4	\$00	data byte in

- Notes:
1. Not all instructions are applicable for all parts
  2. a = address
  3. Bits are programmed '0', unprogrammed '1'.
  4. To ensure future compatibility, unused Fuses and Lock bits should be unprogrammed ('1') .
  5. Refer to the correspondig section for Fuse and Lock bits, Calibration and Signature bytes and Page size.
  6. See <http://www.atmel.com/avr> for Application Notes regarding programming and programmers.

If the LSB in RDY/BSY data byte out is '1', a programming operation is still pending. Wait until this bit returns '0' before the next instruction is carried out.

Within the same page, the low data byte must be loaded prior to the high data byte.

After data is loaded to the page buffer, program the EEPROM page, see [Figure 27-12](#).

8. Poll for Fuse write complete using programming instruction 6g, or wait for  $t_{WLRH}$  (refer to [Table 27-12 on page 279](#)).

#### **27.8.21 Programming the Lock Bits**

1. Enter JTAG instruction PROG\_COMMANDS.
2. Enable Lock bit write using programming instruction 7a.
3. Load data using programming instructions 7b. A bit value of “0” will program the corresponding lock bit, a “1” will leave the lock bit unchanged.
4. Write Lock bits using programming instruction 7c.
5. Poll for Lock bit write complete using programming instruction 7d, or wait for  $t_{WLRH}$  (refer to [Table 27-12 on page 279](#)).

#### **27.8.22 Reading the Fuses and Lock Bits**

1. Enter JTAG instruction PROG\_COMMANDS.
2. Enable Fuse/Lock bit read using programming instruction 8a.
3. To read all Fuses and Lock bits, use programming instruction 8e.  
To only read Fuse High byte, use programming instruction 8b.  
To only read Fuse Low byte, use programming instruction 8c.  
To only read Lock bits, use programming instruction 8d.

#### **27.8.23 Reading the Signature Bytes**

1. Enter JTAG instruction PROG\_COMMANDS.
2. Enable Signature byte read using programming instruction 9a.
3. Load address 0x00 using programming instruction 9b.
4. Read first signature byte using programming instruction 9c.
5. Repeat steps 3 and 4 with address 0x01 and address 0x02 to read the second and third signature bytes, respectively.

#### **27.8.24 Reading the Calibration Byte**

1. Enter JTAG instruction PROG\_COMMANDS.
2. Enable Calibration byte read using programming instruction 10a.
3. Load address 0x00 using programming instruction 10b.  
Read the calibration byte using programming instruction 10c.

## 31. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
<b>ARITHMETIC AND LOGIC INSTRUCTIONS</b>					
ADD	Rd, Rr	Add two Registers	$Rd \leftarrow Rd + Rr$	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	$Rd \leftarrow Rd + Rr + C$	Z,C,N,V,H	1
ADIW	Rdi,K	Add Immediate to Word	$Rdh:Rdl \leftarrow Rdh:Rdl + K$	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	$Rd \leftarrow Rd - Rr$	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	$Rd \leftarrow Rd - K$	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	$Rd \leftarrow Rd - K - C$	Z,C,N,V,H	1
SBIW	Rdi,K	Subtract Immediate from Word	$Rdh:Rdl \leftarrow Rdh:Rdl - K$	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	$Rd \leftarrow Rd \bullet Rr$	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	$Rd \leftarrow Rd \bullet K$	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	$Rd \leftarrow Rd \vee Rr$	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	$Rd \leftarrow Rd \vee K$	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	$Rd \leftarrow Rd \oplus Rr$	Z,N,V	1
COM	Rd	One's Complement	$Rd \leftarrow 0xFF - Rd$	Z,C,N,V	1
NEG	Rd	Two's Complement	$Rd \leftarrow 0x00 - Rd$	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	$Rd \leftarrow Rd \bullet (0xFF - K)$	Z,N,V	1
INC	Rd	Increment	$Rd \leftarrow Rd + 1$	Z,N,V	1
DEC	Rd	Decrement	$Rd \leftarrow Rd - 1$	Z,N,V	1
TST	Rd	Test for Zero or Minus	$Rd \leftarrow Rd \bullet Rd$	Z,N,V	1
CLR	Rd	Clear Register	$Rd \leftarrow Rd \oplus Rd$	Z,N,V	1
SER	Rd	Set Register	$Rd \leftarrow 0xFF$	None	1
MUL	Rd, Rr	Multiply Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
MULS	Rd, Rr	Multiply Signed	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	$R1:R0 \leftarrow Rd \times Rr$	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	$R1:R0 \leftarrow (Rd \times Rr) \ll 1$	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	$R1:R0 \leftarrow (Rd \times Rr) \ll 1$	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	$R1:R0 \leftarrow (Rd \times Rr) \ll 1$	Z,C	2
<b>BRANCH INSTRUCTIONS</b>					
RJMP	k	Relative Jump	$PC \leftarrow PC + k + 1$	None	2
JMP		Indirect Jump to (Z)	$PC \leftarrow Z$	None	2
JMP	k	Direct Jump	$PC \leftarrow k$	None	3
RCALL	k	Relative Subroutine Call	$PC \leftarrow PC + k + 1$	None	3
ICALL		Indirect Call to (Z)	$PC \leftarrow Z$	None	3
CALL	k	Direct Subroutine Call	$PC \leftarrow k$	None	4
RET		Subroutine Return	$PC \leftarrow STACK$	None	4
RETI		Interrupt Return	$PC \leftarrow STACK$	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if $(Rd = Rr)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
CP	Rd,Rr	Compare	$Rd - Rr$	Z, N, V, C, H	1
CPC	Rd,Rr	Compare with Carry	$Rd - Rr - C$	Z, N, V, C, H	1
CPI	Rd,K	Compare Register with Immediate	$Rd - K$	Z, N, V, C, H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if $(Rr(b)=0)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
SBRB	Rr, b	Skip if Bit in Register is Set	if $(Rr(b)=1)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if $(P(b)=0)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if $(P(b)=1)$ $PC \leftarrow PC + 2$ or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if $(SREG(s) = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if $(SREG(s) = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BREQ	k	Branch if Equal	if $(Z = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRNE	k	Branch if Not Equal	if $(Z = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRCS	k	Branch if Carry Set	if $(C = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRCC	k	Branch if Carry Cleared	if $(C = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRSH	k	Branch if Same or Higher	if $(C = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRLO	k	Branch if Lower	if $(C = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRMI	k	Branch if Minus	if $(N = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRPL	k	Branch if Plus	if $(N = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if $(N \oplus V = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if $(N \oplus V = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if $(H = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if $(H = 0)$ then $PC \leftarrow PC + k + 1$	None	1/2
BRTS	k	Branch if T Flag Set	if $(T = 1)$ then $PC \leftarrow PC + k + 1$	None	1/2

## 32.3 ATmega645

Speed (MHz) <sup>(3)</sup>	Power Supply	Ordering Code <sup>(2)</sup>	Package Type <sup>(1)</sup>	Operational Range
8	1.8 - 5.5V	ATmega645V-8AU ATmega645V-8AUR <sup>(4)</sup> ATmega645V-8MU ATmega645V-8MUR <sup>(4)</sup>	64A 64A 64M1 64M1	Industrial (-40°C to 85°C)
16	2.7 - 5.5V	ATmega645-16AU ATmega645-16AUR <sup>(4)</sup> ATmega645-16MU ATmega645-16MUR <sup>(4)</sup>	64A 64A 64M1 64M1	

- Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
  2. Pb-free packaging alternative, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
  3. For Speed Grades see [Figure 28-1 on page 299](#) and [Figure 28-2 on page 299](#).
  4. Tape & Reel

Package Type	
<b>64A</b>	64-lead, 14 x 14 x 1.0mm, Thin Profile Plastic Quad Flat Package (TQFP)
<b>64M1</b>	64-pad, 9 x 9 x 1.0mm, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

## 35. Datasheet Revision History

Please note that the referring page numbers in this section are referring to this document. The referring revision in this section are referring to the document revision.

### 35.1 Rev. 2570N – 05/11

1. Added Atmel QTouch Library Support and QTouch Sensing Capability Features.
2. Updated the last page with Atmel® trademarks and Microsoft Windows® trademarks.

### 35.2 Rev. 2570M – 04/11

1. Removed “Preliminary” from the front page
2. Removed “Disclaimer” section from the datasheet
3. Updated [Table 28-5 on page 301](#) “BODLEVEL Fuse Coding(1)”
4. Updated [“Ordering Information” on page 343](#) to include the “Tape & Reel” devices. Removed “AI” and “MI” devices.
5. Updated [“Errata” on page 350](#).
6. Updated the datasheet according to the Atmel new brand style guide, including the last page.

### 35.3 Rev. 2570L – 08/07

1. Updated [“Features” on page 1](#).
2. Added [“Data Retention” on page 9](#)
3. Updated [“Serial Programming Algorithm” on page 281](#).
4. Updated [“Speed Grades” on page 299](#).
5. Updated [“System and Reset Characteristics” on page 301](#).
6. Updated the Register Description at the end of each chapter.

### 35.4 Rev. 2570K – 04/07

1. Updated [“Errata” on page 350](#).

### 35.5 Rev. 2570J – 11/06

1. Updated [Table 28-7 on page 304](#).
2. Updated note in [Table 28-7 on page 304](#).