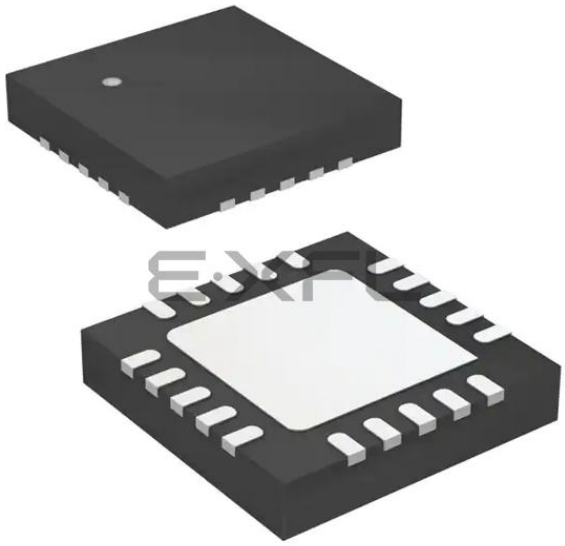


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	USI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	12
Program Memory Size	2KB (1K x 16)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	20-WFQFN Exposed Pad
Supplier Device Package	20-QFN-EP (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny24-15mz

Some of the status flags are cleared by writing a logical one to them. Note that, unlike most other AVR®s, the CBI and SBI instructions will only operate on the specified bit, and can, therefore, be used on registers containing such status flags. The CBI and SBI instructions work with registers 0x00 to 0x1F only.

The I/O and peripherals control registers are explained in later sections.

6.4.1 General Purpose I/O Registers

The ATtiny24/44/84 contains three general purpose I/O registers. These registers can be used for storing any information, and they are particularly useful for storing global variables and status flags. General purpose I/O registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI, CBI, SBIS, and SBIC instructions.

6.5 Register Description

6.5.1 EEARH – EEPROM Address Register

Bit	7	6	5	4	3	2	1	0	
0x1F (0x3F)	–	–	–	–	–	–	–	**EEAR8**	EEARH
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	X	

- **Bits 7..1 – Res: Reserved Bits**

These are reserved bits in the ATtiny24/44/84, and will always read as zero.

- **Bit 0 – EEAR8: EEPROM Address**

The EEPROM address register, EEARH, specifies the most-significant bit for EEPROM address in the 512-byte EEPROM space for Tiny84. This bit is reserved in the ATtiny24/44, and will always read as zero. The initial value of EEAR is undefined. A proper value must be written before the EEPROM may be accessed.

6.5.2 EEARL – EEPROM Address Register

Bit	7	6	5	4	3	2	1	0	
0x1E (0x3E)	**EEAR7**	**EEAR6**	**EEAR5**	**EEAR4**	**EEAR3**	**EEAR2**	**EEAR1**	**EEAR0**	EEARL
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	

- **Bits 7..0 – EEAR7..0: EEPROM Address**

The EEPROM address register, EEARL, specifies the EEPROM address. In the 128-byte EEPROM space in ATiny24, bit 7 is reserved and will always read as zero. The EEPROM data bytes are addressed linearly between 0 and 128/256/512. The initial value of EEAR is undefined. A proper value must be written before the EEPROM may be accessed.

6.5.3 EEDR – EEPROM Data Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	**EEDR7**	**EEDR6**	**EEDR5**	**EEDR4**	**EEDR3**	**EEDR2**	**EEDR1**	**EEDR0**	EEDR
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	

- **Bits 7..0 – EEDR7..0: EEPROM Data**

For the EEPROM write operation the EEDR register contains the data to be written to the EEPROM in the address given by the EEAR register. For the EEPROM read operation, the EEDR contains the data read out from the EEPROM at the address given by EEAR.

7.9 System Clock Prescaler

The Atmel® ATtiny24/44/84 system clock can be divided by setting the clock prescaler register – CLKPR. This feature can be used to decrease power consumption when the requirement for processing power is low. This can be used with all clock source options, and it will affect the clock frequency of the CPU and all synchronous peripherals. $clk_{I/O}$, clk_{ADC}, clk_{CPU}, and clk_{FLASH} are divided by a factor as shown in Table 7-10 on page 30.

7.9.1 Switching Time

When switching between prescaler settings, the system clock prescaler ensures that no glitches occur in the clock system and that no intermediate frequency is higher than either the clock frequency corresponding to the previous setting or the clock frequency corresponding to the new setting.

The ripple counter that implements the prescaler runs at the frequency of the undivided clock, which may be faster than the CPU's clock frequency. Hence, it is not possible to determine the state of the prescaler, even if it were readable, and the exact time it takes to switch from one clock division to another cannot be exactly predicted.

From the time the CLKPS values are written, it takes between T1 + T2 and T1 + 2 × T2 before the new clock frequency is active. In this interval, twp active clock edges are produced. Here, T1 is the previous clock period, and T2 is the period corresponding to the new prescaler setting.

7.10 Register Description

7.10.1 Oscillator Calibration Register – OSCCAL

Bit	7	6	5	4	3	2	1	0	
0x31 (0x51)	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	OSCCAL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	Device Specific Calibration Value								

The oscillator calibration register is used to trim the calibrated internal RC oscillator to remove process variations from the oscillator frequency. A pre-programmed calibration value is automatically written to this register during chip reset, giving the factory calibrated frequency as specified in Table 22-2 on page 157. The application software can write this register to change the oscillator frequency. The oscillator can be calibrated to frequencies as specified in Table 22-2 on page 157. calibration outside that range is not guaranteed.

Note that this oscillator is used to time EEPROM and flash write accesses, and these write times will be affected accordingly. If the EEPROM or flash are written, do not calibrate to more than 8.8MHz. Otherwise, the EEPROM or flash write may fail.

The CAL7 bit determines the range of operation for the oscillator. Setting this bit to logical zero gives the lowest frequency range, setting this bit to logical one gives the highest frequency range. The two frequency ranges are overlapping, in other words a setting of OSCCAL = 0x7F gives a higher frequency than OSCCAL = 0x80.

The CAL6..0 bits are used to tune the frequency within the selected range. A setting of 0x00 gives the lowest frequency in that range, and a setting of 0x7F gives the highest frequency in the range.

7.10.2 Clock Prescaler Register – CLKPR

Bit	7	6	5	4	3	2	1	0	
0x26 (0x46)	**CLKPCE**	–	–	–	**CLKPS3**	**CLKPS2**	**CLKPS1**	**CLKPS0**	**CLKPR**
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bit 7 – CLKPCE: Clock Prescaler Change Enable**

The CLKPCE bit must be written to logical one to enable change of the CLKPS bits. The CLK- PCE bit is only updated when the other bits in CLKPR are simultaneously written to logical zero. CLKPCE is cleared by hardware four cycles after it is written or when the CLKPS bits are written. Rewriting the CLKPCE bit within this time-out period does neither extend the time-out period nor clear the CLKPCE bit.+

9.3 Power-on Reset

A power-on reset (POR) pulse is generated by an on-chip detection circuit. The detection level is defined in Section 22.4 “System and Reset Characterizations” on page 158. The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the start-up reset, as well as to detect a failure in supply voltage.

A POR circuit ensures that the device is reset from power-on. Reaching the power-on reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after V_{CC} rise. The RESET signal is activated again, without any delay, when V_{CC} decreases below the detection level.

Figure 9-2. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}

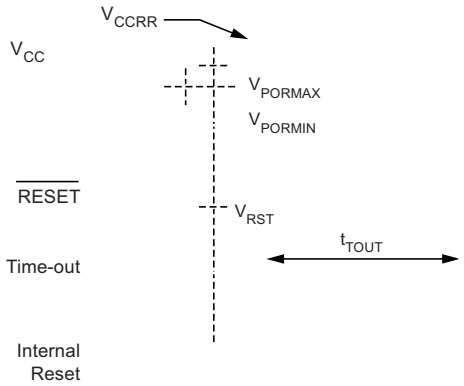


Figure 9-3. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally

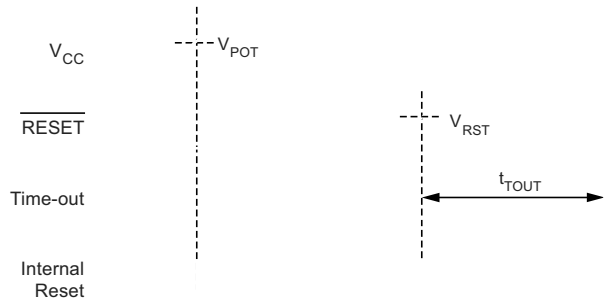


Table 9-1. Power On Reset Specifications

Parameter	Symbol	Min	Typ	Max	Unit
Power-on reset threshold voltage (rising)	V_{POT}	1.1	1.4	1.7	V
Power-on reset threshold voltage (falling)(1)		0.8	1.3	1.6	V
VCC max. start voltage to ensure internal power-on reset signal	V_{PORMAX}			0.4	V
VCC Min. start voltage to ensure internal power-on reset signal	V_{PORMIN}	-0.1			V
VCC rise rate to ensure power-on reset	V_{CCRR}	0.01			V/ms
$\overline{\text{RESET}}$ pin threshold voltage	V_{RST}	0.1 V_{CC}		0.9V_{CC}	V

Note: 1. Before rising, the supply has to be between V_{PORMIN} and V_{PORMAX} to ensure a reset.

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

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

13.5.1 Force Output Compare

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

13.5.2 Compare Match Blocking by TCNT0 Write

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

13.5.3 Using the Output Compare Unit

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

The setup of the OC0x should be performed before setting the data direction register for the port pin to output. The easiest way of setting the OC0x value is to use the force output compare (0x) strobe bits in normal mode. The OC0x registers keep their values even when changing between waveform generation modes.

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

13.6 Compare Match Output Unit

The compare output mode (COM0x1:0) bits have two functions. The waveform generator uses the COM0x1:0 bits for defining the output compare (OC0x) state at the next compare match. Also, the COM0x1:0 bits control the OC0x pin output source. Figure 13-4 on page 67 shows a simplified schematic of the logic affected by the COM0x1:0 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 COM0x1:0 bits are shown. When referring to the OC0x state, the reference is for the internal OC0x register, not the OC0x pin. If a system reset occurs, the OC0x register is reset to “0”.

13.9 Register Description

13.9.1 TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x30 (0x50)	COM0A1	COM0A**0**	COM0B**1**	COM0B**0**	–	–	**WGM01**	**WGM00**	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – COM0A1:0: Compare Match Output A Mode**

These bits control the output compare pin (OC0A) behavior. If one or both of the COM0A1:0 bits are set, the OC0A output overrides the normal port functionality of the I/O pin it is connected to. However, note that the data direction register (DDR) bit corresponding to the OC0A pin must be set in order to enable the output driver.

When OC0A is connected to the pin, the function of the COM0A1:0 bits depends on the WGM02:0 bit setting. Table 13-2 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 13-2. Compare Output Mode, non-PWM Mode

COM01	COM00	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on compare match
1	0	Clear OC0A on compare match
1	1	Set OC0A on compare match

Table 13-3 shows the COM0A1:0 bit functionality when the WGM01:0 bits are set to fast PWM mode.

Table 13-3. Compare Output Mode, Fast PWM Mode(1)

COM01	COM00	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal port operation, OC0A disconnected. WGM02 = 1: Toggle OC0A on compare match.
1	0	Clear OC0A on compare match, set OC0A at BOTTOM (non-inverting mode)
1	1	Set OC0A on compare match, clear OC0A at BOTTOM (inverting mode)

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the compare match is ignored, but the set or clear is done at BOTTOM. See Section 13.7.3 “Fast PWM Mode” on page 69 for more details.

Table 13-4 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 13-4. Compare Output Mode, Phase Correct PWM Mode(1)

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

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 13.7.4 “Phase Correct PWM Mode” on page 70 for more details.

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

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

When OC0B is connected to the pin, the function of the COM0B1:0 bits depends on the WGM02:0 bit setting. Table 13-2 on page 73 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 13-5. Compare Output Mode, non-PWM Mode

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

Table 13-3 on page 73 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to fast PWM mode.

Table 13-6. Compare Output Mode, Fast PWM Mode(1)

COM01	COM00	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Reserved
1	0	Clear OC0B on compare match, set OC0B at BOTTOM (non-inverting mode)
1	1	Set OC0B on compare match, clear OC0B at BOTTOM (inverting mode)

Note: 1. A special case occurs when OCR0B equals TOP and COM0B1 is set. In this case, the compare match is ignored, but the set or clear is done at BOTTOM. See Section 13.7.3 “Fast PWM Mode” on page 69 for more details.

Table 13-4 on page 73 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 13-7. Compare Output Mode, Phase Correct PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Reserved
1	0	Clear OC0B on compare match when up-counting. Set OC0B on compare match when down-counting.
1	1	Set OC0B on compare match when up-counting. Clear OC0B on compare match when down-counting.

Note: 1. A special case occurs when OCR0B equals TOP and COM0B1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 13.7.4 “Phase Correct PWM Mode” on page 70 for more details.

- **Bits 3, 2 – Res: Reserved Bits**

These bits are reserved bits in the ATtiny24/44/84 and will always read as zero.

- **Bits 1:0 – WGM01:0: Waveform Generation Mode**

Combined with the WGM02 bit found in the TCCR0B register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 13-8 on page 75. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), clear timer on compare match (CTC) mode, and two types of pulse width modulation (PWM) modes (see Section 13.7 “Modes of Operation” on page 67).

The following code examples show how to do an atomic write of the TCNT1 Register contents. Writing any of the OCR1A/B or ICR1 registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_WriteTCNT1:
       ; Save global interrupt flag
       in     r18,SREG
       ; Disable interrupts
       cli
       ; Set TCNT1 to r17:r16
       out    TCNT1H,r17
       out    TCNT1L,r16
       ; Restore global interrupt flag
       out    SREG,r18
       ret
```

C Code Example[1]

```
void TIM16_WriteTCNT1( unsigned int i )
{
       unsigned char sreg;
       unsigned int i;
       /* Save global interrupt flag */
       sreg = SREG;
       /* Disable interrupts */
       _CLI();
       /* Set TCNT1 to i */
       TCNT1 = i;
       /* Restore global interrupt flag */
       SREG = sreg;
}
```

Note: 1. See Section 4. “About Code Examples” on page 8.

The assembly code example requires that the r17:r16 register pair contains the value to be written to TCNT1.

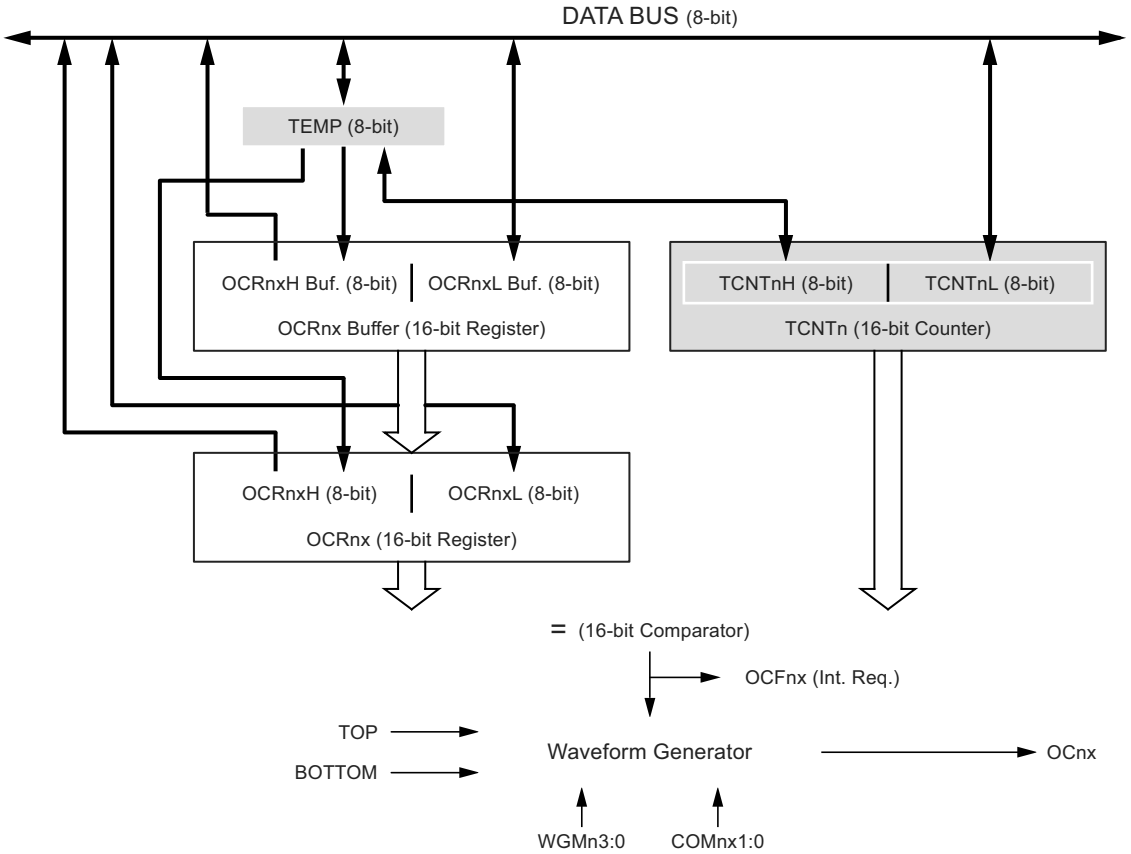
14.3.1 Reusing the Temporary High Byte Register

If when writing to more than one 16-bit register the high byte is the same for all registers writ- ten, then the high byte only needs to be written once. However, note that the same rule of atomic operation described previously also applies in this case.

14.4 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 (CS12:0) bits located in the Timer/Counter control register B (TCCR1B). For details on clock sources and prescaler, see Section 15. “Timer/Counter Prescaler” on page 103.

Figure 14-4. 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 prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR1x register access may seem complex, but this is not the case. When the double buffering is enabled, the CPU has access to the OCR1x buffer register, and if double buffering is disabled the CPU will access the OCR1x directly. The content of the OCR1x (buffer or compare) register is changed only by a write operation (the Timer/Counter does not update this register automatically as it does for the TCNT1 and ICR1 registers). Therefore, OCR1x is not read via the high byte temporary register (TEMP). However, it is a good practice to read the low byte first, as when accessing other 16-bit registers. Writing the OCR1x registers must be done via the TEMP register because the compare of all 16 bits is done continuously. The high byte (OCR1xH) has to be written first. When the high byte I/O location is written by the CPU, the TEMP register will be updated by the value written. Then when the low byte (OCR1xL) is written to the lower eight bits, the high byte will be copied into the upper 8-bits of either the OCR1x buffer or the OCR1x compare register in the same system clock cycle.

For more information of how to access the 16-bit registers, refer to Section 14.3 “Accessing 16-bit Registers” on page 80.

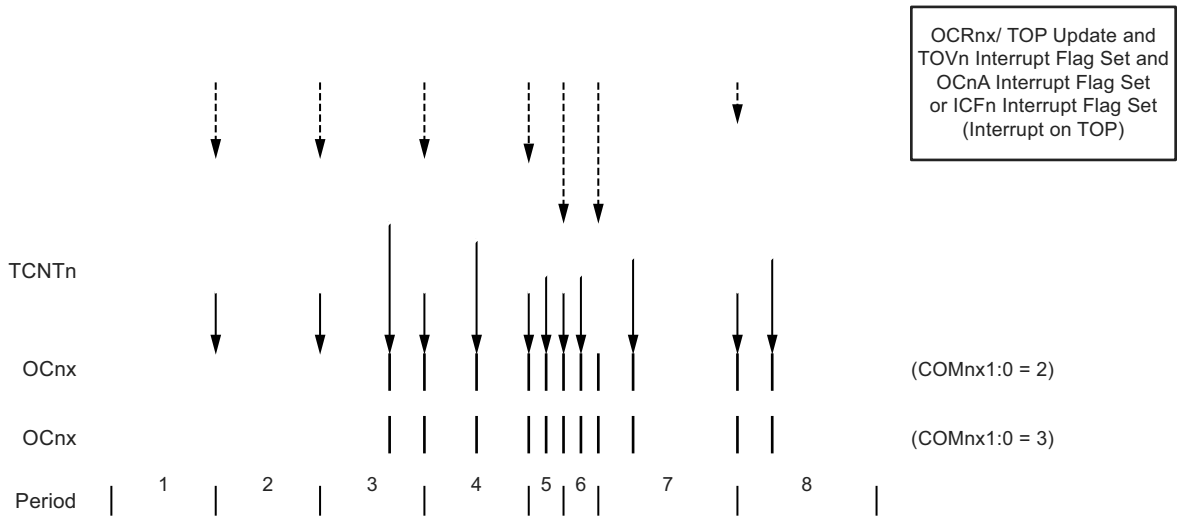
14.7.1 Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a logical one to the force output compare (1x) bit. Forcing compare match will not set the OCF1x flag or reload/clear the timer, but the OC1x pin will be updated as if a real compare match had occurred (the COM11:0 bit settings define whether the OC1x pin is set, cleared or toggled).

14.7.2 Compare Match Blocking by TCNT1 Write

All CPU writes to the TCNT1 register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR1x to be initialized to the same value as TCNT1 without triggering an interrupt when the Timer/Counter clock is enabled.

Figure 14-7. 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 on the same timer clock cycle on which 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 TCNT1 and 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 that for 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 no 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 when TCNT1 matches top. The update is done on the same timer clock cycle on which 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. How- ever, 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 two will produce a non-inverted PWM, and an inverted PWM out- put can be generated by setting the COM1x1:0 to three (see Table 14-3 on page 97). 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 on the timer clock cycle on which the counter is cleared (changes from top to bottom). The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk\_I/O}}{N \times (1 + TOP)}$$

The variable N represents the prescaler divider (1, 8, 64, 256, or 1024). The extreme values for the OCR1x register represent special cases when generating a PWM waveform output in the fast PWM mode. If the OCR1x is set equal to bottom (0x0000), the output will be a narrow spike for each top+1 timer clock cycle. Setting OCR1x equal to top will result in a constant high or low output (depending on the polarity of the output set by the COM1x1:0 bits.) A frequency waveform output (with 50% duty cycle) in fast PWM mode can be achieved by setting OC1A to toggle its logical level on each compare match (COM1A1:0 = 1). The waveform generated will have a maximum frequency of $_{1A} = f_{clk\_I/O}/2$ when OCR1A is set to zero (0x0000). This feature is similar to the OC1A toggle in CTC mode, except the double buffer feature of the output compare unit is enabled in the fast PWM mode.

14.11 Register Description

14.11.1 TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
0x2F (0x4F)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – COM1A1:0: Compare Output Mode for Channel A**

- **Bit 5:4 – COM1B1:0: Compare Output Mode for Channel B**

The COM1A1:0 and COM1B1:0 control the output compare pins' (OC1A and OC1B, respectively) behavior. If one or both of the COM1A1:0 bits are written to logical one, the OC1A output overrides the normal port functionality of the I/O pin it is connected to. If one or both of the COM1B1:0 bit are written to logical one, the OC1B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the data direction register (DDR) bit corresponding to the OC1A or OC1B pin must be set in order to enable the output driver.

When OC1A or OC1B is connected to the pin, the function of the COM1x1:0 bits is dependent on the WGM13:0 bit settings. Table 14-2 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to a normal or CTC mode (non-PWM).

Table 14-2. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on compare match.
1	0	Clear OC1A/OC1B on compare match (Set output to low level).
1	1	Set OC1A/OC1B on compare match (set output to high level).

Table 14-3 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

Table 14-3. Compare Output Mode, Fast PWM(1)

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13=0: Normal port operation, OC1A/OC1B disconnected. WGM13=1: Toggle OC1A on compare match, OC1B reserved.
1	0	Clear OC1A/OC1B on compare match, set OC1A/OC1B at BOTTOM (non-inverting mode)
1	1	Set OC1A/OC1B on compare match, clear OC1A/OC1B at BOTTOM (inverting mode)

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. In this case the compare match is ignored, but the set or clear is done at BOTTOM. See Section 14.9.3 “Fast PWM Mode” on page 90 for more details.

14.11.3 TCCR1C – Timer/Counter1 Control Register C

Bit	7	6	5	4	3	2	1	0	
0x22 (0x42)	FOC1A	FOC1B	–	–	–	–	–	–	TCCR1C
Read/Write	W	W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

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

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

The FOC1A/FOC1B bits are only active when the WGM13:0 bits specify a non-PWM mode. However, for ensuring compatibility with future devices, these bits must be set to zero when TCCR1A is written while 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 bit settings. 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.

A FOC1A/FOC1B strobe will not generate any interrupt, nor will it clear the timer in clear timer on compare match (CTC) mode using OCR1A as top. The FOC1A/FOC1B bits are always read as zero.

- **Bit 5..0 – Reserved Bit**

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

14.11.4 TCNT1H and TCNT1L – Timer/Counter1

Bit	7	6	5	4	3	2	1	0	
0x2D (0x4D)	TCNT1[15:8]								TCNT1H
0x2C (0x4C)	TCNT1[7:0]								TCNT1L
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 two Timer/Counter I/O locations (TCNT1H and TCNT1L, combined TCNT1) give direct access for both read and for write operations to the Timer/Counter unit's 16-bit counter. To ensure that both the high and low bytes are read and written simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary high byte register (TEMP). This temporary register is shared by all the other 16-bit registers. See Section 14.3 “Accessing 16-bit Registers” on page 80.

Modifying the counter (TCNT1) while the counter is running introduces a risk of missing a compare match between TCNT1 and one of the OCR1x registers.

Writing to the TCNT1 register blocks (removes) the compare match on the following timer clock for all compare units.

14.11.5 OCR1AH and OCR1AL – Output Compare Register 1 A

Bit	7	6	5	4	3	2	1	0	
0x2B (0x4B)	OCR1A[15:8]								OCR1AH
0x2A (0x4A)	OCR1A[7:0]								OCR1AL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 0 – TOIE1: Timer/Counter1, Overflow Interrupt Enable**

When this bit is written to logical one and the I-flag in the status register is set (interrupts globally enabled), the Timer/Counter 1 overflow interrupt is enabled. The corresponding interrupt vector (see Section 10. “Interrupts” on page 44) is executed when the TOV1 flag, located in TIFR1, is set.

14.11.9 TIFR1 – Timer/Counter Interrupt Flag Register 1

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	–	–	ICIF1	–	–	OCF1B	OCF1A	TOV1	**TIFR1**
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7,6,4,3 – Reserved Bit**

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

- **Bit 5– ICF1: Timer/Counter1, Input Capture Flag**

This flag is set when a capture event occurs on the ICP1 pin. When the input capture register (ICR1) is set by the WGM13:0 to be used as the top value, the ICF1 flag is set when the counter reaches the top value.

ICF1 is automatically cleared when the input capture interrupt vector is executed. Alternatively, ICF1 can be cleared by writing a logical one to its bit location.

- **Bit 2– OCF1B: Timer/Counter1, Output Compare B Match Flag**

This flag is set in the timer clock cycle after the counter (TCNT1) value matches the output compare register B (OCR1B).

Note that a forced output compare (1B) strobe will not set the OCF1B flag.

OCF1B is automatically cleared when the output compare match B interrupt vector is executed. Alternatively, OCF1B can be cleared by writing a logical one to its bit location.

- **Bit 1– OCF1A: Timer/Counter1, Output Compare A Match Flag**

This flag is set in the timer clock cycle after the counter (TCNT1) value matches the output compare register A (OCR1A).

Note that a forced output compare (1A) strobe will not set the OCF1A flag.

OCF1A is automatically cleared when the output compare match A interrupt vector is executed. Alternatively, OCF1A can be cleared by writing a logical one to its bit location.

- **Bit 0– TOV1: Timer/Counter1, Overflow Flag**

The setting of this flag is dependent of the WGM13:0 bit settings. In normal and CTC modes, the TOV1 flag is set when the timer overflows. See Table 14-5 on page 98 for the TOV1 flag behavior when using another WGM13:0 bit setting.

TOV1 is automatically cleared when the Timer/Counter 1 overflow interrupt vector is executed. Alternatively, TOV1 can be cleared by writing a logical one to its bit location.

```
        out    USICR,r17
        out    USICR,r16 ; LSB
        out    USICR,r17

        in     r16,USIDR
ret
```

16.3.3 SPI Slave Operation Example

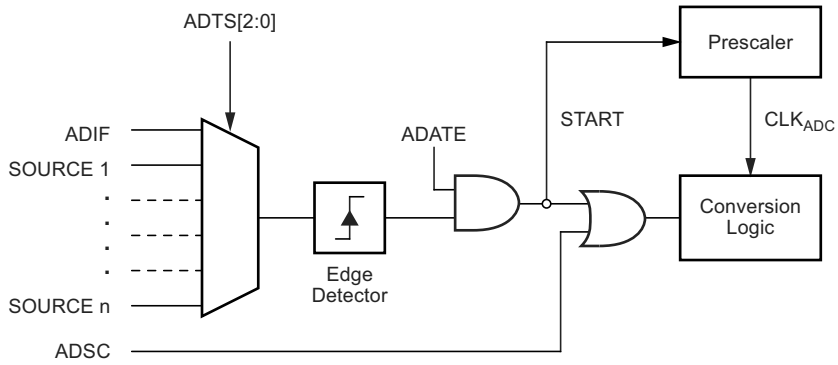
The following code demonstrates how to use the USI module as a SPI slave:

```
init:
        ldi    r16,(1<<USIWM0)|(1<<USICS1)
        out    USICR,r16
...
SlaveSPITransfer:
        out    USIDR,r16
        ldi    r16,(1<<USIOIF)
        out    USISR,r16
SlaveSPITransfer_loop:
        in     r16, USISR
        sbrs   r16, USIOIF
        rjmp   SlaveSPITransfer_loop
        in     r16,USIDR
        ret
```

The code is size optimized using only eight instructions (+ ret). The code example assumes that the DO is configured as output and USCK pin is configured as input in the DDR register. The value stored in register r16 prior to the function being called is transferred to the master device, and when the transfer is completed, the data received from the master is stored back into the r16 register.

Note that the first two instructions are for initialization only and need only to be executed once. These instructions set the three-wire mode and positive edge shift register clock. The loop is repeated until the USI counter overflow flag is set.

Figure 18-2. ADC Auto Trigger Logic

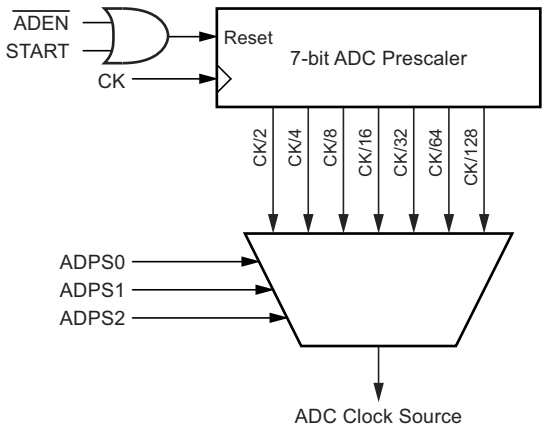


Using the ADC interrupt flag as a trigger source makes the ADC start a new conversion as soon as the ongoing conversion has finished. The ADC then operates in free running mode, constantly sampling and updating the ADC data register. The first conversion must be started by writing a logical one to the ADSC bit in ADCSRA. In this mode, the ADC will perform successive conversions independently of whether the ADC interrupt flag (ADIF) is cleared or not.

If auto triggering is enabled, single conversions can be started by writing ADSC in ADCSRA to logical one. ADSC can also be used to determine if a conversion is in progress. The ADSC bit will be read as logical one during a conversion independently of how the conversion was started.

18.5 Prescaling and Conversion Timing

Figure 18-3. ADC Prescaler



By default, the successive approximation circuitry requires an input clock frequency between 50kHz and 200kHz to get maximum resolution. If a lower resolution than 10 bits is needed, the input clock frequency to the ADC can be higher than 200kHz to get a higher sample rate.The ADC module contains a prescaler, which generates an acceptable ADC clock frequency from any CPU frequency above 100kHz.

The prescaling is set by the ADPS bits in ADCSRA. The prescaler starts counting from the moment the ADC is switched on by setting the ADEN bit in ADCSRA. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low.

When initiating a single-ended conversion by setting the ADSC bit in ADCSRA, the conversion starts at the following rising edge of the ADC clock cycle.

A normal conversion takes 13 ADC clock cycles. The first conversion after the ADC is switched on (ADEN in ADCSRA is set) takes 25 ADC clock cycles in order to initialize the analog circuitry.

The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of a normal conversion and 14.5 ADC clock cycles after the start of a first conversion. When a conversion is complete, the result is written to the ADC data registers, and ADIF is set. In single-conversion mode, ADSC is cleared simultaneously. The software may then set ADSC again, and a new conversion will be initiated on the first rising ADC clock edge.

- **Bit 4 – ADLAR: ADC Left Adjust Result**

The ADLAR bit affects the presentation of the ADC conversion result in the ADC data register. Write logical one to ADLAR to left adjust the result. Otherwise, the result is right adjusted. Changing the ADLAR bit will affect the ADC data register immediately, regardless of any ongoing conversions. For a complete description of this bit, see Section 18.10.3 “ADCL and ADCH – ADC Data Register” on page 133.

- **Bit 3 – Res: Reserved Bit**

This bit is reserved bit in the Atmel® ATtiny24/44/84, and will always read as what was written there.

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

If ADATE in ADCSRA is written to logical 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 18-7. 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

18.10.5 DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
0x01 (0x21)	ADC7D	ADC6D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
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	

- **Bits 7..0 – ADC7D..ADC0D: ADC7..0 Digital Input Disable**

When this bit is written logical 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 ADC7..0 pin and the digital input from this pin is not needed, this bit should be written logical one to reduce power consumption in the digital input buffer.

21.5 Page Size

Table 21-7. No. of Words in a Page and No. of Pages in the Flash

Device	Flash Size	Page Size	PCWORD	No. of Pages	PCPAGE	PCMSB
ATtiny24	1Kwords (2Kbytes)	16 words	PC[3:0]	64	PC[9:4]	9
ATtiny44	2Kwords (4Kbytes)	32 words	PC[4:0]	64	PC[10:5]	10
ATtiny84	4Kwords (8Kbytes)	32 words	PC[4:0]	128	PC[11:5]	11

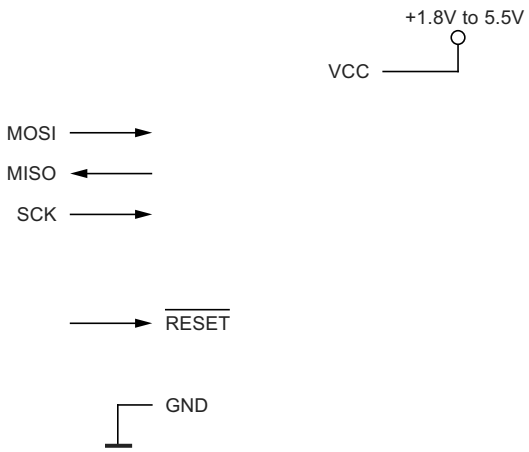
Table 21-8. No. of Words in a Page and No. of Pages in the EEPROM

Device	EEPROM Size	Page Size	PCWORD	No. of Pages	PCPAGE	EEAMSB
ATtiny24	128 bytes	4 bytes	EEA[1:0]	32	EEA[6:2]	6
ATtiny44	256 bytes	4 bytes	EEA[1:0]	64	EEA[7:2]	7
ATtiny84	512 bytes	4 bytes	EEA[1:0]	128	EEA[8:2]	8

21.6 Serial Downloading

Both the flash and EEPROM memory arrays can be programmed using the serial SPI bus while $\overline{\text{RESET}}$ is pulled to GND. The serial interface consists of pins SCK, MOSI (input) and MISO (output). After $\overline{\text{RESET}}$ is set low, the programming enable instruction needs to be executed first before program/erase operations can be executed. NOTE, in Table 21-9, the pin mapping for SPI programming is listed. Not all parts use the SPI pins dedicated for the internal SPI interface.

Figure 21-1. Serial Programming and Verify(1)



Note: 1. If the device is clocked by the internal oscillator, it is not needed to connect a clock source to the CLKI pin.

Table 21-9. Pin Mapping Serial Programming

Symbol	Pins	I/O	Description
MOSI	PA6	I	Serial data in
MISO	PA5	O	Serial data out
SCK	PA4	I	Serial clock

Table 22-7. ADC Characteristics, Differential Channels, T_A = –40°C to +125°C

Parameter	Condition	Symbol	Min	Typ	Max	Units
Resolution	Gain = 1x			8		Bits
	Gain = 20x			8		Bits
Absolute accuracy	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz	TUE		2.5	5.0	LSB
	Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz			3.0	6.0	LSB
Integral non-linearity (INL)	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz			0.5	2.5	LSB
	Bipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz	INL		0.5	3.0	LSB
	Unipolar – Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz			1.5	5.0	LSB
Differential non-linearity (DNL)	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz			0.4	1.0	LSB
	Bipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz	DNL		0.4	1.0	LSB
	Unipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz			0.7	2.0	LSB
Gain error	Bipolar – gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–5.0	+2.3	+5.0	LSB
	Unipolar – gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–5.0	–2.8	+5.0	LSB
	Bipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–7.0	+2.2	+7.0	LSB
	Unipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–7.0	–1.8	+7.0	LSB
Offset error	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–5.0	+2.0	+5.0	LSB
	Bipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–5.0	+2.0	+5.0	LSB
	Unipolar – gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 to 200kHz		–6.5	+2.0	+6.5	LSB
Clock frequency			50		200	kHz
Conversion time			65		260	µs

Table 22-7. ADC Characteristics, Differential Channels, T_A = –40°C to +125°C (Continued)

Parameter	Condition	Symbol	Min	Typ	Max	Units
Reference voltage		V_{REF}	2.56		AVCC – 0.5	V
Input voltage		V_{IN}	GND		AVCC	V
Input differential voltage		V_{DIFF}	$–V_{REF}$/gain		V_{REF}/gain	V

22.6 Serial Programming Characteristics

Figure 22-3. Serial Programming Timing

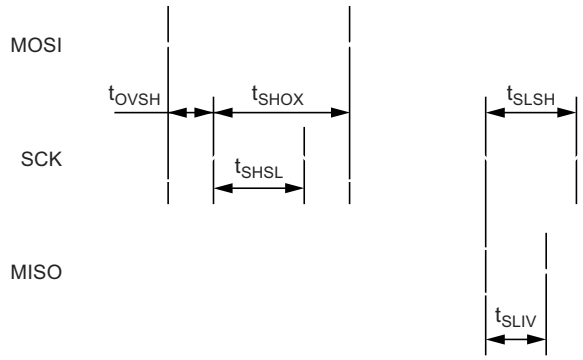


Figure 22-4. Serial Programming Waveforms

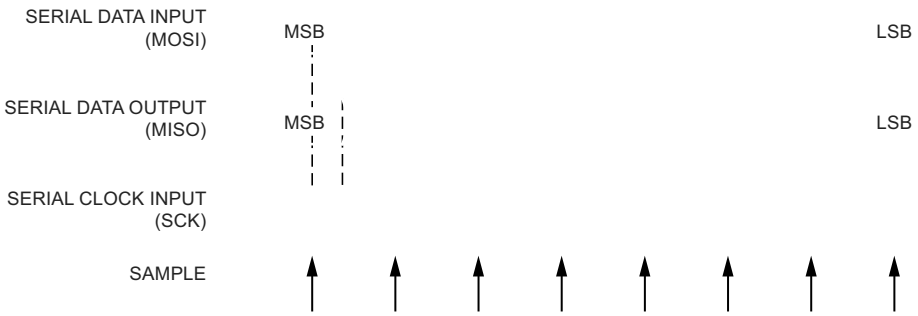


Table 22-8. Serial Programming Characteristics, T_A = –40°C to +125°C, V_{CC} = 2.7 to 5.5V (Unless Otherwise Noted)

Parameter	Symbol	Min	Typ	Max	Units
Oscillator frequency (Atmel® ATtiny24/44/84V)	$1/t_{CLCL}$	0		4	MHz
Oscillator period (Atmel ATtiny24/44/84V)	t_{CLCL}	250			ns
Oscillator frequency (ATtiny24/44/84, V_{CC} = 4.5V to 5.5V)	$1/t_{CLCL}$	0		20	MHz
Oscillator period (ATtiny24/44/84, V_{CC} = 4.5V to 5.5V)	t_{CLCL}	50			ns
SCK pulse width high	t_{SHSL}	2 t_{CLCL*}			ns
SCK pulse width low	t_{SLSH}	2 t_{CLCL*}			ns
MOSI setup to SCK high	t_{OVSH}	t_{CLCL}			ns
MOSI hold after SCK high	t_{SHOX}	2 t_{CLCL}			ns
SCK low to MISO valid	t_{SLIV}	TBD	TBD	TBD	ns

Note: 2 t_{CLCL} for f_{ck} < 12MHz, 3 t_{CLCL} for f_{ck} ≥ 12MHz

25. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
Arithmetic and Logic Instructions					
ADD	Rd, Rr	Add two registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with carry two registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl, K	Add immediate to word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two registers	Rd ← Rd – Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract constant from register	Rd ← Rd – K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with carry two registers	Rd ← Rd – Rr – C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with carry constant from Reg.	Rd ← Rd – K – C	Z,C,N,V,H	1
SBIW	Rdl, K	Subtract immediate from word	Rdh:Rdl ← Rdh:Rdl – K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND registers	Rd ← Rd × Rr	Z,N,V	1
ANDI	Rd, K	Logical AND register and constant	Rd ← Rd × K	Z,N,V	1
OR	Rd, Rr	Logical OR registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR register and constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One’s complement	Rd ← 0xFF – Rd	Z,C,N,V	1
NEG	Rd	Two’s complement	Rd ← 0x00 – Rd	Z,C,N,V,H	1
SBR	Rd, K	Set bit(s) in register	Rd ← Rd v K	Z,N,V	1
CBR	Rd, K	Clear bit(s) in register	Rd ← Rd × (0xFF – K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V	1
TST	Rd	Test for zero or minus	Rd ← Rd × Rd	Z,N,V	1
CLR	Rd	Clear register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set register	Rd ← 0xFF	None	1
Branch Instructions					
RJMP	k	Relative jump	PC ← PC + k + 1	None	2
IJMP		Indirect jump to (Z)	PC ← Z	None	2
RCALL	k	Relative subroutine call	PC ← PC + k + 1	None	3
ICALL		Indirect call to (Z)	PC ← Z	None	3
RET		Subroutine return	PC ← STACK	None	4
RETI		Interrupt return	PC ← STACK	I	4
CPSE	Rd, Rr	Compare, skip if equal	if (Rd = Rr) PC ← 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 ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if bit in register is set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if bit in I/O register cleared	if (P(b)=0) PC ← 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 ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if status flag set	if (SREG (s) = 1) then PC ← PC + k + 1	None	1/2
BRBC	s, k	Branch if status flag cleared	if (SREG(s) = 0) then PC ← PC + k + 1	None	1/2