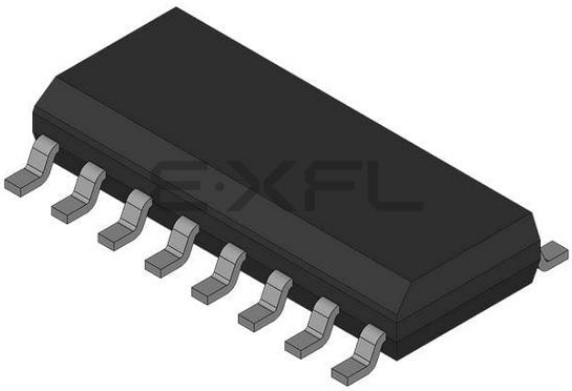


Welcome to E-XFL.COM

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.

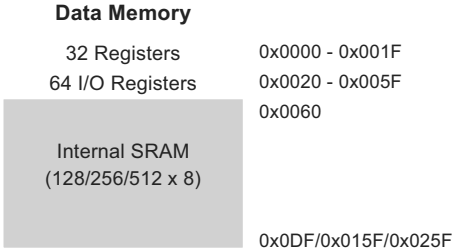
Applications of "Embedded - Microcontrollers"

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	4KB (2K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 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	14-SOIC (0.154", 3.90mm Width)
Supplier Device Package	14-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/atmel/attiny44-15ssz

2. Overview

The Atmel® ATtiny24/44/84 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 ATtiny24/44/84 achieves throughputs approaching 1MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

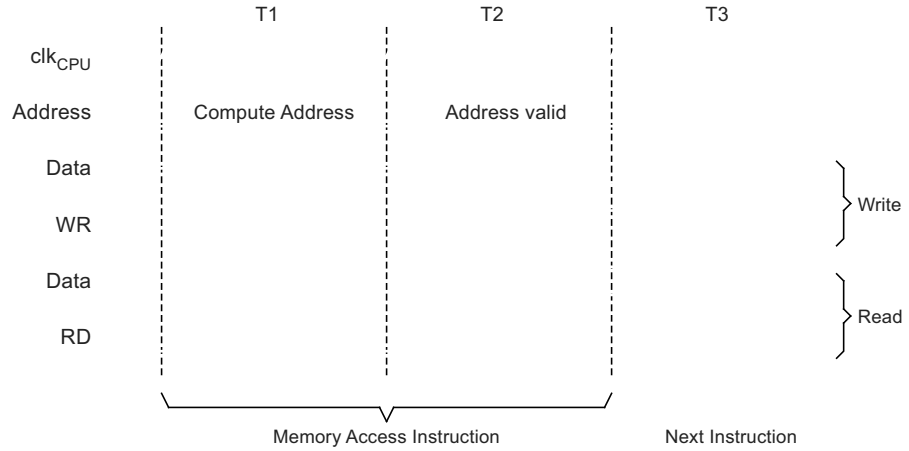
Figure 6-2. Data Memory Map



6.2.1 Data Memory Access Times

This section describes the general access timing concepts for internal memory access. The internal data SRAM access is performed in two clk_{CPU} cycles as described in Figure 6-3.

Figure 6-3. On-chip Data SRAM Access Cycles



6.3 EEPROM Data Memory

The Atmel® ATtiny24/44/84 contains 128/256/512 bytes of EEPROM data memory. It is organized as a separate data space in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following sections specifying the EEPROM address registers, the EEPROM data register, and the EEPROM control register. For a detailed description of serial data downloading to the EEPROM, see Section 21.6 “Serial Downloading” on page 144.

6.3.1 EEPROM Read/Write Access

The EEPROM access registers are accessible in the I/O space.

The write access times for the EEPROM are given in Table 6-1 on page 22. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains instructions that write to the EEPROM, some precautions must be taken. In heavily filtered power supplies, VCC is likely to rise or fall slowly on power-up/down. This causes the device for some period of time to run at a voltage lower than the specified minimum for the clock frequency used. See Section 6.3.6 “Preventing EEPROM Corruption” on page 20 for details on how to avoid problems in these situations.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. See Section 6.3.2 “Atomic Byte Programming” on page 18 and Section 6.3.3 “Split Byte Programming” on page 18 for details on this.

When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed.

The next code examples show assembly and C functions for reading the EEPROM. The examples assume that interrupts are controlled so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
EEPROM_read:
        ; Wait for completion of previous write
        sbic    EECR,EEPE
        rjmp    EEPROM_read
        ; Set up address (r17) in address register
        out     EEARL, r17
        ; Start eeprom read by writing EERE
        sbi     EECR,EERE
        ; Read data from data register
        in      r16,EEDR
        ret
```

C Code Example

```
unsigned char EEPROM_read(unsigned char ucAddress)
{
        /* Wait for completion of previous write */
        while(EECR & (1<<EEPE))
        ;
        /* Set up address register */
        EEARL = ucAddress;
        /* Start eeprom read by writing EERE */
        EECR |= (1<<EERE);
        /* Return data from data register */
        return EEDR;
}
```

Note: The code examples are only valid for Atmel® ATtiny24 and Atmel ATtiny44, using 8-bit addressing mode.

6.3.6 Preventing EEPROM Corruption

During periods of low VCC, the EEPROM data can be corrupted because the supply voltage is too low for the CPU and the EEPROM to operate properly. A regular write sequence to the EEPROM requires a minimum voltage to operate correctly, and the CPU itself can execute instructions incorrectly if the supply voltage is too low. These issues are the same as for board level systems using EEPROM, and the same design solutions should be applied.

EEPROM data corruption can easily be avoided by following this design recommendation: Keep the AVR® RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal brown-out detector (BOD). If the detection level of the internal BOD does not match the needed detection level, an external low $–V_{CC}$ reset protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided the power supply voltage is sufficient.

6.4 I/O Memory

The I/O space definition of the Atmel ATtiny24/44/84 is shown in Section 23-43 “Minimum Reset Pulse Width versus V_{CC}” on page 181.

All Atmel ATtiny24/44/84 I/Os and peripherals are placed in the I/O space. All I/O locations may be accessed by the LD/LDS/LDD and ST/STS/STD instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. See the instruction set summary for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O registers as data space using LD and ST instructions, 0x20 must be added to these addresses.

For compatibility with future devices, reserved bits should be written with a logical zero if accessed. Reserved I/O memory addresses should never be written.

6.5.4 EECR – EEPROM Control Register

Bit	7	6	5	4	3	2	1	0	
0x1C (0x3C)	–	–	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE	EECR
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	X	X	0	0	X	0	

- **Bit 7 – Res: Reserved Bit**

This bit is reserved for future use, and will always read as 0 in Atmel® ATtiny24/44/84. For compatibility with future AVR® devices, always write this bit to a logical zero. After reading, mask out this bit.

- **Bit 6 – Res: Reserved Bit**

This bit is reserved in the Atmel ATtiny24/44/84 and will always read as zero.

- **Bits 5, 4 – EEPM1 and EEPM0: EEPROM Mode Bits**

The EEPROM programming mode bits define which programming action will be triggered when writing EEPE. It is possible to program data in one atomic operation (erase the old value and program the new value) or split the erase and write operations into two separate operations. The programming times for the different modes are shown in Table 6-1. While EEPE is set, any write to EEPMn will be ignored. During reset, the EEPMn bits will be reset to 0b00 unless the EEPROM is busy programming.

Table 6-1. EEPROM Mode Bits

EEPM1	EEPM0	Programming Time	Operation
0	0	3.4ms	Erase and write in one operation (atomic operation)
0	1	1.8ms	Erase only
1	0	1.8ms	Write only
1	1	–	Reserved for future use

- **Bit 3 – EERIE: EEPROM Ready Interrupt Enable**

Writing EERIE to logical one enables the EEPROM ready interrupt if the I-bit in SREG is set. Writing EERIE to logical zero disables the interrupt. The EEPROM ready interrupt generates a constant interrupt when non-volatile memory is ready for programming.

- **Bit 2 – EEMPE: EEPROM Master Program Enable**

The EEMPE bit determines whether writing EEPE to logical one will have effect or not. When EEMPE is set, setting EEPE within four clock cycles will program the EEPROM at the selected address. If EEMPE is logical zero, setting EEPE will have no effect. When EEMPE has been written to logical one by software, hardware clears the bit to logical zero after four clock cycles.

- **Bit 1 – EEPE: EEPROM Program Enable**

The EEPROM program enable bit, EEPE, is the programming enable signal to the EEPROM. When EEPE is written, the EEPROM will be programmed according to the EEPMn bits setting. The EEMPE bit must be written to logical one before a logical one is written to EEPE, otherwise no EEPROM write will take place. When the write access time has elapsed, the EEPE bit is cleared by hardware. When EEPE has been set, the CPU is halted for two cycles before the next instruction is executed.

- **Bit 0 – EERE: EEPROM Read Enable**

The EEPROM read enable signal, EERE, is the read strobe for the EEPROM. When the correct address is set up in the EEAR register, the EERE bit must be written to logical one to trigger the EEPROM read. The EEPROM read access takes one instruction, and the requested data are available immediately. When the EEPROM is read, the CPU is halted for four cycles before the next instruction is executed. The user should poll the EEPE bit before starting the read operation. If a write operation is in progress, it is not possible to read the EEPROM or change the EEAR register.

9.10 Register Description

9.10.1 MCUSR – MCU Status Register

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

Bit	7	6	5	4	3	2	1	0	
0x34 (0x54)	–	–	–	–	WDRF	BORF	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bits 7..4 – Res: Reserved Bits**

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

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

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

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

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

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

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

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

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

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

9.10.2 WDTCSR – Watchdog Timer Control and Status Register

Bit	7	6	5	4	3	2	1	0	
0x21 (0x41)	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	WDTCSR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	X	0	0	0	

- **Bit 7 – WDIF: Watchdog Time-out Interrupt Flag**

This bit is set when a time-out occurs in the watchdog timer and the watchdog timer is configured for interrupt. WDIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, WDIF is cleared by writing a logic one to the flag. When the I-bit in SREG and WDIE are set, the watchdog time-out interrupt is executed.

- **Bit 6 – WDIE: Watchdog Timeout Interrupt Enable**

When this bit is written to logical one, WDE is cleared, and the I-bit in the status register is set, the watchdog time-out interrupt is enabled. In this mode the corresponding interrupt is executed instead of a reset if a time-out in the watchdog timer occurs.

If WDE is set, WDIE is automatically cleared by hardware when a time-out occurs. This is useful for keeping the watchdog reset security while using the interrupt. After the WDIE bit is cleared, the next time-out will generate a reset. To avoid the watchdog reset, WDIE must be set after each interrupt.

10. Interrupts

This section describes the specifics of the interrupt handling as performed in Atmel® ATtiny24/44/84. For a general explanation of the AVR® interrupt handling, see Section 5.8 “Reset and Interrupt Handling” on page 14.

10.1 Interrupt Vectors

Table 10-1. Reset and Interrupt Vectors

Vector No.	Program Address	Source	Interrupt Definition
1	0x0000	RESET	External pin, power-on reset, brown-out reset, watchdog reset
2	0x0001	INT0	External interrupt request 0
3	0x0002	PCINT0	Pin change interrupt request 0
4	0x0003	PCINT1	Pin change interrupt request 1
5	0x0004	WDT	Watchdog time-out
6	0x0005	TIMER1 CAPT	Timer/Counter1 capture event
7	0x0006	TIMER1 COMPA	Timer/Counter1 compare match A
8	0x0007	TIMER1 COMPB	Timer/Counter1 compare match B
9	0x0008	TIMER1 OVF	Timer/Counter0 overflow
10	0x0009	TIMER0 COMPA	Timer/Counter0 compare match A
11	0x000A	TIMER0 COMPB	Timer/Counter0 compare match B
12	0x000B	TIMER0 OVF	Timer/Counter0 overflow
13	0x000C	ANA_COMP	Analog comparator
14	0x000D	ADC	ADC conversion complete
15	0x000E	EE_RDY	EEPROM ready
16	0x000F	USI_START	USI START
17	0x0010	USI_OVF	USI overflow

13. 8-bit Timer/Counter0 with PWM

13.1 Features

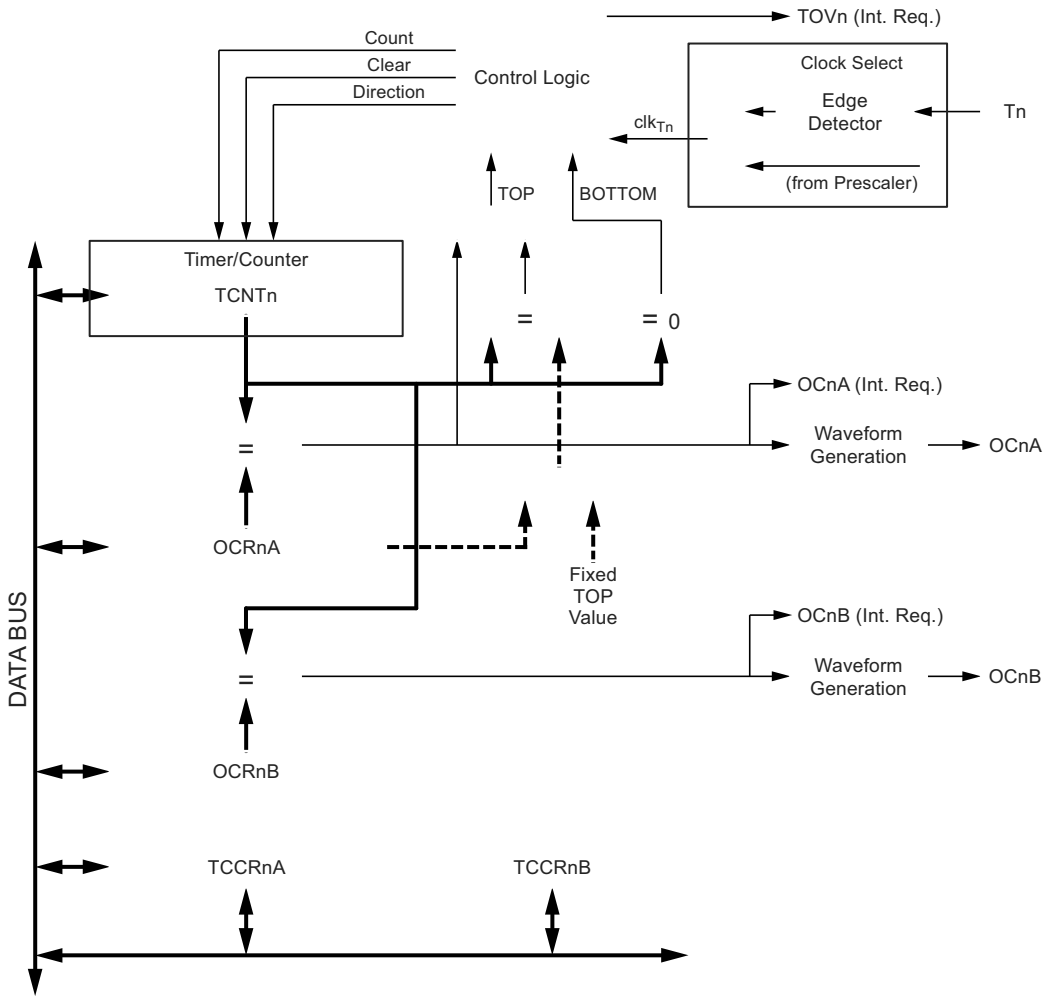
- Two independent output compare units
- Double buffered output compare registers
- Clear timer on compare match (auto reload)
- Glitch free, phase correct pulse width modulator (PWM)
- Variable PWM period
- Frequency generator
- Three independent interrupt sources (TOV0, OCF0A, and OCF0B)

13.2 Overview

Timer/Counter 0 is a general purpose 8-bit Timer/Counter module, with two independent out- put compare units, and with PWM support. It allows accurate program execution timing (event management) and wave generation.

A simplified block diagram of the 8-bit Timer/Counter is shown in Figure 13-1. For the actual placement of I/O pins, refer to Figure 1-1 on page 3. CPU accessible I/O registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O register and bit locations are listed in the Section 13.9 “Register Description” on page 73.

Figure 13-1. 8-bit Timer/Counter Block Diagram



The extreme values for the OCR0A register represent special cases when generating a PWM waveform output in the fast PWM mode. If the OCR0A is set equal to bottom, the out- put will be a narrow spike for each max+1 timer clock cycle. Setting the OCR0A equal to max will result in a constantly high or low output (depending on the polarity of the output set by the COM0A1:0 bits).

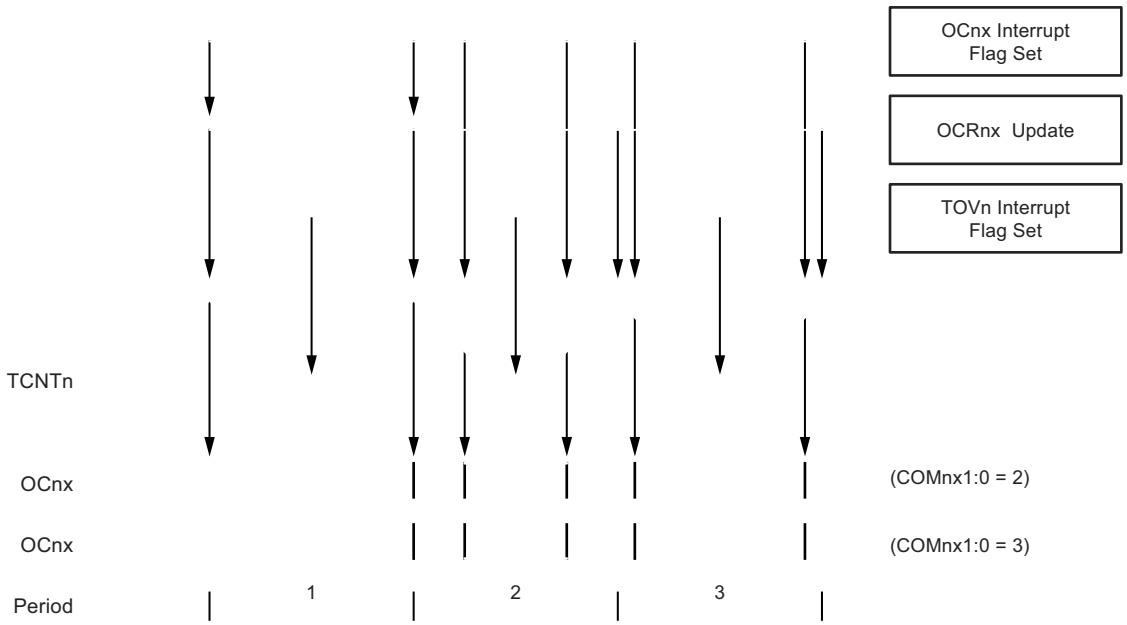
A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC0x to toggle its logical level on each compare match (COM0x1:0 = 1). The waveform generated will have a maximum frequency of $f_0 = f_{clk\_I/O}/2$ when OCR0A is set to zero. This feature is similar to the OC0A toggle in CTC mode, except the double buffer feature of the output compare unit is enabled in the fast PWM mode.

13.7.4 Phase Correct PWM Mode

The phase correct PWM mode (WGM02:0 = 1 or 5) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is based on a dual-slope operation. The counter counts repeatedly from bottom to top and then from top to bottom. Top is defined as 0xFF when WGM2:0 = 1, and as OCR0A when WGM2:0 = 5. In non-inverting compare output mode, the output compare (OC0x) is cleared on the compare match between TCNT0 and OCR0x while up-counting, and set on the compare match while down-counting. In inverting output compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

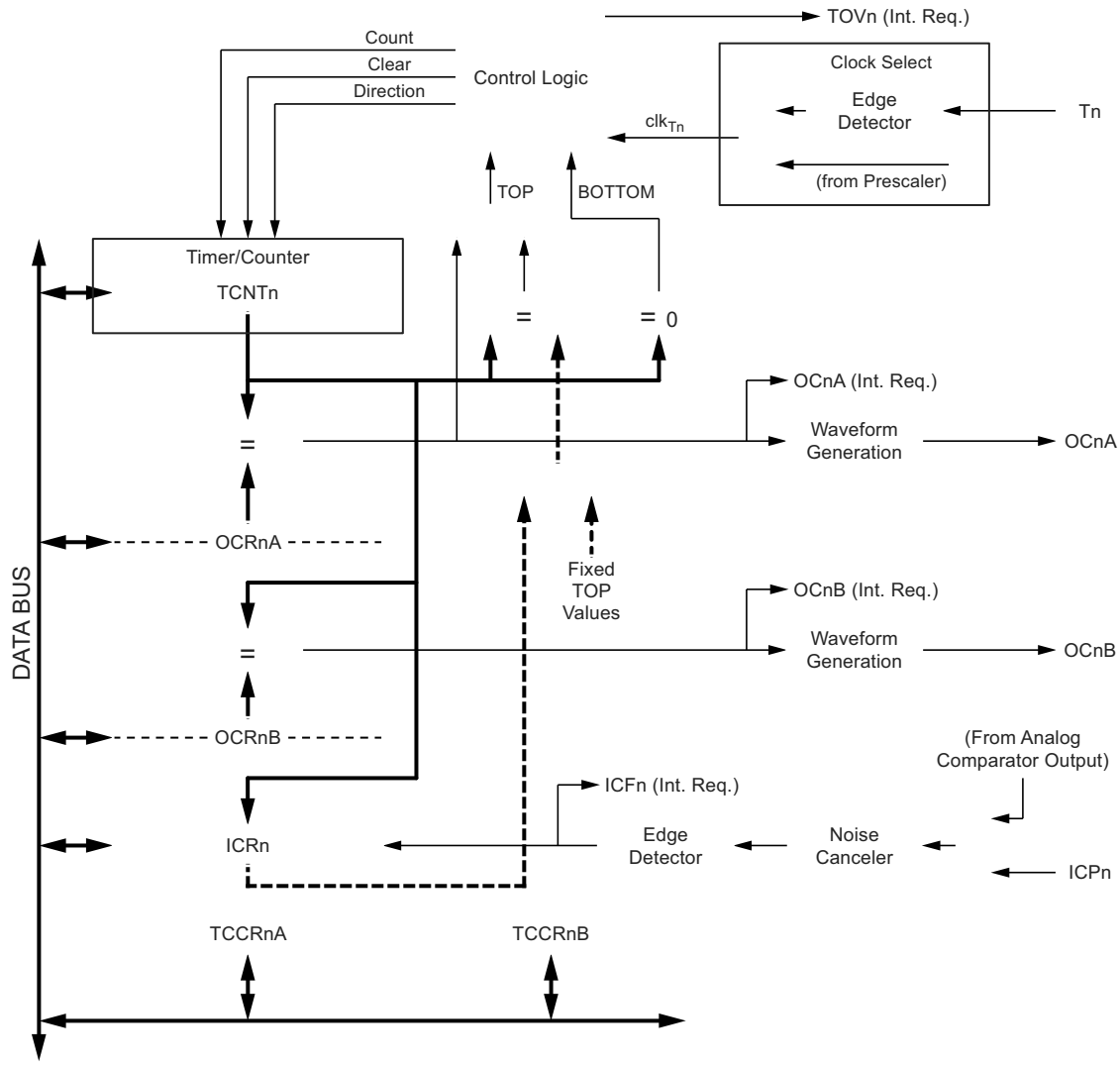
In phase correct PWM mode the counter is incremented until the counter value matches TOP. When the counter reaches TOP, it changes the count direction. The TCNT0 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 13-7. The TCNT0 value is in the timing diagram, which is shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal lines on the TCNT0 slopes represent compare matches between OCR0x and TCNT0.

Figure 13-7. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter overflow flag (TOV0) is set each time the counter reaches bottom. The interrupt flag can be used to generate an interrupt each time the counter reaches the bottom value.

Figure 14-1. 16-bit Timer/Counter Block Diagram(1)



Note: 1. See Figure 1-1 on page 3 for Timer/Counter1 pin placement and description.

14.2.1 Registers

The Timer/Counter (TCNT1), output compare registers (OCR1A/B), and input capture register (ICR1) are all 16-bit registers. Special procedures must be followed when accessing the 16-bit registers. These procedures are described in the Section 14.3 “Accessing 16-bit Registers” on page 80. The Timer/Counter control registers (TCCR1A/B) are 8-bit registers, and have no CPU access restrictions. Interrupt request (abbreviated to Int.Req. in the figure) signals are all visible in the timer interrupt flag register (TIFR). All interrupts are individually masked with the timer interrupt mask register (TIMSK). TIFR and TIMSK are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T1 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_{T1}).

The double buffered output compare registers (OCR1A/B) are 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 (OC1A/B). See Section 14.7 “Output Compare Units” on page 86. The compare match event will also set the compare match flag (OCF1A/B) which can be used to generate an output compare interrupt request.

The following code examples show how to access the 16-bit timer registers, assuming that no interrupts updates the temporary register. The same principle can be used directly for accessing the OCR1A/B and ICR1 registers. Note that when using C, the compiler handles the 16-bit access.

Assembly Code Examples[1]

```
...
; Set TCNT1 to 0x01FF
ldi    r17,0x01
ldi    r16,0xFF
out    TCNT1H,r17
out    TCNT1L,r16
; Read TCNT1 into r17:r16
in     r16,TCNT1L
in     r17,TCNT1H
...
```

C Code Examples[1]

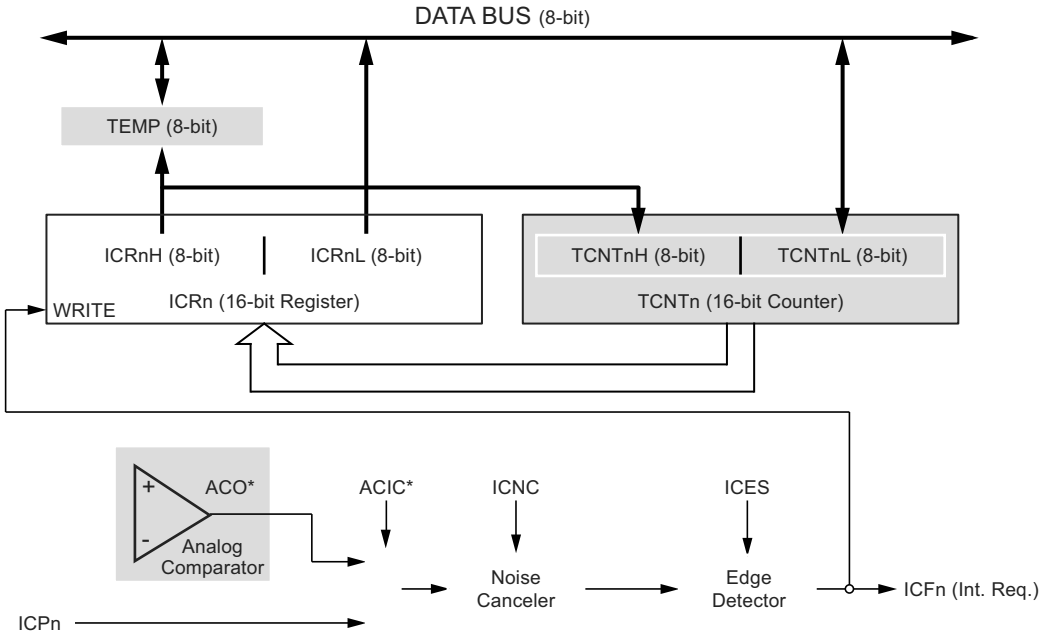
```
unsigned int i;
...
/* Set TCNT1 to 0x01FF */
TCNT1 = 0x1FF;
/* Read TCNT1 into i */
i = TCNT1;
...
```

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

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register and the interrupt code updates the temporary register by accessing the same or any of the other 16-bit timer registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the temporary register, the main code must disable the interrupts during the 16-bit access.

Figure 14-3. Input Capture Unit Block Diagram



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

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

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

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

14.6.1 Input Capture Trigger Source

The main trigger source for the input capture unit is the input capture pin (ICP1). Timer/Counter 1 can alternatively use the analog comparator output as trigger source for the input capture unit. The analog comparator is selected as trigger source by setting the analog comparator input capture (ACIC) bit in the analog comparator control and status register (ACSR). Be aware that changing the trigger source can trigger a capture. The input capture flag must, therefore, be cleared after the change.

Both the input capture pin (ICP1) and the analog comparator output (ACO) are sampled using the same technique as for the T1 pin (Figure 15-1 on page 103). The edge detector is also identical. However, when the noise canceller is enabled, additional logic is inserted before the edge detector, which increases the delay by four system clock cycles. Note that the inputs of the noise canceller and edge detector are always enabled unless the Timer/Counter is set in a waveform generation mode that uses ICR1 to define top.

An input capture can be triggered by software by controlling the port of the ICP1 pin.

Figure 14-12 shows the count sequence close to TOP in various modes. When using phase and frequency correct PWM mode the OCR1x register is updated at BOTTOM. The timing diagrams will be the same, but TOP should be replaced by BOTTOM, TOP-1 by BOTTOM+1 and so on. The same renaming applies for modes that set the TOV1 flag at BOTTOM.

Figure 14-12. Timer/Counter Timing Diagram, no Prescaling

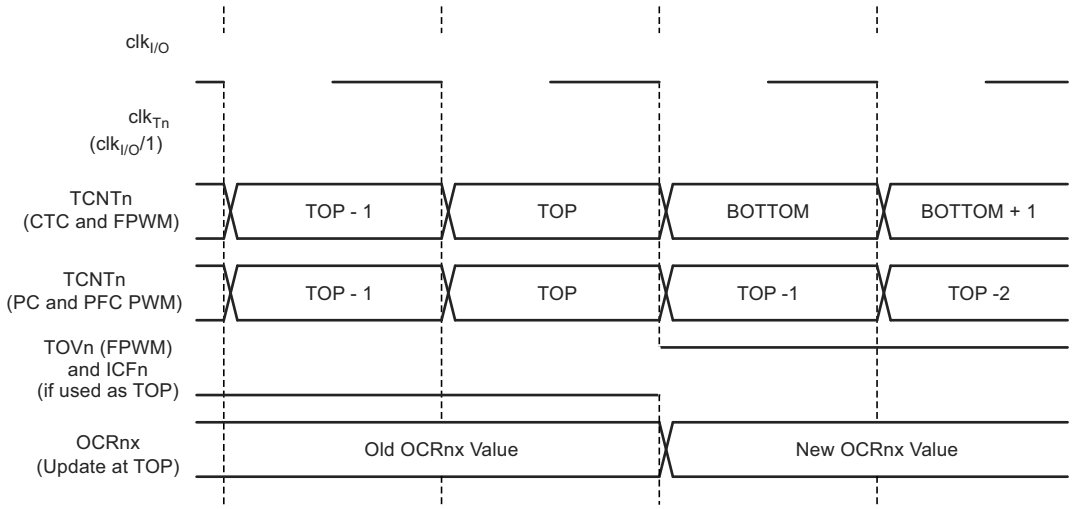
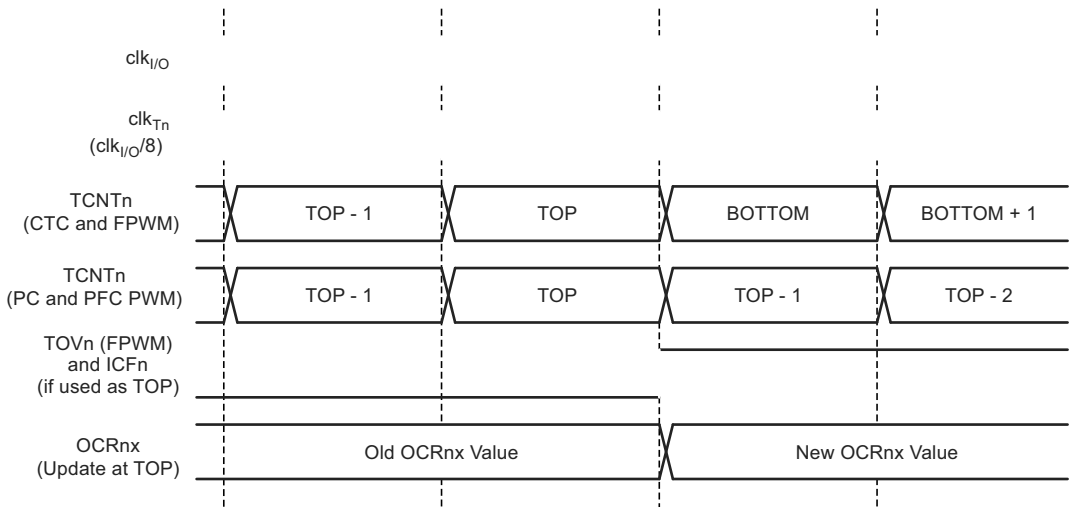


Figure 14-13 shows the same timing data, but with the prescaler enabled.

Figure 14-13. Timer/Counter Timing Diagram, with Prescaler ($f_{clk\_I/O}/8$)



- **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.

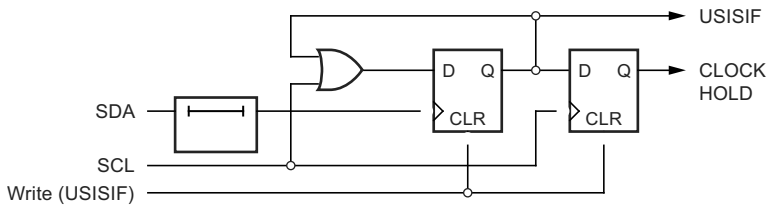
TOV1 is automatically cleared when the Timer/Counter 1 overflow interrupt vector is executed. Alternatively, TOV1 can be cleared by writing a logical one to its bit location.

Referring to the timing diagram (Figure 16-5 on page 109), a bus transfer involves the following steps:

1. The start condition is generated by the master by forcing the SDA line low while the SCL line is high (A). SDA can be forced low either by writing a logical zero to bit 7 of the shift register, or by setting the corresponding bit in the PORT register to zero. Note that the data direction register bit must be set to one for the output to be enabled. The slave device's start detector logic (Figure 16-6) detects the start condition and sets the USISIF flag. The flag can generate an interrupt if necessary.
2. In addition, the start detector will hold the SCL line low after the master has forced a negative edge on this line (B). This allows the slave to wake up from sleep or complete its other tasks before setting up the shift register to receive the address. This is done by clearing the start condition flag and resetting the counter.
3. The master set the first bit to be transferred and releases the SCL line (C). The Slave samples the data and shift it into the serial register at the positive edge of the SCL clock.
4. After eight bits containing the slave address and data direction (read or write) are transferred, the slave counter overflows and the SCL line is forced low (D). If the slave is not the one the master has addressed, it releases the SCL line and waits for a new start condition.
5. If the slave is addressed, it holds the SDA line low during the acknowledgment cycle before holding the SCL line low again (i.e., the counter register must be set to 14 before releasing SCL at (D)). Depending on the state of the R/W bit, the master or slave enables its output. If the bit is set, a master read operation is in progress (i.e., the slave drives the SDA line). The slave can hold the SCL line low after the acknowledgement cycle (E).
6. Multiple bytes can now be transmitted, all in same direction, until a stop condition is given by the master (F), or a new start condition is given.

If the slave is not able to receive more data, it does not acknowledge the data byte it has last received. When the master does a read operation, it must terminate the operation by forcing the acknowledge bit low after the last byte is transmitted.

Figure 16-6. Start Condition Detector, Logic Diagram



16.3.5 Start Condition Detector

The start condition detector is shown in Figure 16-6. The SDA line is delayed (in the range of 50 to 300ns) to ensure valid sampling of the SCL line. The start condition detector is only enabled in two-wire mode.

The start condition detector is working asynchronously, and can, therefore, wake up the processor from the power-down sleep mode. However, the protocol used might have restrictions on the SCL hold time. Therefore, when using this feature in this case, the oscillator start-up time set by the CKSEL fuses (see Section 7.1 “Clock Systems and their Distribution” on page 24) must also be taken into consideration. See the USISIF bit description in Section 16.5.3 “USISR – USI Status Register” on page 112 for further details.

16.3.6 Clock speed considerations

Maximum frequency for SCL and SCK is f_{CK} /4. This is also the maximum data transmit and receive rate in both two- wire and three-wire mode. In two-wire slave mode the two-wire clock control unit will hold SCL low until the slave is ready to receive more data. This may reduce the actual data rate in two-wire mode.

Table 18-2. Temperature versus Sensor Output Voltage (Typical Case)

Temperature/°C	–40°C	+25°C	+85°C	+125°C
Voltage/mV	243mV	314mv	380mV	424mV

The values described in Table 18-2 are typical values. However, due to the process variation, the temperature sensor output voltage varies from one chip to another. To be capable of achieving more accurate results, the temperature measurement can be calibrated in the application software. The software calibration requires that a calibration value be measured and stored in a register or EEPROM for each chip as a part of the production test. The software calibration can be done utilizing the formula:

```
T = {[(ADCH << 8) | ADCL] - TOS} / k
```

where ADCn are the ADC data registers, k is a fixed coefficient and T_{OS} is the temperature sensor offset value determined and stored into EEPROM as a part of the production test.

To obtain best accuracy the coefficient k should be measured using two temperature calibrations. Using offset calibration, set k = 1.0, where k = (1024 × 1.07mV/°C) /1.1V~1.0 [1/°C].

18.10 Register Description

18.10.1 ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
0x07 (0x27)	REFS1	REFS0	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – REFS1:REFS0: Reference Selection Bits**

These bits select the voltage reference for the ADC, as shown in Table 18-3. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSR is set).

Special care should be taken when changing differential channels. Once a differential channel has been selected, the stage may take as much as 25 ADC clock cycles to stabilize to the new value. Thus conversions should not be started within the first 13 clock cycles after selecting a new differential channel. Alternatively, conversion results obtained within this period should be discarded.

The same settling time should be observed for the first differential conversion after changing ADC reference (by changing the REFS1:0 bits in the ADMUX register).

For channels where differential gain is used (i.e., the gain stage), using VCC or an optional external AREF higher than (VCC – 1V) is not recommended, as this will affect ADC accuracy. The internal voltage reference may not be connected to the AREF pin if an external voltage is already being applied to it. The internal voltage reference is connected to the AREF pin when REFS1:0 is set to the value “11”.

Table 18-3. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	V_{CC} used as analog reference, disconnected from PA0 (AREF).
0	1	External voltage reference at PA0 (AREF) pin, internal voltage reference turned off.
1	0	Internal 1.1V voltage reference.
1	1	Reserved.

- **Bits 5:0 – MUX5:0: Analog Channel and Gain Selection Bits**

The value of these bits selects which combination of analog inputs are connected to the ADC. In the case of differential input, gain selection is also made with these bits. Selections on Table 18-4 on page 130 show values for single-ended channels and where the differential channels and the offset calibration selections are located. Selecting the single-ended channel ADC8 enables the temperature measurement. See Table 18-4 on page 130 ffor details. If these bits are changed during a conversion, the change will not go into effect until this conversion is complete (ADIF in ADCSRA is set).

20. Self-Programming the Flash

The device provides a self-programming mechanism (SPM) for downloading and uploading program code by the MCU itself. The self programming can use any available data interface and associated protocol to read code and write (program) that code into program memory.

Program memory is updated in a page-by-page fashion. Before programming a page with the data stored in the temporary page buffer, the page must be erased. The temporary page buffer is filled one word at a time using SPM, and the buffer can be filled either before the page erase command or between a page erase and a page write operation:

Alternative 1 - fill the buffer before a page erase

- Fill the temporary page buffer
- Perform a page erase
- Perform a page write

Alternative 2 - fill the buffer after a page erase

- Perform a page erase
- Fill the temporary page buffer
- Perform a page write

If only a part of the page needs to be changed, the rest of the page must be stored (for example in the temporary page buffer) before the erase, and then be re-written. When using alternative 1, the boot loader provides an effective read-modify-write feature, which allows the user software to first read the page and do the necessary changes, and then write back the modified data. If alternative 2 is used, it is not possible to read the old data while loading because the page is already erased. The temporary page buffer can be accessed in a random sequence. It is essential that the page address used in both the page erase and page write operation is addressing the same page.

20.1 Performing Page Erase by SPM

To execute a page erase, set up the address in the Z-pointer, write “00000011” to SPMCSR, and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 are ignored. The page address must be written to PCPAGE in the Z-register. Other bits in the Z-pointer will be ignored during this operation.

The CPU is halted during the page erase operation.

20.2 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 SPMCSR and execute SPM within four clock cycles after writing SPMCSR. 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 CTPB bit in SPMCSR. 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.

If the EEPROM is written in the middle of an SPM page load operation, all data loaded will be lost.

20.3 Performing a Page Write

To execute a page write, set up the address in the Z-pointer, write “00000101” to SPMCSR, and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 are ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

The CPU is halted during the page write operation.

20.4 Addressing the Flash During Self-Programming

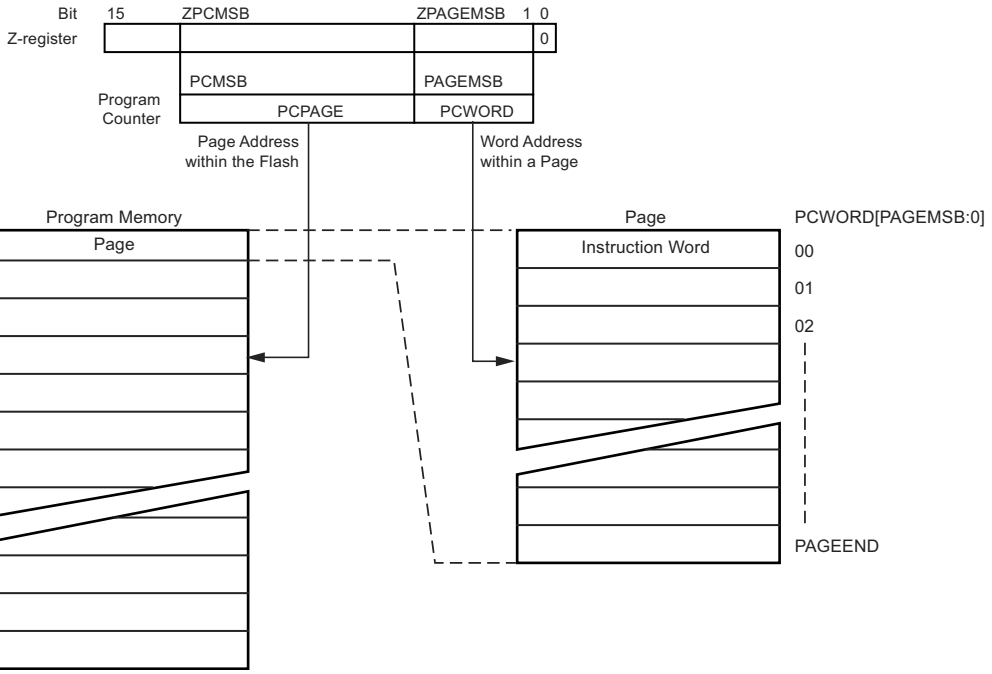
The Z-pointer is used to address the SPM commands.

Bit	15	14	13	12	11	10	9	8
ZH (R31)	**Z15**	**Z14**	**Z13**	**Z12**	**Z11**	**Z10**	**Z9**	**Z8**
ZL (R30)	**Z7**	**Z6**	**Z5**	**Z4**	**Z3**	**Z2**	**Z1**	**Z0**
	7	6	5	4	3	2	1	0

Because the flash is organized in pages (see Table 21-7 on page 144), the program counter can be treated as having two different sections. One section, consisting of the least significant bits, is addressing the words within a page, while the most significant bits are addressing the pages. This is shown in Figure 21-1 on page 144. Note that the page erase and page write operations are addressed independently. Therefore it is of major importance that the software addresses the same page in both the page erase and page write operation.

The LPM instruction uses the Z-pointer to store the address. Because this instruction addresses the flash byte-by-byte, the LSB (bit Z0) of the Z-pointer is also used.

Figure 20-1. Addressing the Flash During SPM(1)



Note: 1. The different variables used in Figure 20-1 are listed in Table 21-7 on page 144.

20.4.1 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 check the status bit (EEPE) in the EECR register and verify that the bit is cleared before writing to SPMCSR.

20.5 Register Description

20.5.1 SPMCSR – Store Program Memory Control and Status Register

The store program memory control and status register contains the control bits needed to control program memory operations.

Bit	7	6	5	4	3	2	1	0	
0x37 (0x57)	–	–	–	CTPB	**RFLB**	**PGWRT**	**PGERS**	**SPMEN**	**SPMCSR**
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..5 – Res: Reserved Bits**

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

- **Bit 4 – CTPB: Clear Temporary Page Buffer**

If the CTPB bit is written while filling the temporary page buffer, the temporary page buffer will be cleared and the data will be lost.

- **Bit 3 – RFLB: Read Fuse and Lock Bits**

An LPM instruction within three cycles after RFLB and SPMEN are set in SPMCSR will read either the lock bits or the fuse bits (depending on Z0 in the Z-pointer) in the destination register. See Section 20.4.1 “EEPROM Write Prevents Writing to SPMCSR” on page 138 for details.

- **Bit 2 – PGWRT: Page Write**

If this bit is written to one at the same time as SPMEN, the next SPM instruction within four clock cycles executes page write, with the data stored in the temporary buffer. The page address is taken from the high part of the Z-pointer.

The data in R1 and R0 are ignored. The PGWRT bit will auto-clear upon completion of a page write, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire page write operation.

- **Bit 1 – PGERS: Page Erase**

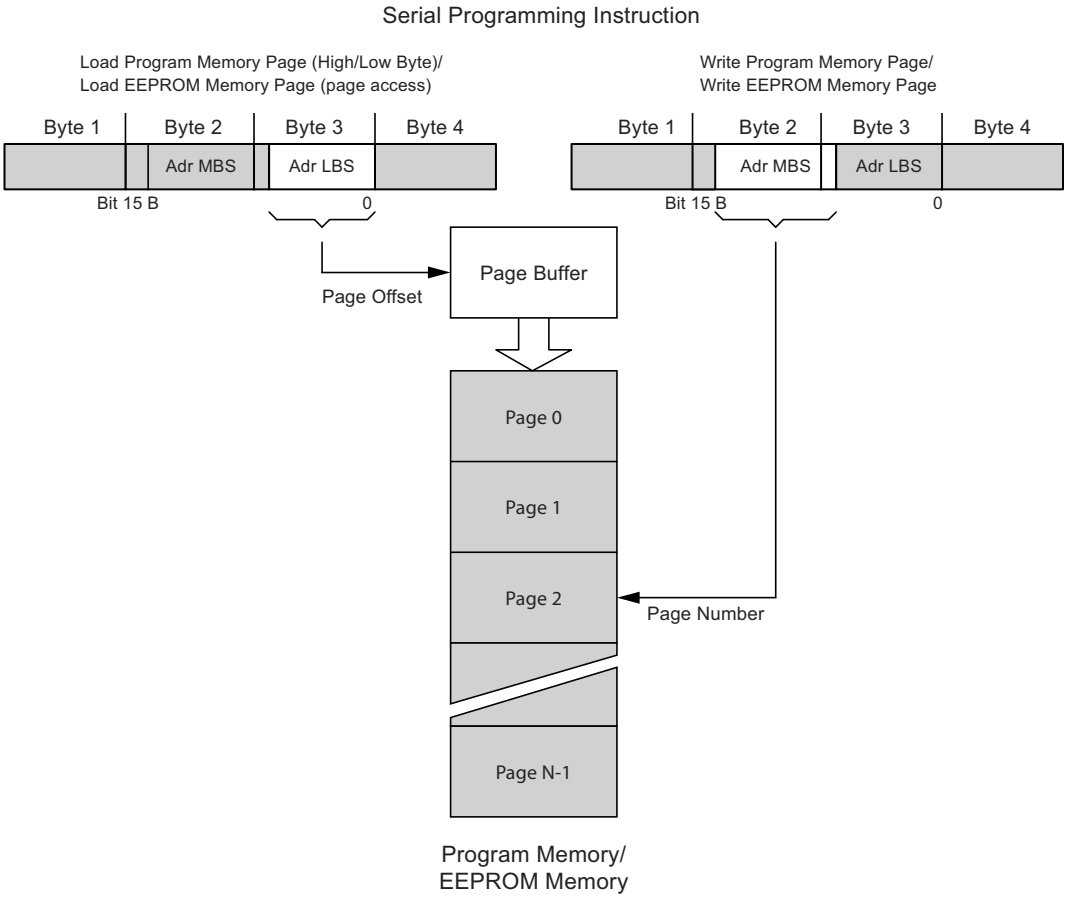
If this bit is written to logical one at the same time as SPMEN, the next SPM instruction within four clock cycles executes a page erase. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGERS bit will auto-clear upon completion of a page erase, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire page erase operation.

- **Bit 0 – SPMEN: Store Program Memory Enable**

This bit enables the SPM instruction for the next four clock cycles. If written to logical one together with either CTPB, RFLB, PGWRT, or PGERS, the following SPM instruction will have a special meaning (see description above). If only SPMEN is written, the following SPM instruction will store the value in R1:R0 in the temporary page buffer addressed by the Z-pointer. The LSB of the Z-pointer is ignored. The SPMEN bit will auto-clear upon completion of an SPM instruction, or if no SPM instruction is executed within four clock cycles. During page erase and page write, the SPMEN bit remains high until the operation is completed.

Writing any combination other than “10001”, “01001”, “00101”, “00011”, or “00001” in the lower five bits will have no effect.

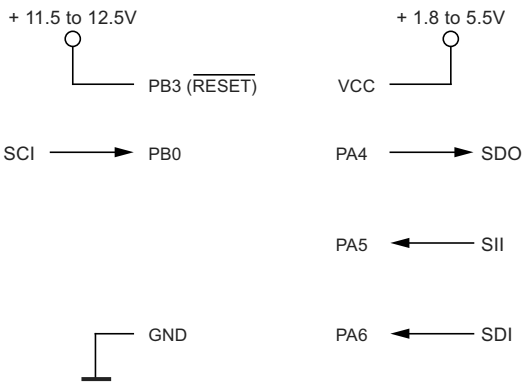
Figure 21-2. Serial Programming Instruction Example



21.7 High-voltage Serial Programming

This section describes how to program and verify flash program memory, EEPROM data memory, lock bits and fuse bits in the Atmel® ATtiny24/44/84.

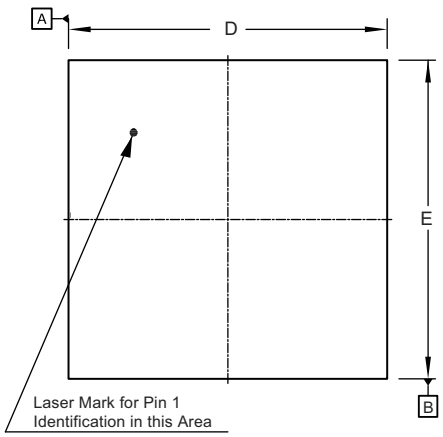
Figure 21-3. High-voltage Serial Programming



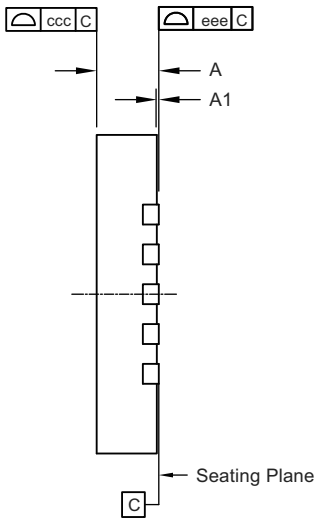
27. Packaging Information

27.1 PC

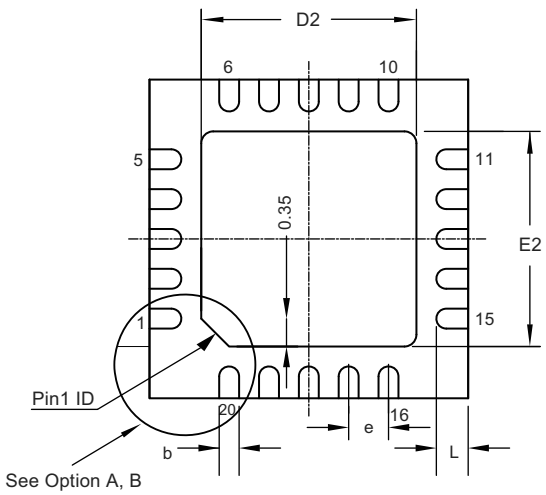
DRAWINGS NOT SCALED



Top View



Side View



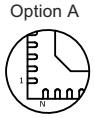
Bottom View

COMMON DIMENSIONS

(Unit of Measure = mm)

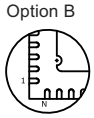
SYMBOL	MIN	NOM	MAX	NOTE
A	0.70	0.75	0.80	
A1	0.00	0.02	0.05	
D/E	3.90	4.00	4.10	
D2/E2	2.50	2.60	2.70	
L	0.35	0.45	0.55	
b	0.20	0.25	0.30	
e	0.50 BSC			
Tolerances of Form and Position				
ccc	0.10			
eee	0.08			
n	20			

Option A



Pin 1# Chamfer
(C 0.35 max)

Option B



Pin 1# Notch
(0.20 R)

08/01/14

	TITLE	GPC	DRAWING NO.	REV.
Atmel **Package Drawing Contact:** packagedrawings@atmel.com	PC, 20-Lds - 0.50mm Pitch, 4x4x0.8mm Body size Very very Thin Quad Flat No Lead Package (WQFN) Sawn	ZVF	PC	J