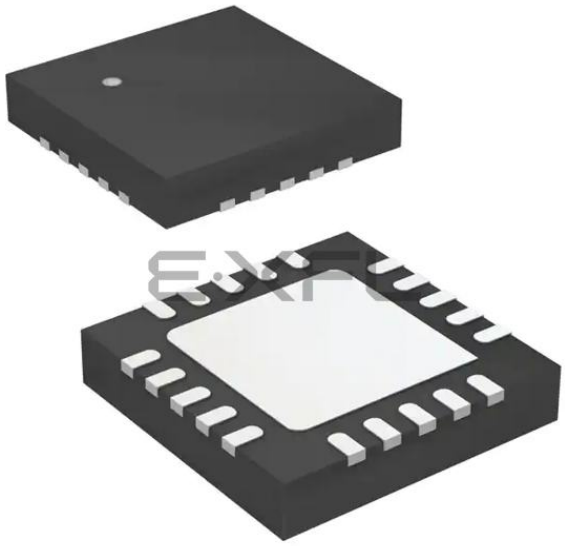


Welcome to E-XFL.COM

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	20-WFQFN Exposed Pad
Supplier Device Package	20-WQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny44-15mz

- Operating voltage:
  - 2.7 - 5.5V for Atmel ATtiny24/44/84
- Speed grade
  - Atmel ATtiny24/44/84: 0 - 8MHz at 2.7 - 5.5V, 0 - 16MHz at 4.5 - 5.5V
- Automotive temperature range
- Low power consumption
  - Active mode:
    - 1MHz, 2.7V: 800µA
  - Power-down mode:
    - 2.7V: 2.0µA

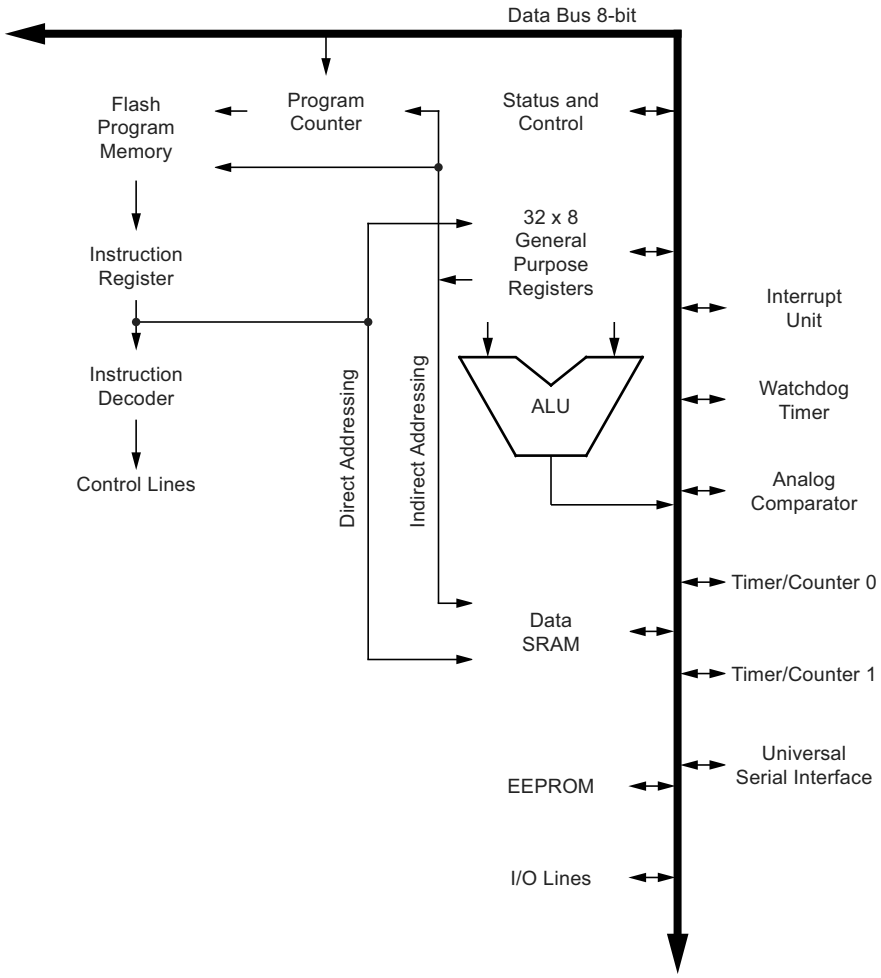
5. CPU Core

5.1 Overview

This section discusses the Atmel® AVR® core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must, therefore, be able to access memories, perform calculations, control peripherals, and handle interrupts.

5.2 Architectural Overview

Figure 5-1. Block Diagram of the AVR Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture, with separate memories and buses for program and data. Instructions in the program memory are executed with a single-level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is in-system reprogrammable flash memory.

6. Memories

This section describes the different memories in the Atmel® ATtiny24/44/84. The AVR® architecture has two main memory spaces, the data memory space and the program memory space. In addition, the Atmel ATtiny24/44/84 features an EEPROM memory for data storage. All three memory spaces are linear and regular.

6.1 In-System Re-programmable Flash Program Memory

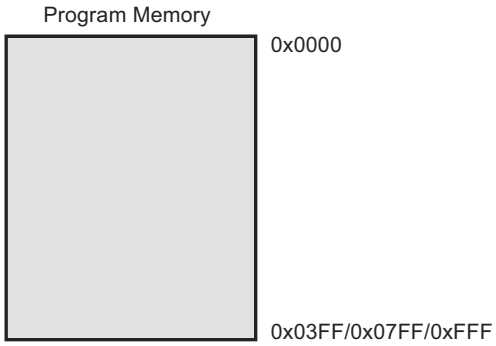
The Atmel ATtiny24/44/84 contains 2/4/8K bytes of on-chip, in-system, reprogrammable flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the flash memory is organized as 1024/2048/4096 x 16.

The flash memory has an endurance of at least 10,000 write/erase cycles. The Atmel ATtiny24/44/84 program counter (PC) is 10/11/12 bits wide, thus addressing the 1024/2048/4096 program memory locations. Section 21. “Memory Programming” on page 141 contains a detailed description on flash data serial downloading using the SPI pins.

Constant tables can be allocated within the entire program memory address space (see the LPM – load program memory instruction description).

Timing diagrams for instruction fetch and execution are presented in Section 5.7 “Instruction Execution Timing” on page 14.

Figure 6-1. Program Memory Map



6.2 SRAM Data Memory

Figure 6-2 on page 17 shows how the ATtiny24/44/84 SRAM memory is organized.

The lower 160 data memory locations address the register file, the I/O memory, and the internal data SRAM. The first 32 locations address the register file, the next 64 locations the standard I/O memory, and the last 128/256/512 locations address the internal data SRAM.

The five different addressing modes for the data memory are: direct, indirect with displacement, indirect, indirect with pre-decrement, and indirect with post-increment. In the register file, registers R26 to R31 feature the indirect addressing pointer registers.

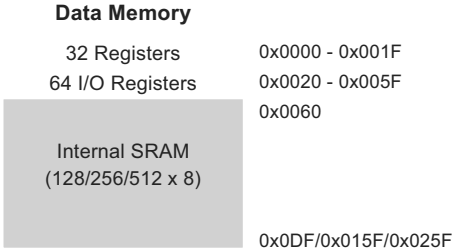
The direct addressing mode reaches the entire data space.

The indirect with displacement mode reaches 63 address locations starting from the base address given by the Y- or Z-register.

When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y, and Z are decremented or incremented.

The 32 general purpose working registers, 64 I/O registers, and 128/256/512 bytes of internal data SRAM in the Atmel ATtiny24/44/84 are all accessible through all these addressing modes. The register file is described in Section 5.5 “General Purpose Register File” on page 12.

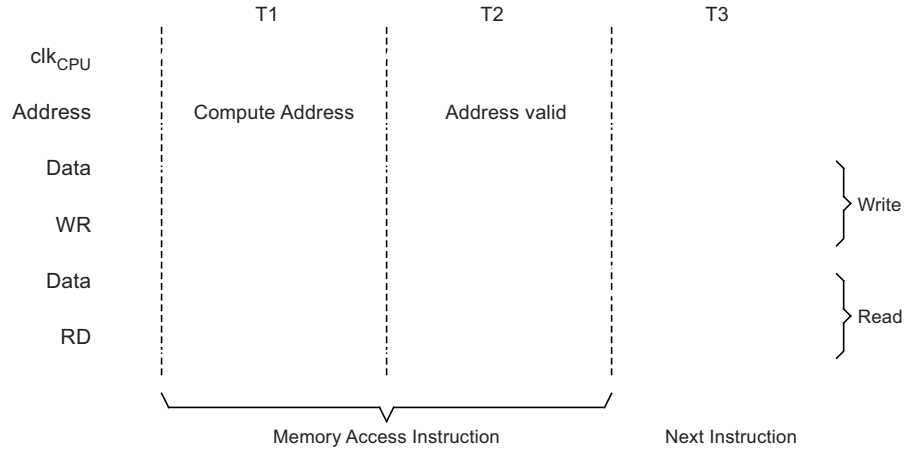
Figure 6-2. Data Memory Map



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.

If the program never enables an interrupt source, the interrupt vectors are not used, and regular program code can be placed at these locations. The most typical and general program setup for the reset and interrupt vector Addresses in Atmel® ATtiny24/44/84 is:

```
Address      Labels Code                Comments
0x0000              rjmp   RESET        ; Reset Handler
0x0001              rjmp   EXT_INT0     ; IRQ0 Handler
0x0002              rjmp   PCINT0       ; PCINT0 Handler
0x0003              rjmp   PCINT1       ; PCINT1 Handler
0x0004              rjmp   WATCHDOG     ; Watchdog Interrupt Handler
0x0005              rjmp   TIM1_CAPT    ; Timer1 Capture Handler
0x0006              rjmp   TIM1_COMPA   ; Timer1 Compare A Handler
0x0007              rjmp   TIM1_COMPB   ; Timer1 Compare B Handler
0x0008              rjmp   TIM1_OVF     ; Timer1 Overflow Handler
0x0009              rjmp   TIM0_COMPA   ; Timer0 Compare A Handler
0x000A              rjmp   TIM0_COMPB   ; Timer0 Compare B Handler
0x000B              rjmp   TIM0_OVF     ; Timer0 Overflow Handler
0x000C              rjmp   ANA_COMP     ; Analog Comparator Handler
0x000D              rjmp   ADC          ; ADC Conversion Handler
0x000E              rjmp   EE_RDY       ; EEPROM Ready Handler
0x000F              rjmp   USI_STR      ; USI STart Handler
0x0010              rjmp   USI_OVF      ; USI Overflow Handler
;
0x0011       RESET: ldi    r16, high(RAMEND); Main program start
0x0012              out    SPH,r16      ; Set Stack Pointer to top of RAM
0x0013              ldi    r16, low(RAMEND)
0x0014              out    SPL,r16
0x0015              sei                 ; Enable interrupts
0x0016              <instr> xxx
...    ...    ...    ...
```

Table 12-6. Overriding Signals for Alternate Functions in PA1..PA0

Signal Name	PA1/ADC1/AIN0/PCINT1	PA0/ADC0/AREF/PCINT0
PUOE	0	RESET × (REFS1 × REFS0 + REFS1 × REFS0)
PUOV	0	0
DDOE	0	RESET × ($\overline{\text{REFS1}}$ × REFS0 + REFS1 × REFS0)
DDOV	0	0
PVOE	0	RESET × ($\overline{\text{REFS1}}$ × REFS0 + REFS1 × REFS0)
PVOV	0	0
PTOE	0	0
DIEOE	PCINT1 × PCIE0 + ADC1D	PCINT0 × PCIE0 + ADC0D
DIEOV	PCINT1 × PCIE0	PCINT0 × PCIE0
DI	PCINT1 input	PCINT0 input
AIO	ADC1/analog comparator positive input	ADC1 input analog reference

12.3.2 Alternate Functions of Port B

The port B pins with alternate function are shown in Table 12-7.

Table 12-7. Port B Pins Alternate Functions

Port Pin	Alternate Function	
PB0	XTAL1:	Crystal oscillator input.
	PCINT8:	Pin change interrupt 1 source 8.
PB1	XTAL2:	Crystal oscillator output.
	PCINT9:	Pin change interrupt 1 source 9.
PB2	INT0:	External interrupt 0 input.
	OC0A:	Timer/Counter0 compare match A output.
	CKOUT:	System clock output.
	PCINT10:	Pin change interrupt 1 source 10.
PB3	RESET:	Reset pin.
	dW:	debugWire I/O.
	PCINT11:	Pin change interrupt 1 source 11.

- **Port B, Bit 0 – XTAL1/PCINT8**

XTAL1: Chip clock oscillator pin 1. Used for all chip clock sources except the internal calibrated RC oscillator. When used as a clock pin, the pin cannot be used as an I/O pin. When using internal calibrated RC oscillator as a chip clock source, PB0 serves as an ordinary I/O pin.

PCINT8: Pin change interrupt source 8. The PB0 pin can serve as an external interrupt source for pin change interrupt 1.

- **Port B, Bit 1 – XTAL2/PCINT9**

XTAL2: Chip clock oscillator pin 2. Used as clock pin for all chip clock sources except the internal calibrated RC oscillator and external clock. When used as a clock pin, the pin cannot be used as an I/O pin. When using internal calibrated RC oscillator or external clock as a chip clock sources, PB1 serves as an ordinary I/O pin.

PCINT9: Pin change interrupt source 9. The PB1 pin can serve as an external interrupt source for pin change interrupt 1.

13. 8-bit Timer/Counter0 with PWM

13.1 Features

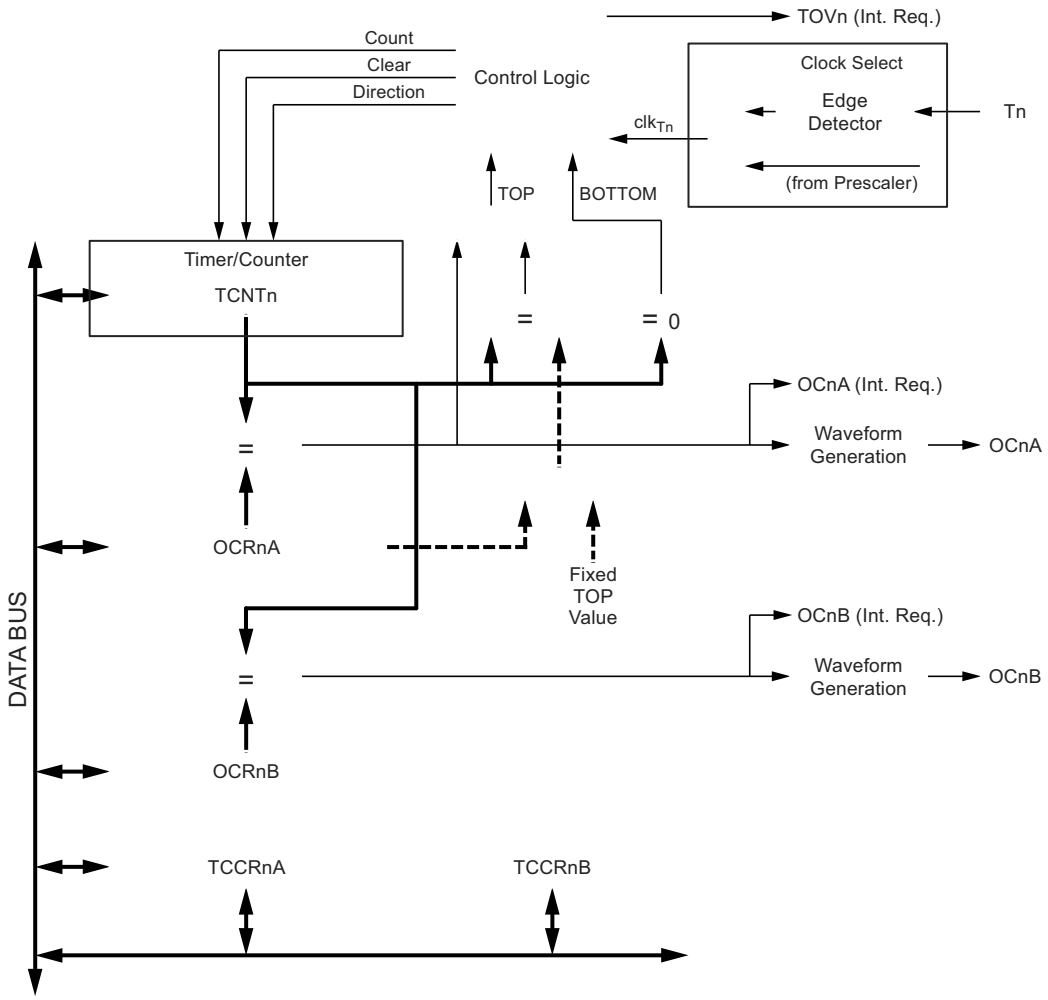
- Two independent output compare units
- Double buffered output compare registers
- Clear timer on compare match (auto reload)
- Glitch free, phase correct pulse width modulator (PWM)
- Variable PWM period
- Frequency generator
- Three independent interrupt sources (TOV0, OCF0A, and OCF0B)

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



13.2.1 Registers

The Timer/Counter (TCNT0) and output compare registers (OCR0A and OCR0B) are 8-bit registers. Interrupt request (abbreviated to Int.Req. in the figure) signals are all visible in the Timer/Counter 0 interrupt flag register (TIFR0). All interrupts are individually masked with the timer interrupt mask register (TIMSK0). TIFR0 and TIMSK0 are not shown in the figure.

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

The double buffered output compare registers (OCR0A and OCR0B) is compared with the Timer/Counter value at all times. The result of the compare can be used by the waveform generator to generate a PWM or variable frequency output on the output compare pins (OC0A and OC0B). See Section 13.5 “Output Compare Unit” on page 65 for details. The compare match event will also set the compare flag (OCF0A or OCF0B) which can be used to generate an output compare interrupt request.

13.2.2 Definitions

Many register and bit references in this section are written in general form. A lower case “n” replaces the Timer/Counter number, in this case 0. A lower case “x” replaces the output compare unit, in this case compare unit A or compare unit B. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT0 for accessing Timer/Counter 0 counter value and so on.

The definitions in Table are also used extensively throughout the document.

Table 13-1. Definitions

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

13.3 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the clock select logic which is controlled by the clock select (CS02:0) bits located in the Timer/Counter control register (TCCR0B). For details on clock sources and prescaler, see Section 15. “Timer/Counter Prescaler” on page 103.

13.4 Counter Unit

The main part of the 8-bit Timer/Counter is the programmable bi-directional counter unit. Figure 13-2 shows a block diagram of the counter and its surroundings.

Figure 13-2. Counter Unit Block Diagram

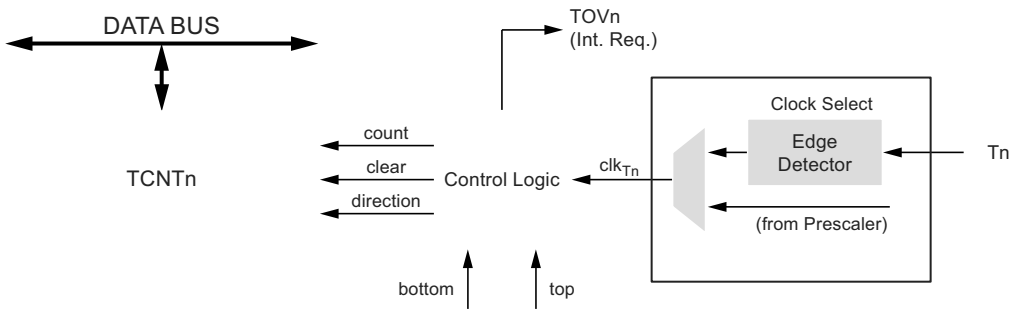
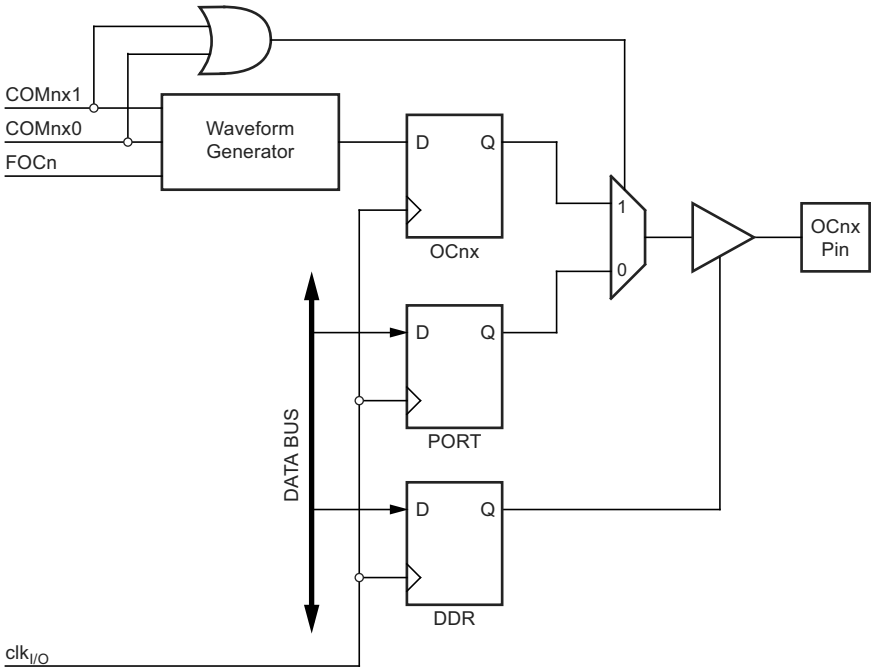


Figure 13-4. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the output compare (OC0x) from the waveform generator if either of the COM0x1:0 bits are set. However, the OC0x pin direction (input or output) is still controlled by the data direction register (DDR) for the port pin. The data direction register bit for the OC0x pin (DDR_OC0x) must be set as output before the OC0x value is visible on the pin. The port override function is independent of the waveform generation mode.

The design of the output compare pin logic allows initialization of the OC0x state before the output is enabled. Note that some COM0x1:0 bit settings are reserved for certain modes of operation, see
Section 13.9 “Register Description” on page 73.

13.6.1 Compare Output Mode and Waveform Generation

The waveform generator uses the COM0x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM0x1:0 = 0 tells the waveform generator that no action on the OC0x register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 13-2 on page 73. For fast PWM mode, refer to Table 13-3 on page 73, and for phase correct PWM refer to Table 13-4 on page 73.

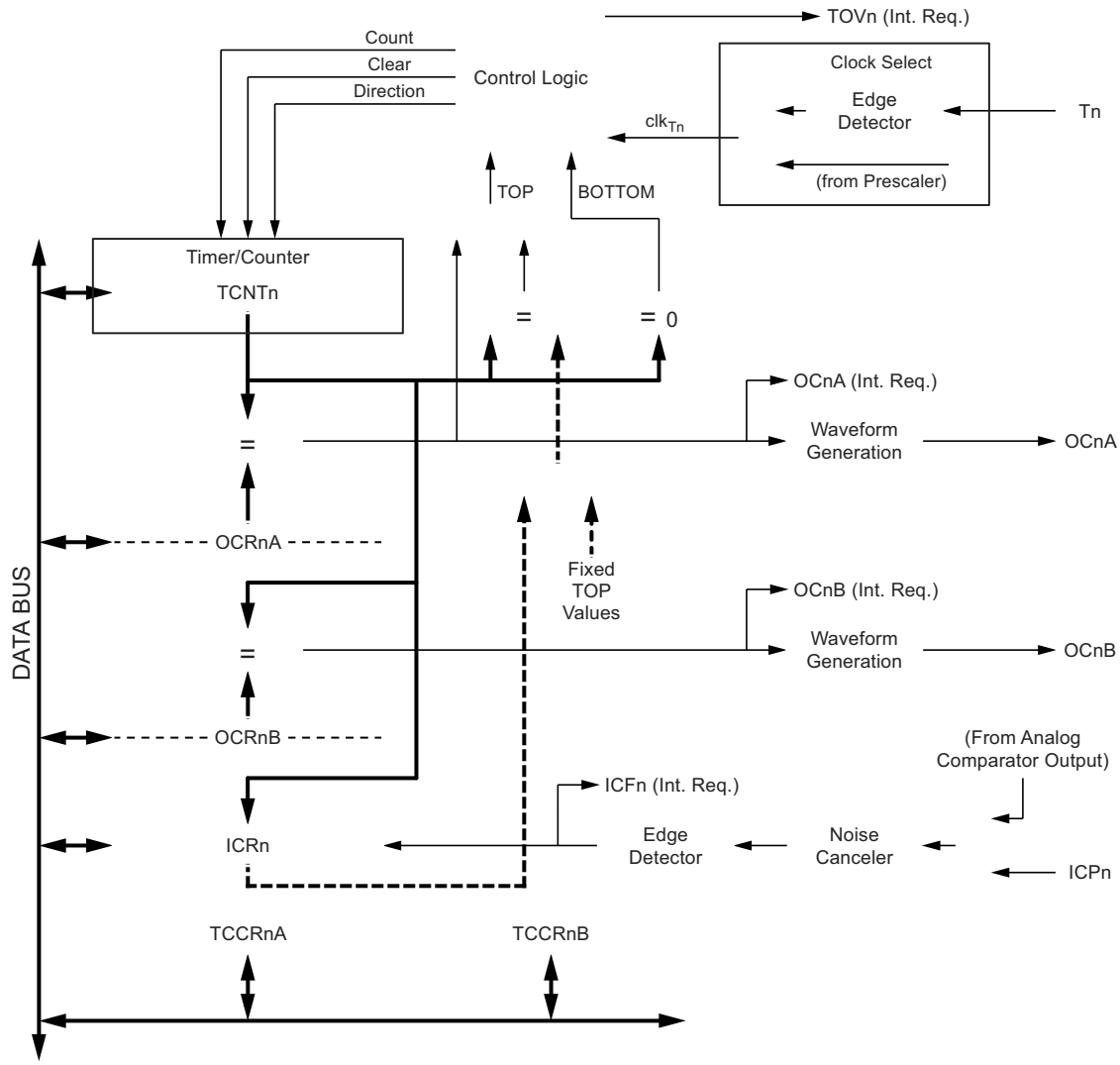
A change of the COM0x1:0 bit states will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the 0x strobe bits.

13.7 Modes of Operation

The mode of operation, i.e., the behavior of the Timer/Counter and the output compare pins, is defined by the combination of the waveform generation mode (WGM02:0) and compare output mode (COM0x1:0) bits. The compare output mode bits do not affect the counting sequence, while the waveform generation mode bits do. The COM0x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM0x1:0 bits control whether the output should be set, cleared, or toggled at a compare match (See Section 13.7 “Modes of Operation” on page 67).

For detailed timing information refer to Figure 13-8 on page 71, Figure 13-9 on page 72, Figure 13-10 on page 72 and Figure 13-11 on page 72 in Section 13.8 “Timer/Counter Timing Diagrams” on page 71.

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.

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	

16.3.4 Two-wire Mode

The USI two-wire mode is compliant with the inter-IC (I2C or TWI) bus protocol, but without slew rate limiting on outputs and input noise filtering. Pin names used by this mode are SCL and SDA.

Figure 16-4. Two-wire Mode Operation, Simplified Diagram

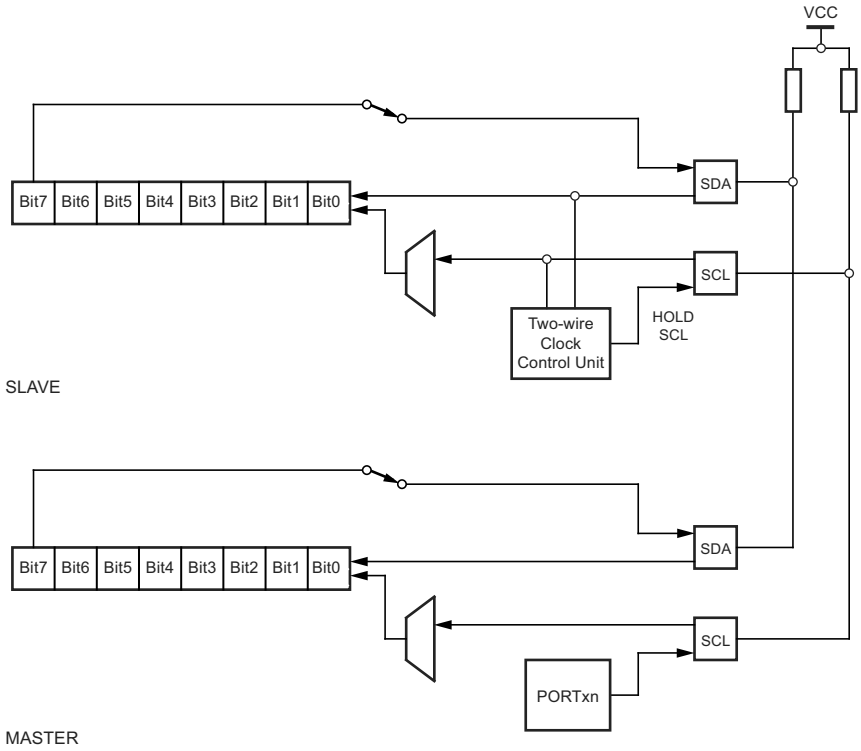
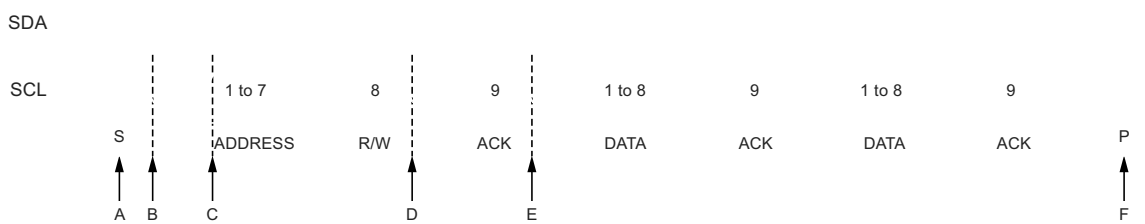


Figure 16-4 shows two USI units operating in two-wire mode, one as master and one as slave. Only the physical layer is shown because the system operation is highly dependent of the communication scheme used. The main differences between the master and slave operation at this level are that the serial clock generation is always done by the master, and only the slave uses the clock control unit. Clock generation must be implemented in software, but the shift operation is done automatically by both devices. Note that only clocking on the negative edge to shift data is practical in this mode. The slave can insert wait states at the start or end of a transfer by forcing the SCL clock low. This means that the master must always check if the SCL line was actually released after it has generated a positive edge.

Because the clock also increments the counter, a counter overflow can be used to indicate that the transfer has completed. The master generates clock by the by toggling the USCK pin via the PORT register.

The data direction is not given by the physical layer. A protocol, like the one used by the TWI-bus, must be implemented to control the data flow.

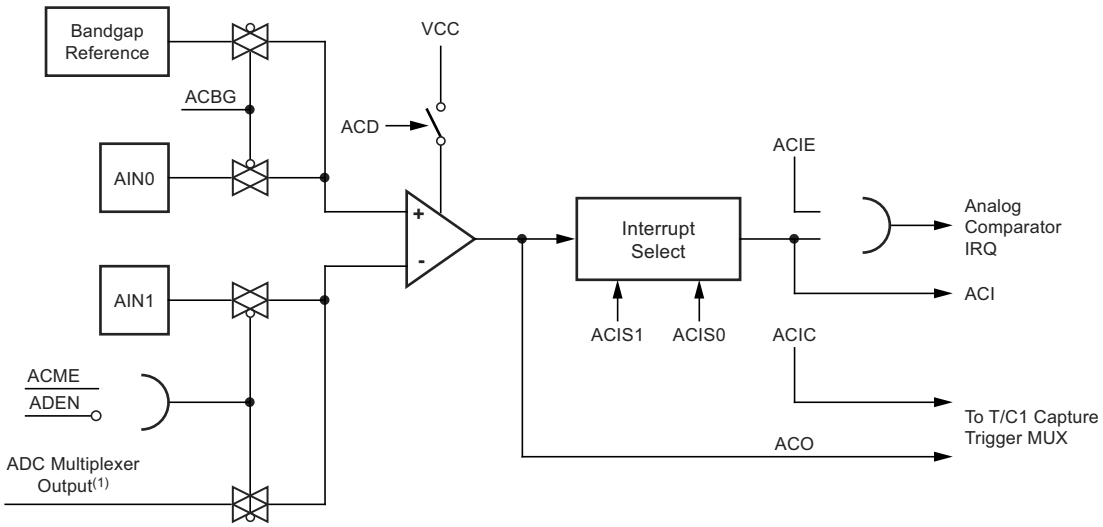
Figure 16-5. Two-wire Mode, Typical Timing Diagram



17. Analog Comparator

The analog comparator compares the input values on the positive pin (AIN0) and negative pin (AIN1). When the voltage on the positive pin (AIN0) is higher than the voltage on the negative pin (AIN1), the analog comparator output (ACO) is set. The comparator can trigger a separate interrupt, exclusive to the analog comparator. The user can select Interrupt triggering on comparator output rise, fall or toggle. A block diagram of the comparator and its surrounding logic is shown in Figure 17-1 on page 115.

Figure 17-1. Analog Comparator Block Diagram(1)



Notes: 1. See Table 17-1.

2. See Figure 1-1 on page 3 and Table 12-9 on page 61 for analog comparator pin placement.

17.1 Analog Comparator Multiplexed Input

When the analog-to-digital converter (ADC) is configured as a single-ended input channel, it is possible to select any of the ADC7..0 pins to replace the negative input to the analog comparator. The ADC multiplexer is used to select this input, and, consequently, the ADC must be switched off to utilize this feature. If the analog comparator multiplexer enable bit (ACME in ADCSRB) is set and the ADC is switched off (ADEN in ADCSRA is zero), MUX1..0 in ADMUX select the input pin to replace the negative input to the analog comparator, as shown in Table 17-1. If ACME is cleared or ADEN is set, AIN1 is applied to the negative input of the analog comparator.

Table 17-1. Analog Comparator Multiplexed Input

ACME	ADEN	MUX4..0	Analog Comparator Negative Input
0	x	xx	AIN1
1	1	xx	AIN1
1	0	00000	ADC0
1	0	00001	ADC1
1	0	00010	ADC2
1	0	00011	ADC3
1	0	00100	ADC4
1	0	00101	ADC5
1	0	00110	ADC6
1	0	00111	ADC7

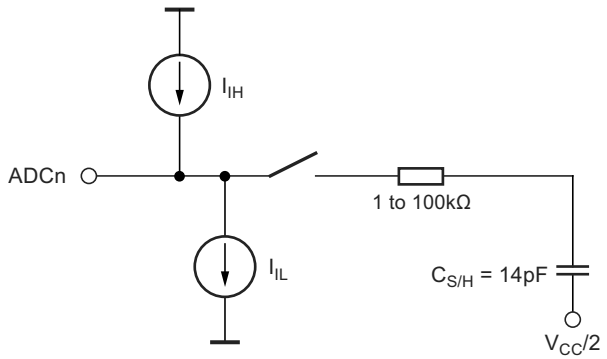
18.7.1 Analog Input Circuitry

The analog input circuitry for single-ended channels is illustrated in Figure 18-8 on page 125. An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC or not. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long the source needs to charge the S/H capacitor, with can vary widely. The user is recommended to only use low impedant sources with slowly varying signals, since this minimizes the required charge transfer to the S/H capacitor.

Signal components higher than the nyquist frequency ($f_{ADC/2}$) should not be present to avoid distortion from unpredictable signal convolution. The user is advised to remove high-frequency components with a low-pass filter before applying the signals as inputs to the ADC.

Figure 18-8. Analog Input Circuitry



18.7.2 Analog Noise Canceling Techniques

Digital circuitry inside and outside the device generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

a. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.

b. Use the ADC noise canceller function to reduce induced noise from the CPU.

c. If any port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress.

21.8.3 Chip Erase

The chip erase will erase the flash and EEPROM[1] memories, as well as lock bits. The lock bits are not reset until the program memory has been completely erased. The fuse bits are not changed. A chip erase must be performed before the flash and/or EEPROM are reprogrammed.

Note: 1. The EEPROM memory is preserved during chip erase if the EESAVE fuse is programmed.

1. Load “chip erase” command (see Table 21-15 on page 152).
2. Wait after Instr. 3 until SDO goes high for the “chip erase” cycle to finish.
3. Load “no operation” command.

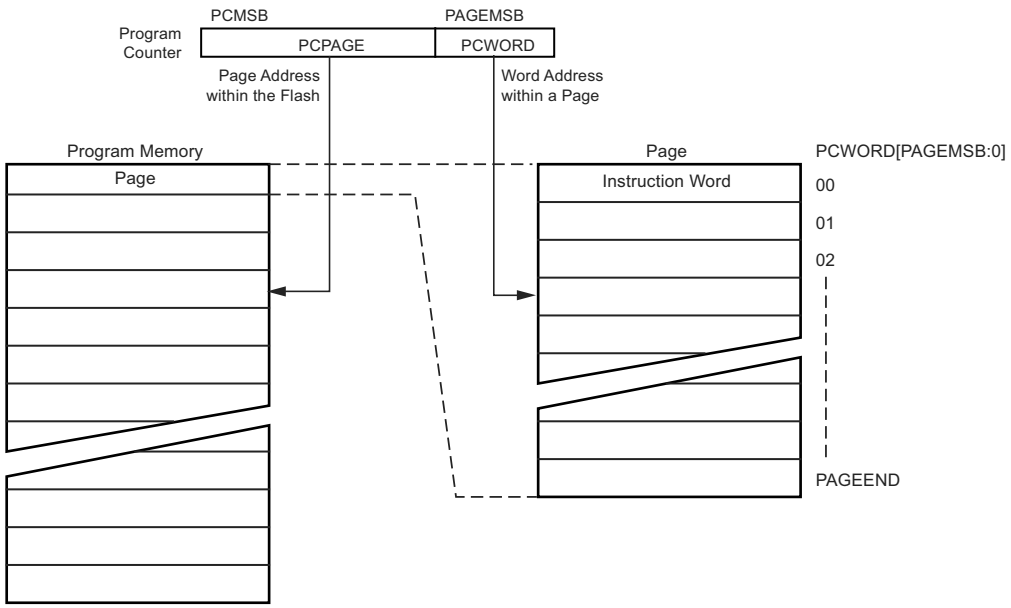
21.8.4 Programming the Flash

The flash is organized in pages, see Section 21.5 “Page Size” on page 144. When programming the flash, the program data are latched into a page buffer. This allows one page of program data to be programmed simultaneously. The following procedure describes how to program the entire flash memory:

1. Load “write flash” command (see Table 21-15 on page 152).
2. Load flash page buffer.
3. Load flash high address and program page. Wait after Instr. 3 until SDO goes high for the “page programming” cycle to finish.
4. Repeat 2 and 3 until the entire flash is programmed, or until all data has been programmed.
5. End page programming by loading “no operation” command.

When writing or reading serial data to the Atmel® ATtiny24/44/84, data are clocked on the rising edge of the serial clock. See Figure 22-5 on page 162, Figure 21-3 on page 148 and Table 22-9 on page 162 for details.

Figure 21-4. Addressing the Flash which is Organized in Pages



22.7 High-voltage Serial Programming Characteristics

Figure 22-5. High-voltage Serial Programming Timing

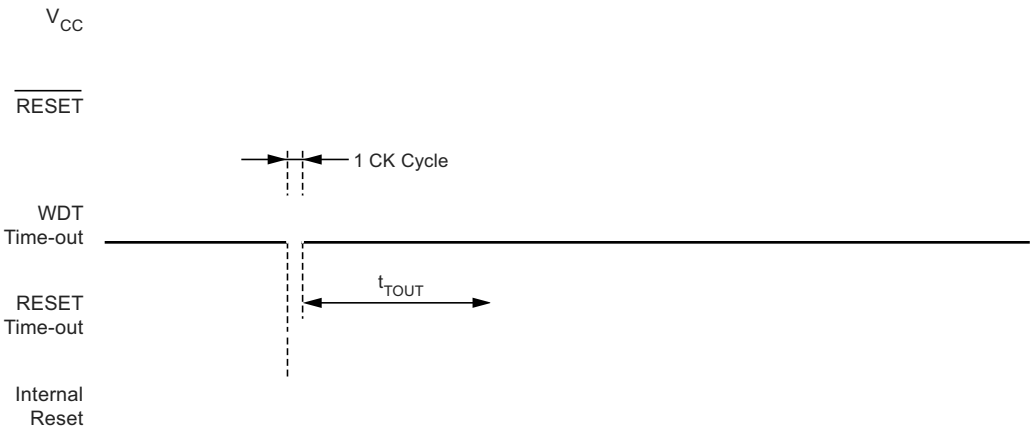


Table 22-9. High-voltage Serial Programming Characteristics
T_A = 25°C ±10%, V_{CC} = 5.0V ±10% (Unless otherwise noted)

Parameter	Symbol	Min	Typ	Max	Units
SCI (PB0) pulse width high	t_{SHSL}	110			ns
SCI (PB0) pulse width low	t_{SLSH}	110			ns
SDI (PA6), SII (PB1) valid to SCI (PB0) high	t_{IVSH}	50			ns
SDI (PA6), SII (PB1) hold after SCI (PB0) high	t_{SHIX}	50			ns
SCI (PB0) high to SDO (PA4) valid	t_{SHOV}		16		ns
Wait after Instr. 3 for write fuse bits	t_{WLWH_PFB}		2.5		ms

Figure 23-27. Reset Input Threshold Voltage versus V_{CC} (V_{IL}, IO pin Read as '0')

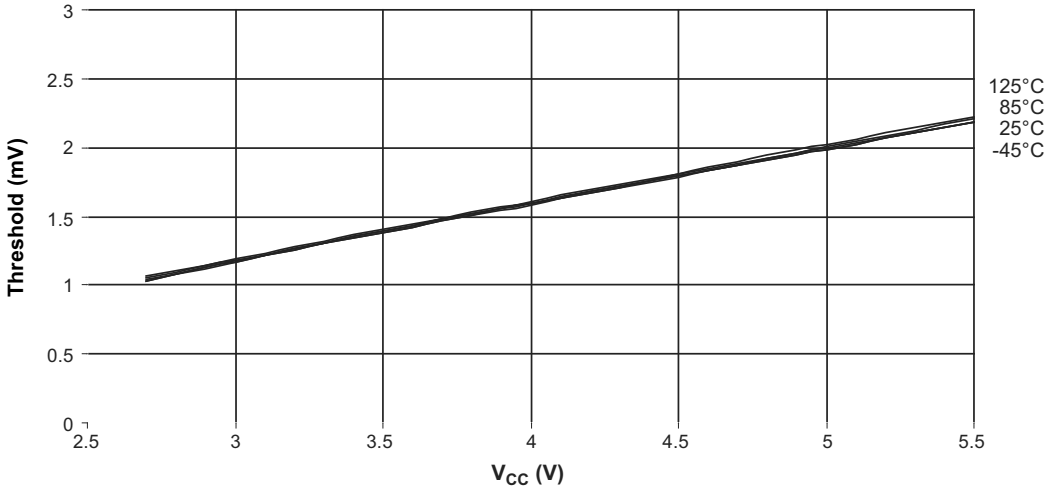
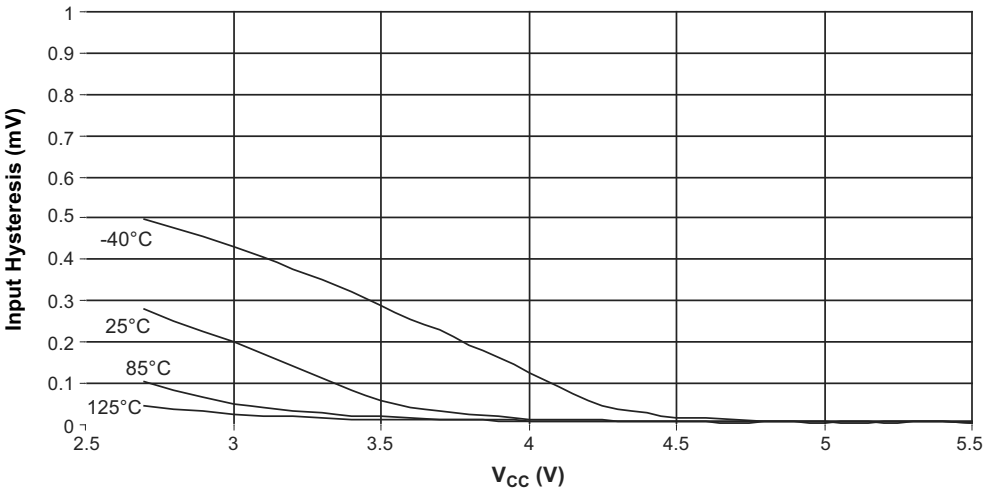


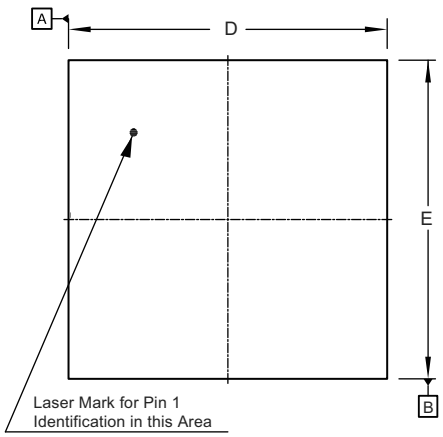
Figure 23-28. Reset Pin Input Hysteresis versus V_{CC}



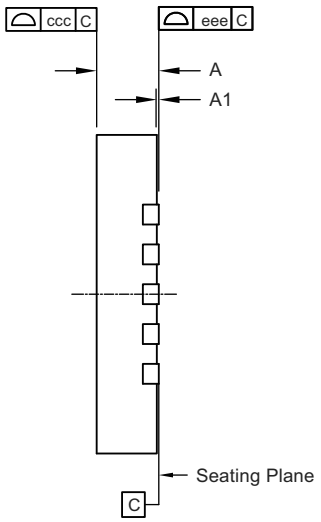
27. Packaging Information

27.1 PC

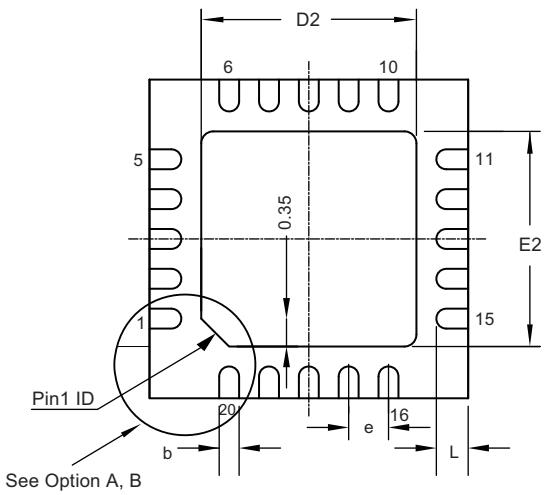
DRAWINGS NOT SCALED



Top View



Side View



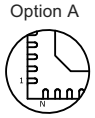
Bottom View

COMMON DIMENSIONS

(Unit of Measure = mm)

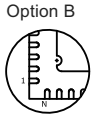
SYMBOL	MIN	NOM	MAX	NOTE
A	0.70	0.75	0.80	
A1	0.00	0.02	0.05	
D/E	3.90	4.00	4.10	
D2/E2	2.50	2.60	2.70	
L	0.35	0.45	0.55	
b	0.20	0.25	0.30	
e	0.50 BSC			
Tolerances of Form and Position				
ccc	0.10			
eee	0.08			
n	20			

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